Hybrid Residual Reinforcement Learning on a Multi-Modal Legged Robot

Chen Yu and Andre Rosendo

Abstract— **While quadruped robots usually have good stability and load capacity, bipedal robots offer a higher level of flexibility / adaptability to different tasks and environments. A multi-modal legged robot can take the best of both worlds. In this paper, we show a complete pipeline of design, control, and sim-to-real transfer of a multi-modal legged robot. We 1) design an additional supporting structure for a quadruped robot, 2) evaluate a class of novel hybrid learning-based algorithms combining different reinforcement learning algorithms (Twin Delayed Deep Deterministic Policy Gradients and Soft Actor Critic) and black-box parameter optimisers (Evolutionary Strategy and Bayesian Optimisation), and 3) propose a sim-to-real transfer technique. We use parameter optimisers to tune a conventional feedback controller simultaneously with the training of an RL agent that solves the residual task. Experimental results show that our proposed algorithms have the best performance in simulation and competitive performance on the real robot with our sim-to-real technique. Overall, our multi-modal robot could successfully switch between biped and quadruped, and walk in both modes. Experiment videos and code are available at `https://chenaah.github.io/multimodal/`.**

I. INTRODUCTION

The development of quadruped robots is an interesting and popular research field in robotics since they are simpler than hexapod robots while having better stability and load capacity than biped robots [1]. Quadruped robots have been rapidly developed recently, such as BigDog [2], Spot [3], Mini Cheetah [4], and ANYmal [5]. These robots are implemented with bionic structures to benefit from the quadruped performance from animals. On the other hand, bipedal robots are also of interest to researchers due to their humanoid structures, mimicking human locomotion [6]. In comparison to quadruped robots, humanoid robots usually have a higher level of flexibility/adaptability to different environments and applications. Current examples of well known bipedal robots are Atlas from Boston Dynamics [7] and Cassie [8]. Motivated by the advantages of both quadruped and bipedal robots, we propose a novel multi-modal locomotion strategy based on the Mini Cheetah [4] to increase the versatility of legged robots. The robot therefore could walk quadrupedally following the same MPC strategy proposed by Carlo et al. [9] and walk bipedally with our proposed learning-based control method, while the same body structure and actuators are used for both locomotion modes. We realise this through a reconfiguration design, learning-based bipedal control methods, and a sim-to-real transfer strategy.

Chen Yu and Andre Rosendo are with School of Information Science and Technology, ShanghaiTech University, Shanghai, China {yuchen, arosendo}@shanghaitech.edu.cn

Multi-Modal Locomotion. Fukuda et al. [10] design a multi-locomotion robot, Gorilla Robot, inspired by male gorillas that can perform bipedal walking, quadruped walking, and brachiation. They propose a method called Passive Dynamic Autonomous Control and a corresponding gait selection strategy [11]. Huang et al. [12] design a crawling strategy for a biped walking with a rigid-flexible waist using CPG control. Earlier works involve controlling humanoid robots to crawl with hands and feet [13] or hands and knees [14], [15], [16]. However, these robots are either mechanically complicated or energy-consuming. For instance, the 22kg Gorilla Robot is powered by 24 AC motors of 20-30W. In addition, some of them walk relatively slowly and unstably because of their application of moving in narrow spaces.

Bipedal Locomotion Control. The locomotion control for multi-modal robots is more challenging than for robots with a single locomotion strategy due to the potential conflict of mechanical and dynamics model designs for each mode of locomotion. The difficulty of converting a quadruped robot into a biped comes from 1) a higher limitation on load capacity per leg, as the full weight falls in two limbs, and 2) the small foot area with a limited degree of freedom of the foot to accommodate the potentially conflicting configuration space of the quadruped locomotion. Control of a bipedal robot is usually based on a simplified dynamics model, such as Linear Inverted Pendulum, and after keeping the kinematics consistent, physically feasible trajectories are usually designed within a zero-moment-point (ZMP) constraint [17]. Adjusting the ZMP dynamically during locomotion can make the walking process more robust [18], [19], while another popular class of control algorithms use Reinforcement Learning (RL) to search for a policy without any assumption on the dynamics model. It is shown that model-free RL is able to solve complicated 3D humanoid control tasks [20], [21], [22].

Sim-to-Real Transfer. Although RL shows promising results in simulation, unfortunately, capabilities demonstrated by simulated agents can hardly be realized by their physical counterparts due to the reality gap[23]. It is shown that this gap could be bridged by 1) accurate identification of simulated models [24], [25], [26], 2) training robust policies [27], [23], [28], or 3) fine-tuning a trained policy with real-world data [29], [30], [31].

We use a mechanical structure to provide a supporting polygon for bipedal mode, which does not affect the quadruped locomotion controller. A static action sequence is designed to enable the robot to switch smoothly between

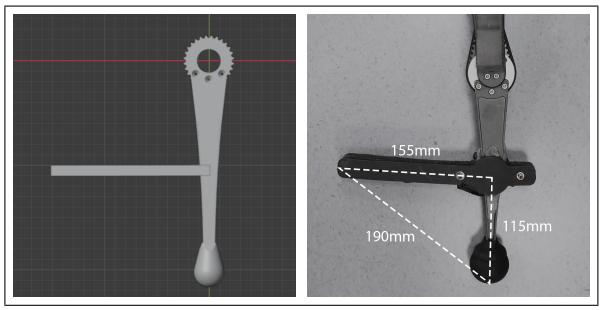

Fig. 1: A demonstration of the installed supporting stick in the 3D design software (left) and the real robot (right).

