Learning Diverse Quadruped Locomotion Gaits via Reward Machines

Anonymous Author(s)
Affiliation
Address
email

Abstract: Learning diverse locomotion gaits for legged robots is important in order to efficiently and robustly move in different environments. Learning a specified gait frequently requires a reward function that accurately describes the gait. Our objective is to develop a simple mechanism for specifying the gaits at a high level (e.g. alternate between moving front feet and back feet), without providing labor-intensive motion priors such as reference trajectories. In this work, we leverage a recently developed framework called Reward Machine (RM) for high-level gait specification using Linear Temporal Logic (LTL) formulas over foot contacts. Our RM-based approach, called **R**eward **M**achine based **L**ocomotion **L**earning (RMLL), facilitates the learning of specified locomotion gaits, while providing a mechanism to dynamically adjust gait frequency. This is accomplished without the use of motion priors. Experimental results in simulation indicates that leveraging RM in learning specified gaits is more sample-efficient than baselines which do not utilize RM. We also demonstrate these learned policies with a real quadruped robot.

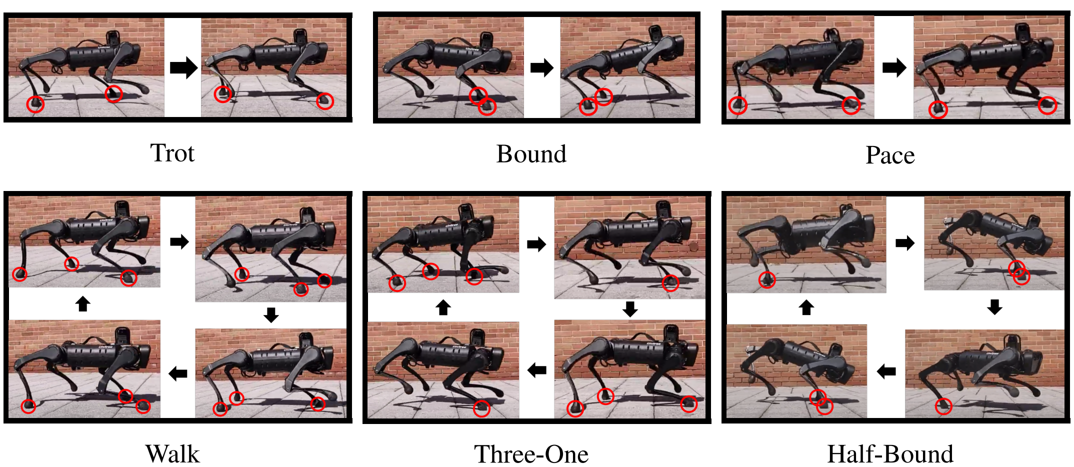

Figure 1: Snapshots of important poses of each of the six gaits learned with six different RMs. Specifying and learning the gaits (except for Half-Bound) require defining no more than eight logical rules. Red circles are around feet making contact with the ground.

1 Introduction

Legged animals are capable of performing a variety of locomotion gaits, in order to move efficiently and robustly at different speeds and environments [1, 2]. The same can be said of legged robots, where different locomotion gaits have been shown to minimize energy consumption at different speeds and environments [3, 4, 5]. Still, leveraging the full diversity of possible locomotion gaits has not been thoroughly explored. As legged robots can better perform a larger variety of gaits, new

Submitted to the 7th Conference on Robot Learning (CoRL 2023). Do not distribute.

possibilities in traversability and customized behaviors become possible. Unfortunately, learning *specific* quadruped locomotion gaits is a challenging problem. To accomplish this, it is necessary to design a reward function which can express the desired behavior. Commonly used reward functions for quadruped locomotion encourage maximizing velocity command tracking, while minimizing energy consumption [6, 7]. While training over these types of reward functions oftentimes yields high quality locomotion policies, they do not specify any particular gait.

In order to incentivize the agent to learn a specific gait, the reward function must be encoded with such gait-specific knowledge. It is possible to design a naive reward function which explicitly encourages specific sequences of milestone foot contacts, which we refer to as poses. Unfortunately, this breaks the Markov property, because historical knowledge of previous poses within the gait is necessary to know which pose should be reached next in order to adhere to the specified gait. Quadruped locomotion controllers are commonly run at 50 Hz or more [8, 9], which generates a long history of states between each pose of a gait. Thus, naively satisfying the Markov property would require including all of these historical states in the state space, and would make the learning process more challenging as the policy would need to figure out which portion of this history is relevant.

Some researchers have taken advantage of motion priors in order to encode gait-specific knowledge in a reward function. One popular method for encoding such knowledge in a reward function is to maximize the similarity between the robot's motion and a reference trajectory [10, 11, 12]. While this approach has been successfully demonstrated on real robots, it requires significant manual effort to obtain reference trajectories, and constrains the robot's motion to the given trajectory.

In this paper, we alleviate the above mentioned problem of gait specification by leveraging Reward Machines (RMs) [13], which specify reward functions through deterministic finite automatons. The RM transition function is defined through LTL formulas over propositional symbols, which in our case specify foot contacts. Thus, changing the automaton state corresponds to reaching the next pose within the gait. The reward function is Markovian when considering the low-level state (robot sensor information), along with the current automaton state, because the automaton state encodes the relevant gait-level information needed to determine the next pose. This approach enables us to easily specify and learn diverse gaits via logical rules, without the use of motion priors.

We refer to our approach as RM-based Locomotion Learning (RMLL), and train policies for six different gaits in simulation without the use of reference trajectories. Each policy is trained over a range of gait frequencies, which we can dynamically adjust during deployment. The reward function of each gait is easily defined through an automaton over desired foot contacts. We conduct an ablation study to evaluate the sample efficiency of RMLL in training the six different gaits, and deploy all gaits on a real Unitree A1 quadruped robot (see Figure 1). We compare RMLL to three baselines, each of which is designed to evaluate whether knowledge of the automaton state during training is actually beneficial in terms of sample efficiency. Results show that RMLL improves sample efficiency over its ablations for all gaits, which is more substantial for more complex gaits.

