Meta Reinforcement Learning for Optimal Design of Legged Robots

presentor: 沈奕宁、
2024.3.19

仿生设计与学习实验室
Bionic Design & Learning Lab
In Collaboration with MIT
AncoraSIR.com
SUSTech
Southern University of Science and Technology

Main Problem

Unclear correlation between design parameters and robot behavior

Leg-Shape (A. Ananthanarayanan et al., 2012)

Table 3. Fixed leg dimensional parameters.

Parameter	Symbol	Vector	Value (mm)
Foot length	M	$\vec{OA}$	90
Radius length	R	$\vec{AB}$	240
Tricep extension length	L	$\vec{BC}$	62
Humerus	H	$\vec{BS}$	220
Tricep	T_r	$\vec{CT}$	100
Tricep connector length	C	$\vec{ST}$	48
Ladder lock connector	L	$\vec{CG}$	20

Mini Cheetah (B. Katz et al.,2019)

TABLE I: Actuator Specifications

Mass	440g
Dimensions	96 mm O.D., 40 mm axial length
Maximum Torque	17 N m
Continuous Torque	6.9 N m
Maximum Output Speed	40 rad/s@24 volts
Maximum Output Power	$+250/-680$watts
Current Control Bandwidth	4.5kHz@4.5N m, 1.5kHz@17N m
Output Inertia	0.0023 kg m^2

HyQ (C. Semini et al., 2011)

Table III Mass and inertia properties of the HyQ robot *LegV2*

Leg Segment/Part	Mass	Inertia
Leg-torso attachment	1.31kg	-
Electric motor	1.53kg	-
Hip assembly (with hip cylinder)	2.48kg	0.00675 kg m^2
Upper leg (with knee cylinder)	1.77kg	0.0704 kg m^2
Lower leg	1.48kg	0.0486 kg m^2
Foot	0.37kg	-
Total	8.94kg	-

Table V Technical specifications of the quadruped robot HyQ

Description	Value
Dimensions (fully stretched legs)	1.0m x 0.5m x 0.98m (Length x Width x Height)
Leg length (*hip a/a* axis to ground) (uncompressed spring)	from 0.339m (q_0=0°, q_1=-70°, q_2=140°, q_3=0m) to 0.789m (q_0=0°, q_1=-10°, q_2=20°, q_3=0m)
Distance of left to right *hip a/a* axis	0.414m
Distance of front to hind *hip f/e* axis	0.747m
Weight	70kg (external hydraulic system), 91kg (onboard hydraulic system)
Number of active DOF	12 (8 hydraulic and 4 electric)
Joint range of motion	120° (for each joint)
Hydraulic actuator type	double-acting cylinders (80mm stroke and 16mm bore)
Electric actuator type	DC brushless motor with harmonic gear (1:100)
Maximum torque (hydraulic)	145Nm (peak torque at Pmax=16MPa)
Maximum torque (electric)	140Nm (peak torque at nominal voltage)
Onboard sensors	joint position (relative and absolute), joint torque, cylinder pressure, foot spring compression, IMU
Onboard computer	PC104 Pentium, real-time Linux
Control frequency	1kHz

Table IV Geometric parameters of leg and hydraulic joint kinematics

Location	Parameter	Value
Leg	l_0	0.08m
	l_1	0.35m
	l_2	0.35m
	l_3	0.02m
hip a/a	q_0	range: [-90° to +30°]
hip f/e	a_1	0.322m
	b_1	0.045m
	c_1	see equation (2)
	e_{11}	6.24°
	L_{eff1}	see equation (5)
	q_1	range: [-70° to +50°]
knee f/e	a_2	0.322m
	b_2	0.045m
	c_2	see equation (6)
	e_{21}	8.04°
	e_{22}	6.0°
	L_{eff2}	see equation (7)
	q_2	range: [20° to 140°]
ankle (passive)	q_3	range: [-0.035m to 0m]

AncoraSIR.com

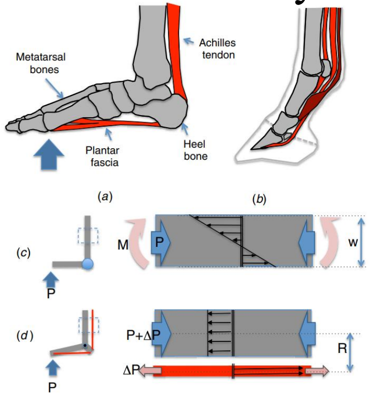
Limitations of Prior Work Ⅰ

In conventional paradigm of design optimization

Sparse principle in conventional robotic design

- Bio-inspired Leg

(A. Ananthanarayanan et al., 2012)



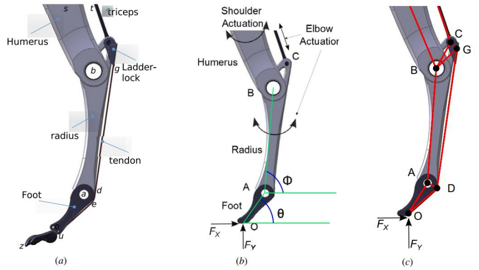


Figure 2. (a) The leg design of the MIT Robotic Cheetah. The parts undertaking tensions are made of high-strength material for minimizing bending on the bone. (b) Parameters indicated without tendon. (c) Tendon–bone co-location design. The red lines represent an equivalent pin-jointed structure.

