Learning Agile and Dynamic Motor Skills for Legged Robots

Presenter: 陈逸飞 方艺钧
杨德宇 宁晨辉 朱思颖

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

Problem

present situation

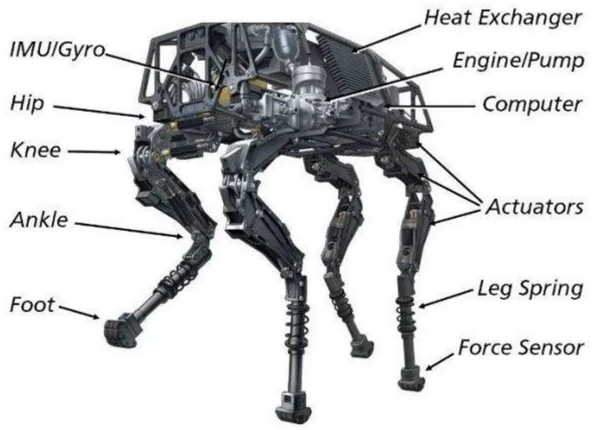
- So far, reinforcement learning research for legged robots is mainly limited to simulation, and only few and comparably simple examples have been deployed on real systems.

what we do?

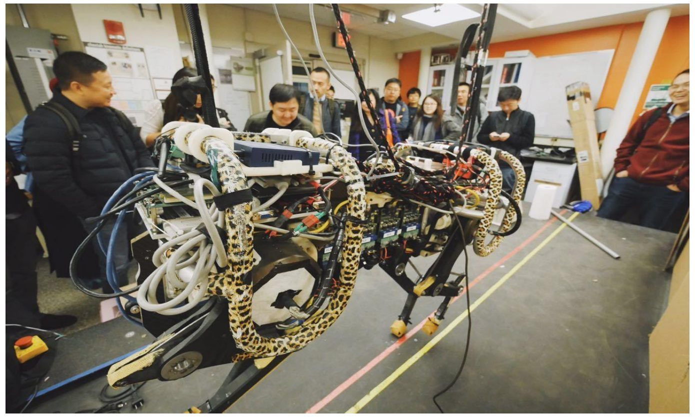
- In the present work, we report a new method for train_x0002_ing a neural network policy in simulation and transfer_x0002_ring it to a state-of-the-art legged system, thereby weleverage fast, automated, and cost-effective data gener_x0002_ation schemes.

Problem

Legged systems have the potential to perform any physical activity humans and animals are capable of.



Big dog by Boston Dynamics

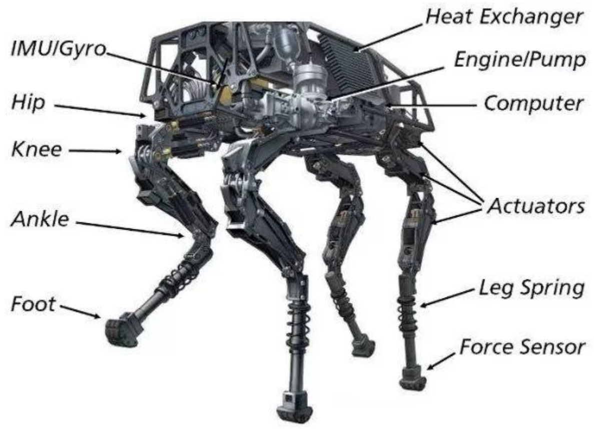


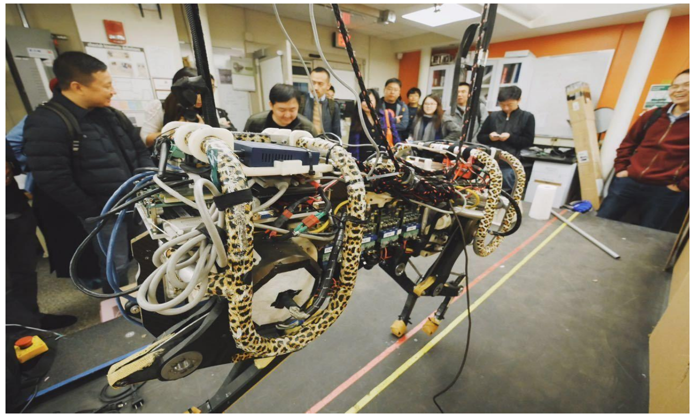
Cheetah by MIT

SpotMini by Boston Dynamics

AncoraSIR.com

Problem





Big dog by Boston Dynamics
 large scale， generate smoke and noise
limiting indoor environment

Cheetah by MIT
has not been thoroughly evaluated with respect to battery life, turn_x0002_ing capability, mechanical robustness, and outdoor applicability

SpotMini by Boston Dynamics
the complicated actuator design
 increases cost and compromises the power output of the robot.

AncoraSIR.com

Related Work

Designing control algorithms for these hardware platforms
remains exceptionally challenging.
1. high-dimensional and non-smooth systems cause uncertainties in the dynamics
2 lengthy design process and arduous parameter tuning.

Popular approach
controlling physical legged systems is modular controller design

Example
Robust Quadrupedal Locomotion on Sloped Terrains:
A Linear Policy Approach

Related Work

Trajectory optimization approaches have been proposed to mitigate the aforementioned problems.

planning :uses rigid-body dynamics and numerical optimization to compute an optimal path thatthe robot should follow to reach the desired goal.

tracking:follow the path

AncoraSIR.com

Related Work

Data-driven methods
By learning effective controllers directly from experience. Direct application of learning methods to physical legged systems is therefore complicated and has only been demonstrated on relatively simple and stable platforms or in a limited context

AncoraSIR.com

State-of-art

Reason:
The discrepancy between simulation and the real system in terms of dynamics and perception

Solution：
There are two general approaches to bridging the reality gap
improve simulation fidelity or accept the imperfections of simulation