2 Related Work

In this section, we discuss prior work on RMs, and legged locomotion via Reinforcement Learning (RL). We then focus on existing methods of gait specification and learning for legged locomotion, with and without motion priors.

Reward Machine Since the introduction of Reward Machines (**RMs**) [14], there have been various new research directions such as learning the RM structure [15, 16, 17], RM for partially observable environments [18], probabilistic RMs [19], RM for lifelong RL [20], and RM for multi-agent settings [21] to name a few. While these works primarily focused on RM algorithmic improvements and theoretical analysis, their applications did not go beyond toy domains. RMs have also been used for simulated robotic arm pick-and-place tasks, which learn RM structures from demonstrations [22]. However, their approach was not implemented or evaluated in real-world robotic continuous control problems with high-dimensional action spaces. We use RM for robot locomotion learning in this work.

RL-based Locomotion Learning There are numerous works on applications of RL for robot locomotion [23, 7, 24, 25, 26, 27, 8, 11, 28, 29, 9, 30]. Approaches of this type often lead to robust locomotion gaits, some of which can transfer to real robots. However, these approaches generally focus on learning robust locomotion policies, and do not support the specification of particular gaits. Exceptions that support RL-based locomotion learning of specific gaits are described next.

Diverse Locomotion Gaits Various works have demonstrated diverse locomotion gaits for quadruped robots. MPC based approaches have demonstrated such gait diversity [31], however these methods require accurate dynamics models, and significant manual tuning. Different gaits can naturally emerge through minimizing energy [3], or selected from a high-level policy which selects foot contact configurations or contact schedules [4, 5]. While this enables gait transitions for efficient locomotion in different environments, it does not provide the ability to learn any *arbitrary* gait or gait frequency specified beforehand. Other works provide such ability to specify quadruped locomotion gaits. Some methods do this through motion priors such as trajectory generators [32] or motion references [10]. Obtaining these priors require extensive human (and sometimes even animal) effort, and restricts the robot to following the specified trajectory with little variation. While motion references can be generated, it requires highly tuned foot trajectory polynomials and phase generation functions [33]. Our approach does not require such motion priors and can easily specify different gaits via a few logical rules. Our policies also have freedom to explore variations of the specified gait on its own and is not restricted by a predefined trajectory.

Learning without Motion Priors In work more similar to ours, a single quadruped locomotion policy which can perform various gaits is trained and demonstrated without the use of motion priors [34]. While useful to adapt to different environments, this approach can only learn simple two-beat gaits, and is unable to learn any arbitrary gait specified from desired foot contact sequences. Another work similar to ours enabled learning diverse gaits for a bipedal robot without requiring motion priors [35]. These gaits were trained over a reward function which specifies swing and stance phases and timings per leg. To ensure a Markovian reward, they added cycle time offsets and phase ratio vectors per each leg to the state. By comparison, RMLL (ours) does not need explicit leg-specific timing information. Instead, RMLL leverages an abstract representation of the current pose within the gait (i.e., the RM state) to facilitate the learning of diverse gaits.

3 RM-based Locomotion Learning

We present our RM-based reinforcement learning approach for learning quadruped locomotion policies below. Figure 2 presents an overview of how we use RMs to specify a diverse set of quadruped locomotion gaits and facilitate efficient policy learning.

3.1 Reward Machines: Concepts and Terminologies

Reward Machines are typically used in settings where we have a set of "milestone" sub-goals to achieve in order to complete some larger task. Reward functions which do not encode these subgoals are oftentimes too sparse, while reward functions which explicitly reward sub-goal completion can be non-Markovian. An RM allows for specification of these sub-goals through an automaton, which can be leveraged to construct an MDP. Thus, through an RM, the reward function can give positive feedback for completing sub-goals, while also defining an MDP with a Markovian reward function.

Formally, an RM is defined as the tuple $(U, u_0, F, \delta_u, \delta_r)$ [14], where U is the set of automaton states, u_0 is the start state, F is the set of accepting states, $\delta_u : U \times 2^{\mathbf{P}} \to U \cup F$ is the automaton transition function, while $\delta_r : U \times 2^{\mathbf{P}} \to [S \times A \times S \to \mathbb{R}]$ is the reward function associated with each automaton transition. This RM definition assumes the existence of set $\mathbf{P}$, which contains propositional symbols that refer to high-level events from the environment that the agent can detect.

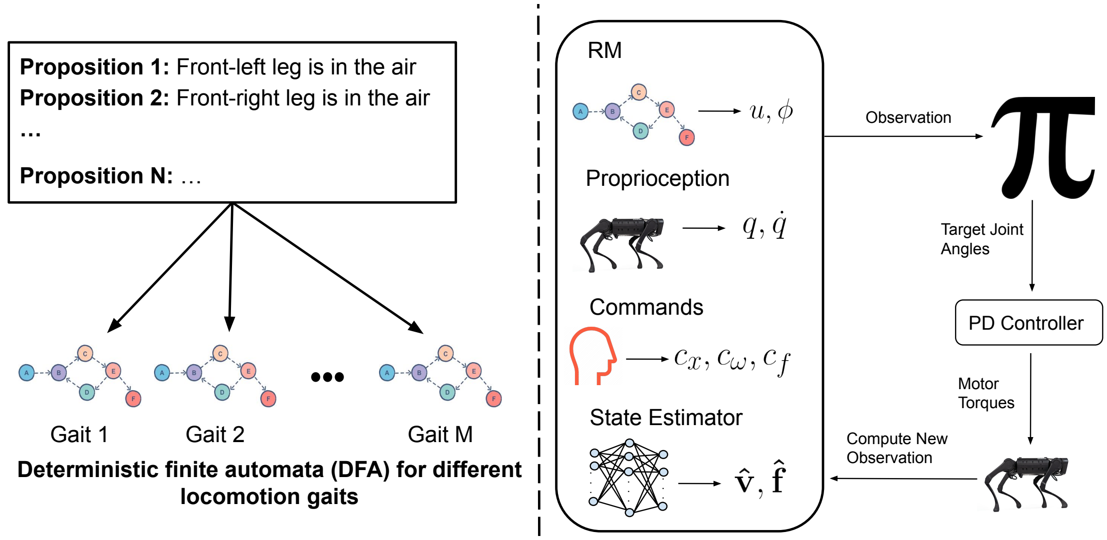

Figure 2: Overview of RM-based Locomotion Learning (RMLL). We consider propositional statements specifying foot contacts. We then construct an automaton via LTL formulas over propositional statements for each locomotion gait (left side). To train gait-specific locomotion policies, we use observations which contain information from the RM, proprioception, velocity and gait frequency commands, and variables from a state estimator (right side).

