MITSUBISHI ELECTRIC RESEARCH LABORATORIES
https://www.merl.com

MOBIUS: A Multi-Modal Bipedal Robot that can Walk, Crawl, Climb, and Roll

Schperberg, Alexander; Tanaka, Yusuke; Di Cairano, Stefano; Hong, Dennis

TR2026-110 July 14, 2026

Abstract

This paper presents the MOBIUS platform, a bipedal robot capable of walking, crawling, climbing, and rolling. MOBIUS features four limbs, two 6-DoF arms with two-finger grippers for manipulation and climbing, and two 4-DoF legs for locomotion–enabling smooth transitions across diverse terrains without reconfiguration. A hybrid control architecture combines reinforcement learning for locomotion and admittance control enhanced for safety by a Reference Governor and auto-tuning toward compliant contact interactions during manipulation. A high-level MIQCP planner autonomously selects locomotion modes to balance stability and energy efficiency. Hardware experiments demonstrate robust gait transitions, dynamic climbing, and full-body load support via pinch grasp. Overall, MOBIUS demonstrates the importance of tight integration between morphology, high-level planning, and control to enable mobile locomanipulation and grasping, substantially expanding its interaction capabilities, workspace, and traversability.

Robotics Science and Systems 2026

Mitsubishi Electric Research Laboratories, Inc.
201 Broadway, Cambridge, Massachusetts 02139

MOBIUS: A Multi-Modal Bipedal Robot that can Walk, Crawl, Climb, and Roll

Alexander Schperberg[1*], Yusuke Tanaka[2*], Stefano Di Cairano[1] and Dennis Hong[3]

***Abstract*—This paper presents the MOBIUS platform, a bipedal robot capable of walking, crawling, climbing, and rolling. MOBIUS features four limbs, two 6-DoF arms with two-finger grippers for manipulation and climbing, and two 4-DoF legs for locomotion–enabling smooth transitions across diverse terrains without reconfiguration. A hybrid control architecture combines reinforcement learning for locomotion and admittance control enhanced for safety by a Reference Governor and auto-tuning toward compliant contact interactions during manipulation. A high-level MIQCP planner autonomously selects locomotion modes to balance stability and energy efficiency. Hardware experiments demonstrate robust gait transitions, dynamic climbing, and full-body load support via pinch grasp. Overall, MOBIUS demonstrates the importance of tight integration between morphology, high-level planning, and control to enable mobile loco-manipulation and grasping, substantially expanding its interaction capabilities, workspace, and traversability.**

I. Introduction

Legged robots offer strong versatility for navigating complex, human-made environments, motivating applications in search and rescue, inspection, and exploration. Prior work has produced highly capable but largely specialized platforms, including bipeds (Feng et al. [13], Gong et al. [17], Castillo et al. [8]), quadrupeds (Bellegarda et al. [4], Hutter et al. [20], Seok et al. [36], Bouman et al. [6]), and hopping or jumping robots (Raibert et al. [30], Yim and Fearing [46]). Parallel advances in manipulation have enabled loco-grasping behaviors (Shi et al. [37], Fu et al. [14], Su et al. [38], Gong et al. [16]), but only a small number of systems address the extreme case of free climbing in discrete, vertical environments. Most climbing robots rely on adhesion (Kim et al. [22], Saunders et al. [32], Austin et al. [2]) or passive spine-based grasping (Parness et al. [26], Uno et al. [43], Nagaoka et al. [25]), which limits dexterity, and dynamic climbing. While SCALER (Tanaka et al. [40]) demonstrates grasping-based climbing under Earth gravity, it sacrifices general ground mobility. More broadly, existing legged robots typically excel in a single locomotion regime, limiting adaptability across continuous and discrete terrain.

[1]A. Schperberg and S. Di Cairano are with Mitsubishi Electric Research Laboratories (MERL), Cambridge, USA (email: schperberg@merl.com, dicairano@ieee.org).

[2]Y. Tanaka is with the Robotic Systems Lab, ETH Zurich, Zurich, Switzerland (email: yutanaka@ethz.ch).

[3]D. Hong is with the Robotics and Mechanisms Laboratory, Department of Mechanical and Aerospace Engineering, University of California, Los Angeles, Los Angeles, USA (email: dennishong@g.ucla.edu).

[†]Hardware designed and developed at the Robotics and Mechanisms Laboratory.

[*]Denotes equal contribution.

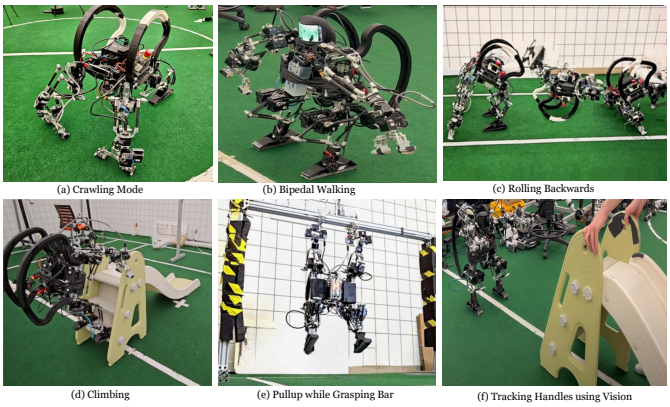


Fig. 1: MOBIUS: **M**ulti-modal **O**perations **B**ipedal **I**ntelligent **U**rban **S**cout.

To address these limitations, we introduce MOBIUS (**M**ulti-modal **O**perations **B**ipedal **I**ntelligent **U**rban **S**cout), a unified platform capable of bipedal walking, crawling, rolling, and dynamic free climbing. MOBIUS transitions seamlessly between modes to balance efficiency, stability, and manipulation. Equipped with two grippers (Tanaka et al. [39]), it can support its full body weight and perform pinching-based pull-ups. To the best of our knowledge, MOBIUS is the first bipedal robot with finger grippers to perform walking, crawling, free-climbing [7], rolling, and pull-ups within a single, non-reconfigurable morphology—requiring no part swapping, appendage attachment/detachment, or changes to hardware topology.

Overall, we provide the following **contributions**:

1) A novel multi-modal robot combining two 6-DoF dexterous arms with two-finger grippers and two 4-DoF legs, with integrated rails for fall protection and rolling.
2) A unified morphology enabling bipedal walking, crawling, rolling, vertical free climbing, and pull-up maneuvers under Earth gravity, with extensive hardware validation.
3) A system-level software stack consisting of an adaptive and safety-enforced admittance control for contact-rich tasks, reinforcement learning for locomotion, and a MIQCP-based high-level planner for mode selection.

A. Paper Organization

The paper is organized as follows. Related works on multi-modal robots, covering both hardware and software aspects, are presented in Sec. II. The MOBIUS hardware is described in Sec. III, including design challenges such as multi-modality requirements, performance–robustness trade-offs, and integration and scalability considerations. Further details on the

limb and gripper design, actuator selection, fatigue analysis, and reliability are left as Supplementary Material (Sec. A of Appendix). A summary of locomotion modes is given in Sec. III-C. The software architecture (Fig. 4) covers RL-based planning and control (Sec. IV-A), where details on the model-based MPC baseline (used to compare against RL) is given in Sec. C of Appendix. Additional modules include the force controller (Sec. IV-B), and high-level multimodal planner (Sec. IV-C). Other modules such as our state estimator and visual-servo controller are in the Supplementary Material (Sec. G and Sec. H of Appendix respectively). Finally, Sec. V presents simulation and hardware integration and extensive experimental results on locomotion, climbing, pull-ups, force control, comparisons between MPC and RL, multimodal planning, and visual-servo tracking.

