Legged Robots Practical: Project 2

29.10.2024

Plan

Single Leg

- **W1-3** (10.09, 17.09, 24.09)
  - Introduction + double pendulum kinematics and dynamics
  - Jacobian (Cartesian PD + Force Control)
  - Inverse Kinematics (compare with force control)
  - Single-leg hopping

Biped

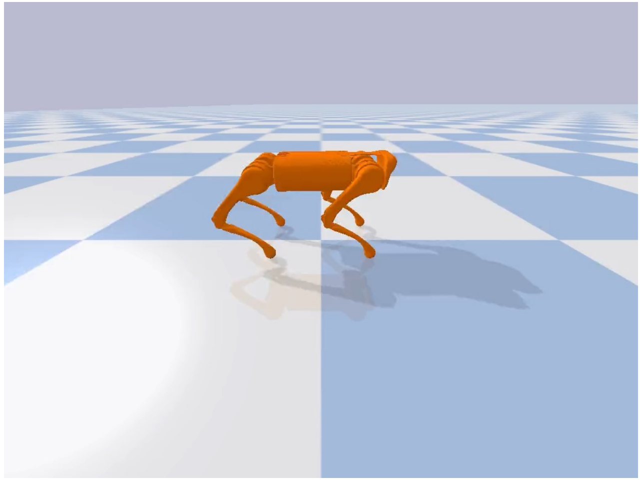
- **W4-6** (01.10, 08.10, 15.10)
  - Biped Walking with Divergent Component of Motion (BD Atlas)
    [Biped Report – 15% of grade]

Quadruped

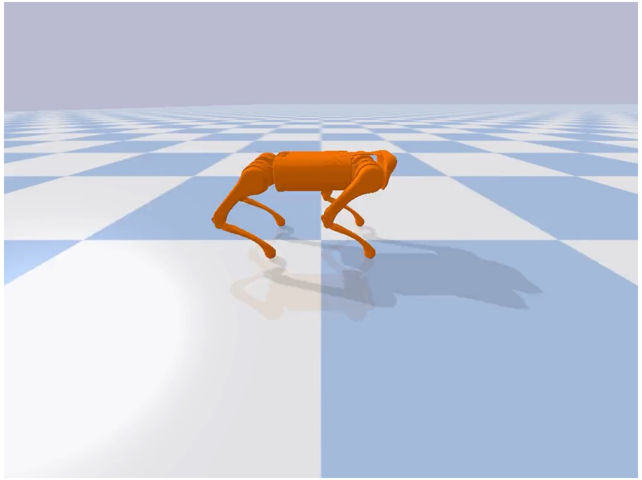
- **W7-14** (29.10-17.12)
  - Quadruped CPG Trot
  - Quadruped Locomotion Project (CPGs, Deep RL)
    [Quadruped Report – 35% of grade]
    [COMPETITION/DEMO 17.12.2024]

