

Learning Arm Motion Strategies for Balance Recovery of Humanoid Robots

Masaki Nakada*, Brian Allen†, Shigeo Morishima*, Demetri Terzopoulos†

*Faculty of Science and Engineering
Waseda University, Tokyo, Japan
Email: masakinakada@akane.waseda.jp

†Department of Computer Science
University of California, Los Angeles, USA

Abstract—Humans are able to robustly maintain balance in the presence of disturbances by combining a variety of control strategies using posture adjustments and limb motions. Such responses can be applied to balance control in two-armed bipedal robots. We present an upper-body control strategy for improving balance in a humanoid robot. Our method improves on lower-body balance techniques by introducing an arm rotation strategy (ARS). The ARS uses Q-learning to map sensed state to the appropriate arm control torques. We demonstrate successful balance in a physically-simulated humanoid robot, in response to perturbations that overwhelm lower-body balance strategies alone.

I. INTRODUCTION

Balance control is an important topic for humanoid robotics and is becoming increasingly necessary for humanoid robots that must function within a human-centric environment. Regardless of the quality of bipedal locomotion, a humanoid robot must still be prepared for unexpected perturbations that could throw it off balance. These events are unpredictable and potentially unavoidable, therefore, it is necessary to have robust controllers for balance maintenance and recovery.

When the disturbance is relatively small, it is sufficient to add torque at the ankles in order to create an angular momentum that recovers balance. In the face of larger perturbations, however, additional recovery strategies are needed, such as bending at the hips to produce an additional restoring momentum or taking protective steps to bring the center of pressure back within the support region.

Balance recovery is difficult, as are other aspects of bipedal locomotion research. Humanoid robot walking dynamics are non-linear and high dimensional. Moreover, it is sometimes impossible to actuate torques when the ground is too rough or the robot stands on one leg. In the former case, it is hard to compute when and where to step because the expected force and torque is ambiguous and it remains a difficult problem to recover from the unstable condition. However, humans and other animals can maintain their balance in spite of these theoretical difficulties. This implies the existence of good, biomimetic balancing controllers for humanoid robots.

While many researchers have considered the role of the lower body in balance, the upper body has been relatively

neglected. The inverted pendulum with a massless leg has been a very common model for biped research because of the convenience (e.g., the Linear Inverted Pendulum Model). This approach has contributed to much research in bipedal locomotion. However, the human upper body also plays a role in locomotion and the maintenance of balance. Human movements such as a forward lunge and arm rotation strategies (ARS) are able to alter the angular momentum in order to maintain balance. Humans can accomplish useful ARS behaviors subconsciously.

The focus of this paper is the use of ARS to maintain balance, thereby increasing the stability of bipedal robots. Unfortunately, it is nontrivial to calculate the necessary torque because of the complexity of the dynamics and random effects from the environment. The timing and strength of the required torque are very important. Rather than attempting a physical computation, we use machine learning to determine how to react appropriately depending on the situation.

Reinforcement learning is useful when we have a specific goal but are uncertain how to achieve it. Our problem is of this sort, where the final goal is defined as a stable balance, but the process to reach the goal is unclear. We employ the Q-Learning method, which is one of the most popular reinforcement learning algorithms. Even though this algorithm is simple, it is powerful enough to find an optimal solution that achieves the goal.

II. RELATED WORK

We discuss three areas related to our research: (1) bipedal balancing, (2) balancing methods that consider the torso's angular momentum, and (3) the use of machine learning for balancing.

Humanoid postural stability has been of research interest for a considerable length of time. The zero-moment point (ZMP) was introduced forty years ago [1], providing an instantaneous measure of bipedal balance. When statically stable, the ZMP is equivalent to the measured center of pressure (CoP) [2].

Hermami and Katbab [3] presented ankle and hip strategies with an inverted pendulum model. This enabled a push recovery for small perturbation. The stepping strategy presented by

Goddard was introduced as a third strategy to prepare for a larger perturbation. These three strategies; ankle, hip, and step, were combined by Hofmann et al. [4] and they have been used as a basic strategy for biped balancing.

The linear inverted pendulum model (LIPM) [5] considers a bipedal robot with a COM constrained to the horizontal plane; therefore, the dynamics can be analyzed in one-dimension. Because of its simplicity, the model has been used in much research on bipedal walking.

