Learning to Control a Biped with Feet

Bernhard Hengst
School of Computer Science
and Engineering
University of New South Wales
Sydney, Australia
Email: bernhardh@cse.unsw.edu.au

Manuel Lange
Eberhard Karls University of Tübingen
Wilhelm-Schickard-Institut für Informatik
Email: manuel.lange@student.uni-tuebingen.de
and
School of Computer Science
and Engineering
University of New South Wales
Sydney, Australia

Brock White
School of Computer Science
and Engineering
University of New South Wales
Sydney, Australia
Email: brockw@cse.unsw.edu.au

***Abstract*—This paper investigates the learning of a controller for a flat-footed bipedal robot using reinforcement learning to optimally respond to (1) external disturbances induced by stepping on objects or being pushed, and (2) rapid reversal of demanded walk direction. The reinforcement learning method employed learns an optimal policy by actuating the ankle joints to assert pressure along the support foot, and optionally the leg joints to determine the next swing foot placement. The controller is learnt in simulation using an inverted pendulum model and the policy transferred to two small physical humanoid robots.**

I. Introduction

Bipedal locomotion is often subjected to large impact forces induced by a robot inadvertently stepping on objects or by being pushed. In robotic soccer, for example, it is not uncommon for robots to step on each others feet or to be jostled by opposition players. At current RoboCup [1] competitions robots regularly fall over for these reasons in both humanoid and standard platform league matches[1]. Another requirement in soccer environments is that bipedal robots should be able to react optimally to rapidly changing directional goals. In soccer it is often necessary to stop suddenly after walking at maximum speed or to reverse direction as quickly as possible.

Reinforcement learning is a machine learning technique that can learn optimal control actions given a goal specified in terms of future rewards. In this paper we use reinforcement learning to decide actions that apply ankle torques to the support foot to move the centre of pressure (CoP). The policy can be changed while in mid-stride. We can also learn the placement of the swing foot. The result is an optimal policy that arbitrates both ankle and foot placement actions to pursue a changing goal in the face of continual disturbances.

We are interested in learning a dynamically stable gait for a planar biped. Reinforcement learning relies on many trials which makes learning directly on real robots expensive. Instead the controller is learnt using a simulated inverted pendulum model that has been parameterised to closely correspond to the physical robot. The policy is then transferred to the real robot without modification. Our approach leaves open the ability to continue learning on the physical robot using the accumulated experience from the simulator as a starting point.

[1]See video accompanying this paper

Our contribution is learning a controller for a flat-footed biped that can optimally react to environmental disturbances and rapid changes in goal by simultaneously actuating the ankle of the support foot while positioning the swing foot.

In the rest of this paper we first describe our simulated system. We then provide a brief background on reinforcement learning and outline our approach to learning on the simulated biped. The behaviour for both sudden changes in policy and impulse forces in simulation are described. We also show how the policy is implemented on two physical robots by addressing practical aspects of system state estimation and policy implementation. Finally we discuss results, related and future work.

II. Simulation

We model the flat-footed humanoid as an inverted pendulum with the pivot located at the centre of pressure along the bottom of the support foot as shown in Figure 1. We can control the pivot position by actuating the ankle joint. For simulation purposes we discretise the pivot to be in one of three positions – at the toe, centre, or heel of the foot.

The state s of the system is defined by four variables $(x, \dot{x}, w, t)$ where x is the horizontal displacement from the centre of the support foot to the centre of mass, $\dot{x}$ is the horizontal velocity of the centre of mass, w is the horizontal displacement from the centre of mass to the centre of the swing foot, and t is the time-step from the start of each walk-cycle.

The control actions a are defined by a couple (c, d), where c chooses the centre of pressure for the support foot relative to the foot centre and d chooses a step change in the swing foot displacement w. We model the swing foot displacement in this way to ensure it is moved into position progressively.

