Learning-based legged locomotion; state of the art and future perspectives

Sehoon Ha, Joonho Lee, Michiel van de Panne, Zhaoming Xie, Wenhao Yu, Majid Khadiv

김진원

최근 사족보행로봇의 발전 방향에 대한 Survey 논문

- 지난 30년간의 모델 기반 및 학습 기반 접근 방식을 소개함
- 딥러닝, 시뮬레이션, 하드웨어 등을 소개
- 휴머노이드 연구 증가에 따라 이족보행 기술 발전 언급
- 해결되지 않은 문제 및 사회적 영향에 대한 논의

Table of Contents

1. Introduction

2. Theoretical background

3. Components of MDP for Locomotion

4. Learning Frameworks

5. Sim-to-Real Transfer

6. ~~Combining Control and Learning~~

7. ~~From Quadrupeds to Bipeds~~

8. ~~Unsolved Problems and Research Frontiers~~

Introduction

1. Hardware

2. Simulators

3. Control and Learning Algorithms

4. ~~Related Survey Papers~~

Introduction

- 다족로봇에 대한 연구는 1980년대 Raibert 교수를 필두로 1,2,4 족으로 발전되어 옴.
- 특히, 최근 10년 사이에 급속도로 보행로봇의 개발이 진행되고 있음.
- 이는 1. 하드웨어, 2. 시뮬레이션, 3. 학습 알고리즘 등의 이유 등이 있음.
- 본 리뷰 논문에서는 다리 이동에 대한 학습을 주로 다룸

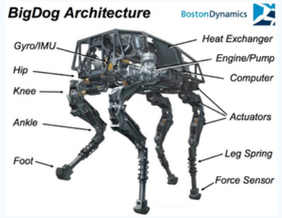
사족보행의 역사

1. M. H. Raibert. Hopping in legged systems—modeling and simulation for the two-dimensional one-legged case. IEEE Transactions on Systems, Man, and Cybernetics, pages 451463, 1984.
2. M. H. Raibert, K. Blankespoor, G. M. Nelson, and R. Playter. Bigdog, the rough-terrain quadruped robot. IFAC Proceedings Volumes, 41:10822–10825, 2008

Introduction

1. Hardware

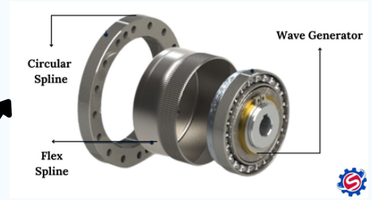
- 대부분 **hydraulic, electric motors with torque sensors, series elastic actuators** 중 하나를 사용.



Hydraulic motor

- 중량 대비 출력 비율이 크기 때문에 매우 역동적인 동작을 수행하고 큰 탑재량을 운반 가능
- **Expensive, need specialized expertise for design, maintenance, and repair**

Electric motor

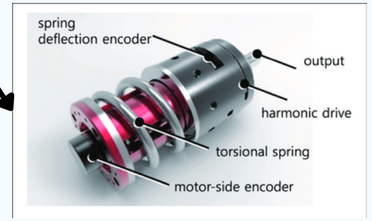


Harmonic drive

- 고토크 전달: 높은 감속비로 인해 전동기의 출력을 충분한 토크로 변환 가능.
- 정밀한 위치 제어: 백래시(Backlash)가 적어 정밀한 제어 가능.
- 높은 정적 마찰: 기어박스의 정적 마찰, 전류 제어만으로 출력 토크 제어 불가.

Quasi direct drive

- 고토크 밀도: 작고 가벼운 크기로도 높은 토크를 출력 가능. -> 전류 제어 가능
- 고속 응답성: 높은 제어 대역폭으로 정밀한 힘/토크 제어 가능.
- 백드라이버블리티 (Backdrivability): 충격 흡수 및 상호작용 제어가 용이, 외부 힘에 대응 가능.



Series elastic actuator

- 충격 흡수 및 내구성 향상: 탄성 요소가 외부 충격을 흡수하여 시스템의 내구성을 높임.
- 정밀한 힘 제어: 탄성 요소의 변위를 측정하여 힘을 제어할 수 있어, 정밀한 힘 제어가 가능.

1. M. H. Raibert. Hopping in legged systems—modeling and simulation for the two-dimensional one-legged case. IEEE Transactions on Systems, Man, and Cybernetics, pages 451463, 1984.
2. M. H. Raibert, K. Blankespoor, G. M. Nelson, and R. Playter. Bigdog, the rough-terrain quadruped robot. IFAC Proceedings Volumes, 41:10822–10825, 2008

Introduction

1. Hardware

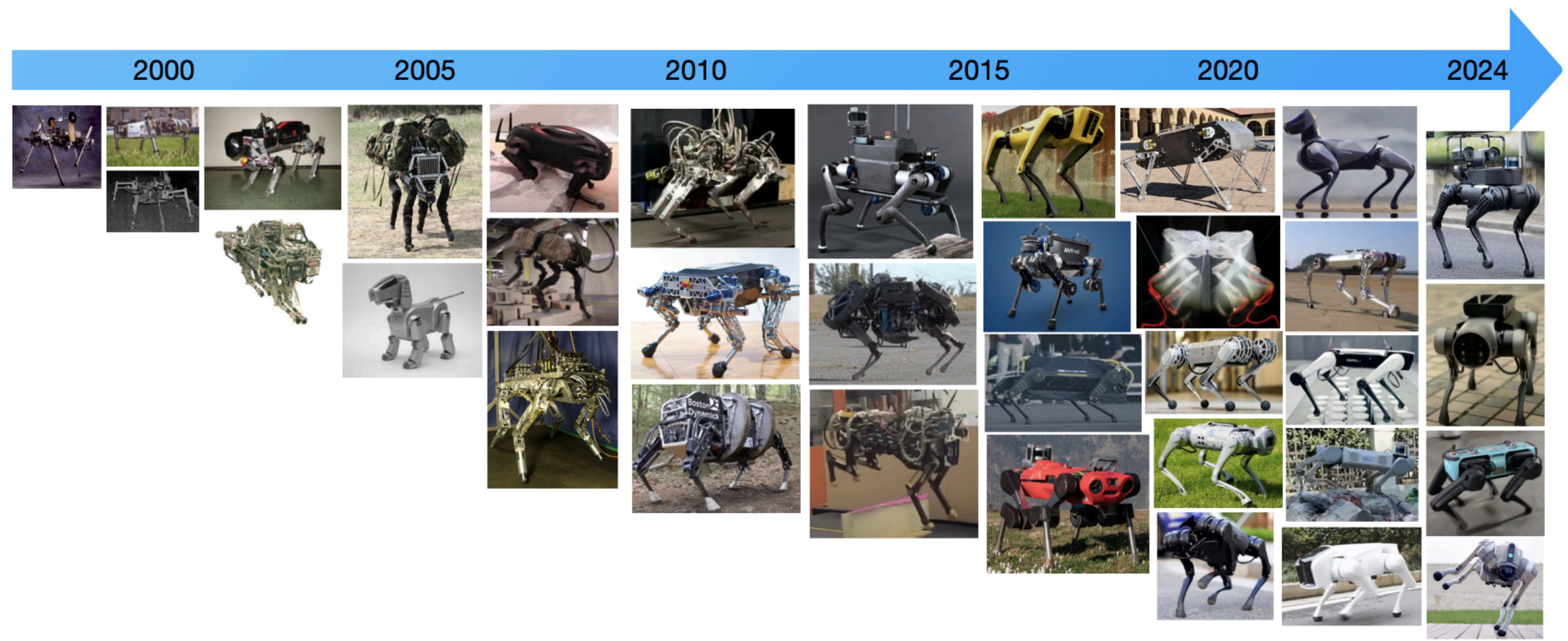


Figure 2. Evolution of quadruped hardware over time

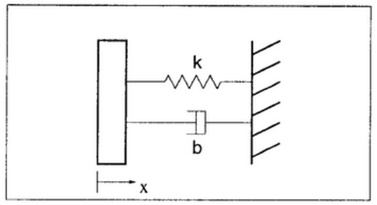
1. M. H. Raibert. Hopping in legged systems—modeling and simulation for the two-dimensional one-legged case. IEEE Transactions on Systems, Man, and Cybernetics, pages 451463, 1984.
2. M. H. Raibert, K. Blankespoor, G. M. Nelson, and R. Playter. Bigdog, the rough-terrain quadruped robot. IFAC Proceedings Volumes, 41:10822–10825, 2008

Introduction

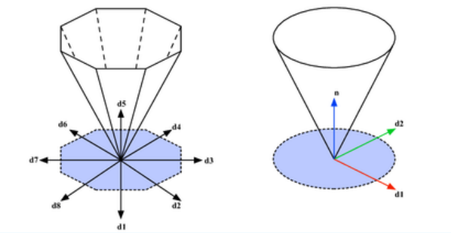
2. Simulators

- **forward and inverse dynamics**를 효율적으로 계산할 수 있게되면서 고차원의 시스템을 시뮬레이션을 할 수 있게 됨
  - 이는 접촉이 없을 때를 가정하는 것이며, 접촉이 있는 경우는 아래와 같이 발전함

- 페널티 기반의 스프링과 댐퍼를 사용하는 접촉 모델링
  - 이는 유연한 접촉을 시뮬레이션할 수 있지만, 기본적으로 **penetration**이 발생해서 현실을 정확히 반영하지 못한다는 단점 존재
  - 이를 해결하기 위해서는 시스템의 강성을 높이고, **timestep**을 줄여야하므로 시뮬레이션 속도가 느려지는 문제 발생



- 이를 해결하기 위해 **elastic or inelastic**의 강체 접촉 모델을 사용
  - **complementarity condition**과 **friction cone constraints**의 제약조건을 사용하여 시뮬레이션 제작



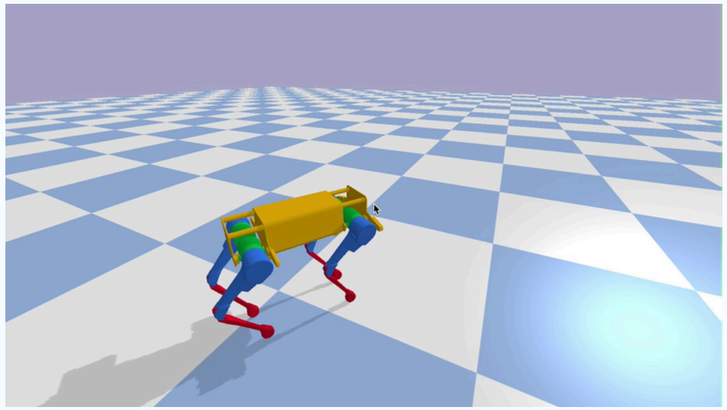
Coulomb friction cone model

1. D. W. Marhefka and D. E. Orin. Simulation of contact using a nonlinear damping model. In Proceedings of IEEE international conference on robotics and automation, volume 2, pages 1662–1668. IEEE, 1996.

Introduction

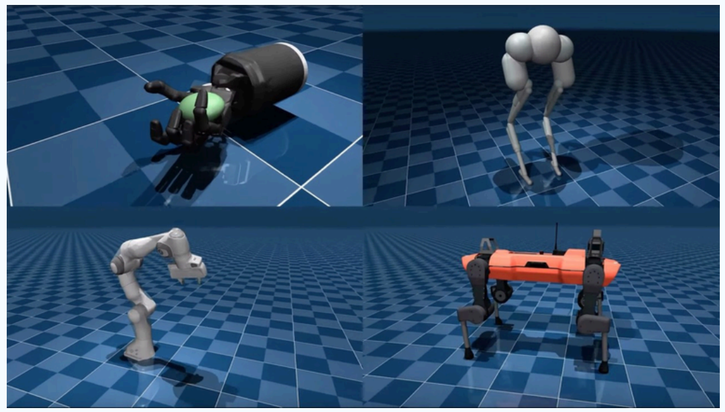
2. Simulators

Simulator	Random External Forces	RGBD + LiDAR	Force Sensor	Multiple Physics Engines	Realistic Rendering
Raisim	✓	✓	✗	✗	✓, Unity
Gazebo	✓	✓	✓	✓	✗
Nvidia Isaac	✓	✓	✗	✗	✓, Unity + Unreal
MuJoCo	✓	✓	✓	✗	✗
PyBullet	✓	✓	✓	✗	✗
CARLA	✗	✓	✗	✗	✓, Unreal
Webots	✓	✓	✓	✗	✗
CoppeliaSim	✓	✓	✓	✓	✗

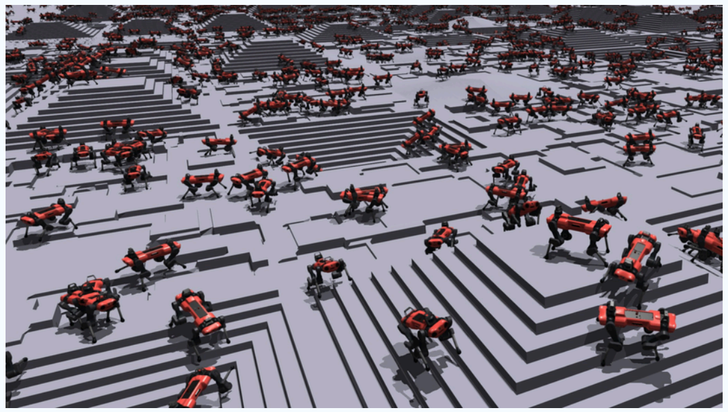


PyBullet

RaiSim



MuJoCo



Nvidia Isaac

1. D. W. Marhefka and D. E. Orin. Simulation of contact using a nonlinear damping model. In Proceedings of IEEE international conference on robotics and automation, volume 2, pages 1662–1668. IEEE, 1996.