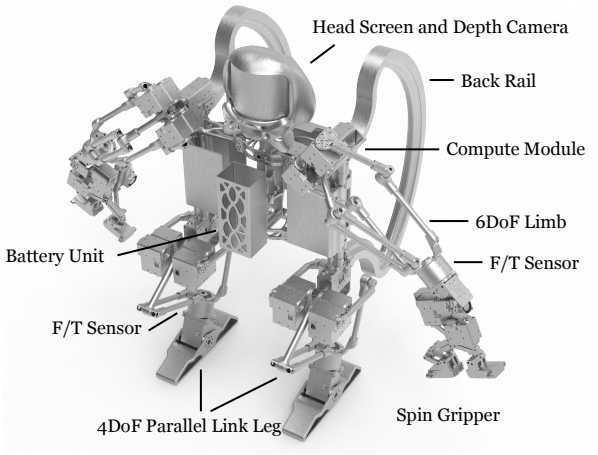


Fig. 2: MOBIUS overall structure rendering.

II. Related Work

We review prior work from two perspectives: (i) hardware designs enabling multi-modal locomotion in Sec. II-1, and (ii) planning and control architectures for coordinating multiple locomotion modes in Sec. II-2.

1) Hardware for Multi-Modality: Multi-modal robots leverage multiple locomotion strategies within a single platform to improve adaptability (Xu et al. [45], Baines et al. [3]). Harpy (Dangol et al. [11]) and LEONARDO (Kim et al. [21]) combine bipedal walking with torso-mounted thrusters to overcome obstacles via short-duration flight. AuxBots (Chin et al. [10]) achieve flipper-style locomotion through modular, volume-changing assemblies, while origami-inspired robots (Chen et al. [9]) use shape morphing and variable stiffness for transitions between rigid and soft locomotion. ALPHRED (Hooks et al. [19]) employs a radially symmetric quadrupedal morphology to enable simultaneous locomotion and dual-arm manipulation. RiSE (Saunders et al. [32]) uses compliant legs with micro-spine feet for vertical climbing on rough surfaces. ANYmal-Wheel (Bjelonic et al. [5]) integrates legged and wheeled locomotion for efficient mode switching, and GOAT-SR (Polzin et al. [28]) supports driving, rolling, and swimming via active and passive deformation. The soft wheel-leg robot (Ai et al. [1]) combines pneumatic actuation and wheel-like appendages to traverse constrained environments through rolling and crawling. Most prior systems rely on mechanical reconfiguration or mode-specific appendages, increasing complexity and mass. In contrast, MOBIUS achieves walking, crawling, climbing, and rolling with a single unified morphology. Its limbs equipped with two-finger spine-based grippers (Tanaka et al. [39]) serve both as compliant contacts and high-force grasping tools. While pull-up behaviors have been demonstrated on limited platforms such as ARMStrong Dex (Korea Atomic Energy Research Institute [23]), MOBIUS is the first mid- to large-scale biped to perform a pinch-grasp pull-up from a full dead-hang configuration.

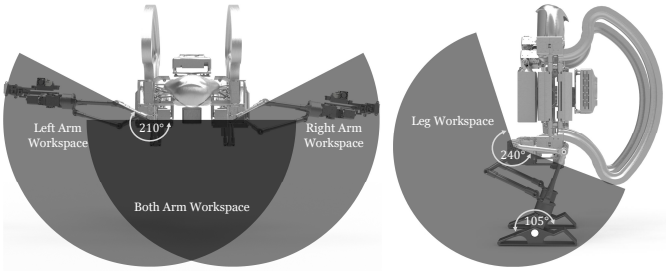
2) Multi-Modal Planning and Control: Coordinating multiple locomotion modes require control frameworks that handle discontinuous dynamics and varying contact conditions. Harpy (Dangol et al. [11]) regulates ground reaction forces via thrusters, while AuxBots (Chin et al. [10]) control inter-module distances for efficient flipper motion. Origami-based robots (Chen et al. [9]) analytically compute variable-stiffness joint configurations to switch between walking, crawling, and grasping. RiSE (Saunders et al. [32]) adapts climbing and walking behaviors using distributed sensing and contact feedback. ANYmal-Wheel (Bjelonic et al. [5]) employs hierarchical whole-body control with ZMP-based optimization and nonholonomic rolling constraints. The soft wheel-leg robot (Ai et al. [1]) coordinates pneumatic and wheel actuation to switch between crawling strategies. Hybrid learning-based approaches such as from Yu and Rosendo [47] use Automated Residual Reinforcement Learning to transition between quadruped and bipedal gaits, combining classical control with model-free RL methods including SAC and TD3.

Building on these ideas, MOBIUS employs a unified hybrid control stack. Model-free RL governs bipedal and crawling behaviors, while a model-based approach is used for contact-rich tasks through combining an adaptive admittance control with a Reference Governor (Garone et al. [15]) for safety. A high-level MIQCP planner autonomously selects locomotion modes to minimize energy subject to reachability and terrain constraints. This integration enables autonomous transitions across modes without manual triggering or hardware reconfiguration.

III. Hardware and System Mechanisms

A. Design Objectives and Challenges

MOBIUS builds upon the mechanism design from Tanaka et al. [40] to support fundamentally different locomotion regimes, including bipedal walking, crawling, rolling, and vertical climbing. Unlike that mechanisms design, which is optimized for climbing, MOBIUS must support its full body weight on a single limb during bipedal walking while maintaining sufficient swing velocity and stability. This imposes conflicting requirements on kinematics, force capability, and actuator selection. To address this, MOBIUS adopts limb kinematics that balance velocity and force isotropy at nominal bipedal and crawling configurations, while retaining dexterous arm linkage for climbing and manipulation. The gripper serves



(a) MOBIUS limb workspace in the top view with a 210° range.

(b) MOBIUS leg workspace in the side view with 105° foot pitch.

Fig. 3: Kinematic ranges of the MOBIUS limb modules.

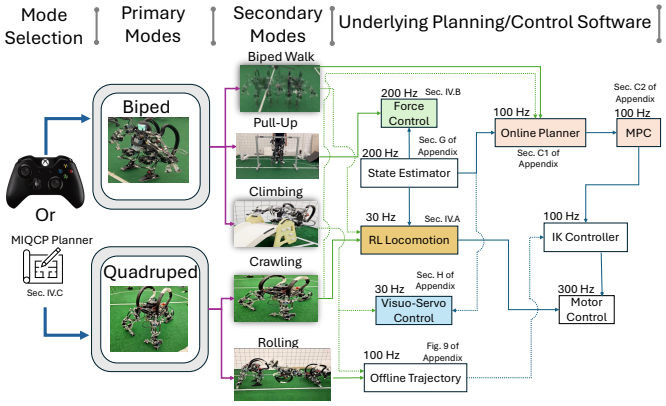


Fig. 4: We demonstrate the overall flowchart, from user primary mode selection, to the secondary modes associated with each primary mode. The modes can be selected either by the user through a joystick, or autonomously, if given a 2D map of the environment with relevant features, using our high level MILP-based planner (see Sec. IV-C). The underlying planners and controllers corresponding to each secondary mode are shown on the right side of the figure, along with their operating frequencies and references to the sections of this work where each module is discussed.

as both a dexterous end-effector and a load-bearing contact during crawling, enabling seamless transitions between manipulation and locomotion without mechanical reconfiguration.