The state transition function is determined by the inverted pendulum dynamics, the natural progression of time, the walking gait that determines when the swing and support feet alternate, and the change in w based on action d. The system difference equations with time indexed by k and time-step Δt

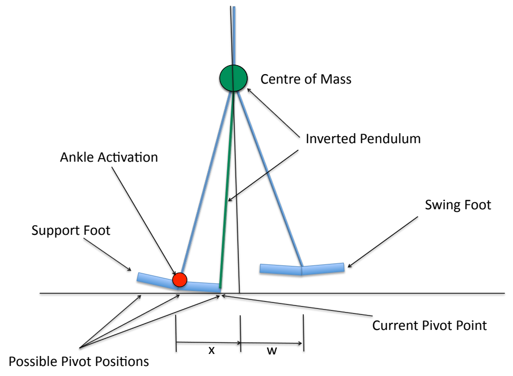

Fig. 1. The inverted pendulum model of a flat-footed bipedal robot used for simulation and reinforcement learning.

are:

$$x_{k+1} = x_k + \dot{x}_k\,\Delta t + \ddot{x}_k\frac{\Delta t^2}{2} \quad (1)$$

$$\dot{x}_{k+1} = \dot{x}_k + \ddot{x}_k\Delta t \quad (2)$$

$$\ddot{x}_{k+1} = g\; sin(\theta_{k+1})\; cos(\theta_{k+1}) \quad (3)$$

$$w_{k+1} = w_k + d \quad (4)$$

$$t_{k+1} = t_k + \Delta t \quad (5)$$

where g is the acceleration due to gravity and θ_k is the lean of the inverted pendulum in the clock-wise direction. θ is dependent on the pivot point p_k determined by c and the height of the centre of mass. The height h of the centre-of-mass of the pendulum is modelled on gait characteristics of the robot. We use a *linear inverted pendulum* – one for which the height of the CoM is constant , hence $\theta_k = \tan^{-1}((x_k - p_k)/h)$.

The period of a complete walk cycle is T. We assume that the time that the system is in a double support is small and can be ignored for the purposes of system identification. The support and swing feet alternate as the time passes through $t = T/2$ and $t = T = 0$. At these times the above transition equations are augmented by:

$$x_{k+1} = -w_{k+1} \quad (6)$$

$$w_{k+1} = -x_{k+1} \quad (7)$$

The state-space wraps around on itself after each walk-cycle. That is, if $t_{k+1} \geq T$ then $t_{k+1} = t_{k+1} - T$.

Figure 2 shows several frames from an animation of the inverted pendulum. The robot is depicted in both plan and elevation views, showing the feet, the centre of mass, and the currently actuated pivot.

III. Reinforcement Learning Representation

The formalism underpinning reinforcement learning (RL) is a Markov Decision Problem (MDP) $\langle S, A, T, R\rangle$, where S is a set of system states, A is a set of actions, $T : S \times A \times S \rightarrow [0, 1]$ is a stochastic transition function and

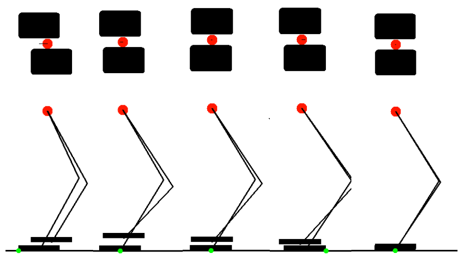

Fig. 2. A schematic depiction of the animated simulator showing several frames in plan-view (top) and side-view (bottom).

