CS 535 Deep Learning Project:

Learning Locomotion Behaviors using Deep Reinforcement Learning

Yathartha Tuladhar

03/20/2018

Project Outline

- Current approaches and its limitations
- Reinforcement Learning (problem setup, algorithm, reward shaping)
- Results (training a robot to stand)

Hand Designed Controllers: What is the problem?

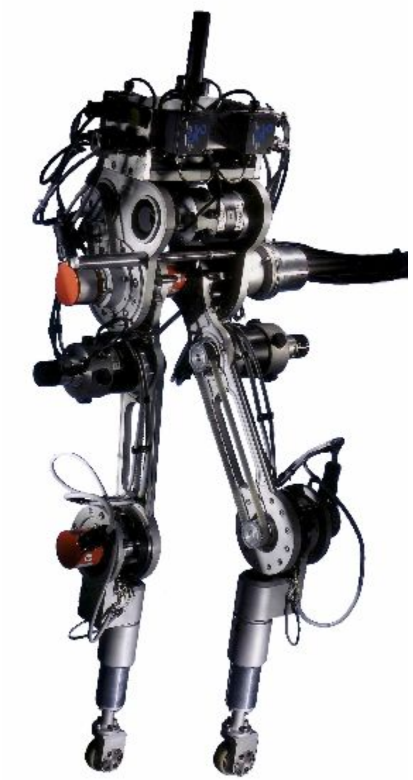


Figure 1a: Bipedal Robot "rabbit"

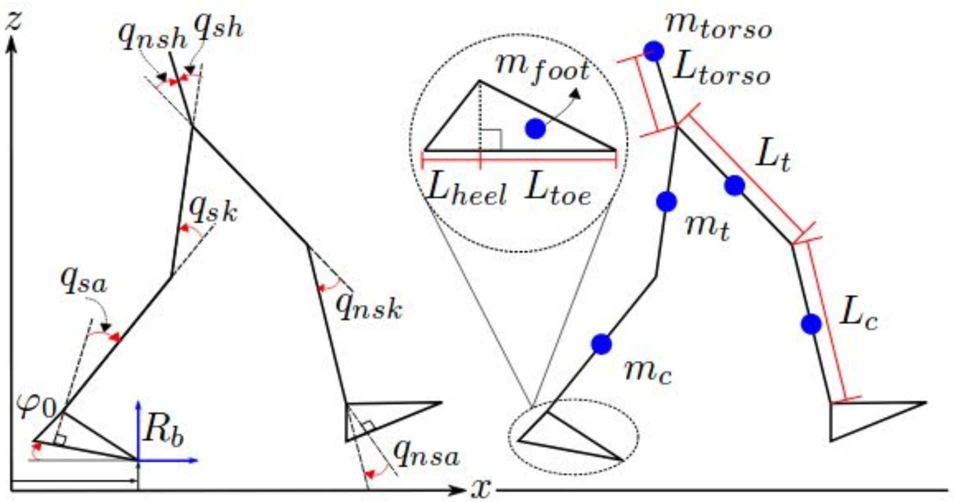


Figure 1b: Coordinates of 9 degree of freedom footed biped

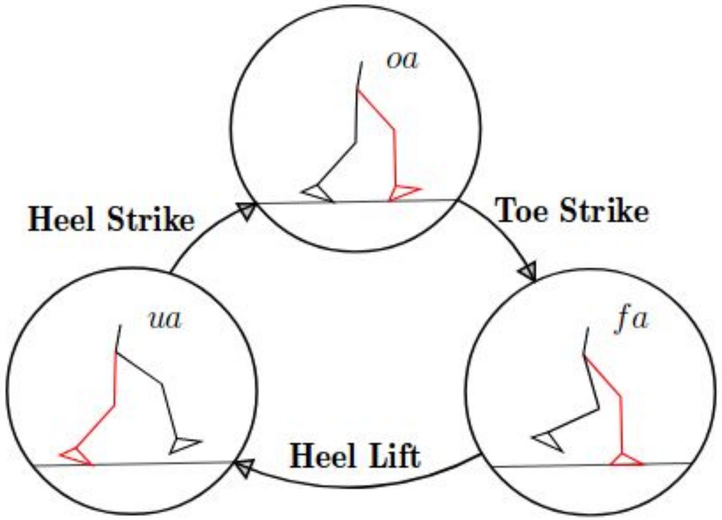


Figure 1c: Directed graph associated with 3 domain walking. The stance leg is red and non-stance leg is black.