B. Mechanical Architecture Overview

MOBIUS employs two 6-DoF arms with two-finger spine-based grippers and two 4-DoF legs with flat feet (Fig. 2). MOBIUS hybrid parallel mechanism linkages help to withstand repeated impact loads during bipedal locomotion while being capable of power-intensive and dexterous contact-rich climbing. Curved back rails passively guide the robot into recoverable postures following backward falls, reducing control and workspace requirements (Fig. 3). Further details on limb kinematics and Jacobian properties, fatigue analysis, and end-effector design choice is detailed in Sec. A of Appendix. The total weight of the robot is 10.3 kg.

C. Multi-Modal Locomotion Capabilities

MOBIUS supports five primary capabilities: bipedal walking, crawling, rolling, vertical mobility, and autonomous transitions between modes. Each mode offers distinct trade-offs in speed, stability, and energy efficiency, enabling task- and terrain-dependent operation (Fig. 5).

In bipedal mode, MOBIUS walks omnidirectionally while freeing its arms for manipulation or climbing. Crawling uses

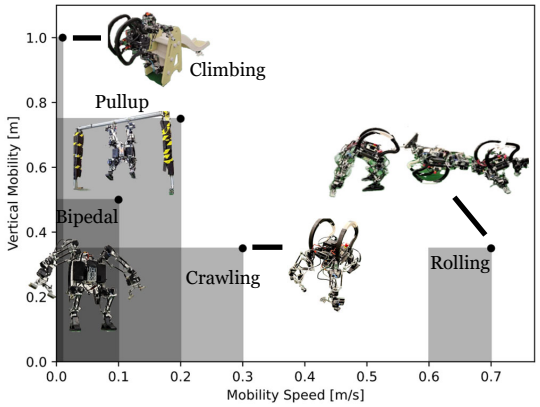


Fig. 5: MOBIUS mobility envelope across locomotion modes.

all four limbs to maximize stability on rough terrain. Back-mounted rails enable backward rolling, providing both fall protection and an energy-efficient locomotion mode. Vertical mobility is achieved through grasping-based climbing and pinch-grasp pull-ups using the grippers, enabling traversal of discrete structures such as ladders and bars. Finally, MOBIUS can transitions between modes, including fall recovery and standing-to-crawling transitions (and vice-versa), and crawling to rolling. We visually show and detail these transition modes in Sec. I, and Fig. 9 of Appendix.

IV. Software Architectures for Planning and Control

The full software architecture and locomotion modes are shown in Fig. 4, with pointers to the relevant subsections describing them. For ablations, we compare a reinforcement learning approach (Sec. IV-A) against a model-based baseline for biped and crawling locomotion (baseline implementation details are in Sec. C of Appendix).

A. Reinforcement Learning for Planning and Control

Compared to conventional bipeds and humanoids, MOBIUS has forearms that are 55% heavier than its lower body. While this asymmetry facilitates transitions between bipedal and crawling modes, it significantly complicates bipedal locomotion due to uneven mass distribution and arm-induced momentum. Although model-based control has proven effective for many bipeds and quadrupeds (Raibert [29], Kuindersma et al. [24]), it is insufficient for MOBIUS due to poorly characterized dynamics. We therefore adopt a fully model-free reinforcement learning (RL) approach, which has shown strong performance in prior work (Tsounis et al. [42]), and avoid reference trajectories or motion priors (Vollenweider et al. [44], Escontrela et al. [12]).

1) Problem Definition: The RL controller enables MOBIUS to track user joystick commands and is primarily used for bipedal locomotion, where stability is most challenging. Crawling is inherently stable and follows commanded velocities using the model-based controller described in Sec. C of Appendix (although we also show comparisons between RL and model-based for crawling in Fig. 8). Locomotion is formulated as a Partially Observable Markov Decision Process

(POMDP), and policies are trained using Proximal Policy Optimization (PPO). Training proceeds in two stages: first on flat terrain, then fine-tuned on rough terrain. Hyperparameters, training durations, domain randomization, and disturbances are detailed in Sec. B of Appendix.

2) Observation and Action: The observation vector is defined as

$$\mathbf{O}_t = [\mathbf{X}_t, \mathbf{\Theta}_t, \mathbf{A}_{t-1}, \mathbf{v}_{xy}^{\text{des}}, \omega_z^{\text{des}}, \mathbf{O}_{t-N:t-1}],$$

where $\mathbf{X}$ represents the estimated trunk state, including orientation, angular velocity, and linear velocity; $\mathbf{\Theta}$ denotes joint encoder values; $\mathbf{A}_{t-1}$ is the previous action; and $\mathbf{v}_{xy}^{\text{des}}$ and ω_z^{des} are desired linear and angular velocities. The action space consists of target joint positions, with dimensions depending on locomotion mode. A history of $N = 15$ observations is included, resulting in observation dimensions of $\mathbb{R}^{416}$ for bipedal mode and $\mathbb{R}^{608}$ for crawling.

Actions are filtered using a moving average to improve smoothness for hardware execution:

$$\mathbf{\Theta}_{\text{action}} = \mathbf{\Theta}_{\text{default}} + \left(\frac{1}{n_{\text{actions}}} \sum_{i=1}^{n_{\text{actions}}} \mathbf{A}_{t-i} \right) a_{\text{scale}}, \quad (1)$$

where $\mathbf{\Theta}_{\text{default}}$ is the nominal pose and a_{scale} is a scaling factor.

3) Reward Function: The reward function employed balances stability, tracking accuracy, and smoothness. Specifically, it penalizes vertical motion, pitch and roll rates, large action changes, foot slip, and short episodes, while rewarding planar velocity tracking, yaw-rate tracking, standing still at low commanded speeds, and sufficient foot air time to promote proper stepping. See Sec. B of Appendix for details on reward and penalty functions.

B. Grasping Force Control

MOBIUS can lift its full body weight using a two-finger gripper. To keep grasping smooth and safely, both for task execution and under external disturbances (e.g., pushing and pulling), we introduce compliance via force control. We employ an admittance controller that maps measured wrench to a desired motion (change in position). This is suitable for robots with position-controlled actuators that must still behave compliantly during contact. For clarity, we refer to the end-effector as the *fingertip* and the wrist as the *gripper base*.

We build on the auto-tuning admittance framework from Schperberg et al. [33, 35], which adapts controller gains online to track position and wrench profiles, minimize slip, and avoid kinematic singularities. Prior work does not provide formal safety guarantees, so we add a Reference Governor (RG) (Garone et al. [15]) to enforce constraints during force control (Sec. IV-B1). The overall architecture is shown in Fig. 6. Details of the auto-tuning module and ablations are found in Schperberg et al. [33]; we summarize the core controller here.