Variable	Values	Range	Increment
x	21	-50 to 50mm	5mm
$\dot{x}$	25	-300 to 300mm/sec	25mm/sec
w	21	-50 to 50mm	5mm
t	24	0 to 479 millisec	20millisec
c	3	-40 to 40mm	40mm
d	3	-5 to 5mm	5mm

TABLE I
THE NUMBER OF DISCRETE VALUES FOR STATE AND ACTION VARIABLES THAT ARE USED TO DEFINE THE SIZE OF THE Q TABLE AND THEIR MEANING IN TERMS OF A SPECIFIC INSTANCE OF THE ROBOT

$R : S \times A \times \mathbf{R} \rightarrow [0, 1]$ is a stochastic real-valued reward function. At each time-step k, the system transitions from the current state $s \in S$ to a next state $s' \in S$, given an action $a \in A$, and receives a reward $r \in \mathbf{R}$. The system trajectory is characterised by a sequence of states, actions and rewards $s_k, a_k, r_k, s_{k+1}, a_{k+1}, r_{k+1}, s_{k+2}, \ldots$. The objective is to maximise the future discounted sum of rewards $\sum_{t=0}^{\infty} E[\gamma^t r]$ where $t = 0$ is the current time-step, E is the expectation operator, $r \in \mathbf{R}$, and γ is a discount rate. We use *Q-Learning* [2], an off-policy temporal difference approach to learning the Q action-value function $Q : S \times A \rightarrow \mathbf{R}$.[2] After learning an optimal Q function the optimal control action a^* in state s is $max_a Q(s, a)$.

We now specify an instance of the above simulator and represent it as an MDP for reinforcement learning. The specific values of variables we use are: $T = 480$ milliseconds, centre of mass height $h = 260$mm or 300mm depending on the robot. We use straightforward discertisation of the above continuous variables as our linear function approximator [2]. The simulator uses a time-step of one millisecond, while the learner runs at 100Hz, the frequency that the physical robots are able to drive the motors and make inertial measurements. Table III provides the number of values used for the Q action-value table and the range of each variable.

Rewards are chosen to achieve certain goal states and avoid others. We use an arbitrary reward of -1000 if any of the state variables x, $\dot{x}$, and w move outside their range. If the goal is to balance the robot in an upright position we specify a reward of 1000 for each state where x and $\dot{x}$, and w are close to zero. We also add negative rewards for taking actions that

[2] Q-learning will in future work allow us to continue to learn on the physical robots after transferring the action-value function from simulation.

require motor movements, -1 for asserting toe or heel pressure and -10 for moving the swing foot, to encourage parsimonious movement.

If the goal is to walk forwards or backwards at a specified velocity we set the reward to 1000 for states x close to zero, $\dot{x}$ at the desired velocity, and t near $T/4$ and $3T/4$. This provides the reinforcement learner with two way-points in the walk-cycle, similar to [3] selecting actions at Poincaré sections [4], where the pendulum is upright and the centre of mass has the specified velocity component. The discount factor γ is set to 0.9 and the Q-learning rate is set to 0.05.

The simulator is used to provide training examples for the learner. While the simulator is a deterministic system, the function approximation makes the system only approximately Markov. The transitions function for the discrete system is approximated by a stochastic transition function. Care must be exercised in the learning regime. The pseudo stochastic transitions are actually a function of the policy and depend on a longer history of states and actions. We have a circular phenomenon where the policy is determined by the value function and the value function is dependent on the policy.

The trick we use is to start the simulator repeatedly at a random point in the state-space and execute a trajectory of 100 transitions (about two walk cycles) using an ϵ-greedy exploration function that executes the latest best policy 80% of the time. The effect is to bootstrap the stochastic transition function to operate with the latest best policy. Our other strategy to ameliorate the Markov approximation is to use an eligibility trace that has the added advantage in that it speeds up learning by accelerating the back-propagation of the reward signal.

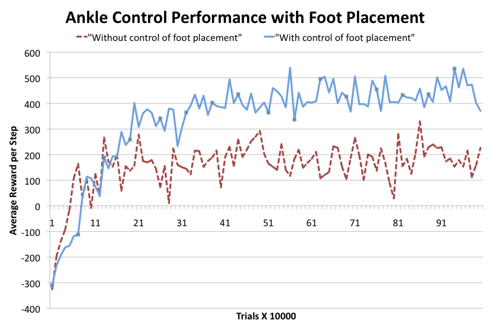

Fig. 3. The average reward per step during reinforcement learning. With a relatively small foot size (40mm) controlling both ankle and foot placement (top curve) shows a better performance than with ankle control alone.