For each environment step the agent takes, the agent evaluates which automaton state transition to take via δ_u, and receives reward via δ_r.

Reward machines are defined alongside state space S, which describe the low-level observations the agent receives after each step in the environment. In order to construct an MDP from the non-Markovian reward defined by the RM, the agent considers its own observations from S, along with its current RM state from U. Training over state space $S \times U$ no longer violates the Markov property, because knowledge of the current RM state indicates which sub-goal was previously completed. The inclusion of this subsection is simply for the completeness of this paper. More details are available in the RM article [13].

3.2 RM for Quadruped Locomotion

We use RMs to specify the sequence of foot contacts expected of the gait. In our domain, we consider $\mathbf{P} = \{P_{FL}, P_{FR}, P_{BL}, P_{BR}\}$, where $p \in \mathbf{P}$ is a Boolean variable. These indicate whether the front-left (FL), front-right (FR), back-left (BL), and back-right (BR) feet are making contact with the ground. Automaton states in U correspond to different poses in the gait, where u_0 corresponds to the last pose. Meanwhile, δ_u changes the automaton state when the next pose in the gait is reached. We define δ_r as:

$$\delta_r(u_t, a) = \begin{cases} R_{\textbf{walk}}(s) * b & \delta_u(u_t, a) \neq u_t \\ R_{\textbf{walk}}(s) & otherwise \end{cases}$$

where $R_{\textbf{walk}}$ encourages maximizing velocity command tracking while minimizing energy consumption [29], and is fully defined in Table 1. Reward function δ_r encourages taking RM transitions which correspond to the specified gait, because $R_{\textbf{walk}}$ is scaled by bonus b when such transitions occur. We leave F empty for all gaits, as quadruped locomotion is an infinite-horizon task.

We define our state space $S = (u, \phi, q, \dot{q}, a_{t-1}, c_x, c_\omega, c_f, \hat{\mathbf{v}}, \hat{\mathbf{f}})$, where u is the current RM state, ϕ is the number of time steps which occurred since the previous RM state changed, q and $\dot{q}$ are the 12 joint angles and joint velocities respectively, a_{t-1} is the previous action, c_x and c_ω are base

Term Description	Definition	Scale
Linear Velocity x	$exp(-\|\mathbf{c_x} - \mathbf{v_x}\|^2/0.25)$	$1.0dt$
Linear Velocity z	$\mathbf{v_z}^2$	$-2.0dt$
Angular Velocity x, y	$\|\omega_{\mathbf{x,y}}\|^2$	$-0.05dt$
Angular Velocity z	$exp(-(\mathbf{c}_\omega - \omega_\mathbf{z})^2/0.25)$	$0.5dt$
Joint Torques	$\|\tau\|^2$	$-0.0002dt$
Joint Accelerations	$\|(\dot{\mathbf{q}}_\mathbf{last} - \dot{\mathbf{q}})/dt\|^2$	$-2.5e-7dt$
Feet Air Time	$\sum_{f=1}^{4}(\mathbf{t_{air,f}} - 0.5)$	$1.0dt$
Action Rate	$\|\mathbf{a_{last}} - \mathbf{a}\|^2$	$-0.01dt$

Table 1: All terms of $R_{\textbf{walk}}$. $\mathbf{v}$ refers to base velocity, $\mathbf{c}$ refers to commanded linear and angular base velocity, ω refers to base angular velocity, τ refers to joint torques, $\dot{\mathbf{q}}$ refers to joint velocities, $\mathbf{t_{air}}$ refers to each foots air time, $\mathbf{a}$ refers to an action, and dt refers to the simulation time step.

linear and angular velocity commands respectively, c_f is the gait frequency command, and $\hat{\mathbf{v}}, \hat{\mathbf{f}}$ is estimated base velocity and foot heights. The RM state is encoded as a one-hot vector, making the dimensions of $S \in [49, 52]$ based on the number of RM states defining the gait.

Gait Frequency: Aside from gait specification, we also leverage RMs to specify gait frequency. Our definition of δ_r naturally encourages high frequency gaits, because maximizing the number of pose transitions maximizes total accumulated reward. Thus, we introduce gait frequency command c_f, which denotes the minimum number of environment steps which must be taken until the agent is allowed to transition to a new RM state. When the agent maximizes the number of RM transitions it takes, while being restricted by c_f, then the commanded gait frequency is followed. Adding c_f on its own would cause the reward function to be non-Markovian, because the agent needs to remember how many environment steps have occurred since the RM state last changed. Thus, we also add timing variable ϕ to our observations, which keeps track of how many environment steps have occurred since the RM state has changed last. At every environment time step, we compare ϕ with c_f, and do not allow an RM transition to take place if $\phi < c_f$. Adding c_f and ϕ enable gait frequency to be dynamically adjusted during policy deployment, and is demonstrated on hardware in our supplementary video.

