Identifying Conditions for Autonomous Dynamic Gait Transitions in Quadruped Robots

Intelligent Robotics

Emanuele Cuzzocrea

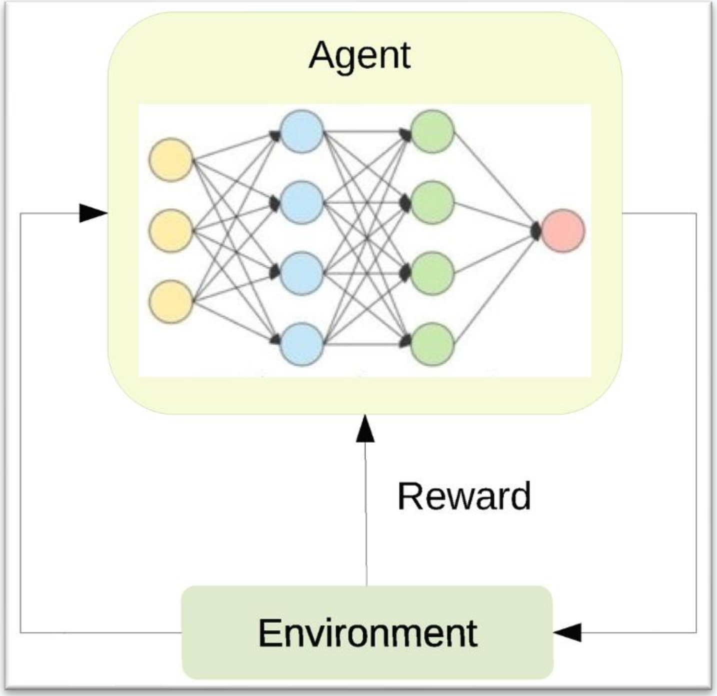
Prof. Alberto Finzi

DIETI UNIVERSITÀ DEGLI STUDI DI NAPOLI FEDERICO II

Index

1. Introduction

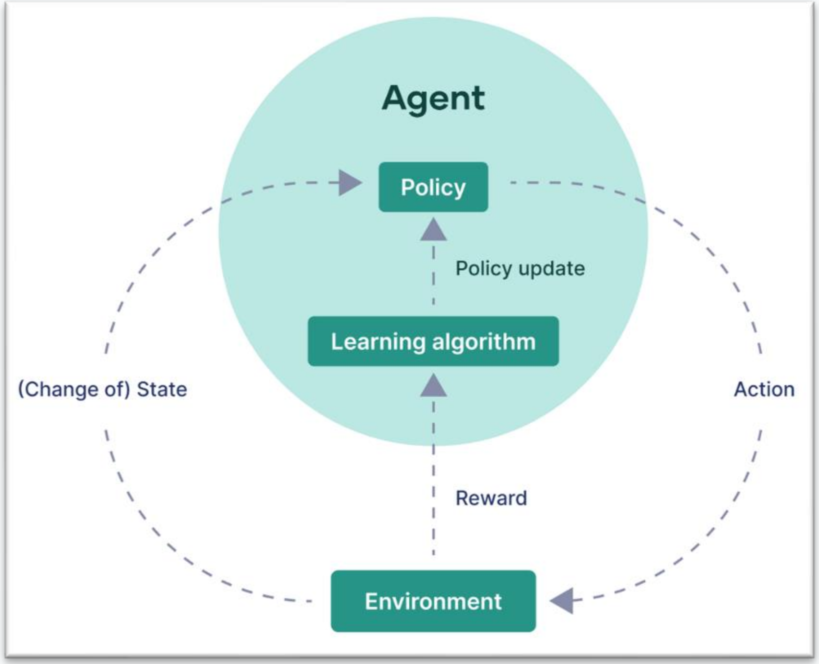
- Using deep reinforcement learning for controlling a quadruped robot
- Use reward machines for learning different gaits
- Explore transitions of policies
- Condition for autonomous transitions for the higher level control system



