Implementation of Deep Deterministic Policy Gradients for Controlling Dynamic Bipedal Walking

Chujun Liu(✉), Andrew G. Lonsberry(✉), Mark J. Nandor(✉),
Musa L. Audu(✉), and Roger D. Quinn(✉)

Department of Mechanical and Aerospace Engineering, Case Western Reserve
University, 10900 Euclid Ave., Cleveland, OH 44106, USA
{cxl936,agl10,mjn18,mxa93,roger.quinn}@case.edu
http://biorobots.case.edu/

Abstract. A control system for simulated two-dimensional bipedal walking was developed. The biped model was built based on anthropometric data. At the core of the control is a Deep Deterministic Policy Gradients (DDPG) neural network that is trained in GAZEBO, a physics simulator, to predict the ideal foot location to maintain stable walking under external impulse load. Additional controllers for hip joint movement during stance phase, and ankle joint torque during toe-off, help to stabilize the robot during walking. The simulated robot can walk at a steady pace of approximately 1 m/s, and during locomotion it can maintain stability with a 30 N-s impulse applied at the torso. This work implement DDPG algorithm to solve biped walking control problem. The complexity of DDPG network is decreased through carefully selected state variables and distributed control system.

Keywords: Biped · DDPG neural network · Gait

1 Introduction

A robust control algorithm for biped locomotion is presented as a means to assist individuals with spinal cord injury (SCI). Using Functional Neuro-muscular Stimulation (FNS) and a powered lower limb exoskeleton, locomotion can be restored to such individuals [1,2]. The methods presented are designed to apply control to the powered lower limb exoskeleton. To make the system robust for any user, the control approach must be able to adapt to various sizes of humans [7]. It should thus function with limited information about the human. To accomplish this, exploratory reinforcement based optimization algorithms such as Deep Q-Networks (DQN) can be applied. As biped control is continuous, a variation of DQN called deep deterministic policy gradients (DDPG) [8] will be utilized. In total, three separate controllers are designed to operate together to produce stable walking control. The use of three separate controllers actually reduces the

V. Vouloutsi et al. (Eds.): Living Machines 2018, LNAI 10928, pp. 276–287, 2018.
https://doi.org/10.1007/978-3-319-95972-6_29

complexity of the control system, making the DDPG network easier to train. As degrees of freedom increase, neural networks can have certain issues such as covariate shift [5] and increased training time. Since the application at hand is time sensitive, the speed of learning is crucial [3]. Furthermore, using three separate controllers allows for easy parallelization of the processes and dedicated threading.

For this work, one of the three controllers is a trained DDPG network and the other two are conventional PID feedback controller and an open loop controller.

2 Methods

A DDPG network is trained to work in conjunction with two other PID feedback controllers. DDPG is a model free policy learning algorithm. It consists of an actor network that updates the policy parameters, and a critic network that estimates the action-value function. DDPG uses the expected gradient of the action-value function as a policy gradient instead of a stochastic policy gradient so as to estimate the correct gradient much more efficiently. [8] Below we introduce the model, the simulation environment, the target locomotion, and the controllers.

2.1 Biped Model

A biped model, based partially on anthropometric data, is used in simulation to both train and verify the effectiveness of the DDPG network. The model contains 7 rigid bodies: the torso as well as the left and right thigh, shank, and foot. Additionally, the model has the following 6 joints: left and right hip, knees, and ankles. The hip and ankle joints can rotate along both the x and y axises. Two frictionless walls are added in the simulation environment to constrain biped in two-dimension, so x axis rotation of the ankle is the major. There is a small gap between the biped and the wall which can cause the biped to slight tilt sideways. This gap is left intentionally, because this will reduce the impact generate by the imperfection collision model in ODE(open dynamic engine). So y axis rotation of the ankle is kept because so the foot can have a solid contact with ground when biped is tilting sideways. The knees are constrained to just the x axis, giving the system a total of 10 degrees of freedom. The proportion of mass and length of the biped's bodies are found from anthropometric data, while the shape and the rotary inertia of the bodies are simplified to a regular box shape to speed simulations. All the components are proportional to the height, thus making resizing of the simulated biped easier. In this work, the height is set at 1.8 m. A simulated IMU sensor is attached to the center of the torso to measure its velocity and acceleration. This replicates what might be implemented on a powered exoskeleton. Touch sensors are added on both the left and right feet to detect ground contact and contact force. All joint angles and joint velocities can be directly read from the simulation environment.

