Curating Tunable, Compliant Legs for Specialized Tasks

Journal Title
XX(X):1–14
©The Author(s) 2024
Reprints and permission:
sagepub.co.uk/journalsPermissions.nav
DOI: 10.1177/ToBeAssigned
www.sagepub.com/
SAGE

Fuchen Chen[1] and Daniel M. Aukes[1]

Abstract

Having a well-rounded fixed leg design for a quadruped inevitably limits performance across diverse tasks, while tunability enables specialization and leads to better performance. This paper introduces a sub-500-gram quadruped robot with a rich leg design space. Made with laminate design and fabrication techniques, its legs have a range of tunable design parameters, including leg length, transmission ratio, and passive parallel and series stiffness. The legs are also straightforward to model, low-cost, and fast to manufacture. We propose methods to span the leg's feasible design space and construct simulation environments for training a locomotion policy with reinforcement learning to remove the need for manual controller design and tuning. This policy not only works across leg designs but also exploits the unique dynamics of each leg for better locomotion. A curation process is employed to select designs given performance goals, which is more interpretable than optimization and provides insights for design improvements and discoveries of design principles. Thanks to the tight integration of design, fabrication, simulation, and control, our proposed pipeline produces leg designs with performance that aligns with the simulation, while the learned locomotion policy can be used successfully on the real robot. The fast longitudinal running design reaches a maximum speed of 0.7 m/s or 5.4 body lengths per second, and the low cost of transport (COT) design has a COT of 0.3.

Keywords

legged robots, mechanism design, passive compliance, laminate devices, reinforcement learning

1 Introduction

This paper introduces new methods for spanning a quadruped robot's leg design space and learning locomotion policy that exploits each leg's unique dynamic properties, enabling selections of legs for different use cases. Although many biologists including Ghigliazza et al. (2003); Geyer et al. (2006) and Pontzer (2007), and roboticists including Galloway et al. (2011); Spröwitz et al. (2013); Hubicki et al. (2016); Hutter et al. (2016); Badri-Spröwitz et al. (2022) and Haldane et al. (2016) have shown that leg designs significantly impact legged locomotion performance and the optimum design depends on the operation environment and performance objectives, most quadruped robots in the literature and on the market made by Hutter et al. (2016); Katz et al. (2019); Grimminger et al. (2020); Boston Dynamics (2024) and Unitree (2024) have only one fixed leg design. While they may work well enough for most tasks, some performance will inevitably be sacrificed when dealing with specific scenarios. In this paper, we propose that the legs of quadruped robots should be more tunable so that the best designs can be selected, fabricated, and installed based on the tasks and environments. This is also more cost-effective than developing a brand-new robot for every niche scenario. Additionally, this approach allows designers to discover trends by quickly testing different designs to make more informed decisions and improvements.

As shown in Fig. 1, this paper consolidates this idea into a quadruped robot that has a footprint similar to an adult hand, weighs under 500 grams, and is actuated by eight servo motors. Its legs are designed and fabricated with laminate techniques, enabling us to include many tunable design parameters such as leg length, transmission ratio, and passive stiffness in a compact package. These key parameters can also be automatically mapped to the actual leg geometry, simplifying tuning and remaking. Through our extensive testings and studies performed on these laminate devices over the years, their behaviors can be more easily captured and simulated than conventional design and fabrication processes. This tighter loop of designing, simulating, making, and testing allow us to better explore the robot's design space.

This work also highlights the use of reinforcement learning for training a locomotion policy purely in simulation that directly works across drastically different leg designs in the real world without manual controller design and tuning. More importantly, it can exploit the unique dynamic properties of each leg design for better locomotion performance. Our efforts in aligning the sensor values, physical properties, and performance metrics between the simulated and real environments enable us to evaluate and study the robot's performance in simulation and expect good agreement from the selected designs in the real world. Moreover, since the simulation can be run in parallel across

[1]Ira A. Fulton Schools of Engineering, Arizona State University, Mesa, AZ 85212, USA.

Corresponding author:
Daniel M. Aukes, Ira A. Fulton Schools of Engineering, Arizona State University, Mesa, AZ 85212, USA
Email: danaukes@asu.edu

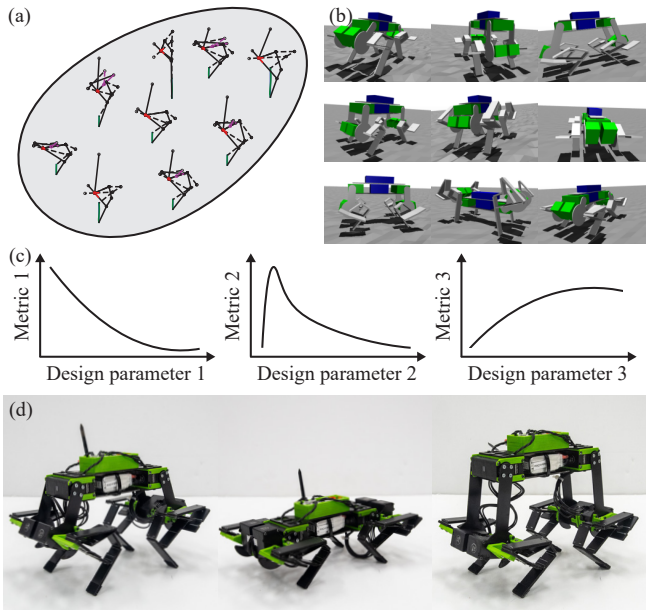

Figure 1. The main components of the proposed quadruped robot with tunable legs. (a) Its design space contains a range of laminate legs with different geometry and passive compliance. (b) A locomotion policy is trained with reinforcement learning in simulation to exploit the dynamics of each leg. (c) It also helps us identify trends between locomotion metrics and design parameters. (d) Three curated standout leg designs have fast longitudinal speed, fast turning speed, and low cost of transport from left to right, respectively. The supplementary video is available at https://youtu.be/A7lSHpgceH4.

multiple CPU cores and at a speed much faster than the real time, we can thoroughly investigate the relationship between performance metrics and design parameters, enabling us to identify potential design improvement directions, confirm existing theories, and even discover new design principles.

In summary, the contributions of this paper are: 1. a quadruped robot platform with a rich leg design space that exists in both the simulation and real world bridged together with laminate design and fabrication techniques; 2. a reinforcement learning framework for training policies that exploit different leg dynamics toward better locomotion performance; 3. and an interpretable and insightful curation process for navigating the robot's performance map.

The remainder of the paper is organized as follows: Section 2 lays out the related literature and justifies the novelties and contributions of this work. Section 3 to 6 dive deep into the multiple sub-components of our proposed method, including the quadruped platform, tunable leg, locomotion policy, and curation. After the experiment details in section 7, section 8 presents and discusses interesting interpretations and discoveries of the results. Section 9 concludes the paper and points out possible future extensions.

2 Background

The literature of biology and robotics has many studies on the impact of leg designs on locomotion performance. The fact that walking and running dynamics can respectively be described by the classical inverted pendulum and spring-loaded inverted pendulum (SLIP) models as discussed by Geyer et al. (2006) highlights the importance of leg length and stiffness for legged locomotion. Pontzer (2007) discovered that the energy cost of transport (COT) for terrestrial animals has a simple inverse relationship with effective limb length or hip height. Ghigliazza et al. (2003) showed that tuning the leg stiffness can achieve robust and stabilized gait even with a straightforward fixed-leg reset control strategy. Many robots have also been developed by Saranli et al. (2001); Spröwitz et al. (2013); Hubicki et al. (2016); Hutter et al. (2016); Haldane et al. (2016) and Badri-Spröwitz et al. (2022) to demonstrate the role of leg stiffness for locomotion speed, stability, impact resistance, efficiency, and power output. Additionally, Galloway et al. (2011) conducted experimental investigations with a hexapod with tunable stiffness C-legs and revealed that payload and terrain both impact the optimum leg stiffness in terms of speed and efficiency.

These prior works are our inspiration and guidance for designing tunable legs and selecting their design parameters. One of our previous works studies the fabrication, tuning, modeling, and simulation of laminate robot legs that weigh around 15 grams in Chen and Aukes (2023). We have demonstrated the ability to design their stiffness coefficients and nonlinearities, affecting their jumping performance. Based on similar principles, we have also designed a 400-gram quadruped with tunable compliant legs in Chen et al. (2023). We have observed various performance responses when adjusting its leg stiffness against open-loop walking gaits.

Although open-loop feed-forward controllers have also been utilized by other researchers including Saranli et al. (2001); Spröwitz et al. (2013) and Badri-Spröwitz et al. (2022), they are often limited to simple conditions and require extensive tuning for different designs. Model-based controllers are promising alternatives for integrating sensory feedback and achieving stable, robust, and fast locomotion as demomnstrated by Katz et al. (2019). Recently, learning-based controllers or policies trained through reinforcement learning are widely adopted by researchers including Lee et al. (2020); Margolis et al. (2024) and Aractingi et al. (2023) for quadrupeds and achieved impressive results thanks to their ability to handle more complex and hard-to-model dynamic systems. Raffin et al. (2022) and Bjelonic et al. (2023) even trained learning-based controllers to work with passive, compliant legs for better performance. Since our proposed robot has many different leg designs with drastically different physical properties, we have selected a learning-based approach to reduce the amount of manual modeling and tuning.

Although there are many advancements in both the design and control of legged robots, most works tend to focus on only one of the aspects mainly: 1. complex leg designs with specialized transmission and passive compliance are usually paired with simple controllers, or 2. state-of-the-art control algorithms are developed and validated on relatively simple and generic platforms. This separation leads to inefficiency in the design process, increases customization difficulties for different applications, and limits the performance ceiling.