Reinforcement learning

$$M = (S, A, T, R, \gamma)$$

$$\pi : S \rightarrow A$$

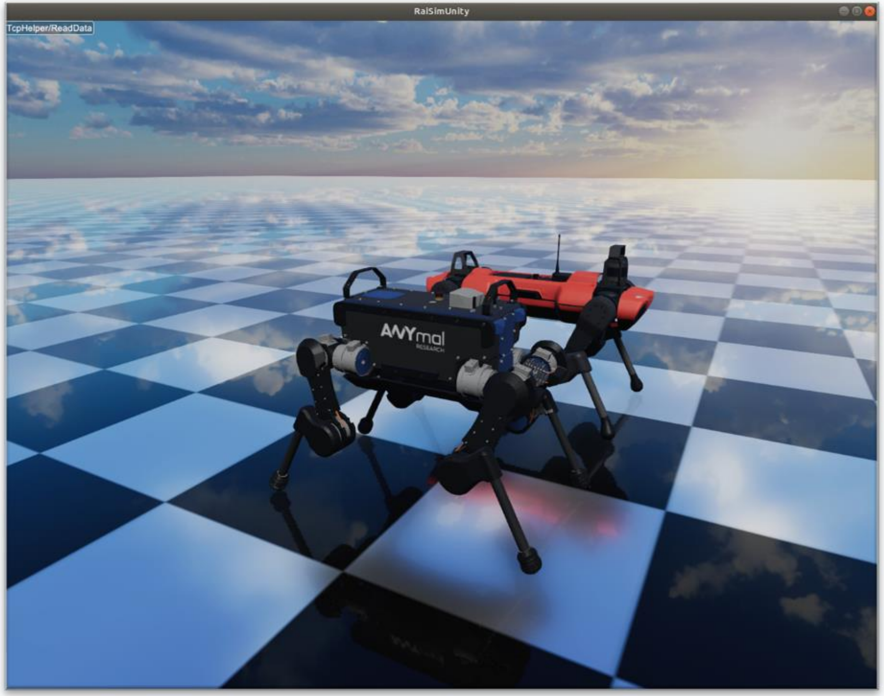


Robot used: ANYmal B

- Quadruped robot developed by ANYbotics
- 4 legs with each three identical motors (ANYdrives): electrically actuated and can sense force
- Compliant walking
- Depth cameras, LIDAR, …

Simulator: Raisim

- Cross-Platform multi-body physics engine for robotics and AI
- Deep Reinforcement Learning
- Uses Unity or Unreal for visualization
- Pytorch
- Best, after Isaac Gym from NVIDIA