Fundamentally, balancing a biped requires maintaining the CoP within a region of support. Small changes in the CoP can be directly actuated by applying torques at the ankles. LIPM has been extended, by considering angular momentum, which is known as the reaction mass pendulum (RMP) model [6].

Pratt et al. [7] suggest that angular momentum can be regulated to improve balance, proposing the linear inverted pendulum plus flywheel model. This model adds a rotational inertia component to the LIPM. The augmented model has an enlarged capture region, defined as the area a potential protective-stepping foot can be placed to keep balance.

Goswami and Kallem [8] quantify the role of angular momentum in balance with the introduction of the ground point with zero rate of change of angular momentum (ZRAM).

In the field of character animation, [9] proposed a linear and angular momentum controller for both CoM and CoP. This work focused both on linear and angular momentum and made it possible to control them simultaneously, leading to more natural looking results. The arm was lifted when the body was pushed off balance by an external force, resulting in a natural human-like behavior.

In this paper, we consider the application of torque at the shoulder joint of our biomechanical model to create arm movement, which we expect to change the angular momentum, thus restoring balance.

Various machine learning algorithms have been applied to bipedal walking balance. The approach in [10] was to determine the capture point; i.e. the point where the robot should step in order to recover balance after a perturbation. Even though, in theory, they could compute where to step, it was difficult to realize this computation in a real-time simulation because of the modeling assumptions and errors; even tiny errors can result in a balance failure. Therefore, they applied a machine learning algorithm to learn how to adjust those errors and get the appropriate capture points, leading to greater robustness.

Ito et al. [11] applied a machine learning technique to learn the torque pattern for balancing. They could eliminate the process of feedback from the ground reaction force. The information is essential to maintain balance; however, it is not easy to calculate the right magnitude. They could ignore this information by learning to add a torque with periodic external forces. This learning process yielded a controller for each situation.

III. THE ARM ROTATION STRATEGY

Human balance strategies can be divided into three categories based on the magnitude of the balance-disturbing perturbation [12]. When the perturbation is small, balance is maintained by applying torque at the ankle. In a more severe perturbation, the hip joint is also recruited. For perturbations that cannot be balanced by hip and ankle torques, humans will take a protective step [13].

In response to some perturbations, humans augment lower-body balancing torques with upper-body motions. When pushed, our arms automatically engage in reactive behavior so that we can maintain the stability of our body. This action is usually automatically created by our neuromuscular system; therefore we seldom realize how we react to a certain perturbation.

One of the reasons that we move our arms is simply to change the position of our CoM and CoP. If we are leaning forward on the edge of box, for example, we will try to keep balance by stretching out our arms backward. This is very simple way to maintain balance and the arms clearly has an important role in controlling balance.

As a next stage, assume that we are unexpectedly pushed by someone. We will generally apply a balance-restoring torque at the ankles as was done in prior research as a first attempt to maintain balance. The ankles play an important role in balance maintenance; however, it is not the only action that we take as a first recovery strategy. We also use our upper body and rotate our arms to maintain balance. Therefore, this also needs to be considered in humanoid robots or physically simulated characters in order to improve balance control and yield more human-like behavior.

The main purpose of the ARS is to create angular momentum by the rotation and to defeat the angular momentum associated with falling. The magnitude of the momentum depends on the speed of the rotation, weight, and length of the arms. This momentum affects the torso and, in theory, the amount can be calculated through articulated-body dynamics. However, it is not easy to compute in the real world because of the high dimensionality and the unexpected nature of external forces.

Additionally, the timing of the ARS is also very important, because the angular momentum created by a perturbation changes consistently. Therefore, we need to apply adequate compensatory torque with specific timing that counteracts the falling momentum and returns the body to a stable state.

We apply a reinforcement learning algorithm for humanoids so that they can learn how to rotate their arms depending on their state and the strength of the perturbations.