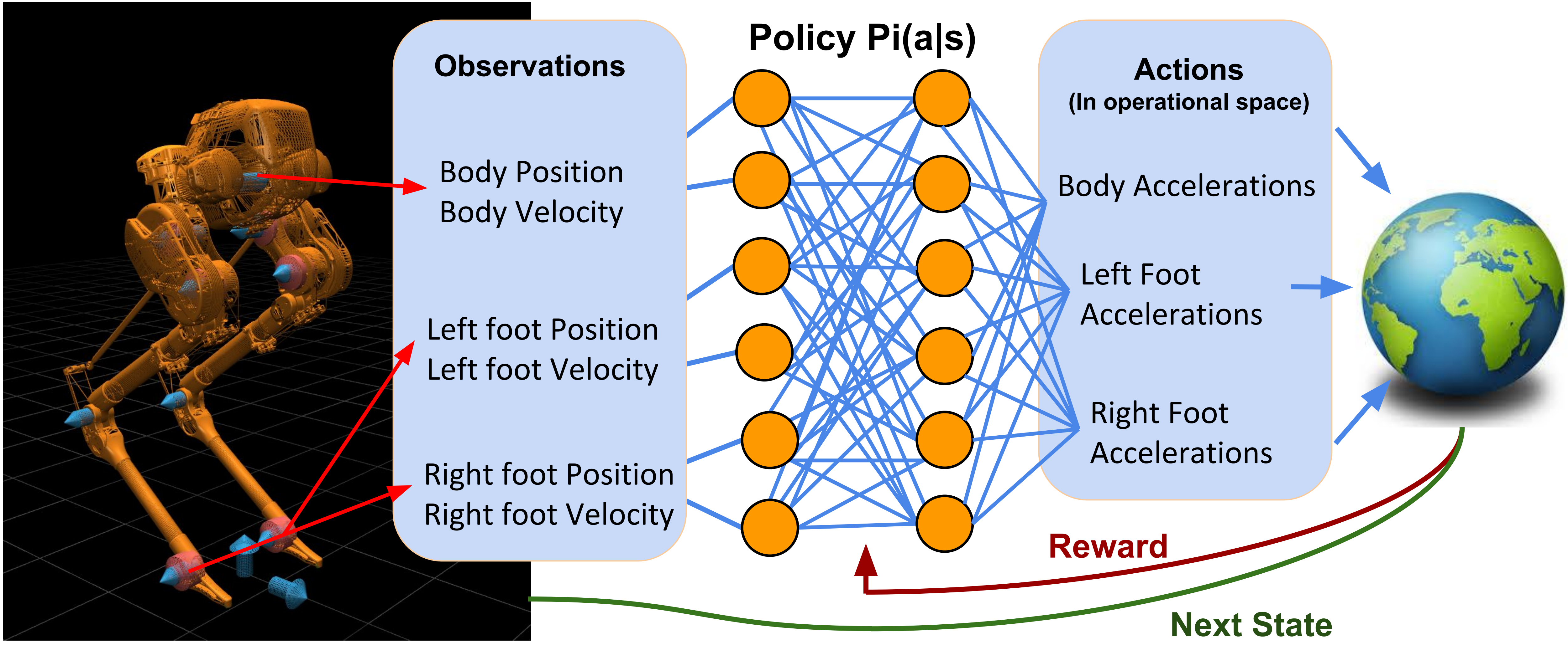
- Designing controllers by hand for high degree-of-freedom (DOF) systems can be very hard
- Many behaviors cannot be written down with a set of rules
- Most methods will only be statically stable

"Planar Multi-Contact Bipedal Walking Using Hybrid Zero Dynamics", ICRA'14
"Rapidly Exponentially Stabilizing Control Lyapunov Functions and Hybrid Zero Dynamic", Transactions of Automatic Control

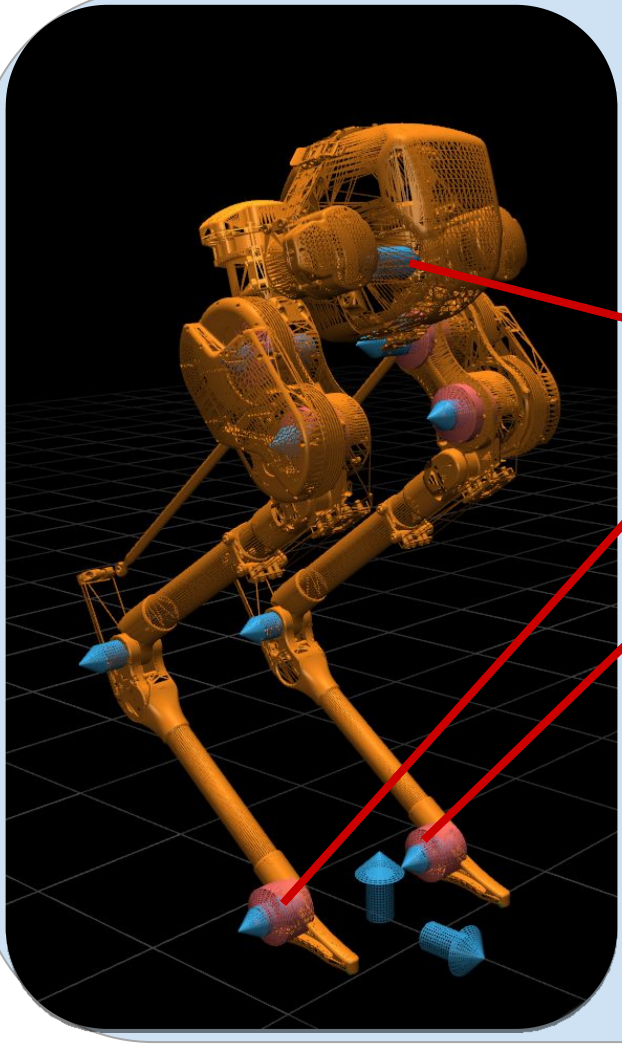
Hand Designed Controllers: What is the problem?