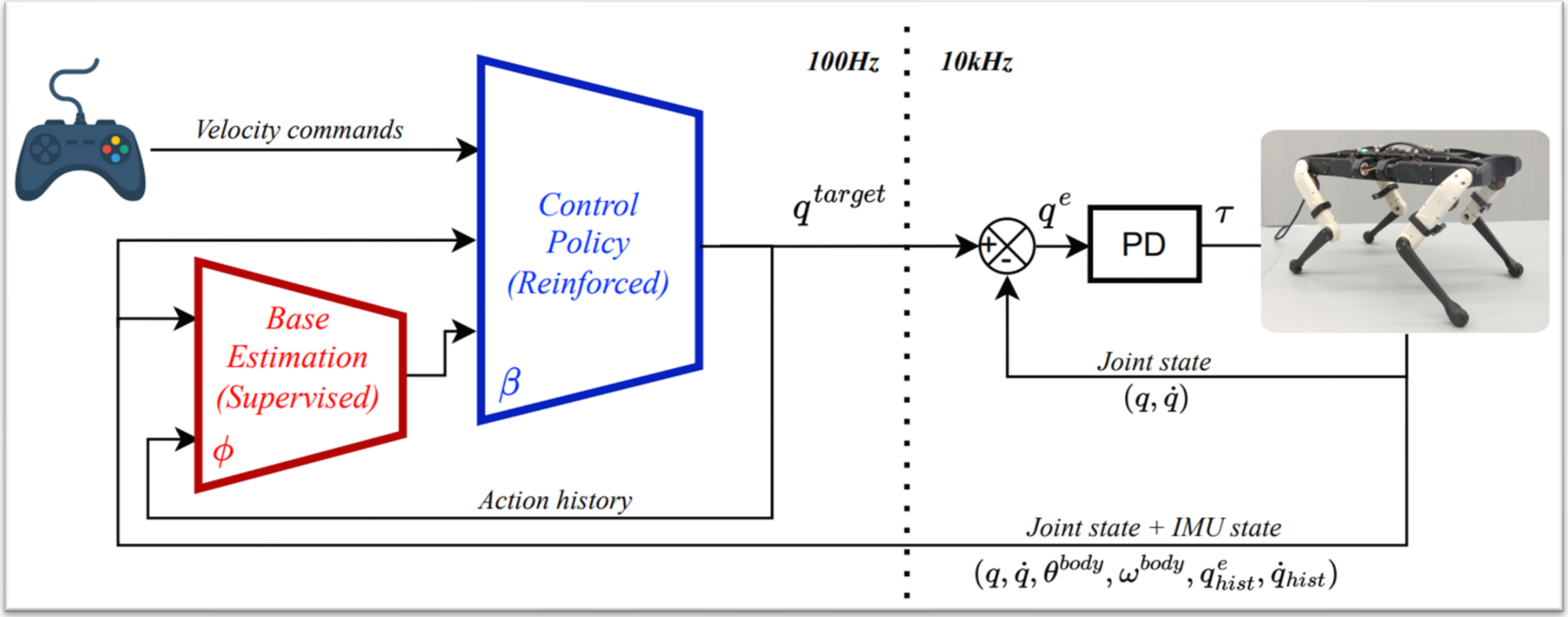


Necessity of different gaits

2. Reward Machines

- Used to manage milestone sub-goals in larger tasks
- Addressing issues of sparce or non-Markovian reward functions
- Specify sub-goals through a finite-state automaton
- Keeps the reward function Markovian
- Can speed up the training phase

3. RM for Quadruped Locomotion



State space

- q → 12 joint angles
- $\dot{q}$ → 12 joint velocities
- z → Body height
- ϕ → Body orientation
- v → Body linear velocity
- ω → Body angular velocity
- u → Current RM state
- δ → Number of steps since previous RM state change
- P → Vector of four boolean variables, that are 1 if the corresponding foot is touching the ground, 0 otherwise

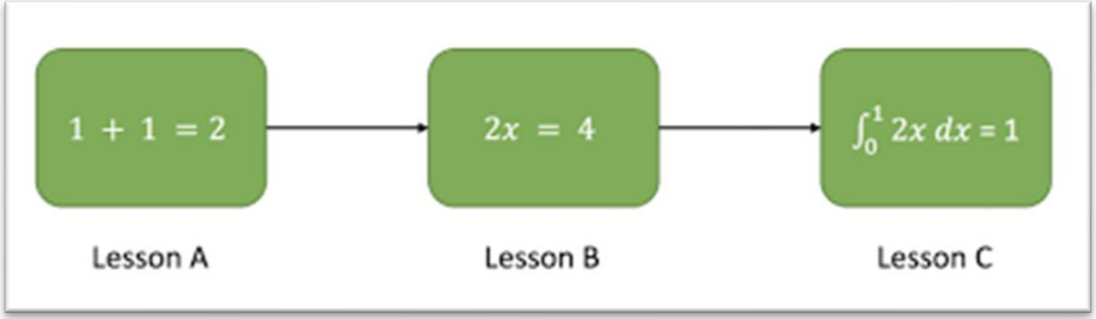
$$S = (q, \dot{q}, z, \phi, v, \omega, u, \delta, P)$$

Reward function (fixed forward velocity)

Term Description	Definition	Wheight
Linear Velocity x	$exp(-\|v_{d,x} - v_x\|^2)$	10
Linear Velocity y	v_y^2	-10
Angular Velocity x, y	$\|\omega_{x,y}\|^2$	-0.5
Angular Velocity z	$\|\omega_z\|^2$	-25
Joint Torques	$\|\tau\|^2$	-0.00004
Joint Position	$\|q - q_{init}\|^2$	-0.1
Joint Velocity	$\|\dot{q}\|^2$	-0.01