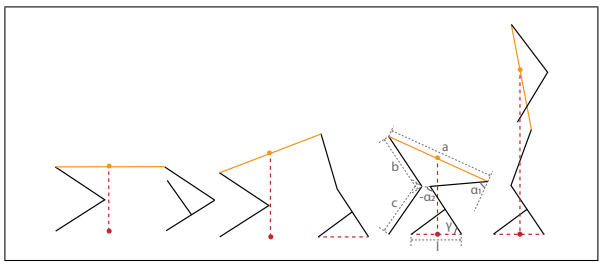

Fig. 2: An 2D illustration of the action sequence for the transition from quadruped mode to bipedal mode, where the orange line stands for the torso, the grey lines stand for legs, and the red lines demonstrate the projection of the CoM of the robot. During the transformation between the third state and the fourth, the CoM of the robot is always projected on the center of the equivalent feet.

two modes. Based on our unique mechanical and application constraints, we consider and evaluate a variety of learning-based algorithms: 1) a structured controller with parameters optimised by black-box optimisers; 2) model-free RL methods, and 3) our proposed hybrid strategy which combines those two. To improve the sim-to-real transferability, we propose a technique to narrow the reality gap where the action amplitude is progressively increased from zero when the robot starts to walk.

The contribution of this paper is 1) the design of a multi-modal legged locomotion strategy, 2) the proposal of a novel class of hybrid residual RL algorithms, and 3) the proposal of a sim-to-real technique.

II. Multi-Modal Transition

A. Mechanical Design

We design a 3D-printed stick and installed it on the shank of the hind legs of the original quadruped robot. The dimensions are designed to make sure that the robot has a) enough configuration space for quadruped locomotion in its quadruped mode and b) enough supporting convex polygons area for robust locomotion control in its bipedal mode. The design is illustrated in Fig. 1.

B. Multi-Modal Transformation

The action sequence of multi-modal transformation is demonstrated in Fig. 2. The mode transition starts from a state as a normal quadruped robot where the added supporting structure will not affect its quadruped locomotion. The hind legs of the robot will then bend to a position that all the legs and the supporting sticks are touching the floor. The hands of the front feet of the robot then move toward the hind legs horizontally, which is done by calculating the inverse kinematics (IK). Finally, in the standing phase of the transformation, the projection of the CoM of the robot on the floor is kept at the centre of the equivalent footprint. Denoting the position of hip joint as α_1 and position of knee joint as α_2, we have an equation according to IK:

$$\alpha_1 = -\frac{\pi}{2} - \alpha_2 + \gamma - \arccos\left(\frac{(l/2 - b\cos(\alpha_2 - \gamma) - c\cos\gamma)}{ak}\right), \quad (1)$$

where a is the length of the torso, b is the length of thigh, c is the length of the shank, γ is the angle between the shank and the ground when both the toe and the installed stick is touching the floor, and k is a real number describing the position of the CoM of the robot projected on the torso.

III. Learning-Based Bipedal Control

To solve the locomotion control problem of the robot in the bipedal mode, we propose and evaluate a variety of learning-based algorithms.

A. Preliminaries

The control problem can be formularised as a continuous state space Markov Decision Process (MDP), written as a tuple $M = (\mathcal{S}, \mathcal{A}, r, \mathcal{P}, \gamma, d_0)$, where $\mathcal{S} \in \mathcal{R}^{d_s}$ and $\mathcal{A} \in \mathcal{R}^{d_a}$ are the sets of states and actions; $r_t(\mathbf{s}_t, \mathbf{a}_t, \mathbf{s}_{t+1})$ is the reward function describes the objective of the task. $\mathcal{P}(\cdot|\mathbf{s}_t, \mathbf{a}_t)$ is the transition probability distribution, $\gamma \in [0, 1)$ is the discounting factor, and d_0 is the distribution of the initial state . The objective of an RL algorithm is to learn a policy that maximises the total rewards within an episode.

B. Off-Policy Reinforcement Learning

RL methods usually estimate the Q-value, which recursively evaluates the expected discounted return after taking action $\mathbf{a}$ at state $\mathbf{s}$. It can be described by the Bellman equation:

$$Q(\mathbf{s}_t, \mathbf{a}_t) = \mathcal{E}_{\mathbf{s}_{t+1}}\left[r_t + \gamma \max_{\mathbf{a}_{t+1}} Q(\mathbf{s}_{t+1}, \mathbf{a}_{t+1})\right]. \quad (2)$$

In this paper we specifically consider two state-of-the-art off-policy RL algorithms that optimise the Bellman equation by sampling off-policy transitions $(\mathbf{s}_t; \mathbf{a}_t; r_t; \mathbf{s}_{t+1})$ from a replay buffer.

Soft Actor Critic (SAC). SAC is an stochastic off-policy actor-critic reinforcement learning algorithm [20]. It would maximise the entropy of policy besides the expected reward from the environment during training, which encourages state exploration of the agent.

Twin Delayed Deep Deterministic Policy Gradients (TD3). TD3 is a deterministic off-policy actor-critic reinforcement learning algorithm based on DDPG [32], [33]. It uses two Q-functions, delayed policy update, and noise on the target action to improve the performance over baseline DDPG.

In our problem, the state $\mathcal{S}$ include the roll, pitch, yaw , the angular velocities of the torso in the world frame along

these three axes, and the eight motor angles (two for front abduction joints, four for all hip joints, and two for hind knee joints). For the action $\mathcal{A}$, we choose to use the position control mode of the actuators which is shown to be more controllable for RL [34]. It controls two front abduction joints, all the four hip joints, and two hind knee joints. More specifically, the actions from the policy π_θ are served as an incremental angle of a joint after multiplying by a real number k_a. We design the reward function inspired by previous related work [35], [36] as followed:

$$r = \text{height} - 0.25\text{max}(|\mathbf{a}_{\text{front}}|) + \text{distance} - \text{cost}_l + 0.5, \quad (3)$$

where $\mathbf{a}_{\text{front}}$ represents the actions for the front legs and $cost_l = 0.1$ when an action is reaching the joint limit otherwise $cost_l = 0$. 0.5 is a bonus for living. This reward function encourages the agent to walk as far as possible with lowest possible energy consumption. An episode ends when the e_{pitch} in (5) > 0.65 or the number of time steps reaches the maximum (1000). We score an episode with T steps by $\sum_{t=0}^{T-1} r_t$. In this paper, all the reward function, stop condition, and the scoring rule are consistent between simulation and real-world experiment.