Part 1: Central Pattern Generators

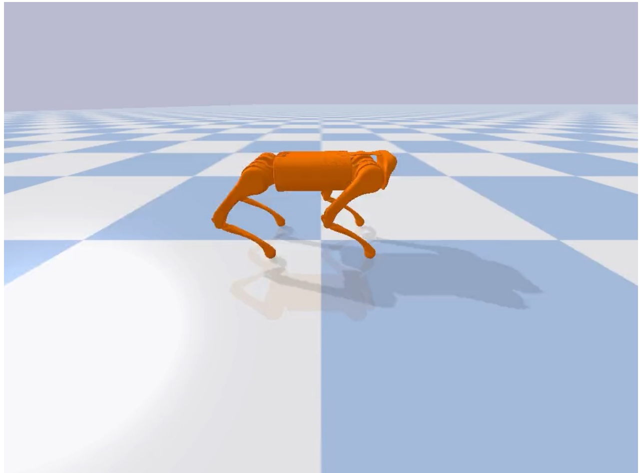
Trot



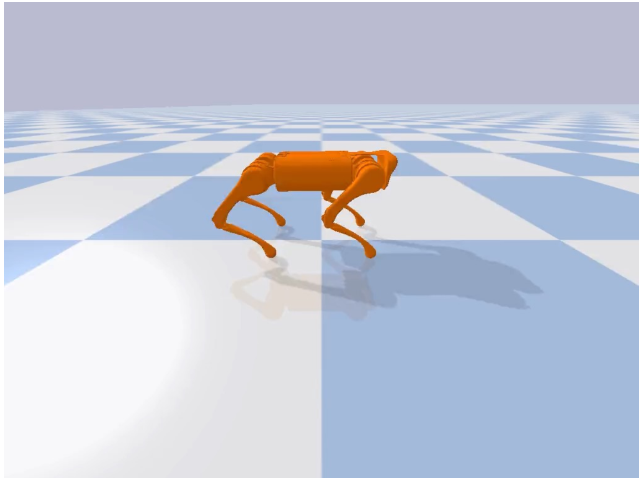
Bound



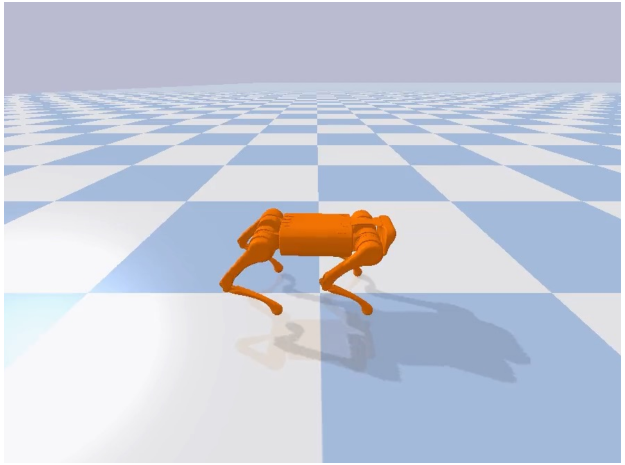
Pace

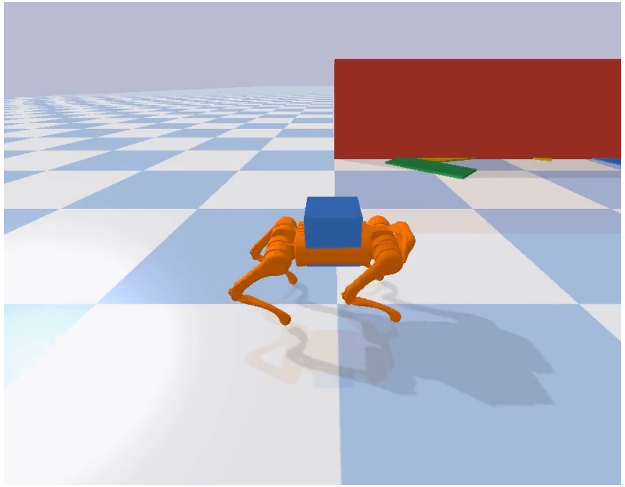


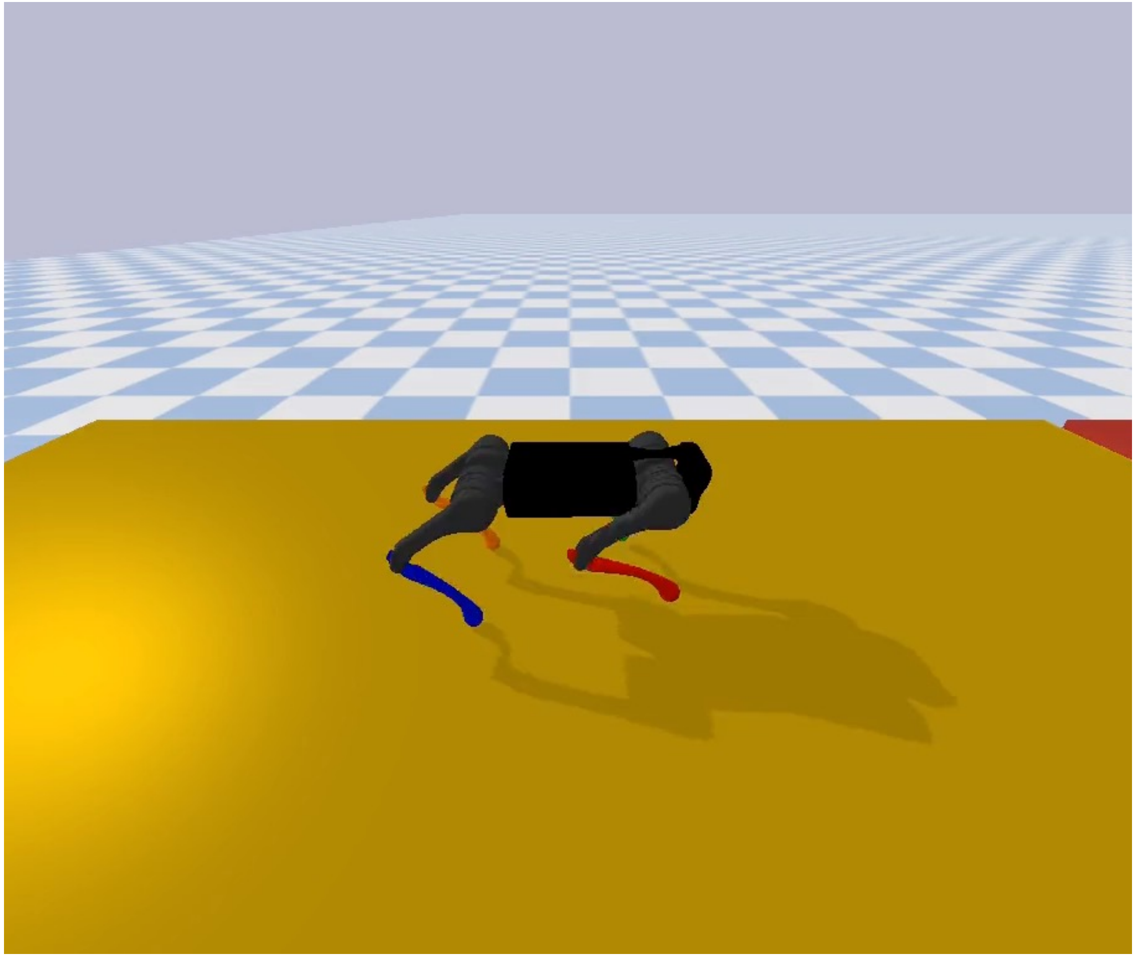
Walk



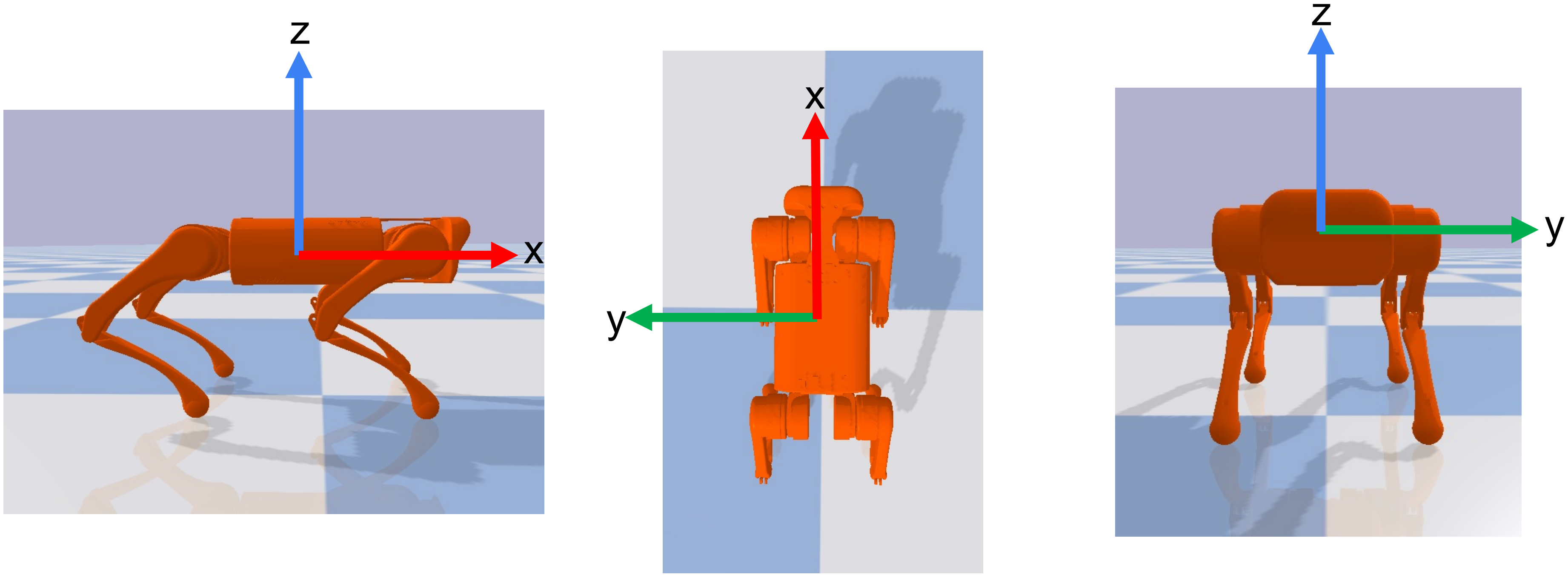
Part 2: Deep Reinforcement Learning



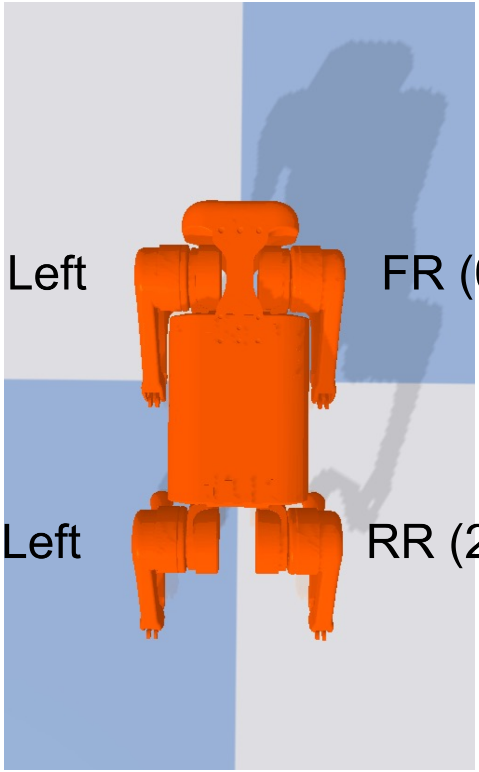




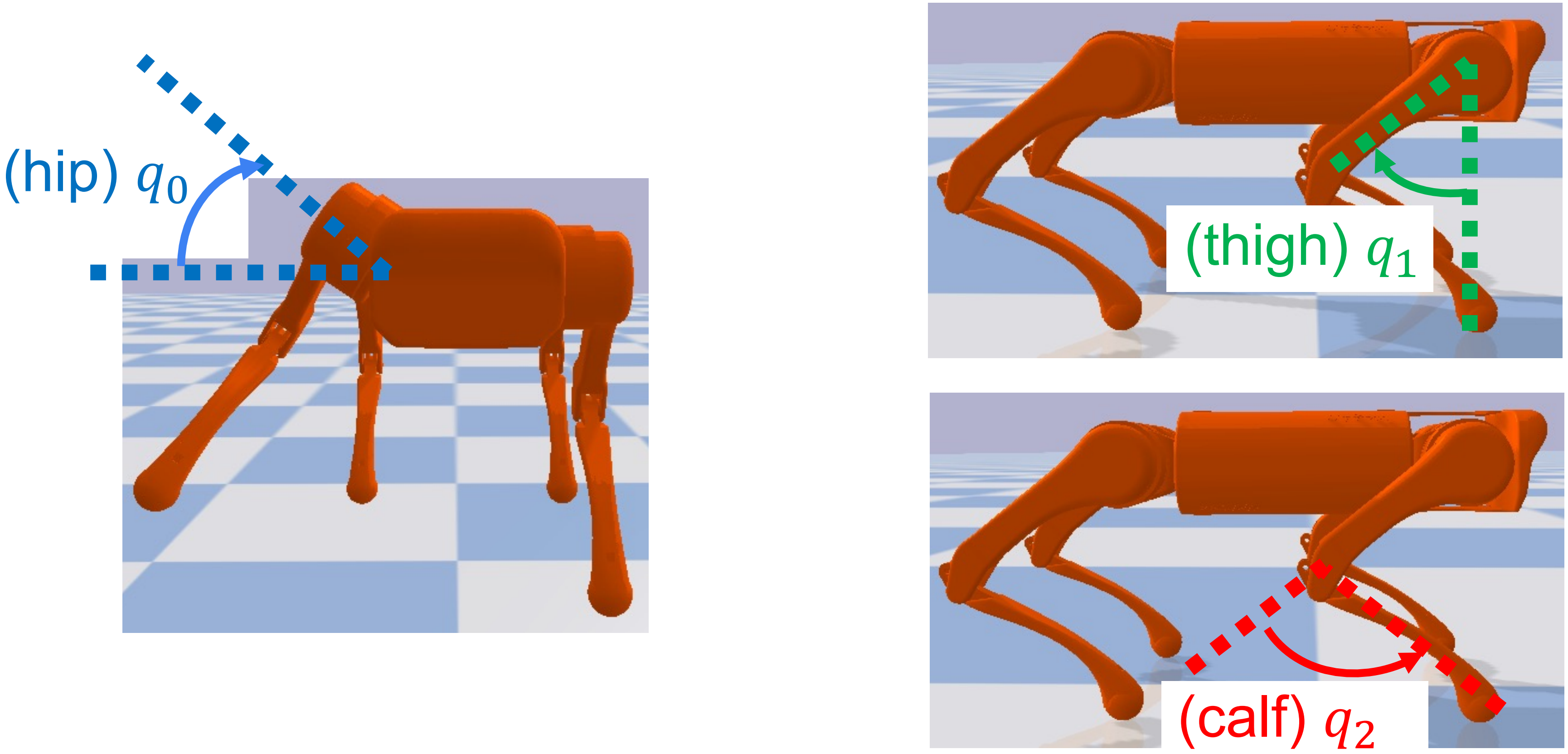
Quadruped Model Reference Frame