Illustrative Gait: We now discuss specifying a well known quadruped locomotion gait [36], **Trot**, via RM. Figure 3 shows the RM associated with this gait. In this **Trot** automaton, we want to synchronize lifting the FL leg with the BR leg, and the FR leg with the BL leg. LTL formula $P_{FL} \wedge \neg P_{FR} \wedge \neg P_{BL} \wedge P_{BR}$ evaluates to true when only the FR and BL feet are in the air simultaneously, while $\neg P_{FL} \wedge P_{FR} \wedge P_{BL} \wedge \neg P_{BR}$ evaluates to true when only the FL and BR feet are in the air simultaneously. The two RM states correspond to which combination of feet were previously in the air. If the agent is in state q_1, then $P_{FL} \wedge \neg P_{FR} \wedge \neg P_{BL} \wedge P_{BR}$ must have been evaluated as true at some point earlier. Note that when the agent does not achieve the desired pose, then the agent takes a self-loop to remain in the current RM state.[1]

Remark It is an intuitive idea of training a gait-specific locomotion policy via RM, because along with low-level sensor information, the policy also has access to the current RM state, which is an abstract representation of the historical foot contacts relevant to the current pose in the gait. Rather than attempting to learn this from a long history of states, the RM state explicitly encodes the previously reached gait pose. Thus, the policy can learn different gaits in a sample-efficient manner, because at each time step it can reference the RM state to indicate which pose within the gait to reach next.

4 Experiments

We train six different locomotion gaits via RMLL in simulation, and perform an ablation study to evaluate whether knowledge of the RM state improves sample efficiency when compared to ablations which do not access the RM state during training. We demonstrate all learned gaits on a Unitree A1 robot.

[1]We provide the RMs for all other gaits we trained in Appendix A.

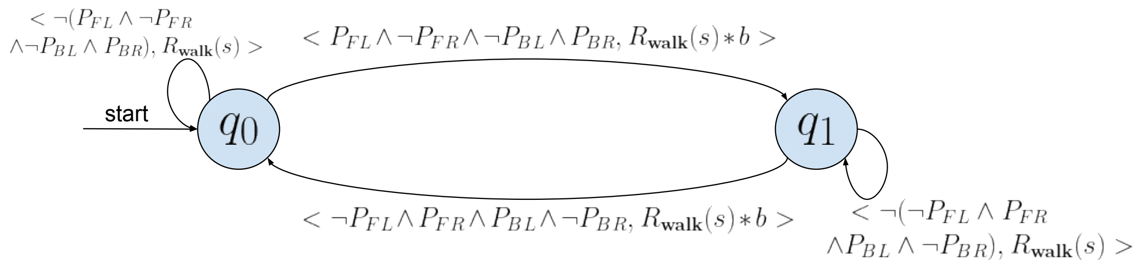

Figure 3: Reward Machine for **Trot** gait, where we want to synchronize lifting the FL leg with the BR leg, and the FR leg with the BL leg. **Trot** is one of the six gaits considered in this work.

4.1 Training Details

State, Action, Reward We estimate base velocity $\hat{\mathbf{v}}$ and foot heights $\hat{\mathbf{f}}$ concurrently with the policy, via supervised learning [37]. Note that during training we only consider a foot in the air if it is higher than 0.03 meters. Actions include the target joint positions of each joint. These are input to a PD controller which computes the joint torques. The PD controller has a proportional gain $K_p = 20$ and derivative gain $K_d = 0.5$. The policy is queried at 50 Hz, and control signals are sent at 200 Hz. We set bonus $b = 1000$ in δ_r for all gaits.

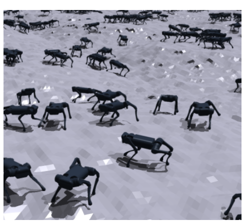

Figure 4: Isaac Gym simulation environment.

Environment Details We use the Isaac Gym [38] physics simulator and build upon a legged locomotion environment [29] to train our policies. We use a terrain called `random_uniform_terrain`, which is seen in Figure 4. The robot traverses more challenging versions of this terrain based on a curriculum which increases terrain difficulty after the robot learns to traverse flatter versions of the terrain. Each episode lasts for 20 seconds, and ends early if the robot makes contact with the ground with anything other than a foot, if joint angle limits are exceeded, or if the base height goes below 0.25 meters. After each training episode, we sample a new velocity and gait frequency command for the robot to track. To facilitate sim-to-real transfer, we perform domain randomization over surface frictions, add external pushes, and add noise to observations [29]. Additional details and code are available in the Appendix.

Model Training We train our policy via PPO [39], with actor and critic architectures as 3-layer multi-layer perceptrons (MLPs) with hidden layers of size 256. Each policy is trained for 100 million time steps, where parameters are updated every 100,000 time steps. Data is collected from 4096 agents running simultaneously.

4.2 Ablation Study

We run an ablation study to determine whether knowledge of the RM state actually improves sample efficiency. We design the following baselines which we compare RMLL against:

1. **No-RM**: Remove the RM state from the state space, keeping everything else the same.
2. **No-RM-Foot-Contacts**: Remove the RM state from the state space, and add foot contacts.
3. **No-RM-History**: Remove the RM state from the state space, and add foot contacts. Expand the state space to include states from the past 12 time steps.

Comparing against **No-RM** indicates whether the RM state is useful at all. Comparing against **No-RM-Foot-Contacts** indicates whether RM state is only useful because it contains information about foot contacts. Comparing against **No-RM-History** indicates whether the information provided

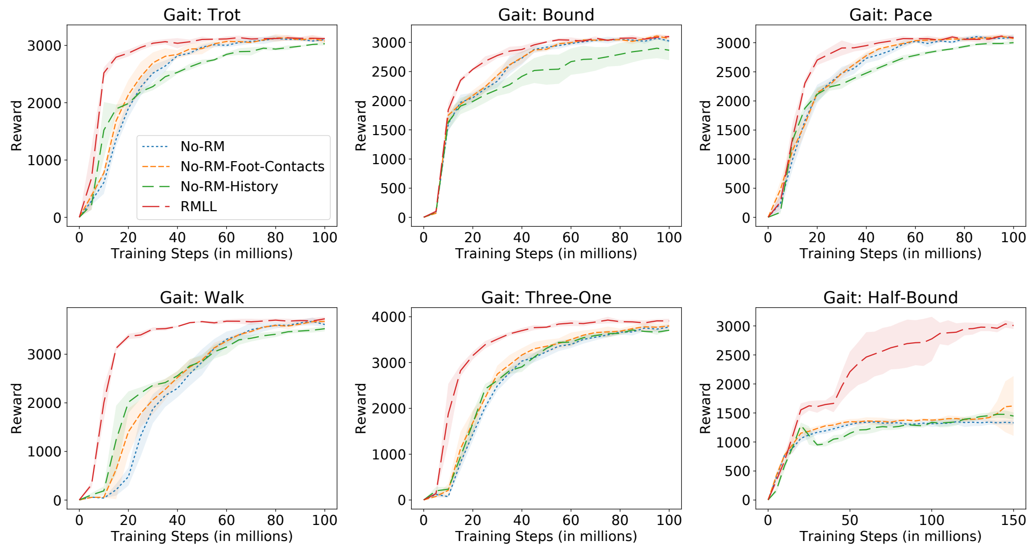

Figure 5: Reward curves for all gaits. RMLL more efficiently accumulates reward for each gait.