C. Residual Reinforcement Learning

We use the strategy of residual RL [37] to speed up the training process. The action a_t is chosen by additively combining a model-free RL policy $\pi_\theta(\mathbf{a}_t|\mathbf{s}_t)$ and a simpler parametric policy $\pi_{\theta'}(\mathbf{s}_t)$:

$$\mathbf{a}'_t = \pi_\theta(\mathbf{s}_t) + \pi_{\theta'}(\mathbf{s}_t). \quad (4)$$

We assume a simple dynamics model and use two PD controllers, which serves as part of the basic controllers $\pi_{\theta'}$ of our experiments, to utilise the front legs to compensate the movement of CoM. One PD controller to control the hip joints according to the pitch error $e_{pitch}(t)$ and the other to control the abduction joints according to the yaw error $e_{yaw}(t)$. The PD controller equation is

$$u_{\text{hip}} = K_{pp} \cdot e_{\text{pitch}}(t) + K_{dp} \cdot \frac{de_{\text{pitch}}(t)}{dt} \quad (5)$$

$$u_{\text{abduction}} = K_{py} \cdot e_{\text{yaw}}(t) + K_{dy} \cdot \frac{de_{\text{yaw}}(t)}{dt}. \quad (6)$$

Here the referenced pitch is the pitch of the torso when the robot just finishes the multi-modal transformation described in (1) and the referenced yaw is 0. u_{hip} and $u_{abduction}$ are the incremental angles of the front hip joints and abduction joints. $\{K_{pp}, K_{dp}, K_{py}, K_{dy}\}$ is a subset of parameters π' for controller $\pi_{\theta'}$.

Another part of the basic controllers $\pi_{\theta'}$ is an open-loop controller to generate gait patterns. Denote the position of hip and knee joints of a leg as $\alpha = [\alpha_1, \alpha_2]$ We design four gaits as followed.

Line. We design a baseline gait that does not step forward, which changes the angles of the hip and knee joins in different direction with the same amplitude.

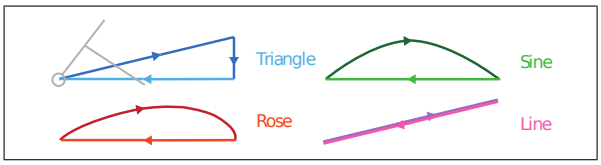

Fig. 3: Four gait patterns for an open-loop leg controller.

Sine. We design a gait trajectory that follows the pattern of half a period of sine function as followed.

$$\alpha = \begin{cases} IK(x_0 + \omega t \delta_x/\pi, y_0 + A\sin(\omega t)), & t < \pi/\omega, \\ IK(x_0 + \delta_x - \omega t \delta_x/\pi, y_0), & t \geq \pi/\omega \end{cases} \quad (7)$$

Rose. We also design a gait trajectory that follows the pattern of part of a rose function. We calculate $\alpha = IK(x_t, y_t)$ by

$$x_t = \begin{cases} x_0 + \delta_x \cos(2\alpha_p)\cos(\alpha_p), & t < \pi/\omega, \\ x_0 + \delta_x - (t - \pi/\omega) \cdot b\delta_x/\pi, & t \geq \pi/\omega \end{cases} \quad (8)$$

$$y_t = \begin{cases} y_0 + 4A/\delta_x \cos(2\alpha_p)\sin(\alpha_p), & t < \pi/\omega, \\ y_0, & t \geq \pi/\omega \end{cases} \quad (9)$$

$$\alpha_p = \begin{cases} \omega/4 \cdot (\pi/\omega - t), & t < \pi/\omega, \\ \omega/4 \cdot (2\pi/\omega - t), & t \geq \pi/\omega. \end{cases} \quad (10)$$

Triangle. This trajectory moves the toe upward, forward, and returning to the original spot, drawing a triangle pattern. We calculate $\alpha = IK(x_t, y_t)$ by

$$x_t = \begin{cases} x_0 + \frac{\sin(2\omega t - \frac{\pi}{2})+1}{2} \cdot (x_1 - x_0), & t < \frac{\pi}{2\omega}, \\ x_1 - \frac{\sin(2\omega t - \frac{\pi}{2})+1}{2} \cdot (x_2 - x_1), & \frac{\pi}{2b} \leq t < \frac{\pi}{\omega}, \\ x_0 + \delta_x - (t - \pi/\omega) \cdot \omega\delta_x/\pi, & t \geq \frac{\pi}{\omega} \end{cases} \quad (11)$$

$$y_t = \begin{cases} y_0 + \frac{\sin(2\omega t - \pi/2)+1}{2} \cdot (y_1 - y_0), & t < \pi/(2\omega), \\ y_0, & t \geq \pi/(2\omega). \end{cases} \quad (12)$$

For all the gait patterns, two legs perform with a phase shift of half a period. An illustration of these gait patterns is shown in Fig. 3.

The controllers for the front legs and hind legs described above constitute the basic controller $\pi_{\theta'}$ for our experiment, where $\theta' = \{A/\omega, \omega, K_{pp}, K_{dp}, K_{py}, K_{dy}, \delta_x\}$. The range for the parameter A/ω is $[8, 11]$ for the triangle gait and $[0, 3]$ for others. The ranges for K_{pp}, K_{dp}, K_{py}, and K_{dy} are all $[0.00, 0.1]$. The range for δ_x is $[0, 0.05]$. All the parameters are randomly initialised before the training process.

D. Evolution Strategy and Bayesian Optimisation

Instead of hand-tuning the parameter θ' for the additive parametric policy $\pi_{\theta'}$, we consider a few black-box optimisers to find the optimum of these parameters during the training process.

Covariance Matrix Adaptation Evolution Strategy (CMAES). CMAES is an evolutionary algorithm with an "evolutionary path" storing the update direction of generations [38]. Previous works have shown its efficiency in control [39], [40], [41].

Test-Based Population Size Adaptation (TBPSA). TBPSA is a specific implementation of pcCMAES [42], a variant of CMAES. It evaluates points with strong mutation rate and perform small steps in the best direction, relying on the longer-range trends of the objective landscape [43].