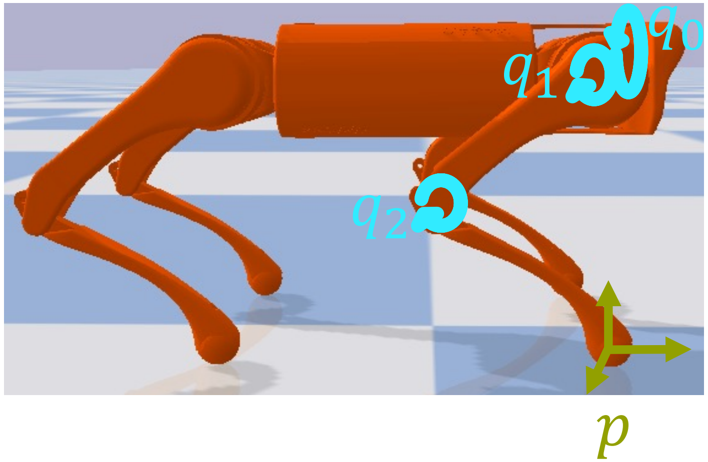
Quadruped Model Leg References



Quadruped Model Joint References



Joint angles ←→ Cartesian space (in leg frame)



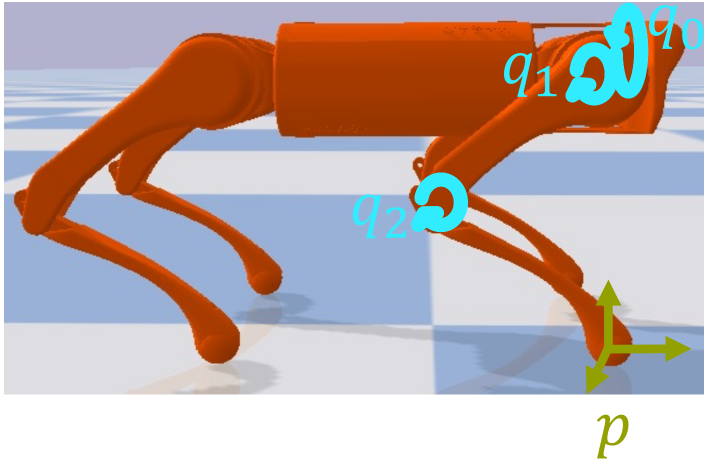
$$p = f(q)$$ Forward kinematics

$$q = f^{-1}(p)$$ Inverse kinematics

$$\dot{p} = v = J(q)\dot{q}$$ Foot linear velocity

$$\tau = J^T(q)F$$ Map desired end effector force to torques

Joint angles ←→ Cartesian space (leg frame control)