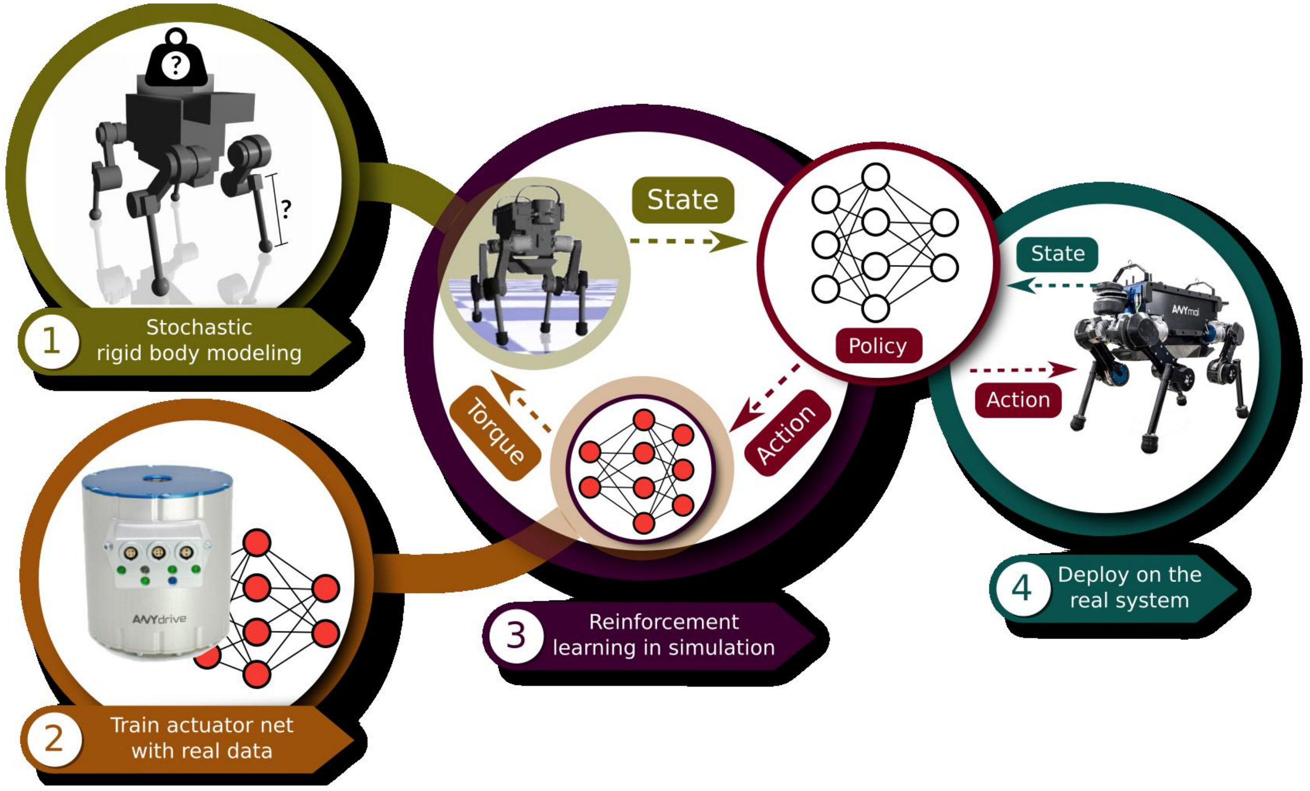
AncoraSIR.com

State-of-art

ANY-mal robot

ANYmal has a much larger leg length relative to footprint, therefore more difficult tocontrol

Solution:


?
?
1
Stochastic rigid body modeling
2
AΝΥdrive
Train actuator net with real data
State
Policy
Torque
Action
3
Reinforcement learning in simulation
State
ANYmal
Action
4
Deploy on the real system

AncoraSIR.com

Method

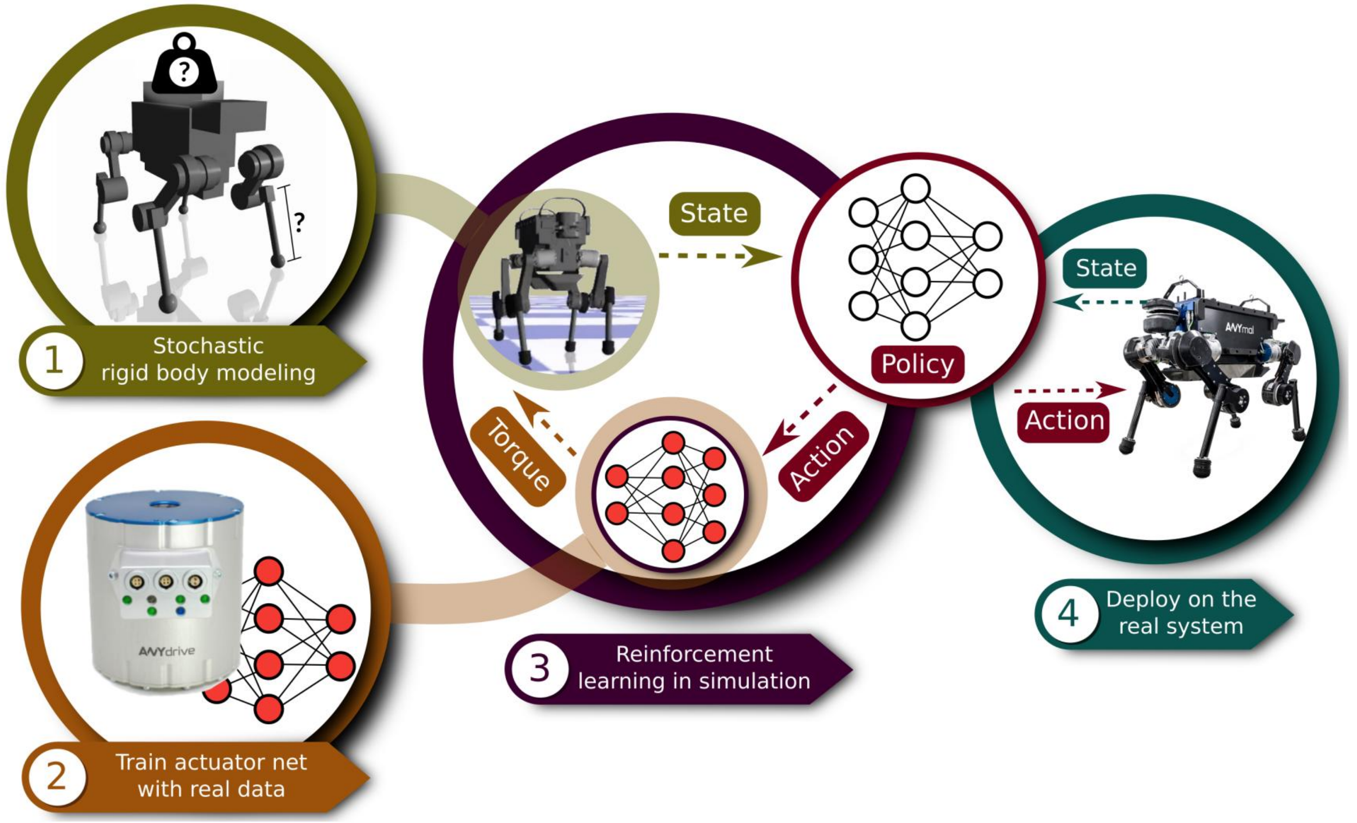
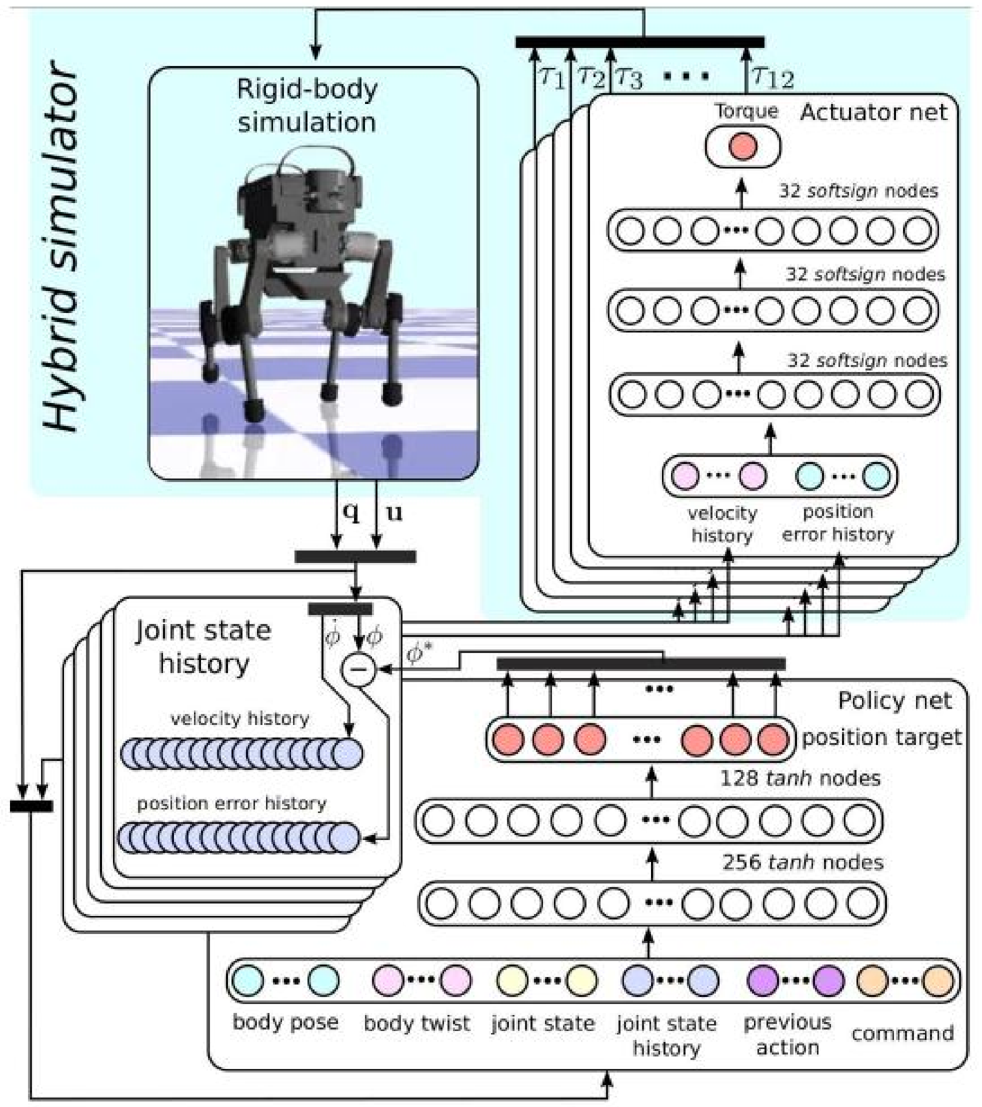