The admittance dynamics are:

$$\ddot{\mathbf{x}} = \mathbf{M}_d^{-1}\Big(- \mathbf{D}_d \mathbf{V}_{\text{cur}} - \mathbf{K}_d(\mathbf{X}_{\text{cur}} - \mathbf{X}_{\text{ref}}) + \mathbf{K}_f(\mathcal{W}_{\text{meas}} - \mathcal{W}_{\text{ref}})\Big), \quad (2)$$

where $\mathcal{W}_{\text{meas}}, \mathcal{W}_{\text{ref}} \in \mathbb{R}^{6k}$ are measured and desired wrenches at fingertip or contact k, with $\mathcal{W} = [\mathbf{f}^\top, \boldsymbol{\tau}^\top]^\top$ ($\mathbf{f} \in \mathbb{R}^{3k}$ forces and $\boldsymbol{\tau} \in \mathbb{R}^{3k}$ torques). $\mathbf{M}_d$ (invertible), $\mathbf{D}_d$, and $\mathbf{K}_d \in \mathbb{R}^{6k \times 6k}$ are diagonal desired mass, damping, and stiffness matrices, and $\mathbf{K}_f \in \mathbb{R}^{6k \times 6k}$ scales wrench sensitivity. $\mathbf{X}_{\text{cur}}$ and $\mathbf{V}_{\text{cur}}$ are fingertip pose and velocity with $\mathbf{X}_{\text{cur}} = [\mathbf{p}^\top, \mathbf{\Theta}^\top]^\top$, where $\mathbf{p} \in \mathbb{R}^{3k}$ and $\mathbf{\Theta} \in \mathbb{R}^{3k}$ is the fingertip position and orientation. $\mathbf{X}_{\text{ref}}$ is the reference trajectory, and Euler-angle differencing is handled as in Piovan and Bullo [27]. The resulting $\ddot{\mathbf{x}}$ is discretized using Euler integration to obtain $\mathbf{x}$, then mapped to joint commands via inverse kinematics, $f_{\text{kin}}^{-1}(\mathbf{x}) = \theta_{\text{joint angles}}$, and sent to low-level motor controllers.

1) Reference Governor for Safety Guarantees: Force control must respect state and control limits while maintaining performance. We enforce safety using a Reference Governor (RG), an add-on module that modifies the commanded reference in real time to avoid constraint violations while remaining as close as possible to the original input. The RG is based on the Maximal Output Admissible Set (MOAS), defined as the set of initial states and references for which constraints are never violated:

$$\mathcal{O}_\infty = \{(x_0, v) \mid y(t) \in \mathcal{Y} \ \forall t \geq 0\}, \quad (3)$$

where x_0 is the initial state, v the reference, $y(t)$ the system output, and $\mathcal{Y}$ the admissible output set determined by safety and performance limits.

2) Control Law for Reference Governor: The RG feasibility check uses the closed-loop control law

$$\mathbf{u}(t) = \ddot{\mathbf{x}} - \mathbf{K}_{I_x}\mathbf{z}_x - \mathbf{K}_{I_f}\mathbf{z}_f, \quad (4)$$

where $\ddot{\mathbf{x}}$ is from (2), $\mathbf{z}_x, \mathbf{z}_f$ are integral error terms for position and wrench, and $\mathbf{K}_{I_x}, \mathbf{K}_{I_f}$ are integral gains that promote smoothness. To prevent windup under saturation or large disturbances, we reset integrators when acceleration exceeds bounds:

$$\text{if } |\ddot{\mathbf{x}}| > \ddot{\mathbf{x}}_{\max} \Rightarrow \mathbf{z}_x = \mathbf{z}_f = 0.$$

We compute $\mathcal{O}_\infty$ offline by simulating trajectories over a finite horizon using second-order Euler integration. For each sampled $(\mathbf{x}_{\text{cur}}, \mathbf{v}_{\text{cur}}, \mathbf{x}_{\text{ref}}, \mathbf{W}_{\text{cur}}, \mathbf{W}_{\text{ref}})$, we roll out (4). Samples whose trajectories satisfy position, velocity, and wrench constraints are retained in $\mathcal{O}_\infty$. We apply this per axis (x, y, z) for position, velocity, and force, and include torque about the gripper-base normal. Full algorithmic details are in Sec. D of Appendix.

3) Employing Reference Governor: Given $\mathcal{O}_\infty$, the RG modifies the reference setpoint at the current state so that a feasible control sequence exists and constraints are satisfied for all future time. To enable fast online checks, we store discrete samples of $\mathcal{O}_\infty$ in a KD-Tree for logarithmic-time nearest-neighbor queries. For a query o_q, if $o_q \notin \mathcal{O}_\infty$, we find the closest admissible point

$$o^* = \arg\min_{o \in \mathcal{O}_\infty} \|o - o_q\|, \quad (5)$$

with $o = [\mathbf{x}, \mathbf{v}, \mathbf{x}_{\text{ref}}, \mathbf{W}, \mathbf{W}_{\text{ref}}]$. We keep $(\mathbf{x}, \mathbf{v}, \mathbf{W})$ fixed and replace the references with $(\mathbf{x}_{\text{ref}}^*, \mathbf{W}_{\text{ref}}^*)$ so the system is steered back into the admissible set.

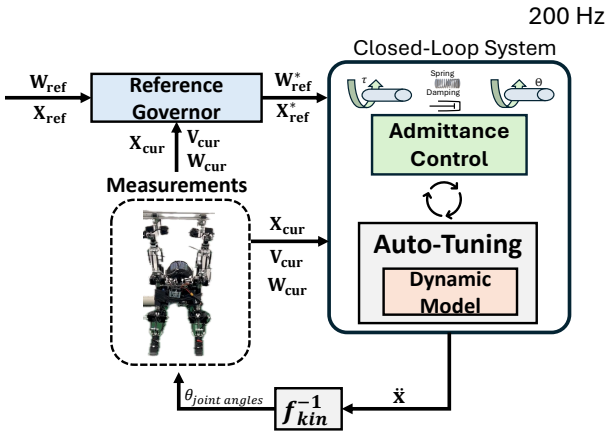


Fig. 6: The force controller used during the pull-up mode is demonstrated. It consists of an admittance controller where the gains are adapted during online operation as done in Schperberg et al. [33]. To promote safety guarantees during force control, we expand that work with the addition of a Reference Governor (RG), outlined in Sec. IV-B1. The RG takes as input the current state, and desired reference, and outputs a modified reference (if necessary) to steer the system into regions of stability based on the maximal output admissible set.

C. High-Level Multi-Modal Planning

We present a high-level planner that lets a multi-modal robot optimize its trajectory while switching modes online. The problem is formulated as a Mixed Integer Quadratic Constraint Programming (MIQCP) problem, enabling joint optimization of grid-based path planning and discrete mode selection. Note that we chose MIQCP over simpler planners to encode terrain, robot state, and locomotion mode coupling constraints. The objective balances exploration, goal reaching within a time horizon, and energy or stability costs, while terrain and obstacle constraints ensure safe, feasible navigation.