$$p = f(q)$$ Forward kinematics

$$q = f^{-1}(p)$$ Inverse kinematics

$$\dot{p} = v = J(q)\dot{q}$$ Foot linear velocity

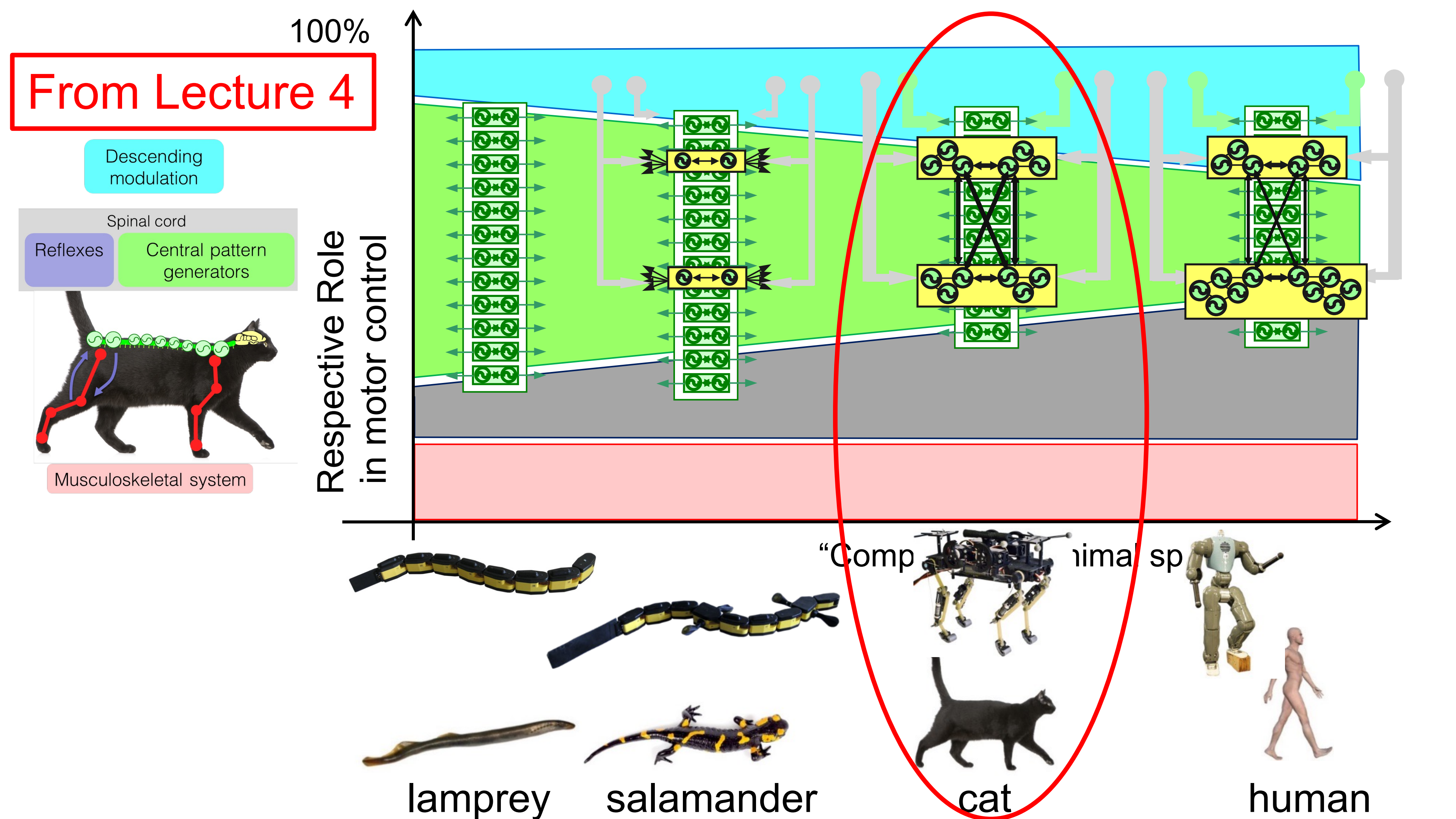
$$\tau = J^T(q)F$$ Map desired end effector force to torques

$$\tau_{joint} = K_{p,joint}(q_d - q) + K_{d,joint}(\dot{q}_d - \dot{q})$$ Joint PD

$$\tau_{Cartesian} = J^T(q)\left[K_{p,Cartesian}(p_d - p) + K_{d,Cartesian}(v_d - v)\right]$$ Cartesian PD

$$\tau_{final} = \tau_{joint} + \tau_{Cartesian}$$ Contributions from both joint PD and Cartesian PD

Central Pattern Generators: Review



From Lecture 4
100%
Descending modulation
Spinal cord
Reflexes
Central pattern generators
Musculoskeletal system
Respective Role in motor control
"Comp
imal sp
lamprey
salamander
cat
human

Modeling the CPG with coupled oscillators (Quadruped)

Amplitude:

$$\dot{r}_i = \alpha(\mu - r_i^2)r_i$$

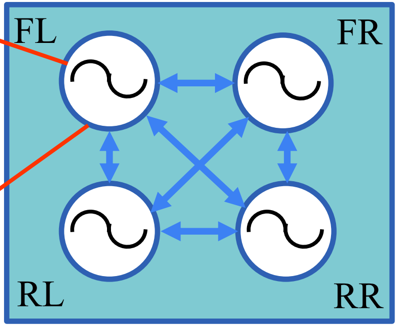
Phase:

$$\dot{\theta}_i = \omega_i + \sum_{j=0}^{3} r_j w_{ij} \sin(\theta_j - \theta_i - \phi_{ij})$$

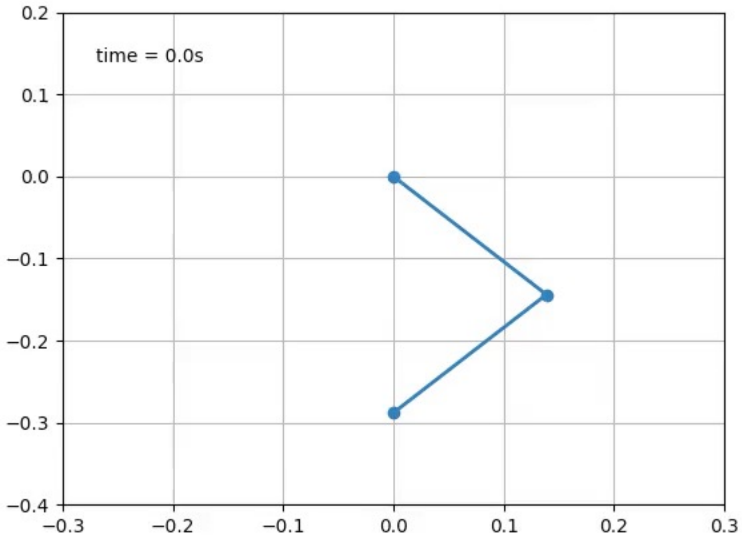
Output:

$$x_{\text{foot}} = -d_{step} r_i \cos(\theta_i)$$

$$z_{\text{foot}} = \begin{cases} -h + g_c \sin(\theta_i) & \text{if } \sin(\theta_i) > 0 \\ -h + g_p \sin(\theta_i) & \text{otherwise} \end{cases}$$



Mapping CPG States to Foot Positions with Inverse Kinematics



$$\dot{r}_i = \alpha(\mu - r_i^2)r_i$$

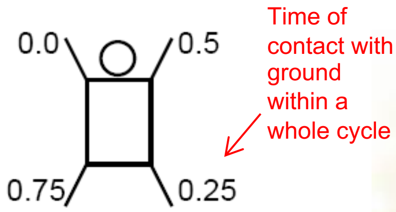
$$\dot{\theta}_i = \omega_i$$

$$x_{\text{foot}} = -d_{step} r_i \cos(\theta_i)$$

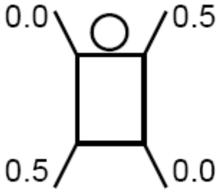
$$z_{\text{foot}} = \begin{cases} -h + g_c \sin(\theta_i) & \text{if } \sin(\theta_i) > 0 \\ -h + g_p \sin(\theta_i) & \text{otherwise} \end{cases}$$