?
?
Stochastic rigid body modeling
1
State
Policy
Torque
Action
3
Reinforcement learning in simulation
2
Train actuator net with real data
ANYdrive
State
Action
4
Deploy on the real system

Fig. 1. Creating a control policy. In the first step, we identify the physical parameters of the robot and estimate uncertainties in the identification. In the second step, we train an actuator net that models complex actuator/software dynamics. In the third step, we train a control policy using the models produced in the first two steps. In the fourth step, we deploy the trained policy directly on the physical system.

1. Modeling rigid-body dynamics
2. Modeling the actuation
3. Reinforcement learning
4. Application

AncoraSIR.com

Method



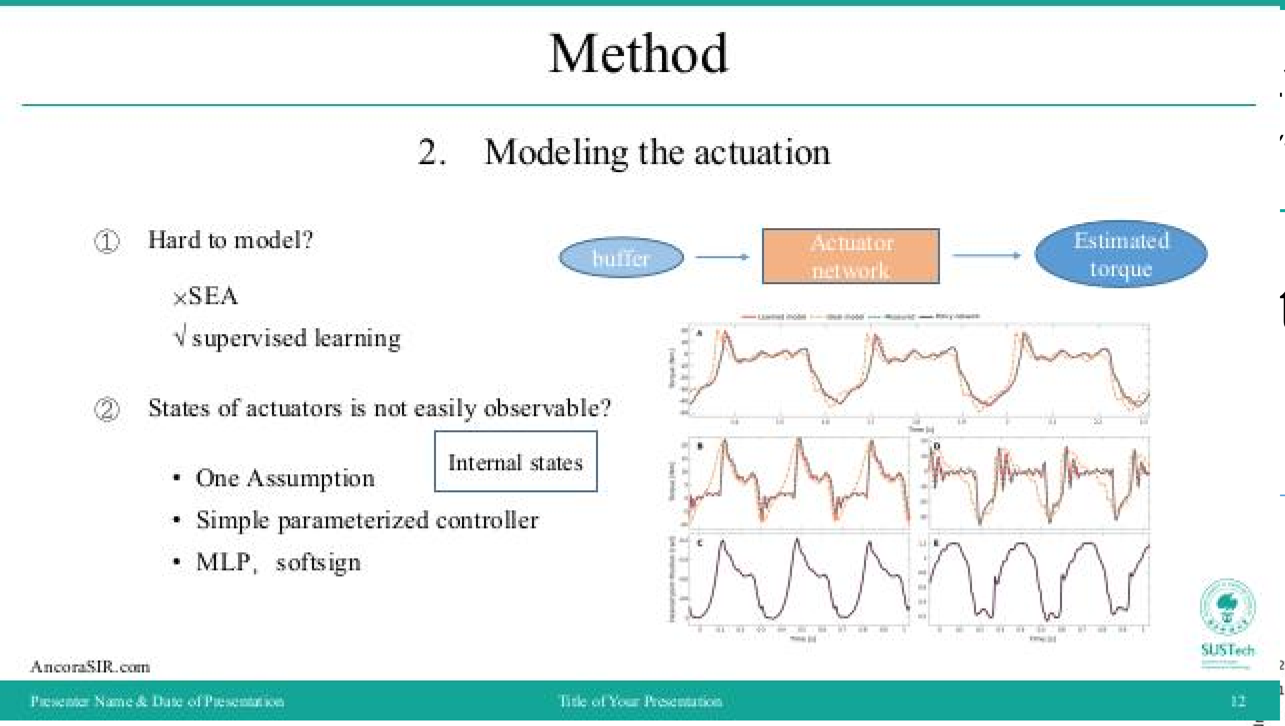
1. Modeling rigid-body dynamics
   - Build a sumulation platform

① Intermittent contacts?

using rigid body contact solver

② Inertial properties of links?

robustifying the policy by training with 30 different ANYmal models with stochastically sampled inertial properties



Method

2. Modeling the actuation

① Hard to model?

×SEA

√ supervised learning

② States of actuators is not easily observable?

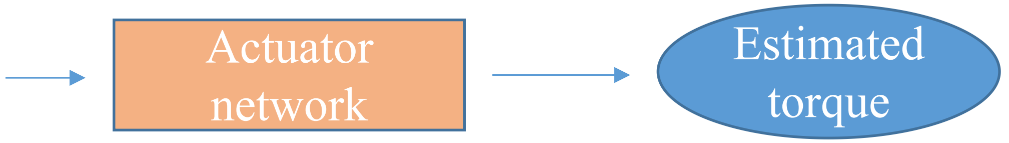
Internal states

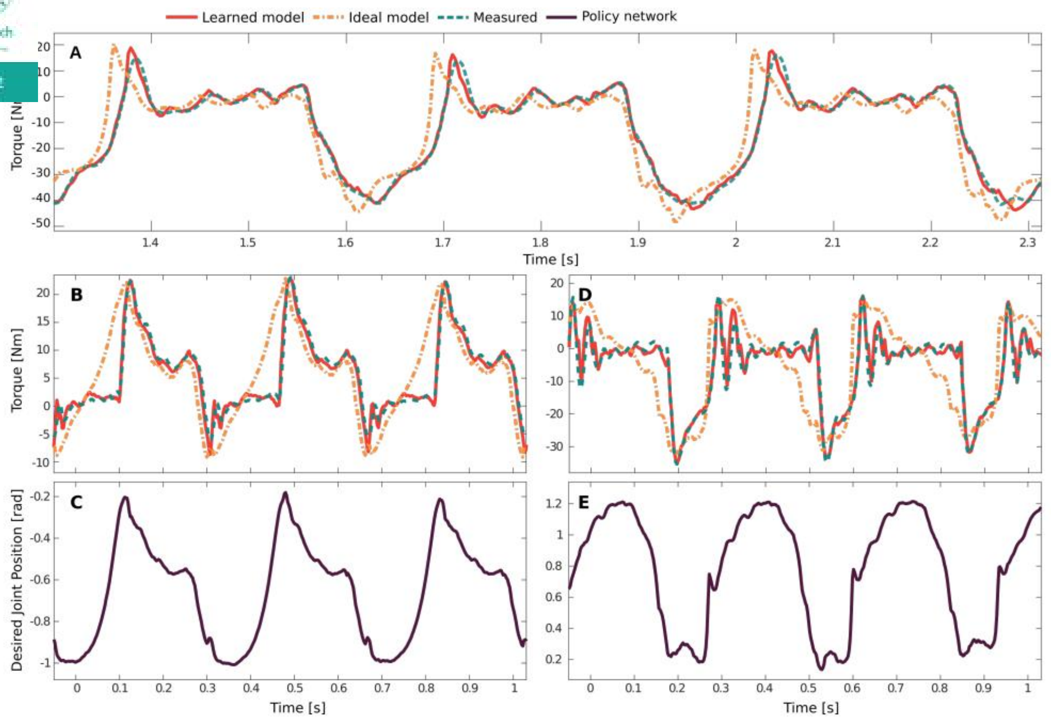
- One Assumption
- Simple parameterized controller
- MLP, softsign

AncoraSIR.com

hod

the actuation





② States of actuators is not easily observable?