To address these issues for quadruped robots, many efforts have been made to simultaneously optimize the leg design and controller, also called "co-design". The process

of robot co-design generally involves an outer loop for optimizing hardware and an inner loop for solving motion planning or optimal control problems similar to works from Chadwick et al. (2020) and Dinev et al. (2022). Interestingly, reinforcement learning is also leveraged for these problems, first to train a design-conditioned policy that works across robots of different designs, and then used to find optimal designs for certain objectives as in Belmonte-Baeza et al. (2022) and Bjelonic et al. (2023). Our approach to identifying an appropriate leg design for a certain performance goal is very similar to this idea with one deviation: instead of relying on optimization to find the design, we span a discretized version of the design space and evaluate all its designs to uncover relationships between performance metrics and design parameters, and to better understand the trends and trade-offs for more informed decision-making. These general trends may also apply to other platforms. Moreover, many existing works focus mostly on simulation with few considerations of fabrication constraints and limited hardware validation. Their robots also have relatively simple structures with minimal closed-loop linkages and passive leg stiffness. In contrast, our work emphasizes generating a rich leg design space that is straightforward and fast to fabricate, promoting more experimental validation and rapid adaptation in the field.

3 Quadruped Platform

As shown in Fig. 2, the proposed quadruped platform has a footprint similar to an adult's hand, weighs under 500 grams, and is symmetric about its xz and yz plane. Each of its legs is actuated by two identical servo motors acting in series: the hip servo swings the leg in the sagittal plane, and the knee servo extends and retracts it. The abduction and adduction joint has been omitted to reduce system complexity, weight, and cost. The robot is also equipped with a microcontroller (MCU) to handle the low-level logic of the servos and an inertial measurement unit (IMU). The locomotion policy is run on a laptop, and the robot communicates with it through a USB cable. The robot is powered by a battery on board. The mechanisms connected to the hip servo are called the "tunable leg". Table 6 in the appendix lists the details of the parts.

From our experience developing and testing the predecessor of this platform in Chen et al. (2023), this relatively small form factor works well with the intended materials and fabrication method and reduces fabrication time and cost. While we have restricted ourselves to studying leg tuning in the sagittal plane, the proposed methods could be extended to more complex platforms. Some design principles discovered should also be applicable to other legged robots.

4 Tunable Leg

4.1 Design and Fabrication

Inspired by the SLIP model, the proposed tunable leg is designed to function similarly to a pogo stick, as shown in Fig. 3. It is composed of four main components: a femur link that connects the hip and knee joint, a four-bar linkage that converts knee rotation into linear foot translation going through the hip, a parallel spring that resists leg retraction,

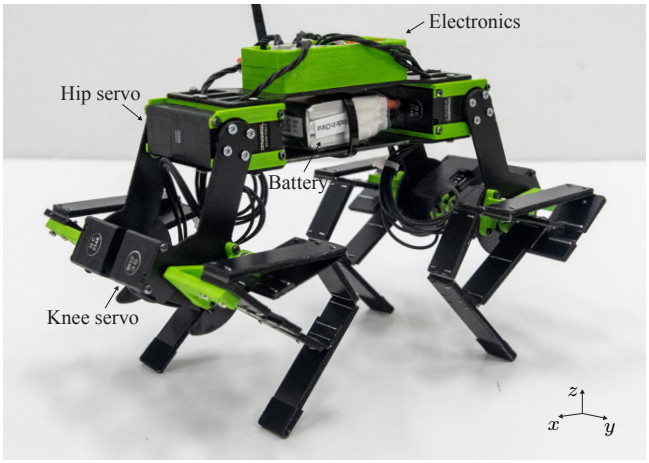

Figure 2. The fabricated quadruped platform with the leg design $\boldsymbol{d}_{v_x}$ for fast longitudinal running. Each of its legs has two servos for the hip and knee joints. An onboard battery powers it. The electronics handle the low-level logic and communication with a laptop.

and a series spring that allows the foot to be displaced even if the knee servo is fixed.

The core of the leg is a multi-layer, multi-material laminate device consisting of two layers of fiberglass sheets with 0.45 mm and 0.72 mm thickness, one layer of 0.1 mm polyester sheet, and two layers of heat-activated adhesives. Table 6 in the appendix lists the details of the materials. The function of a specific section can be altered by changing its number of layers. When all layers are present, the section acts as a rigid link. If only the thin polyester layer is left, this flexure behaves as a simple pin joint. The compliance of the springs can be attributed to the flexible links that only use one of the fiberglass layers, 0.45 mm one for the parallel spring and 0.72 mm one for the series spring as shown in Fig. 3(b) and (c).

To fabricate a leg, the laminate device is first made using the same process in Chen and Aukes (2023). An extra layer of 1.7 mm fiberglass sheet is added to links designed not to bend to improve rigidity. Then, the femur link is also cut from the 1.7 mm fiberglass sheet. Finally, the leg is assembled with screws and some 3D-printed parts. Friction tape is also attached to the foot. Excluding the servo, a single leg weighs around 30 grams.

4.2 Design Parameters

As marked in Fig. 3(d), we define five independent design parameters: the total travel length of the foot $l^d_{ee'}$, the offset between the foot travel and the hip $l^d_{e'o}$, the range of input knee rotation $\theta^d_{faf'}$, parallel stiffness k^d_p, and series stiffness k^d_s. These parameters are more aligned with the SLIP model and form a more feasible and intuitive design space than specifying all the dimensional variables of the proposed leg.

To describe the nonlinear relationship between these parameters and the leg shape, a kinematic model consisting of rigid links and pin joints is developed for the leg, as shown in Fig. 3(d). Although the links of the springs are flexible, our previous study has shown that the pseudo-rigid-body model (PRBM) for an end-loaded cantilever beam with end forces is a simple but effective approximation of their behaviors,

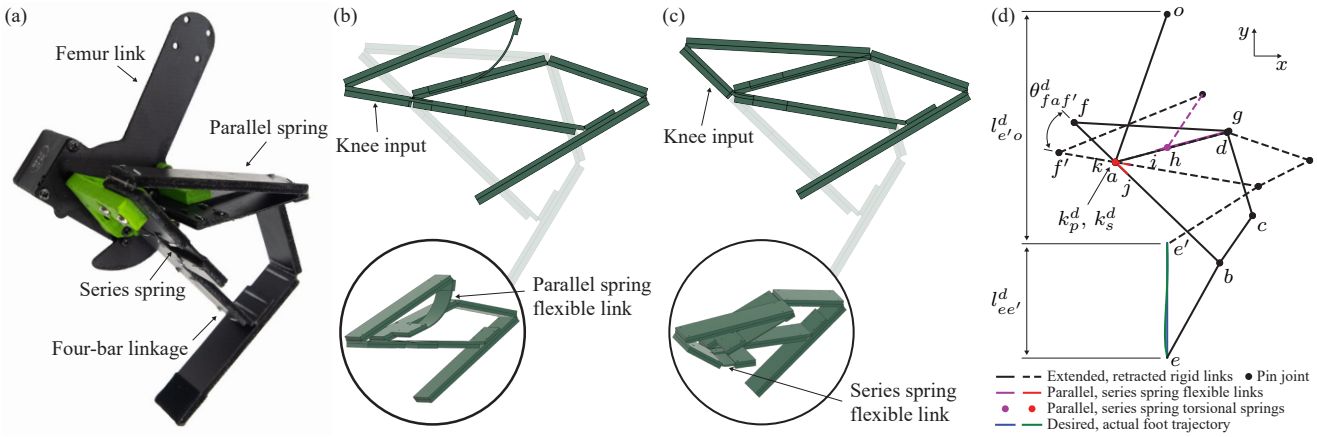

Figure 3. (a) An example of the tunable laminate leg as built. (b) Retracing the foot will bend the flexible link of the parallel spring. (c) The series spring allows foot retraction through the flexible link, even if the input is fixed. (d) Kinematic model of the tunable leg with the design parameters marked.

which converts a flexible link into two rigid links connected through a torsion spring whose spring constant depends on the link geometry and the material's Young's modulus.

For the parallel spring, bending of the link di becomes rotation of the torsion spring at joint h, and the torque required at the knee joint τ_a can be derived with static analysis. Theoretically, the relationship between τ_a and the input knee rotation is not linear, leading to nonconstant parallel stiffness. Still, we approximate it as $k_p = \tau_a^{max}/\theta_{faf'}$, where τ_a^{max} is the required input torque at maximum knee rotation.

For the series spring, since the flexible link jk is purposefully positioned so that the virtual torsion spring aligns with the knee joint, the series stiffness k_s equals the torsion spring stiffness.

4.3 Leg Geometry Optimization

With this kinematic model, a constrained optimization problem is formulated to find the appropriate leg shape that matches a set of design parameters. Specifically, its input variables are the link lengths of the four-bar linkage $\{l_{ab}, l_{bc}, l_{cd}, l_{ad}, l_{be}\}$, the initial crank angle θ_{bag}, and the flexible link lengths for the parallel and series springs $\{l_{gi}, l_{jk}\}$. Other dimensional parameters are either predetermined constants or depend on these. For example, l_{af} is fixed, and l_{fg} is calculated based on the initial configuration.

The selected cost function is the total link length, based on the assumption that a smaller leg mass is desirable for most locomotion scenarios. A smaller mass usually means less energy expenditure, lower material cost, and smaller form factors that lead to less chance of colliding with other legs or environments.