Introduction

3. Control and Learning Algorithms

- **Template** 모델 기반의 제어가 많이 사용되어 옴
  - **Raibert heuristic**
  - **Central Pattern Generators (CPG)**

- 이후, **Optimal Cotrol(OC)**과 **Reinforcement Learning(RL)**가 발전해옴
  - **OC: forward model of the system dynamics to help solve for a locally optimal time-indexed control policy by minimizing a performance cost, typically over a finite future horizon**
  - **RL: state-indexed optimal policy by maximizing the expected reward based on collected samples from rolling out a control policy**

Introduction

3. Control and Learning Algorithms

- **Optimal Control: quadratic program으로 문제를 변환 후, Convex optimzation을 활용**
  - **linear dynamics over a finite future time horizon**

$$\dot{x}(t) = 5at^4 + 4bt^3 + 3ct^2 + 2dt + e, \quad (8)$$

$$\ddot{x}(t) = 20at^3 + 12bt^2 + 6ct + 2d. \quad (9)$$

Squaring and integrating the function for acceleration, we get the cost function for a single spline segment, which can be written in matrix form as:

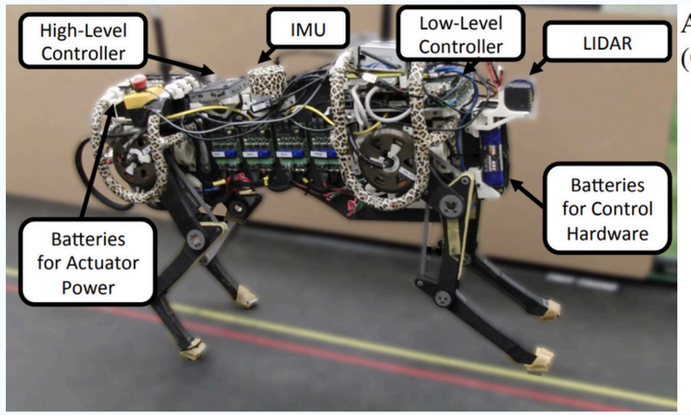
$$\int_0^{\mathrm{T}} \ddot{x}^2(t)\, dt = \mathbf{q}^{\mathrm{T}}\mathbf{G}\mathbf{q}, \quad (10)$$

where

$$\mathbf{q} = \begin{bmatrix} a & b & c & d \end{bmatrix}^{\mathrm{T}},$$

$$\mathbf{G} = \begin{bmatrix} \frac{400}{7}T^7 & 40T^6 & \frac{120}{5}T^5 & 10T^4 \\ 40T^6 & \frac{144}{5}T^5 & 18T^4 & 8T^3 \\ \frac{120}{5}T^5 & 18T^4 & 12T^3 & 6T^2 \\ 10T^4 & 8T^3 & 6T^2 & 4T \end{bmatrix},$$

ZMP-constrained Body Trajectory Generation



An MPC problem is formulated as a quadratic programming (QP) problem with optimal cost $c(N)$ as:

$$c(N) = \min \ (\mathbf{x}_N - \mathbf{x}_N^d)^T \mathbf{Q}_F (\mathbf{x}_N - \mathbf{x}_N^d) + \frac{1}{N}\sum_{i=0}^{N-1} r_i \Delta v_i^2 \quad (6)$$

$$\text{s.t. } \mathbf{x}_{i+1} = \mathbf{A}\,\mathbf{x}_i + \mathbf{B}\,\Delta v_i$$

$$\underline{\mathbf{x}}_N \le \mathbf{x}_N \le \overline{\mathbf{x}}_N$$

$$\underline{v} \le v_i \le \overline{v}$$

$$|\Delta v_i| \le \beta v_i$$

MPC formulation

1. M. Kalakrishnan, J. Buchli, P. Pastor, M. Mistry, and S. Schaal. Learning, planning, and control for quadruped locomotion over challenging terrain. The International Journal of Robotics Research, 30(2):236–258, 2011.
2. H.-W. Park, P. M. Wensing, S. Kim, et al. Online planning for autonomous running jumps over obstacles in high-speed quadrupeds. In Proceedings of Robotics: Science and Systems, 2015.

Introduction

3. Control and Learning Algorithms

- **Optimal Control: quadratic program으로 문제를 변환 후, Convex optimzation을 활용**
  - **the current instant in time, acting as an inverse dynamics controller**

4.2.1 Acceleration and Torque Optimization

A well known method from inverse dynamics control with floating base systems (e.g. Mistry et al., 2010) is to use projected system dynamics as illustrated in (3)

$$\mathbf{P}_F \left(\mathbf{M}\ddot{\mathbf{q}} + \mathbf{h}\right) = \mathbf{P}_F \mathbf{S}^T \boldsymbol{\tau}, \tag{26}$$

such that the dimensionality of the optimization vector can be reduced to

$$\mathbf{x}^{rF} = \begin{pmatrix} \ddot{\mathbf{q}} \\ \boldsymbol{\tau} \end{pmatrix}. \tag{27}$$

While there exist different choices for $\mathbf{P}_F$ (Righetti et al., 2011a), we apply in the experimental section of this paper the well established QR decomposition of $\mathbf{J}_s^T = \mathbf{Q}\begin{bmatrix}\mathbf{R}^T & \mathbf{0}\end{bmatrix}^T$ with the orthogonal matrix $\mathbf{Q}^T = \mathbf{Q}^{-1}$ and the upper right triangular matrix $\mathbf{R}$. Through the decomposition of $\mathbf{Q} = \begin{bmatrix}\mathbf{Q}_c & \mathbf{Q}_u\end{bmatrix}$ into constrained and unconstrained components, the linear mapping is written as $\mathbf{P}_F = \mathbf{Q}_u^T \in \mathbb{R}^{n_m \times n}$. The QR decomposition gives additionally direct access to the contact force

$$\mathbf{F}_s = \mathbf{R}^{-1}\mathbf{Q}_c^T \left(\mathbf{S}^T\boldsymbol{\tau} - \left(\mathbf{M}\ddot{\mathbf{q}} + \mathbf{h}\right)\right). \tag{28}$$

Using this methodology, the optimization procedure results to

Table 2: Least squares optimization matrices for the reduced optimization vector $\mathbf{x}^{rF}$ (27).

EoM (26)	motion task (13)	torque task (14)	force task (15), (28)
$\mathbf{A} = \mathbf{P}_F\left[\mathbf{M}, -\mathbf{S}^T\right]$ (29)	$\mathbf{A} = \left[\mathbf{J}_i, \mathbf{0}\right]$ (31)	$\mathbf{A} = \left[\mathbf{0}, \mathbf{W}_\tau\right]$ (33)	$\mathbf{A} = \mathbf{W}_F\mathbf{R}^{-1}\mathbf{Q}_c^T\left[-\mathbf{M}, \mathbf{S}^T\right]$ (35)
$\mathbf{b} = -\mathbf{P}_F\mathbf{h}$ (30)	$\mathbf{b} = \ddot{\mathbf{r}}_{i,\mathrm{des}} - \dot{\mathbf{J}}_i\dot{\mathbf{q}}$ (32)	$\mathbf{b} = \mathbf{b}_\tau$ (34)	$\mathbf{b} = \mathbf{b}_F + \mathbf{W}_F\mathbf{R}^{-1}\mathbf{Q}_c^T\mathbf{h}$ (36)

1. M. Hutter, H. Sommer, C. Gehring, M. Hoepflinger, M. Bloesch, and R. Siegwart. Quadrupedal locomotion using hierarchical operational space control. The International Journal of Robotics Research, 33(8):1047–1062, 2014.
2.

Introduction

3. Control and Learning Algorithms

- 이후, 다음과 같은 알고리즘들이 개발됨
  - 다중 접촉 행동에 대한 계획 및 제어
    - **differential dynamic programming (DDP) with relaxed contact**
    - **contact-invariant optimization**
    - **contact-implicit optimization**
    - **mixed integer convex optimization**
    - **phase-based parameterization of the end-effector trajectories**
  - **OC**는 아래의 문제들이 있음 -> 강화학습으로 넘어가는 토대가 됨
    - **Real-Time Computational Challenges**
    - **Handling Uncertainties in Contact Interactions**
    - **Integration of Sensor Modalities**

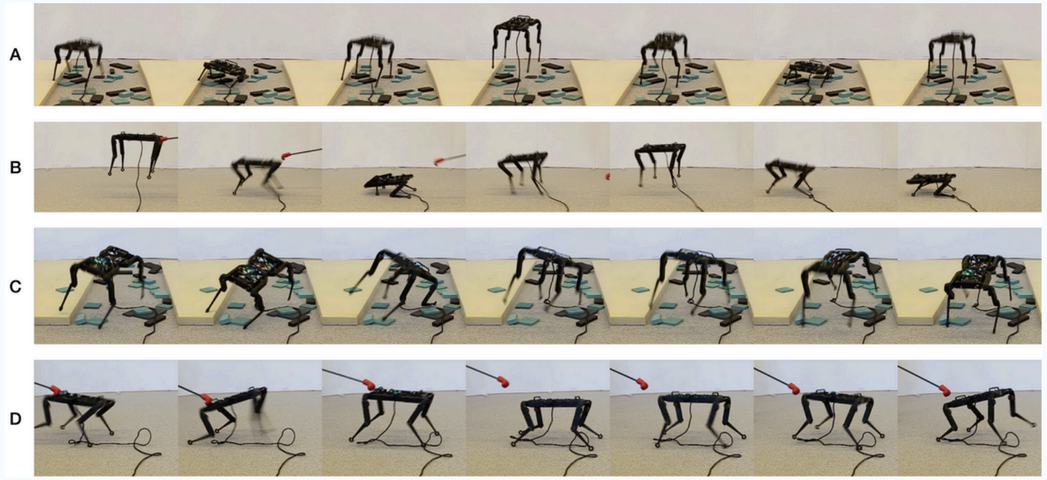
1. M. Hutter, H. Sommer, C. Gehring, M. Hoepflinger, M. Bloesch, and R. Siegwart. Quadrupedal locomotion using hierarchical operational space control. The International Journal of Robotics Research, 33(8):1047–1062, 2014.
2.

Introduction

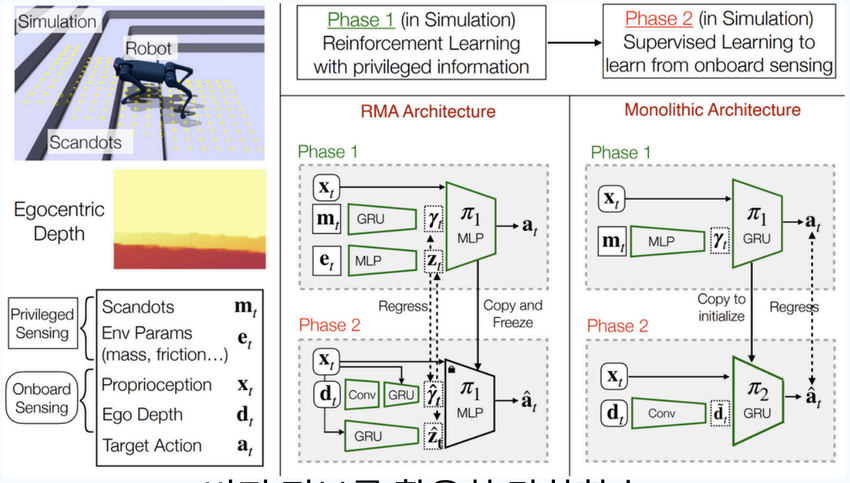
3. Control and Learning Algorithms

- **Reinforcement Lenarning**
  - 딥러닝의 진화와 더불어, 딥강화학습 또한 빠르게 발전해왔음.

강화학습



Demonstartion learning



비전 정보를 활용한 강화학습

1. J. Lee, J. Hwangbo, L. Wellhausen, V. Koltun, and M. Hutter. Learning quadrupedal locomotion over challenging terrain. Science robotics, 5(47):eabc5986, 2020.
2. M. Bogdanovic, M. Khadiv, and L. Righetti. Modelfree reinforcement learning for robust locomotion using demonstrations from trajectory optimization. Frontiers in Robotics and AI, 9, 2022.
3. A. Agarwal, A. Kumar, J. Malik, and D. Pathak. Legged locomotion in challenging terrains using egocentric vision. In Conference on Robot Learning, pages 403–415. PMLR, 2023.