Gait Terminology

- *Stride duration* = the duration of a complete cycle (the period)
- *Swing phase* of a limb (period during which the limb is off the ground)
- *Stance phase* (period during which the limb touches the ground)
- *Duty factor* = Stance duration / Stride duration

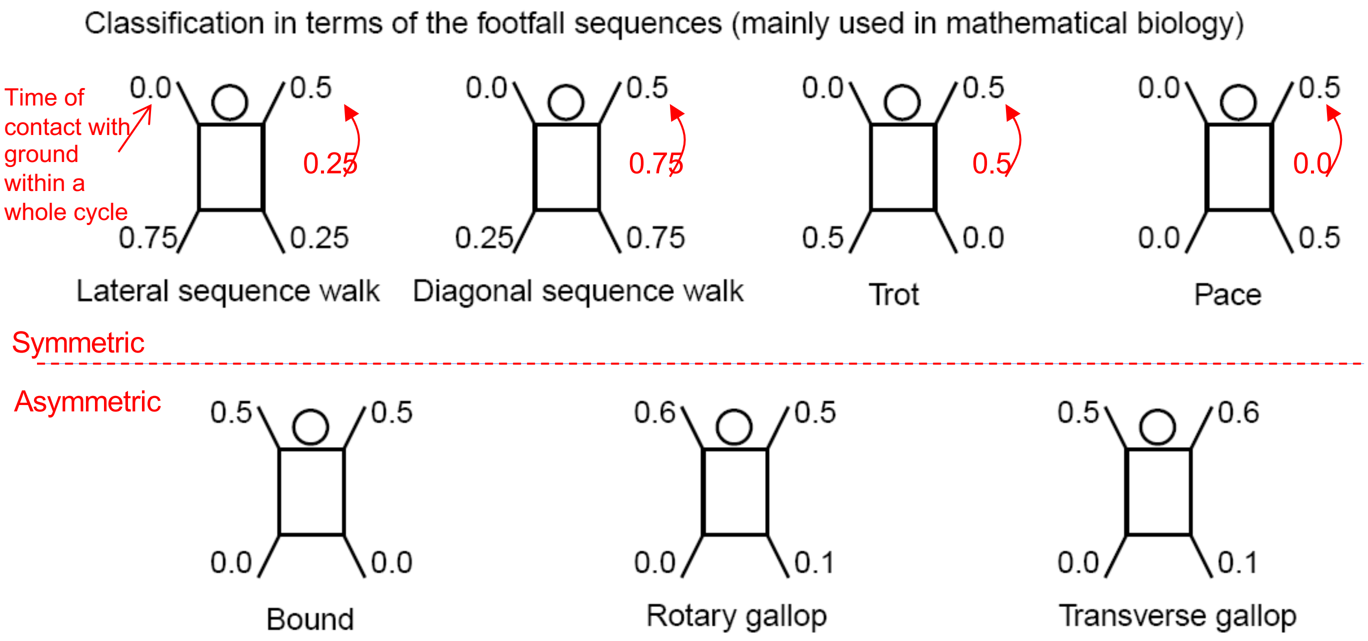


Lateral sequence walk



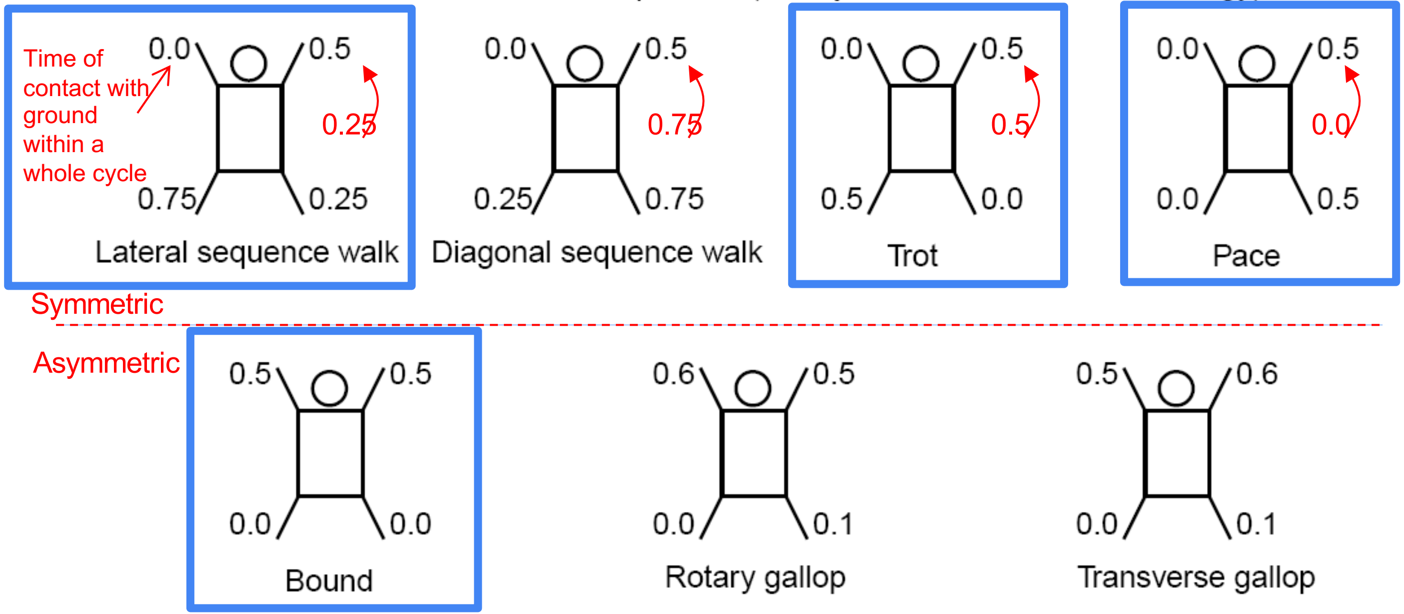
Trot

Most common quadruped gaits



Most common quadruped gaits

Classification in terms of the footfall sequences (mainly used in mathematical biology)



This project

$$\dot{r}_i = \alpha(\mu - r_i^2)r_i$$

$$\dot{\theta}_i = \omega_i + \sum_{j=0}^{3} r_j w_{ij} \sin(\theta_j - \theta_i - \phi_{ij})$$

What should $\boldsymbol{\phi}$ be for each gait?

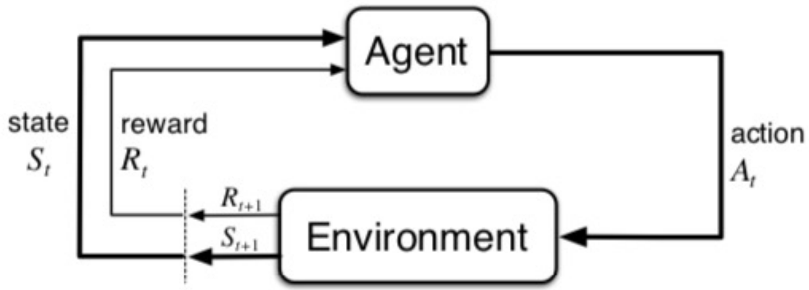
Deep Reinforcement Learning: Review

From Lecture 7

Reinforcement Learning

An MDP is defined by:

- Set of states S
- Set of actions A
- Transition function $P(s' \mid s, a)$
- Reward function $R(s, a, s')$
- Start state s_0
- Discount factor γ
- Horizon H