Many constraints are applied to generate desired leg designs, as listed in Table 1, whose main objectives are linear foot travel with the desired offset and length, correct parallel and series stiffness, and a reasonable leg proportion and dimension. The foot trajectory is constrained to deviate less than 2% of the total travel length from the desired straight-line trajectory. These constraints are enforced for 11 leg states corresponding to evenly spaced input angles. To be noted, the desired foot trajectory is also chosen to be evenly spaced. The parallel and series stiffness are constrained to be within 2% of the desired values. To ensure the parallel spring is not deformed beyond yield by the servo, the torque applied to its PRBM joint is limited to a maximum of 0.1 N·m. The transmission angle of the four-bar linkage needs to be greater than 30°(0.52 rad) to stay away from its kinematic singularity. Some vertical clearance between the foot and the remaining points is necessary to avoid other links hitting the floor. The centroid of all the points should also be closer to the foot trajectory horizontally for a more balanced leg. Upper bounds are set for the total length of the leg and all link lengths to reduce unintentional bending and twisting. Lower bounds are also set for all the link lengths to ensure manufacturability. The width of the leg is limited to prevent the front and rear legs from hitting each other or having a tiny swinging range. The leg optimization problem is solved with differential evolution proposed by Storn and Price (1997) and implemented in the SciPy Python package.

Table 1. Leg optimization constraints

Term	Equation
Foot trajectory	$\lvert x_e - x_e^d \rvert < 0.02 l_{ee'}$
	$\lvert y_e - y_e^d \rvert < 0.02 l_{ee'}$
Parallel stiffness	$\lvert k_p - k_p^d \rvert < 0.02 k_p^d$
Series stiffness	$\lvert k_s - k_s^d \rvert < 0.02 k_s^d$
Parallel spring PRBM joint torque	$\tau_h < 0.1\,N \cdot m$
Transmission angle	$\pi - \theta_{bcg} > 0.52\,rad$
Foot clearance	$y_e - y_{other} < -0.02\,m$
Leg Centroid	$\lvert \frac{1}{n_{all}} \sum_{all} x - x_o \rvert$
	$< 0.01\,m$
Total leg length	$l_{eo} < 0.14\,m$
Leg width	$\lvert x_{any} - x_o \rvert < 0.05\,m$
Femur link length	$0.028 < l_{ao} < 0.1\,m$
Parallel spring flexible link length	$0.005 < l_{gi} < 0.1\,m$
Series spring flexible link length	$0.005 < l_{jk} < 0.1\,m$
Other link lengths	$0.02 < l_{other} < 0.1\,m$

4.4 Design Space

The leg design space $\boldsymbol{D} \in \boldsymbol{R}^5$ formed by the vectors of design parameters $\boldsymbol{d} = [l_{oe'}, l_{ee'}, \theta_{faf'}, k_p, k_s]$ is discretized with a step size $\boldsymbol{s_D}$. A simple algorithm similar to the "flood fill" used in computer graphics for painting a region as documented in Newman and Sproull (1979) is proposed to span $\boldsymbol{D}$, as shown in Algorithm 1. The algorithm grows the design space by repeatedly checking the neighbors of known valid designs and adding new valid ones. A design is considered valid if the optimization routine can find a leg design that satisfies all the constraints. The neighbor of a design $\boldsymbol{d}$ is defined as $\boldsymbol{d_n} = \boldsymbol{d} + \boldsymbol{s_D} \cdot \boldsymbol{I_{row}}$ where $\boldsymbol{I_{row}}$ is one of the rows of the identity matrices $\boldsymbol{I_5}$ and $-\boldsymbol{I_5}$, which is an extension to the 2 dimensional 4-connectivity. We also require that $\boldsymbol{d_n} \geq \boldsymbol{d_0}$ where $\boldsymbol{d_0}$ is the lower feasible limit of the design parameters. It should be noted that, due to the stochastic nature of our chosen optimization algorithm, multiple trials may be needed to find a valid leg for the same design parameter; invalid designs are added back to a buffer $\boldsymbol{D'}$ for checking again. For the same reason, our current implementation may miss a small number of valid designs.

Algorithm 1 Spanning the design space

$\boldsymbol{D'} = \{\boldsymbol{d_0}\}$
$\boldsymbol{D} \leftarrow \emptyset$
for iteration=1,2,...,n **do**
  remove and get last element $\boldsymbol{d}$ from $\boldsymbol{D'}$
  if $\boldsymbol{d}$ generates valid design **then**
    $\boldsymbol{D} \leftarrow \boldsymbol{D} \cup \{d\}$
    for $\boldsymbol{d_n} \in$ neighbors of $\boldsymbol{d}$ **do**
      if $\boldsymbol{d_n} \notin \boldsymbol{D'}$ **and** $\boldsymbol{d_n} \notin \boldsymbol{D}$ **and** $\boldsymbol{d_n} \geq \boldsymbol{d_0}$ **then**
        $\boldsymbol{D'} \leftarrow \boldsymbol{D'} \cup \{\boldsymbol{d_n}\}$
      end if
    end for
  else
    $\boldsymbol{D'} \leftarrow \{\boldsymbol{d}\} \cup \boldsymbol{D'}$
  end if
end for
return $\boldsymbol{D}$

5 Locomotion Policy

5.1 Policy Design

In this work, learning a neural network for design-aware locomotion control is preferred over model-based approaches for the following reasons: robot dynamics are complex and difficult to model due to leg properties that can change design-to-design, including the range of motion, inertia, and passive compliance; modeling errors and stochasticity are also inevitably introduced from the fabrication processes and testing environments, but we want to avoid manual tuning of each leg design; furthermore, the hardware has limited torque control accuracy and bandwidth, further increasing the difficulty of implementing model-based controllers. While this approach requires longer training time and sacrifices some interpretability, it represents an acceptable trade-off and best suits our case.

To control the quadruped with any legs discovered in the design space, we propose a locomotion policy:

$$\pi(\boldsymbol{a_t}|\boldsymbol{v_t^{cmd}}, \boldsymbol{q_t}, \boldsymbol{\dot{q}_t}, \boldsymbol{g_t}, \boldsymbol{w_t}, \boldsymbol{d}, \boldsymbol{a_{t-1}}) \,. \tag{1}$$

The main goal of the policy is to make the robot follow a velocity command $\boldsymbol{v_t^{cmd}}$ consisting of the longitudinal speed v_x^{cmd} and turning speed w_z^{cmd}. Since our robot lacks active abduction and adduction, we do not command lateral speeds v_y^{cmd}. The policy has access to a range of sensory feedback that is also available on the actual prototype, including joint angles $\boldsymbol{q_t} \in \boldsymbol{R}^8$, joint velocities $\boldsymbol{\dot{q}_t} \in \boldsymbol{R}^8$, body orientation $\boldsymbol{g_t} \in \boldsymbol{R}^3$ expressed as the gravity direction in the robot's body frame, and body angular velocities $\boldsymbol{w_t} \in \boldsymbol{R}^3$. Since the policy must accommodate different leg designs, the design vector $\boldsymbol{d}$ is also provided in the observations. Additionally, as discovered in preliminary experiments, the policy observes its last action $\boldsymbol{a_{t-1}}$, essential for successful learning. The action $\boldsymbol{a_t} \in \boldsymbol{R}^8$ represents the desired joint positions sent to the servos.

The policy is a fully connected neural network with three hidden layers of size $[512, 256, 128]$ and ELU activations operating at 100 Hz. The actions and observations are scaled and offset to approximately within the range of $[-1, 1]$ before being fed into the policy.

5.2 Learning Environment

5.2.1 Simulation MuJoCo has been selected as the physics simulator for its capability of simulating closed-loop linkages, fast computation, and versatile modeling options. Since the legs of our robot are mostly made of simple geometries, it is straightforward to parameterize its simulation model so that the sizes and poses of all the links will be automatically updated to reflect a given leg design. As the robot's servos do not have accurate torque control, we decided to model the dynamics of the entire servo, including its motor, gearbox, and internal position PID controller, as an inertia-spring-damper system, which takes a desired angle as the input and outputs a torque. A maximum output torque constraint and static friction within the motor are also added. These values are determined through a set of system identification experiments.

The frame rate of the simulation is 500 Hz, 5 times the operation rate of the locomotion policy. Since our locomotion task has no definite termination condition, a constant time limit of 5 seconds is employed for every episode. The velocity command $\boldsymbol{v^{cmd}}$ for the entire episode is sampled with a curriculum strategy that will be discussed later. To expose the policy to different legs, a leg design is also uniformly sampled from the design space D for each episode.

5.2.2 Domain randomization As listed in Table 2, randomization of the simulation environment, including servo dynamics parameters, floor properties, sensor noise, and system latency, is also applied to reduce overfitting and make the policy directly deployable in the real world. The range of the servo parameters are $\pm 10\%$ around the experimentally identified values to accommodate changes of the battery voltage, identification inaccuracy, and variations between individuals. The floor friction coefficient is set to a relatively large range because the condition of the feet's

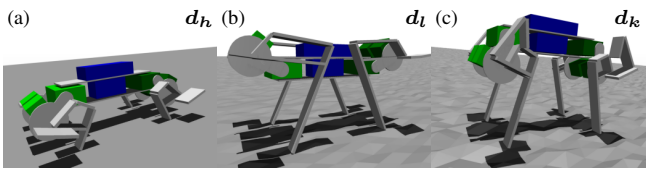

Figure 4. Three examples of the simulation environment for training the locomotion policy. The floor tilt is -5, 0, and 5 degrees, and roughness is 0, 0.005, 0.01 m for (a), (b), and (c), respectively. The three robots have leg designs of the shortest height, longest foot travel, and highest stiffness.

Table 2. Domain randomization

Term	Min	Max	Unit
Servo spring constant	1.64	2.00	N·m/rad
Servo damping constant	0.0457	0.0559	Nm/(rad/s)
Servo inertia	4.65e-4	5.69e-4	kg·m^2
Servo friction	0.0201	0.0245	N·m
Servo maximum torque	0.519	0.635	N·m
Floor friction coefficient	0.4	1.0	
Floor roll and pitch	-0.0873	0.0873	rad
Floor roughness	0.00	0.01	m
Joint angle noise	-0.05	0.05	rad
Joint velocity noise	-1	1	rad/s
Body orientation noise	-0.05	0.05	9.81 m/s^2
Body angular velocity noise	-0.2	0.2	rad/s
Latency	0.03	0.07	s
Latency noise	-0.005	0.005	s

friction tape due to wear and tear and the floor type and cleanliness can affect the friction significantly. Tilting and adding roughness to the floor improves the robustness of the learned policy. Sensor noise and latency are added to better reflect real scenarios. We find that latency significantly impacts the robot's velocity tracking performance, and its range is based on the experimentally measured value with some extra tunings. Examples of the simulation environment are shown in Fig. 4.