2.2 Simulation Environment

The biped is simulated using GAZEBO and controlled by ROS (Robot Operating System). GAZEBO is an open source simulator, while ROS is a set of software libraries and convenient tools used for robotic systems. ROS has become a popular platform for robotics research [4]. Joint movement is controlled in the simulation in two ways. Firstly, we can call an "ApplyJointEffort" ROS service directly to set a torque value for some duration. Secondly, we use GAZEBO's controller plug-in. The controller plug-in provides three different PID control methods: torque feedback, velocity feedback and position feedback. In this work, the plug-in's velocity feedback control is utilized. The PID parameters are tuned to react in a fast and stable manner. For the work here we focus on constrained locomotion. The 10 DOF biped is restricted by two frictionless walls to prevent any lateral movement, constraining the model to only move in sagittal plane.

2.3 Target Locomotion

Human gait is a complex process. [6,11] The target gait is simplified into 4 sections for each leg: early swing, terminal swing, stance, and toe-off as depicted in Fig. 1.

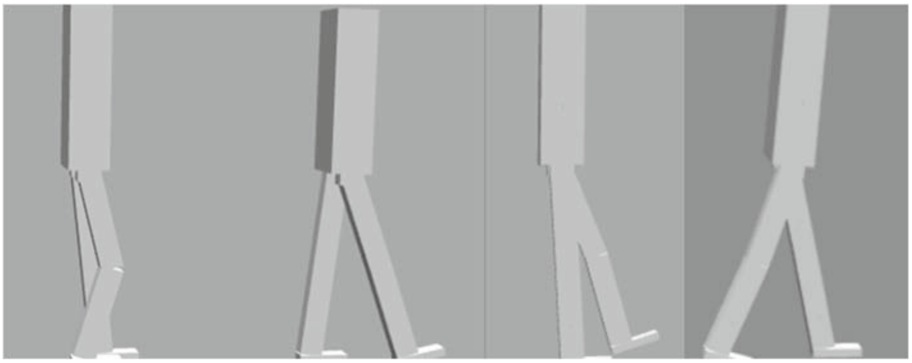

Fig. 1. Simplified gait cycle for right leg

Early Swing. Through this phase, the thigh will swing forward. The knee is bent to prevent the swing foot from hitting the ground. The swing angle of the hip joint and the duration of the swing is determined by the output of the reinforcement learning process.

Terminal Swing. Following early swing, the hip joint is locked for a short duration allowing the knee to straighten. This move is in preparation for making ground contact.

Stance. Once the foot touches the ground, the biped will rotate around the ankle joint like an inverted pendulum. The hip joint is then unlocked. A PID controller is tuned to control the torso pitch via control of hip joint velocity.

Toe-Off. The stance of the current leg will end when the opposite foot makes contact with the ground and enters its stance phase. The current leg will enter the toe-off phase. To do so, a torque is applied to the ankle joint to drive the foot to push off. This pushing action will propel the biped forward. The amount of the torque is determined by the current walking speed. Following the pushing action, a torque is applied on the ankle joint to quickly retract the foot from the ground.