- Dynamics of robots with high DOFs can be very hard to model
- We want to develop better controllers for legged locomotion that are capable of rich dynamic behaviors
- Deep reinforcement learning can be used to learn control policy parameters or the policy itself

Reinforcement Learning in Operational Space



Reinforcement Learning: Action Spaces



Operational Space

Body (x, z, tau) accelerations

Right foot (x, z, tau) accelerations

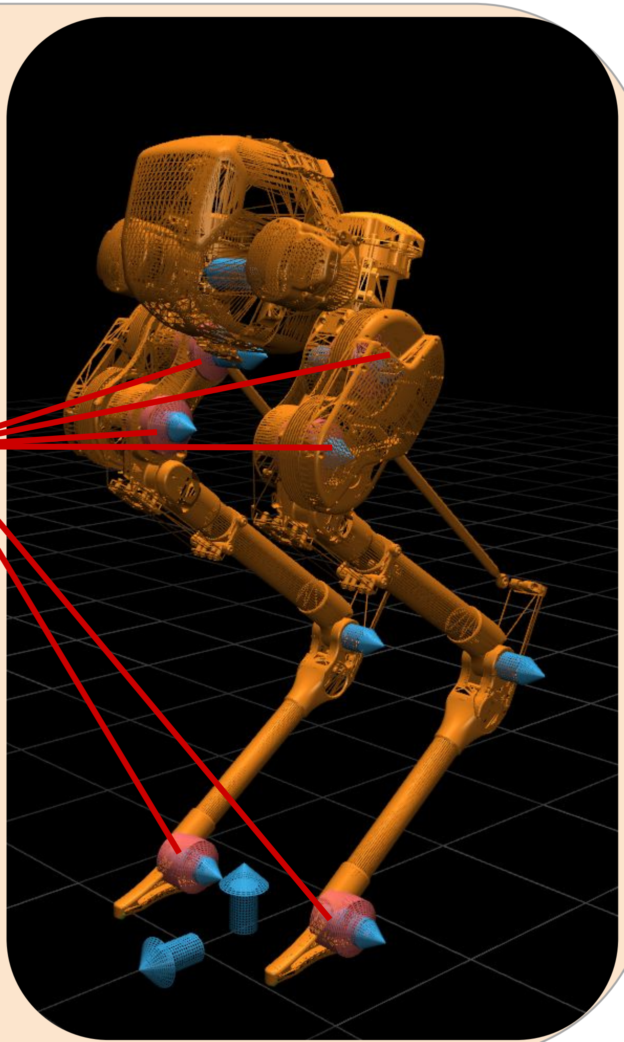
Left foot (x, z, tau) accelerations

The required motor torques were calculated using an operational space controller. The accelerations were bounded