Internal states

- One Assumption
- Simple parameterized controller
- MLP

AncoraSIR.com

Method

Reinforcement Learning

Trust Region Policy Optimization(TRPO)

- The agent selects actions according to a **stochastic policy**

$$\pi^* = \underset{\pi}{\operatorname{argmax}} \mathbb{E}_{\boldsymbol{\tau}(\pi)} \left[\sum_{t=0}^{\infty} \gamma^t r_t \right],$$

where γ ∈ (0, 1) is the discount factor, τ(π) is the trajectory distribution under policy π,
rt is the scalar reward,

- Find a policy that maximizes the discounted sum of rewards over an infinite horizon

AncoraSIR.com

Method

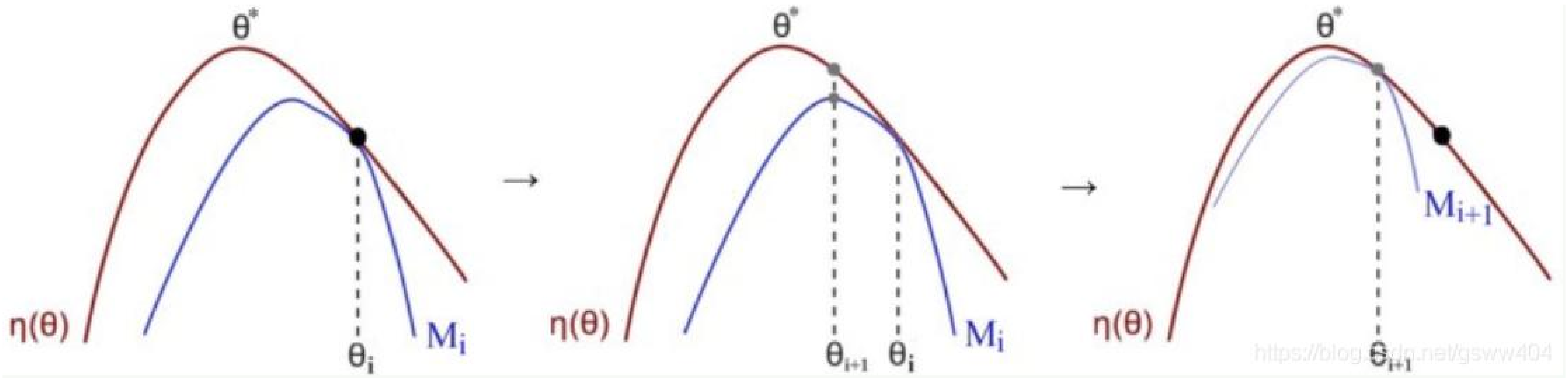
Trust Region Policy Optimization(TRPO)

AncoraSIR.com

Method

Surrogate function

$$\theta_{\text{new}} = \theta_{\text{old}} + \alpha \nabla_\theta \mathrm{J} \quad \eta(\pi_\theta) = \mathbb{E}_{\tau\sim\pi_\theta}\left[\sum_{t=0}^{\infty} \gamma^{\mathrm{t}} \mathbf{r}_{\mathrm{t}}\right]$$



AncoraSIR.com

Proposed Approach / Algorithm / Method

Further explaination of the title with supporting evidence

1-5 slides

Describe algorithm or framework (pseudocode and flowcharts can help)

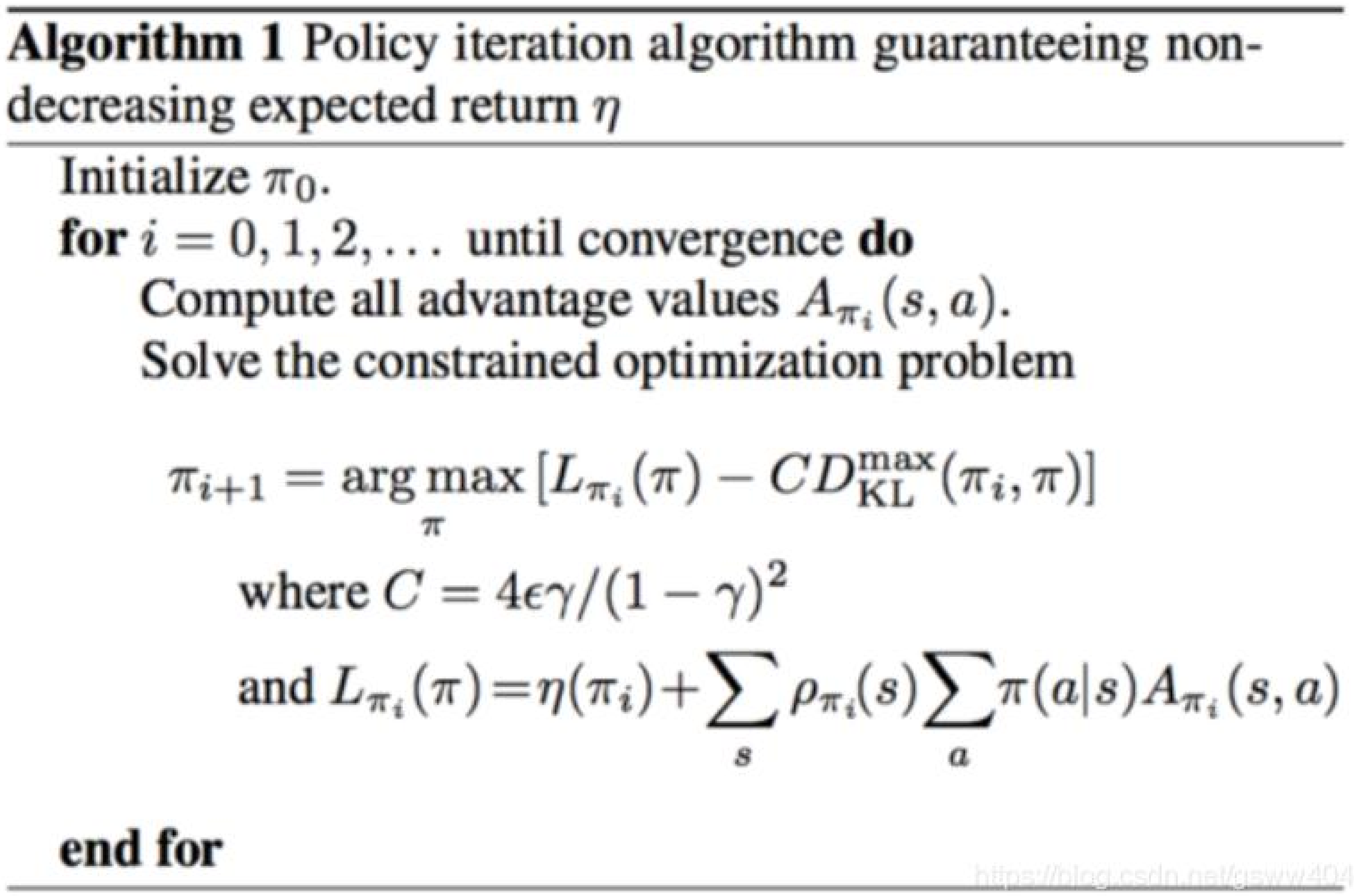
- What is the optimization objective?
- What are the core technical innovations of the algorithm/framework?