AncoraSIR.com

- Cheetah Leg Design by Approximation

(B. Katz et al.,2019)

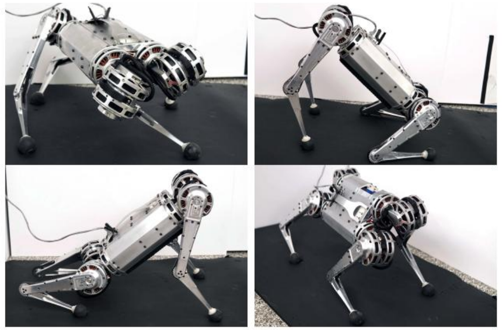


Fig. 2: The robot can easily reach a wide range of orientations without moving its feet, thanks to the large range of motion at every joint

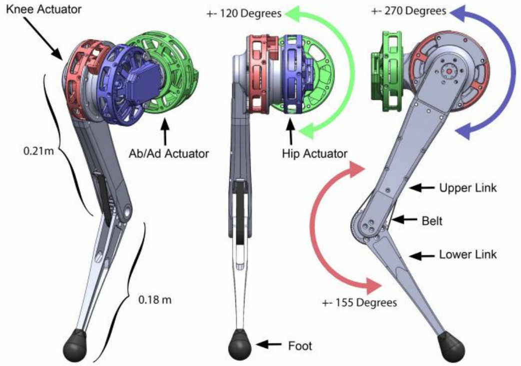
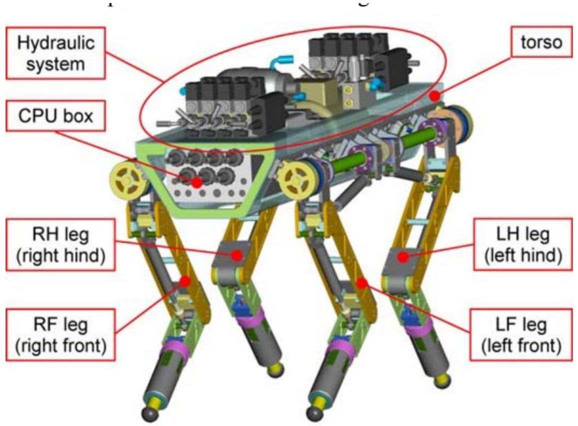


Fig. 3: CAD Diagram of a Mini Cheetah Leg. Ab/ad actuator is highlighted green, hip actuator purple, and knee actuator red.

- HyQ

(C. Semini et al., 2011)



"This elasticity, however, is believed to have a negative impact on the controller bandwidth and has to be further analysed.
The performance of two-stage servovalves with zero overlap and faster response are currently being investigated along with the effect of hose length on the actuator bandwidth." (C. Semini et al., 2011)

Limitations of Prior Work II

In computational paradigm of design optimization

Dependence of model-based approach in computational optimization

- Over-simplified model constraints
- Results specifying in predetermined tasks or trajectories

B. Motion generation

We utilise the single rigid body dynamics (SRBD) trajectory generation framework *TOWR* to generate motion plans for the robots described in this paper. The framework allows us to abstract the task for the user by simply using computer-aided design (CAD) model of the robot, a desired (complex) terrain and preferred gait parameters. A SRBD model is a dynamic model used in trajectory optimization, which is based on centroidal dynamics. Here, the individual rigid bodies of the robot are lumped together into a SRBD model with constant inertia anchored at the center of mass (COM), which is controlled by the contact forces at the end-effectors (EE) [19].

C. Task Description

For trotting, the high-level motion task is to take two steps forward, each of $0.05\mathrm{m}$, with a fixed step height of $0.05\mathrm{m}$. We allocated 22 and 37 knots[1] for the swing and double support phases of the motion, respectively, and used a symplectic Euler integrator with time-step of $10\mathrm{ms}$.

For jumping, the high-level motion task is to jump forward $0.1\mathrm{m}$ with a step height of $0.15\mathrm{m}$. We used the same integrator and time-step as in the trotting case. We defined 20 knots for the flight phase and 40 knots for the take-off and landing phases.

Model-Free RL in Design Parameters Controlling

Actor-Mimic multi-task DRL model (E. Parisotto et al., 2015)