5.2.3 Reward Several reward terms are designed to guide the policy through the reinforcement learning process toward desirable behaviors. They can be grouped into four categories: velocity command tracking, motion posture, action smoothness, and energy efficiency. All the terms are listed in Table 3, and the total reward is their weighted sum. We intentionally kept the number of terms minimal to avoid introducing too much bias and provide the training process more freedom to discover better strategies. Most of these rewards are commonly used in the literature such as Lee et al. (2020); Aractingi et al. (2023) and Margolis et al. (2024). We add a term that penalizes the hip joint for not moving around the home position, which helps speed up the learning process and prevents the robot from settling at a sub-optimal gait with the legs spreading outward.

Table 3. Reward terms

Term	Equation	Weight
Velocity command tracking	$exp(-25\|\boldsymbol{v} - \boldsymbol{v}^{\boldsymbol{cmd}}\|^2)$	2
Body roll and pitch	$\|\boldsymbol{g}'_{\boldsymbol{xy}}\|^2$	-10
Hip joint off-center motion	$\|\frac{1}{50}\sum_{i=t-49}^{t}\boldsymbol{q}_{\boldsymbol{hip}_i}\|^2$	-10
Knee joint limit violation	$\|max(\boldsymbol{q}_{\boldsymbol{knee}} - \theta_{faf'}, \boldsymbol{0}) - min(\boldsymbol{q}_{\boldsymbol{knee}}, \boldsymbol{0})\|^2$	-100
Self-collisions	$n_{self-collisions}$	-1
Action rate	$\|\dot{\boldsymbol{a}}\|^2$	-1e-5
Joint torque	$\|\boldsymbol{\tau}\|^2$	-0.5
Joint power	$\|max(\boldsymbol{\tau}\dot{\boldsymbol{q}}, \boldsymbol{0})\|$	-0.02

5.3 Curriculum

For a single quadruped design, the Grid Adaptive curriculum proposed by Margolis et al. (2024) is a promising way to gradually push a robot to its locomotion limit and determine its feasible velocity command space. Although our strategy is similar to this existing work and the concept also aligns with the algorithm for spanning the leg design space, a description is still included for clarity and to highlight the modifications.

The discretized velocity command space $\hat{\boldsymbol{V}}^{\boldsymbol{cmd}} \in \boldsymbol{R}^2$ contains velocity commands $\hat{\boldsymbol{v}}^{\boldsymbol{cmd}} = [\hat{v}_x^{cmd}, \hat{w}_z^{cmd}]$ with a step size $\boldsymbol{s}_{\hat{\boldsymbol{V}}^{\boldsymbol{cmd}}}$. As shown in Algorithm 2, the curriculum keeps expanding the command space by adding the neighbors of a command that the policy can track. The policy is considered capable of tracking a command if $ema(\bar{r}_v) > r_v^{th} = exp(-25\|0.5\boldsymbol{s}_{\hat{\boldsymbol{V}}^{\boldsymbol{cmd}}}\|^2)$, where $ema(\bar{r}_v)$ is the exponential moving average (EMA), with a 0.2 smoothing factor, of the average velocity reward of an episode. The EMA is not implemented in the original paper. Still, we find it useful to ensure the policy truly masters that velocity region instead of passing it because of luck. Additionally, our definition of neighbors of a command includes not only the 4-connected ones but also their reflections about both axes. This can be expressed as $\hat{\boldsymbol{v}}_{\boldsymbol{n}}^{\boldsymbol{cmd}} = (\hat{\boldsymbol{v}}^{\boldsymbol{cmd}} + \boldsymbol{s}_{\hat{\boldsymbol{V}}^{\boldsymbol{cmd}}} \cdot \boldsymbol{I}_{\boldsymbol{row}}) \cdot \boldsymbol{T}$ where $\boldsymbol{I}_{\boldsymbol{row}}$ is one of the rows of the identity matrices $\boldsymbol{I_2}$ and $-\boldsymbol{I_2}$, and $\boldsymbol{T} \in \{[1,1],[-1,1],[1,-1],[-1,-1]\}$. Since our robot is symmetric, this encourages the policy to learn locomotion in all directions equally. It is worth mentioning that the actual velocity command for an episode is $\boldsymbol{v}^{\boldsymbol{cmd}}$ which is uniformly chosen within $(\hat{\boldsymbol{v}}^{\boldsymbol{cmd}} - 0.5\boldsymbol{s}_{\hat{\boldsymbol{V}}^{\boldsymbol{cmd}}}, \hat{\boldsymbol{v}}^{\boldsymbol{cmd}} + 0.5\boldsymbol{s}_{\hat{\boldsymbol{V}}^{\boldsymbol{cmd}}})$, where $\hat{\boldsymbol{v}}^{\boldsymbol{cmd}}$ is also uniformly sampled from the latest $\hat{\boldsymbol{V}}^{\boldsymbol{cmd}}$.

For a quadruped with varying leg designs, offering an individual curriculum for every design may slow down the training. Many episodes will be wasted on verifying whether the policy has mastered the same task instead of exposing it to new and more challenging ones. It may also require too much memory if the design space is enormous. Therefore, we only provide individual curriculum to a smaller number of delegates, each representing a group of similar designs, which should have similar locomotion characteristics and a single velocity command space covers them. The delegates

Algorithm 2 Velocity command curriculum

$\hat{\boldsymbol{V}}^{\boldsymbol{cmd}} \leftarrow \{\boldsymbol{0}\}$
while training **do**
  get $\hat{\boldsymbol{v}}^{\boldsymbol{cmd}}$ of an episode when it finishes
  if $ema(\bar{r}_v) > r_v^{th}$ **then**
    for $\hat{\boldsymbol{v}}_{\boldsymbol{n}}^{\boldsymbol{cmd}} \in$ neighbors of $\hat{\boldsymbol{v}}^{\boldsymbol{cmd}}$ **do**
      if $\hat{\boldsymbol{v}}_{\boldsymbol{n}}^{\boldsymbol{cmd}} \notin \hat{\boldsymbol{V}}^{\boldsymbol{cmd}}$ **then**
        $\hat{\boldsymbol{V}}^{\boldsymbol{cmd}} \leftarrow \hat{\boldsymbol{V}}^{\boldsymbol{cmd}} \cup \{\hat{\boldsymbol{v}}_{\boldsymbol{n}}^{\boldsymbol{cmd}}\}$
      end if
    end for
  end if
end while

are found by applying the k-means clustering algorithm over the entire leg design space $\boldsymbol{D}$, during which every leg design is also labeled with a corresponding delegate for locating the curriculum during training.

6 Curation

6.1 Evaluation

Three metrics are established for evaluating the locomotion performance of each leg design paired with the trained policy in the simulation. The first metric represents how fast a design can run along its longitudinal direction while tracking the commanded speed and is calculated as

$$m_{v_x} = \sum |[1,0] \cdot \hat{\boldsymbol{V}}^{\boldsymbol{cmd}}|_i \,, \tag{2}$$

where $\hat{\boldsymbol{V}}^{\boldsymbol{cmd}}$ is the design's feasible velocity command space in the form of a matrix whose rows are different $\hat{\boldsymbol{v}}^{\boldsymbol{cmd}}$. This essentially sums up all the trackable longitudinal speed commands $\hat{v}_x^{cmd}$.

Similarly, the second metric represents how fast a design can turn while tracking the commanded speed and is calculated as

$$m_{w_z} = \sum |[0,1] \cdot \hat{\boldsymbol{V}}^{\boldsymbol{cmd}}|_i \,. \tag{3}$$

Finally, the third metric represents the average COT of a design tracking velocity commands with nonzero longitudinal speed:

$$m_{COT} = \frac{1}{n_{\hat{v}_x^{cmd} \neq 0}} \sum COT_{\hat{v}_x^{cmd} \neq 0} \,, \tag{4}$$

where COT is the average COT for a velocity command region around $\hat{\boldsymbol{v}}^{\boldsymbol{cmd}}$.

To determine each design's feasible velocity command space $\hat{\boldsymbol{V}}^{\boldsymbol{cmd}}$, the same Algorithm 2 is reused with two modifications. First, for a design to be considered capable of tracking velocity commands around $\hat{\boldsymbol{v}}^{\boldsymbol{cmd}}$, the 100 episodes average of the average velocity reward $\bar{r}_v$ after the robot reaches steady state has to be greater than r_v^{th}. For evaluation, each episode is 2 seconds long, and the robot is considered in a steady state after 1 second, which is determined based on an observation that most runs settle after 0.4 seconds and applying a safety factor. Second, the neighbors no longer include reflections that is $\hat{\boldsymbol{v}}_{\boldsymbol{n}}^{\boldsymbol{cmd}} = (\hat{\boldsymbol{v}}^{\boldsymbol{cmd}} + \boldsymbol{s}_{\hat{\boldsymbol{V}}^{\boldsymbol{cmd}}} \cdot \boldsymbol{I}_{\boldsymbol{row}})$. In post-processing, we only keep commands whose reflections are also trackable so that the feasible velocity command space remains symmetric.

The COT in (4) is estimated with the data from the same episodes. For each episode, $COT_i = \frac{\bar{p}}{mg|\bar{v}_x|}$ where $\bar{p}$ and $\bar{v}_x$ are the average positive mechanical power and average actual longitudinal speed during steady state. We do not consider the electrical power because it is not modeled in the simulation, but the COT is calculated similarly using experimental data from the real robots.