Torque Space

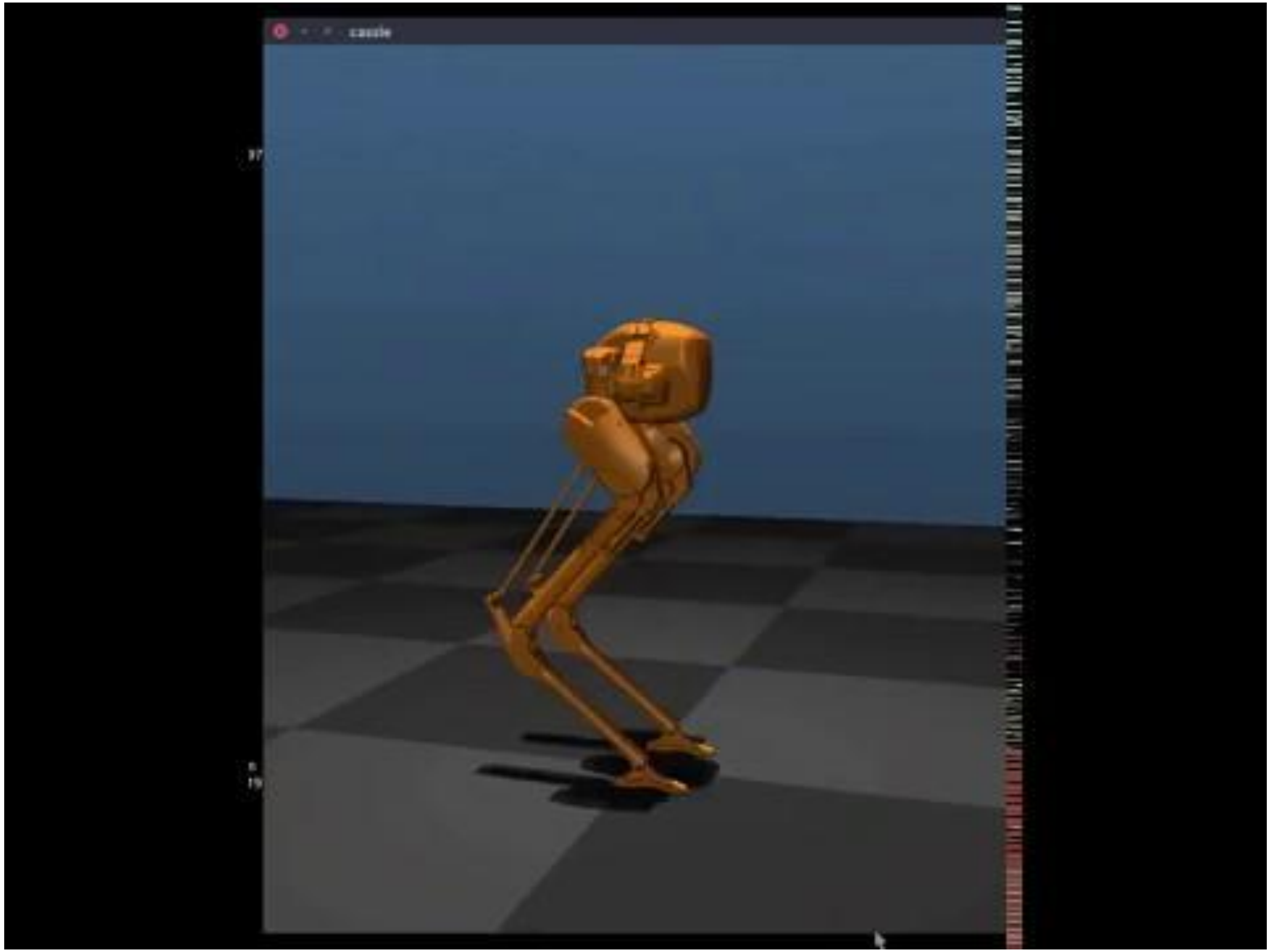
Torques for all individual actuators

The torques were bounded based on the capabilities of the real hardware



RL training example: Input - Policy - Output

- Observations:
  - Joint positions
  - Joint velocities
- Policy is a neural network that has two hidden layers with 32 units each
- Policy Output is a gaussian of actions:
  - OSC: body and foot accelerations
  - Torque: Individual torques



Improving Policy using Policy Gradients

REINFORCE algorithm: *(Vanilla Policy Gradient)*

1. sample $\{\tau^i\}$ from $\pi_\theta(\mathbf{a}_t|\mathbf{s}_t)$ (run it on the robot)
2. $\nabla_\theta J(\theta) \approx \sum_i \left(\sum_t \nabla_\theta \log \pi_\theta(\mathbf{a}_t^i|\mathbf{s}_t^i)\right)\left(\sum_t r(\mathbf{s}_t^i, \mathbf{a}_t^i)\right)$
3. $\theta \leftarrow \theta + \alpha \nabla_\theta J(\theta)$

Pseudocode for Trust Region Policy Optimization

for iteration=1, 2, . . . **do**

  Run policy for T timesteps or N trajectories

  Estimate advantage function at all timesteps

$$\underset{\theta}{\text{maximize}} \sum_{n=1}^{N} \frac{\pi_\theta(a_n \mid s_n)}{\pi_{\theta_{\text{old}}}(a_n \mid s_n)} \hat{A}_n$$

$$\text{subject to} \quad \overline{\mathrm{KL}}_{\pi_{\theta_{\text{old}}}}(\pi_\theta) \le \delta$$

end for

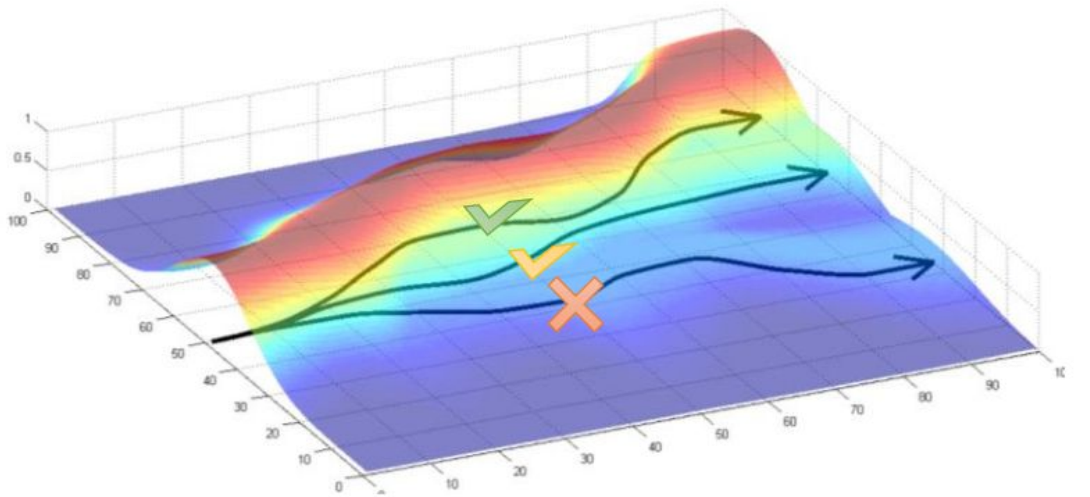


Figure 2: The policy update shifts the mean and variance of a gaussian policy based on rewards from rollouts. It makes the good actions more likely, and bad actions less likely.