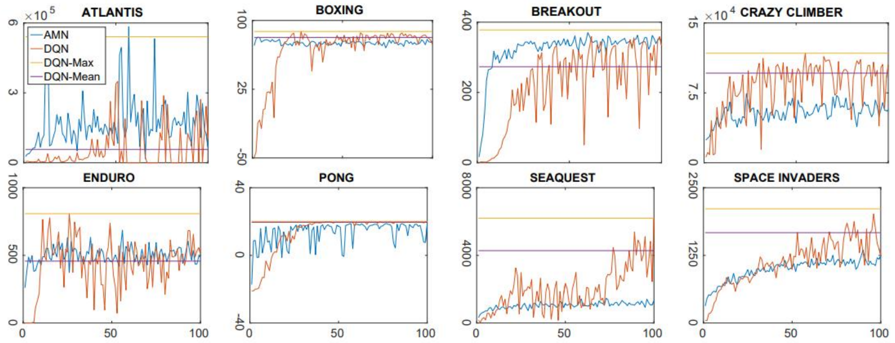
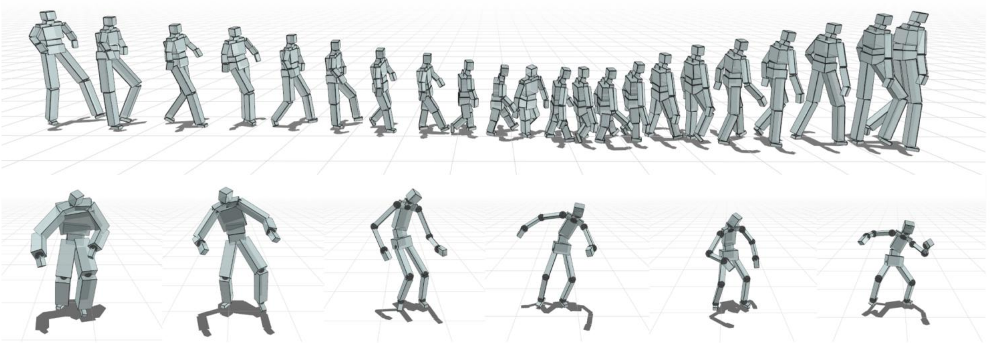
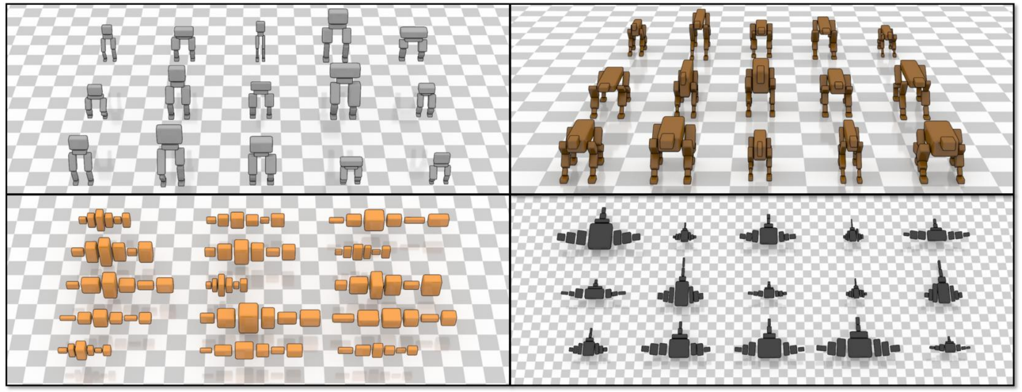


Figure 1: The Actor-Mimic and expert DQN training curves for 100 training epochs for each of the 8 games. A training epoch is 250,000 frames and for each training epoch we evaluate the networks with a testing epoch that lasts 125,000 frames. We report AMN and expert DQN test reward for each testing epoch and the mean and max of DQN performance. The max is calculated over all testing epochs that the DQN experienced until convergence while the mean is calculated over the last ten epochs before the DQN training was stopped. In the testing epoch we use $\epsilon = 0.05$ in the ϵ-greedy policy. The y-axis is the average unscaled episode reward during a testing epoch. The AMN results are averaged over 2 separately trained networks.

Single Controller managing a range of parameters (J. Won et al., 2019)





Multi-task training policy based on Meta-RL (C. Finn et al., 2017)