6.2 Selection

Once the metrics for all leg designs are obtained, selecting a standout design in a specific metric can be accomplished by sorting and picking the best one. However, due to the discretization of the velocity command space, it is possible to have multiple designs with identical scores. Moreover, since the evaluation process is based on the randomized simulation environment, the design with the best score may be a fluke and not represent the majority of the top designs. Therefore, for each metric, the design with the smallest average Euclidean distance $\|\boldsymbol{d} - \boldsymbol{d'}\|$ to all the top 10% designs is selected.

7 Experiments

The proposed algorithms and reinforcement learning were implemented with Python and run on a workstation computer*. The first step of our experiment was to span the design space of the proposed tunable leg. We chose a lower limit $\boldsymbol{d_0} = [0.04\,m, 0.02\,m, 0.3\,rad, 0\,N \cdot m/rad, 0.4\,N \cdot m/rad]$ and a step size $\boldsymbol{s_D} = [0.02, 0.02, 0.3, 0.1, 0.4]$. These values were determined empirically to have a space that uncovers trends but is not too dense to waste computation time. We ran Algorithm 1 for 20000 steps or about 10.5 hours to span the design space. Since the rate of design discovery became very slow at the end, the design space was considered sufficiently explored, and enough designs were gathered.

With the design space established, we then trained the locomotion policy with a custom implementation of Proximal Policy Optimization reinforcement learning developed by Schulman et al. (2017) modified from CleanRL by Huang et al. (2022) with additional time limit handling proposed by Pardo et al. (2018). The simulation environment was computed on the CPU, and the policy optimization was performed on the GPU. The hyperparameters are listed in Table 4. For the curriculum, we chose a velocity step size $\boldsymbol{s}_{\hat{\boldsymbol{V}}^{\boldsymbol{cmd}}} = [0.2\,m/s, 0.2\,rad/s]$, which provided a reasonable resolution for the feasible velocity command space. 10 delegate designs were used since we observed that further increasing the number of delegates does not result in noticeable performance improvement from early runs. The training process was terminated after 63000 updates or about 21 hours, at which the command spaces' growth rate became very slow. Once the training finished, the final policy was used for evaluation, which took about 11 hours.

To validate that our proposed methods can curate leg designs from performance metrics and that the real robot

*AMD Ryzen Threadripper PRO 7975WX, NVIDIA GeForce RTX 4090

Table 4. Training hyperparameters

Parameter	Value
# of environments	1024
# of steps per update	50
Optimizer	Adam
Learning rate	adaptive
# of mini-batches	5
# of epochs	5
Discount factor	0.99
GAE parameter	0.95
Clip coefficient	0.2
Policy loss coefficient	1
Value loss coefficient	0.5
Entropy coefficient	0.01
Value loss clipping	true
Advantages normalization	true
Maximum gradient norm	0.5

behaves similarly as predicted by the simulation, three standout designs were selected based on the metrics defined in section 6 A. The legs for each design were then fabricated and attached to the base. This took about 4 hours from scratch for an experienced person. To measure their performance, each version was given velocity commands that were combinations of $v_x^{cmd} \in \{-0.8, -0.6, \ldots, 0.8\}$ and $w_z^{cmd} \in \{-0.4, -0.2, \ldots, 0.4\}$. Three trials were performed for each command. In each trial, the robot started from standing still and moved for 2 seconds under the control of the trained locomotion policy, which makes the same observations and outputs the same actions as in (1). The velocity data of the robot were recorded by a motion capture system[†]. Servo current and joint velocity data were also recorded to calculate the COT. For comparison, similar experiments were carried out in simulation.

8 Results and Discussions

8.1 Design Space

With the proposed tunable leg and design space spanning method as in section 4 and related settings in section 7, a total of 409 leg designs were found. Some design examples are in Fig. 6. To visualize the five-dimensional design space, plots of it projected onto selected planes formed by pairs of design parameters are shown in Fig. 5. This is a useful tool for analyzing the distribution of design parameters and their interactions, providing insights on the limits of current design and directions of improvement, as explained next.

In Fig. 5(a), the designs all lie within a triangular region on the travel offset and travel length $l_{oe'}l_{ee'}$ plane because the total leg length $l_{oe'} + l_{ee'}$ is explicitly limited during the leg optimization as in Table 1.

The input range and travel length $l_{ee'}\theta_{faf'}$ plane deserves some attention as plotted in Fig. 5(b) because it is related to the transmission ratio or moment arm $r = l_{ee'}/\theta_{faf'}$ of the four-bar linkage that converts knee rotation to linear foot travel, which balances the force and speed capability of the leg in retraction and extension direction. In this case, the tunable range of the moment arm is from 0.022 m to 0.067 m, which is implicitly determined by the constraints applied and the proposed four-bar linkage design.

As shown in Fig. 5(c), the diversity of achievable parallel stiffness k_p decreases almost exponentially as the input range $\theta_{faf'}$ increases. This is because the deformation of the parallel spring required by a large input range is not physically feasible, and this is enforced by constraining the maximum PRBM joint torque of the parallel spring. In contrast, the achievable series stiffness is not affected by the input range, as shown in Fig. 5(d).

To be noted, although the proposed tunable leg can achieve a variety of designs, their distribution is far from uniform. There are more designs around short travel length, small input range, and no parallel stiffness, which are related to the nonlinear behaviors from using linkages to convert rotation to translation and provide parallel stiffness under constraints of physical feasibility.

In summary, the proposed tunable leg covers a variety of design parameters, and Algorithm 1 is able to span the design space. The visualization tool also helps us identify potential design improvements. For example, if the range of the moment arm needs to be expanded, some constraints should be loosened, or another mechanism such as a six-bar linkage or a rack and pinion should be explored. In addition, ways to increase the range of parallel stiffness should be considered, such as changing the material or spring design.

8.2 Locomotion Performance

Fig. 6 shows evaluation results of 10 arbitrarily chosen leg designs out of all 409 designs evaluated using the method described in section 6 A. Our trained locomotion policy generates different feasible velocity command spaces for different leg designs. The maximum trackable longitudinal speed is 0.8 m/s, and the maximum turning speed is 0.4 rad/s. As the magnitude of velocity command increases, the velocity reward or tracking performance decreases. Additionally, the COT is also significantly affected by the leg design. The COT also does not decrease monotonically, and there is an optimum longitudinal speed for minimum COT, which also depends on the leg design.

Since we evaluated all leg designs, it is possible to study the relationship between the locomotion performance and design parameters. This can reveal trends that confirm existing or even discover new design principles, helping designers make more informed decisions. Specifically, the scores of the performance metrics m_{v_x}, m_{w_z}, and m_{COT} can be plotted against the design parameters as in Fig. 7. Instead of using the design parameters that forms the design space, we established 4 more derived design parameters that we found are more impactful on the performance. They are total leg length $l_{oe'} + l_{ee'}$, moment arm $r = l_{ee'}/\theta_{faf'}$, equivalent parallel stiffness k_p/r^2, and equivalent series stiffness k_s/r^2. The equivalent stiffness represents the leg stiffness along the retraction and extension direction as in the SLIP model.

When looking at the average scores versus design parameters, there are several trends we want to highlight.

[†]NaturalPoint OptiTrack Prime 17W

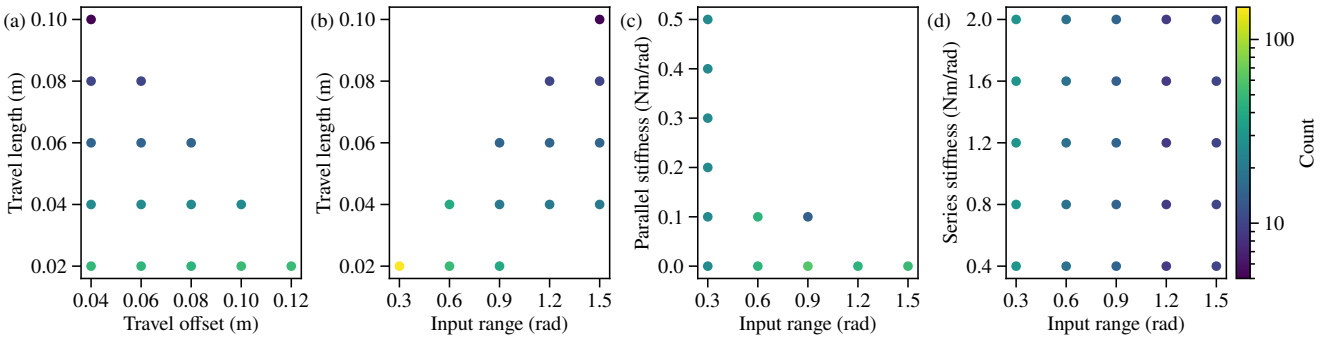

Figure 5. Visualization of the leg design space projected onto selected planes. The colors indicate the number of designs for that point. (a) The travel offset and travel length plane shows that all designs follow the total leg length constraint. (b) The input range and travel length plane indicates the achievable transmission ratio or moment arm. The (c) parallel or (d) series stiffness and the input range planes show that the variety of parallel stiffness depends more on the input range.