2.4 Control

DDPG. In this work the DDPG network is used to control the step length and step duration in the forward swing phase.It was previously believed that the deterministic policy gradient of a model free network did not exist, but later it is proved that it does indeed exist [8] and is easier to compute than stochastic policy gradient for it only need to integrate in the state space. The deterministic policy gradient is:

$$\nabla_\theta J(\pi_\theta) = \int_S \rho^\pi(s) \nabla_\theta \pi_\theta(s) \nabla_a Q^\pi(s,a)|_{a=\pi_\theta(s)} ds \tag{1}$$

the deterministic policy gradient can be treated as two parts. One is the gradient of the action value to actions, and another is the gradient of the policy to the policy parameters. DDPG uses actor critic framework. The action value is approximate by critic using a DNN. The parameter of the network is update using temporal-difference method in the similar way as traditional actor-critic. The actor also uses a DNN as policy. The policy parameters are update by deterministic policy gradient $\nabla_\theta J(\pi_\theta)$. DDPG also uses replay buffer to store transitions to break correlation in the sample trajectory. When training the actor network, the policy will change constantly. So the temporal difference is calculated by a copy of the actor, critic network. It is called target network. These network only update after a period of time, or update at a very small changes. This off policy method allow the behavior to be more stochastic to explore the environment and keep the prediction deterministic. The target network is update by soft replacement method.

$$\theta' \leftarrow \tau\theta + (1-\tau)\theta' \tag{2}$$

The full biped system state, includes position, velocity, and acceleration terms for all 10 degrees of freedom. This many inputs can lead to network convergence issues and require the use of a very large network to sufficiently understand the interaction of the different state variables together. We simplify the input. As the biped will never leave the ground during normal operation, walking is limited to the sagittal plane by frictionless rails in the simulation, and lastly there exists a controller to stabilize torso pitch angle, we reduce the model to include only the following: ϕ: torso pitch angle, v: torso forward speed, l: the actual step length and d_{zmp}: the distance between the ZMP and the foot. The input of the network, state s, is,

$$s = [v, \phi, d_{zmp}, l]. \tag{3}$$

Note that the “existing controller for torso pitch” is a PID that controls the torso pitch through the hip joint velocity. This controller will be explained in the next subsection (Fig. 2).

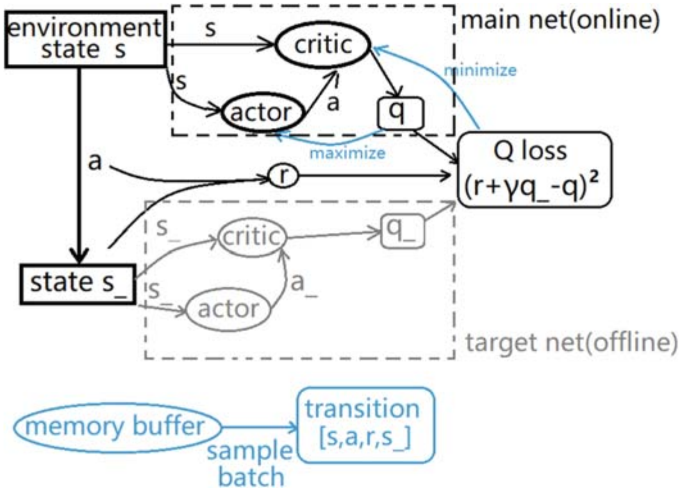

Fig. 2. The critic and actor network (evaluate/target) both have two hidden layers, the first layer has 400 neurons and the second layer has 300 neurons. The activation function is Relu. And the output of the actor network goes through a tanh activation function. The network has a memory of 70000 step. The learning rate of the actor and critic are set to be 1e-8 and 2e-8. The reward discount is set to 0.99. The training batch is 32 samples.

ZMP has been often used in biped control to evaluate the stability of the system and to drive control algorithms. If the ZMP is outside the support area, the biped can tip over and fall [9]. The state variables phi and v are measured by the IMU sensor attached to the torso. The step length l can be calculated from the forward kinematics in real robot, in simulation, it can be read directly from GAZEBO. ZMP is calculated after measuring the acceleration. From Cart-Table Model:

$$y_{zmp} = y_{com} - \frac{\ddot{y}_{com}}{g} z_{com}. \tag{4}$$

Although the biped is in a simulated environment, the acceleration measured by the IMU has noise due to the surface contact model from physics engine itself.