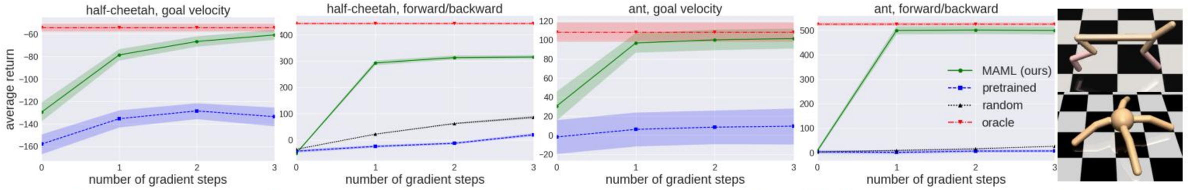


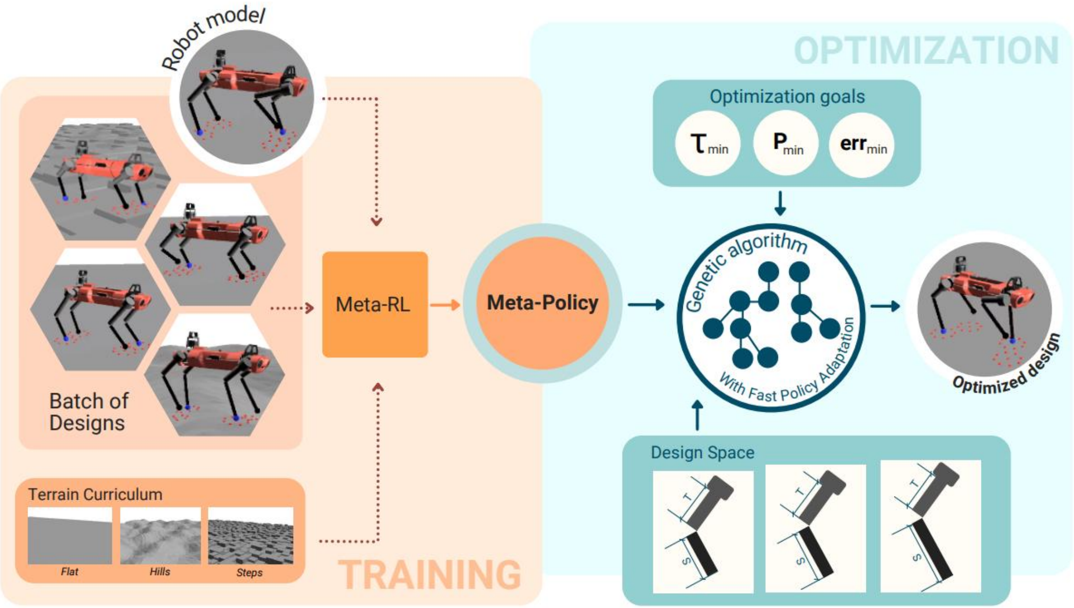
Figure 5. Reinforcement learning results for the half-cheetah and ant locomotion tasks, with the tasks shown on the far right. Each gradient step requires additional samples from the environment, unlike the supervised learning tasks. The results show that MAML can adapt to new goal velocities and directions substantially faster than conventional pretraining or random initialization, achieving good performs in just two or three gradient steps. We exclude the goal velocity, random baseline curves, since the returns are much worse (< -200 for cheetah and < -25 for ant).