Image from Sergey Levine, CS294 Deep Reinforcement Learning Fall 2017
Pseudocode from John Schulman, Deep RL Bootcamp 2017

Reward Function: training Cassie to stand

Reward Shaping Example

For every rollout:

- ➔ Initialize Reward, R = 0
- ➔ R -= 5 * (target_height - body_height)**2
- ➔ R -= 5 * (left_foot_position)**2
- ➔ R -= 5 * (right_foot_position)**2
- ➔ R -= 0.01*sum(action**2)
- ➔ R += 1

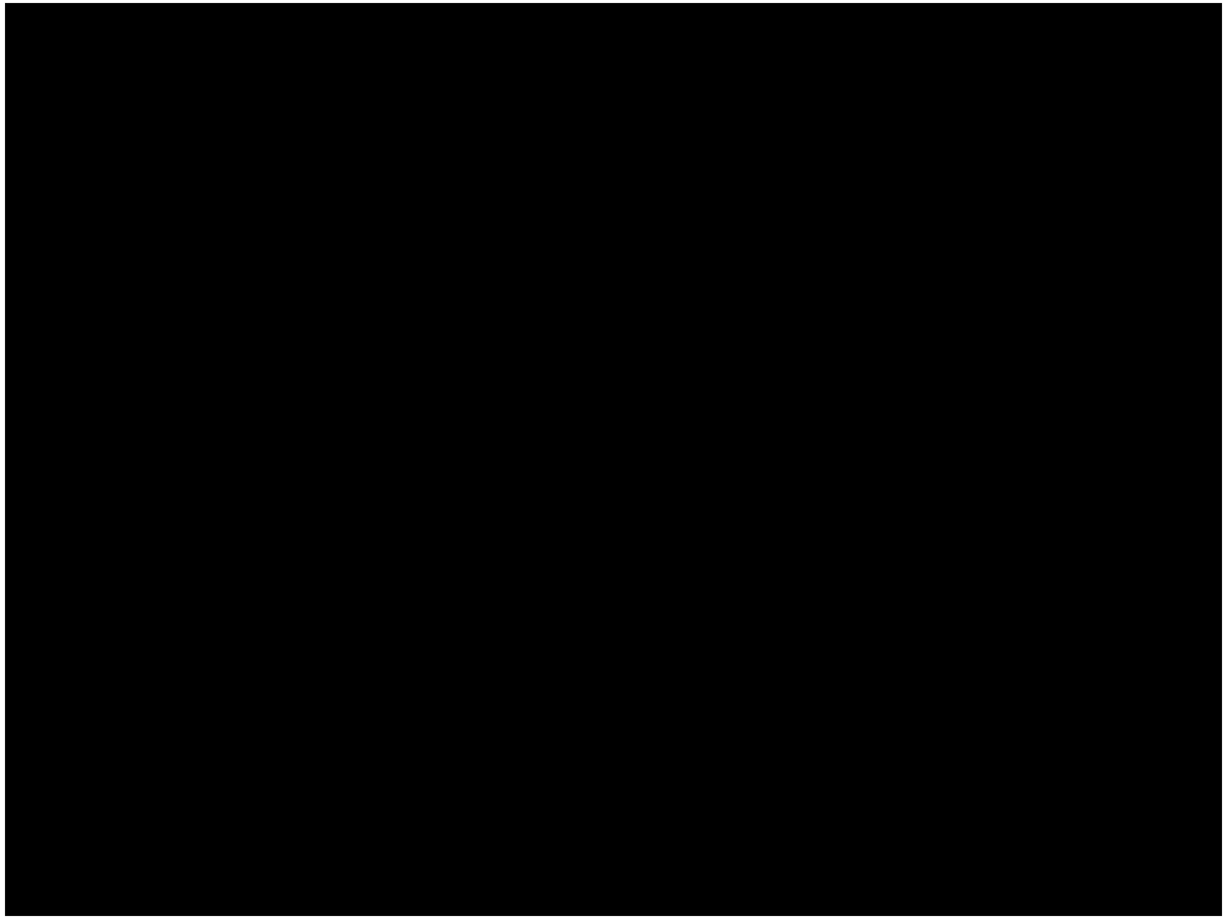
Note: *foot positions are with respect to the body's center of mass (CoM) position*