Reinforcement learning is allowed to continue in simulation until the average reward per time-step settles to a maximum value. Typical learning profiles are shown in Figure 3. The final policy $\pi(x, \dot{x}, t, w) = (c, d)$ is frozen.

IV. Simulation Results

We conducted several experiments with the simulator to test the robustness of the learnt policy to impact forces and random changes in walking direction.

In the first set of experiments we only actuated the ankle to control the pivot. The value of w was set to x which simulated a swing foot movement that mirrored the support foot and placed the swing foot $2x$ from the support foot. In Figure 4 we plot x and $\dot{x}$ against time in two ways. The top diagrams show the time-trace on a coordinate system with horizontal axis x and vertical axis $\dot{x}$. The bottom diagram shows the unfolding time series for x (red) and $\dot{x}$ blue with the current time on the right.

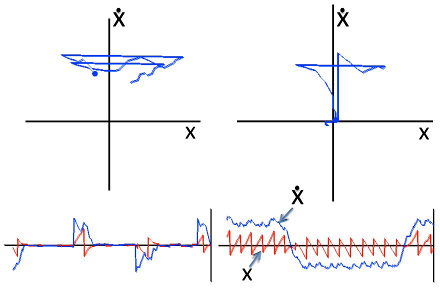

Fig. 4. Time-trace on a coordinate system with horizontal axis x and vertical axis $\dot{x}$ (top). Time series for x (red) and $\dot{x}$ blue with the current time on the right. (bottom). Response to impulse forces (left). Rapid change in walk direction (right)

Figure 4 (left) shows the time-series response for both x and $\dot{x}$ to sudden changes in $\dot{x}$ designed to simulated impact forces. As can be seen from the graph the deceleration induced by actuating ankles joints can persist over several walk cycles. The inclusion of the time variable t from the start of the walk-cycle as a part of the state of the system allows the learner to plan ahead and take appropriate actions now in anticipation of support foot changes.

The right of Figure 4 shows the response to sudden changes in walking direction. Two separate RL controllers are trained for forward and backward walks. By switching controllers the robot can be directed to change walk directions. The response to a new goal is acted on immediately, even while the swing foot is in mid-stride. As can be seen from the Figure, actuating ankles alone quickly achieves the desired outcome in change of direction of the walk.

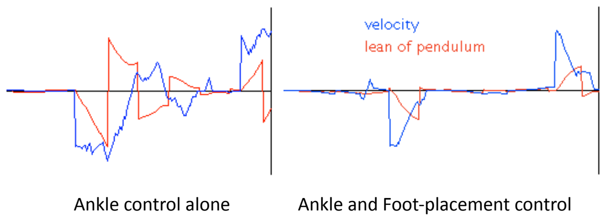

Fig. 5. Time-series impulse response comparison with and without swing-foot placement control for a foot size of 40mm. The time series show both x (red) and $\dot{x}$ blue. Ankle control alone (left). Both ankle and swing-foot placement control (right).

With the swing foot position w determined by incremental movement actions d, the swing foot is placed, in conjunction with the ankle control, to optimally arrest the motion of the robot following sudden impact forces. Figure 5 shows the typical response to impulse forces for a 40mm width foot with and without foot-placement control. For small feet, ankle control alone takes longer (and several steps) to rebalance the robot.

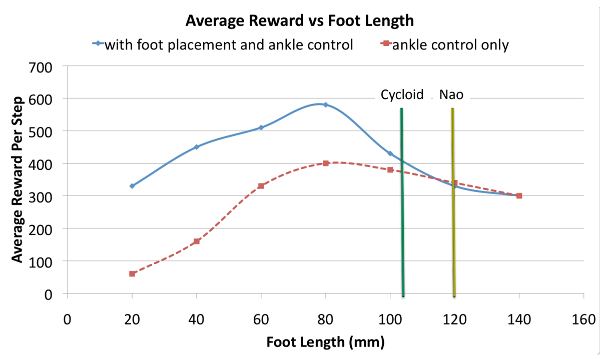

Fig. 6. Average reward per step after 100000 trials for various foot lengths. The advantage in the additional foot placement control is not significant in simulation for the feet size of the Nao and Cycloid.