This noise must be filtered before the value can be used in any calculation. In this work, a mean value and Kalman filter is used on the acceleration data [10]. This is important because, in a physical system, the measurements of the acceleration are often extremely noisy as well. The state is updated at the moment when the front foot contacts the ground, then passed to the network which returns an action. Decaying noise is added to the action chosen to promote initial exploration but then allows refinement over time,

$$a' \sim (N, \sigma^2). \tag{5}$$

Once training is completed the system will run forward without additional noise added to the action selection (Fig. 3).

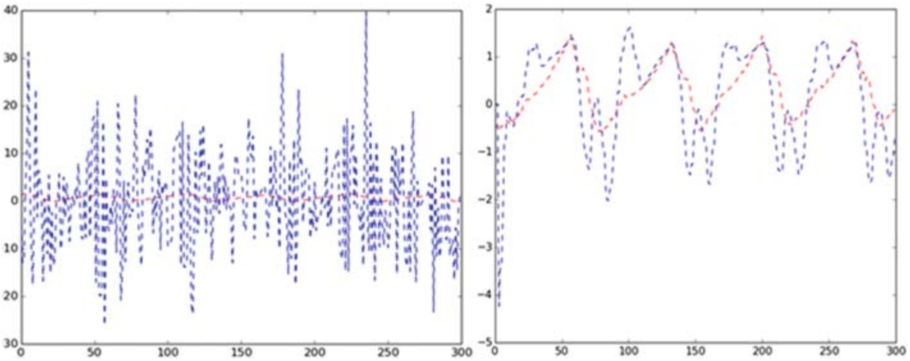

Fig. 3. Before and after the filter. Blue line is the acceleration. Red line is the velocity (Color figure online)

The trained network decides how far and how fast to put the next step based on the velocity, torso pitch angle, step length and ZMP position of the previous step. This way, the network requires less state input. It is noted the states are only sampled with every foot step. Consequently, if there is any major disturbance in between two foot steps, the network will not respond in time to compare with other more quickly updated systems. The network must wait until the foot touches the ground to update the state. But since the output of the network is the length and duration of the next step, as long as the biped won't fall between two steps, it can counter the disturbance by adjusting the output of the next step. To speed up the training, the output is initialized based on Height-to-Stride-Length Ratio. A better starting point makes the network converge more quickly.

In order to train the control network using a reinforcement learning approach, a reward function is created to indicate if the actions taken by the controller are either good or bad. The reward function used here takes into consideration the same variables as the state vector, where every element is normalized and weighted. The weights of every factor can change and the network will try to maximize the most weighted factor at first.

Stance Controller. When the foot touches the ground, the biped will start to rotate around the ankle joint. In this phase, the hip joint needs to move according to the ankle joint to keep the torso up straight and provide power to drive the torso forward. The output of the controller here deemed the "stance controller" is the angular velocity of the hip joint. The goal is to keep the torso upright without overshoot which would cause the torso to pitch back and forth jeopardizing stability. Ideally the torso is pitched slightly forward to maintain momentum and a smooth natural walking gait. To achieve this, a proportional controller is designed, and the residual error from the controller will allow the torso to slightly pitch away from the z axis (Fig. 4).

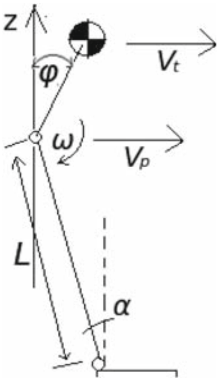

Fig. 4. Biped in stance phase

With the torso pitch remaining constant with respect to the z axis, the horizontal velocity of the hip will be the same as the horizontal velocity of the torso center. Given the following,

$$v_t = v_p, \tag{6}$$

and,

$$\omega = -\dot{\alpha}, \tag{7}$$

the angular velocity of the ankle can be read directly. The moment when the foot impacts the ground, noise will be introduced. So the angular velocity of the ankle is calculated by,

