292B Final Project

Handstand Robot

Jason Abi Chebli ◆ Eric Chuang ◆ Andrew Tai ◆ Eric Foss

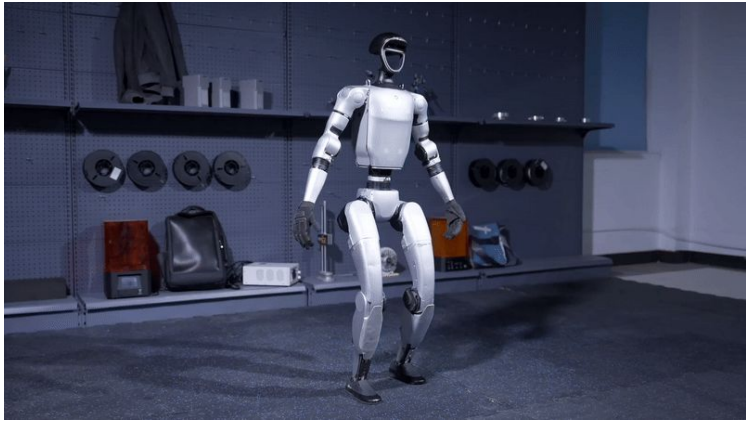
UC Berkeley

Motivation

Real-World Application: Foundation for agile humanoid locomotion (walking, running, jump recovery).

The Goal: Benchmark Model-Based (Classical) vs. Model-Free (RL and IL) approaches for dynamic balance.

[1]



[2]

Problem Statement

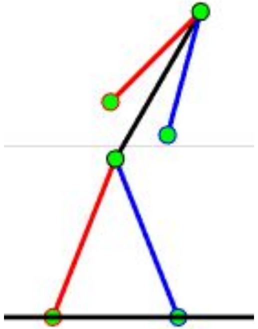
<u>Objective</u>
Start standing normally→ Stabilize in inverted posture

Compare different methods

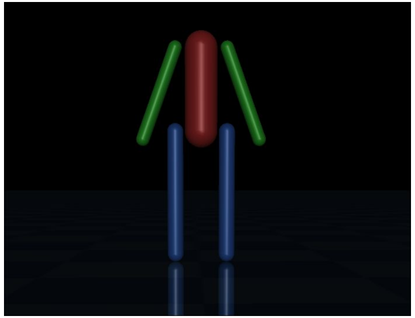
<u>2D System</u>
1 - 5-Link Planar Humanoid (2 legs, torso, 2 arms)
2 - 7-Link Planar Humanoid (2 legs (with knees), torso, 2 arms)

<u>Input</u> Torques at joints (ankles, knees, hips, shoulders).

<u>Constraints</u> Unfixed base (can tip over). Minimize energy consumption.

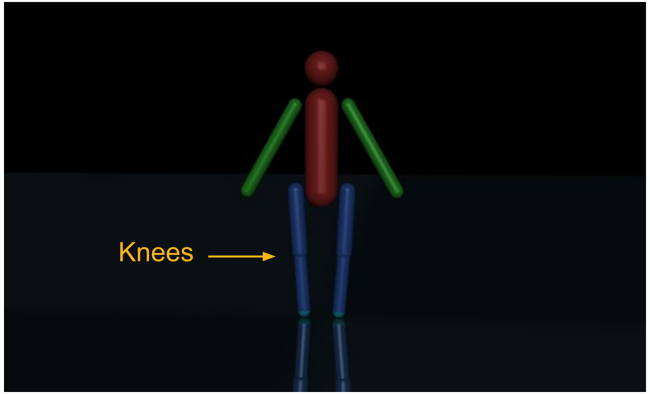


(a) Matlab



(b) Mujoco

5-Link Planar Humanoid



7-Link Planar Humanoid (Mujoco)

Approach

Classical

Reinforcement Learning

Imitation Learning

Approach 1: Classical Control for 5 Link Walker in Matlab

Modeling: Three dynamic phases: walking, planting the first hand, then swinging up to the handstand position.