Figure 6 shows the difference in average reward per step when foot placement control is allowed in addition to ankle control for various foot lengths. As one would expect with increase in foot size the difference is eroded. The surprising drop in average reward with increase in foot size we attribute to the crude three-value discretisation for the pivot position. With larger foot sizes the simulated ankle control torque at the heel and toe is large and has difficulty keeping the robot in the part of the state-space that achieves the high reward. The foot sizes of the two robots we use are 120mm and 106mm. For this reason our physical experiments focus on controlling the ankle tilt.

Even if the swing foot is placed in relation to the support foot and CoM, the learner still takes into consideration the swing-foot policy when deciding the ankle control policy. Figure 7 shows the policy for x and $\dot{x}$ in relief half-way through a swing phase (at $t = 360ms$).

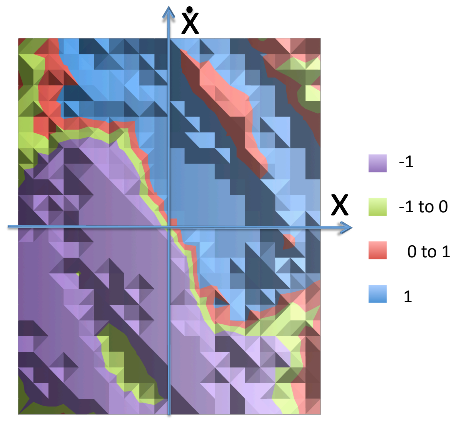

Fig. 7. Reinforcement learnt ankle control policy determining the foot CoP for x vs $\dot{x}$ in the middle of the swing phase. The CoP pivot point can range from -1 (heel) to 1 (toe).

V. Physical Robot Implementation

The policy learnt on the simulator was transferred to the Cycloid and Nao robots (Figure 8). The physical robots execute a walking gait by driving the hip, knee and ankle motors using closed form kinematics to keep the CoM at a constant height. As the robot is moved forward the support foot is assumed to be close to flat on the ground and the swing foot is kept parallel to the ground.

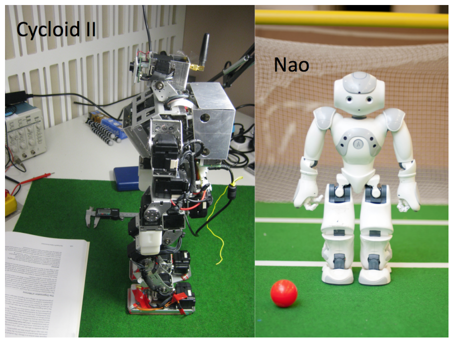

Fig. 8. Cycloid robot from Robotis, Korea modified by Tribotix, Australia with an on-board GEODE processor (left), and the Nao robot from Aldebaran Robotics, France (right).

On the physical robot, the state of the system needs to be estimated from sensor readings. Both robots are equipped with foot-sensors (4 per foot), an IMU unit providing accelerometer and gyroscope measurements, and encoders allowing motor positions to be read.

State estimation is achieved with a recursive Bayesian filter. We choose a steady-state Kalman filter to reduce the amount of on-line computation and perform recursive updates to estimate the state variables x, $\dot{x}$, $\ddot{x}$, w, t. The filter performs

- a prediction update using the linear inverted pendulum model equations from the simulator with the pivot point estimated from centre-of-pressure (CoP) calculations using the foot-sensor readings.
- a correction update based on kinematic, IMU, and foot-sensor observations. The constant gain matrix used for updating variables x, $\dot{x}$, $\ddot{x}$, w, t is $[0.5\ 0.5\ 0.5\ 0\ 0]$.