A. The Role of Angular Momentum

As a first step, we consider the case when the disturbance is not very strong and the ankle strategy suffices to maintain balance. For this strategy, we show the importance of angular velocity by considering the total torques on the body,

$$\sum T_{CoM} = \sum T_{ext} + C_i + r_{oc} \times F_i = 0, \quad (1)$$

where T_{CoM} is an torque added to the center of mass, T_{ext} is the sum of the torque by external force, C_i is the inertia couple, and r_{oc} is the distance from origin to the CoM. F_i is the inertial force. This equation is simply derived from newtons 2nd law.

If we restrict our consideration to two dimensions, we can derive the following formula from equation (1):

$$x_{COP} = x_{COM} + \frac{\dot{H}_G - y_G \dot{L}_x}{\dot{L}_y + mg} + \frac{y_F F_{dist}}{\dot{L}_y + mg} \quad (2)$$

where x_{COP} and x_{COM} are the position of CoP and CoM, respectively, \dot{H}_G is the rate of change of the angular momentum around CoM, \dot{L}_x and \dot{L}_y are the x and y components of the time derivative of the linear momentum, and g is gravity.

This equation shows we can change the position of the CoP by changing \dot{H}_G . This means the time derivative of the angular momentum, which is the rate of change of the angular momentum around the CoM, can make difference of the position of the CoP.

When we control the balance using an ankle strategy, this CoP is a very important constraint. It must be inside of the region of support. Moreover, this determines whether or not balance can be maintained. Therefore, good control of the CoP is necessary and Equation (2) shows angular momentum around CoM affects the position of the CoP. Now, we know that the rate of the change of angular momentum is the factor of balancing, which means we must take the angular momentum into consideration when we investigate humanoid balance.

Additionally, this equation shows that increasing \dot{H}_G decreases the effect of F_{dist} . Therefore, it is better to rotate the body in the direction of the disturbance as fast as we can if we want to minimize the effect of the disturbance.

When our body is in motion, the linear momentum L and angular momentum H can be written as

$$\begin{bmatrix} L_G \\ H_G \end{bmatrix} = \begin{bmatrix} mE & -mr_{oc}^\wedge & M_{\dot{\theta}} \\ 0 & I & H_{\dot{\theta}} \end{bmatrix} \begin{bmatrix} v_B \\ \omega_B \\ \dot{\theta} \end{bmatrix} \quad (3)$$

where E is the 3×3 identity matrix, m is a total scalar mass of the robot, r_{oc} is a vector representing the distance from base to CoM and a^\wedge represents the skew-symmetric matrix of vector a , I is the 3×3 inertia tensor with respect to the CoM, $M_{\dot{\theta}}$ and $H_{\dot{\theta}}$ are $3 \times n$ (where n is the number of degree of freedom) linear and angular inertia matrices.

If we divide each inertia matrix to each part of the body, we can suppose the following relationship:

$$\begin{aligned} M_{\dot{\theta}} &= \{M_{leg1}, M_{leg2}, M_{arm1}, M_{arm2}, M_{free}\} \\ H_{\dot{\theta}} &= \{H_{leg1}, H_{leg2}, H_{arm1}, H_{arm2}, H_{free}\} \end{aligned} \quad (4)$$

where M_{legi} and H_{armi} are the linear and angular inertia matrix of leg and arm, and M_{free} and H_{free} are the matrix of the composite of the rest of body parts, respectively. We can also divide the joint speed vector as follows:

$$\dot{\theta} = \{\dot{\theta}_{leg1}^T, \dot{\theta}_{leg2}^T, \dot{\theta}_{arm1}^T, \dot{\theta}_{arm2}^T, \dot{\theta}_{free}^T\} \quad (5)$$

Then, we can rewrite Equation 3 as,

$$\begin{aligned} \begin{bmatrix} L_G \\ H_G \end{bmatrix} &= \begin{bmatrix} mE & -mr_{oc}^\wedge \\ 0 & I \end{bmatrix} \begin{bmatrix} v_B \\ \omega_B \end{bmatrix} \\ &+ \sum_{i=1}^2 \begin{bmatrix} M_{legi} \\ H_{legi} \end{bmatrix} \dot{\theta}_{legi} \\ &+ \sum_{i=1}^2 \begin{bmatrix} M_{armi} \\ H_{armi} \end{bmatrix} \dot{\theta}_{armi} \\ &+ \begin{bmatrix} M_{free} \\ H_{free} \end{bmatrix} \dot{\theta}_{free} \end{aligned} \quad (6)$$