Theoretical background

1. ~~Markov Decision Process and Reinforcement Learning~~

2. ~~Deep Reinforcement Learning~~

3. Behavior Cloning and Imitation Learning

Theoretical background

1. Markov Decision Process and Reinforcement Learning

- **Locomtion 학습 시에는 PPO, TRPO를 주로 사용함**
  - 샘플 효율성을 무시하며 최적의 성능을 낼 수 있도록 함, 이는 시뮬레이션 환경에서의 학습에서 최적임

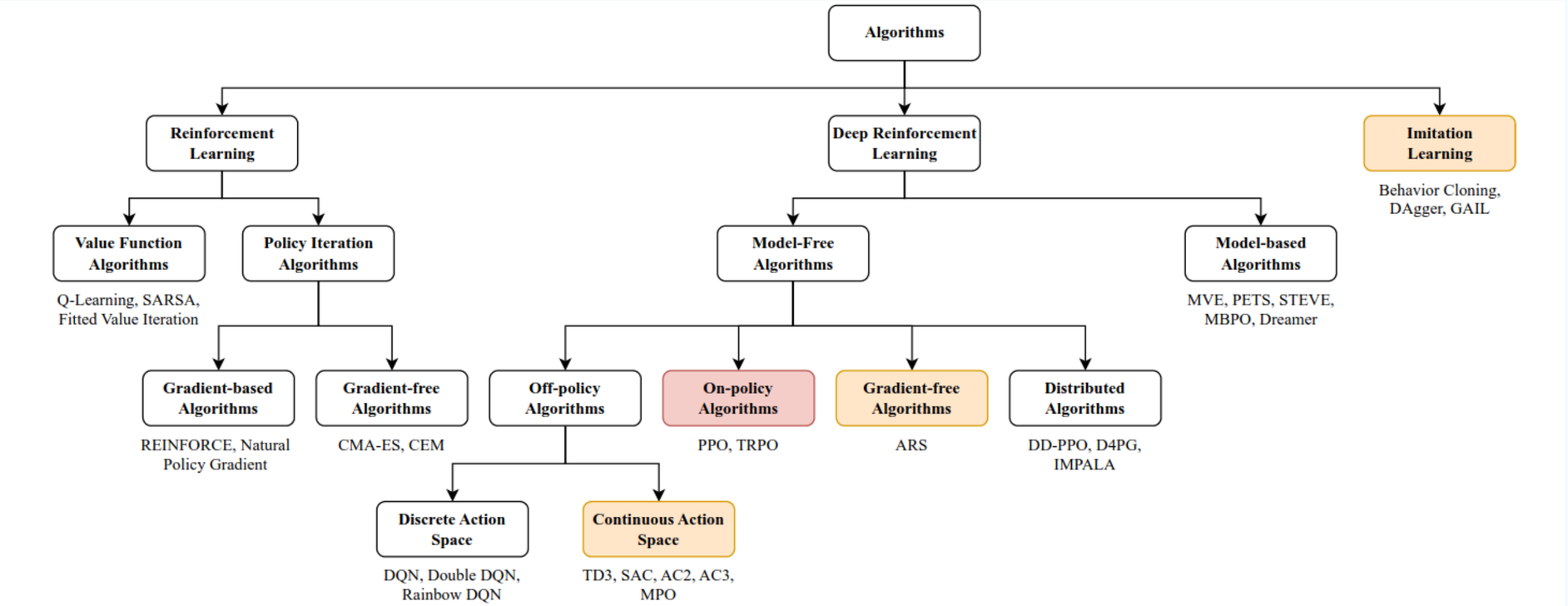
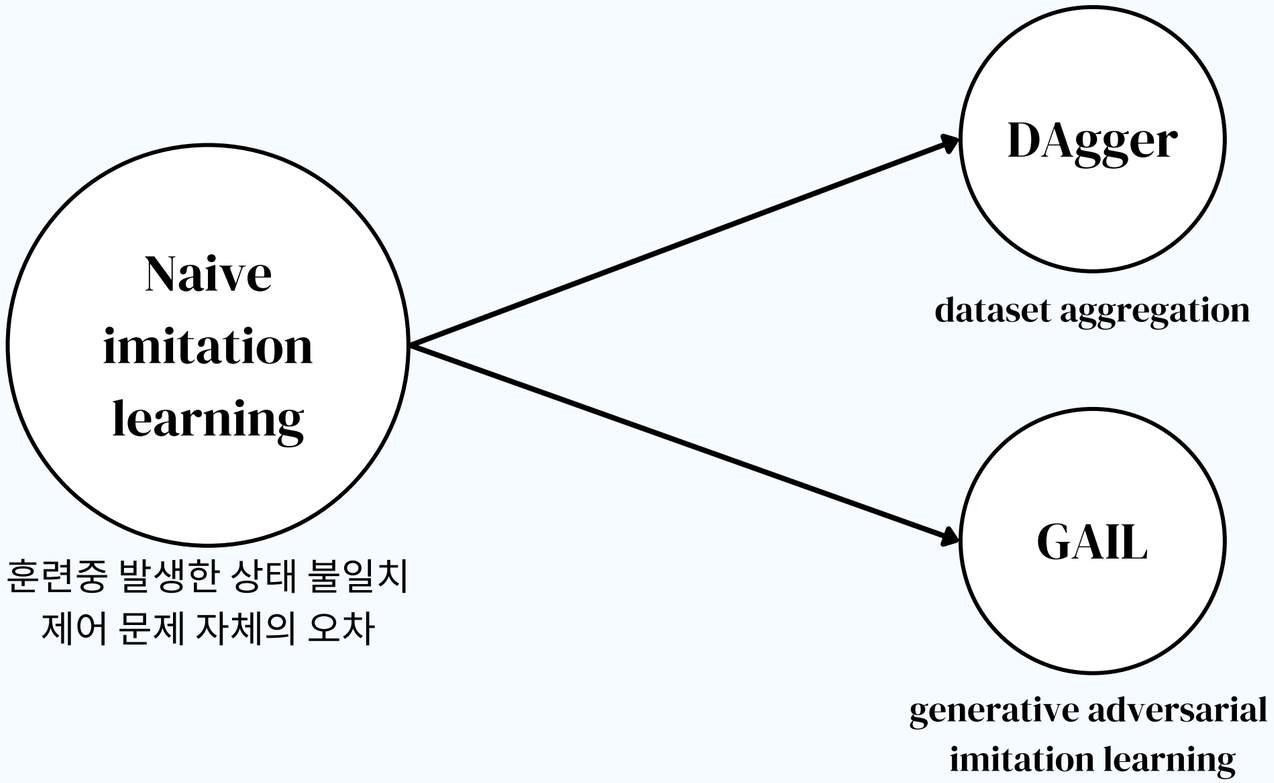


Figure 3. Taxonomy of learning algorithms. Popular algorithms are highlighted in color. On-policy algorithms (red), such as TRPO or PPO, have been the most frequent choice for legged locomotion. Other off-policy, gradient-free, or imitation learning algorithms (orange) are also selected for sample efficiency or better styles. For more details, please refer to the background section.

Theoretical background

3. Behavior Cloning and Imitation Learning

- 일반적인 **RL**은 **sparse reward** 환경에서 수많은 **reward engineering**이 필요함.
- 이에 **expert** 데이터를 활용하여 학습하는 연구가 진행됨.



Initialize $\mathcal{D} \leftarrow \emptyset$.
Initialize $\hat{\pi}_1$ to any policy in Π.
for $i = 1$ **to** N **do**
  Let $\pi_i = \beta_i \pi^* + (1 - \beta_i)\hat{\pi}_i$.
  Sample T-step trajectories using π_i.
  Get dataset $\mathcal{D}_i = \{(s, \pi^*(s))\}$ of visited states by π_i and actions given by expert.
  Aggregate datasets: $\mathcal{D} \leftarrow \mathcal{D} \bigcup \mathcal{D}_i$.
  Train classifier $\hat{\pi}_{i+1}$ on $\mathcal{D}$.
end for
Return best $\hat{\pi}_i$ on validation.

Algorithm 3.1: DAGGER Algorithm.

Algorithm 1 Generative adversarial imitation learning

1: **Input:** Expert trajectories $\tau_E \sim \pi_E$, initial policy and discriminator parameters θ_0, w_0
2: **for** $i = 0, 1, 2, \ldots$ **do**
3: Sample trajectories $\tau_i \sim \pi_{\theta_i}$
4: Update the discriminator parameters from w_i to w_{i+1} with the gradient

$$\hat{\mathbb{E}}_{\tau_i}[\nabla_w \log(D_w(s,a))] + \hat{\mathbb{E}}_{\tau_E}[\nabla_w \log(1 - D_w(s,a))] \tag{17}$$

5: Take a policy step from θ_i to θ_{i+1}, using the TRPO rule with cost function $\log(D_{w_{i+1}}(s,a))$. Specifically, take a KL-constrained natural gradient step with

$$\hat{\mathbb{E}}_{\tau_i}[\nabla_\theta \log \pi_\theta(a|s) Q(s,a)] - \lambda \nabla_\theta H(\pi_\theta), \quad \text{where } Q(\bar{s},\bar{a}) = \hat{\mathbb{E}}_{\tau_i}[\log(D_{w_{i+1}}(s,a)) \mid s_0 = \bar{s}, a_0 = \bar{a}] \tag{18}$$

6: **end for**

1. S. Ross, G. Gordon, and D. Bagnell. A reduction of imitation learning and structured prediction to no-regret online learning. In Proceedings of the fourteenth international conference on artificial intelligence and statistics, pages 627–635. JMLR Workshop and Conference Proceedings, 2011.
2. J. Ho and S. Ermon. Generative adversarial imitation learning. Advances in neural information processing systems, 29, 2016.

Components of MDP for Locomotion

1. Dynamics

2. Observation

3. Reward

4. Action Space

Components of MDP for Locomotion

1. Dynamics

- **MDP**로 공식화 하기 위해, **Dynamics**를 이산화 하면 아래와 같음

$$\mathbf{s}_{t+1} = \mathbf{s}_t + f(\mathbf{s}_t, \mathbf{a}_t)dt, \quad p(\mathbf{s}_{t+1}|\mathbf{s}_t, \mathbf{a}_t) \quad (\mathbf{s}_t, \mathbf{a}_t, \mathbf{r}_t, \mathbf{s}_{t+1})$$

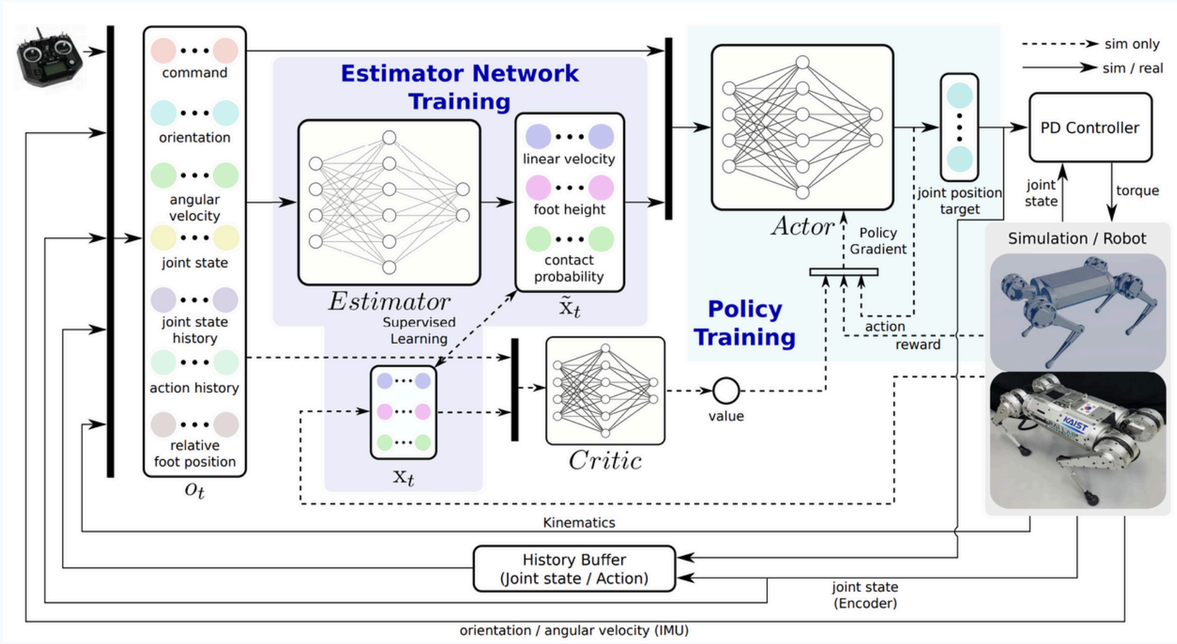
- **Dynamics**를 구현하여, 로봇을 학습하는 것은 **simulation** / **real-world** 두가지 방법이 있음
  - **Simulation:** 학습 데이터를 얻기 쉬움, 실제 세상의 물리 법칙과 오류가 있을 수 있음
  - **Real-world:** 실제 세상에서의 데이터를 취득 가능, 많은 데이터를 얻기 어려우며 데이터 취득 시 안전을 고려해야함

1. S. Ha, P. Xu, Z. Tan, S. Levine, and J. Tan. Learning to walk in the real world with minimal human effort. arXiv preprint arXiv:2002.08550, 2020..
2. T.-Y. Yang, T. Zhang, L. Luu, S. Ha, J. Tan, and W. Yu. Safe reinforcement learning for legged locomotion. In 2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pages 2454–2461. IEEE, 2022b.
3. L. Smith, J. C. Kew, T. Li, L. Luu, X. B. Peng, S. Ha, J. Tan, and S. Levine. Learning and adapting agile locomotion skills by transferring experience. arXiv preprint arXiv:2304.09834, 2023.