Curriculum learning

- Increase slowly the weight of the penalties
- The robot can find a local reward maximum by doing nothing!
- Fondamental when not using reward machines
- Optional when using reward machines
- The training can become very slow



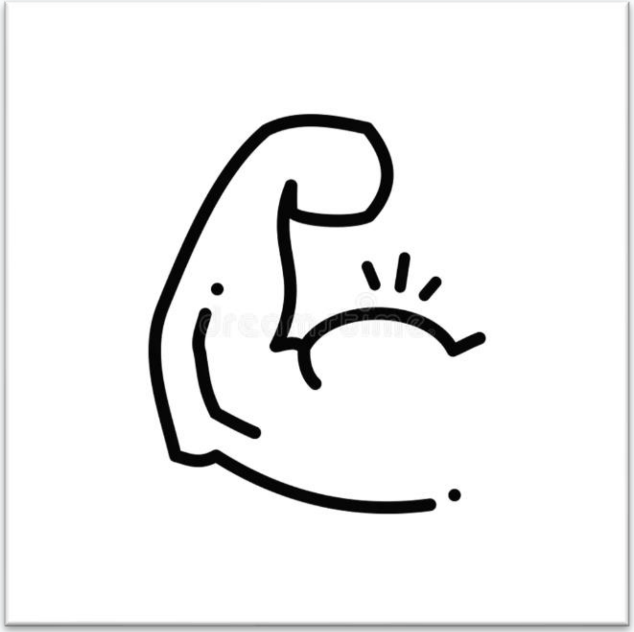
Training of trot gait

Training of bound gait

Training of pace gait

4. Robust gait transitions

- Specific training can be done on transitions
- Domain randomization
- External disturbances



Domain and dynamic randomization

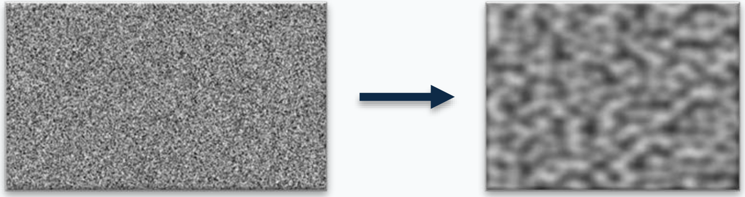
- Significantly improve the sim-to-real gap
- Make the locomotion more robust
- Can be implemented with curriculum learning

- Adding noise to the state observation
- Adding noise to the motors
- Adding noise to the parameters of control (K_p, K_d)

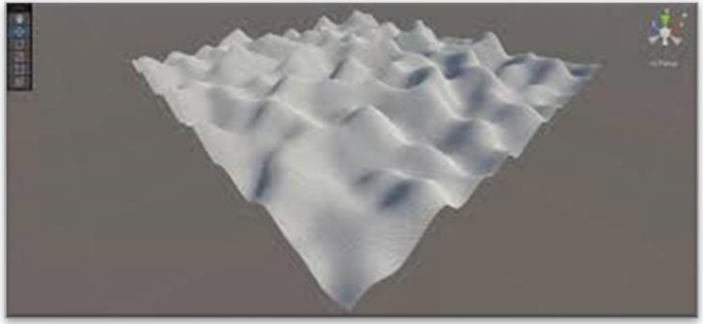
- Randomize the terrain
- Randomize the obstacles
- Randomize the dynamics

Terrain randomization with Perlin noise

- Type of gradient noise developed by Ken Perlin in 1983



- Many uses: procedurally generating terrain, pseudo-random changes to variables, assisting in the creation of image textures