From this, we can derive the following equation for H_G :

$$\begin{aligned} H_G &= I\omega_B + \sum_{i=1}^2 H_{legi} \dot{\theta}_{legi} + \sum_{i=1}^2 H_{armi} \dot{\theta}_{armi} \\ &+ \sum_{i=1}^2 H_{free} \dot{\theta}_{free} \end{aligned} \quad (7)$$

This equation shows that when $\dot{\theta}_{armi}$ changes, H_G also changes depending on the value of $\dot{\theta}_{armi}$. This means that the joint speed of the arms affect angular momentum. That is to say, ARS can control the angular momentum around the CoM. As we explained in the previous section, H_G is a key element for balance; therefore, Equation (7) shows the ARS can contribute to bipedal balancing.

Goswami and Kalleem [8] considered the robot as rotationally stable when $\dot{H}_G = 0$, thus we use this as a control target for the robot.

1) *Application of Angular Momentum:* In the previous section, we could prove that principled changes in angular momentum are useful for bipedal balance. In this section, we show how the torque generated by the momentum works in balancing. We assume that the torque at the center of mass is τ , which is generated by angular momentum with ARS. We begin with a simple planar model. The equations of motion are represented as follows:

$$m\ddot{x} = f_R \sin(\theta) - \frac{\tau}{l} \cos(\theta) \quad (8)$$

$$m\ddot{z} = -mg + f_R \cos(\theta) + \frac{\tau}{l} \sin(\theta) \quad (9)$$

$$J\ddot{\theta} = \tau \quad (10)$$

where the f_R is the reaction force from the ground, m is the total mass of the model, θ and J is the tilting angle and rotational inertia of the body, respectively, g is the gravitational acceleration constant, x and z are the position of the CoM coordinates and l is the half length of the body.

Even though we did not use the simplified model in our experiments, let us consider the Linear Inverted Pendulum Model to illustrate the theory. Using this abstracted model, we can suppose z is a constant value, $z = z_0$ and $\ddot{z} = 0$, because it does not move to z direction. Then, we can solve for f_R from equation (9) as

$$f_R = \frac{mg}{\cos(\theta)} - \frac{1}{l} \tan(\theta)\tau \quad (11)$$

Replacing $\cos(\theta) = \frac{z_0}{l}$ and $\sin(\theta) = \frac{x}{l}$, Equation (11) can be written as

$$f_R = \frac{mg}{z_0}l - \frac{1}{l} \frac{x}{z_0} \tau \quad (12)$$

Next, we represent the f_R with constant values, x and τ . We can substitute f_R in the equation (8) and we obtain

$$\ddot{x} = \frac{g}{z_0}x - \frac{1}{mz_0} \tau \quad (13)$$

This equation shows that the value of τ changes the acceleration of the CoM. Therefore, we can accelerate the body by changing the amount of torque. Also, from Equation (10) we can derive the following equation:

$$\ddot{\theta} = \frac{1}{J} \tau \quad (14)$$

which shows that the torque can change the acceleration of the rotation of the body. Equations (13) and (14) show the torque around the CoM affects the balancing of the body. This torque is originally from the ARS; therefore, we can determine the effect of the ARS to the balance of the body in this way.

B. Reinforcement Learning

Machine learning is one of the most popular areas because of its potential uses in numerous fields. Many machine learning methods had been introduced. One way to categorize them is if they are supervised or unsupervised. Reinforcement learning [14] is an unsupervised machine learning method. One big advantage of reinforcement learning is that we only need to set the final goal and the remainder of the process can be learned by the agent itself. This is useful when the goal and the method are defined but the process is unknown.

In this work, we have a specific goal, which is to keep the body upright, but the best method to obtain this result is unclear. It is not easy to compute the exact torque and the timing of the reactive movement needed to recover balance. Therefore, we apply the reinforcement learning algorithm to our model, allowing it to learn how to maintain balance under a given perturbation, action settings, and environment. We expect it to learn an optimal process to reach its goal and realize balance control.

1) *Q-Learning*: We have chosen a common reinforcement learning algorithm, Q-Learning [15], based on the temporal difference approach. Q-Learning has an agent as a main character for the problem and an environment in which the agent can act and the state transitions is described.