- Return over a trajectory $\tau = (s_0, a_0, s_1, a_1, \dots)$

$$R(\tau) = \sum_{t=0}^{\infty} \gamma^t r_t$$

- Policy $\pi(a_t|s_t)$ maps from states s_t to actions a_t (Goal: find policy maximizing above return)
- Value function: $V^\pi(s) = \mathbb{E}_{\tau\sim\pi}[R(\tau)|s_0 = s]$
- Action-value function: $Q^\pi(s,a) = \mathbb{E}_{\tau\sim\pi}[R(\tau)|s_0 = s, a_0 = a]$
- Advantage function: $A^\pi(s,a) = Q^\pi(s,a) - V^\pi(s)$

R. Sutton and A. Barto. Introduction to Reinforcement Learning. MIT Press 1998
Deep RL Bootcamp. Berkeley CA, August 2017

From Lecture 7

Many Existing Tools for Reinforcement Learning

- RL algorithm implementations
  - stable-baselines3 https://github.com/DLR-RM/stable-baselines3 PPO, SAC
  - ray[rllib] https://github.com/ray-project/ray
  - spinningup https://github.com/openai/spinningup
  - tianshou https://github.com/thu-ml/tianshou/
  - … many others!
- Physics simulators
  - pybullet https://github.com/bulletphysics/bullet3
  - MuJoCo https://mujoco.org
  - RaiSim https://raisim.com
  - Isaac-Gym https://developer.nvidia.com/isaac-gym
  - … and others!

From Lecture 7

RL Considerations

Algorithm

- On/off policy
- Hyperparameters
- Network architecture
- Random seeds/trials

…implementation dependent!

MDP Design Decisions

- Observation space
- Action space
- Reward function

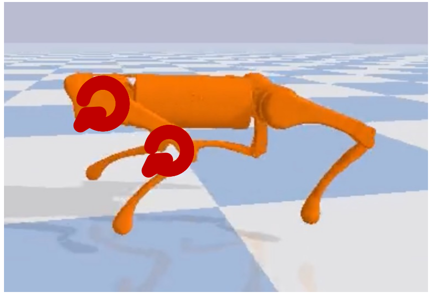
Environment Parameters

- Simulator dynamics
- Control gains – joint/Cartesian
- Control/environment time step
- Noise, latency

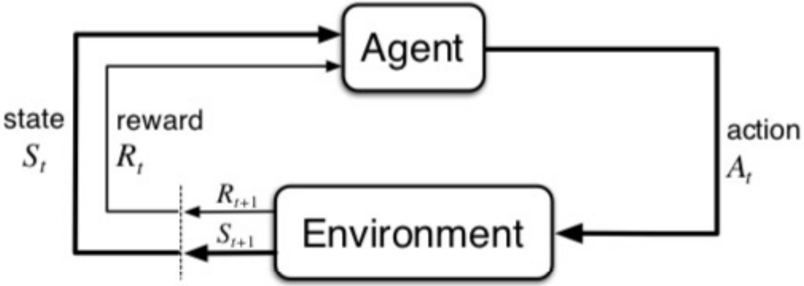
P. Henderson et al. *Deep Reinforcement Learning that Matters*. arXiv:1709.06560, 2017

From Lecture 7

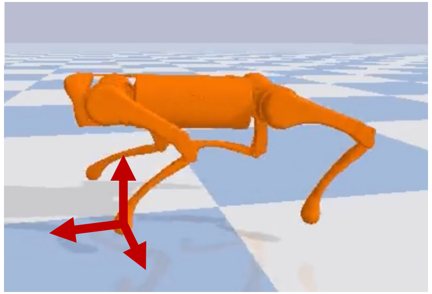
State/Action/Reward Space: A1



s_t ? i.e.
-body (z, r, p, y)
-body velocities
-joint states



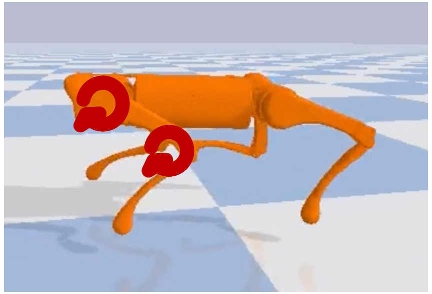
a_t ?
-motor positions/torques
-Cartesian PD
-CPG state modulations



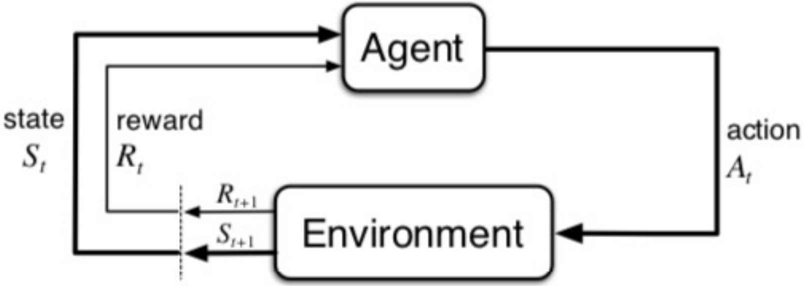
r_t ? i.e.
-body linear velocity
-energy penalty

From Lecture 7

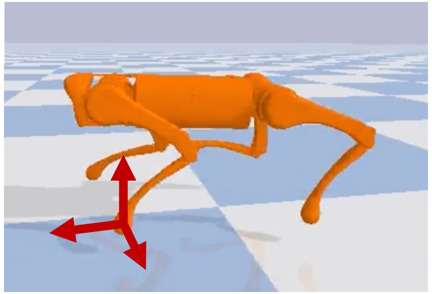
State/Action/Reward Space: A1



s_t ? i.e.
-body (z, r, p, y)
-body velocities
-joint states



a_t ?
-motor positions/torques
-Cartesian PD
-CPG state modulations



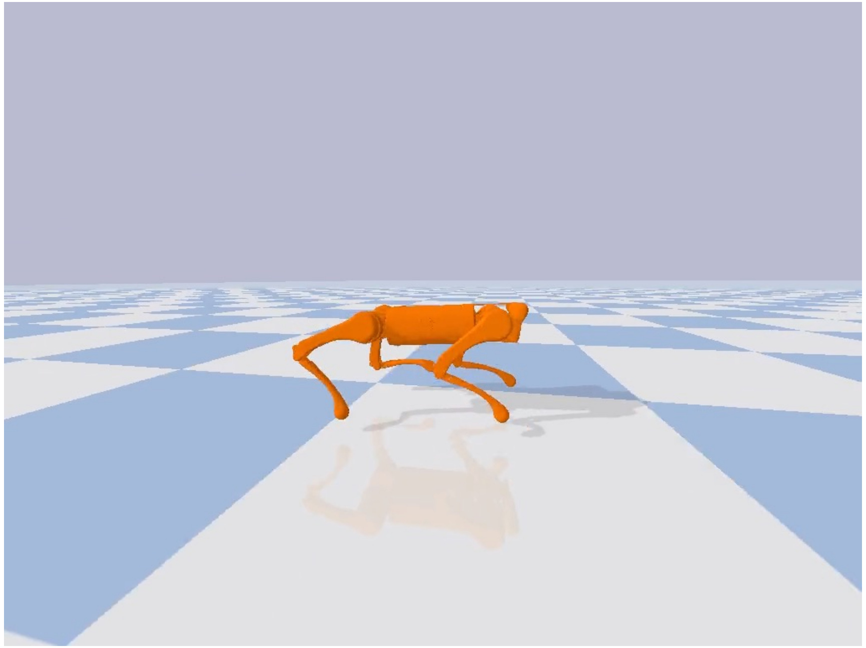
r_t ? i.e.
-body linear velocity
-energy penalty