Bayesian Optimisation (BO). BO is a global optimiser which uses observations to form the posterior distribution over the objective function [44]. It has been widely used in robot control due to its sample efficiency [45], [46], [47].

These optimisers search for the optimal parameters θ'^* based on the training rewards (3) averaged over a horizon of H episodes:

$$\theta'^* = \underset{\theta'}{\operatorname{argmax}} \frac{1}{H} \sum_{n=0}^{H-1} \sum_{t=0}^{T-1} r_t. \tag{13}$$

Note that this evaluation is based on the returns from the training policy π'_θ of an off-policy RL agent, which means that the optimisation process considers both exploration and exploitation of the RL agent. The proposed hybrid algorithm is shown in Alg. 1.

Algorithm 1 Hybrid Residual RL

Require: RL policy π_θ, basic parametric controller $\pi_{\theta'}$

while $t_{\text{total}} \leq t_{\max}$ total steps **do**
  for $n = 0, \ldots, H-1$ episodes **do**
    Sample initial state $\mathbf{s}_0 \sim d_0$.
    for $t = 0, \ldots, T-1$ steps **do**
      Obtain policy action $\mathbf{a}_t \sim \pi_\theta$.
      Calculate action to execute $\mathbf{a}'_t = \mathbf{a}_t + \pi_{\theta'}(\mathbf{s}_t)$.
      Transfer to next state $\mathbf{s}_{t+1} \sim \mathcal{P}(\cdot|\mathbf{s}_t, \mathbf{a}_t)$.
      Store $(\mathbf{s}_t, \mathbf{a}_t, \mathbf{s}_{t+1})$ into the replay buffer $\mathcal{R}$.
      Optimise θ with transitions $(\mathbf{s}, \mathbf{a}, \mathbf{s}) \sim \mathcal{R}$.
    end for
    Calculate return $R_n = \sum_{t=0}^{T-1} r_t$ for episode n.
  end for
  Optimise θ' through (13).
end while

E. Sim-to-Real Transfer

Beyond implementing the trained policy directly to the real robot, we gradually increased the parameter A of the controller $\pi_{\theta'}$ for each episode while satisfying the condition that errors e_{pitch} and $\frac{de_{\text{pitch}}}{dt}$, defined in (5), should always be below a threshold. We also progressively increased the weight of policy $\pi_t heta$ until 1. Specifically, we calculate the error based on the absolute values of $e_{\text{pitch}}(t)$ and $\frac{de_{\text{pitch}}(t)}{dt}$, averaged over a period. Denoting the two errors as $\mathbf{e}$, it yields:

$$A = \begin{cases} k_1 t, & k_1 t < A_{\text{set}} \quad \text{and} \quad \mathbf{e} < \mathbf{e}_{\text{threshold}}, \\ A, & k_1 t < A_{\text{set}} \quad \text{and} \quad \mathbf{e} \geq \mathbf{e}_{\text{threshold}}, \\ A_{\text{set}}, & \text{otherwise} \end{cases} \tag{14}$$

$$a_t = \begin{cases} \pi_\theta(s_t) + k_2 t \pi_{\theta'}(s_t), & k_2 t < 1 \text{ and } \mathbf{e} < \mathbf{e}_{\text{threshold}}, \\ \pi_\theta(s_t) + k_2 t' \pi_{\theta'}(s_t), & k_2 t < 1 \text{ and } \mathbf{e} \geq \mathbf{e}_{\text{threshold}}, \\ \pi_\theta(s_t) + \pi_{\theta'}(s_t), & \text{otherwise}, \end{cases} \tag{15}$$

where $k_2 t'$ denotes the last updated $k_2 t$. This method could help the closed-loop described in (5) to calculate a good upper arms positioning before the full actions of legs are performed, which greatly improves the balance and the sim-to-real transferability.

IV. Results

A. Multi-Modal Transition

The multi-modal transition strategy described in Sec. II and a reverse action sequence can both perform well on the real robot, as shown in Fig. 4 and the supplementary video. In this way, the robot can arbitrarily switch between these two locomotion modes.

B. Bipedal Locomotion Control

The returns calculated by accumulated rewards (3) during the training process of the bipedal locomotion with different gaits are shown in Fig. 5. All the results are averages of three trials. We compare the training curves of different control methods including 1) only using parametric controller $\pi_{\theta'}$ trained by block-box optimisers introduced in Sec. III-D, 2) only using model-free RL algorithms introduced in Sec. III-B to control the abduction joints of the front legs and all hips and knees joints, and 3) using the hybrid algorithms described in Alg. 1. We show the training curves of the pure RL algorithms in Fig. 6 and only their highest scores in Fig. 5 since there is no gait pattern designed for the pure RL cases. For the Sine gait, returns are noticeably distinguished as two clusters: three hybrid methods involving TD3 have similar returns as the pure TD3, with the combination of TD3 and BO being the best, and other methods are stuck in a local maximum with a return around 1000. For the Rose gait, all six hybrid methods have a better performance than all the standard black-box parameter optimisers. The return of the combination TD3+CMAES also surpasses the highest score obtained by the standard TD3. As for the Triangle gait, the returns from three hybrid methods involving TD3 converged to the highest score of standard TD3, while TD3+BO achieves the highest. All standard optimisers have returns similar to or lower than standard SAC.

In term of motion primitives, the design of the gait patterns is essential for this task. For comparison, we evaluate the performances of different algorithms with a gait that does not step forward (Line gait) as shown in Fig. 7. In this case, methods involving the Line gait designs have similar performances which are much worse than that of those using other gait patterns. This reveals the dependency between the effectiveness of our hybrid residual RL methods and the design of motion primitives. Among the Sine, Rose, and Triangle gaits, the highest average return occurs in the Sine gait (at 3259.01, obtained by TD3+BO).

Fig. 4: Snapshots of the transition between bipedal and quadruped modes. In the quadruped mode, the additional structure does not affect the previous locomotion controller, while in the bipedal mode the supporting structure successfully provides a stability polygon for the robot to keep bipedal balance.