When the agent performs an action in the environment, the state is changed. Then, the environment evaluates if the changing state was beneficial in achieving the final goal, and will then return a reward to the agent. Considering the action, state and reward, the agent decides which action it would take as a next step. This process is repeated until it reaches a final goal. In the end, it learns the best way to accomplish the goal.

The way they evaluate the reward is based on the following formula:

$$Q(a, s) \leftarrow Q(a, s) + \alpha [R(s) + \gamma \max_{a'} Q(a', s') - Q(a, s)] \quad (15)$$

This is called the Q value and the agent choose the action so that it can maximize this value. The parameter α is called the learning rate and it determines to what extent the newly acquired value affects the old value. The γ is called the discount factor and it determines how important the future rewards are. As this value increased, the future rewards are more carefully considered. These values must be tuned depending on the intent of the learning. This significantly affects the learning result. The decision-making function is called policy and Q-learning is the algorithm which can find the best policy within the given environment. The policy is stochastic; therefore, it takes some random action in a certain period. This prevents the agent from sticking in a local minimum.

IV. EXPERIMENTAL MODEL AND ENVIRONMENT

The Open Dynamics Engine (ODE) [16] was used to simulate the humanoid robot. Although ideal validation would use a physical robot, humanoid robots can be expensive and difficult to test without risk of damage or injury. On the other hand, simulation has the benefits of easy, safe and inexpensive prototyping. Further, measurement and analysis of the resulting motion is precise.

The system is simulated with realistic values for gravity (9.8 m/s^2) and friction. The simulated robot is of roughly human proportions and mass and bilaterally symmetric hands, arms, legs and feet. The specific dimensions of the model are provided in table I.

TABLE I
PHYSICAL PARAMETERS OF THE SIMULATED HUMANOID ROBOT.

	Mass (kg)	Length (cm)	Width (cm)	DoF
Torso	10	50	30	0
Upper Arm	0.1	0.1	0.1	2
Forearm	2	50	5	0
Thigh	5	50	15	0
Shin	0.1	0.1	0.1	1
Foot	0.5	20	10	0

A. Learning Parameters

As described in section III-B.1, the feature states, learning rate, discount factor, and reward function are important components, and they must be specified carefully. The results can be totally different depending on those factors. The parameter settings in our experiments are as follows:

1) *Feature States*: These are the values which are evaluated to determine whether the trial succeeded or failed. They can also be used in the reward function to define how much reward should be given to the agent.

We chose four parameters that we regard important as a way to judge if the posture of the robot is balancing. The four elements are the position and the velocity of the center of mass, the leaning angle, and the angular velocity of the body.

We also defined the limits for each parameter so that we can define if the trial is a success. When one of the feature states exceeds the limits, it is considered a failed trial and the learning process is reset. In our work, we judged the result as successful when every value for these 4 parameters are within the defined limit.

We also need an discretized table to store the Q values and upload them considering the future rewards. Q-Learning is one of the derivative-based methods from Temporal Difference approach. This approach considers a future result; therefore, it is necessary to keep the temporal result first, and then upload it reflecting the result of the future action.

Position, velocity, and angle are the continuous values; therefore, we need to discretize the space so that we can store the temporal value obtained in a certain state.

2) *Learning Rate*: This rate specifies how quickly the system should incorporate new information and can range from 0, implying no learning, to 1 which would cause the agent to rely only on current information. We use a learning rate of $\alpha = 0.1$.

3) *Discount Factor*: This parameter specifies how expected future rewards are treated, with values of 0 ignoring future rewards and 1 foregoing immediate rewards in favor of only considering long-term rewards. Our system specifies $\gamma = 0.9$.

4) *Reward Function*: The reward function has three possible values based on the instantaneous state of the system. Failure to maintain balance, detected by feature values outside an allowed, predefined range, results in a large negative reward of -4.0 . Feature values within the allowed range imply that the robot is balancing successfully, and a small positive reward of 2.0 is returned. Finally, if the robot is balancing, and the linear and angular velocity are also sufficiently close to zero, then the robot is in a very stable state and a more positive reward of 3.0 is given.