A. Centre of Pressure for Prediction Update

The CoP of the support foot is measured in millimetres (mm) with the origin under the ankle joint. The CoP p is calculated by taking the weighted average of all the foot-sensor readings of the support foot.

$$p(foot) = \frac{\sum d_i * f_i}{\sum f_i} \tag{8}$$

where $foot \in \{L(left), R(ight)\}$, d_i is the horizontal distance from the ankle joint to each foot-sensor i, and f_i is foot-sensor i reading.

The CoP is used as the pivot point of the inverted pendulum for the process update and only has meaning during a single-support phase of the walk cycle, since at other times both feet are touching the ground and the dynamics of the robot are affected by both feet.

We define the fraction of time during the walk-cycle that each foot is in the swing phase as $movFrac = 0.4$. This means the total double support time $D = T(1-2*movFrac)$. Over the whole walk cycle T, the two swing phases are from $t = D/4$ to $t = T/2 - D/4$ and $T/2 + D/4$ to $T - D/4$.

The CoP calculations for the left (pR) and right (pL) support feet are:

$$
\begin{aligned}
pR &= \begin{cases} p(R) & D/4 \leq t < T/2 - D/4 \\ 0 & otherwise \end{cases} \\
pL &= \begin{cases} p(L) & T/2 + D/4 \leq t < T - D/4 \\ 0 & otherwise \end{cases}
\end{aligned}
\tag{9}
$$

The support foot is determined by the sign of the *coronal* (or frontal) CoP component. The coronal gait rocking motion on the physical robot is induced by a form of bang-bang control by switching support foot based on the zero-crossing point of the coronal CoP measured across the support polygon of both feet with the origin between the feet. The period T has been tuned to the natural rocking frequency of the humanoid. Coronal disturbances are corrected by adjusting the update of the gait time t to coincide with the zero-crossing of the coronal CoP to times $t = 0$ and $t = T/2$.

To verify that that state-estimation is producing sensible results we compare the estimate of x to a measure we observe with high reliability and take to be the ground-truth. We place two markers on the physical robot, one at the centre of mass and one at the middle of the support foot. These markers are tracked visually at 30 frames per second and the visual x value compared to that estimated on the robot. Figure 9 shows both the visual and estimated value of x over time for the Cycloid with the robot being prodded occasionally.

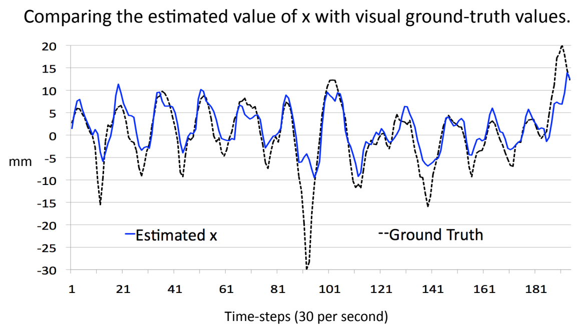

Fig. 9. Cycloid estimated x value plotted over time against the x value observed visually by tracking markers.

B. Control

Once policies for staying upright on the spot while marking time, walking forward and backward are learnt, the policy $\pi : S(x, \dot{x}, w, t) \rightarrow A(c, d)$ is transferred to the physical robot as a table and the latest estimated state is used to lookup the optimal actions to change the CoP on the support foot while moving the swing foot.

The ankle control CoP policy is in the form of a discrete output $c \in \{-1, 0, 1\}$ with the intended meaning to move the CoP of the support foot:

-1	to the back of the foot
0	to the centre of the foot
1	to the front of the foot

On the physical robot, depending on the current inclination of the foot to the floor, rotating the ankle may not have the desired effect. For example if the robot is leaning forward with the heel slightly off the ground, rotating the ankle to put more pressure on the toe is a redundant action, and rotating the ankle pitch slightly to put pressure on the heel may not have any effect. Our approach is to use the sensed current CoP point to adjust the control action in such a way that it has a better chance of being effective. We discretise the CoP p (pR or pL) calculated above in Equation 9, by three values $[-1, 0, 1]$ to represent the ranges: close to the back of the foot; between back and front of the foot; and close to the front of the foot, respectively.