This project: construct the MDP

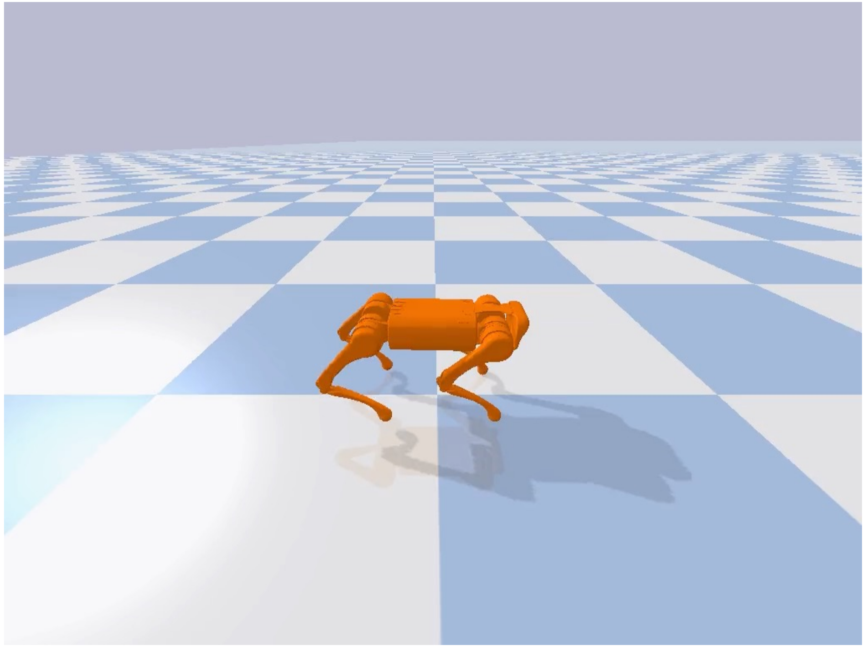
From Lecture 7

Joint Position Control vs. Cartesian PD Control (PPO/SAC)

Action Space: $a_t = q_{1 \ldots N}$

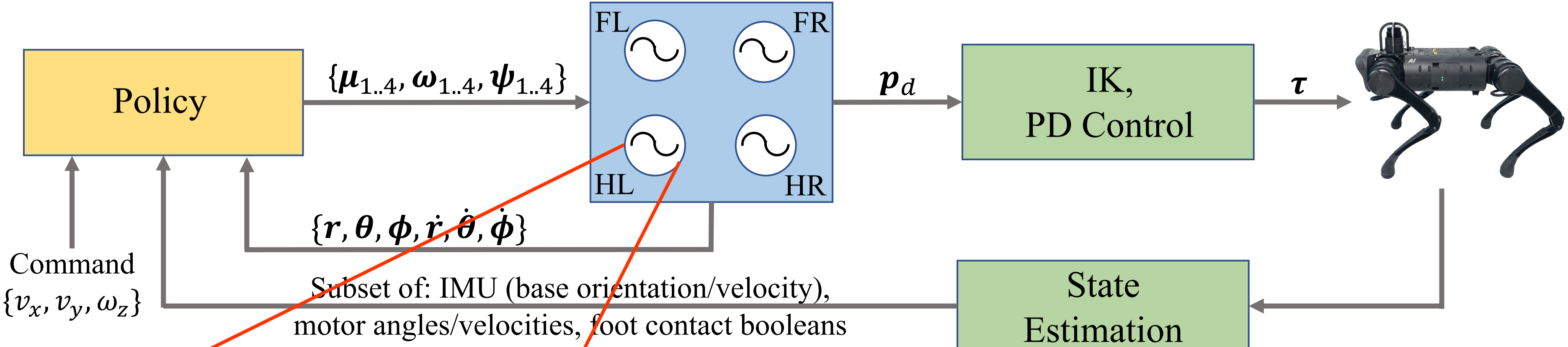


Action Space: $a_t = [x_{ee_i}, y_{ee_i}, z_{ee_i}]$



CPG-RL: Learning Central Pattern Generators for Quadruped Locomotion

From Lecture 7



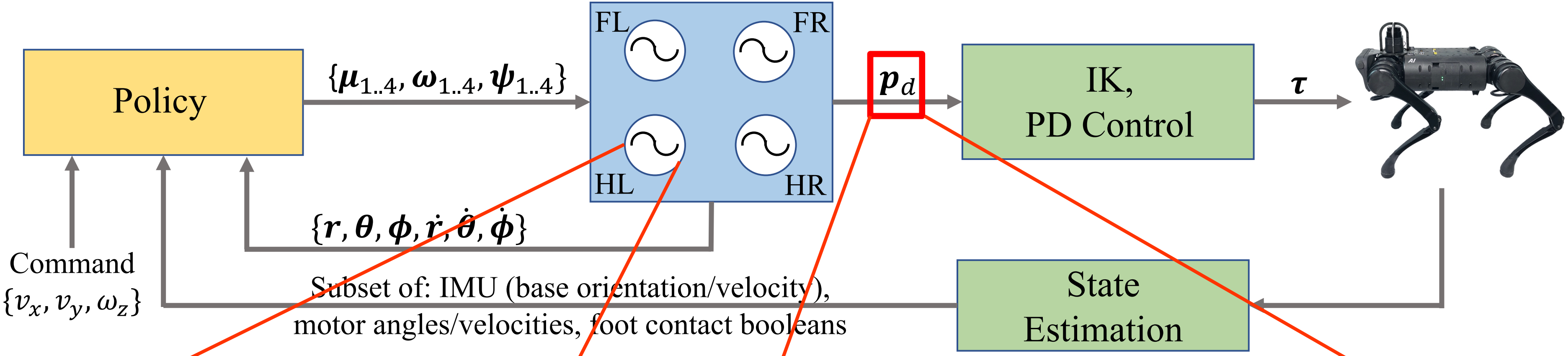
$$\dot{r}_i = \alpha(\mu - r_i^2) r_i$$

$$\dot{\theta}_i = \omega_i + \sum_{j=0}^{3} r_j w_{ij} \sin(\theta_j - \theta_i - \phi_{ij})$$

G. Bellegarda, A. Ijspeert. “CPG-RL: Learning Central Pattern Generators for Quadruped Locomotion,” RA-L 2022

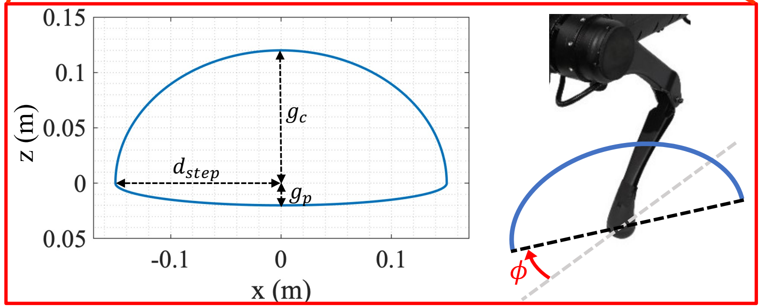
CPG-RL: Learning Central Pattern Generators for Quadruped Locomotion