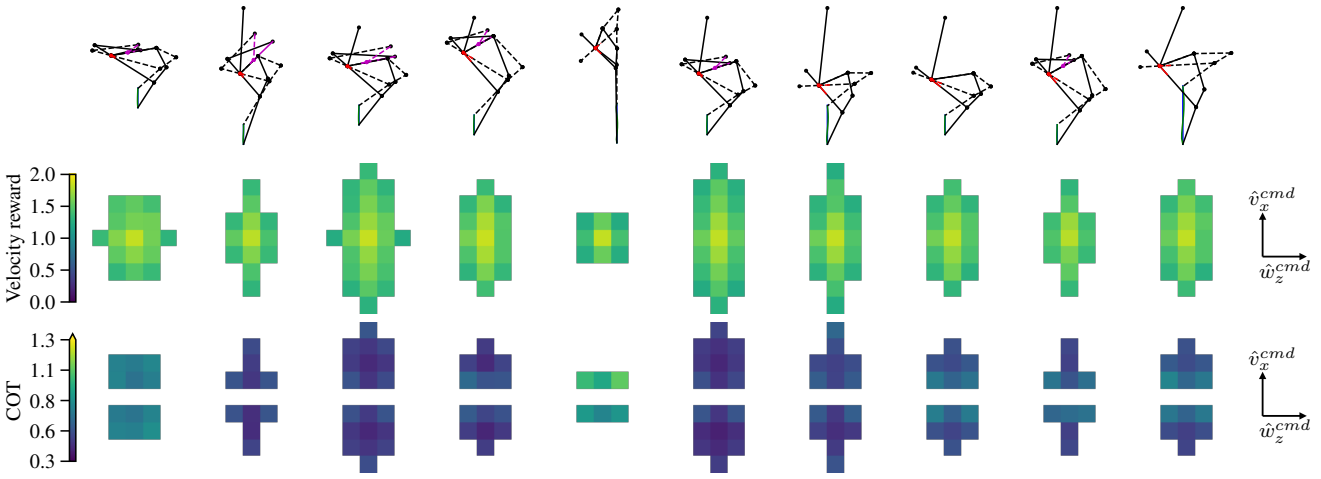

Figure 6. 10 arbitrarily chosen leg designs, and their feasible velocity command space and the corresponding COT. Each cell has a size of 0.2 m/s and 0.2 rad/s. Both commands can be either positive or negative. The COT values for longitudinal speed commands close to zero are omitted.

First of all, there exists a similar optimum series stiffness around 400 N/m for fast longitudinal and turning speed as well as low COT. This stiffness, with our robot weighing about 0.5 kg, corresponds to a natural frequency of 4.5 Hz in a one-dimensional spring-mass system, which aligns with the gait frequency of the locomotion policy. This showcases the effectiveness of our proposed training method in co-designing the series leg stiffness and control policy. Even though the notion of natural frequency and resonance is not introduced anywhere in the process, it is still able to arrive at the same conclusion. This also confirms and emphasizes the usefulness of adding series stiffness to the legs of quadruped robots. Moreover, our proposed method directly generates a paired control policy that exploits this passive compliance without manual controller design and tuning. Similarly but less pronouncedly, the optimum parallel stiffness is around 25 N/m, which we assume is related to the servo torque and operation efficiency.

The second trend is that longer legs have lower COT, which agrees with the LiMb model proposed by Pontzer (2007) that links limb length to COT in biomechanics. We rediscovered this relationship through our approach, though it wasn't part of our original literature review. This experience gives us confidence that our proposed method has the potential to discover new design principles and confirm existing theories that lead to more performant robots. It achieves this through automating the controller development of difficult tasks, which, in this case, is efficient running with a high center of mass and constrained leg workspace due to potential collisions between long legs.

Lastly, a large moment arm improves all three metrics, indicating that speed is preferred over force, but the locations of the optimum points are not conclusive, especially for the fast turning and low COT. This implies that our selected servo motor's 181 to 1 gear ratio is too large. Reducing it in the future will lead to a better understanding of the role of gear ratio in locomotion. This demonstrates that our proposed method can inform designers of directions for improvement.

8.3 Experimental Validation

The three standout designs selected for experimental validation are marked with x's in Fig. 7. Their shapes are plotted in Fig. 8 for comparison. The low COT design d_{COT} has the longest leg length and the fast turning design d_{w_z} has the shortest. The fast longitudinal running design d_{v_x} has longer foot travel. They all have similar moment arms, parallel stiffness, and series stiffness. The photos of the robot equipped with the three different legs are shown in Fig. 1.

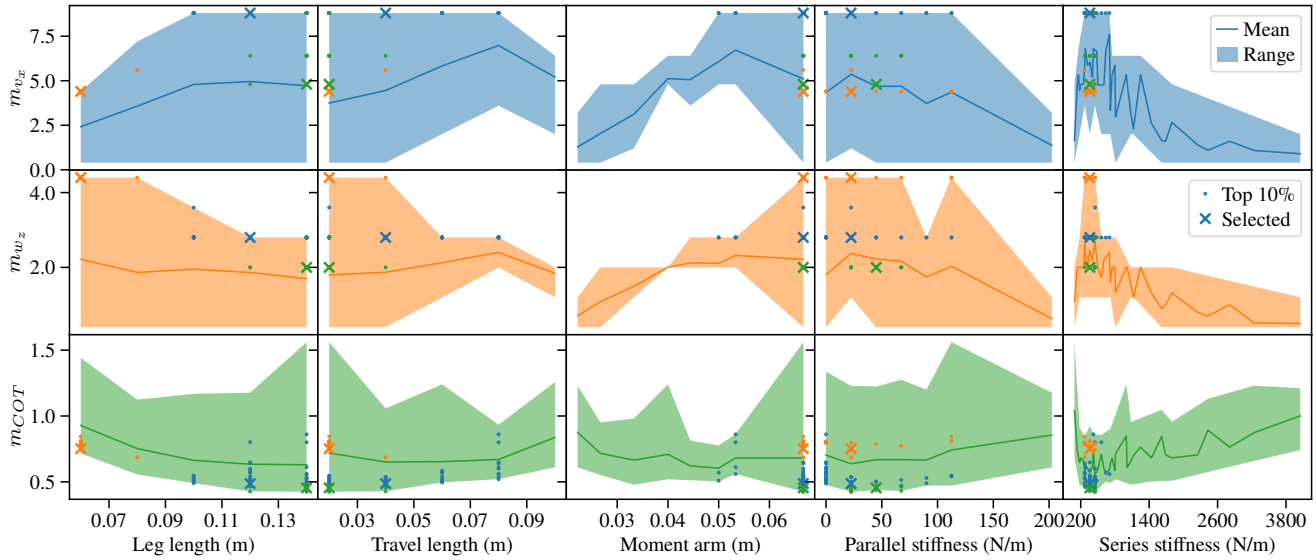

Figure 7. The scores of the three defined performance metrics versus the derived design parameters. The solid lines represent the average score and the shaded regions indicate the full range of the scores. Small dots show the top 10% designs for each metric, and the x's are the selected standout designs among them. There are two most significant trends: an optimal series stiffness exists for all three metrics, and longer legs have lower COT.

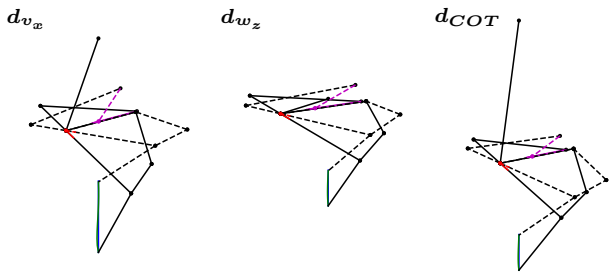

Figure 8. Three selected standout leg designs for (a) fast longitudinal running, (b) fast turning, and (c) low COT.

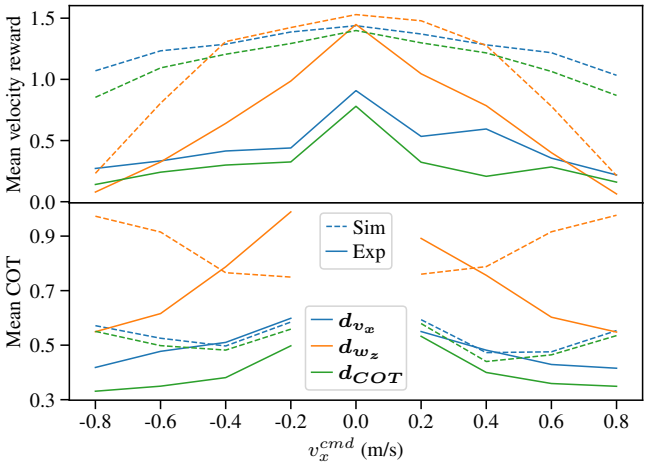

Figure 9. The velocity reward and COT averaged over each commanded longitudinal speed for the three standout leg designs in simulation and real-world experiments.

To compare the measured performance of the three leg designs over the tested velocity command space in both the simulation and real world, the velocity reward and COT averaged over each longitudinal speed command are plotted in Fig. 9. These legs perform differently. Specifically, the fast longitudinal running leg has a higher velocity reward at high speeds. The fast turning leg has a higher reward at lower speeds thanks to its better ability to track larger turning speeds. The low COT leg consistently achieves lower COT across all commanded longitudinal speeds. Although there are some discrepancies in the absolute values, the rankings of the performance in the real world agree with the simulation, indicating that our proposed method is not only able to identify designs with different locomotion specialties but also provides a directly deployable locomotion policy that functions on the real legs.

This is also confirmed by the best average values of longitudinal speed, turning speed, and COT for the three leg designs listed in Table 5. The values are based on the same experimental data. Take $|\bar{w}_z|^{max}$ of $\boldsymbol{d_{w_z}}$ as an example, the maximum absolute steady-state turning speed averaged over the three trials for each velocity command was first calculated. Then, the maximum value among them $|\bar{w}_z|^{max}$ and the corresponding absolute values of the command $|\boldsymbol{v^{cmd}}|$ were listed in the table. The remaining 8 values in the table were acquired similarly. The commands are not necessarily the same for different metrics.

As shown in the table, $\boldsymbol{d_{v_x}}$ and $\boldsymbol{d_{COT}}$ achieved the highest longitudinal speed and lowest COT respectively, aligning with the selection intention. The maximum speed of our robot is around 0.7 m/s or 5.4 body lengths per second, while the lowest COT is 0.31.