With a small ankle-pitch movement Δf, we implement the control policy a as follows:

```
If $D/4 \leq t < T/2 - D/4$ or $T/2 + D/4 \leq t < T - D/4$:
  if $c = -1$
    if ($p = -1$) no-change
    if ($p = 0$) $a = \Delta f$
    if ($p = 1$) $a = 2\Delta f$
  if $c = 0$
    if ($p = -1$) $a = -\Delta f$
    if ($p = 0$) no-change
    if ($p = 1$) $a = \Delta f$
  if $c = 1$
    if ($p = -1$) $a = -2\Delta f$
    if ($p = 0$) $a = -\Delta f$
    if ($p = 1$) no-change
```

If the control action is already in force there is no change. At the other extreme, if the current CoP is at the other end of the foot we accelerate the ankle control action by moving it through twice the usual rotation. The position of the swing foot w was hard-coded as function of x, typically $w = x/1.2$, to account for some energy loss during the gait cycle.

VI. Experimental Results

We reproduce simulator experiments that mimic impulse forces, and walking backward and forward behaviour using the reinforcement learnt policies.

Impulse forces were created by placing a step on the ground in the path of the robot. When the robot stepped on this obstacle with its toe, the robot was pushed backward. The robot would loose balance and fall over depending on the height of the step. Figure 10 shows experimental results for varying step-heights with and without control. With controller active the robot can reliably survive an extra 72% increase in step height. Similar trends and results are seen in simulation.

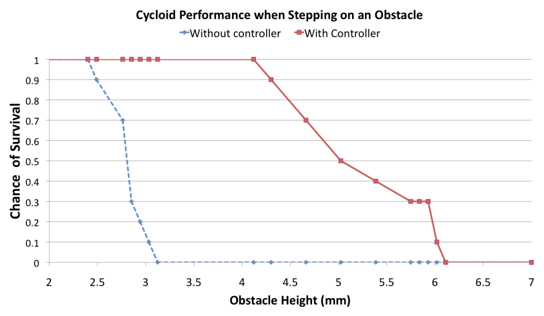

Fig. 10. Probability of falling with and without controller after stepping on various height obstacles.

Experiments also show that a robot with the controller active can survive walking into an object such as a chair or low bar, but fall over otherwise[3].

A shortcoming of our work is the movement of the swing leg. We have found this to be jerky, especially for large disturbances where the leg movement cannot respond quickly enough to demanded positions.

VII. Related Work

The literature on bipedal walking is extensive with several approaches using reinforcement learning, for example: temporally extended actions have been applied to swing foot placement underpinned by semi-MDP theory [3]. The parameters of a central pattern generator are learnt using a policy gradient method [5], and a CMAC function approximation is used in learning the parameters of a swing-leg policy [6]. Bipedal walking with point feet precludes ankle control and restricts control actions to the double support phase or the points in time when placing the swing foot.

Graf and Röfer [7] control a flat footed biped using an iterative analytically method based on an inverted pendulum model to plan the placement of the swing leg. The ankle joint is not directly actuated as a control variable, but by keeping the foot horizontal to the ground, there is an implicit control to counteract unplanned movements.

Tracking and control of the CoM and Zero Moment Point (ZMP) using modern control theory is employed for the HRP series of robots [8] [9]. The approach uses *preview* control, a feedforward mechanism that plans ahead using the anticipated target ZMP. These robots control both the body posture including ankle control and foot placement to stabilise the robot. Our RL technique also provides feedforward control as optimal actions are learnt to maximise future reward.