Components of MDP for Locomotion

2. Observation

- **Proprioception**
  - 일반적으로 **IMU, joint encoder, contact sensor**가 있음.
  - 초기에는 원시센서 값을 그대로 쓰기보다는, 정제하여 **base pose, base twist, joint position, and velocity**를 **observation**으로 사용했음.
  - 최근에는 오히려 원시센서값을 그대로 사용하며, **history** 정보를 추가로 넣어 의미론적인 정보를 학습할 수 있게 만듦



1. G. Ji, J. Mun, H. Kim, and J. Hwangbo. Concurrent training of a control policy and a state estimator for dynamic and robust legged locomotion. IEEE Robotics and Automation Letters, 7 (2):4630–4637, 2022a.

Components of MDP for Locomotion

2. Observation

- **Exteroception (RGB, RGBD, LiDAR 등 외부 센서를 의미)**
  - 초기에는 정제된 데이터를 입력으로 넣어줌 (**elevation map, voxel map 등**)

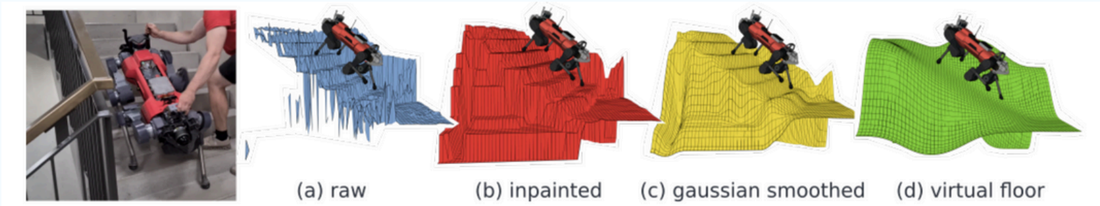
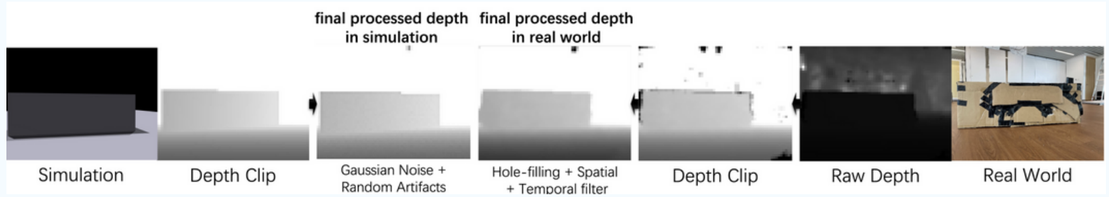
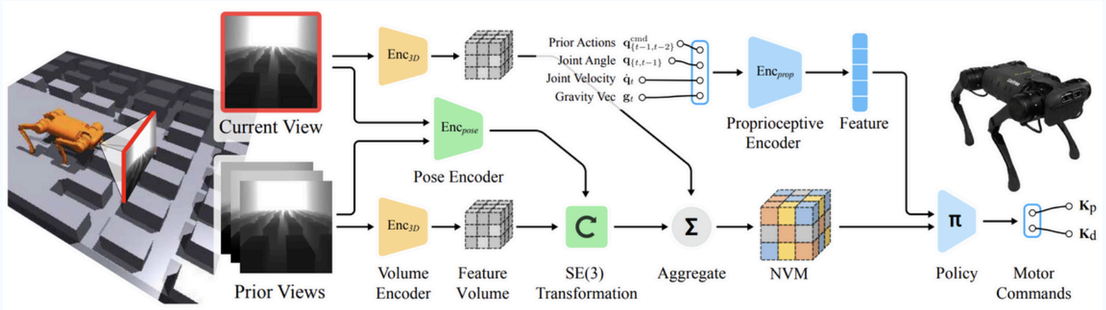


Fig. 9. Different height layers used in TAMOLS [22]: First, the occluded regions in the raw map are in-painted. Then, two additional layers are derived from it, gradually increasing the smoothness.

  - 최근에는, **raw sensory data**를 그대로 입력으로 넣어주는 경향이 있음



  - 또한, 센서의 값을 **latent-space**로 압축하여 사용하기도 함.



1. T. Miki, L. Wellhausen, R. Grandia, F. Jenelten, T. Homberger, and M. Hutter. Elevation mapping for locomotion and navigation using gpu. In 2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pages 2273–2280. IEEE, 2022b.
2. Z. Zhuang, Z. Fu, J. Wang, C. Atkeson, S. Schwertfeger, C. Finn, and H. Zhao. Robot parkour learning. arXiv preprint arXiv:2309.05665, 2023.
3. R. Yang, G. Yang, and X. Wang. Neural volumetric memory for visual locomotion control. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pages 1430–1440, 2023b.

Components of MDP for Locomotion

2. Observation

- **Task-related inputs (Goals)**
  - **command inputs like velocity and pose**
  - **CPG와 같은 구조화된 action을 위한 phase(T), trajectory patterns**

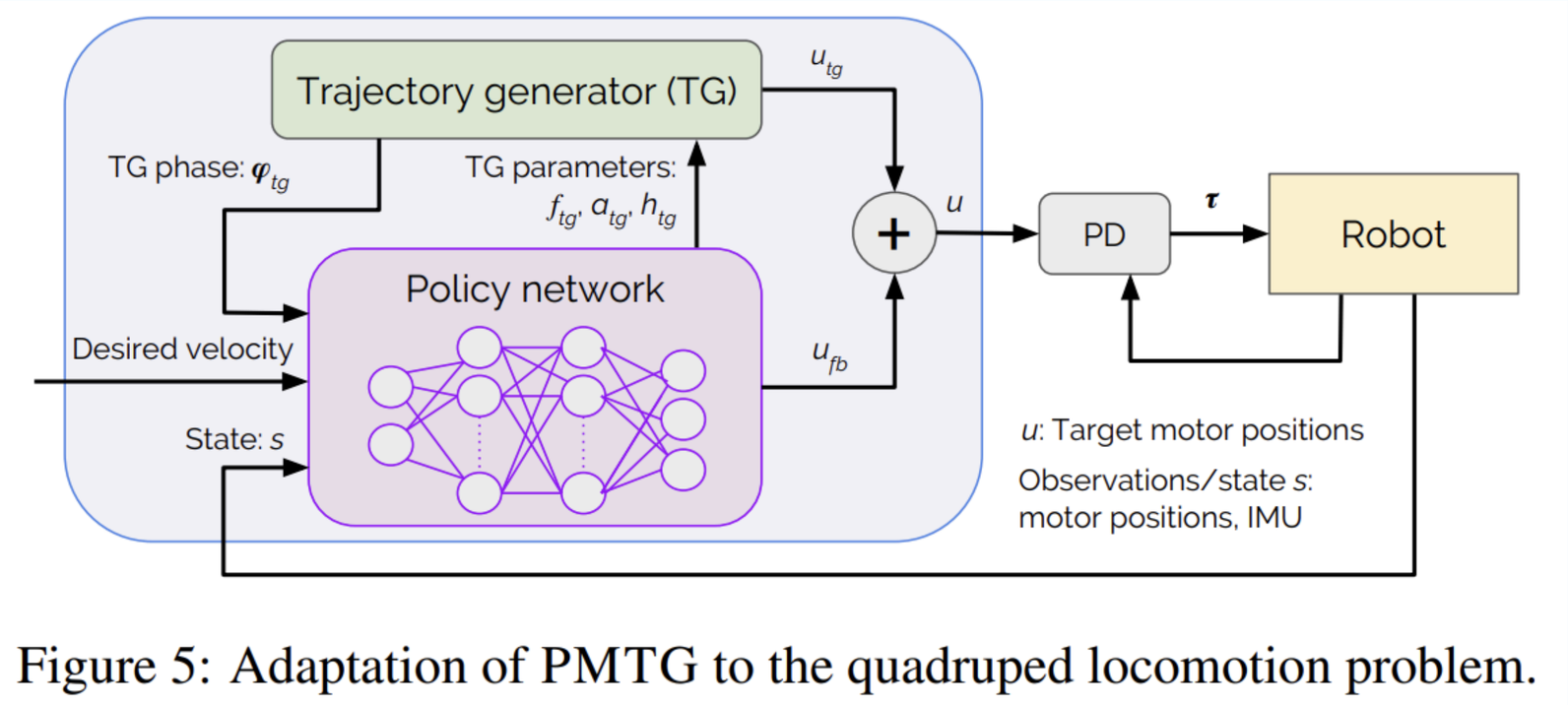


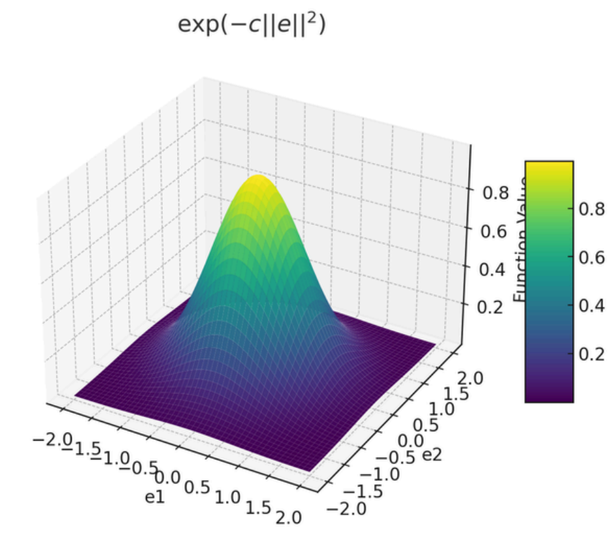
Figure 5: Adaptation of PMTG to the quadruped locomotion problem.

1. A. Iscen, K. Caluwaerts, J. Tan, T. Zhang, E. Coumans, V. Sindhwani, and V. Vanhoucke. Policies modulating trajectory generators. In Conference on Robot Learning, pages 916–926. PMLR, 2018.

Components of MDP for Locomotion

3. Reward

- **linear combination of various reward and penalty terms** $r(\mathbf{s}_t, \mathbf{s}_{t+1}, \mathbf{s}_t) = \sum_i c_i r_i(\mathbf{s}_t, \mathbf{s}_{t+1}, \mathbf{a}_t)$
- **Manual reward shapingManual reward shaping**
  - **velocity tracking, pose tracking, and other regularization terms**
  - **soft cost to this function** $exp(-c||e||^2)$ ⟵ **Bounded function**
    - **e.g., limiting the magnitude of joint velocity, acceleration, and base pose during locomotion**



Bounded function

Table 2. Common rewards used for training quadruped locomotion. An asterisk $(*)$ indicates a target quantity. The function $\phi(x) := \exp(-k||x||^2)$, where k is a scaling factor, is used to define bounded rewards. Here, e_z^b represents the z-axis of the world in the base frame, q denotes the joint angle, τ represents the joint torque, and a_t is the action. Reward functions can be similarly defined using different norms or element-wise operations.

Horizontal velocity tracking	$\phi(v_{xy}^* - v_{xy})$
Yaw rate tracking	$\phi(\omega_z^* - \omega_z)$
Base z motion	$-v_z^2$ or $\phi(v_z)$
Roll and pitch rate	$-\|\|\omega_{xy}\|\|^2$ or $\phi(\omega_{xy})$
orientation	$-\|\|[0,0,1]^T - e_z^b\|\|^2$
Joint velocities	$-\sum_{i\in\text{joints}} \|\dot{q}_i\|^2$
Joint accelerations	$-\sum_{i\in\text{joints}} \|\ddot{q}_i\|^2$
Joint torques	$-\sum_{i\in\text{joints}} \|\tau_i\|^2$
Joint mechanical power	$-\sum_{i\in\text{joints}} (\tau_i * \dot{q}_i)^2$
Action rate	$-\|\mathbf{a}_t - \mathbf{a}_{t-1}\|^2$
Action smoothness	$-\|\mathbf{a}_t - 2\mathbf{a}_{t-1} + \mathbf{a}_{t-2}\|^2$

Components of MDP for Locomotion

3. Reward

- **Imitation reward**
  - **leverage dog motion capture data to enable quadrupedal robots to learn animal like motions**

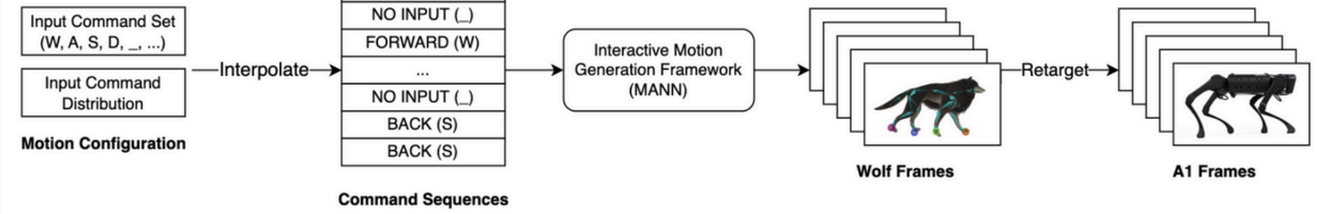


Fig. 2. Motion dataset generation pipeline.