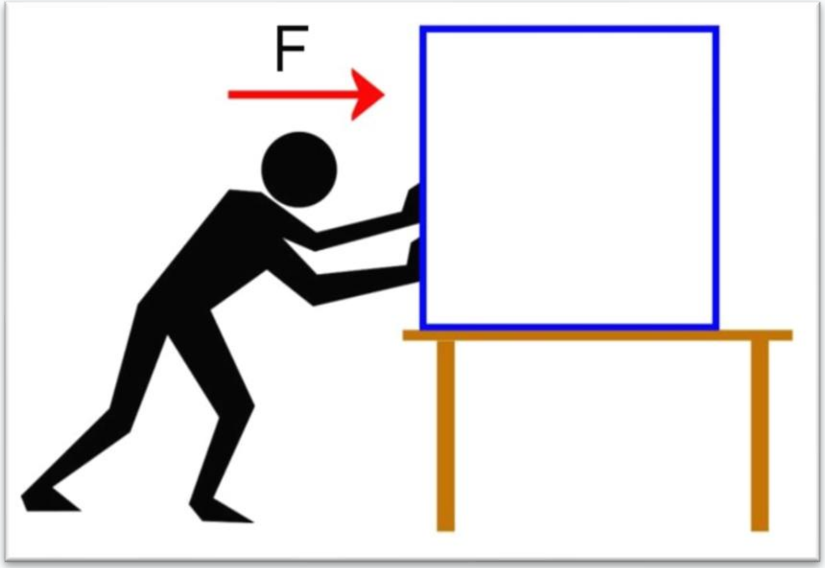


- Can be defined for any number of dimensions
- Change terrain each 5 iterations for limiting computational cost