For $\boldsymbol{d_{w_z}}$, although $\boldsymbol{d_{v_x}}$ and $\boldsymbol{d_{COT}}$ do beat it in terms of the achieved highest turning speed, they have much larger tracking errors. Given that the evaluation and selection metric in (3) only sums up all the trackable turning speeds, requiring both fast speed and small tracking error, this result is still considered relatively aligned with the selection. We do want to acknowledge that we did not observe the high turning speeds for $\boldsymbol{d_{v_x}}$ and $\boldsymbol{d_{COT}}$ in the simulation, indicating that there are some gaps between the simulation and the real world.

Table 5. Best real-world performance of the selected leg designs

$\boldsymbol{d}$	$\lvert\bar{v}_x\rvert^{max}$, $\lvert\boldsymbol{v^{cmd}}\rvert$	$\lvert\bar{w}_z\rvert^{max}$, $\lvert\boldsymbol{v^{cmd}}\rvert$	COT^{min}, $\lvert\boldsymbol{v^{cmd}}\rvert$
$\boldsymbol{d_{v_x}}$	**0.70, [0.8, 0.0]**	0.90, [0.8, 0.4]	0.40, [0.8, 0.0]
$\boldsymbol{d_{w_z}}$	0.52, [0.8, 0.0]	**0.45, [0.6, 0.4]**	0.52, [0.8, 0.0]
$\boldsymbol{d_{COT}}$	0.65, [0.8, 0.0]	1.06, [0.8, 0.4]	**0.31, [0.8, 0.0]**

We believe that the velocity performance gap between the simulation and the real environment can be mostly attributed to the backlash of the laminate legs and servo output shafts, leading to more vibrations of the robot body. There is significant play along the non-axial directions of the servo output shafts, which is mostly due to the wear and tear of the plastic components over time. Although it is possible to model this behavior, adding a bearing to support the output shaft properly would be preferred in the future. It will add some weight and friction to the system, but it should, at the same time, improve predictability and durability. Moreover, if higher quality servos were used, this part of the backlash may be negligible.

Backlash within the laminate leg is more difficult to mitigate physically, due to the nature of the flexure joints and thin beams. In this case, measuring the actual force and displacement data along both the intended deformation direction and unintended directions and modeling them in simulation is more desirable in the future. Generally, backlash can be modeled as additional degrees of freedom with compliance, damping, and joint limits. For MuJoCo, achieving these is relatively straightforward; though we initially included basklash early in this project, we ultimately didn't retain it to prioritize simulation speed and model simplicity.

Another challenging aspect of modeling backlash for the proposed system is its uncertainty. Different legs may behave differently due to uneven wear and tear and manufacturing variations. In this case, adding randomization to the simulation and learning an online system identification module for the controller should also further improve the overall performance.

There are also some other potential gaps in addition to the backlash. The contact friction between the feet and floor is not exactly isotropic, and some stick and slip transitions also exist, which are difficult to model accurately in simulation. The COT mismatch may come from the inaccurate estimate of the motor torque from our lumped servo model. The torque measurement on the real servo based on the current reading and a fixed motor constant from the datasheet may also introduce some errors. In addition, our previous work, Chen and Aukes (2023), shows that the PRBM is only an approximation of the leg stiffness because of fabrication variations and nonideal behaviors of the laminate links and joints. We plan to reduce these gaps with better models and more system calibrations in the future.

8.4 Real-World Demonstrations

To demonstrate that the robot and its learned locomotion policy are relatively robust, we tested our robot with all three leg designs in less controlled environments. As shown in Fig. 10 and the supplementary video, the robot was given remote control commands to traverse various terrains, including rocky or smooth concrete, indoor flooring, and sandy soil. The robot is able to maintain balance and even handle small slopes, drops, and gaps. We observed that the robot's velocity varies on different surfaces due to slippage and stumbling over small obstacles. The fast turning design is also more susceptible to getting trapped in dirt and rocks because of its short leg length. The low COT design has a very high center of mass and tends to tip over when passing through a large slope or drop. These findings showcase the importance of fast and accessible prototyping of physical robots, allowing them to be tested in the real world against scenarios that are difficult to model and capture for revising designs and performance metrics.

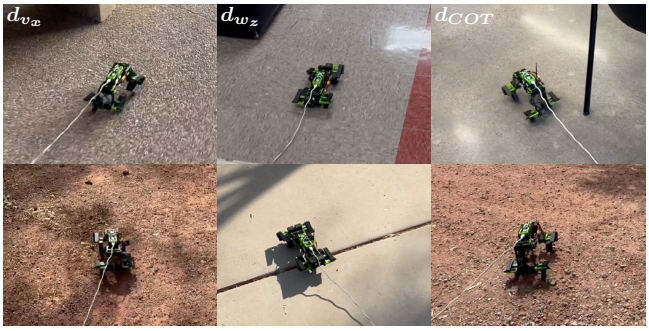

Figure 10. The robot traversing different terrains with the three standout leg designs.

8.5 Handling Novel Applications

Though we selected three standout members based on common metrics for benchmarking quadruped robots, our design space also includes leg designs that may be useful for other specific or niche applications, which can be exposed by altering our performance metrics. For example, the leg with the shortest height $\boldsymbol{d_h}$ makes the robot more suitable for crawling through low-clearance places. The leg with the longest foot travel $\boldsymbol{d_l}$ enables the robot to step over tall obstacles and may be desirable for rough terrains. The leg with the highest parallel and series stiffness $\boldsymbol{d_k}$ is potentially better at carrying heavy payloads. Their corresponding robots in the simulation are shown in Fig. 4.

9 Conclusions

This paper introduces a quadruped robot with tunable laminate legs. We propose methods to identify all its feasible leg designs, train a locomotion controller that exploits each design's unique properties, and curate standout designs for specific applications. We leverage laminate design and fabrication techniques to make low-cost, fast-to-fabricate, and highly tunable quadruped leg designs, simplifying the

process of bringing robots from simulation to the real world and aligning simulation with real-world data. This digital twin of the robot, combined with the adaptability, robustness, and creativity of reinforcement learning, allows us to tackle the co-design problem with more realistic models and richer design space. The generated designs directly work in the real world and exhibit expected locomotion performance characteristics. The curation methods and analysis tools presented take a different approach from the literature, emphasizing visualization and interpretation of the connections between design parameters and performance.

Our proposed methods simplify robot customization, expand design freedom, and encourage exploration and discovery of design principles. This demonstrates the benefits and importance of a tight integration of design, fabrication, simulation, and control. We believe this concept of streamlining the creation of robots is the key to more accessible, diverse, and performant robots around us.

This work also opens up many research directions. An immediate extension of this work based on the same quadruped platform is studying how other performance metrics are related to design parameters. For example, we plan to study how the leg transmission ratio and parallel stiffness affect the robot's ability to exert forces and manipulate objects. One limitation of this work is that to focus on constructing the pipeline and validating the concept, we restrict ourselves to only varying the leg design and using a SLIP-inspired template for a reasonably sized design space that contains mostly functional designs. In the future, we plan to increase the robot's design space by loosening the constraints and adding new parameters to explore novel designs and study less understood questions, such as the benefits of having different front and rear legs and a compliant spine for quadruped robots.

For more complicated designs, the main components of the pipeline will stay similar, including the generation of diverse designs from interesting design parameters, training a unified design-aware controller, performance evaluation, trends discovery, and real-world validation. We anticipate that one of the biggest challenges is how to efficiently explore a higher-dimensional design space. Even though the algorithms proposed in this paper should work with a few modifications, the computation time can quickly increase to an intractable amount. Two approaches to combat the "curse of dimensionality" are proposed. First, we could leverage GPU acceleration, optimize code implementation, and even switch to better and faster computation hardware. Second, we could make the process iterative by allowing designers to zoom in or out on the design space. The resolution of the design space could be adjusted by changing the step size s_D for spanning the design space. Boundaries could also be enforced by checking if the valid neighbors are within the desired range in Algorithm 1. In this way, the designers could first explore the design space coarsely and only perform detailed analysis on focused regions.

Besides the design space, other aspects of the pipeline would also need attention. The complexity of the neural network control policy should be increased to handle more diverse designs. The trends discovery process may need to be automated with principal component analysis or more advanced statistical analysis methods. Ensuring manufacturable designs and the generation of the fabrication plan will require more care. The gaps between the simulation and the real world should also be narrowed with real-world data.

Our long-term goal is to extend this design pipeline beyond quadruped and legged robots, bringing the benefits of complex but well-engineered compliance, kinematics, and dynamics to a broader range of robotic devices such as underwater or flying robots, grippers, and manipulators.

Acknowledgements

The authors would like to thank Weijia Tao for helping to take the photos and videos of this paper.

Funding

This material is based upon work supported by the National Science Foundation under Grant No. 1944789.

References

Aractingi M, Léziart PA, Flayols T, Perez J, Silander T and Souères P (2023) Controlling the Solo12 quadruped robot with deep reinforcement learning. *Scientific Reports* 13(1): 11945. DOI: 10.1038/s41598-023-38259-7. Number: 1 Publisher: Nature Publishing Group.

Badri-Spröwitz A, Aghamaleki Sarvestani A, Sitti M and Daley MA (2022) BirdBot achieves energy-efficient gait with minimal control using avian-inspired leg clutching. *Science Robotics* 7(64): eabg4055. DOI:10.1126/scirobotics.abg4055.

Belmonte-Baeza Á, Lee J, Valsecchi G and Hutter M (2022) Meta Reinforcement Learning for Optimal Design of Legged Robots. *IEEE Robotics and Automation Letters* 7(4): 12134–12141. DOI:10.1109/LRA.2022.3211785.