Desired Behavior

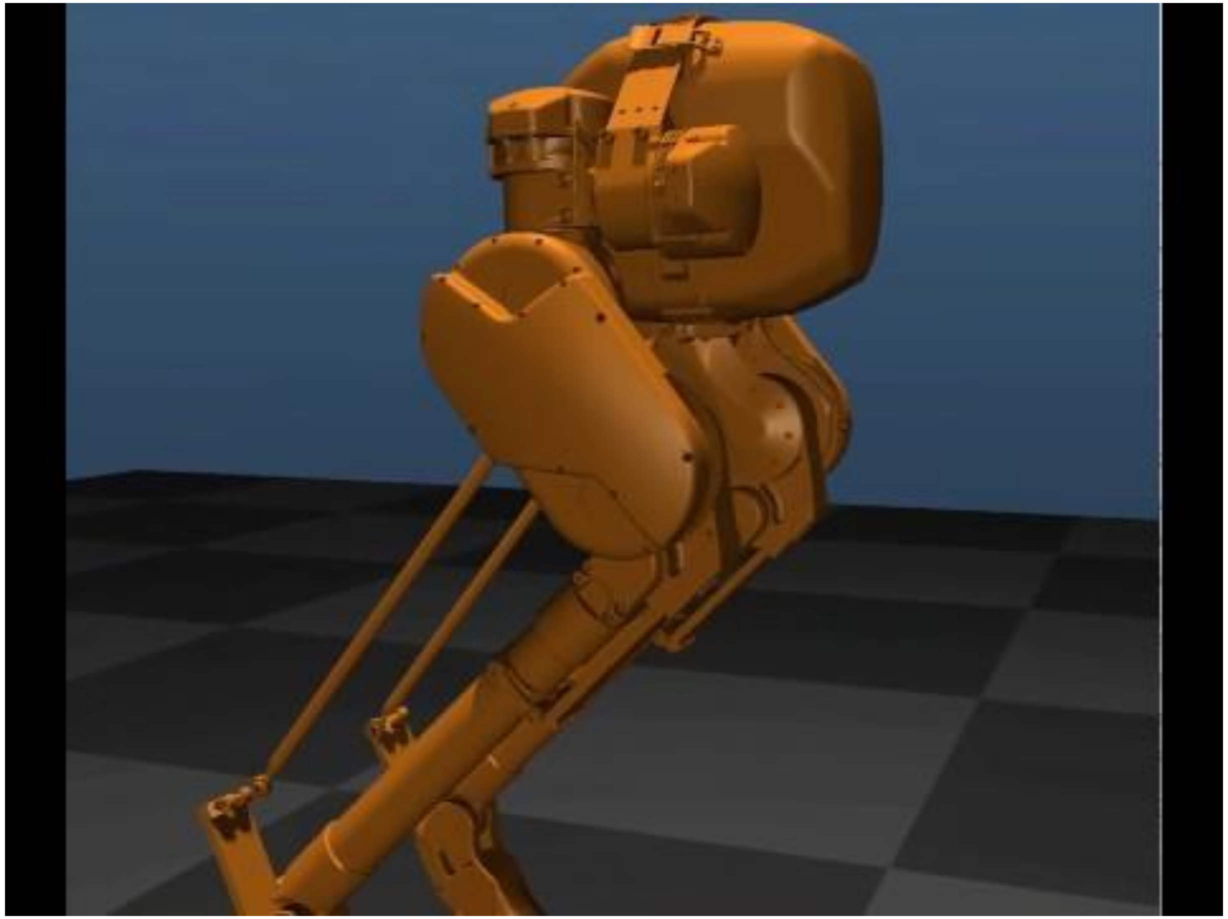
- Maintain a desired height
- Place foot underneath body's CoM
- Minimize actions (accelerations or torques)
- Stay alive *(helps when initially learning to stand)*

Results with Operational Space Control

Learns to stand some way.



Learns to stand far better after shaping rewards.