Implementation details should be left out here, but may be discussed later if its relevant for limitations / experiments

AncoraSIR.com

Method

iteration algorithm



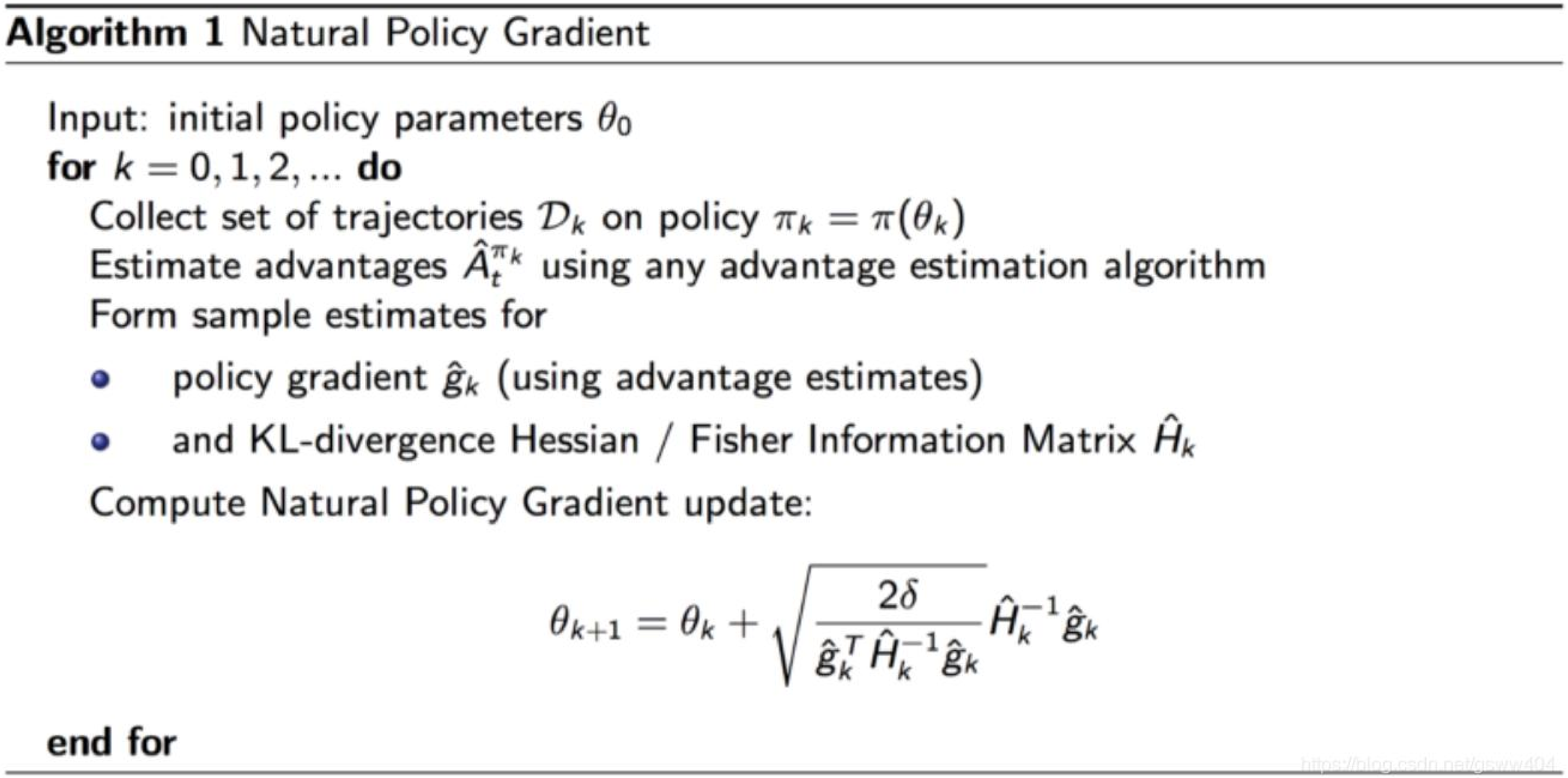
Algorithm 1 Policy iteration algorithm guaranteeing non-decreasing expected return η

```
Initialize $\pi_0$.
for $i = 0, 1, 2, \dots$ until convergence do
    Compute all advantage values $A_{\pi_i}(s, a)$.
    Solve the constrained optimization problem

        $\pi_{i+1} = \arg\max_{\pi} \left[ L_{\pi_i}(\pi) - C D_{\mathrm{KL}}^{\max}(\pi_i, \pi) \right]$
            where $C = 4\epsilon\gamma/(1-\gamma)^2$
            and $L_{\pi_i}(\pi) = \eta(\pi_i) + \sum_s \rho_{\pi_i}(s) \sum_a \pi(a|s) A_{\pi_i}(s, a)$

end for
```

SUSTech
Southern University of Science and Technology

Method

natural policy gradien



Algorithm 1 Natural Policy Gradient

```
Input: initial policy parameters $\theta_0$
for $k = 0, 1, 2, \dots$ do
    Collect set of trajectories $\mathcal{D}_k$ on policy $\pi_k = \pi(\theta_k)$
    Estimate advantages $\hat{A}_t^{\pi_k}$ using any advantage estimation algorithm
    Form sample estimates for
        • policy gradient $\hat{g}_k$ (using advantage estimates)
        • and KL-divergence Hessian / Fisher Information Matrix $\hat{H}_k$
    Compute Natural Policy Gradient update:
        $\theta_{k+1} = \theta_k + \sqrt{\frac{2\delta}{\hat{g}_k^T \hat{H}_k^{-1} \hat{g}_k}} \hat{H}_k^{-1} \hat{g}_k$
end for
```

AncoraSIR.co

Method

Application

Observation and action

- The joint state history was essential in training a locomotion policy

Policy training details

- MLP with two hidden layers, with 256 and 128 units each and tanh nonlinearity
- Bounded activation functions, such as tanh, yield less aggres_x0002_sive trajectories

Deployment on the physical system

AncoraSIR.com

Experiment1

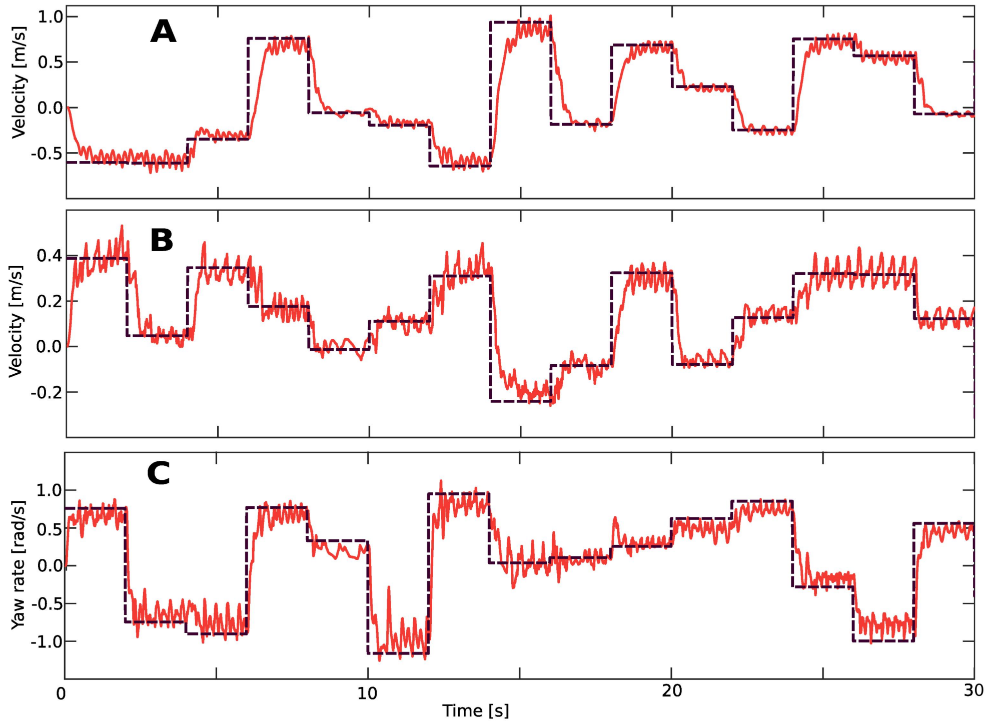
- We first qualitatively evaluate this learned locomotion policy by giving random commands using a joystick. Additionally, the robot is disturbed during the experiment by multiple external pushes to the main body.