$$\dot{\alpha} = \frac{v_p}{\cos\alpha * L}. \tag{8}$$

We thus design a controller given that,

$$\omega = K * \phi = -\dot{\alpha}, \tag{9}$$

where if the pitch angle is larger than the target pitch,

$$\phi > \phi_0, \tag{10}$$

then,

$$|\omega| > |\dot{\alpha}|. \tag{11}$$

Thus the pitch angle will decrease and vice versa. The control gain K will be,

$$K = \frac{-\dot{\alpha}}{\phi_0} = \frac{-v_p}{\cos\alpha * L * \phi_0}, \tag{12}$$

where target pitch is chosen to be close to zero,

$$\phi_0 = 0.02. \tag{13}$$

it cannot be too close to zero, otherwise it will produce a very large gain, causing the system to be sensitive to noise.

Ankle Torque Control. The ankle joint is passive except in the toe-off phase. The advantages of setting the ankle to be a passive joint are as follows: (1) smoother ground contact for the foot; (2) dynamic property of the inverted pendulum is maintained; (3) minimal force is needed to drive the biped around the ankle when the foot is in contact with the ground; and (4) total noise of the system is reduced. The damping coefficient of the ankle is set to 1. This amount of damping helps to absorb the impact from the ground contact without hindering the swing motion (Fig. 5).

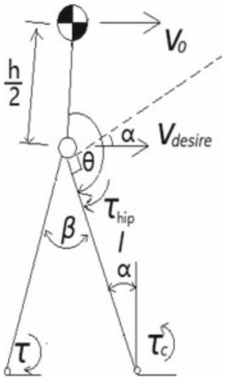

Fig. 5. Biped in toe-off phase

In the toe-off phase, a torque is applied on the ankle to propel the biped forward. The torque is determined by the current walking speed. The goal is to maintain the momentum of the biped within a certain range. If the desired walking speed is given then,

$$\Delta v = v_0 - v_{desire}, \tag{14}$$

and if the torso pitch remains constant, then the angular velocity of the torso is zero,

$$\omega_{torso} = 0. \tag{15}$$

Subsequently the velocity of the hip is equivalent to the velocity of the center of the torso,

$$\Delta v_{center} = \Delta v_{hip}. \tag{16}$$

assuming the toe-off phase is very short, the hip joint angle of the rear leg keeps the same during toe-off, and the momentum of the rear foot can be overlooked. To keep the torso angular velocity $\omega_{torso} = 0$, a torqueτ_{hip} must act on the hip joint of the front leg.

$$\tau_{hip} * \Delta t = J_{torso} * \Delta\dot{\alpha}, \tag{17}$$

$$J_{torso} \approx \frac{1}{3}mh^2 \tag{18}$$

About the front foot ankle joint we have the following,

$$(\tau - \tau_c - \tau_{hip}) * \Delta t = J_{leg} * \Delta\dot{\alpha} \tag{19}$$

$$\Delta\dot{\alpha} = \frac{\Delta v_{hip}/\cos\alpha}{l} \tag{20}$$

$$J_{leg} \approx \frac{1}{12}m_l l^2 + m_l[l^2 sin^2\beta + (l\cos\beta - \frac{l}{2})^2] + \frac{1}{3}m_l l^2, \tag{21}$$

$$\tau_c = c * \dot{\alpha}, \tag{22}$$

J_{leg} is the Moment of inertia of front and rear leg about front ankle joint. c is the damping coefficient of the ankle joint.

This controller in the future could also be changed to a network trained using the reinforcement learning paradigm.