Control: Drive links to predefined configurations using Input-Output Linearization and finite-time stabilizing control law.

Output Dynamics: 4 control inputs -> 4 outputs to manage 5 links.

Pros: Full understanding of control system -> theoretical guarantees

Cons: Computationally intensive + numerically sensitive -> difficult to generate optimal trajectories

$$y_{walking} = \begin{bmatrix} \theta_5 - \theta_{5,des} \\ \theta_4 - \theta_{4,des} \\ \theta_3 - \theta_{3,des} \\ \theta_2 + \theta_1 \end{bmatrix}, \quad y_{handplant} = \begin{bmatrix} \theta_5 - \theta_{5,des} \\ \theta_4 - \theta_{4,des} \\ \theta_3 - \theta_{3,des} \\ (\theta_2 - \frac{\pi}{2}) - \theta_1 \end{bmatrix}, \quad y_{swingup} = \begin{bmatrix} \theta_5 - \theta_{5,des} \\ \theta_4 + \theta_3 \\ \theta_2 - \theta_{2,des} \\ \theta_1 - \theta_{1,des} \end{bmatrix}$$

3 Phase Output Dynamics

3 Phase Output Dynamics

Approach 2: Reinforcement Learning

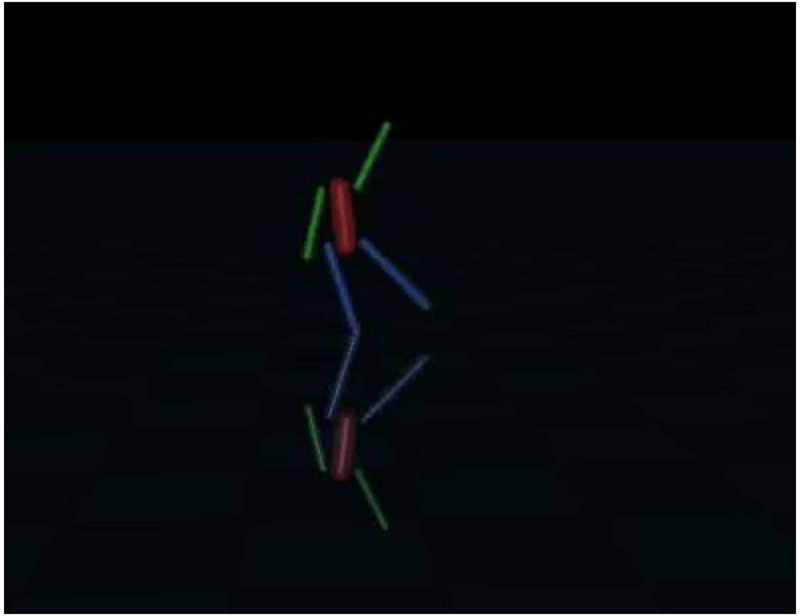
System: 5-Link Walker

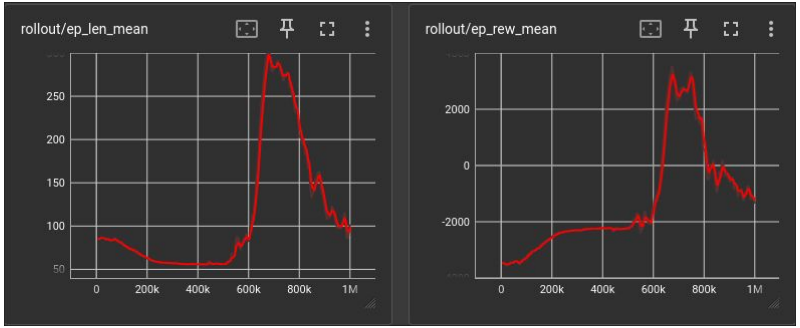
Observation Space: Joint positions, joint velocities, hand/foot sensors

Action Space: Joint torques

Approaches we tried

- Reward forwards angular velocity -> The robot shakes back and forth -> Only reward forwards angular velocity
- Reward angular velocity at the end of state to incentivize optimization prior to end of state
- Staged rewards (different reward scheme for different stages) for stability at the end