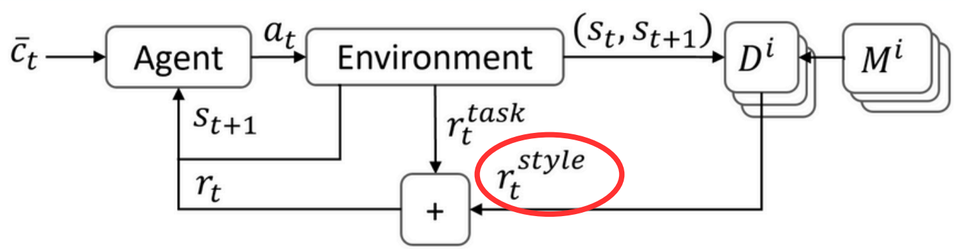


Fig. 2. Multi-AMP overview: The discriminator predicts a style reward s_t^{style} which is high if the policy's behavior is similar to the motions of the motion-data base M^i, by distinguishing between state transitions (s_t, s_{t+1}) of both sources. The style reward is added to the task reward, which finally leads to the policy fulfilling the task while applying the motion data's style.

1. X. B. Peng, E. Coumans, T. Zhang, T.-W. Lee, J. Tan, and S. Levine. Learning agile robotic locomotion skills by imitating animals. arXiv preprint arXiv:2004.00784, 2020.
2. E. Vollenweider, M. Bjelonic, V. Klemm, N. Rudin, J. Lee, and M. Hutter. Advanced skills through multiple adversarial motion priors in reinforcement learning. In 2023 IEEE International Conference on Robotics and Automation (ICRA), pages 51205126. IEEE, 2023.

Components of MDP for Locomotion

4. Action space

- **Low-Level Joint Commands**
  - $\tau = k_p(a(s) - \theta) - k_d\dot{\theta}$

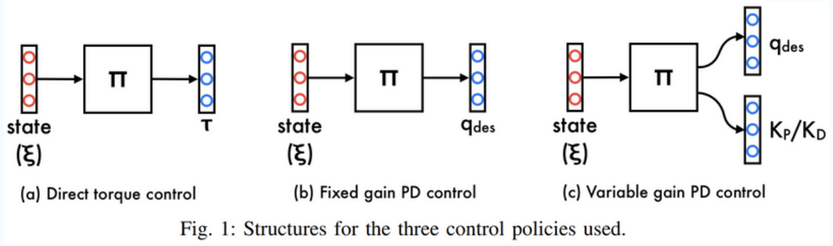
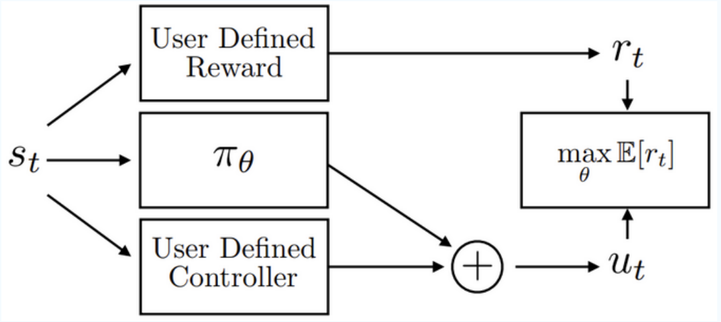
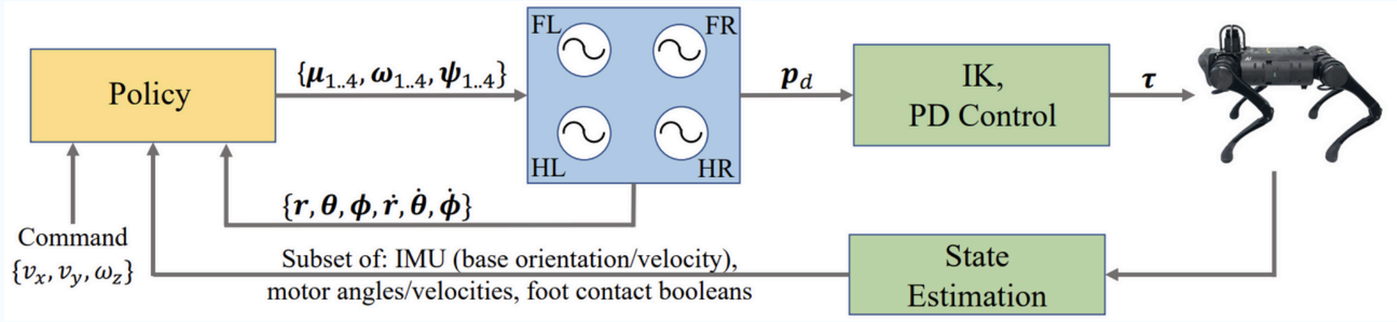


Fig. 1: Structures for the three control policies used.

PD output: 50Hz 정도만 되면 됨
토크 *output:* 구조적 제약 없음, *1KHz* 이상을 내야함

- **Structured Action Spaces**
  - 사전 지식을 통합하여 제어를 함 (CPG, gait sequence 등)

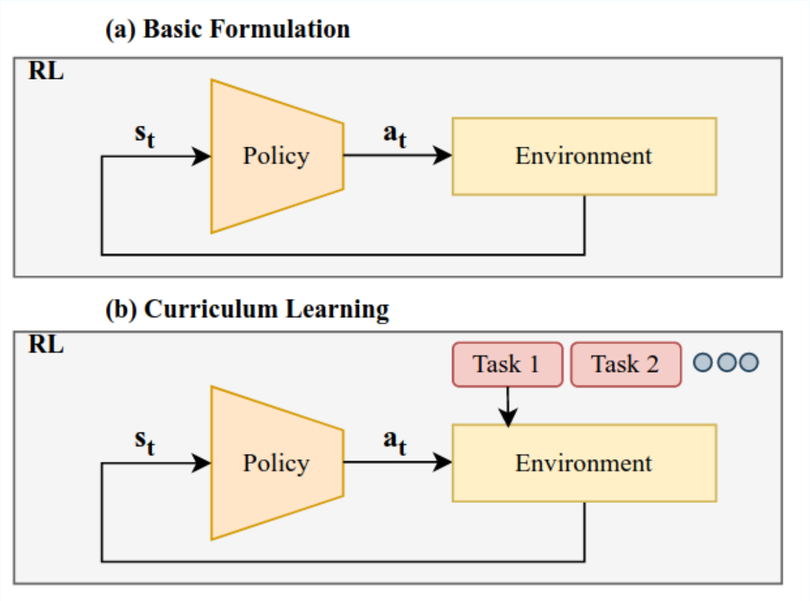


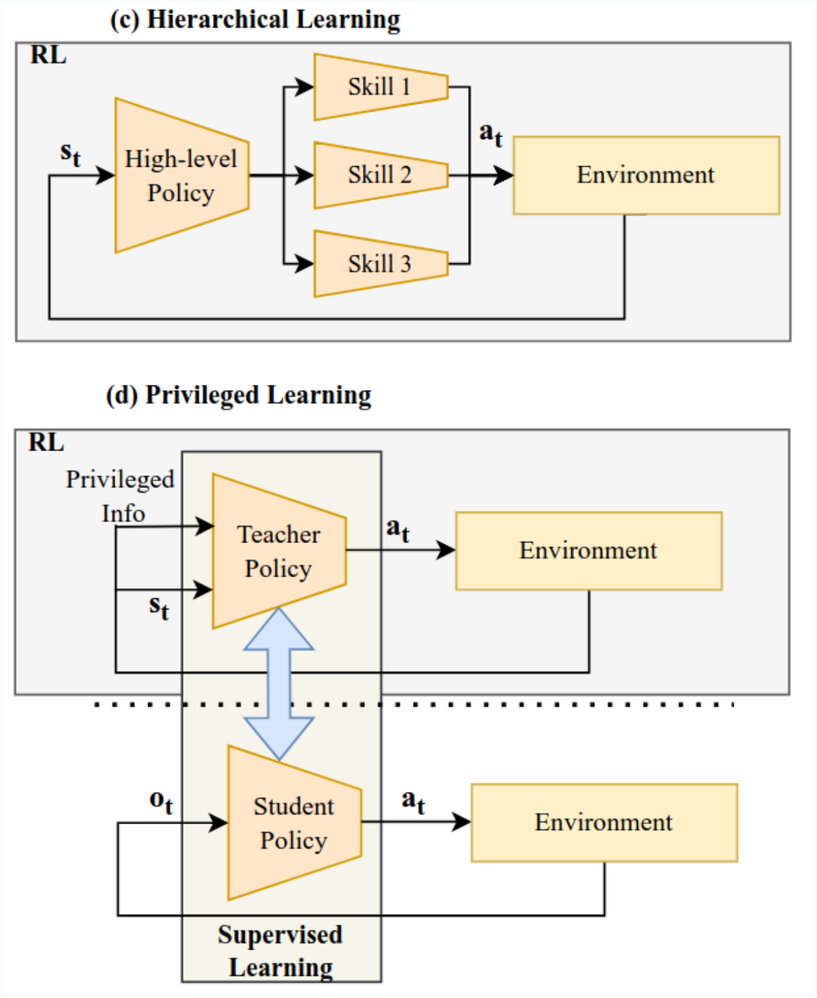


1. M. Bogdanovic, M. Khadiv, and L. Righetti. Learning variable impedance control for contact sensitive tasks. IEEE Robotics and Automation Letters, 5(4):6129–6136, 2020.
2. T. Johannink, S. Bahl, A. Nair, J. Luo, A. Kumar, M. Loskyll, J. A. Ojea, E. Solowjow, and S. Levine. Residual reinforcement learning for robot control. In 2019 International Conference on Robotics and Automation (ICRA), pages 6023–6029. IEEE, 2019.
3. G. Bellegarda and A. Ijspeert. Cpg-rl: Learning central pattern generators for quadruped locomotion. IEEE Robotics and Automation Letters, 7(4):12547–12554, 2022.

Learning Frameworks

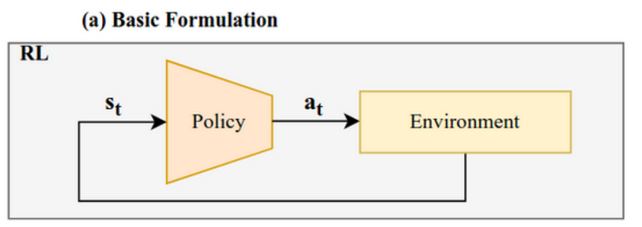
1. End-to-end Learning
2. Curriculum Learning
3. Hierarchical Learning
4. Privileged Learning





Learning Frameworks

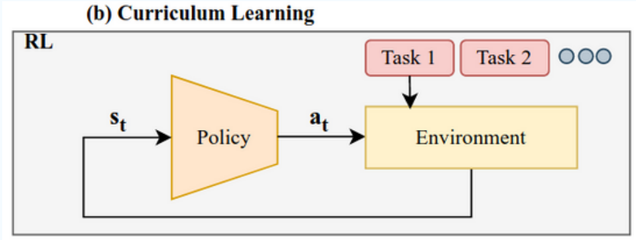
1. End-to-end Learning



- **MDP**를 단일 구조로 보고 **DRL** 알고리즘으로 처음부터 끝까지 해결.
- 주요 알고리즘:
  - **on-policy: TRPO, PPO** (보수적 업데이트로 안정적인 학습 제공).
  - **off-policy: DDPG, SAC** (샘플 효율성이 중요한 경우 적합).
- 문제점:
  - 초기 정책이 학습 신호를 탐색하지 못하면 **end-to-end** 학습 성과 저하.
  - **e.g)** 매우 어려운 환경에서 처음에 걷는 것 자체를 시도하는 것이 어려움
- 대안:
  - **Curriculum and hierarchical learning.**

Learning Frameworks

2. Curriculum Learning



- 점진적으로 난이도를 높이는 방식으로 더 어려운 문제 해결하도록 설계
  - 점진적 환경 난이도 증가: 가상 및 실제 에이전트 훈련으로 고난도 환경 처리(**Heess et al., 2017; Xie et al., 2020b 등**).
  - 정책 강건성 향상: 훈련 중 방해 요소와 무작위성 점진적 증가(**Akkaya et al., 2019**).
  - 제약 조건 만족 학습: 초기 완화된 조건에서 시작해 점차 강화(**Zhuang et al., 2023**).