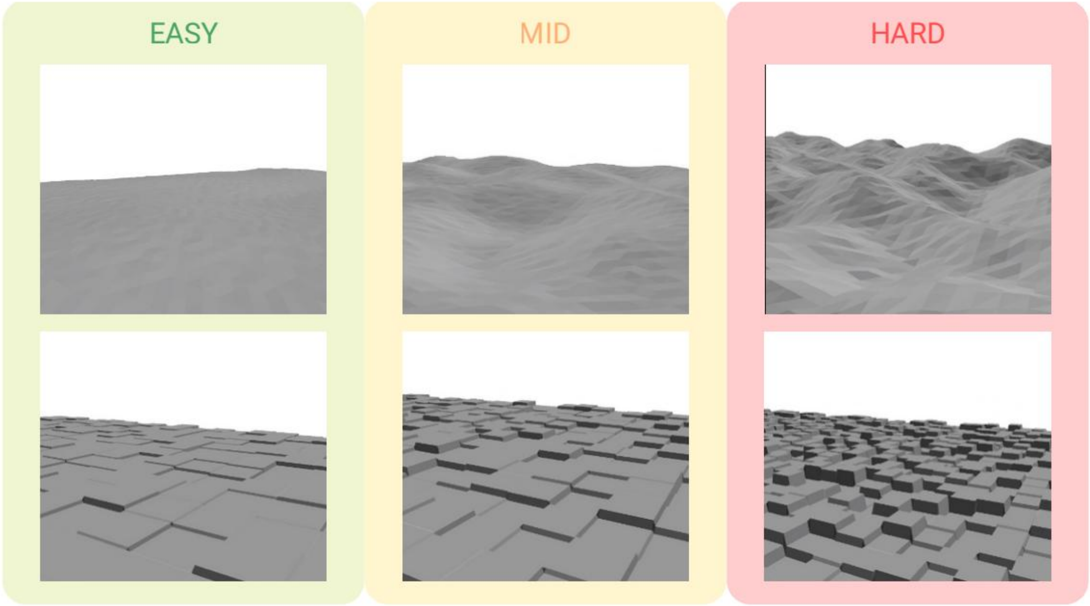
AncoraSIR.com

SUSTech
Southern University of Science and Technology

Proposal of Meta-RL to Quadrupedal Locomotion

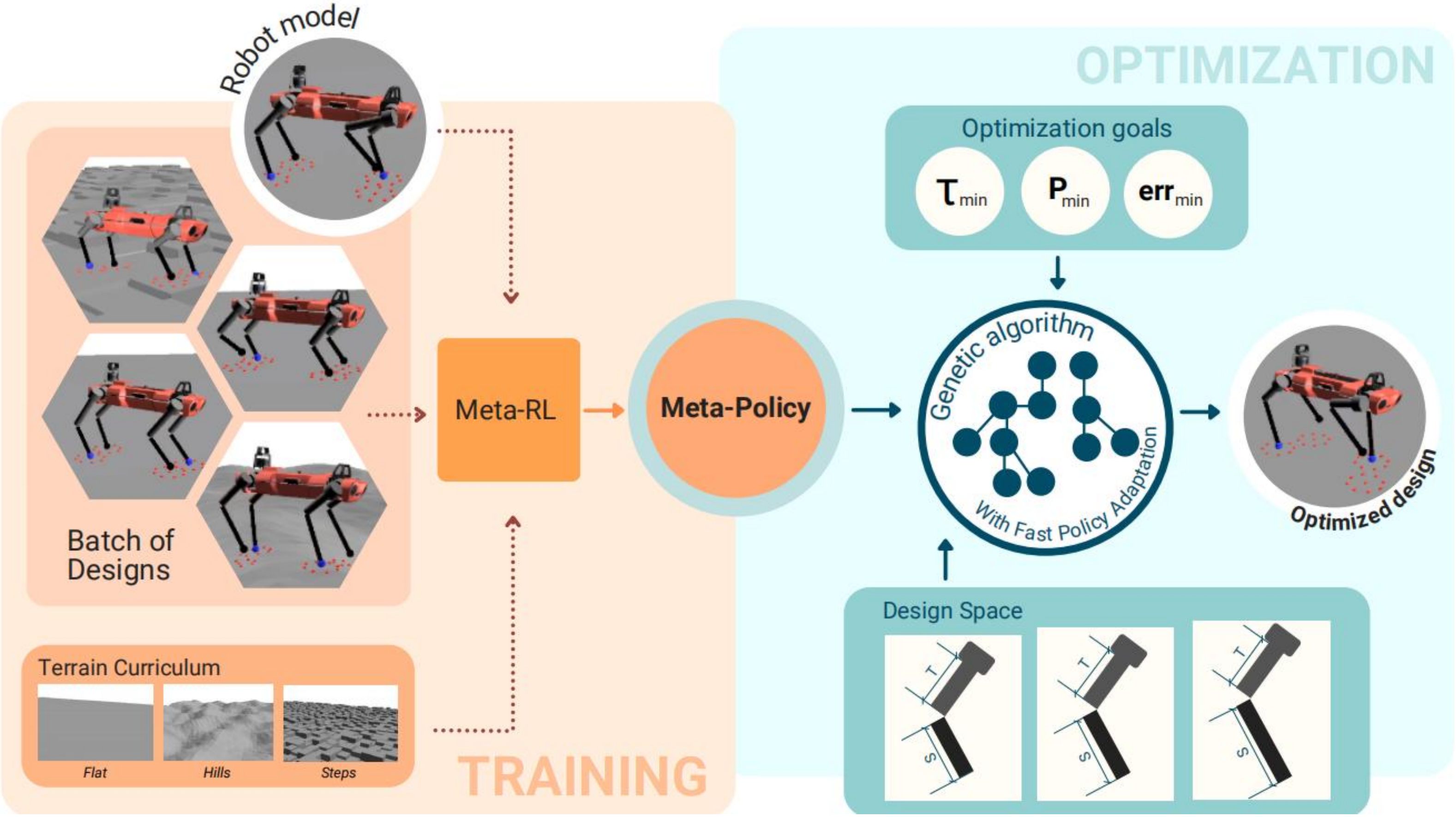
- A robust, adaptive neural network controller framework
- A Meta-RL locomotion control policy
- Experimental results on a quadrupedal robot





AncoraSIR.com

Proposed Approach



Policy training:

Meta Reinforcement Learning(Meta-RL)

Design optimization:

genetic algorithm

AncoraSIR.com

Algorithm

Markov Decision Process(MDP)

Defined by:

A tuple of state space S

Action space A

The transition probability density $\mathcal{P}(s_{t+1}|s_t, a_t)$

A reward function $\mathcal{R}(s_t, a_t, s_{t+1})$ $: \mathcal{S} \times \mathcal{A} \times \mathcal{S} \rightarrow \mathbb{R}$

The objective of RL:

Obtain an optimal policy π_* that maximizes the cumulative discounted rewards

$$\mathbb{E}[\sum_{t=j}^{\infty} \gamma^t r_t]$$

AncoraSIR.com

SUSTech
Southern University of Science and Technology

Algorithm

Fast Adaptation with Meta-learning

Model-Agnostic Meta-Learning (MAML)

a distribution of tasks p(T)

training process:

Algorithm 1: Policy meta-training with MAML

Input: Parametrized policy π_θ, Distribution over tasks $p(\mathcal{T})$, Number of policy updates N, Meta-batch size M, length of collected rollouts K. Step-size hyperparameters α, β

```
Initialize θ;
for N policy updates do
    Sample batch of M design parameter tuples
      T_i ~ p(T);
    foreach T_i do
        Sample policy rollouts of length K
          D = {(s_1, a_1, r_1, s_2, ..., s_K)};
        Compute adapted parameters for current task:
        θ'_i = θ − α∇_θ L_{T_i}(π_θ);
        Sample new trajectories D'_i using adapted
          policy π_{θ'_i} in T_i;
    end
    Update θ ← θ − β∇_θ Σ_{T_⟩} L_{T_i}(π_{θ'_i}), using the
      collected D'_i;
end
```

Step 1: Initialize θ;

Step 2: **for** *N policy updates* **do**

Step 3: Sample batch of M design parameter tuples $\mathcal{T}_i \sim p(\mathcal{T})$;

Step 4: **foreach** $\mathcal{T}_i$ **do**

Step 5: Sample policy rollouts of length K $\mathcal{D} = \{(s_1, a_1, r_1, s_2, \ldots, s_K)\}$;

Step 6: Compute adapted parameters for current task:

Step 7: $\theta_i' = \theta - \alpha \nabla_\theta \mathcal{L}_{\mathcal{T}_i}(\pi_\theta)$;

Step 8: Sample new trajectories $\mathcal{D}_i'$ using adapted policy $\pi_{\theta_i'}$ in $\mathcal{T}_i$;

Step 9: **end**

Step 10: Update $\theta \leftarrow \theta - \beta \nabla_\theta \sum_{\mathcal{T}_\rangle} \mathcal{L}_{\mathcal{T}_i}(\pi_{\theta_i'})$, using the collected $\mathcal{D}_i'$;