1) MIQCP and Mixed Integer Formulation: MOBIUS can transition between crawling, biped, and rolling to satisfy task demands. Our MIQCP planner exploits this by selecting both a mode sequence and a trajectory, using big-M to handle *if-then* logic. The task is to move from a start to a goal within T steps, avoid obstacles, and maximize exploration of a discretized $N_{\text{grid}} \times N_{\text{grid}}$ 2D map. While the task is solvable in a single mode, multi-modal planning yields higher coverage and efficiency (Sec. V-E).

The robot state on the grid is represented by variables $x(t,0)$ and $x(t,1)$ for planar position, and binary variables $V(i,j)$ encode visited cells and modes. Movement propagation depends on the active mode, with per-mode step sets for $d(t)$, and a one-hot constraint m enforces a single mode at each time. Full variable definitions, constraint interpretations, and analysis on hyper-parameter selections are given in Sec. E of Appendix.

2) Terrain and Obstacle Handling: We incorporate terrain-dependent mode constraints and obstacle avoidance into the MIQCP. Terrain regions are modeled as circles that activate binary indicators when the robot is inside them, and these indicators restrict allowable modes (e.g., crawling on rough terrain, biped on elevated-visibility regions). Obstacles are modeled as rectangles that the robot is forbidden to enter. See Sec. E in Appendix for the explicit constraint logic.

3) Objective Functions and Trade-off: The objective trades off exploration against goal reaching, and includes mode penalties to discourage costly modes unless needed:

$$\max_{V_{i,j},x} \quad W_{\text{exp}} \sum_{i=0}^{N_{\text{grid}}-1} \sum_{j=0}^{N_{\text{grid}}-1} V_{i,j} - W_{\text{goal}} \left\| x_T - x_{\text{goal}} \right\|^2 - \sum_{t=0}^{T-1} \sum_{k=1}^{3} P_k \, m_k(t) \quad \text{s.t.} \quad (1)-(20), \text{see Table I.}$$

Here W_{exp} and W_{goal} represent weight exploration and goal tracking, and P_k penalizes mode k. Crawling has the lowest penalty (most stable and efficient), biped is higher, and rolling is highest due to modeling the localization instability. This biases the planner toward crawling by default, biped when required by the task or terrain, and rolling only when large strides are needed to meet the horizon objective.

TABLE I: MIQCP Constraints (1)-(20)

Visiting Grid Constraints	
(1)	$z(t,i,j) \in \{0,1\}$
(2)	$z_{\text{sum}}(i,j) = \sum_{t=0}^{T-1} z(t,i,j), \quad \forall i,j \in \{0,1,\ldots,N_{\text{grid}}-1\}$
(3)	$V(i,j) \leq z_{\text{sum}}(i,j), \quad \forall i,j$
(4)	$V(i,j) \leq T \cdot V(i,j), \quad \forall i,j$
(5)	$x(t,0) - i \geq -M(1 - z(t,i,j))$
(6)	$x(t,1) - j \geq -M(1 - z(t,i,j))$
Propagation Update	
(7)	$x(t+1) = x(t) + d(t)$
Mode Types Constraints	
(8)	$m_1(t) = 1 \implies d(t) \in \{-1,1\}$
(9)	$m_2(t) = 1 \implies d(t) \in \{-2,-1,1,2\}$
(10)	$m_3(t) = 1 \implies d(t) \in \{-3,3\}$
(11)	$m_1(t) + m_2(t) + m_3(t) = 1$
Terrain Type Constraints	
(12)	$d_{\text{circle}} = (x(t,0) - x_{\text{center}})^2 + (x(t,1) - y_{\text{center}})^2$
(13)	$d_{\text{circle}} \leq r_{\text{circle}}^2 + M(1 - b_{\text{circle}}^k(t))$
(14)	$d_{\text{circle}} \geq r_{\text{circle}}^2 + \epsilon - M b_{\text{circle}}^k(t)$
(15)	$m_k(t) \geq b_{\text{circle}}^k(t), \quad b_{\text{circle}}^k \in \{0,1\}$
Obstacle Constraints	
(16)	$x(t,0) \geq x_{\text{min}} + M(1 - b_1^{\text{rect}}(t))$
(17)	$x(t,0) \leq x_{\text{max}} - M(1 - b_2^{\text{rect}}(t))$
(18)	$x(t,1) \geq y_{\text{min}} + M(1 - b_3^{\text{rect}}(t))$
(19)	$x(t,1) \leq y_{\text{max}} - M(1 - b_4^{\text{rect}}(t))$
(20)	$b_1^{\text{rect}}(t) + b_2^{\text{rect}}(t) + b_3^{\text{rect}}(t) + b_4^{\text{rect}}(t) \leq 3$

TABLE II: Maximum Locomotion Velocities

Mode	Biped Mode	Crawling Mode	Rolling Mode
Max velocity (m/s)	0.1	0.4	0.7

V. Experimental Results

A. Simulation and Hardware Integration

The simulation-to-hardware integration uses policies trained in the high-fidelity MuJoCo (Todorov et al. [41]) simulation with domain randomization over mass ($\mathcal{U}(-5.0, 5.0)$kg), friction ($\mathcal{U}(-0.3, 0.5)$), actuator gains/biases ($\mathcal{U}(\pm 20)$), and observation delays. RL policies run at 60 Hz, outputting smoothed joint positions via a moving average, tracked by 300 Hz low-level PID motor controllers. For contact-rich tasks like pull-ups, an admittance controller with auto-tuning operates at 300 Hz to ensure compliant force tracking, while safety is enforced using a Reference Governor and a precomputed

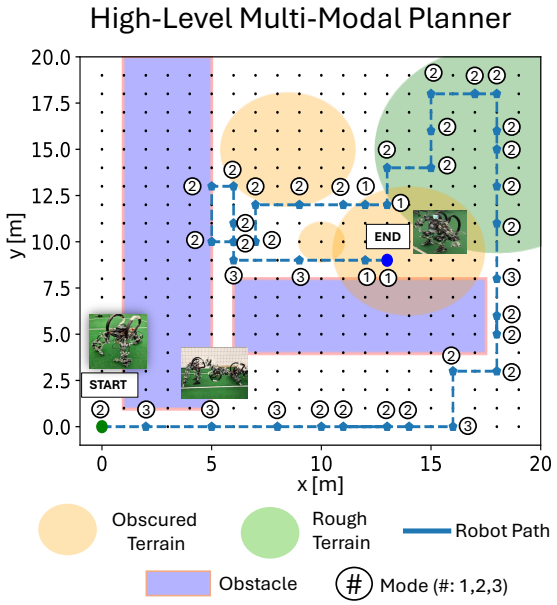


Fig. 7: We show one example map, along with the output of our MIQCP-based planner, where the output consist of connected waypoints, where each waypoint is designated with the desired robot mode selection, numbered from (1) to (3). Mode (1) means the robot must be in biped mode, mode (2) is crawling, and mode (3) is rolling. The solver aims to optimize for reaching the goal or end state within a limited time (or N points), achieve low energy consumption, visit as many cells as possible, avoid obstacles, and meet terrain requirements which may enforce being in one mode or another, see Sec. IV-C.