- The result show experiment without a single failure, which manifests the robustness of the learned policy.

AncoraSIR.com

Experiment 2

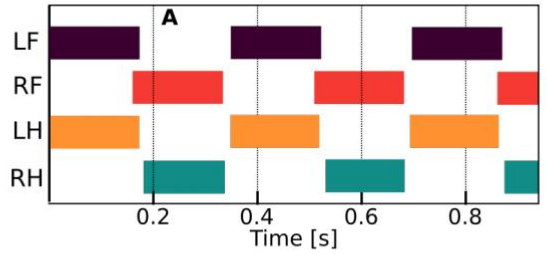
Next, we quantitatively evaluate this learned locomotion policy by driving the robot with randomly-sampled commands
The base velocity plot is shown in fig. S1. The average linear velocity error was 0.143 m/s and the average yaw rate error was 0.174 rad/s.



AncoraSIR.com

Experiment 3

- We now compare the learned controller to the best performing existing locomotion controller available for ANYmal
Movie S3 illustrates the experiments for both the learned policy and the model-based policy.



- The control performance was also evaluated and compared in forward running. To this end, we sent a step input of four different speed commands (0.25, 0.5, 0.75, and 1.0 m/s) for 4.5 s each.
The results, including a comparison to the prior method , are shown in Fig. 2.

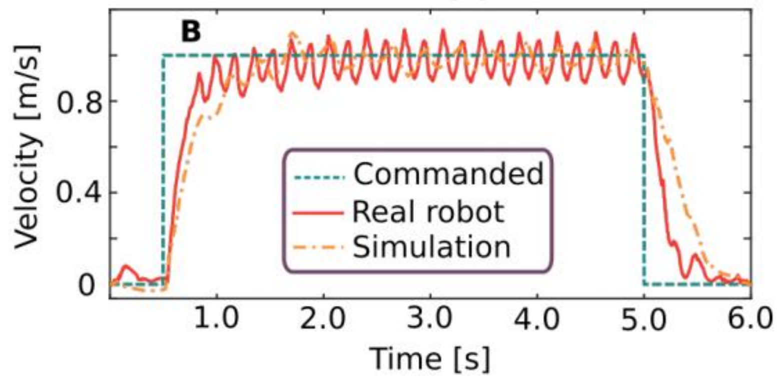
https://youtu.be/aqVPyIgZ15M

AncoraSIR.com

SUSTech Southern University of Science and Technology

Experiment 3

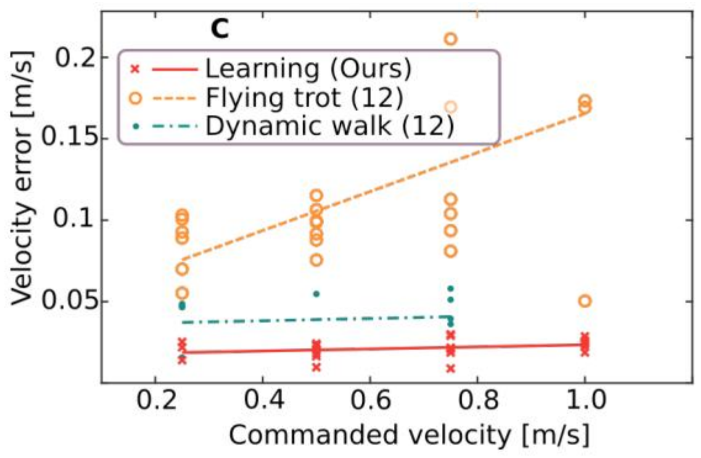
Figure 2B shows the velocity tracking accuracy of the policy both in simulation and on the real robot. Note that the oscillation of the observed velocity around the commanded one is a well-known phenomenon in legged systems, including humans [43]. In terms of average velocity, the learned policy has an error of 2.2 % on the real robot, 1.1 % higher than in a simulation.

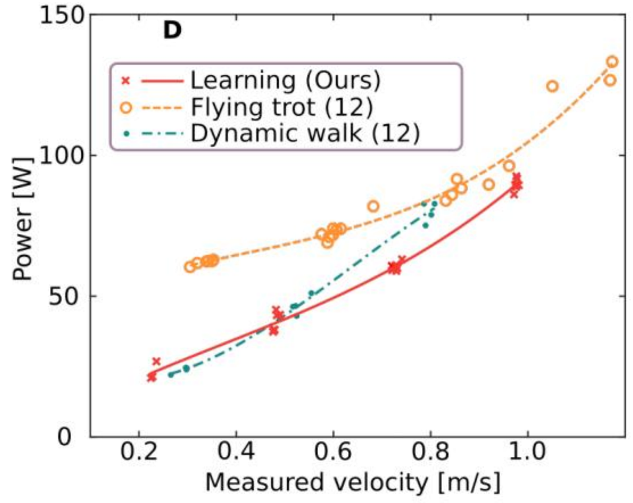


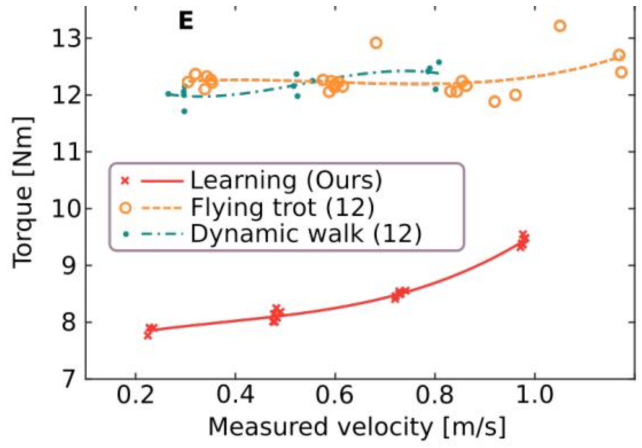
AncoraSIR.com

Experiment 3

Figure 2C, 2D, and 2E compare the performance of the learned controller to the approach of Bellicoso et al. in terms of accuracy and efficiency



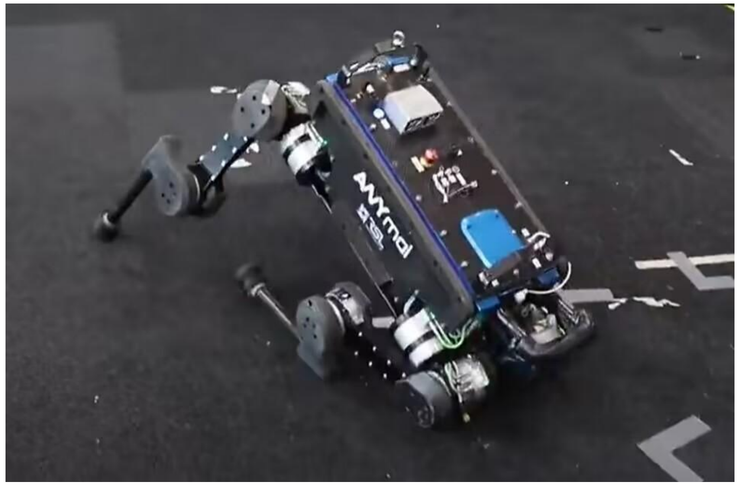




AncoraSIR.com

Experiment 4

Next, we compare our method to ablated alternatives: training with an ideal actuator model and training with an analytical actuator model. We observed violent shaking of the limbs, probably due to not accounting for various delays properly. Even though the analytical model contains multiple delay sources that are tuned using real data, accurately modeling all delay sources is complicated when the actuator has limited bandwidth



