

Supplementary Materials for UBSOFT: A Simulation Platform for Robotic Skill Learning in Unbounded Soft Environments

Anonymous Author(s)

Affiliation

Address

email

1 A Implementation Details on the UBSOFTEngine

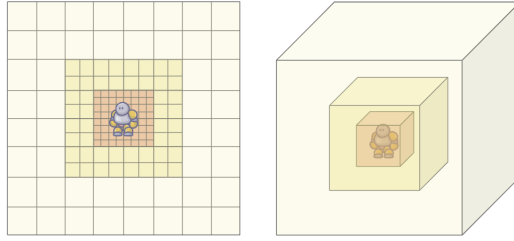


Figure 1: Hierarchical Grids for Spatially Adaptive Scheme.

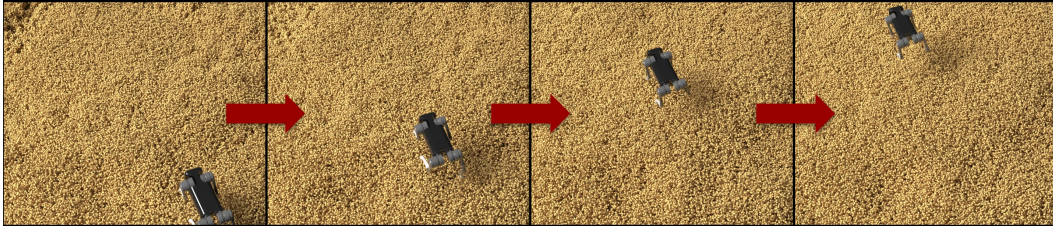


Figure 2: A robot dog walks in a desert with dunes.

2 A.1 Spatial Adaptive Scheme

3 To improve the efficiency of our proposed algorithm, we implement a dynamic data structure based
4 on tagging and amortized restructuring to support the deletion and addition of particles during the
5 resampling process.

6 The hierarchical grid system introduced in section 3.2 is illustrated in Figure 1. Centering on the
7 robot's position, grids of varying granularity are nested, enabling spatial discretization to varying
8 degrees based on the robot's position. Unlike traditional MPM, which applies a uniform granularity
9 across the entire spatial domain, our hierarchical grid *adapts* dynamically according to the distance to
10 the agent.

11 With the proposed spatial adaptive scheme, UBSOFT is able to efficiently simulate large-scale
12 scenes filled with soft materials, enabling real-time and long-horizon robot learning in *unbounded*
13 environments, such as a robot dog walking on a desert in Figure 2.

14 B UBSOFT Tasks and Evaluation Details

15 B.1 Task Details

16 In RL algorithms, we employ stratified sampling and downsampling from all particles to construct
17 the observation input.

18 In all manipulation tasks, we utilize a simulation step of 2e-4 seconds, and we establish a maximum
19 action range specific to each task to ensure system stability. In locomotion tasks, the simulation step
20 is 2e-3 seconds composed of 10 sub-steps for easier control policy learning. All tasks run faster than
21 real time (more than 60 FPS where each frame is a simulation step) on a laptop computer equipped
22 with an Nvidia RTX 4090 GPU and an Intel i9 CPU.

23 B.2 Reward Design

24 In each task, the reward is defined as $\mathcal{R} = -\alpha\mathcal{L} + \beta$, where L is the total loss of the entire episode,
25 α and β are task-specific constant for reward scaling.

26 **Sand Painting.** We use the state of particles in the ground-truth policy as the target state, compute
27 the chamfer distance d_t between the current state and the target state at each time step, and sum them
28 up as the total loss, $\mathcal{L} = \sum_t d_t$.