by the RM state can be easily learned when given sufficient history. Note that we do not compare to existing works which demonstrate diverse locomotion gaits, because we consider a different setting under different assumptions. We claim RMLL facilitates the learning of a larger diversity of gaits with less manual effort required, not that the final gaits are necessarily *better* than existing ones.

We experiment over six different locomotion gaits: **Trot**, **Pace**, **Bound**, **Walk**, **Three-One**, and **Half-Bound**. See Appendix A for the RMs defining each gait. For each approach (ablation or not), we trained over five different random seeds per gait. For each training run, we save the policy after every 5 million steps. We then deploy each of those saved policies for 100 episodes, and average the accumulated reward over the five runs per approach. We report the resulting reward curves in Figure 5, where the shaded region indicates the standard deviation of the total accumulated reward across the five training runs.

The results indicate that knowledge of the RM state improves sample efficiency for all gaits when compared with the ablations. We believe this is the case, because the RM state can efficiently inform the policy of gait-relevant historical foot contacts, whereas the ablations either do not have access to historical foot contacts, or must learn the relevant contacts from history. The results also show that **No-RM-History** does not perform better than the other ablations without history, indicating that it is challenging to learn gait-relevant information directly from 12 time steps of historical states. We also notice that **No-RM** performs similarly to **No-RM-Foot-Contacts**, which indicates **No-RM** learns to implicitly estimate foot contacts from the state. Finally, we notice a large performance gap between RMLL and all other ablations. We believe this is the case due to the additional complexity of this gait, which can be seen in Appendix A.

4.3 Qualitative Results

Foot Contacts In simulation, we deploy each gait with a linear velocity command of 0.75 meters/second (0.5 meters/second for **Walk**), while initializing c_f to its maximum training value, updating c_f to its minimum after 50 time steps, and again updating c_f to its maximum after 100 time steps. We record the foot contacts of each gait in Figure 6, which shows that each of our gaits follows the expected foot contact sequence and gait frequencies. For example, green and orange bars in **Trot** are synchronized, indicating BR/FL feet are coordinated. Also note the length of the bars decrease in the middle of the trial, corresponding to when c_f was decreased.

Hardware Demonstration We run our learned policies on a Unitree A1 robot, without any additional fine-tuning. Each trial is on a concrete walkway, where we increase and decrease gait

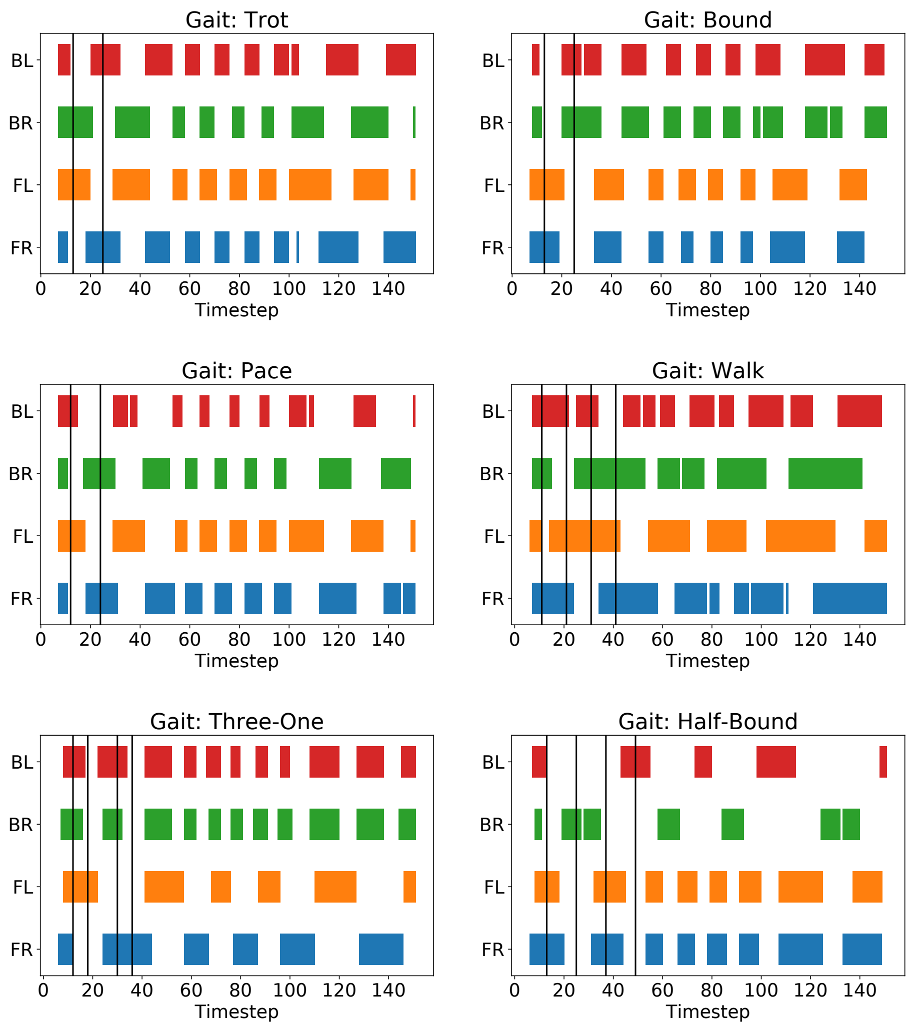

Figure 6: Foot contact plots for each gait. We report foot contacts from simulated trials running each gait, and display colored horizontal bars to indicate when the specified foot makes contact. We add vertical bars specifying RM transitions for the first cycle of unique poses for each gait. Note that in each trial, gait frequency starts low, increases in the middle, and decreases toward the end.

frequency throughout the trial. We find that RMLL policies from all gaits successfully transfer to hardware, and the intended foot contact sequence and gait frequency is realized. A video capturing each of these trials is included in Supplementary Materials.