AncoraSIR.com

Experiment 5

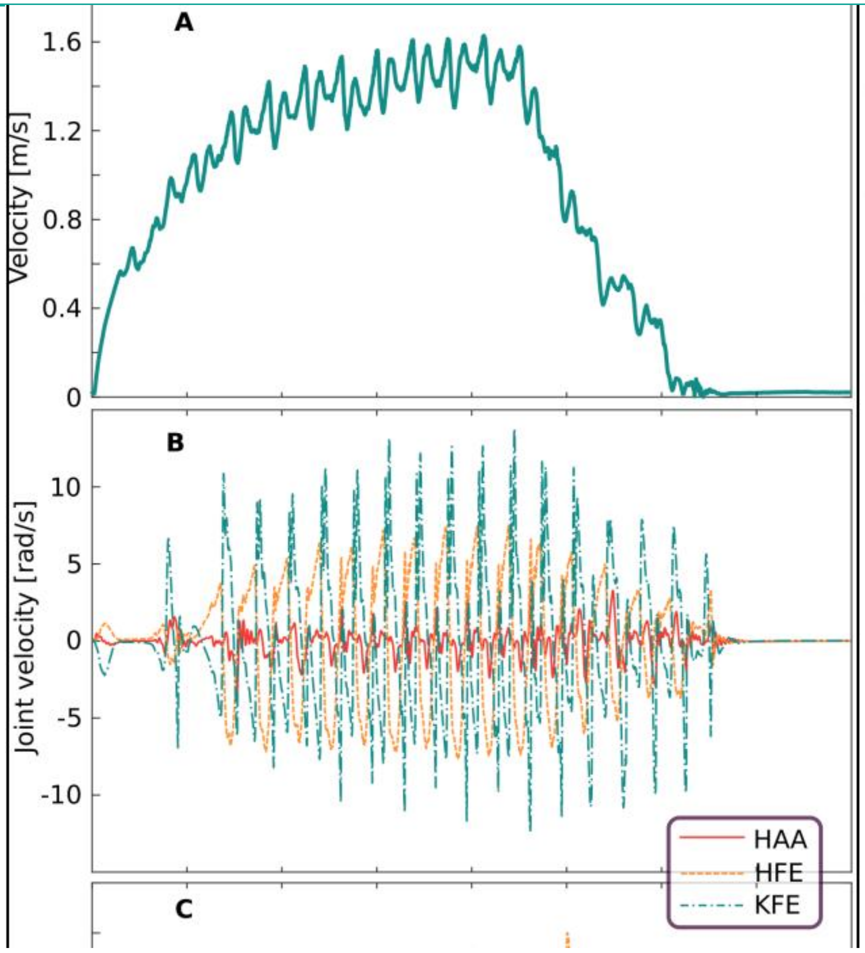
High-speed locomotion

The current speed record on ANYmal is 1.2 m/s and has been set using the flying trot gait . Although this may not seem high, it is 50 % faster thanthe previous speed record on the platform . Such velocities are challenging to reach via conventional controller design while respecting all limits of the hardware.

We have used the presented methodology to train a high speed locomotion controller. This controller was tested on the physical system by slowly increasing the commanded velocity to 1.6 m/s and lowering it to zero after 10 meters. The forward speed and joint velocities/torques are shown in Fig. 3.

AncoraSIR.com

Experiment 5



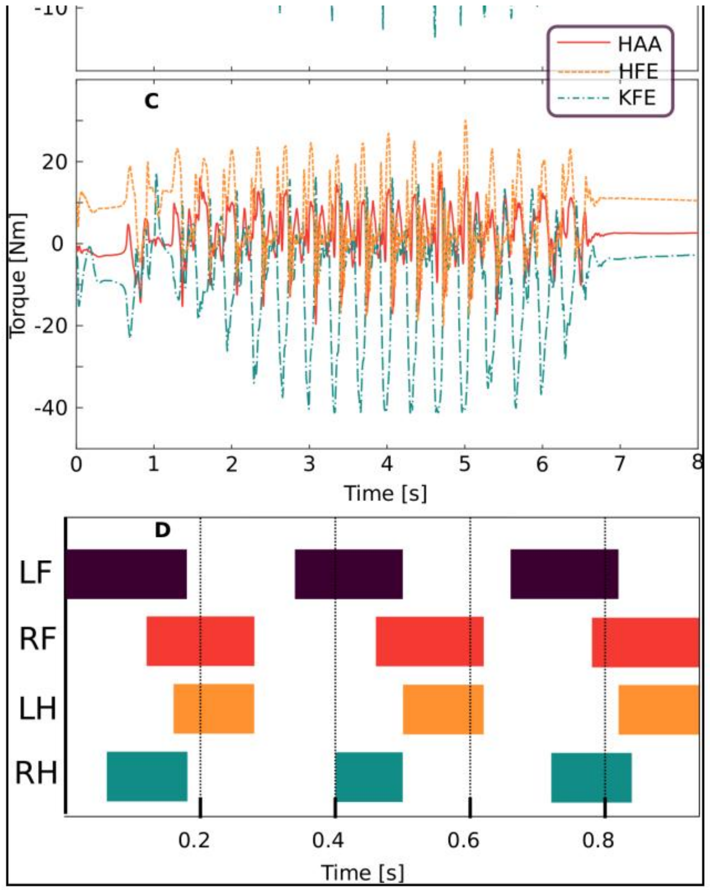


Fig. 3. Evaluation of the trained policy for high-speed loco_x0002_motion. (A) Forward velocity of ANYmal. (B) Joint velocities.(C) Joint torques. (D) Gait pattern. The abbreviations stand for
Left Front (LF) leg,
Right Front (RF) leg,
Left Hind (LH) leg,and
Right Hind (RH) leg, respectively

AncoraSIR.com

Experiment 5

Even state-of-the-art methods cannot limit the actuation during planning due to limitations of their planning module. Modules in their controllers are not aware of the constraints in the later stages and, consequently,their outputs may not be realizable on the physical system in Fig. 3D is distinct from the one exhibited by the command-conditioned locomotion controller. It is close to a flying trot but with significantly longer flight phase and asymmetric flight phase duration.. This is not a commonly observed gait pattern in nature and we suspect that it is among multiple near-optimal solution modes for this task.