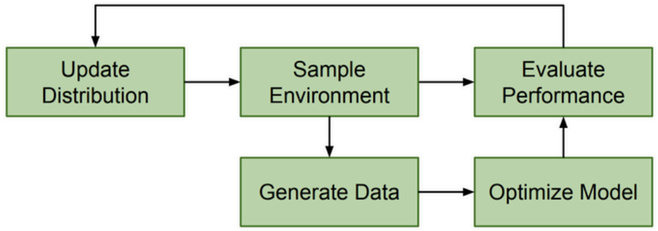


Figure 10: Overview of ADR. ADR controls the distribution over environments. We sample environments from this distribution and use it to generate training data, which is then used to optimize our model (either a policy or a vision state estimator). We further evaluate performance of our model on the current distribution and use this information to update the distribution over environments automatically.

$$(1-s) * l_{\text{easy}} + s * l_{\text{hard}}$$

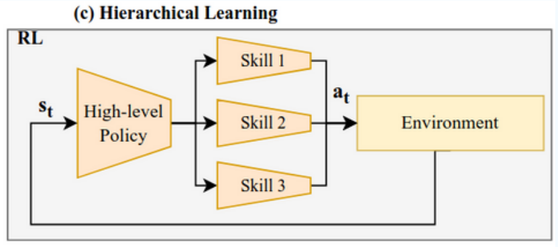
Skill	Obstacle Properties	Training Ranges ($[l_{\text{easy}}, l_{\text{hard}}]$)	Test Ranges ($[l_{\text{easy}}, l_{\text{hard}}]$)
Climb	obstacle height	[0.2, 0.45]	[0.25, 0.5]
Leap	gap length	[0.2, 0.8]	[0.3, 0.9]
Crawl	clearance	[0.32, 0.22]	[0.3, 0.2]
Tilt	path width	[0.32, 0.28]	[0.3, 0.26]

Table 1: Ranges for obstacle properties for each skill during training, measured in meters.

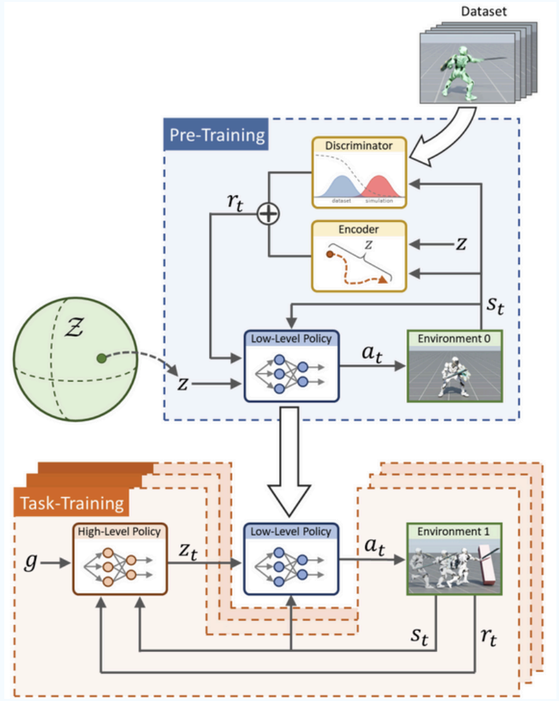
1. Akkaya, M. Andrychowicz, M. Chociej, M. Litwin, B. McGrew, A. Petron, A. Paino, M. Plappert, G. Powell, R. Ribas, et al. Solving rubik's cube with a robot hand. arXiv preprint arXiv:1910.07113, 2019.
2. Z. Zhuang, Z. Fu, J. Wang, C. Atkeson, S. Schwertfeger, C. Finn, and H. Zhao. Robot parkour learning. arXiv preprint arXiv:2309.05665, 2023.

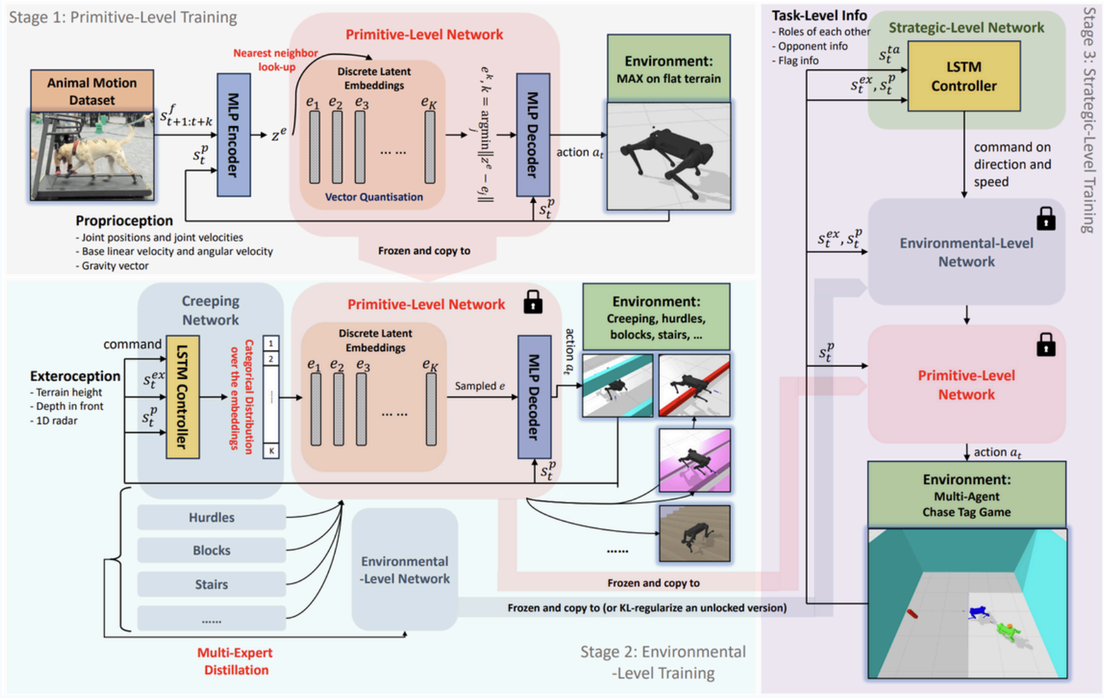
Learning Frameworks

3. Hierarchical Learning



- 매우 어려운 문제의 경우, 쪼개서 학습할 수 있음
  - **e.g. long horizons like navigation:** 상위 기술과 하위 기술로 나뉘며 하위 기술은 상위 기술을 입력으로 받음





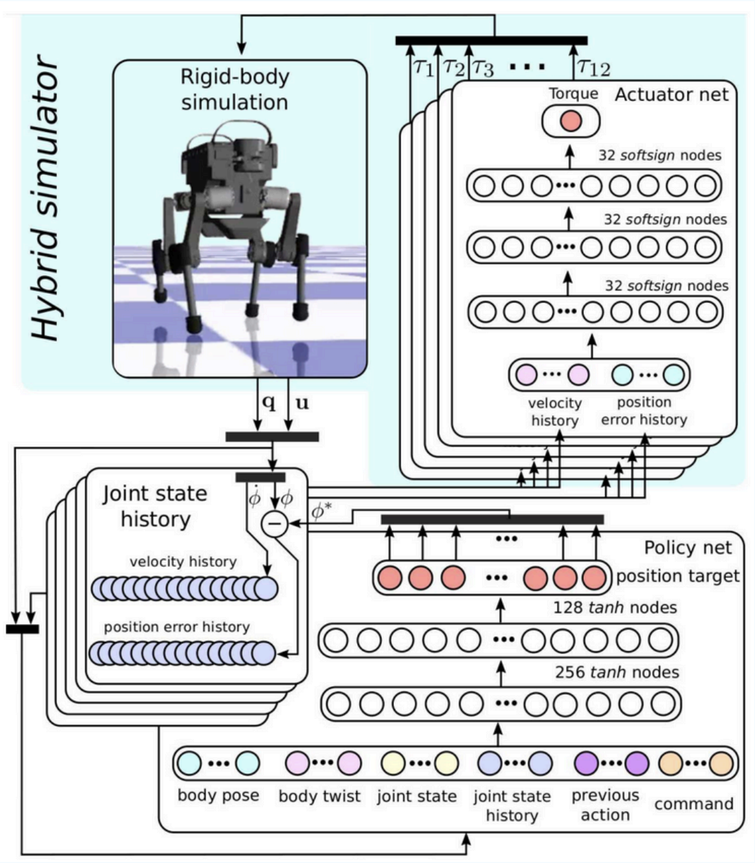
- VQ-VAE를 사용하여 동물 모션 캡처 데이터를 재현하는 저수준 정책 학습.
- 고수준 정책이 VQ-VAE의 잠재 공간을 조작하여 내비게이션 및 추격 게임 작업 수행.

1. X. B. Peng, Y. Guo, L. Halper, S. Levine, and S. Fidler. Ase: Largescale reusable adversarial skill embeddings for physically simulated characters. ACM Transactions On Graphics (TOG), 41(4):1–17, 2022.
2. L. Han, Q. Zhu, J. Sheng, C. Zhang, T. Li, Y. Zhang, H. Zhang, Y. Liu, C. Zhou, R. Zhao, et al. Lifelike agility and play on quadrupedal robots using reinforcement learning and generative pre-trained models. arXiv preprint arXiv:2308.15143, 2023.

Learning Frameworks

4. Privileged Learning

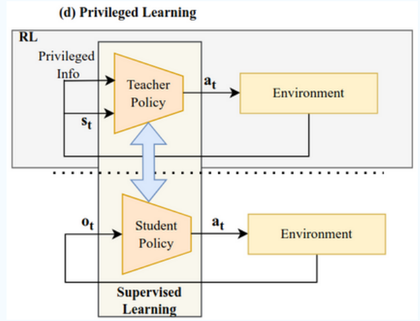
- 실제 세계의 로봇 작업은 **partially observable**하며, 제어기를 학습하는데 어려움을 초래함
  - **payload weight, friction coefficient etc...**
- 해결을 위해 **history** 정보를 활용함
  - **stacking a sequence of previous observations**
  - **TCN or RNN network**
  - 큰 입력 공간을 가진 복잡한 신경망 정책을 학습하는 것은 시간 소모가 크고 어려움
- **Network** 공간 효율성을 위해, **privileged learning** 기법이 나옴



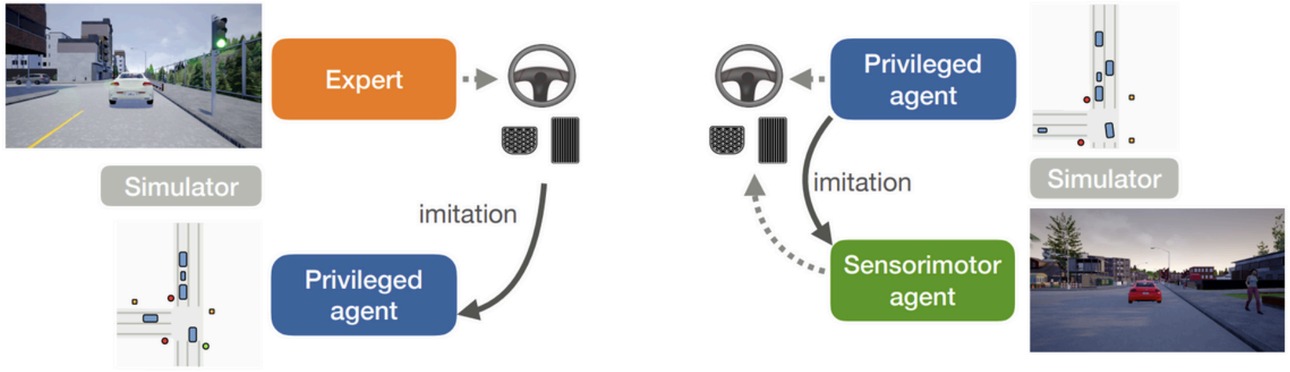
1. J. Hwangbo, J. Lee, A. Dosovitskiy, D. Bellicoso, V. Tsounis, V. Koltun, and M. Hutter. Learning agile and dynamic motor skills for legged robots. Science Robotics, 4(26):eaau5872, 2019.