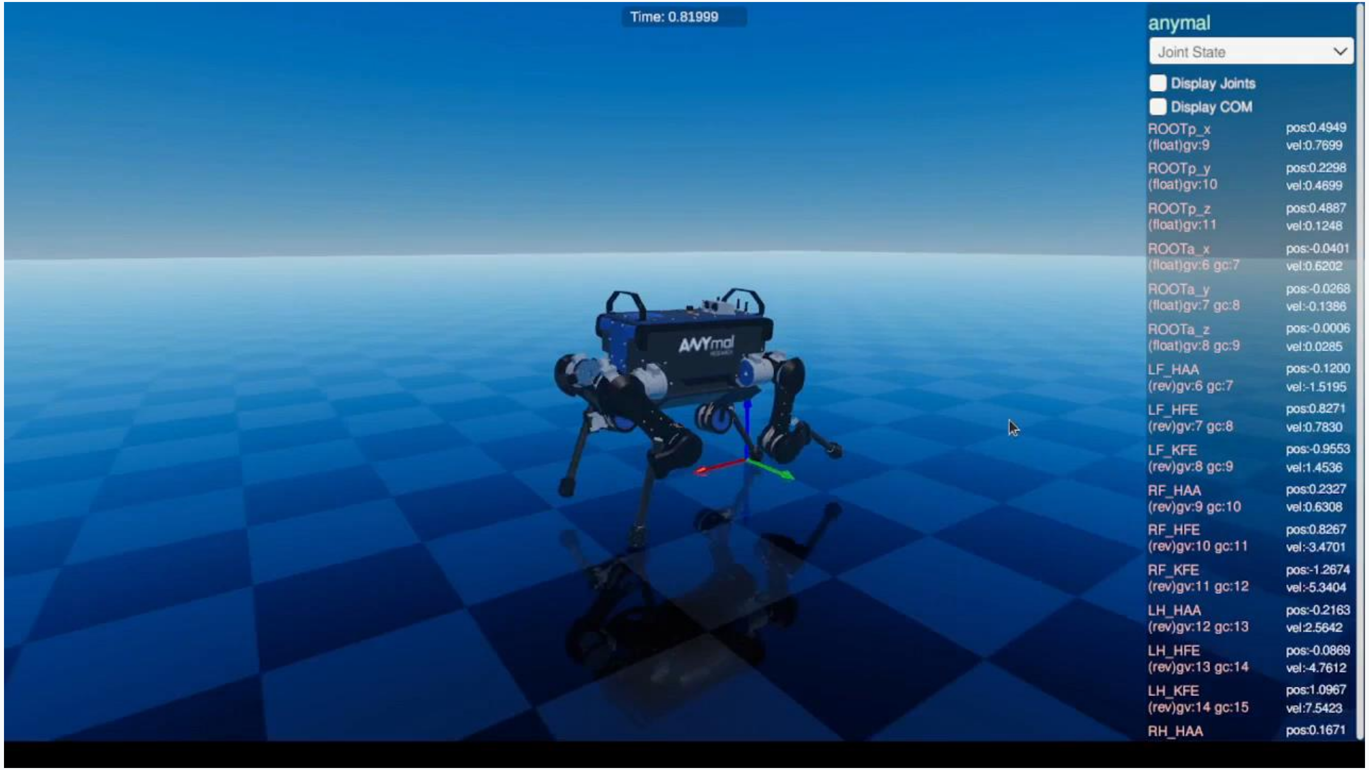
Terrain randomization with Perlin noise

External disturbances

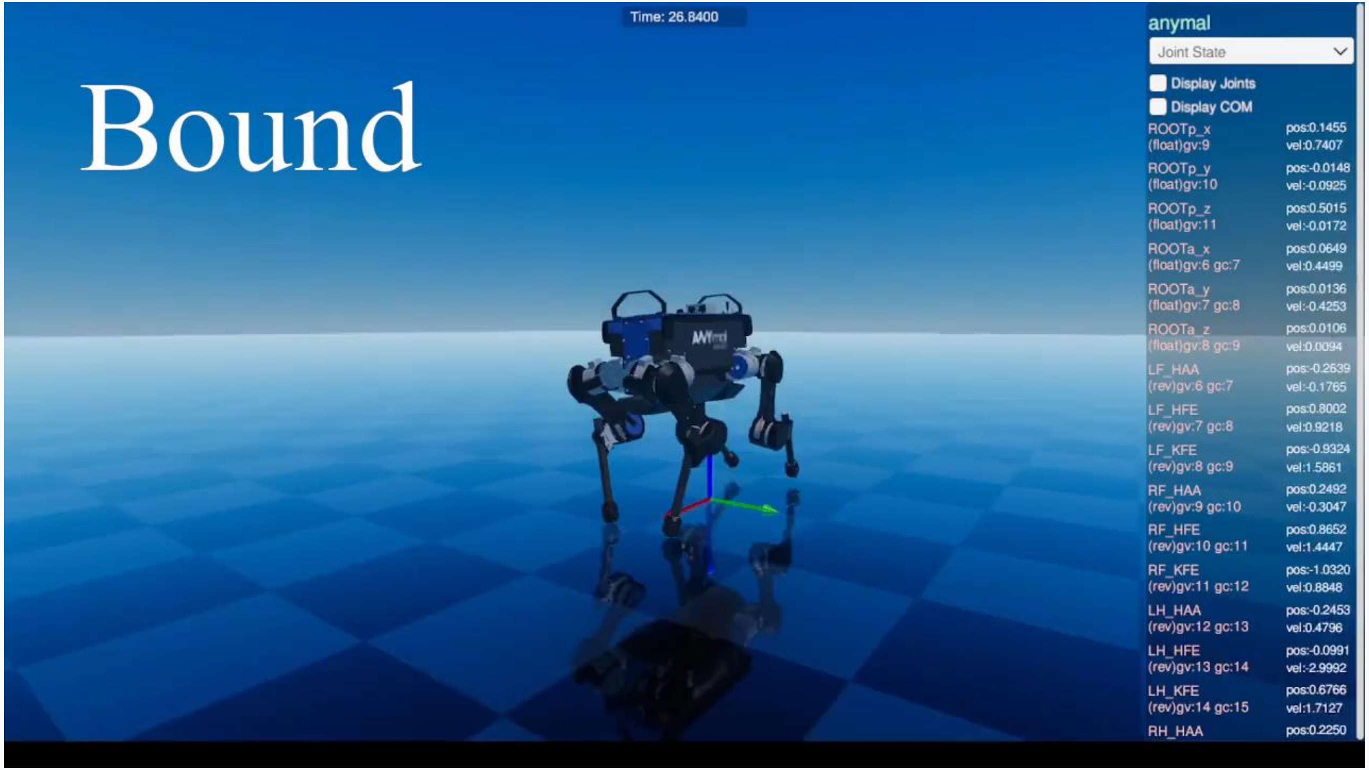
- Random forces along x, y, z
- Used to recover from difficoult position
- Applied to the base frame
- Applied on the legs
- Preparation for the real world



External disturbances (trot gait example)



Gait transitions example (manual switching)



5. Personal contribution: Condition for autonomous transition

- We need a simple condition to be checked so that the **higher-level controller** knows when to perform the policy change, all autonomously.