Results with Operational Space Control

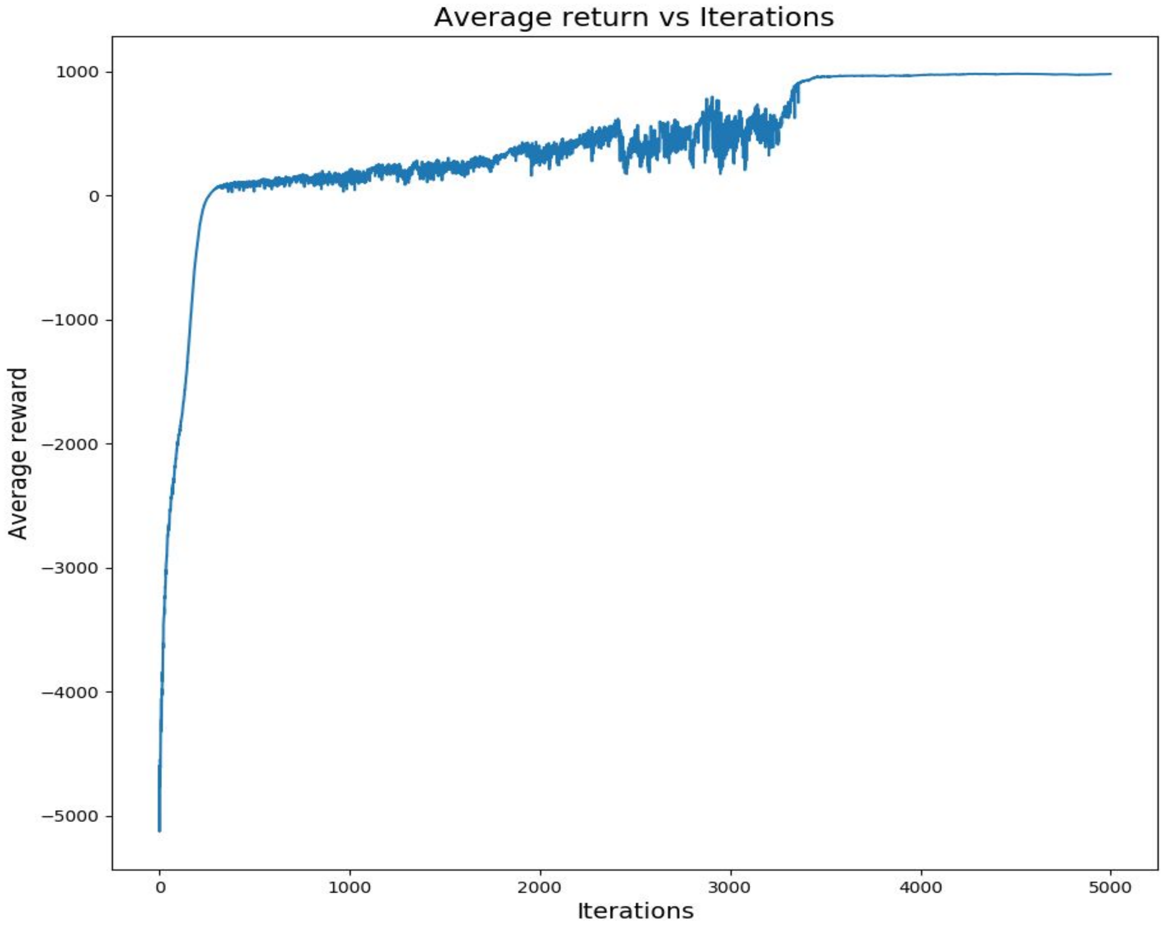
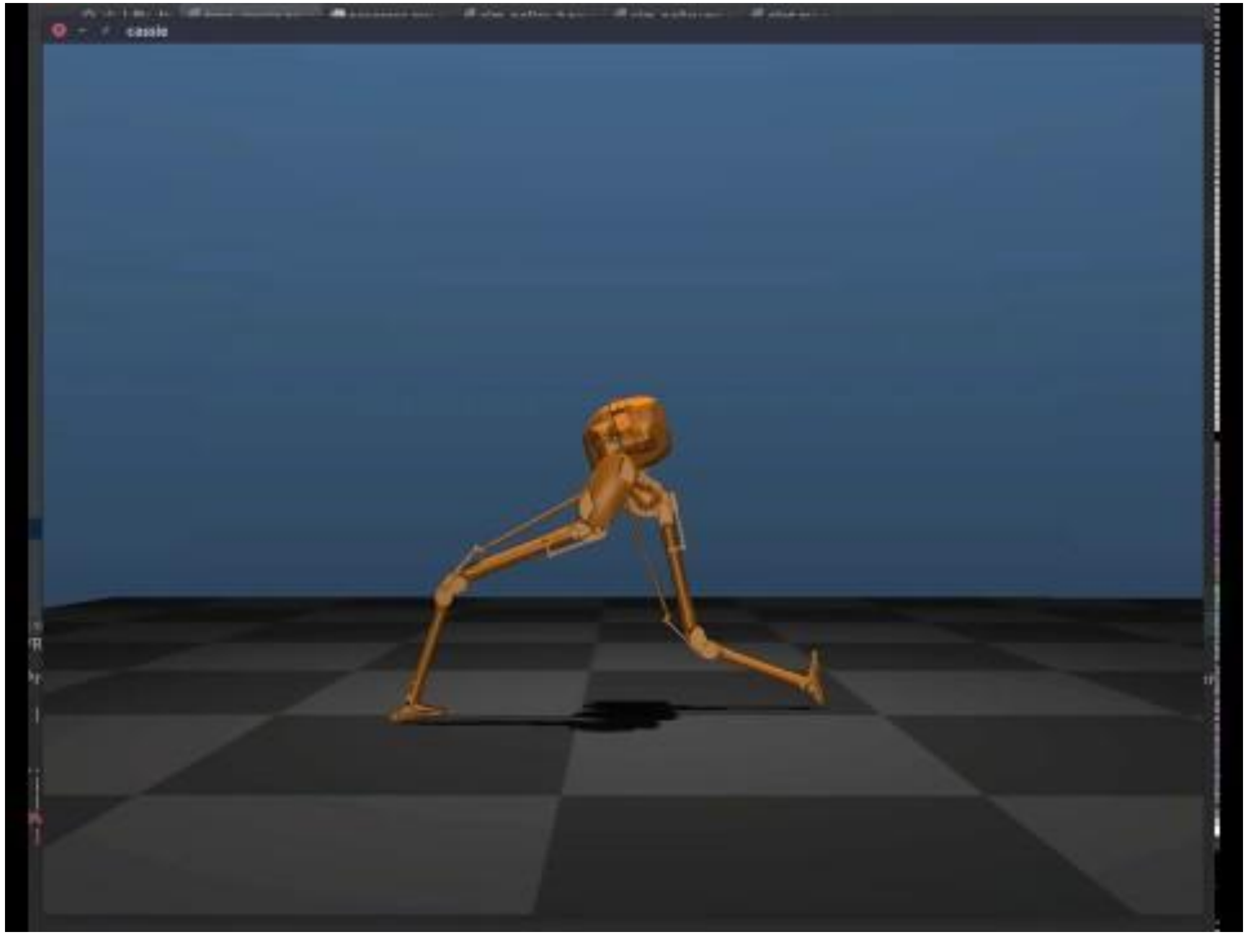