3 Results and Conclusion

The average walking speed of the biped was approximately 1 m/s. The maximum recorded speed occurs just before the front foot contacts the ground, when the stance leg is perpendicular to the ground. To test the stability of the biped while walking an impulse was applied to different locations on the robot. It was found that the biped was able to remain stable and continue walking after a maximum impulse of 30 N-s was applied to the back of the robot as well as after a maximum impulse of 40 N-s was applied to the front of the robot. During testing all impulses were applied for a duration of 0.1 s. It can be seen in Fig. 6, that after applying the impulse, the robot's velocity drastically increases or decreases, depending on the direction of the impulse, but then returns to a consistent oscillation in less than 5 s. Keeping the pitch of the torso below −0.15 rad during walking, keeps the oscillation less than 0.1 rad. It was found experimentally that the biped was able to resist larger disturbances when the robot was in the toe-off phase of the gait compared to the forward-swing. Increasing the target walking speed v_{desire}

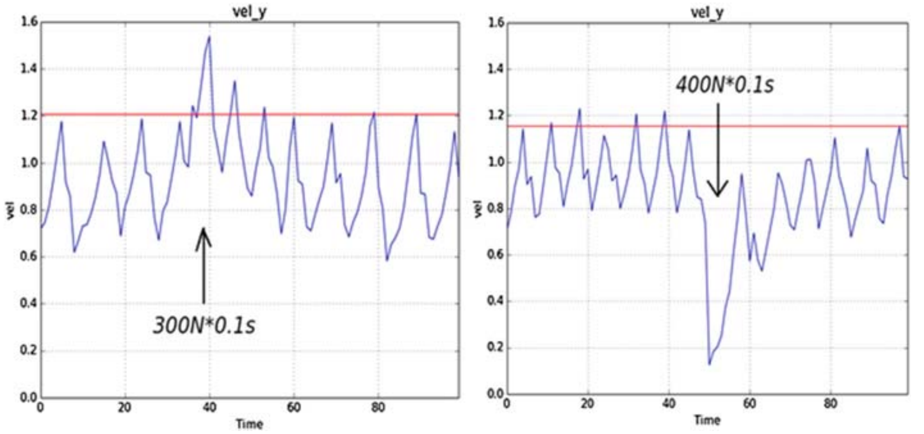

Fig. 6. Biped velocity response under impulse load

and lowering the damping coefficient of the ankle joint were shown to increase the overall speed of the robot but reduced the robustness of the system causing instability at lower impulses (Fig. 7).

In Fig. 8 it can be seen that a positive impulse disturbance applied to the biped will cause an increase in torso speed. To recover from this disturbance the DDPG network increases step length and decreases step duration accordingly to regain stability. When a negative impulse is applied to the biped, the DDPG network reduces step length and increases step duration to adapt to a lower speed. All the adjustments made by the DDPG network to retain stability were