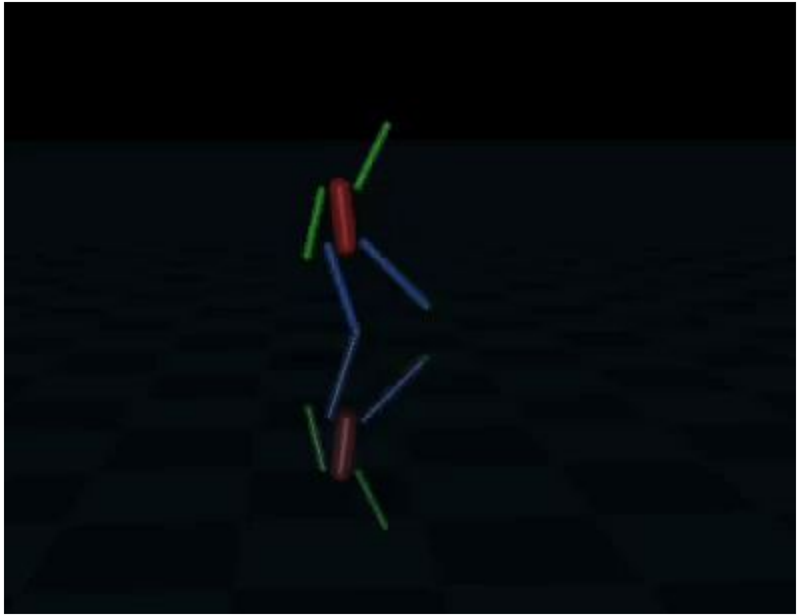


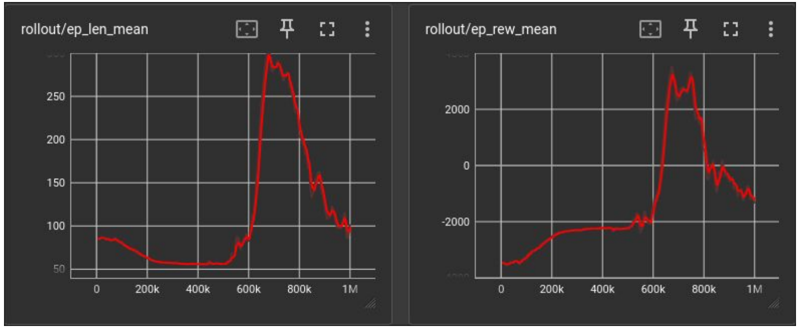


Approach 2: Reinforcement Learning

Reward Function:

- Large reward for holding the desired handstand state (both hands on ground, both feet in air)
- Rewards for getting torso inverted and hands contacting ground in order to guide initial learning
- Penalties for undesired states such as feet touching ground or torso hitting ground