Figure 3. Average return for the “Perfect Standing” policy for operational space control.

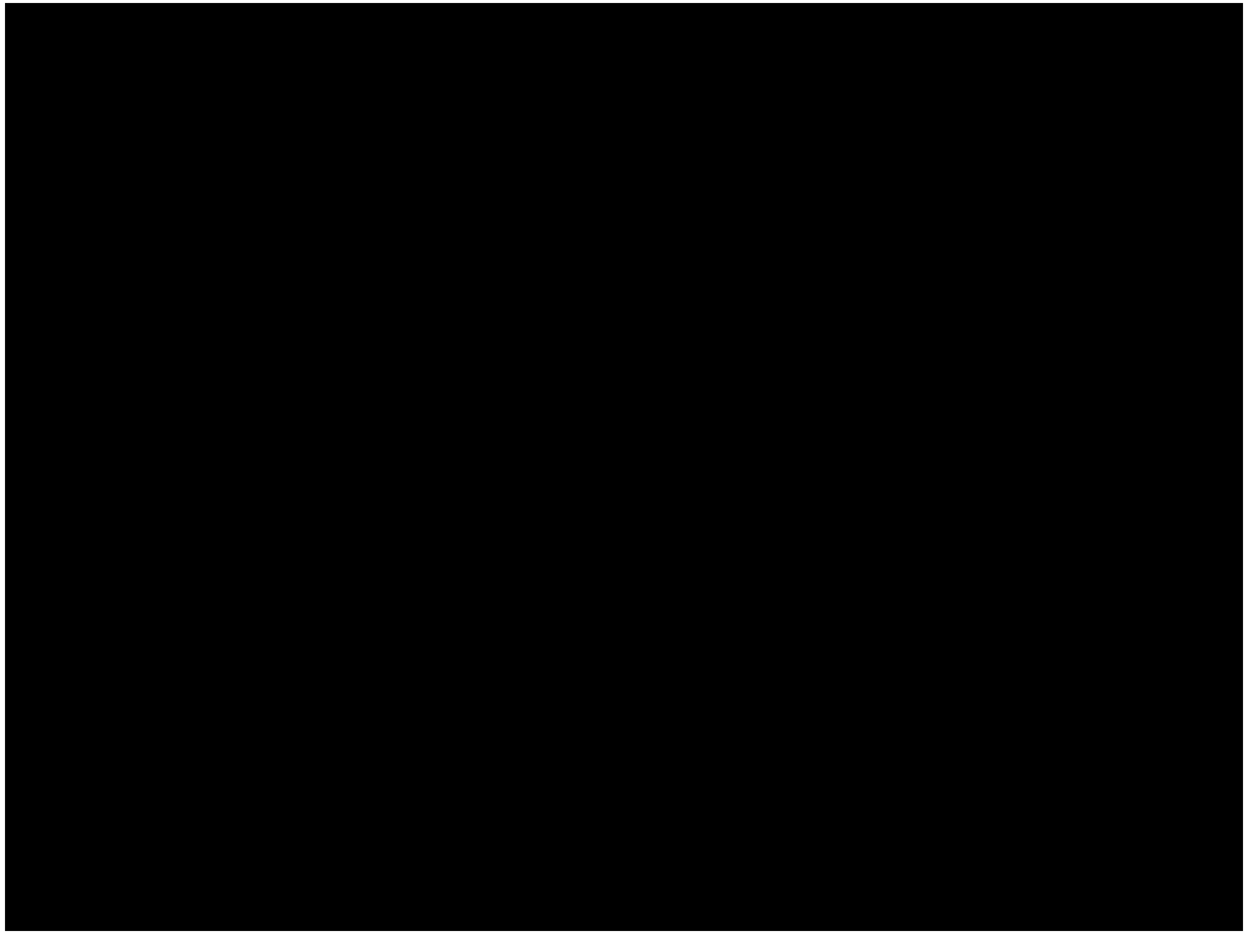
This training time was 2 days, 9 hours, ~41s per iteration, on a 4.20 Ghz i7, single worker

Results with Torque Control

Learns to stand some way, that maximized the reward.



Learns to stand far better after reward shaping.



Questions?