VIII. Future Work

We plan to extend the control to include coronal motion, again though both ankle and foot placement control. Not only is this expected to lead to omni-directional reactive behaviour, but it should improve the performance because both sagittal and coronal motions can be jointly optimised. The state space is expected to grow and we have a barrage of reinforcement learning techniques at our disposal to mitigate the expected scaling issue, such as more cost-effective function approximation using instance based methods [10] and hierarchical techniques [11].

The inverted pendulum is only an approximation to the physical robot. We propose to improve the policy by improving the simulator's representation of the real robot and by continuing learning on the real robot using the simulated policy as a starting point to speed up convergence.

Beside ankle and foot placement control we intend to include a hip reaction in the sagittal plane to assist balancing.

IX. Conclusions

The paper describes an approach to learn an ankle and foot placement policy to arrest impulse disturbances and to react to rapidly changing bipedal walking goals. The system is specified by including time in the state-space and using RL to optimise the arbitration between the two control actions. In practice with large footed bipeds we find ankle control is sufficient to achieve the objectives. Results transferring learnt simulated inverted pendulum policies to physical robots show promise and suggest several avenues for future work scaling these techniques to 3D bipedal locomotion.

[3]The video accompanying this paper shows various scenarios of this behaviour for both the Nao and Cycloid robots.

References

[1] "Robocup http://www.robocup.org/," Board of Trustees, 2011.

[2] R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*. Cambridge, Massachusetts: MIT Press, 1998.

[3] J. Morimoto, G. Cheng, C. Atkeson, and G. Zeglin, "A simple reinforcement learning algorithm for biped walking," in *Robotics and Automation, 2004. Proceedings. ICRA '04. 2004 IEEE International Conference on*, vol. 3, april-1 may 2004, pp. 3030 – 3035 Vol.3.

[4] E. R. Westervelt, J. W. Grizzle, C. Chevallereau, J. H. Choi, and B. Morris, *Feedback Control of Dynamic Bipedal Robot Locomotion*. Boca Raton: CRC Press, 2007, vol. 1.

[5] T. Mori, Y. Nakamura, M.-A. Sato, and S. Ishii, "Reinforcement learning for a cpg-driven biped robot," in *Proceedings of the 19th national conference on Artifical intelligence*, ser. AAAI'04. AAAI Press, 2004, pp. 623–630. [Online]. Available: http://portal.acm.org/citation.cfm?id=1597148.1597249

[6] C.-M. Chew and G. A. Pratt, "Dynamic bipedal walking assisted by learning," *Robotica*, vol. 20, pp. 477–491, September 2002. [Online]. Available: http://portal.acm.org/citation.cfm?id=1011547.1011549

[7] C. Graf and T. Röfer, "A closed-loop 3d-lipm gait for the robocup standard platform league humanoid," in *Proceedings of the Fourth Workshop on Humanoid Soccer Robots in conjunction with the 2010 IEEE-RAS International Conference on Humanoid Robots*, C. Zhou, E. Pagello, S. Behnke, E. Menegatti, T. Röfer, and P. Stone, Eds., 2010.

[8] S. Kajita, F. Kanehiro, K. Kaneko, K. Fujiwara, and K. H. K. Yokoi, "Biped walking pattern generation by using preview control of zero-moment point," in *in Proceedings of the IEEE International Conference on Robotics and Automation*, 2003, pp. 1620–1626.

[9] S. Kajita, M. Morisawa, K. Miura, S. Nakaoka, K. Harada, K. Kaneko, F. Kanehiro, and K. Yokoi, "Biped walking stabilization based on linear inverted pendulum tracking," *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2010)*, pp. 4489–4496, 2010.

[10] J. C. Santamaria, R. S. Sutton, and A. Ram, "Experiments with reinforcement learning in problems with continuous state and action spaces," *Adaptive Behavior, 6(2)*, 1998. [Online]. Available: citeseer.nj.nec.com/173817.html

[11] B. Hengst, "Model approximation for HEXQ hierarchical reinforcement learning," in *ECML*, 2004, pp. 144–155.