Bjelonic F, Lee J, Arm P, Sako D, Tateo D, Peters J and Hutter M (2023) Learning-Based Design and Control for Quadrupedal Robots With Parallel-Elastic Actuators. *IEEE Robotics and Automation Letters* 8(3): 1611–1618. DOI:10.1109/LRA.2023.3234809.

Boston Dynamics (2024) Spot. URL https://bostondynamics.com/products/spot/.

Chadwick M, Kolvenbach H, Dubois F, Lau HF and Hutter M (2020) Vitruvio: An Open-Source Leg Design Optimization Toolbox for Walking Robots. *IEEE Robotics and Automation Letters* 5(4): 6318–6325. DOI:10.1109/LRA.2020.3013913. Conference Name: IEEE Robotics and Automation Letters.

Chen F and Aukes DM (2023) Direct Encoding of Tunable Stiffness Into an Origami-Inspired Jumping Robot Leg. *Journal of Mechanisms and Robotics* 16(031012). DOI:10.1115/1.4056958.

Chen F, Tao W and Aukes DM (2023) Development of A Dynamic Quadruped with Tunable, Compliant Legs. In: *2023 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. Detroit, MI, USA: IEEE. ISBN 978-1-66549-190-7, pp. 495–502. DOI:10.1109/IROS55552.2023.10342283.

Dinev T, Mastalli C, Ivan V, Tonneau S and Vijayakumar S (2022) A Versatile Co-Design Approach For Dynamic Legged Robots. In: *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. pp. 10343–10349. DOI:10.1109/IROS47612.2022.9981378. ISSN: 2153-0866.

Galloway KC, Clark JE, Yim M and Koditschek DE (2011) Experimental investigations into the role of passive variable compliant legs for dynamic robotic locomotion. In: *2011 IEEE International Conference on Robotics and Automation*. Shanghai, China: IEEE. ISBN 978-1-61284-386-5, pp. 1243–1249. DOI:10.1109/ICRA.2011.5979941.

Geyer H, Seyfarth A and Blickhan R (2006) Compliant leg behaviour explains basic dynamics of walking and running. *Proceedings of the Royal Society B: Biological Sciences* 273(1603): 2861–2867. DOI:10.1098/rspb.2006.3637. Publisher: Royal Society.

Ghigliazza RM, Altendorfer R, Holmes P and Koditschek D (2003) A Simply Stabilized Running Model. *SIAM Journal on Applied Dynamical Systems* 2(2): 187–218. DOI:10.1137/S1111111102408311.

Grimminger F, Meduri A, Khadiv M, Viereck J, Wuthrich M, Naveau M, Berenz V, Heim S, Widmaier F, Flayols T, Fiene J, Badri-Sprowitz A and Righetti L (2020) An Open Torque-Controlled Modular Robot Architecture for Legged Locomotion Research. *IEEE Robotics and Automation Letters* 5(2): 3650–3657. DOI:10.1109/LRA.2020.2976639.

Haldane DW, Plecnik MM, Yim JK and Fearing RS (2016) Robotic vertical jumping agility via series-elastic power modulation. *Science Robotics* 1(1): eaag2048. DOI:10.1126/scirobotics.aag2048.

Huang S, Dossa RFJ, Ye C, Braga J, Chakraborty D, Mehta K and Araújo JGM (2022) CleanRL: High-quality Single-file Implementations of Deep Reinforcement Learning Algorithms. *Journal of Machine Learning Research* 23(274): 1–18.

Hubicki C, Grimes J, Jones M, Renjewski D, Spröwitz A, Abate A and Hurst J (2016) ATRIAS: Design and validation of a tether-free 3D-capable spring-mass bipedal robot. *The International Journal of Robotics Research* 35(12): 1497–1521. DOI:10.1177/0278364916648388.

Hutter M, Gehring C, Jud D, Lauber A, Bellicoso CD, Tsounis V, Hwangbo J, Bodie K, Fankhauser P, Bloesch M, Diethelm R, Bachmann S, Melzer A and Hoepflinger M (2016) ANYmal - a highly mobile and dynamic quadrupedal robot. In: *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. Daejeon, South Korea: IEEE. ISBN 978-1-5090-3762-9, pp. 38–44. DOI:10.1109/IROS.2016.7758092.

Katz B, Carlo JD and Kim S (2019) Mini Cheetah: A Platform for Pushing the Limits of Dynamic Quadruped Control. In: *2019 International Conference on Robotics and Automation (ICRA)*. Montreal, QC, Canada: IEEE. ISBN 978-1-5386-6027-0, pp. 6295–6301. DOI:10.1109/ICRA.2019.8793865.

Lee J, Hwangbo J, Wellhausen L, Koltun V and Hutter M (2020) Learning quadrupedal locomotion over challenging terrain. *Science Robotics* 5(47): eabc5986. DOI:10.1126/scirobotics.abc5986.

Margolis GB, Yang G, Paigwar K, Chen T and Agrawal P (2024) Rapid locomotion via reinforcement learning. *The International Journal of Robotics Research* 43(4): 572–587. DOI:10.1177/02783649231224053. Publisher: SAGE Publications Ltd STM.

Newman WM and Sproull RF (1979) *Principles of interactive computer graphics*. 2nd ed. edition. USA: McGraw-Hill, Inc. ISBN 0-07-046338-7.

Pardo F, Tavakoli A, Levdik V and Kormushev P (2018) Time Limits in Reinforcement Learning. In: *Proceedings of the 35th International Conference on Machine Learning*. PMLR, pp. 4045–4054. ISSN: 2640-3498.

Pontzer H (2007) Effective limb length and the scaling of locomotor cost in terrestrial animals. *Journal of Experimental Biology* 210(10): 1752–1761. DOI:10.1242/jeb.002246.

Raffin A, Seidel D, Kober J, Albu-Schäffer A, Silvério J and Stulp F (2022) Learning to Exploit Elastic Actuators for Quadruped Locomotion DOI:10.48550/ARXIV.2209.07171. Publisher: arXiv Version Number: 1.

Saranli U, Buehler M and Koditschek DE (2001) RHex: A Simple and Highly Mobile Hexapod Robot. *The International Journal of Robotics Research* 20(7): 616–631. DOI:10.1177/02783640122067570.

Schulman J, Wolski F, Dhariwal P, Radford A and Klimov O (2017) Proximal Policy Optimization Algorithms. DOI:10.48550/arXiv.1707.06347. ArXiv:1707.06347 [cs].

Spröwitz A, Tuleu A, Vespignani M, Ajallooeian M, Badri E and Ijspeert AJ (2013) Towards dynamic trot gait locomotion: Design, control, and experiments with Cheetah-cub, a compliant quadruped robot. *The International Journal of Robotics Research* 32(8): 932–950. DOI:10.1177/0278364913489205.

Storn R and Price K (1997) Differential Evolution – A Simple and Efficient Heuristic for global Optimization over Continuous Spaces. *Journal of Global Optimization* 11(4): 341–359. DOI:10.1023/A:1008202821328.

Unitree (2024) Unitree A1. URL https://www.unitree.com/a1/.

Appendix

A Cost of the Prototype

The main items that make up the material cost of the prototype are listed in Table 6. The prices for the parts in US dollars were taken on Dec 23rd, 2024, from major vendors in the United States and did not include tax and shipping costs. The materials needed for making the laminate legs are based on the low cost of transport design $\boldsymbol{d_{COT}}$, the largest one fabricated in this paper. The estimated amount of the sheet materials includes necessary supporting structures. The 3D-printed parts are only based on their weights. The cost for the miscellaneous items is a rough estimate. The exact material cost of the prototype will vary depending on the location, vendors, time, etc.

For the proposed prototype, the structures that are mostly the legs is only 2.6% of the total cost. If only the laminate legs are accounted for, the value is around 2%. At the same time, as shown in our paper, the design of the legs affects the robot's performance for different tasks. Therefore, this further shows that providing multiple leg designs for a single robot body can be a cost-effective and performant way to tailor quadrupeds for different applications.

The labor and machine cost of the prototype are omitted because they depend on factors that are difficult to capture in a lab setting. Instead, we only provide some fabrication statistics here. It takes $\sim$2 hours to cut out all four laminate legs for a single design. The 3D-printed parts take $\sim$6 hours and do not need to be reprinted when changing the leg design. The assembly time of the legs is $\sim$2 hours for an experienced

Table 6. Estimation of the prototype's material cost in US dollars

Part	Details	Unit cost	Quantity	Total cost	Category	%
Servos	ROBOTIS XC330-T181-T	89.90	8.0000	719.20	Servos	88.2
MCU	ROBOTIS OpenCM9.04-C	23.80	1.0000	23.80		
Battery	TATTU 11.1V 3S 450mAh 75C XT30 plug	15.99	1.0000	15.99	Electronics	9.2
IMU	Adafruit BNO055	34.95	1.0000	34.95		
Fiberglass	ACP Composites FSB-015-05-P 0.015" 36" x 48"	105.45	0.030	3.15		
Fiberglass	ACP Composites FSB-030-05-P 0.030" 36" x 48"	94.55	0.030	2.83		
Fiberglass	ACP Composites FSB-063-05-P 0.063" 36" x 48"	138.80	0.052	7.24		
Polyester	Grafix Dura-Lar 0.005" 40" x 25'	54.10	0.004	0.23	Structures	2.6
Adhesive	DRYTAC MHA25328 25.5" x 328'	436.49	0.002	0.67		
3D printed	Ultimaker PLA 2.85mm 750g	58.00	0.043	2.49		
Screws, staples, wires, zip ties, friction tape, etc.				5.00		
			Total	815.55		

person and the assembly of the rest of the robot needs another $\sim$2 hours. Swapping all four legs from one design to another takes $\sim$0.5 hours. The laser cutter used is an Epilog Fusion M2 40. The 3D printer is a Ultimaker 3.