rUNSWift Is No Pushover

rUNSWift Open Challenge Entry
School of Computer Science and Engineering, University of New South Wales, Sydney, Australia

***Abstract*—We learn a controller for the Nao robot using reinforcement learning to optimally respond to external disturbances induced by stepping on toes or being pushed. The reinforcement learning method employed learns an optimal policy for ankle tilt control to assert pressure along the support foot to keep the Nao balanced. The controller is learnt in simulation using an inverted pendulum model and the policy is transferred to the robot.**

I. Introduction

In local and international RoboCup SPL (and Humanoid) competitions, it is not uncommon to see robots fall over while tussling for the ball. The two major reasons that robots fall over are that they are either pushed by other players or they step on the foot of another player. While falling robots provide significant entertainment value for spectators, behaviours that are resistant to falling would have a distinct advantage.

We tackle this challenge using reinforcement learning (RL) to apply ankle torques to the support foot to control the centre of pressure in an effort to keep a robot balanced. The policy is highly reactive because it can be applied while in mid-stride - we do not have to wait for the next swing foot placement.

II. Simulation

We model the flat-footed humanoid as an inverted pendulum (IP) 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.

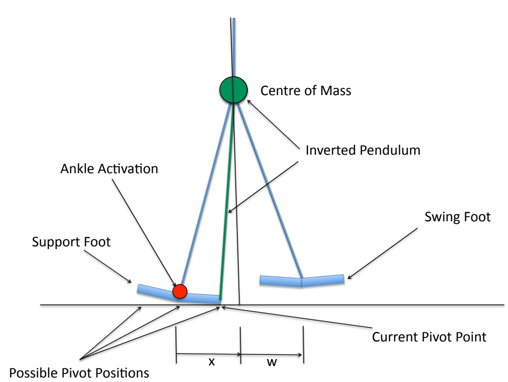

Fig. 1. The inverted pendulum model of a flat-footed bipedal robot used for simulation and reinforcement learning.

The state of the system is defined by three variables $(x, \dot{x}, t)$ 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, and t is the time-step from the start of each walk-cycle. The control actions choose the centre of pressure for the support foot relative to the foot centre. We have adapted Q-Learning to work with the near-Markov system induced by the function approximator.

Figure 2 shows the simulation monitor with the robot in plan and elevation view, and the x and $\dot{x}$ response against time, both on a coordinate system and as an unfolding time series.

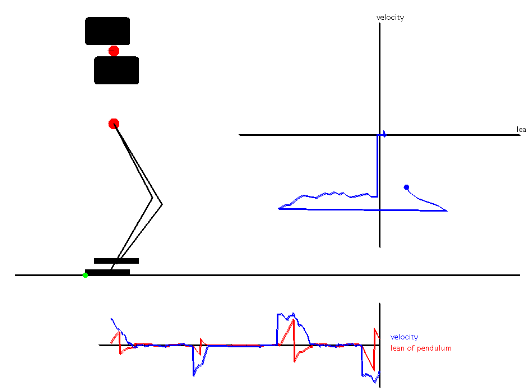

Fig. 2. Simulated robot in plan and elevation view (left). Time-trace on a coordinate system with horizontal axis x and vertical axis $\dot{x}$ (top-right). Time series for x (red) and $\dot{x}$ blue with the current time on the right showing response to impulse forces (bottom).

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.

III. Physical Robot Implementation

The policy learnt on the simulator is transferred to the Nao. On the physical robot, the state of the system needs to be estimated. We choose a steady-state Kalman filter to reduce the amount of on-line computation. We perform recursive process updates using the IP model and correction updates from sensor readings.

For the RoboCup 2011 SPL Open Challenge we plan to demonstrate the Nao staying upright when being pushed or when it steps on a toe-like obstacle.

References

[1] "rUNSWift home page http://cgi.cse.unsw.edu.au/ robocup/2010site/."
[2] B. Hengst, M. Lange, and B. White, "Learning to control a biped with feet," *In Preparation*, 2011.