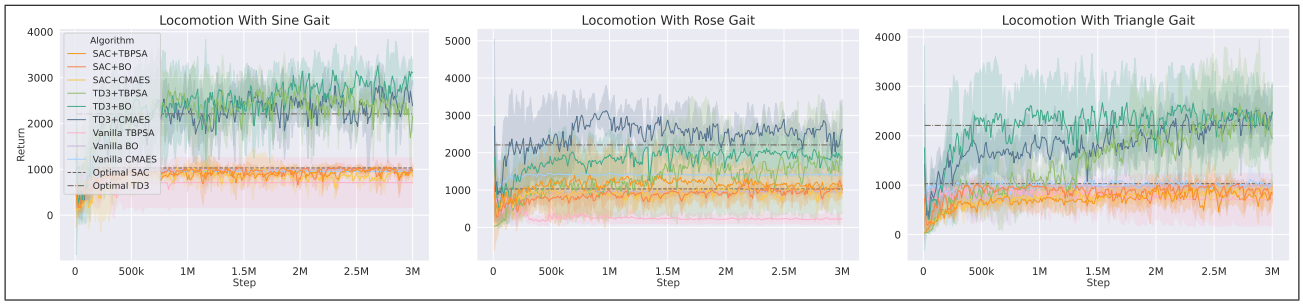

Fig. 5: Training curves of our robot in bipedal mode trained with different algorithms using Sine, Rose, and Triangle gait patterns. For all these gaits, the returns from three hybrid methods involved TD3 are converging to the highest score of TD3, while TD3+BO achieves the highest in the Sine and Triangle cases, and TD3+CMAES achieves the best score in the Rose case. All the other hybrid methods and pure black-box optimisation methods have returns similar to or even lower than that of SAC.

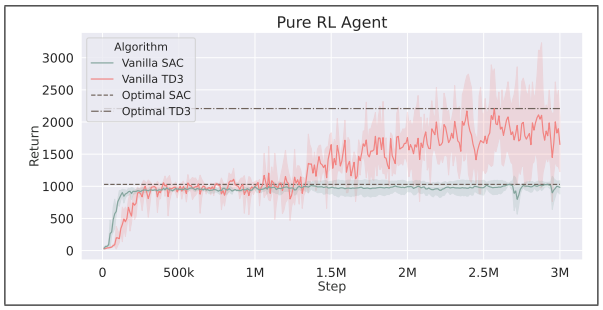

Fig. 6: The training curves of the robot in bipedal locomotion mode using pure RL. During the training process, the agent of SAC reaches an optimum score at 1029.52 and the agent from TD3 obtains a maximum return of 2209.23.

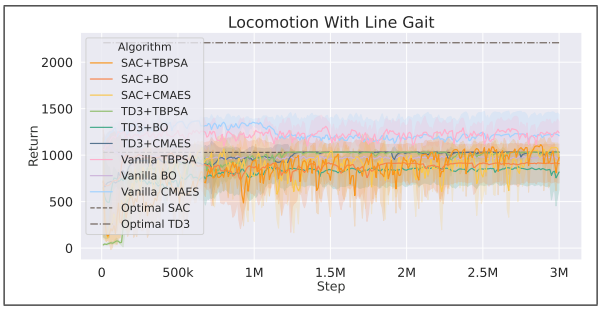

Fig. 7: The training curve for the baseline Line gait. Performances here are much worse than those using other gait patterns. This shows the importance of the design of gait patterns (motion primitives) for our hybrid residual RL methods.

The performance of our hybrid methods depends heavily on the choice of RL agents. For two standard RL agents, it is shown in Fig. 6 that the training of SAC agents is stuck in a local maximum in this task compared with TD3. This happens when SAC not only tries to optimise the discounted accumulated rewards but also the entropy of the policy, which adds too much noise to the optimisation of the Bellman equation (2) in this specific task setting. Therefore, the combination of SAC policy and parametric policy $\pi_{\theta'}$ also does not work well in our task. On the other hand, the combination of TD3 policy and parametric policy $\pi_{\theta'}$ shows the superiority of our proposed hybrid strategy, which can find a better policy than pure TD3 independent of which gait is chosen.

In terms of parameter optimisers, all of them can synergise with the learning process of RL agents within our hybrid framework. For three standard optimisers, it is shown that vanilla CMAES has a better performance than standard BO and TBPSA in this task. However, while they are combined with RL agents, TD3+BO outperforms TD3+CMAES for Sine and Triangle gaits. This shows that the Gaussian process behind BO can manage the exploration-exploitation trade-off well during its cooperation with RL training.

Overall, standard TD3 can outperform SAC and all the parametric controllers with parameter optimisers, while our proposed hybrid methods with TD3 can even surpass the standard TD3. Furthermore, the results show that TD3+BO is a good choice for the Sine gait and the Triangle gait, while TD3+CMAES is a good choice for the Rose gait.

C. Sim-to-Real Transfer

Firstly, we implement the policies trained by each method directly on the real robot. Snapshots of the walking of simulated and real robots are shown in Fig. 8. Returns of each algorithm are shown in Tab. I with the same scoring metrics (accumulation of (3)) as in the simulation (also averaged over three trials). The results show a significant reality gap

Fig. 8: Snapshots of the bipedal walking in the simulation (left) and real world (right).