29 **Scoop Out.** This task has two phases, inserting the shovel into the sand and then lifting the shovel
30 to scoop out the duck. We found that simple matching with goal patterns is hard to learn and not
31 robust enough. Therefore, we divided the learning process into two stages as well. In the first stage,
32 the goal is to encourage the shovel to approach a position under the duck, thus $l_1 = \|p_{shovel} - p_t\|$,
33 where p_t is dynamically computed as $p_t = p_{duck} - (0, 0, 0.03)$. In the second stage, the goal is to
34 encourage the shovel to lift and scoop out the duck, thus $l_2 = -z_{shovel} + n_{d,t}$, where $n_{d,t}$ is the
35 number of particles within 0.1m around the duck at time step i . Finally, total loss is computed as
36 $\mathcal{L} = \sum_t (\sigma_t \cdot l_1 + (1 - \sigma_t) \cdot l_2)$, where σ_t indicates whether step t is in the first stage.

37 **Dig A Hole.** Similar to the Scoop Out task, this task also consists of two stages. In the first stage, the
38 loss function $l_1 = |z_{shovel} - z_p|$ encourages the shovel to descend, where z_p is the z-coordinate of a
39 predefined point below the sand surface. In the second stage, the loss function $l_2 = -(z_{shovel} - z_p)$
40 encourages the shovel to ascend. The final total loss is computed the same way as in Scoop Out,
41 $\mathcal{L} = \sum_t (\sigma_t \cdot l_1 + (1 - \sigma_t) \cdot l_2)$, where σ_t indicates whether step t is in the first stage.

42 **Smooth Surface.** In this task, we use the number of particles above a predefined threshold as the
43 metric. Specifically, the loss is defined as $\mathcal{L} = \sum_{t>1} (n_{s,t} - n_{s,t-1})$, where $n_{s,t}$ is the number of
44 particles above the average sand surface height.

45 **Quadruped Walk.** At each step, we compute the distance d_t between the current state and the target
46 state, represented by the pelvis joint state. The target state is defined as the robot's state that maintains
47 a forward velocity close to 1 m/s. The total loss $\mathcal{L} = \sum_t d_t$ is the sum of these distances.

48 **Humanoid Stand.** To maintain the humanoid robot in a standing pose, we compute the delta height
49 between the current state and the initial state, represented by the pelvis joint state $d_{p,t}$ and the head
50 joint state $d_{h,t}$. The total loss $\mathcal{L} = \sum_t (d_{p,t} + d_{h,t})$ is the sum of these delta heights.

51 **Humanoid Walk.** The target state is defined as the humanoid robot's state that maintains a forward
52 velocity close to 1m/s without falling to the ground. At each step, we compute the distance d_t between
53 the current and the target pelvis state, reflecting the forward velocity. Additionally, we compute the
54 delta head joint height $d_{h,t}$, representing the standing pose. The total loss $\mathcal{L} = \sum_t (d_t + d_{h,t})$ is the
55 sum of the forward distances and the delta heights.

Task	Quadruped Sand Walk	Humanoid Snow Stand	Humanoid Snow Walk
PPO	862.8±94.8	528.3±32.1	518.8±32.7
SAC	1572.8±245.5	553.1±34.7	475.3±34.8
CMA-ES	1037.9±123.3	717.7±47.7	524.5±19.3

Task	Quadruped Snow Walk	Humanoid Snow Stand	Humanoid Snow Walk
PPO	2387.4±336.7	507.0±12.0	460.2±21.4
SAC	1225.0±201.8	753.3±61.3	530.9±37.2
CMA-ES	1160.6±95.9	716.6±39.9	554.1±30.6

Task	Quadruped Elastic Walk	Humanoid Elastic Stand	Humanoid Elastic Walk
PPO	2387.4±514.6	555.7±36.6	485.1±32.2
SAC	1214.6±276.6	653.4±51.2	579.7±37.3
CMA-ES	1224.4±156.5	686.2±31.9	648.8±40.4

Table 1: The final accumulated reward and the standard deviation for each method.