5 Discussion

Limitations and Future Work While our approach can be used to easily specify and learn customized locomotion gaits, we have not studied how to optimally leverage these different gaits to efficiently traverse various terrains, nor have we studied how to smoothly transition between gaits. In future work, researchers can train a hierarchical policy which selects desired gaits, gait frequencies, and velocity commands at a high level, which can be used as input to a wide variety of pre-trained gaits, in order to traverse different environments more efficiently.

Conclusion We leverage reward machines to specify different quadruped locomotion gaits via simple logical rules. We efficiently train locomotion policies in simulation which learn these specified gaits over a range of gait frequencies, without the use of motion priors. We demonstrate these policies on hardware, and find that our robot can perform a variety of different gaits, while dynamically adjusting gait frequency.

References

[1] D. F. Hoyt and C. R. Taylor. Gait and the energetics of locomotion in horses. *Nature*, 292 (5820):239–240, 1981.

[2] Z. Afelt, J. Błaszczyk, and C. Dobrzecka. Speed control in animal locomotion: transitions between symmetrical and nonsymmetrical gaits in the dog. *Acta neurobiologiae experimentalis*, 43(4-5):235–250, 1983.

[3] Z. Fu, A. Kumar, J. Malik, and D. Pathak. Minimizing energy consumption leads to the emergence of gaits in legged robots. *arXiv preprint arXiv:2111.01674*, 2021.

[4] X. Da, Z. Xie, D. Hoeller, B. Boots, A. Anandkumar, Y. Zhu, B. Babich, and A. Garg. Learning a contact-adaptive controller for robust, efficient legged locomotion. In *Conference on Robot Learning*, pages 883–894. PMLR, 2021.

[5] Y. Yang, T. Zhang, E. Coumans, J. Tan, and B. Boots. Fast and efficient locomotion via learned gait transitions. In *Conference on Robot Learning*, pages 773–783. PMLR, 2022.

[6] J. Kober, J. A. Bagnell, and J. Peters. Reinforcement learning in robotics: A survey. *The International Journal of Robotics Research*, 32(11):1238–1274, 2013.

[7] 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. *arXiv preprint arXiv:1804.10332*, 2018.

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

[9] T. Miki, J. Lee, J. Hwangbo, L. Wellhausen, V. Koltun, and M. Hutter. Learning robust perceptive locomotion for quadrupedal robots in the wild. *Science Robotics*, 7(62):eabk2822, 2022.

[10] 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.

[11] L. Smith, J. C. Kew, X. B. Peng, S. Ha, J. Tan, and S. Levine. Legged robots that keep on learning: Fine-tuning locomotion policies in the real world. *arXiv preprint arXiv:2110.05457*, 2021.

[12] 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.

[13] R. T. Icarte, T. Q. Klassen, R. Valenzano, and S. A. McIlraith. Reward machines: Exploiting reward function structure in reinforcement learning. *Journal of Artificial Intelligence Research*, 73:173–208, 2022.

[14] R. T. Icarte, T. Klassen, R. Valenzano, and S. McIlraith. Using reward machines for high-level task specification and decomposition in reinforcement learning. In *International Conference on Machine Learning*, pages 2107–2116. PMLR, 2018.

[15] Z. Xu, I. Gavran, Y. Ahmad, R. Majumdar, D. Neider, U. Topcu, and B. Wu. Joint inference of reward machines and policies for reinforcement learning. In *Proceedings of the International Conference on Automated Planning and Scheduling*, volume 30, pages 590–598, 2020.

[16] D. Neider, J.-R. Gaglione, I. Gavran, U. Topcu, B. Wu, and Z. Xu. Advice-guided reinforcement learning in a non-markovian environment. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 9073–9080, 2021.

[17] J. Corazza, I. Gavran, and D. Neider. Reinforcement learning with stochastic reward machines. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pages 6429–6436, 2022.

[18] R. Toro Icarte, E. Waldie, T. Klassen, R. Valenzano, M. Castro, and S. McIlraith. Learning reward machines for partially observable reinforcement learning. *Advances in Neural Information Processing Systems*, 32:15523–15534, 2019.

[19] T. Dohmen, N. Topper, G. Atia, A. Beckus, A. Trivedi, and A. Velasquez. Inferring probabilistic reward machines from non-markovian reward signals for reinforcement learning. In *Proceedings of the International Conference on Automated Planning and Scheduling*, volume 32, pages 574–582, 2022.

[20] X. Zheng, C. Yu, and M. Zhang. Lifelong reinforcement learning with temporal logic formulas and reward machines. *Knowledge-Based Systems*, 257:109650, 2022.

[21] C. Neary, Z. Xu, B. Wu, and U. Topcu. Reward machines for cooperative multi-agent reinforcement learning. *arXiv preprint arXiv:2007.01962*, 2020.

[22] A. Camacho, J. Varley, A. Zeng, D. Jain, A. Iscen, and D. Kalashnikov. Reward machines for vision-based robotic manipulation. In *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pages 14284–14290. IEEE, 2021.

[23] N. Kohl and P. Stone. Policy gradient reinforcement learning for fast quadrupedal locomotion. In *IEEE International Conference on Robotics and Automation, 2004. Proceedings. ICRA'04. 2004*, volume 3, pages 2619–2624. IEEE, 2004.

[24] T. Haarnoja, S. Ha, A. Zhou, J. Tan, G. Tucker, and S. Levine. Learning to walk via deep reinforcement learning. *arXiv preprint arXiv:1812.11103*, 2018.