Maximal Output Admissible Set (MOAS). State estimation fuses the slower T265 VIO-SLAM (200 Hz) with the OptiState algorithm (Schperberg et al. [34]), enabling robust real-time control across all locomotion modes with zero-shot sim-to-real transfer. The visual-servo control module enables pre-manipulation positioning by tracking the slide's handle bars before climbing. An object detection model (Redmon et al. [31]) runs at 30 Hz, synchronized with the RGB-D camera, to estimate the centroid positions of the handles. The resulting reference positions are converted to joint angles using IK. For the high-level multi-modal planner, the MIQCP optimization can take approximately 10-11 seconds to solve on a 20 m by 20 m map for a 20 second trajectory. We use Gurobi (Gurobi Optimization, LLC [18]) as the optimization solver.

B. Bipedal Locomotion Performance

We compare a model-based controller using online planning and MPC (Sec. C of Appendix) with a model-free RL policy (Sec. IV-A) for biped and crawling locomotion. To evaluate robustness, we apply increasing impulse velocity disturbances of 0.1 s duration to the robot's CoM during standing until failure occurs. The model-based controller fails at approximately 0.05 m/s disturbance, while the RL policy remains stable up to 0.25 m/s. This behavior is expected, as MOBIUS exhibits complex, asymmetric dynamics that are difficult to model accurately, leading to state propagation errors in MPC, whereas the RL policy implicitly captures these effects. The RL controller is further validated under height-map disturbances in simulation and in hardware experiments. We additionally compare velocity tracking performance on hardware for x, y, and yaw motions (Fig. 8a), including lateral, forward, backward, and rotational commands. Target velocity is shown in green, with estimated velocities from the model-based controller (blue) and RL policy (red). The robot achieves up to 0.1 m/s laterally and 0.25 rad/s in yaw. Across all motions, the model-based approach exhibits higher tracking error, while the RL policy consistently improves accuracy. Although the RL policy shows slightly worse forward x-velocity tracking, the model-based controller induces larger errors in undesired y and yaw directions despite a zero velocity reference, indicating superior disturbance rejection and motion decoupling by the RL policy.

C. Crawling Locomotion Analysis

A similar velocity tracking comparison between the model-based approach and RL is given for the crawling locomotion in Fig. 8b. Unlike bipedal locomotion, the model-based and RL approach provided more comparable results, although overall, for sideways, and backward motion, RL performed improved tracking. One reason the RL policy struggles with velocity tracking compared to biped locomotion, may be due to the complexity of contact dynamics introduced by using the robot's two gripper-equipped arms as front limbs. These grippers are not optimized for high-frequency ground contact, and the simulator may not accurately model their contact behavior, leading to discrepancies during sim-to-real transfer. Additionally, crawling locomotion operates at higher speeds than biped mode, making tracking more difficult due to increased dynamic effects. Specifically, crawling mode can reach up to 0.4 m/s in lateral directions and 0.6 rad/s in yaw direction. Finally, the asymmetry in limb morphology, two arms and two legs with different masses, kinematics, and end-effectors, introduces further modeling and control challenges not present in more uniform crawling locomotion.

D. Rolling Locomotion

The rolling motion shares similar key points from transition from crawling to standing or biped mode, but before it reaches the default standing pose, it jumps backward and uses the back rail to roll, as shown in Fig. 1c. This rolling motion is the fastest locomotion at 0.7 m/s (instantaneous speed), which is 600.0 % and 75.0 % faster than biped walking and crawling respectively. Nevertheless, although this motion is more energy-efficient compared to biped mode, it is the least stable as the motion cannot be perfectly controlled, particular during the roll where it passively (rather than actively) performs the motion using its back rails. This introduces errors for state estimation, where the robot cannot be tracked accurately.

E. Multi-Modal High-Level Planning Results

We evaluate our multi-modal high-level planner (Sec. IV-C) using the Energy per Visited Cell (EVC) metric. The planner aims to reach the goal while maximizing exploration and minimizing energy per visited grid cell. Energy per Visited cell is defined as the total energy expenditure (joint velocity $\times$ joint torque) divided by the number of unique grid cells visited.

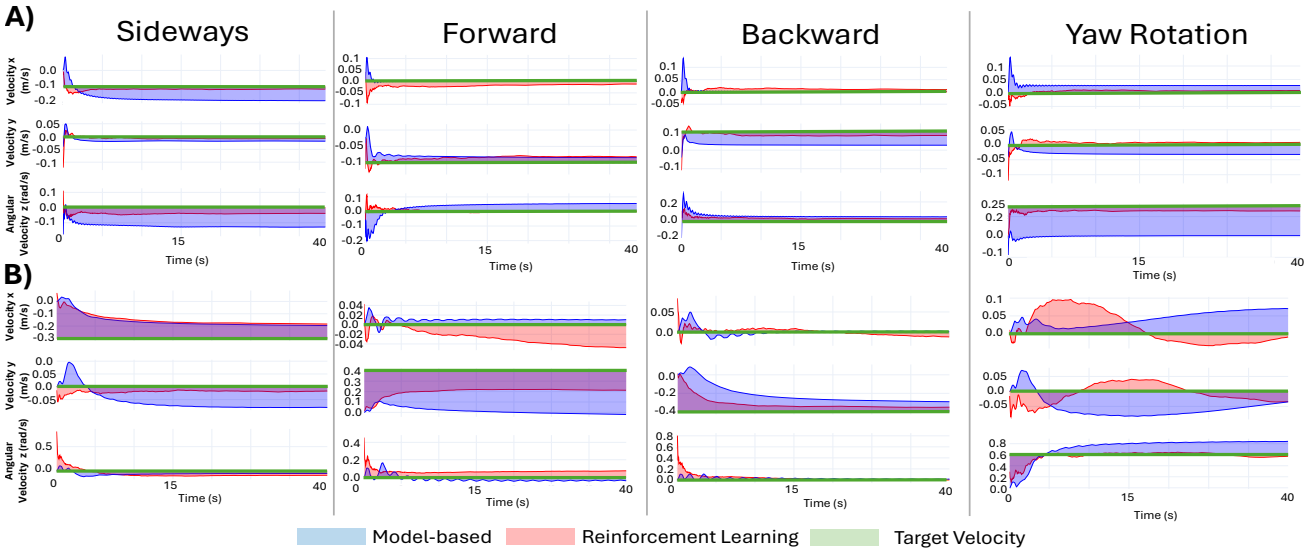


Fig. 8: We compare velocity tracking ability between biped mode shown in row (A) and crawling mode shown in row (B) using the baseline model-based approach, Sec. C of Appendix and using RL, Sec. IV-A for sideways, froward, backward, and yaw rotation motion. The velocity values for model-based is shown in blue, RL in red, and the target or reference velocity shown in green. For example, in the sideways motion for the biped mode, the target velocity in x is -0.1 m/s, and 0.0 m/s in y and angular velocity (since we command the robot to go sideways only). Note, for our frame definition, the y-axis is pointed forwards.

Energy usage is computed from each locomotion mode's maximum velocity (Table II) and travel distance. Crawling traverses 1–2 cells per action (each 0.5×0.5 m), biped follows the same formulation, and rolling is constrained to exactly 3 cells per action.

We first fix the trajectory horizon T and vary all remaining hyperparameter (weight on exploration, weight on goal, epsilon, and Big-M constant), selecting the two configurations that minimize EVC while always reaching the goal. The objective weights W_{exp} and W_{goal} balance exploration and goal-reaching, while ϵ and M define the Big-M constraint. We found that low EVC is achieved when exploration and goal objectives are equally balanced, with ϵ and M both low or both high, and an average mode-switch rate of approximately 25% per time step.