56 C Additional Experiments Results

57 C.1 Locomotion Task Results on More Materials

58 To further investigate our environment, we conduct experiments with locomotion tasks on more
59 materials, including sand particles, snow particles, and elastic particles. The results are presented
60 in Table 1 and Figure 3. Similar to the results reported in the main paper, reinforcement learning
61 algorithms generally struggle with these tasks. In particular, PPO generally fails on humanoid standing
62 tasks across all surfaces. On the other hand, the sampling-based trajectory optimization algorithm,
63 represented by CMA-ES, typically achieves better performance. This pattern is further amplified
64 by the large and complex state and action spaces associated with locomotion tasks. Additionally,
65 maintaining a specified pose or moving forward consistently remains challenging for end-to-end
66 training, especially on diverse rough surfaces. Consequently, developing an improved policy for
67 locomotion tasks remains a significant challenge.

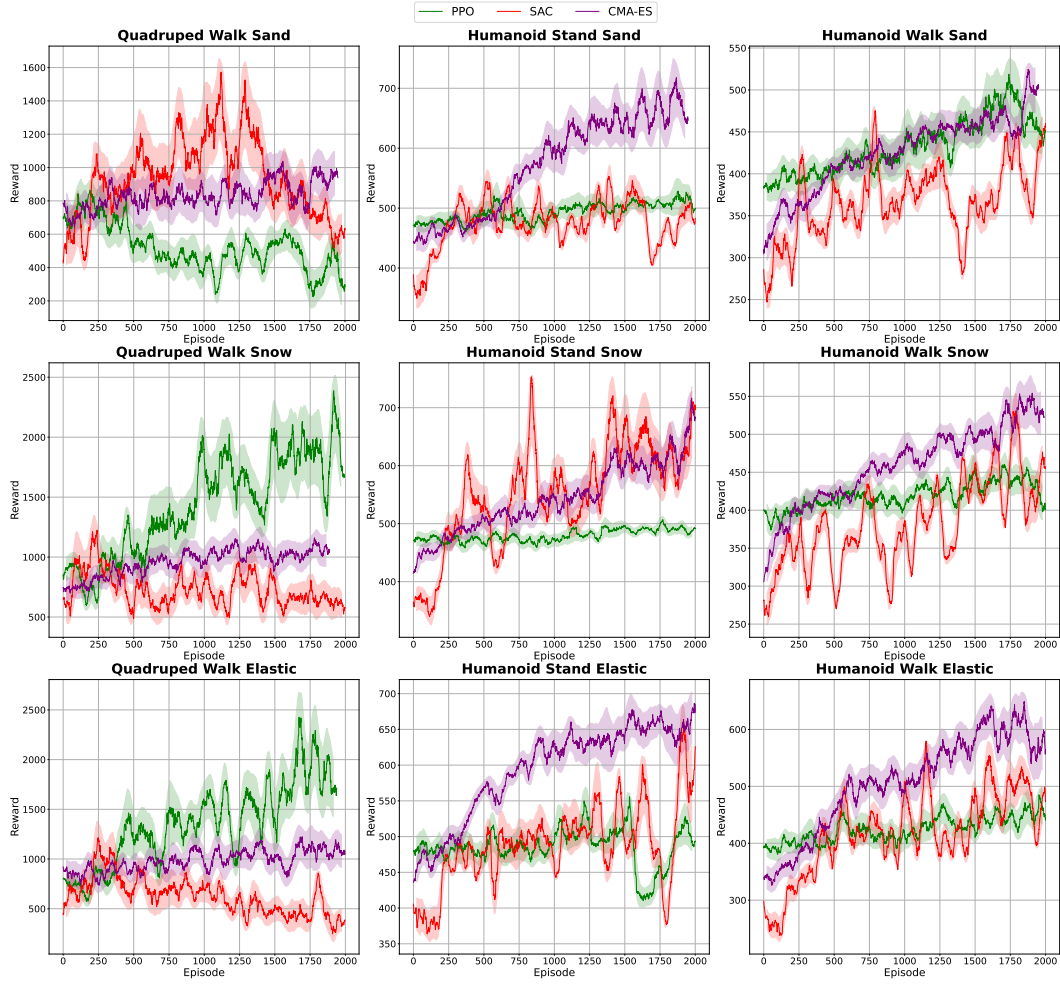


Figure 3: Reward curves for all methods on locomotion tasks, including PPO, SAC, and CMA-ES.