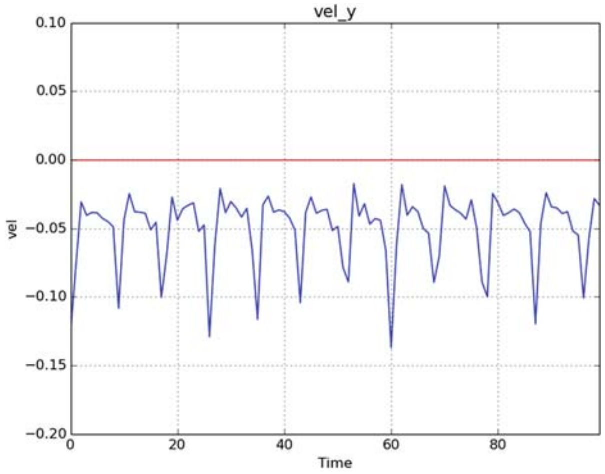

Fig. 7. Torso pitch angle during normal walking

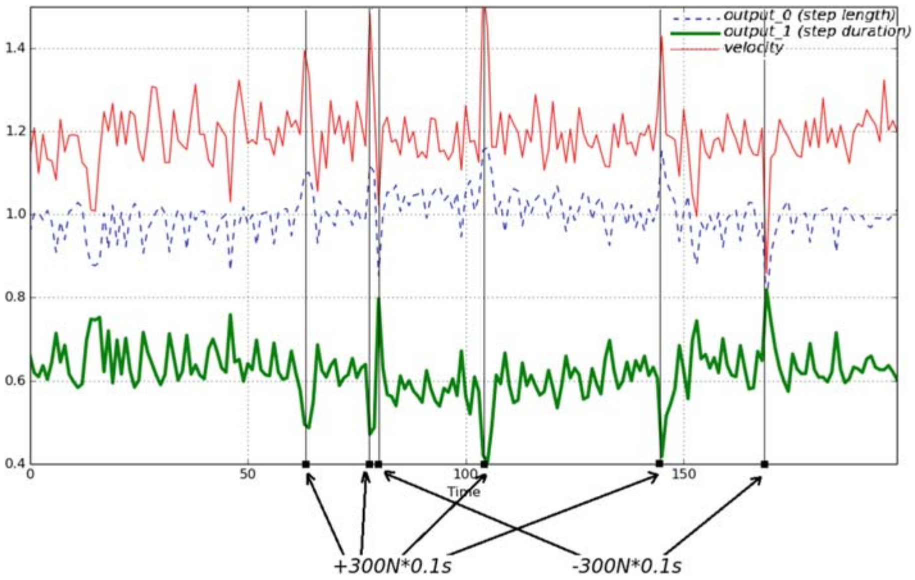

Fig. 8. DDPG network output at different torso speed

learned purely by experience without prior knowledge. When training the DDPG network, a large memory storage is necessary. The simplified input state proved to be sufficient to train a successful network.

An even further simplified state input $s = [d_{zmp}, \phi]$ was additionally used to train a network with the same parameters, but only but even after extended training period, it did not converge. The over-simplified state input cannot describe the environment adequately thus the DDPG network cannot make right decision.

References

1. Lonsberry, A.G., Lonsberry, A.J., Quinn, R.D.: Deep dynamic programming: optimal control with continuous model learning of a nonlinear muscle actuated arm. In: Mangan, M., Cutkosky, M., Mura, A., Verschure, P.F.M.J., Prescott, T., Lepora, N. (eds.) Living Machines 2017. LNCS (LNAI), vol. 10384, pp. 255–266. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-63537-8_22
2. Chang, S.R., Nandor, M.J., Li, L., et al.: A muscle-driven approach to restore stepping with an exoskeleton for individuals with paraplegia. J. NeuroEng. Rehabil. **14**, 48 (2017)
3. Morimoto, J., Doya, K.: Acquisition of stand-up behavior by a real robot using hierarchical reinforcement learning. Robot. Auton. Syst. **36**(1), 37–51 (2001)
4. Cashmore, M., et al.: ROSPlan: planning in the robot operating system. In: ICAPS (2015)
5. Ioffe, S., Szegedy, C.: Batch normalization: accelerating deep network training by reducing internal covariate shift. arXiv preprint arXiv:1502.03167 (2015)

6. Hausdorff, J.M., Peng, C.K., Ladin, Z.V.I., Wei, J.Y., Goldberger, A.L.: Is walking a random walk? Evidence for long-range correlations in stride interval of human gait. J. Appl. Physiol. **78**(1), 349–358 (1995)
7. Sepulveda, F., Wells, D.M., Vaughan, C.L.: A neural network representation of electromyography and joint dynamics in human gait. J. Biomech. **26**(2), 101–109 (1993)
8. Silver, D., Lever, G., Heess, N., Degris, T., Wierstra, D., Riedmiller, M.: Deterministic policy gradient algorithms. In: ICML, June 2014
9. Vukobratović, M., Borovac, B.: Zero-moment point-thirty five years of its life. Int. J. humanoid Robot. **1**(01), 157–173 (2004)
10. Grewal, M.S.: Kalman filtering. In: Lovric, M. (ed.) International Encyclopedia of Statistical Science, pp. 705–708. Springer, Heidelberg (2011). https://doi.org/10.1007/978-3-642-04898-2
11. Song, S., Geyer, H.: Evaluation of a neuromechanical walking control model using disturbance experiments. Front. Comput. Neurosci. **11**, 15 (2017)