Learning Frameworks

4. Privileged Learning

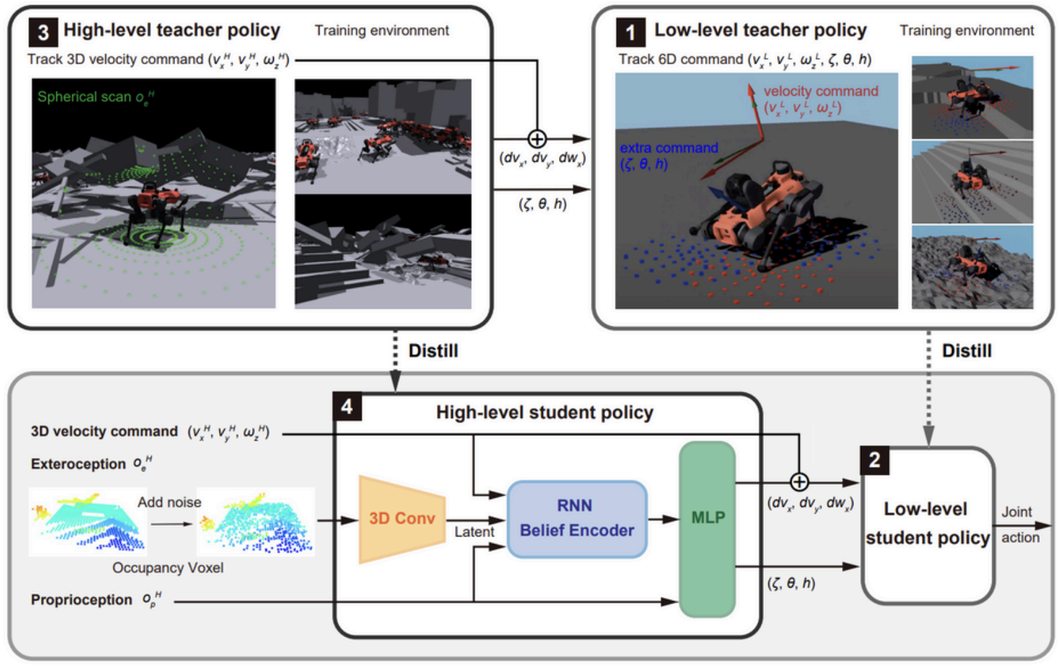


- **learning by cheating -> privileged Learning**
  - 완전히 관찰 가능한 **MDP**를 활용하여 **Teacher policy**을 학습시키고, 이 정책을 **RNN**이나 **TCN** 같은 시퀀스 모델에 기반한 **student policy**에 전달(**distill**)함



(a) Privileged agent imitates the expert (b) Sensorimotor agent imitates the privileged agent

Figure 1: Overview of our approach. **(a)** An agent with access to privileged information learns to imitate expert demonstrations. This agent learns a robust policy by cheating. It does not need to learn to see because it gets direct access to the environment's state. **(b)** A sensorimotor agent without access to privileged information then learns to imitate the privileged agent. The privileged agent is a "white box" and can provide high-capacity on-policy supervision. The resulting sensorimotor agent does not cheat.



1. D. Chen, B. Zhou, V. Koltun, and P. Kra'henbu'hl. Learning by cheating. In Conference on Robot Learning, pages 66–75. PMLR, 2020.
2. T. Miki, J. Lee, L. Wellhausen, and M. Hutter. Learning to walk in confined spaces using 3d representation. arXiv preprint arXiv:2403.00187, 2024.

Sim-to-Real Transfer

1. Good System Design

2. System Identification

3. Domain Randomization

4. Domain Adaptation

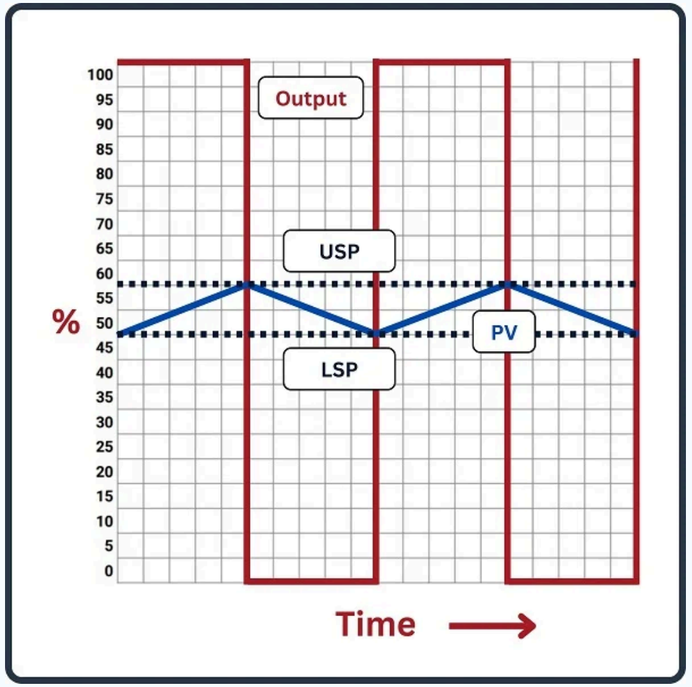
Sim-to-Real Transfer

- 시뮬레이션은 대규모의 데이터셋을 만들어내고, 학습할 수 있다는 장점이 있음.
- 하지만, 아래의 이유로 학습된 에이전트가 현실세계에서 제대로 작동하지 않는 경우가 많음.
  - 모터 가열로 인한 비일관된 동작
  - 하드웨어 통신 지연
  - 로봇 발의 부드러운 소재와 강체 모델 간 차이
- **sim-to-real** 격차를 극복하기 위한 주요 기법 탐색

Sim-to-Real Transfer

1. Good System Design

- 그냥 로봇이 움직이도록 학습하면 **bang-bang control** 이 발생하기도 함
  - 이는 고진동 관절 제어를 만들며, 실제 하드웨어에서는 적합하지 않음
- 이러한 현상을 예방하는 방법은 아래와 같음
  - **Reward Design**
  - **Observation and Action Space Design**
  - **Domain Knowledge**



on-off control (bang-bang control)

Sim-to-Real Transfer

1. Good System Design

- **Reward Design**
  - 보상 함수 설계는 **sim-to-real** 성능에 직접적인 영향을 미침.
  - 예방 조치:
    - **acceleration of the joints penalty:** 불안정한(**jittery**) 동작 방지. $R_{\text{acc}} = -\alpha \sum_i \ddot{q}_i^2$
    - **foot air time:** 발 끌림(**foot-dragging**) 방지. $R_{\text{air}} = \beta \sum_j t_{\text{air},j}$
    - **foot impact penalty:** 강하게 발을 구르는 행동 방지(**Makoviychuk et al., 2021**). $R_{\text{impact}} = -\gamma \sum_j F_{\text{impact},j}^2$
  - **Velocity tracking**과 같은 작업 보상과 정규화 보상 간 균형을 유지하는 것은 매우 까다로운 작업임.

1. V. Makoviychuk, L. Wawrzyniak, Y. Guo, M. Lu, K. Storey, M. Macklin, D. Hoeller, N. Rudin, A. Allshire, A. Handa, et al. Isaac gym: High performance gpu-based physics simulation for robot learning. arXiv preprint arXiv:2108.10470, 2021.

Sim-to-Real Transfer

1. Good System Design

- **Observation and Action Space Design**
  - 정교하게 선정한 **obs** 및 **action**은 **sim-to-real** 전이에 중요한 역할을 함
  - **Xie et al.(2021):**
    - **Laikago** 로봇의 실험에서, **randomization or adaptation** 기술 없이도 **sim-to-real** 전이를 가능하게 하는 주요 요소들을 확인함.
    - 관절 **PD** 제어기의 **proportional gain**을 낮게 설정하여 **compliant** 동작을 유도함으로써 **sim-to-real** 격차를 크게 줄임.
    - 관측에 몸체 속도를 포함하는 **state estimator**를 사용하여 속도 교란을 억제.

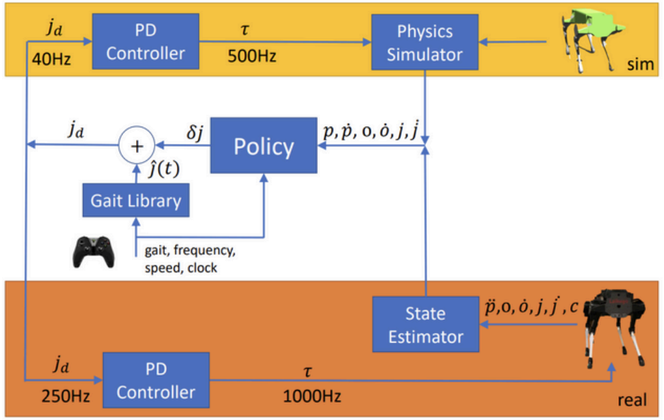


Fig. 2: Overview of our system. The input to the policy includes robot state and user commands. The output is a residual PD target, which is added to a reference target and applied to a joint PD controller. Various motions are achieved by using a library of gaits as reference trajectories.

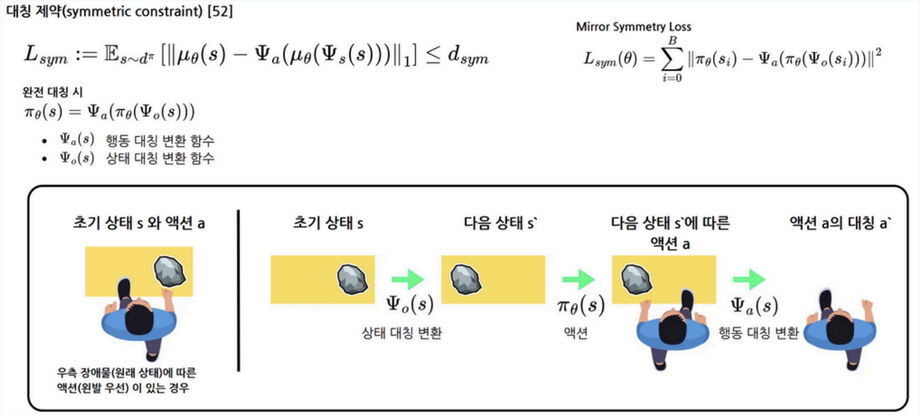
- **a) Observation:**
  - **Previous method:** 원시 센서 데이터(몸체 방향, 관절 각도)만 사용.
  - **Our method:** 상태 추정기와 추가 정보(몸체 속도, 관절 속도) 포함.
  - **Analyze:** 몸체 속도의 정확한 추정이 측면 방향 드리프트와 불안정성을 줄이는 데 중요.
- **b) Proportaion gain:**
  - **Previous method:** 높은 비례 이득(**kp=220k_p = 220kp=220**)으로 강한 제어.
  - **Our method:** 낮은 비례 이득(**kp=40k_p = 40kp=40**)으로 유연한 제어.

1. Z. Xie, X. Da, M. Van de Panne, B. Babich, and A. Garg. Dynamics randomization revisited: A case study for quadrupedal locomotion. In 2021 IEEE International Conference on Robotics and Automation (ICRA), pages 4955–4961. IEEE, 2021.

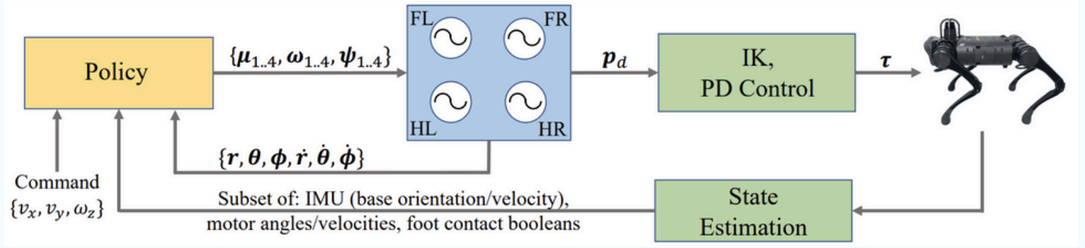
Sim-to-Real Transfer

1. Good System Design

- **Domain Knowledge**
  - **Learning locomotion skills for cassie: Iterative design and sim-to-real:**
    - 양족 보행 정책의 좌우 대칭성을 촉진하기 위해 대칭 제약(**symmetry constraints**)을 사용하여 동작 품질과 **sim-to-real** 성능 모두를 크게 향상.
  - 보상 함수에서 도메인 지식 사용:
    - 모션 캡처 데이터를 보상에 사용하여 특정 동작 스타일을 유도(**Peng et al., 2020; Han et al., 2023**).
    - 정책 구조에 중심 패턴 생성기(CPG)를 활용하여 더 나은 보행 패턴을 생성(**Lee et al., 2020; Bellegarda & Ijspeert, 2022; Iscen et al., 2018**).