- Let's take in a vector $h = \{h_{FL}, h_{FR}, h_{HL}, h_{HR}\}$ the height of each foot last time they touched the ground.

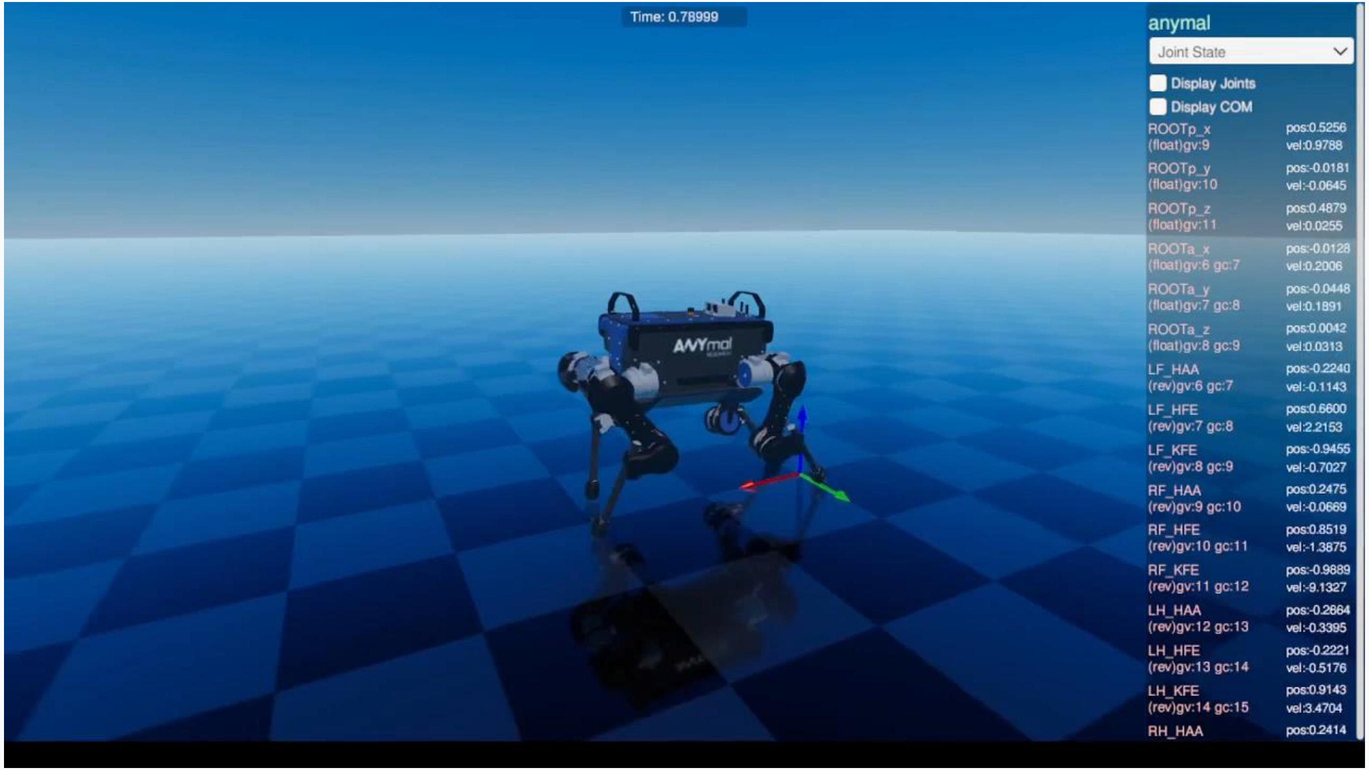
- Only two parameters to tune:
- α → Magnitude of the irregularity of the terrain
- β → Lenght of the irregularity of the terrain
- k → Just a counting variable