AncoraSIR.com

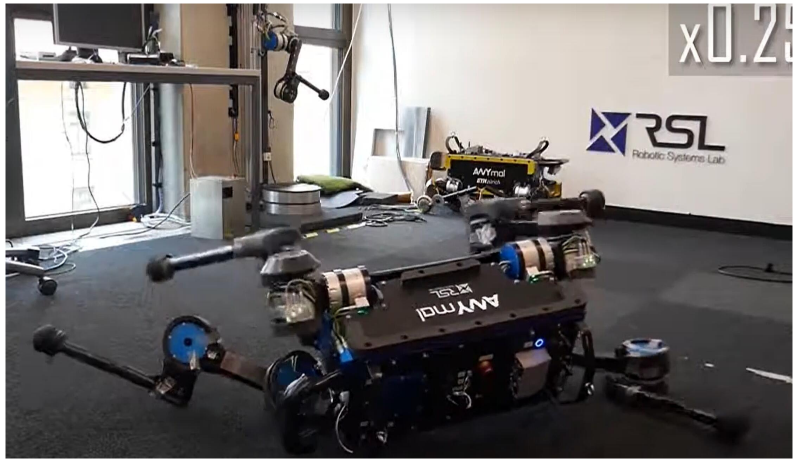
Experiment 6

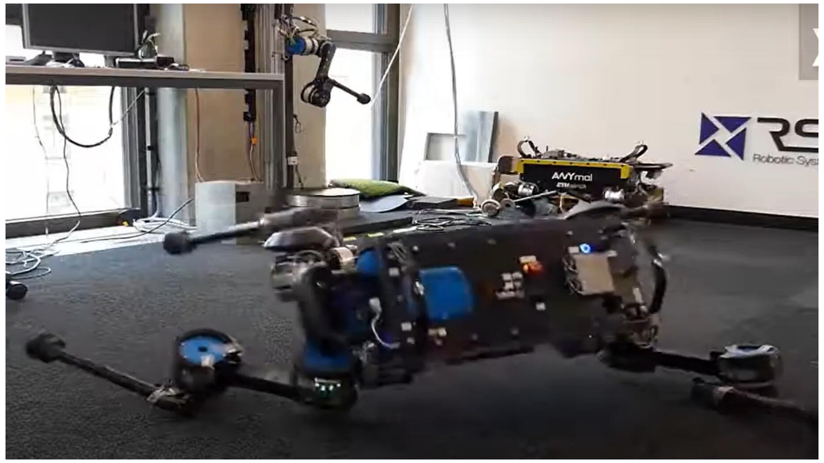
Recovery from a fall

Using the presented methodology, we trained a recovery policy and tested
 it on the real robot.
In all tests, ANYmal successfully flipped itself upright. An example motion is shown in Fig. 4. These agile and dynamic behaviors demonstrate that our approach is able to learn performant controllers for tasks that are difficult or impossible to address with prior methods.

Experiment 6

Recovery from a fall





AncoraSIR.com

Discussion

Outstanding:

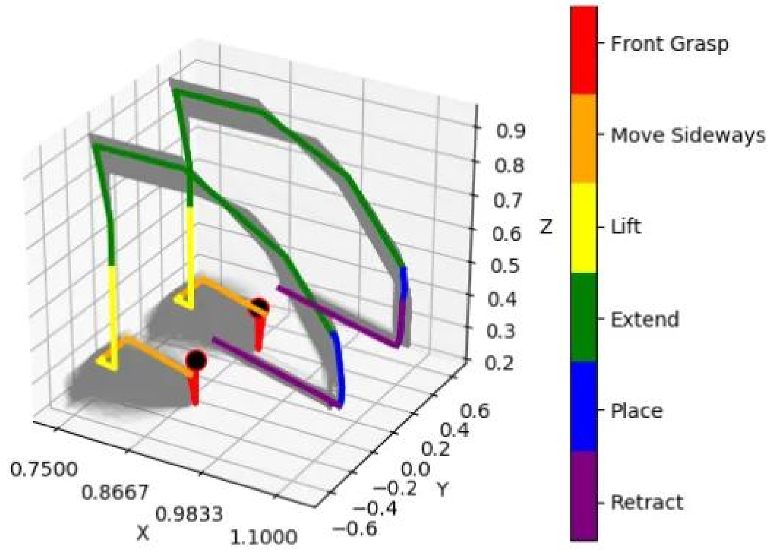
- A more economical and convenient way to control.
- The simulation and learning framework used can be applied to any rigid body system, making it more versatile.

In summary, such a framework can be applied to the development of **complex functions for rigid-body robots in more complex and challenging environments**

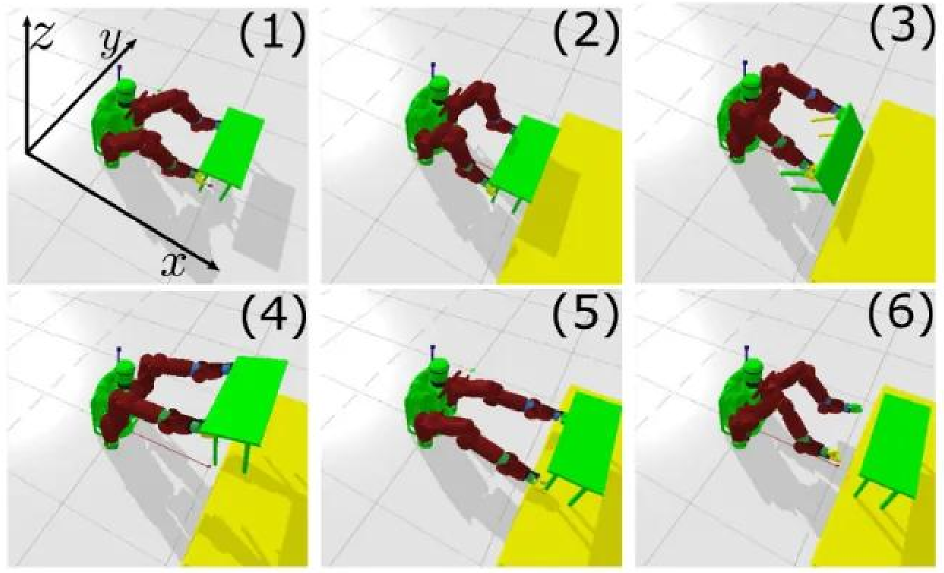
AncoraSIR.com

SUSTech
Southern University of Science and Technology

Discussion



(a)



(b) Image of last step of each primitive, 1. Front Grasp, 2. Move, 3. Lift, 4. Extend, 5. Place, and 6. Retract

Figure 3: Demonstrations for the table lift task. Figure (a) shows the training trajectories in (x, y, z) of left and right grippers for 2500 demonstrations. One sample trajectory is shown in color to highlight the trajectory for each primitive, while the rest are grey. The black dot is the starting location. Figure (b) shows the task executed in our simulator. Each image represents the last step of each primitive. Videos of the demonstrations are provided in the supplementary material.

AncoraSIR.com

Limitations and future work

- Model accuracy: Despite efforts to accurately model the physical components and drive systems of the robot, discrepancies may still exist between the model and the actual system. This can lead to poor performance and safety issues if not properly addressed.

- Time and resource intensive: Modeling the physical components and drive systems of a robot can be a time-consuming and resource-intensive process, especially for complex robots like ANYmal. This can lead to longer and more costly development times.

- Difficulty in modeling certain components: Certain physical components, such as deformed objects or assemblies with complex geometry, may be difficult to model accurately using CAD software. This can lead to errors in the model and affect performance.

AncoraSIR.com

Limitations and future work

- Complexity of drive systems: Certain types of drive systems, such as hydraulic drives with coupled dynamics, may be difficult to model accurately using drive networks. This can lead to performance degradation and safety issues.

- Limited adaptability: Models are often based on assumptions and may not be able to adapt to changes in robot design or environment. This may limit the flexibility of the robot and require further modeling work in the future.

- A single neural network trained in a single training session exhibits a single aspect of behavior that cannot be generalized across multiple tasks. Introducing **hierarchical structures** in policy networks can remedy this and is a promising avenue for future work.

Summary

- Main convern: How to construct a more general and easy-tuning controller of a robot.
- Limits: Complicate models result in time consuming and lots of constrains
- State of art: The most novelty of this method is the use of a four-step approach that integrates physical parameter identification, actuator modeling, and policy training to create a control policy for a physical system.
- This method enables the creation of control policies that can operate directly on the physical system, improving the accuracy and effectiveness of the control while reducing the need for calibration or fine-tuning of the control policy.

SUSTech Design + Learning Lab

陈逸飞 方艺钧
杨德宇 宁晨辉 朱思颖

AncoraSIR.com