TABLE I: Scores of bipedal walking in real-world experiment with direct policy implementation

	Sine	Rose	Triangle	Line
SAC+TBPSA	56.87	24.10	34.27	78.29
SAC+BO	30.10	32.48	22.03	48.24
SAC+CMAES	22.54	40.83	32.50	33.70
TD3+TBPSA	30.35	29.81	21.72	40.54
TD3+BO	22.49	25.28	20.45	28.96
TD3+CMAES	32.72	30.77	29.07	22.01
Vanilla TBPSA	351.51	86.85	57.03	786.87
Vanilla BO	473.95	511.02	350.80	760.31
Vanilla CMAES	527.75	398.56	80.95	780.43
Vanilla SAC	31.9			
Vanilla TD3	31.23			

TABLE II: Scores of bipedal walking in a real-world experiment with the proposed sim-to-real strategy and the corresponding improvement rates

	Sine	Rose	Triangle	Line
SAC+TBPSA	86.89 (52.8%)	39.05 (62.0%)	253.00 (638.3%)	77.62 (-0.9%)
SAC+BO	105.10 (249.2%)	58.79 (81.0%)	237.91 (979.9%)	292.01 (505.3%)
SAC+CMAES	68.90 (205.7%)	105.93 (159.4%)	237.42 (630.5%)	85.46 (153.6%)
TD3+TBPSA	70.10 (131.0%)	62.83 (110.8%)	269.79 (1142.1%)	83.93 (107.0%)
TD3+BO	74.40 (230.8%)	64.32 (154.4%)	188.63 (822.4%)	103.72 (258.1%)
TD3+CMAES	89.93 (174.8%)	71.32 (131.8%)	269.92 (828.5%)	62.63 (184.6%)
Vanilla TBPSA	622.86 (77.2%)	464.04 (434.3%)	692.26 (1113.9%)	785.08 (-0.2%)
Vanilla BO	614.79 (29.7%)	599.63 (17.3%)	735.90 (109.8%)	760.69 (0.0%)
Vanilla CMAES	592.07 (12.2%)	872.04 (118.8%)	661.76 (717.5%)	797.75 (2.2%)
Vanilla SAC	59.95 (87.9 %)			
Vanilla TD3	55.31 (77.1 %)			

especially in the case of pure RL. For instance, while the pure TD3 agent in the simulator can obtain a return of 2209.23, it can only reach a return of 31.23 in the real world.

We then implement our sim-to-real strategy described in Sec. III-E on the robot. The values of $\mathbf{e}_{\text{threshold}}$ in (14, 15) in our experiment are 0.15 and 0.35 for e_{pitch} and $\frac{de_{\text{pitch}}}{dt}$. Returns of each algorithm are shown in Tab. II with the improvement rates shown in brackets. Comparison between returns with and without the progressive control strategy reveals that our sim-to-real technique can significantly improve the walking control performance for most of the cases. For instance, the performance of using the hybrid algorithm TD3+TBPSA with the Triangle gait is improved by 1142.1% after using the progressing control. A comparison among various algorithms show that pure black-box optimisers (Vanilla TBPSA, BO, and CMAES) were capable of reaching the highest scores. Although the hybrid methods are not the best after the sim-to-real transfer, their average performances are still significantly better than pure RL methods (Vanilla TD3 and Vanilla SAC), which shows a competitive sim-to-real transferability. As for the comparison among different motion primitives, the Triangle gait shows its advantage in the sim-to-real transfer. This can be explained by the design principle of the Triangle gait where each foot will first leave the floor horizontally to the ground. This not only creates a margin for positional error, but also allows its deformation without affecting the impact position between the feet and the floor.

V. DISCUSSION AND FUTURE WORK

Multi-modal transition results show that it is possible to explore the versatility of legged robots using only simple mechanical modifications. In comparison to other robots with multiple locomotion modes [12], [11], our robot is lightweight and has a lower cost. This is a step toward commercialising multi-locomotion robots and bringing them into our daily life. This points to the possibility of giving other robots a multi-locomotion ability with similar simple mechanical modifications.

Training results reveal the effectiveness of combining residual RL algorithms and parametric controllers with optimisers. Previous combinations of parameter optimisers and RL are hardly implemented in such a residual RL setting. They either use black-box optimisers for neural architecture search [48], [49], or for enhancement of the update of the RL policy itself [50], [51], [52].

Sim-to-real results suggest that the real dynamics model is complex, which limits the performance of our algorithms. Experiments show that the real robot can walk with the Line gait even further than with others. This can only happen with the deformation of the legs and the inaccuracy of the motor positions, which prevents the feet position from following the preset trajectories. Therefore, the conservative standard parameter optimisers still outstand in the real world, which are hence adopted in our video. Improving our robot and our algorithms to narrow this reality gap is the next step of research, along with mechanical adaptations to explore manipulation with the forelimbs.

VI. CONCLUSIONS

In this paper we propose a hybrid RL for a biped-quadruped robot. After proving the feasibility of our mechanical design for multi-modal locomotion, we evaluate novel hybrid algorithms which use parameter optimisers (CMAES, TBPSA, or BO) during the training of a residual RL agent (SAC or TD3). Our results show their superiority over standard RL methods and standard parameter optimisers. Our control technique improves the performances of all methods by at most 1142.1% on the real robot. In the future we will further improve the performance of our methods on the real robot, and consider manipulation tasks with the same robot.

REFERENCES

[1] J. He, J. Shao, G. Sun, and X. Shao, "Survey of quadruped robots coping strategies in complex situations," *Electronics*, vol. 8, no. 12, p. 1414, 2019.

[2] M. Raibert, K. Blankespoor, G. Nelson, and R. Playter, "Bigdog, the rough-terrain quadruped robot," *IFAC Proceedings Volumes*, vol. 41, no. 2, pp. 10 822–10 825, 2008.

[3] "Spot: Boston dynamics," https://www.bostondynamics.com/spot, accessed: 2021-09-09.

[4] W. Bosworth, S. Kim, and N. Hogan, "The mit super mini cheetah: A small, low-cost quadrupedal robot for dynamic locomotion," in *2015 IEEE International Symposium on Safety, Security, and Rescue Robotics (SSRR)*, 2015, pp. 1–8.

[5] M. Hutter, C. Gehring, D. Jud, A. Lauber, C. D. Bellicoso, V. Tsounis, J. Hwangbo, K. Bodie, P. Fankhauser, M. Bloesch, R. Diethelm, S. Bachmann, A. Melzer, and M. Hoepflinger, "Anymal - a highly mobile and dynamic quadrupedal robot," in *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2016, pp. 38–44.