Pseudocode

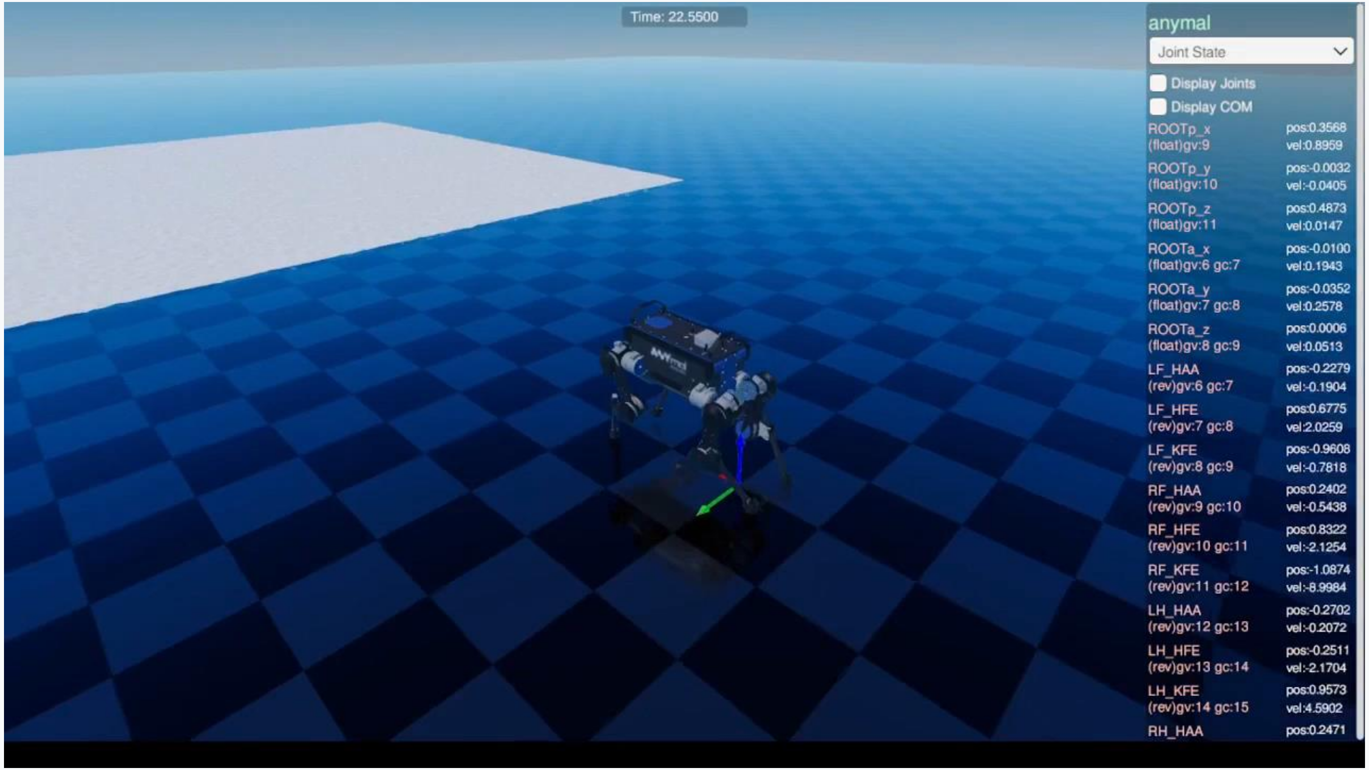
Algorithm 1 Condition for autonomous gait transitions

Run at each step:
for all i, j **do**
    if $|h_i - h_j| > \alpha$ **then**
        $k = k + 1$
    end if
end for
if $k > \beta$ **then**
    Change policy
end if

Autonomous change of gait ($\alpha = 0.2, \beta = 1000$)



Autonomous change of gait (aerial view)



6. Conclusions

- Single gaits were successfully learned thanks to deep reinforcement learning techniques
- Reward Machines turned out to be crucial for the right gait learning
- Transition between gaits were made robust randomizing the terrain, and applying external forces to the robot
- Condition for autonomous gait change proved to be suitable for different terrain, and easy to tune according to the objective

Thanks.