We also tested the case where all parameters (including T) are varied but the robot is restricted to a single locomotion mode, Table III. Mode-specific regions from Fig. 7 are excluded, and only obstacle constraints are enforced. In this test, we found that crawling achieves the lowest EVC (5.11 J/cell), outperforming bipedal (26.89 J/cell) and rolling locomotion (21.96 J/cell). Rolling reaches the goal fastest ($T = 15$) due to its higher speed, but provides limited coverage: only one cell is counted as visited during rolling due to onboard sensor limitations, a constraint enforced in the MIQCP formulation. Biped locomotion exhibits the highest failure rate, as its limited per-step distance requires longer horizons, increasing optimization difficulty. Although biped mode offers greater coverage than rolling, it is substantially less energy-efficient due to higher energy expenditure per step.

Overall, crawling provides the most favorable trade-off between energy efficiency and grid coverage, while rolling prioritizes traversal speed.

TABLE III: Energy and Cost of Transport per Mode

Mode	Energy (J/cell)	CoT
Biped	26.89	0.548
Crawl	5.11	0.104
Roll	21.96	0.448

Biped locomotion is less energy-efficient than crawling. Using realistic COM heights, biped motion requires approximately 104 J/m compared to 65 J/m for crawling, primarily due to increased potential energy costs and higher per-actuator loads. Crawling distributes both kinetic and control effort across more limbs, reducing peak actuator loads and improving thermal efficiency. Further analysis, including hyperparameter selection, and energy comparisons are given in Tables V and VI of Appendix.

F. Vertical Mobility

1) Vertical Climbing on a Kid's Slide: Leveraging MOBIUS's multi-modal locomotion and spine-enhanced grippers, MOBIUS can climb a children's slide, as shown in Fig. 1d. The robot pinch-grasps the side rails while stepping upward and subsequently slides down head-first after reaching the top. Across 10 trials, 8 were successful; failures occurred when a foot failed to advance to the next step (e.g., toe entrapment). A visual-servo controller aligns the end-effectors to the vertical handles (Fig. 1f), after which an open-loop trajectory is executed for climbing and descent, demonstrating hardware capability even without closed-loop control for the robot base. Note, we provide the full visual sequence of climbing the ladder in Fig. 10 of Appendix.

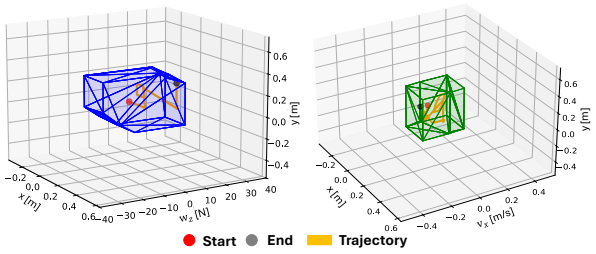


Fig. 9: Example visual representation of the Maximal Output Admissible Set (MOAS), Sec. IV-B1. The trajectory on the left represents a 3D state from estimation following the reference in Fig. 10, where x and y denote end-effector position, w_z the wrench, and v_x the linear velocity.

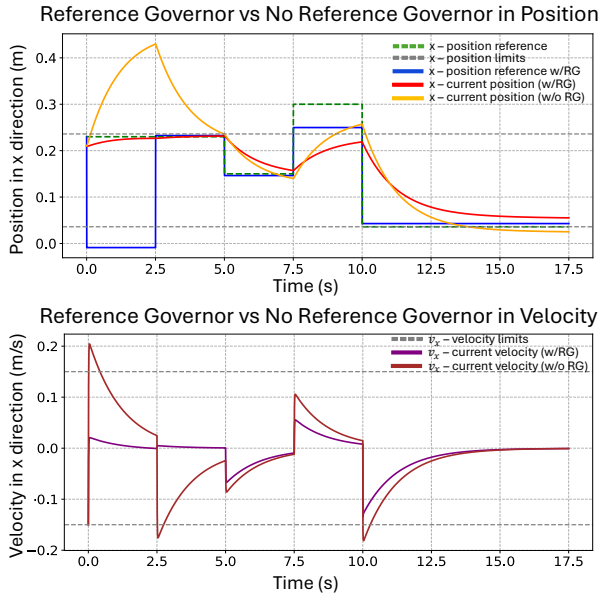


Fig. 10: End-effector x-position and velocity tracking with and without a reference governor. Position (top) and velocity (bottom) constraints are indicated by dashed lines.

2) Pinch Grasp Pull-up: The pinch grasp pull-up task demonstrates the strength and compliance of MOBIUS's forearm and gripper design (Fig. 1e). MOBIUS pinch-grasps aluminum structural bars and lifts its full body from a straight-arm dead-hang configuration. The uniform limb design provides effective load distribution, which is critical for climbing, while admittance control in the forearms enables compliant motion during the pull-up. Spine tips are typically effective on rough surfaces with microcavities; however, in the pull-up task (Fig. 1e), they also support MOBIUS's full body weight on an aluminum bar while executing admittance control. The spines engage grooves parallel to the bar, producing a limit surface that bounds the maximum grasping force (Table IV). The grasp generates a dominant shear force of 124.2 N perpendicular to the groove (z-axis), while frictional force along the groove (x-axis) remains low at 23.6 N. Although the spine limit surface is stochastic, the z-axis force exhibits a higher standard deviation (24.5 N) than the x-axis (4.1 N), indicating that spine engagement, rather than friction, dominates load support, enabling a single gripper to sustain MOBIUS's weight in the z direction. Across 10 trials, 9 were successful; the single failure matched the slide-climbing failure mode (foot advancement error).

TABLE IV: Maximum supporting force of the spine gripper on a slippery aluminum bar in Fig. 1e.

Direction	Maximum Supporting Force
x-axis	23.6 N $\pm$4.1 N
z-axis	124.2 N $\pm$24.5 N

3) Auto-Tuning Admittance Control and Safety Analysis: Manual tuning of the admittance force controller for the pull-up task is difficult due to end-effector constraints and the arms supporting the robot's full body weight. Auto-tuning improves force tracking error by 45% while maintaining controller stability, consistent with results reported from Schperberg et al. [33]. Safety during force control is enforced using the Reference Governor (RG) described in Sec. IV-B1. Figs. 9 and 10 illustrate these guarantees. Fig. 9 shows the MOAS during force control while tracking the position in Fig. 10 with a reference wrench of 0 N. The system trajectory (yellow), from the initial state (red) to the final state (gray), remains entirely within the admissible set, ensuring safety. Fig. 10 compares end-effector motion with and without RG. The x position with RG (red) remains within kinematic limits (gray dashed), whereas without RG (yellow) it violates these constraints. The RG modifies the reference (blue) only when necessary; for example, from 10.0 to 17.5 s, the reference is unchanged as the trajectory stays within the MOAS. The lower plot shows that RG enforces velocity limits: the end-effector velocity with RG (purple) remains within bounds, while without RG (dark red) violations occur at 0.0 s 2.5 s, and 10.0 s. Additional results showing force tracking are provided in Sec. F of Appendix.