제희님 자료 참고



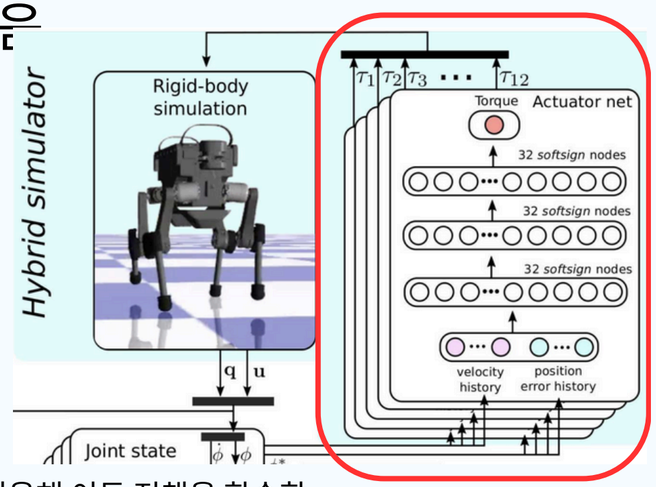
CPG 구조

1. Z. Xie, P. Clary, J. Dao, P. Morais, J. Hurst, and M. Panne. Learning locomotion skills for cassie: Iterative design and sim-to-real. In Conference on Robot Learning, pages 317–329. PMLR, 2020a.
2. https://github.com/TrustRL/study_archive/blob/main/reviews/%5B1004%5DNot%20only%20Reward%20but%20Also%20Constraint_2.pdf

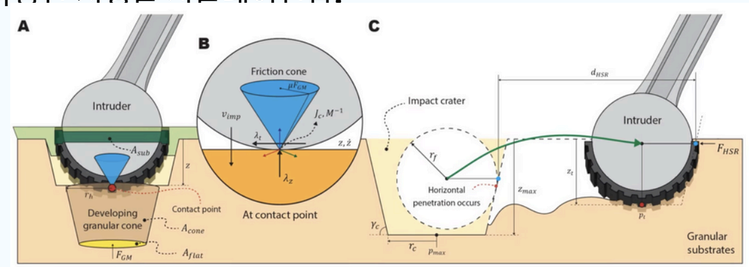
Sim-to-Real Transfer

2. System Identification

- 부정확한 모델링 또는 **unmodeled dynamics**은 **sim-to-real** 전이할 수 있음



  - 모델링 오류의 주요 원인:
    - <u>액추에이터 동역학:</u>
      - 시뮬레이션에서는 정책이 명령한 임의의 토크 프로파일을 모터가 적용할 수 있음.
      - 반면, 실제 액추에이터는 제한된 대역폭과 모터 컨트롤러의 추적 정확도 부족으로 인해 덜 정확한 토크 프로파일을 생성.
      - 사례:
        - 초기 성공 사례: **Tan et al.(2018)**은 **Miniature quadruped**에서 분석적 액추에이터 모델을 사용.
        - 이후 **Hwangbo et al.(2019)**은 블랙박스 모델(신경망)을 사용하여 액추에이터 모델을 학습하고, 이를 시뮬레이션에 적용해 이동 정책을 학습함.
    - <u>접촉 모델 불일치:</u>
      - 대부분의 시뮬레이터는 강체 접촉(**rigid contact**)을 사용하지만, 실제 세계의 상호작용에는 항상 약간의 변형이 존재.
      - 변형 가능한 지형이나 복잡한 형상을 시뮬레이션하는 것은 매우 어렵고, 종종 고비용의 유한요소법(**finite-element methods**)을 필요로 함
      - 사례:
        - **Choi et al.(2023)**은 변형 가능한 지형 모델을 개발하여, 매우 부드러운 해변 모래부터 단단한 아스팔트까지 다양한 지형을 시뮬레이션함.
        - 이 모델로 학습하면 강체 지형 모델로 학습한 경우보다 유사한 실제 지형에서 정책의 성능이 향상됨.



1. 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, 4(26):eaau5872, 2019.
2. https://www.youtube.com/watch?v=dt1u8zwUMok

Sim-to-Real Transfer

3. Domain Randomization

- 학습 중 시스템 매개변수를 무작위화하여 정책의 강건성 및 일반화 능력 향상.
  - 초기에는 로봇 질량, 마찰 계수, 모터 강도, 지연시간 등을 무작위로 변경하며 학습함
- 최근에는 **visual perception**도 **ramdomization**함.
  - **camera intrinsics and extrinsics, height scan etc...**

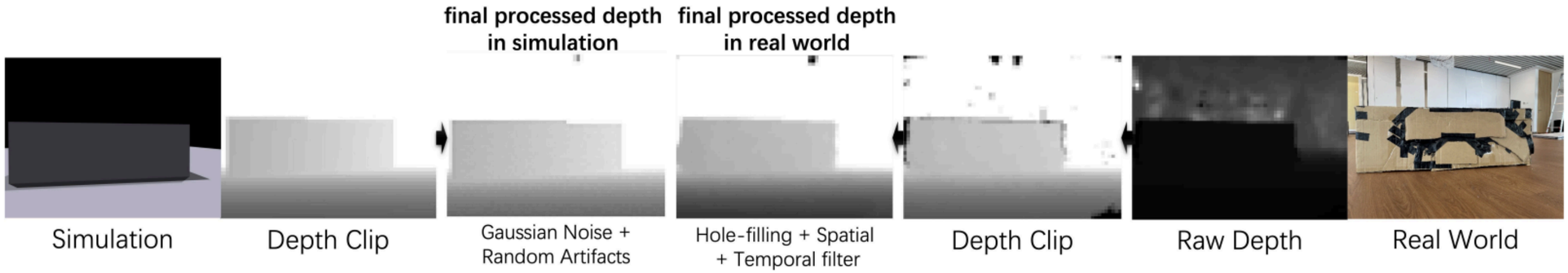


Figure 5: We bridge the visual gap between simulation and real world by applying pre-processing techniques. We use depth clipping, Gaussian noise and random artifacts in simulation, and depth clipping and hole-filling, spatial and temporal filters in the real world.

1. Z. Zhuang, Z. Fu, J. Wang, C. Atkeson, S. Schwertfeger, C. Finn, and H. Zhao. Robot parkour learning. arXiv preprint arXiv:2309.05665, 2023.

Sim-to-Real Transfer

4. Domain Adaptation

- 환경을 식별하고 정책을 최적화하여 실제 환경에서 더 나은 성능과 범위 확장을 가능케 함
  - 이는 잘만 이루어지면, **Domain randomization**보다 좋은 성능을 보임
- **Explicitly identification: Control policy or during policy training중 input**으로 넣음
  - 환경 파라미터를 명시적으로 사용하며, 직관적이지만 많은 파라미터를 다루기 어려움.

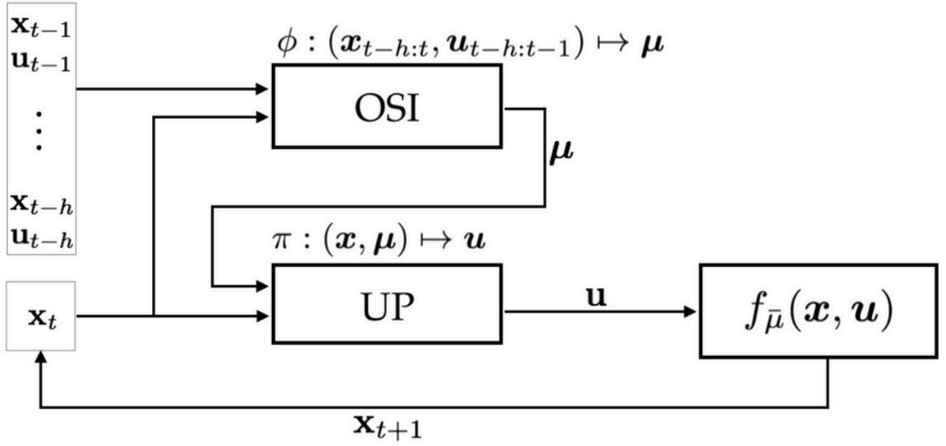


Fig. 1. Overview of UP-OSI. The online system identification model (OSI) takes as input the recent history of the motion and identify the model parameters $\boldsymbol{\mu}$. The universal control policy (UP) then takes the predicted model parameters along with the current state $\mathbf{x}$ to compute the optimal control $\mathbf{u}$.

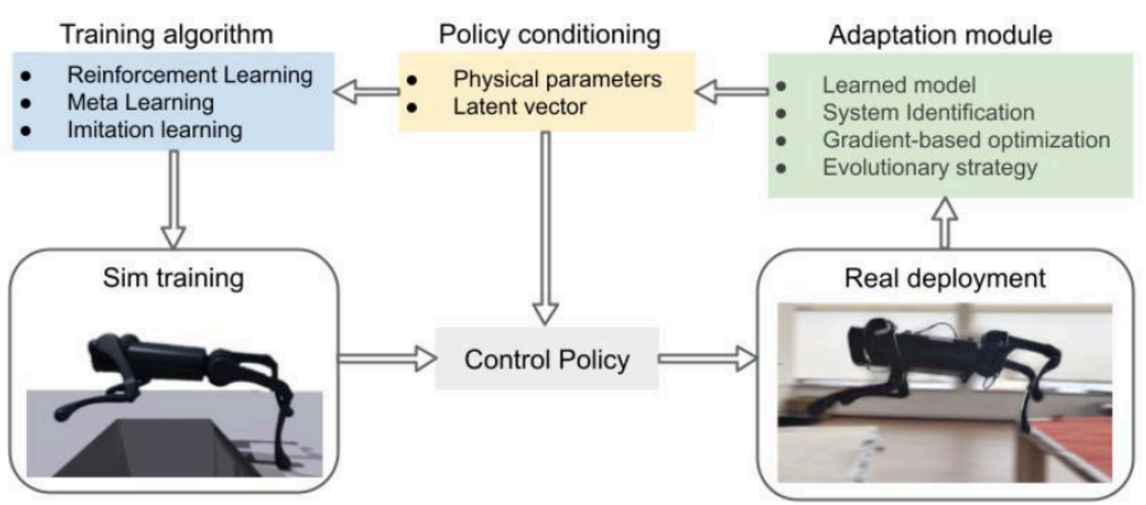


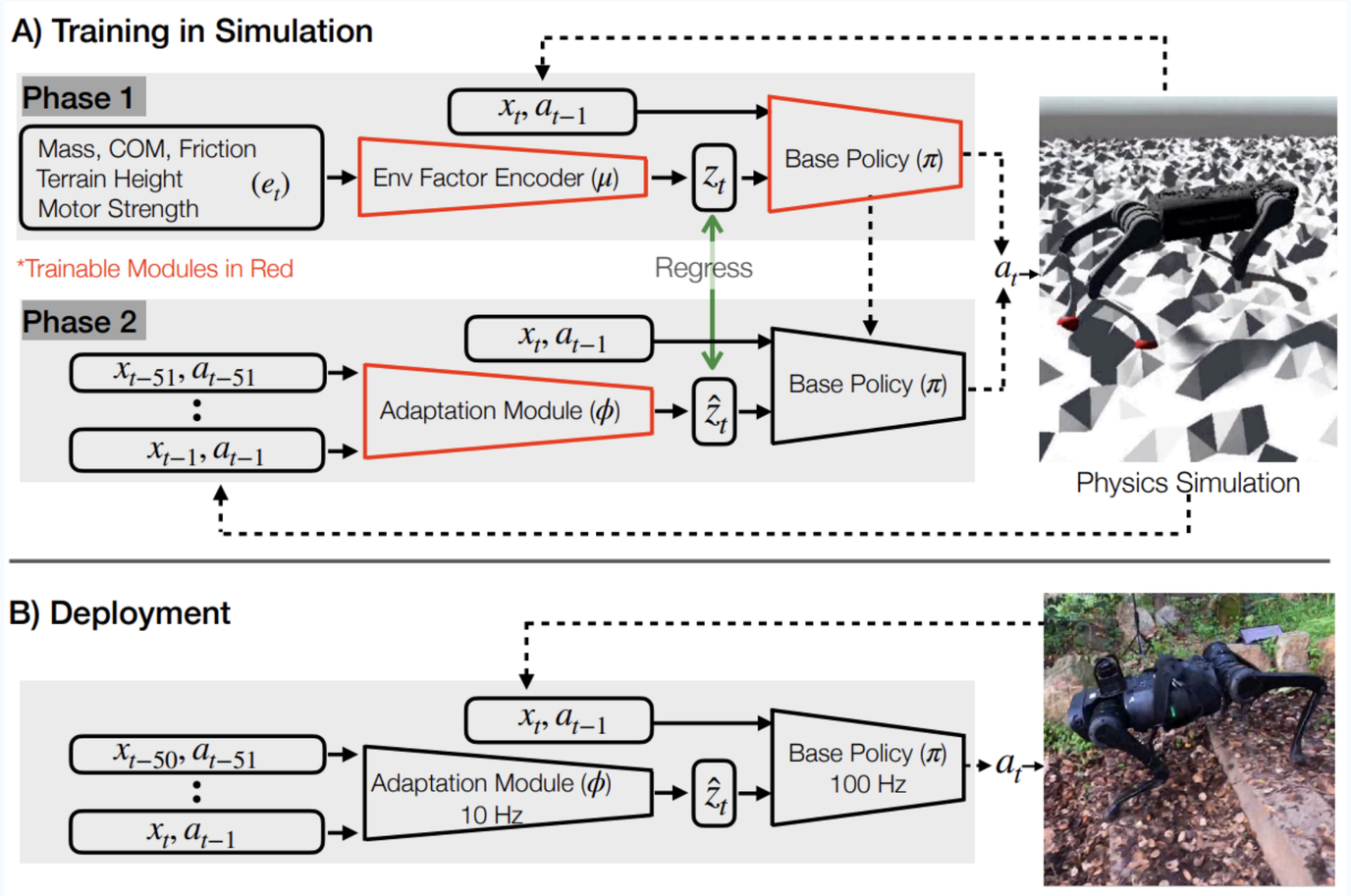
Figure 6. Common components in domain adaptation algorithms.

1. W. Yu, J. Tan, C. K. Liu, and G. Turk. Preparing for the unknown: Learning a universal policy with online system identification. In Proceedings of Robotics: Science and Systems, Cambridge, Massachusetts, July 2017. doi: 10.15607/RSS.2017.XIII.048.

Sim-to-Real Transfer

4. Domain Adaptation

- **Implicit identification:** 고차원 환경 정보를 **latent representation**로 압축하여 확장성 확보



1. A. Kumar, Z. Fu, D. Pathak, and J. Malik. Rma: Rapid motor adaptation for legged robots. arXiv preprint arXiv:2107.04034, 2021.

감사합니다