[25] R. Hafner, T. Hertweck, P. Klöppner, M. Bloesch, M. Neunert, M. Wulfmeier, S. Tunyasuvunakool, N. Heess, and M. Riedmiller. Towards general and autonomous learning of core skills: A case study in locomotion. *arXiv preprint arXiv:2008.12228*, 2020.

[26] J. Lee, J. Hwangbo, L. Wellhausen, V. Koltun, and M. Hutter. Learning quadrupedal locomotion over challenging terrain. *Science robotics*, 5(47), 2020.

[27] 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.

[28] S. Chen, B. Zhang, M. W. Mueller, A. Rai, and K. Sreenath. Learning torque control for quadrupedal locomotion. *arXiv preprint arXiv:2203.05194*, 2022.

[29] N. Rudin, D. Hoeller, P. Reist, and M. Hutter. Learning to walk in minutes using massively parallel deep reinforcement learning. In *Conference on Robot Learning*, pages 91–100. PMLR, 2022.

[30] 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.

[31] 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)*, pages 1–9. IEEE, 2018.

[32] 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.

[33] Y. Shao, Y. Jin, X. Liu, W. He, H. Wang, and W. Yang. Learning free gait transition for quadruped robots via phase-guided controller. *IEEE Robotics and Automation Letters*, 7(2): 1230–1237, 2021.

[34] G. B. Margolis and P. Agrawal. Walk these ways: Tuning robot control for generalization with multiplicity of behavior. In *Conference on Robot Learning*, pages 22–31. PMLR, 2023.

[35] J. Siekmann, Y. Godse, A. Fern, and J. Hurst. Sim-to-real learning of all common bipedal gaits via periodic reward composition. *arXiv preprint arXiv:2011.01387*, 2020.

[36] M. Hildebrand. Symmetrical gaits of horses: Gaits can be expressed numerically and analyzed graphically to reveal their nature and relationships. *Science*, 150(3697):701–708, 1965.

[37] 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, 2022.

[38] 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.

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

A Reward Machines for Other Gaits

In this section, we present the reward machines for the five gaits not already shown: **Bound**, **Pace**, **Walk**, **Three-One**, and **Half-Bound**.