Step 11: **end**

Algorithm

Design Optimization

Obtain a set of design parameters
that maximizes a given fitness function $f(\mathcal{T}) \in \mathbb{R}$
Gradient-free algorithm
Use CMA-ES for the optimization

The fitness function (the Monte-Carlo estimation):

$$\mathcal{C}(s_t) : \mathcal{S} \rightarrow \mathbb{R}, \text{ i.e. } \quad f(\mathcal{T}) = \mathbb{E}_{s_t \sim \xi(\pi_{\theta_{\mathcal{T}}})}[-\mathcal{C}(s_t)]$$

Algorithm 2: Design optimization with meta-policy

Input: Trained meta-policy π_{θ_0}, Number of generations G, Initial design population $\mathcal{P}_0$, step-size hyperparameter α, number of gradient updates U, lenght of collected rollouts T.

for *k in [1...G]* **do**
  foreach $p_i \in \mathcal{P}_k$ **do**
    Set current design to p_i;
    Set policy parameters to initial value: $\theta \leftarrow \theta_0$;
    for *U gradient updates* **do**
      Sample policy rollouts of length T $\mathcal{D} = \{(s_1, a_1, r_1, s_2, \ldots, s_T)\}$;
      Perform adaptation step:
      $\theta \leftarrow \theta - \alpha \nabla_\theta \mathcal{L}_{p_i}(\pi_\theta)$;
    end
    Compute fitness score for p_i and store it;
  end
  Update $\mathcal{P}$ using the computed scores.
end

Experimental Setup

Domain

datasets:

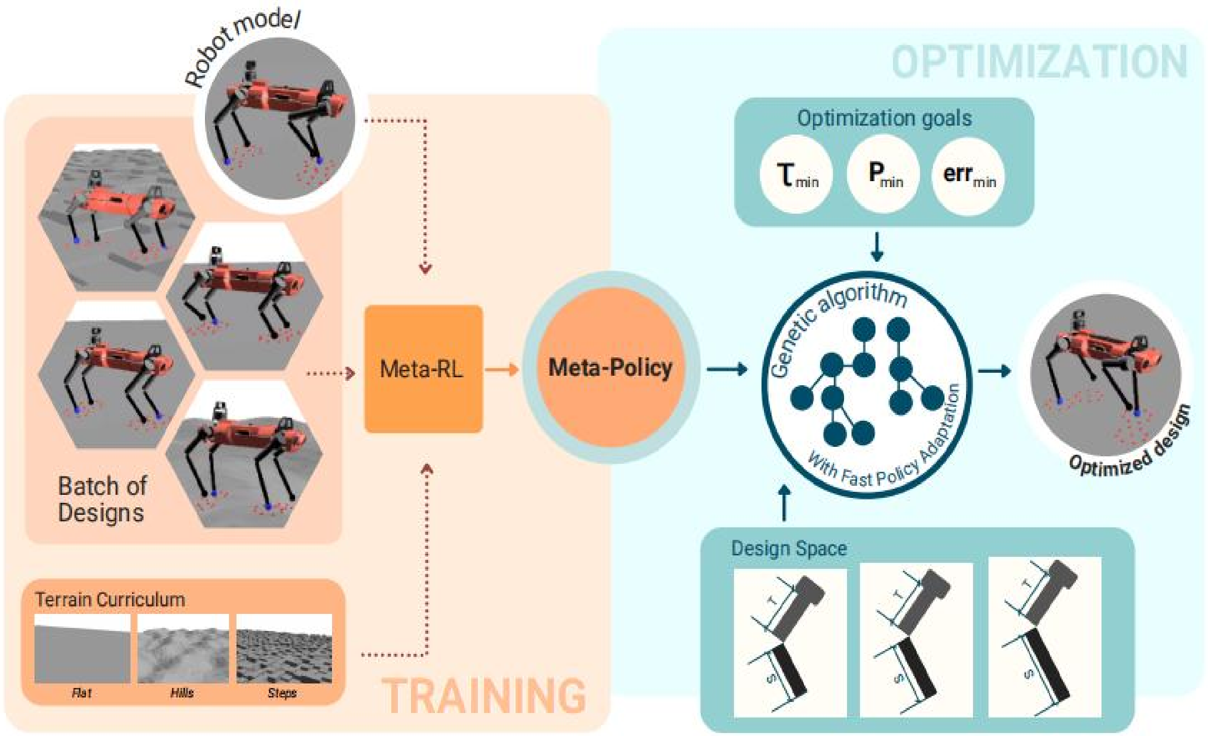
some established leg parameters

tasks:

speed tracking error

joint torque

joint positive mechanical power



robot hardware setups:

simulated environment Raisim, includes simplified models for the speed and torque limitations of real actuators.