[6] S. Saeedvand, M. Jafari, H. S. Aghdasi, and J. Baltes, "A comprehensive survey on humanoid robot development," *The Knowledge Engineering Review*, vol. 34, p. e20, 2019.

[7] S. Kuindersma, R. Deits, M. Fallon, A. Valenzuela, H. Dai, F. Permenter, T. Koolen, P. Marion, and R. Tedrake, "Optimization-based locomotion planning, estimation, and control design for the atlas humanoid robot," *Autonomous robots*, vol. 40, no. 3, pp. 429–455, 2016.

[8] Y. Gong, R. Hartley, X. Da, A. Hereid, O. Harib, J.-K. Huang, and J. Grizzle, "Feedback control of a cassie bipedal robot: Walking, standing, and riding a segway," in *2019 American Control Conference (ACC)*. IEEE, 2019, pp. 4559–4566.

[9] J. Di Carlo, P. M. Wensing, B. Katz, G. Bledt, and S. Kim, "Dynamic locomotion in the mit cheetah 3 through convex model-predictive control," in *2018 IEEE/RSJ international conference on intelligent robots and systems (IROS)*. IEEE, 2018, pp. 1–9.

[10] T. Fukuda, Y. Hasegawa, K. Sekiyama, and T. Aoyama, *Multi-locomotion robotic systems: New concepts of bio-inspired robotics*. springer, 2012, vol. 81.

[11] T. Kobayashi, T. Aoyama, M. Sobajima, K. Sekiyama, and T. Fukuda, "Locomotion selection strategy for multi-locomotion robot based on stability and efficiency," *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 2616–2621, 2013.

[12] Z. Huang, X. Jiang, H. Liu, X. Chen, T. Fukuda, and Q. Huang, "Design of crawling motion for a biped walking humanoid with 3-dof rigid-flexible waist," in *2018 IEEE-RAS 18th International Conference on Humanoid Robots (Humanoids)*. IEEE, 2018, pp. 974–979.

[13] K. Nishiwaki, J. Kuffner, S. Kagami, M. Inaba, and H. Inoue, "The experimental humanoid robot h7: a research platform for autonomous behaviour," *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, vol. 365, no. 1850, pp. 79–107, 2007.

[14] H. Sanada, E. Yoshida, and K. Yokoi, "Passing under obstacles with humanoid robots," in *Experimental Robotics*. Springer, 2009, pp. 283–291.

[15] F. Kanehiro, T. Yoshimi, S. Kajita, M. Morisawa, K. Kaneko, H. Hirukawa, and F. Tomita, "Whole body locomotion planning of humanoid robots based on a 3d grid map," *Journal of the Robotics Society of Japan*, vol. 25, no. 4, pp. 589–597, 2007.

[16] F. Kanehiro, H. Hirukawa, K. Kaneko, S. Kajita, K. Fujiwara, K. Harada, and K. Yokoi, "Locomotion planning of humanoid robots to pass through narrow spaces," in *IEEE International Conference on Robotics and Automation, 2004. Proceedings. ICRA'04. 2004*, vol. 1. IEEE, 2004, pp. 604–609.

[17] M. Vukobratovic and D. Juricic, "Contribution to the synthesis of biped gait," *IEEE Transactions on Biomedical Engineering*, no. 1, pp. 1–6, 1969.

[18] J.-Y. Kim, I.-W. Park, and J.-H. Oh, "Experimental realization of dynamic walking of the biped humanoid robot khr-2 using zero moment point feedback and inertial measurement," *Advanced Robotics*, vol. 20, no. 6, pp. 707–736, 2006.

[19] K. Kaneko, F. Kanehiro, S. Kajita, K. Yokoyama, K. Akachi, T. Kawasaki, S. Ota, and T. Isozumi, "Design of prototype humanoid robotics platform for hrp," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, vol. 3. IEEE, 2002, pp. 2431–2436.

[20] T. Haarnoja, A. Zhou, P. Abbeel, and S. Levine, "Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor," in *International conference on machine learning*. PMLR, 2018, pp. 1861–1870.

[21] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal policy optimization algorithms," *arXiv preprint arXiv:1707.06347*, 2017.

[22] Z. Wang, V. Bapst, N. Heess, V. Mnih, R. Munos, K. Kavukcuoglu, and N. de Freitas, "Sample efficient actor-critic with experience replay," in *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net, 2017.

[23] X. B. Peng, M. Andrychowicz, W. Zaremba, and P. Abbeel, "Sim-to-real transfer of robotic control with dynamics randomization," in *2018 IEEE international conference on robotics and automation (ICRA)*. IEEE, 2018, pp. 3803–3810.

[24] J. Tan, T. Zhang, E. Coumans, A. Iscen, Y. Bai, D. Hafner, S. Bohez, and V. Vanhoucke, "Sim-to-real: Learning agile locomotion for quadruped robots," 06 2018.

[25] J. Hwangbo, J. Lee, A. Dosovitskiy, D. Bellicoso, V. Tsounis, V. Koltun, and M. Hutter, "Learning agile and dynamic motor skills for legged robots," *Science Robotics*, vol. 4, no. 26, 2019.

[26] F. Golemo, A. A. Taiga, A. Courville, and P.-Y. Oudeyer, "Sim-to-real transfer with neural-augmented robot simulation," in *Conference on Robot Learning*. PMLR, 2018, pp. 817–828.

[27] K. Lowrey, S. Kolev, J. Dao, A. Rajeswaran, and E. Todorov, "Reinforcement learning for non-prehensile manipulation: Transfer from simulation to physical system," in *2018 IEEE International Conference on Simulation, Modeling, and Programming for Autonomous Robots (SIMPAR)*. IEEE, 2018, pp. 35–42.

[28] M. Andrychowicz, B. Baker, M. Chociej, R. Józefowicz, B. McGrew, J. Pachocki, A. Petron, M. Plappert, G. Powell, A. Ray, J. Schneider, S. Sidor, J. Tobin, P. Welinder, L. Weng, and W. Zaremba, "Learning dexterous in-hand manipulation," *The International Journal of Robotics Research*, vol. 39, no. 1, pp. 3–20, 2020.