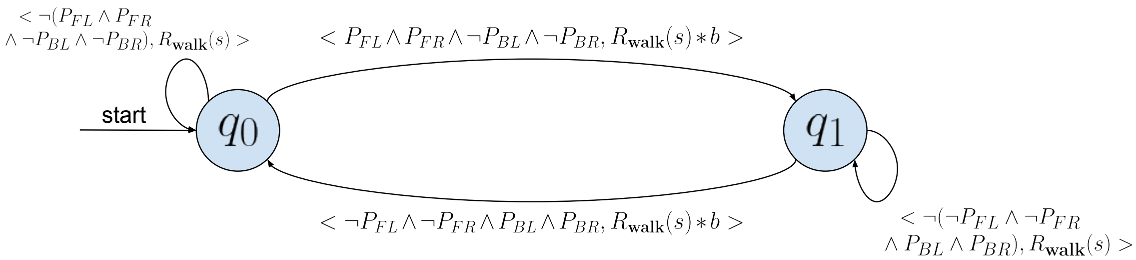

Figure 7: **Bound** gait synchronizes front feet and back feet

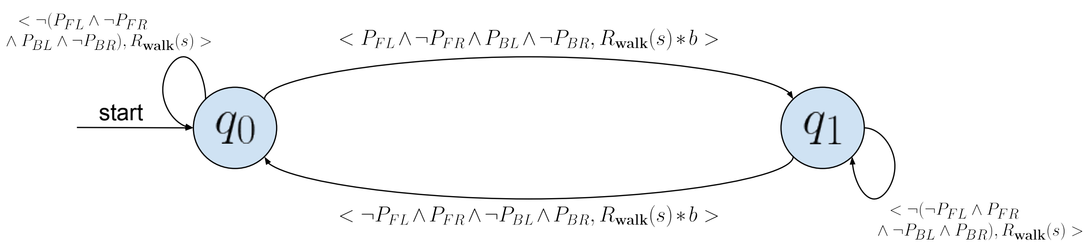

Figure 8: **Pace** gait synchronizes left feet and right feet

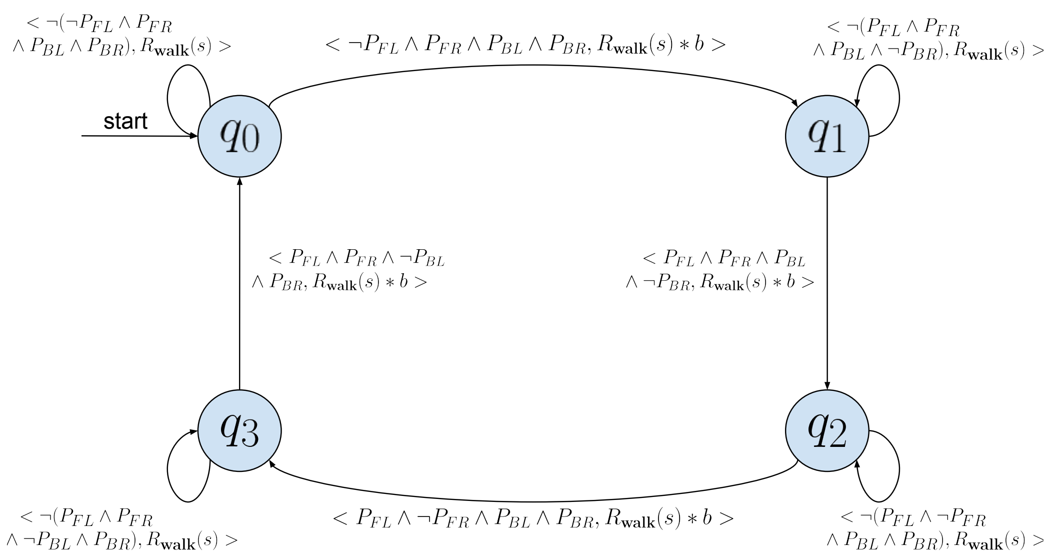

Figure 9: **Walk** gait lifts one foot at a time

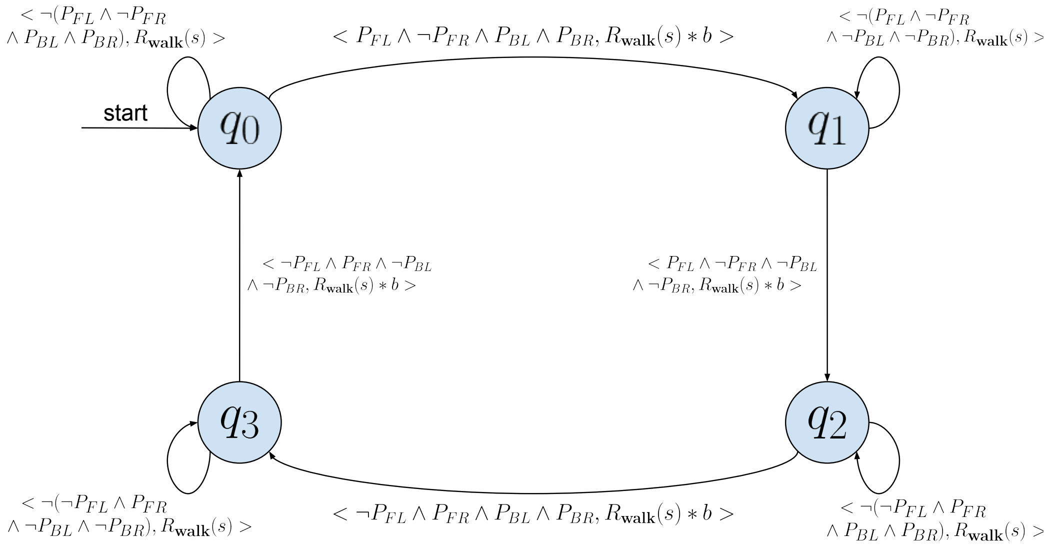

Figure 10: **Three-One** gait alternates three feet with one of the front feet.

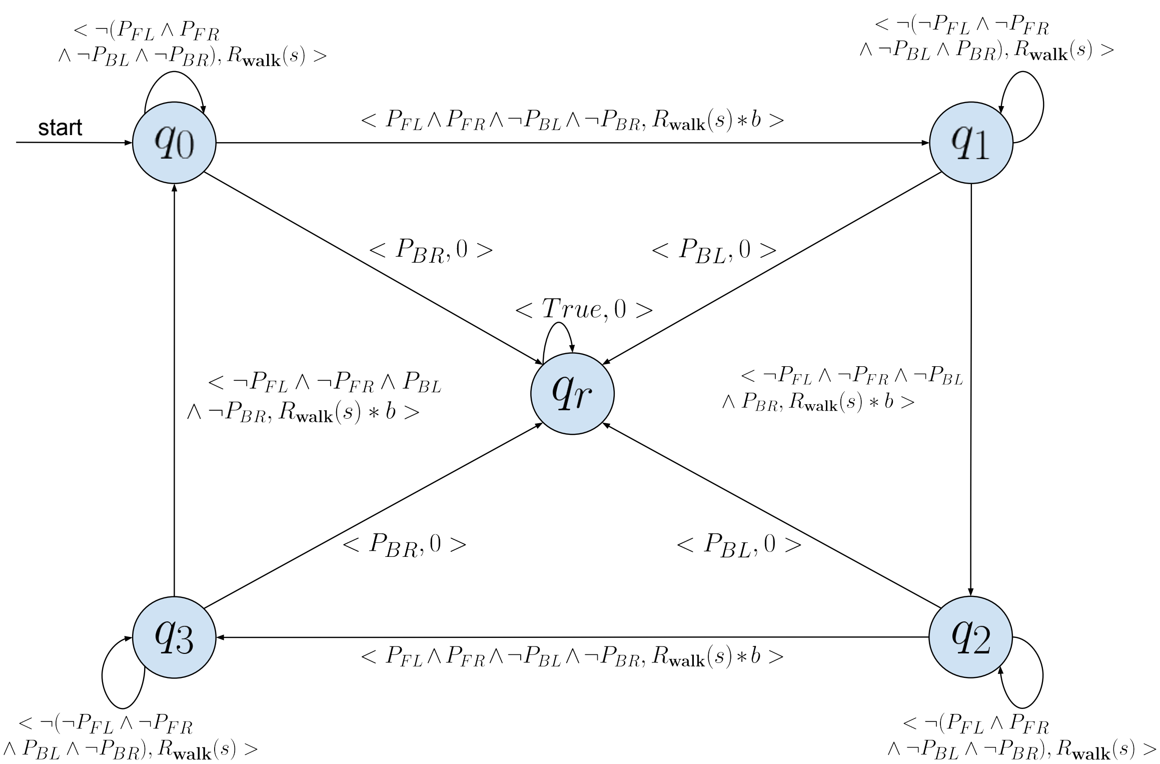

Figure 11: **Half-Bound** gait alternates the front feet with one of the back feet. State q_r discourages extraneous contacts with the wrong back foot, by setting all current and future reward to 0 when reached.

B Gait Specific Training Details

Gaits **Trot**, **Bound**, **Pace**, **Three-One**, and **Half-Bound** sample linear and angular velocity commands from [-1, 1] meters per second, and a gait frequency command from [6, 12] time steps. Meanwhile, **Walk** samples from [-0.5, 0.5] meters per second and [5, 10] respectively. This is because quadruped animals naturally use **Walk** gait for slower locomotion speeds.

All gaits except **Three-One** follow the gait frequency command c_f for all RM transitions that cause a state change. We reduce the amount of time the robot must stand on one leg for **Three-One** gait, by halving c_f for transition $q_1 \rightarrow q_2$ and $q_3 \rightarrow q_0$.

Half-Bound is trained for an additional 50 million time steps than the other gaits, which we find necessary due to the complexity of this gait.