AncoraSIR.com

SUSTech
Southern University of Science and Technology

Experimental Setup

Baselines:

Vitruvio

Scientific hypotheses tested:

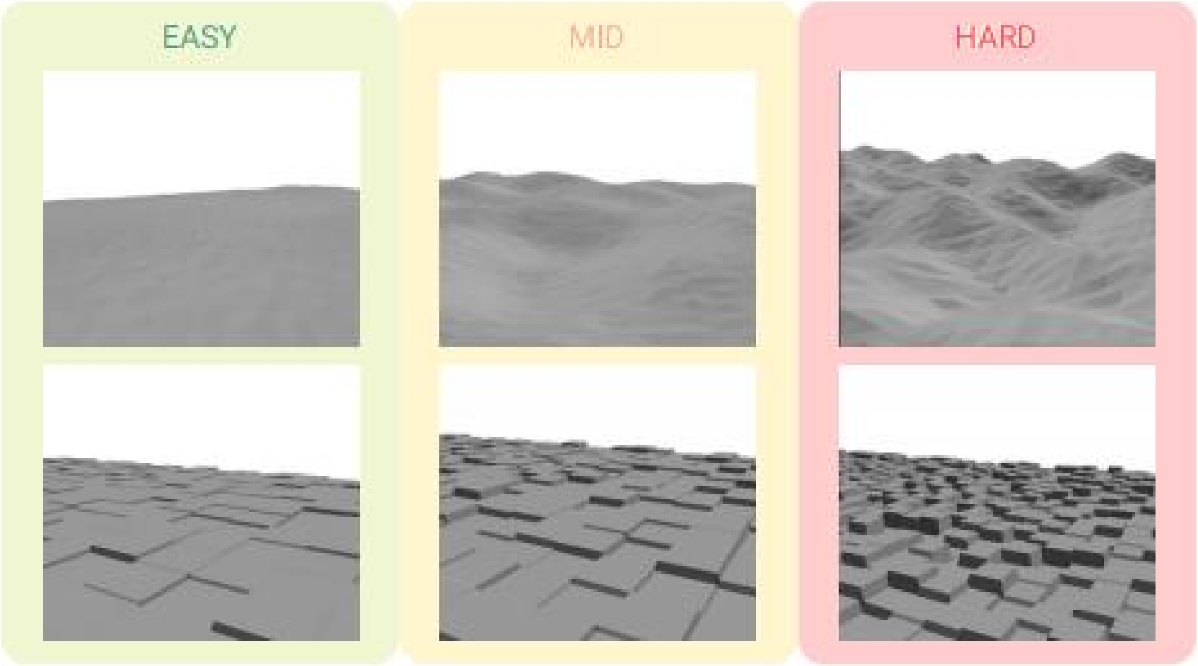
Using a design optimization framework based on meta reinforcement learning (Meta RL) can quickly adapt to different design instances and achieve near optimal performance.

AncoraSIR.com

Experimental Setup

Evaluate metrics:

Analyzing the average rewards obtained under different parameters compared to naive strategies and specifically given strategies.



AncoraSIR.com

Experimental Results

1. The meta-policy consistently outperforms the naive multi-task policy in all cases.

2. After the adaptation steps, the meta-policy reaches rewards comparable to the specialized policies, achieving close-to optimal capabilities.

TABLE I
OPTIMIZED LINK SCALES WITH RESPECT TO THE NOMINAL DESIGN

Objective	Flat		Easy Hills		Mid Hills		Hard Hills		Easy Steps		Mid Steps		Hard Steps	
	Thigh	Shank	Thigh	Shank	Thigh	Shank	Thigh	Shank	Thigh	Shank	Thigh	Shank	Thigh	Shank
$\mathcal{C}_v$	1.02	0.99	1.01	1.01	1.06	1.03	1.23	1.18	1.05	1.0	1.07	1.06	1.21	1.17
$\mathcal{C}_\tau$	0.61	0.63	0.63	0.67	0.64	0.68	0.75	0.80	0.70	0.68	0.76	0.77	0.94	0.97
$\mathcal{C}_p$	1.05	0.94	1.07	0.95	1.06	0.93	1.10	0.97	1.04	0.93	1.07	0.96	1.17	1.13

AncoraSIR.com

Experimental Results

TABLE II
MEAN IMPROVEMENT IN OPTIMIZATION OBJECTIVES COMPARED TO THE NOMINAL DESIGN.

	$\mathcal{C}_v$	$\mathcal{C}_\tau$	$\mathcal{C}_p$
Flat	1.27%	43.53%	4.30%
Easy Hills	2.16%	43.85%	5.07%
Mid Hills	4.32%	39.72%	3.01%
Hard Hills	27.85%	16.36%	13.47%
Easy Steps	4.50%	37.47%	4.10%
Mid Steps	6.45%	28.98%	5.47%
Hard Steps	24.79%	4.13%	16.01%