From Lecture 7



$$\dot{r}_i = \alpha(\mu - r_i^2)r_i$$

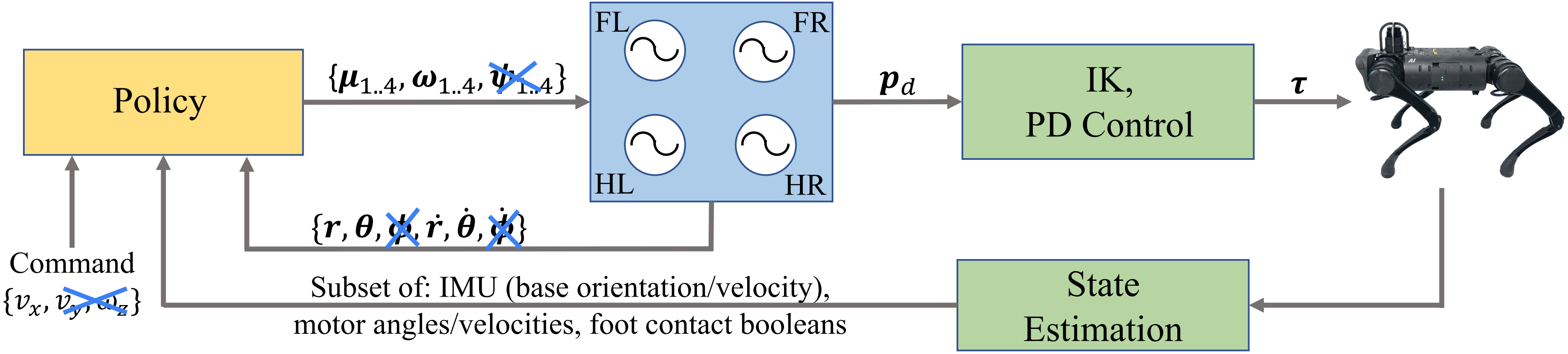
$$\dot{\theta}_i = \omega_i + \sum_{j=0}^{3} r_j w_{ij} \sin(\theta_j - \theta_i - \phi_{ij})$$



G. Bellegarda, A. Ijspeert. "CPG-RL: Learning Central Pattern Generators for Quadruped Locomotion," RA-L 2022

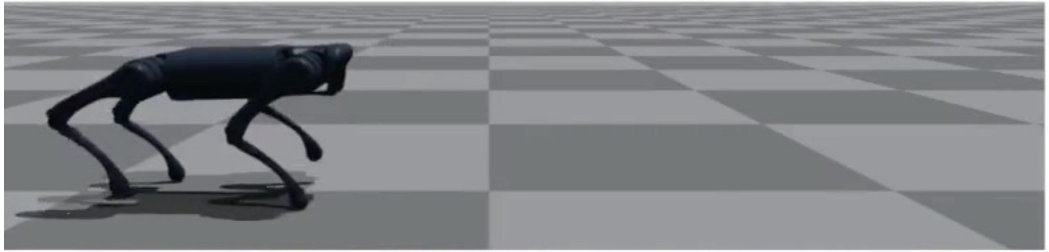
CPG-RL: Learning Central Pattern Generators for Quadruped Locomotion

From Lecture 7



$$\dot{r}_i = \alpha(\boxed{\mu} - r_i^2)r_i$$

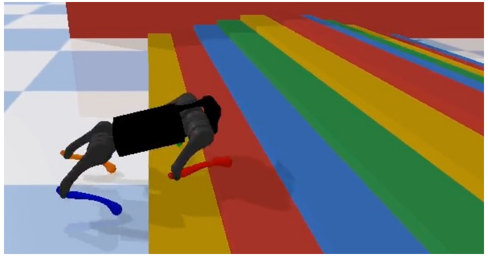
$$\dot{\theta}_i = \boxed{\omega_i} + \cancel{\sum_{j=0}^{3} r_j w_{ij} \sin(\theta_j - \theta_i - \phi_{ij})}$$

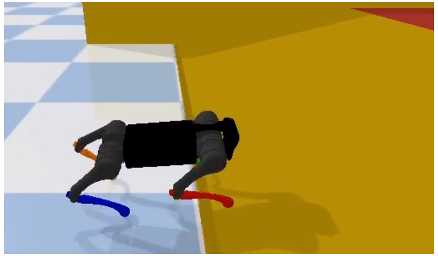


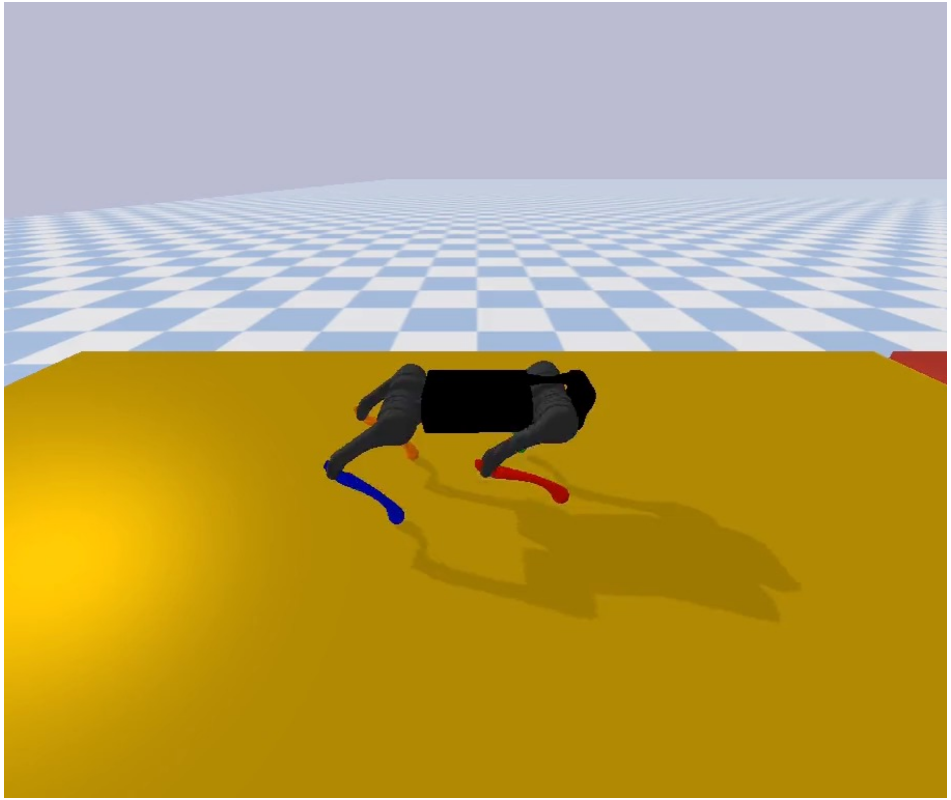
G. Bellegarda, A. Ijspeert. "CPG-RL: Learning Central Pattern Generators for Quadruped Locomotion," RA-L 2022

Choose a task and demonstrate your performance on **17.12.2024**

- Gap crossing
  - Largest gap?
  - Smallest distance between gaps?
- Stairs
  - Width?
  - Height?
- Slopes
  - Largest angle?







Tips

- Monitor episode length and reward mean during training
- Training should complete within a couple million timesteps for simple tasks with reasonable observation space, action space, and reward function choices (with no noise in the environment)
- Start training early!