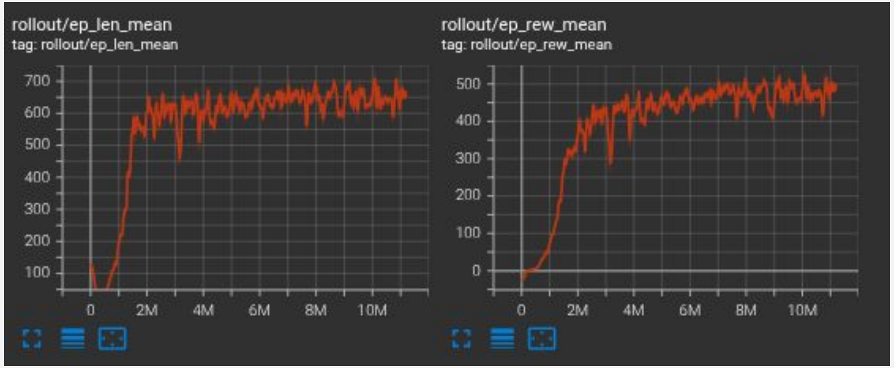


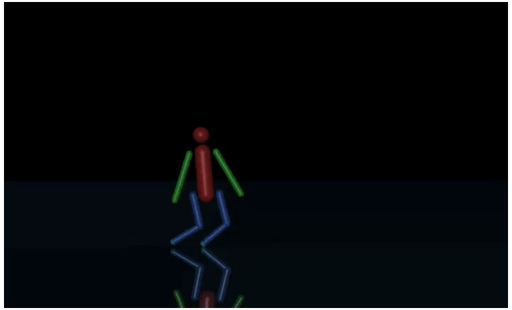
Approach 3: Imitation Learning

System: 7-Link Walker

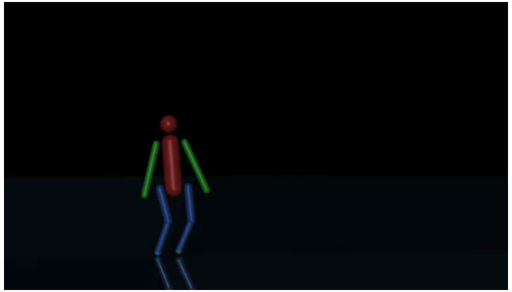
Teacher: Used the **Classical Controller** to generate expert trajectories.

Process: Neural Network trained to map *State* → *Expert Action*.

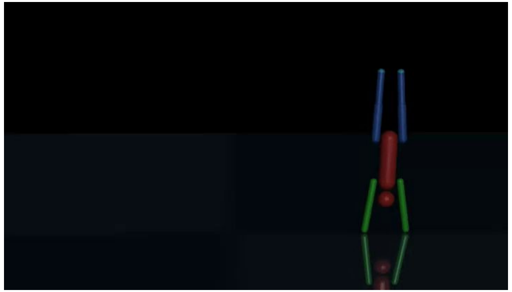




Trajectory (no Physics)



Trajectory (Physics)



Imitation Learning

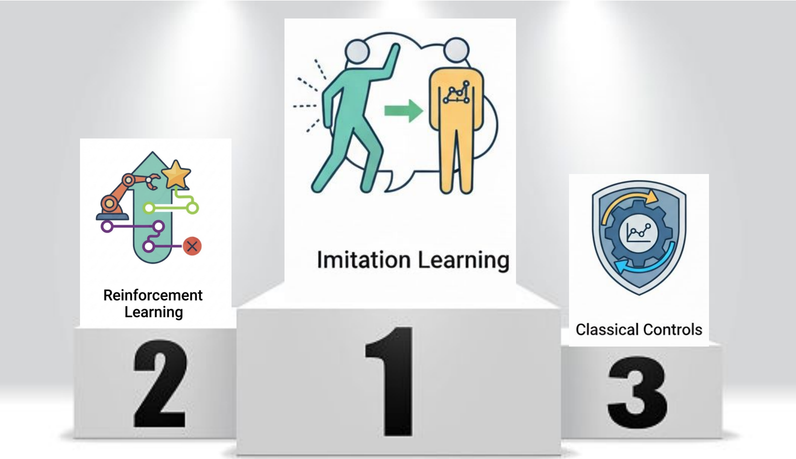
Final Remarks

Performance Comparison

- **Robustness:**
  - **RL:** Superior recovery from large disturbances.
  - **Classical/Imitation:** Struggle with stability under stress; Classical is the most brittle.

- **Smoothness:**
  - **Classical Control:** Produced the most fluid, natural motion.
  - **RL & Imitation:** Prone to high-frequency jitter and excessive torque usage.

- **Ease of Implementation:**
  - **Imitation Learning:** Easiest to deploy successfully.
  - **RL:** Moderate difficulty.
  - **Classical Control:** Hardest (requires precise tuning).



Ease of Implementation Award

Thank you

References

[1] https://www.youtube.com/watch?v=b240jAK-QSM

[2] https://www.youtube.com/watch?v=29xLWhqME2Q