V. RESULTS

Performance of the algorithm is measured in simulation, as described in section IV. Successful balance behaviors were verified visually. Such manual verification was needed because the simple pre-defined range of feature values used for reward did not always imply true success, as the learning system sometimes found unexpected solutions. For example, the system could learn to stay upright by bracing itself with one hand on the ground. While such states qualified as success for the learning reward, they were not counted as successful balancing strategies. Such outcomes were rare, but their existence required the manual validation of balancing behaviors.

A. Learning Results

We considered the learning of bipedal balance maintenance using the Q-learning algorithm. We set the environment, states, parameters and feature values as we explained in the previous section and executed trials until the robot kept balance or fell down. The different parameter settings and the feature states resulted in different outcomes. This tuning is one of the

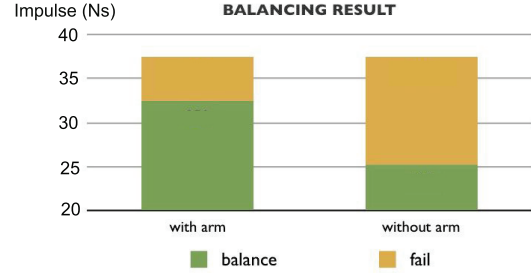


Fig. 1. This is a result of trials with and without ARS when perturbation comes only from back side of the model to forward. We changed the magnitude of the perturbation and got how extent the model can sustain its balance. ARS enlarged the range of perturbation impulses.

tricky parts in reinforcement learning; therefore, we had to try different combinations in order to find the best learning parameters. The learning process took approximately two minutes for each trial. This computational speed is acceptable because the process is off-line.

B. Quantitative Evaluation

Figure 1 shows graph of the range of the perturbation impulses limited to 1 direction. As a first experiment, we limited the direction of the force only to forward and tried with different forces changing every 1 N*s (Newton*seconds). We pushed our model from the back and measured how strongly it could resist the perturbation. The graph at the left shows the result with ARS and the one at the right shows the result without ARS. As we can see in the graphs, the range of the perturbation impulses is enlarged when we added ARS. This result proves the effectiveness of ARS.

As a next step, we expanded the strategy to space. We applied different perturbation forces from different directions—every 30 degrees with different forces. Figure 2 shows the result of the simulation. As shown in graph, the range of perturbation impulses is expanded by using our method. \circ and \triangle shows the maximum perturbations that the robot could withstand without losing balance with and without ARS, respectively. The distance from the center is equal to the strength of the perturbation and the angle from the horizontal axis is the direction of the perturbation. The most effective direction of our method was when the force was from 0, 90, 180, and 270 degrees. The other angles were not very different between the two strategies; however, it was slightly improved. Our method was robust to perturbations between 1.01 and 1.14 times the force required to disrupt the standing balance controller. This result shows that our strategy improves bipedal balancing.

We were able to improve the existing balancing method by using ARS. Our new method can be embedded between the ankle and hip controllers in the conventional push recovery strategy. We can consider our new strategy as a second step,

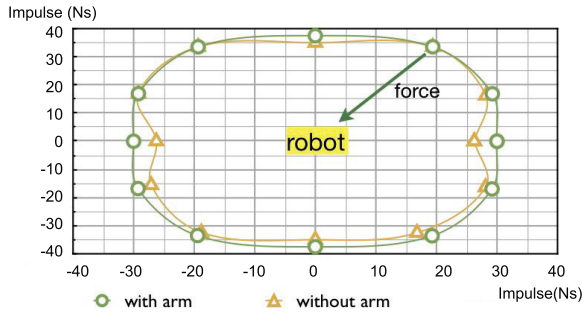


Fig. 2. This is a result of trials with and without ARS when perturbation came from various direction. We changed the magnitude and direction of the perturbation and got how extent the model can keep its balance. ARS enlarged the range of perturbation impulses.

yielding a new $3 + 1$ approach for push recovery.