[29] Y. Chebotar, A. Handa, V. Makoviychuk, M. Macklin, J. Issac, N. Ratliff, and D. Fox, "Closing the sim-to-real loop: Adapting simulation randomization with real world experience," in *2019 International Conference on Robotics and Automation (ICRA)*. IEEE, 2019, pp. 8973–8979.

[30] A. A. Rusu, M. Večerík, T. Rothörl, N. Heess, R. Pascanu, and R. Hadsell, "Sim-to-real robot learning from pixels with progressive nets," in *Conference on Robot Learning*. PMLR, 2017, pp. 262–270.

[31] T. Chen, A. Murali, and A. Gupta, "Hardware conditioned policies for multi-robot transfer learning," in *Advances in Neural Information Processing Systems*, S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, Eds., vol. 31. Curran Associates, Inc., 2018.

[32] S. Dankwa and W. Zheng, "Twin-delayed ddpg: A deep reinforcement learning technique to model a continuous movement of an intelligent robot agent," in *Proceedings of the 3rd International Conference on Vision, Image and Signal Processing*, 2019, pp. 1–5.

[33] T. Lillicrap, J. J. Hunt, A. Pritzel, N. Heess, T. Erez, Y. Tassa, D. Silver, and D. Wierstra, "Continuous control with deep reinforcement learning," *CoRR*, vol. abs/1509.02971, 2016.

[34] X. B. Peng and M. van de Panne, "Learning locomotion skills using deeprl: Does the choice of action space matter?" in *Proceedings of the ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, 2017, pp. 1–13.

[35] J. Schulman, P. Moritz, S. Levine, M. I. Jordan, and P. Abbeel, "High-dimensional continuous control using generalized advantage estimation," in *4th International Conference on Learning Representations, ICLR 2016, San Juan, Puerto Rico, May 2-4, 2016, Conference Track Proceedings*, Y. Bengio and Y. LeCun, Eds., 2016. [Online]. Available: http://arxiv.org/abs/1506.02438

[36] Y. Tassa, T. Erez, and E. Todorov, "Synthesis and stabilization of complex behaviors through online trajectory optimization," in *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, 2012, pp. 4906–4913.

[37] T. Johannink, S. Bahl, A. Nair, J. Luo, A. Kumar, M. Loskyll, J. A. Ojea, E. Solowjow, and S. Levine, "Residual reinforcement learning for robot control," in *2019 International Conference on Robotics and Automation (ICRA)*. IEEE, 2019, pp. 6023–6029.

[38] A. Auger and N. Hansen, "Tutorial cma-es: evolution strategies and covariance matrix adaptation," in *Proceedings of the 14th annual*

conference companion on Genetic and evolutionary computation, 2012, pp. 827–848.

[39] L. Goff, E. Buchanan, E. Hart, A. Eiben, W. Li, M. De Carlo, M. Hale, M. Angus, R. Woolley, J. Timmis, A. Winfield, and A. Tyrrell, "Sample and time efficient policy learning with cma-es and bayesian optimisation," 01 2020, pp. 432–440.

[40] A. Piergiovanni, A. Wu, and M. S. Ryoo, "Learning real-world robot policies by dreaming," in *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2019, pp. 7680–7687.

[41] K. Chatzilygeroudis, R. Rama, R. Kaushik, D. Goepp, V. Vassiliades, and J.-B. Mouret, "Black-box data-efficient policy search for robotics," in *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2017, pp. 51–58.

[42] M. Hellwig and H.-G. Beyer, "Evolution under strong noise: A self-adaptive evolution strategy can reach the lower performance bound-the pccmsa-es," in *International Conference on Parallel Problem Solving from Nature*. Springer, 2016, pp. 26–36.

[43] J. Liu, A. Moreau, M. Preuss, J. Rapin, B. Roziere, F. Teytaud, and O. Teytaud, "Versatile black-box optimization," in *Proceedings of the 2020 Genetic and Evolutionary Computation Conference*, 2020, pp. 620–628.

[44] J. Snoek, H. Larochelle, and R. P. Adams, "Practical bayesian optimization of machine learning algorithms," *Advances in neural information processing systems*, vol. 25, 2012.

[45] C. Mailer, G. Nitschke, and L. Raw, "Evolving gaits for damage control in a hexapod robot," in *Proceedings of the Genetic and Evolutionary Computation Conference*, 2021, pp. 146–153.

[46] R. Guzman, R. Oliveira, and F. Ramos, "Heteroscedastic bayesian optimisation for stochastic model predictive control," *IEEE Robotics and Automation Letters*, vol. 6, no. 1, pp. 56–63, 2020.

[47] J. Zhu, S. Li, Z. Wang, and A. Rosendo, "Bayesian optimization of a quadruped robot during 3-dimensional locomotion," in *Conference on Biomimetic and Biohybrid Systems*. Springer, 2019, pp. 295–306.

[48] Y. Chen, G. Meng, Q. Zhang, S. Xiang, C. Huang, L. Mu, and X. Wang, "Renas: Reinforced evolutionary neural architecture search," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 4787–4796.

[49] K. Maziarz, M. Tan, A. Khorlin, M. Georgiev, and A. Gesmundo, "Evolutionary-neural hybrid agents for architecture search," *arXiv preprint arXiv:1811.09828*, 2018.

[50] Pourchot and Sigaud, "CEM-RL: Combining evolutionary and gradient-based methods for policy search," in *International Conference on Learning Representations*, 2019.

[51] C. Colas, O. Sigaud, and P.-Y. Oudeyer, "Gep-pg: Decoupling exploration and exploitation in deep reinforcement learning algorithms," in *International conference on machine learning*. PMLR, 2018, pp. 1039–1048.

[52] L. Shi, S. Li, Q. Zheng, L. Cao, L. Yang, and G. Pan, "Maximum entropy reinforcement learning with evolution strategies," in *2020 International Joint Conference on Neural Networks (IJCNN)*. IEEE, 2020, pp. 1–8.