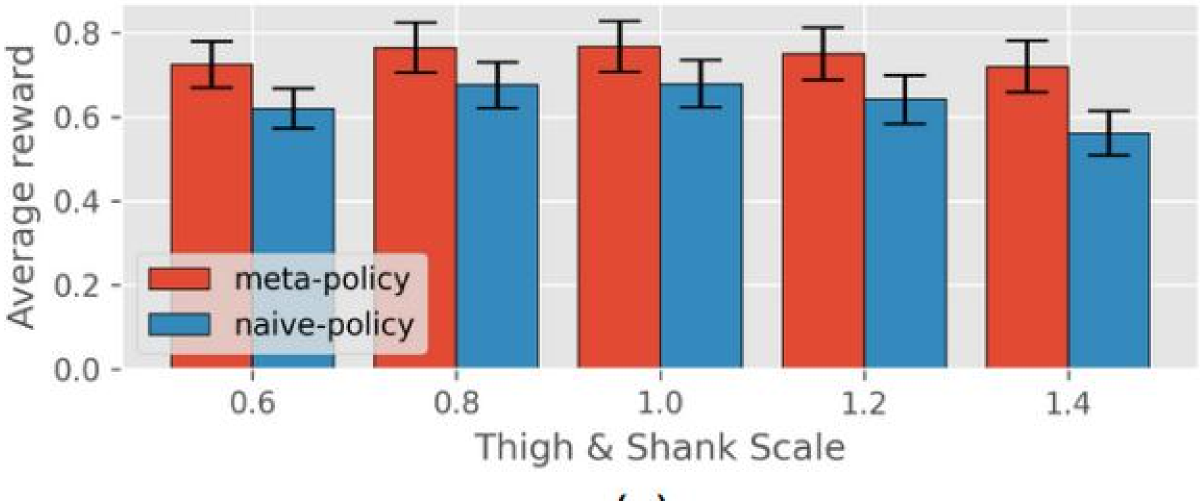
AncoraSIR.com

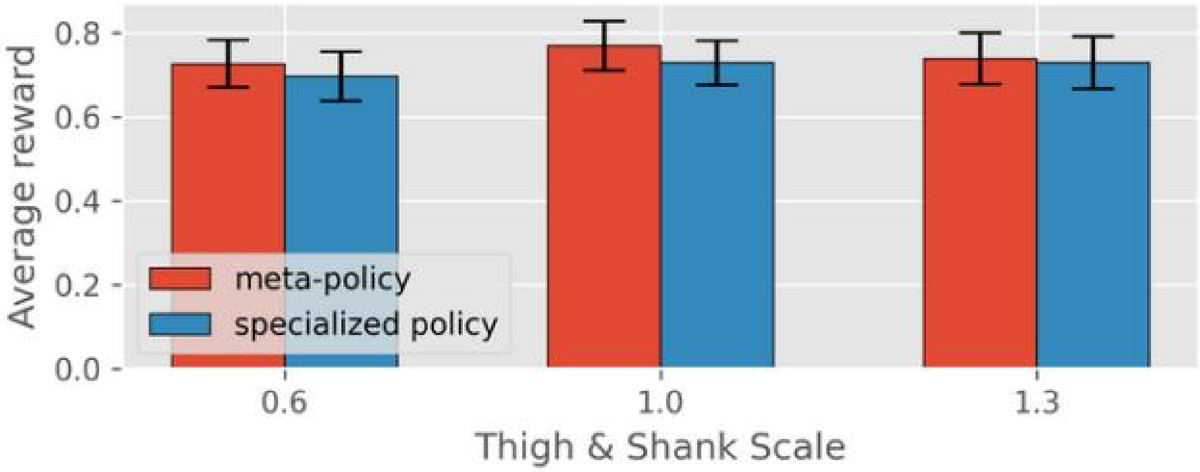
SUSTech Southern University of Science and Technology

Experimental Results

The most interesting result is:

They further verify the performance of our meta-policy by comparing it against a set of policies trained for specific designs (specialized policy). After the adaptation steps, the meta-policy reaches rewards comparable to the specialized policies, achieving close-to optimal capabilities





Discussion of Results

They would like to highlight the flexibility of their approach in considering the robot’s operating environment during the design process, which can be limited in the conventional optimization-based approach, where we need analytic dynamics models.

他们想强调他们方法的灵活性，将机器人的操作环境在设计过程中就加入考虑。这在传统的，需要分析动力学模型的基于优化的方法中是受限的。

AncoraSIR.com

Limitations

1.The cost functions could not capture the actual dynamics of the system.

2.The ratio of different sources isunclear

AncoraSIR.com

SUSTech

Southern University of Science and Technology

Future Work for Paper

Subsequent optimization work

- Adding cost functions for the design optimization.
- Adding more design parameters including discrete and continuous.
- Build prototypes of optimized designs and validate them on the physical system.

AncoraSIR.com

Extended Readings

References & related readings for this paper

[1] M. Chadwick et al., “Vitruvio: An open-source leg design optimization toolbox for walking robots,” IEEE Robotics and Automation Letters, vol. 5, no. 4, pp. 6318–6325, 2020.

[2] F. De Vincenti et al., “Control-aware design optimization for bioinspired quadruped robots,” in 2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), 2021, pp. 1354–1361.

[3] A. W. Winkler et al., “Gait and trajectory optimization for legged systems through phase-based end-effector parameterization,” IEEE Robotics and Automation Letters (RA-L), vol. 3, pp. 1560–1567, July 2018.

[4] T. Miki et al., “Learning robust perceptive locomotion for quadrupedal robots in the wild,” Science Robotics, vol. 7, no. 62, 2022.

Extended Readings

Annual Conference on Robot Learning

http://corl2023.org/

http://corl2022.org/

AncoraSIR.com

Summary

- The reading is discussing optimal design of legged robots by using meta reinforcement learning.
- It is hard because there are many parameters affect final performance.
- The key limitation of prior work is the design needs tedious manual tuning.
- The key insight of the proposed work is Model-Free Reinforcement Learning for robot controlling.
- This paper demonstrate that RL is an ideal solution to solve the inner optimization problem of the design optimization.

Thank you for your listening!

沈奕宁 11911613　　许哲睿 12011230

雍志涛 12110824　　范王卓12111942

乔凯　 12211112　　张草萌 12112223

AncoraSIR.com