VI. CONCLUSION

We have proposed a new arm rotation strategy (ARS) for recovering balance after a perturbation, potentially leading to more stable bipedal robots working in collaboration with traditional lower-body strategies. While simple controllers for upper body (e.g., ankle and hip PD controller) work well, no one has suggested simple and effective controller for lower body. Our strategy has improved the robustness to the severity of the disturbance and the complexity of the ground. It is more robust than traditional ones because it controls the upper body. When the terrain is rough, it is not easy to control the lower body. For instance, the torque at the ankle does not efficiently affect the body. Moreover, finding the capture points for stepping turns out to be extremely difficult. By contrast, the difference of the ground condition does not affect our strategy. Therefore, this strategy is effective in any situation and results in more robust balancing than the one without our method.

In our work, the Q-learning process was off-line; therefore we did not need to pay much attention for the computational cost. Even though the learning process can be computationally expensive, once a controller is found, the on-line execution is very fast.

We have employed simulation to test our method and have demonstrated the utility of ARS. We have simulated the robot and its environment realistically, but our simulator is still not

identical to the real world. The real world has random wind, energy loss, and some other unexpected factors. Therefore, the next step should be to validate our strategy by implementing it in physical anthropomorphic robots.

REFERENCES

- [1] M. Vukobratović, A. Frank, and D. Jurčić, "On the stability of biped locomotion." *IEEE transactions on bio-medical engineering*, vol. 17, no. 1, p. 25, 1970.
- [2] P. Sardain and G. Bessonnet, "Forces acting on a biped robot. center of pressure-zero moment point," *IEEE Transactions on Systems, Man, and Cybernetics, Part A: Systems and Humans*, vol. 34, no. 5, pp. 630–637, 2004.
- [3] H. Hemami and A. Katbab, "Constrained inverted pendulum model for evaluating upright postural stability," *Journal of Dynamic Systems, Measurement, and Control*, vol. 104, p. 343, 1982.
- [4] A. Hofmann, M. Popovic, and H. Herr, "Exploiting angular momentum to enhance bipedal center-of-mass control," *IEEE Trans. Rob. Autom.*, 2007.
- [5] S. Kajita, F. Kanehiro, K. Kaneko, K. Yokoi, and H. Hirukawa, "The 3D Linear Inverted Pendulum Mode: A simple modeling for a biped walking pattern generation," in *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2001, pp. 239–246.
- [6] S. Lee and A. Goswami, "Reaction mass pendulum (RMP): An explicit model for centroidal angular momentum of humanoid robots," in *2007 IEEE International Conference on Robotics and Automation*, 2007, pp. 4667–4672.
- [7] J. Pratt, J. Carff, S. Drakunov, and A. Goswami, "Capture point: A step toward humanoid push recovery," in *Humanoid Robots, 2006 6th IEEE-RAS International Conference on*, 2006, pp. 200–207.
- [8] A. Goswami and V. Kallem, "Rate of change of angular momentum and balance maintenance of biped robots," in *IEEE International Conference on Robotics and Automation*, vol. 4. Citeseer, 2004, pp. 3785–3790.
- [9] A. Macchietto, V. Zordan, and C. Shelton, "Momentum control for balance," *ACM Transactions on Graphics (TOG)*, vol. 28, no. 3, p. 80, 2009.
- [10] J. Rebula, F. Canas, J. Pratt, and A. Goswami, "Learning capture point for improved humanoid push recovery," in *IEEE-RAS 7th International Conference on Humanoid Robots*, Pittsburgh, PA, 2007.
- [11] S. Ito, K. Moriki, H. Kawasaki, and M. Sasaki, "Robot experiment of torque learning for biped balance with respect to periodic external force," in *Advanced Robotics, 2005. ICAR'05. Proceedings., 12th International Conference on*, 2005, pp. 418–423.
- [12] A. Alexandrov, A. Frolov, and J. Massion, "Axial synergies during human upper trunk bending," *Experimental Brain Research*, vol. 118, no. 2, pp. 210–220, 1998.
- [13] B. Maki, W. McIlroy, and S. Perry, "Influence of lateral destabilization on compensatory stepping responses," *Journal of biomechanics*, vol. 29, no. 3, pp. 343–353, 1996.
- [14] R. Sutton and A. Barto, *Reinforcement learning*. MIT Press, 1998.
- [15] C. Watkins and P. Dayan, "Q-learning," *Machine learning*, vol. 8, no. 3, pp. 279–292, 1992.
- [16] R. Smith. Open dynamics engine. [Online]. Available: <http://www.ode.org/>