G. Conclusion and Limitations

We demonstrated that MOBIUS achieves bipedal walking, crawling, rolling, and dynamic climbing, including pinching-based pull-ups. The system combines a compact, fatigue-resistant hardware design with high-DoF limbs and a cohesive control stack integrating model-free RL, admittance force control, and MIQCP-based high-level decision making. Extensive simulation and hardware experiments validate the value of co-designing morphology, control, and planning.

Despite its versatility, MOBIUS involves several trade-offs. The dual use of grippers for grasping and locomotion increases mechanical complexity and wear, while spur gears are susceptible to fatigue under high-impact bipedal loads. The reinforcement learning controllers depend on extensive domain randomization and may generalize poorly to novel surfaces, and state estimation for rolling is unreliable. In addition, the MIQCP-based planner operates offline with assumed terrain knowledge, limiting real-time reactivity, and bipedal locomotion remains energy-intensive. Future work will address real-time high-level replanning through a full vision-stack, long-horizon autonomy in cluttered environments, language-conditioned task specification, and hardware refinements for energy efficiency and outdoor deployment.

References

[1] Xinpei Ai, Hengmao Yue, and Wei Dawid Wang. Crawling soft robot exploiting wheel-legs and multimodal locomotion for high terrestrial maneuverability. *IEEE Transactions on Robotics*, 39(6):4230–4239, 2023. doi: 10.1109/TRO.2023.3299530.

[2] Max P Austin, Jason M Brown, Charles A Young, and Jonathan E Clark. Leg design to enable dynamic running and climbing on bobcat. In *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 3799–3806. IEEE, 2018.

[3] Robert Baines, Sree Kalyan Patiballa, Joran Booth, Luis Ramirez, Thomas Sipple, Andonny Garcia, Frank Fish, and Rebecca Kramer-Bottiglio. Multi-environment robotic transitions through adaptive morphogenesis. 610 (7931):283–289. ISSN 1476-4687. doi: 10.1038/s41586-022-05188-w. URL https://doi.org/10.1038/s41586-022-05188-w.

[4] Guillaume Bellegarda, Yiyu Chen, Zhuochen Liu, and Quan Nguyen. Robust high-speed running for quadruped robots via deep reinforcement learning. In *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 10364–10370, 2022. doi: 10.1109/IROS47612.2022.9982132.

[5] Marko Bjelonic, C. Dario Bellicoso, Yvain de Viragh, Dhionis Sako, F. Dante Tresoldi, Fabian Jenelten, and Marco Hutter. Keep rollin'—whole-body motion control and planning for wheeled quadrupedal robots. *IEEE Robotics and Automation Letters*, 4(2):2116–2123, 2019. doi: 10.1109/LRA.2019.2899750.

[6] Amanda Bouman, Muhammad Fadhil Ginting, Nikhilesh Alatur, Matteo Palieri, David D. Fan, Thomas Touma, Torkom Pailevanian, Sung-Kyun Kim, Kyohei Otsu, Joel Burdick, and Ali-akbar Agha-Mohammadi. Autonomous spot: Long-range autonomous exploration of extreme environments with legged locomotion. In *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 2518–2525, 2020. doi: 10.1109/IROS45743.2020.9341361.

[7] Timothy Bretl. Motion planning of multi-limbed robots subject to equilibrium constraints: The free-climbing robot problem. *The International Journal of Robotics Research*, 25(4):317–342, 2006.

[8] Guillermo A. Castillo, Bowen Weng, Wei Zhang, and Ayonga Hereid. Robust feedback motion policy design using reinforcement learning on a 3d digit bipedal robot. In *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 5136–5143, 2021. doi: 10.1109/IROS51168.2021.9636467.

[9] Zhe Chen, Brandon Tighe, and Jianguo Zhao. Origami-inspired modules enable a reconfigurable robot with programmable shapes and motions. *IEEE/ASME Transactions on Mechatronics*, 27(4):2016–2025, 2022. doi: 10.1109/TMECH.2022.3175145.

[10] Lillian Chin, Max Burns, Gregory Xie, and Daniela Rus. Flipper-style locomotion through strong expanding modular robots. *IEEE Robotics and Automation Letters*, 8(2):528–535, 2023. doi: 10.1109/LRA.2022.3227872.

[11] Pravin Dangol, Eric Sihite, and Alireza Ramezani. Control of thruster-assisted, bipedal legged locomotion of the harpy robot. *Frontiers in Robotics and AI*, 8, 2021. ISSN 2296-9144. doi: 10.3389/frobt.2021.770514. URL https://www.frontiersin.org/articles/10.3389/frobt.2021.770514.

[12] Alejandro Escontrela, Xue Bin Peng, Wenhao Yu, Tingnan Zhang, Atil Iscen, Ken Goldberg, and Pieter Abbeel. Adversarial motion priors make good substitutes for complex reward functions, 2022. URL https://arxiv.org/abs/2203.15103.

[13] Siyuan Feng, X Xinjilefu, Christopher G. Atkeson, and Joohyung Kim. Optimization based controller design and implementation for the atlas robot in the darpa robotics challenge finals. In *2015 IEEE-RAS 15th International Conference on Humanoid Robots (Humanoids)*, pages 1028–1035, 2015. doi: 10.1109/HUMANOIDS.2015.7363480.

[14] Zipeng Fu, Xuxin Cheng, and Deepak Pathak. Deep whole-body control: Learning a unified policy for manipulation and locomotion. In *Conference on Robot Learning (CoRL)*, 2022.

[15] Emanuele Garone, Stefano Di Cairano, and Ilya Kolmanovsky. Reference and command governors for systems with constraints: A survey on theory and applications. *Automatica*, 75:306–328, 2017. ISSN 0005-1098. doi: https://doi.org/10.1016/j.automatica.2016.08.013. URL https://www.sciencedirect.com/science/article/pii/S0005109816303715.

[16] Yifeng Gong, Ge Sun, Aditya Nair, Aditya Bidwai, Raghuram CS, John Grezmak, Guillaume Sartoretti, and Kathryn A. Daltorio. Legged robots for object manipulation: A review. *Frontiers in Mechanical Engineering*, 9, 2023. ISSN 2297-3079. doi: 10.3389/fmech.2023.1142421.

[17] Yukai Gong, Ross Hartley, Xingye Da, Ayonga Hereid, Omar Harib, Jiunn-Kai Huang, and Jessy Grizzle. Feedback control of a cassie bipedal robot: Walking, standing, and riding a segway. In *2019 American Control Conference (ACC)*, pages 4559–4566, 2019. doi: 10.23919/ACC.2019.8814833.

[18] Gurobi Optimization, LLC. Gurobi Optimizer Reference Manual, 2024. URL https://www.gurobi.com.

[19] Joshua Hooks, Min Sung Ahn, Jeffrey Yu, Xiaoguang Zhang, Taoyuanmin Zhu, Hosik Chae, and Dennis Hong. Alphred: A multi-modal operations quadruped robot for package delivery applications. *IEEE Robotics and Automation Letters*, 5(4):5409–5416, 2020. doi: 10.1109/LRA.2020.3007482.

[20] Marco Hutter, Christian Gehring, Dominic Jud, Andreas Lauber, C Dario Bellicoso, Vassilios Tsounis, Jemin Hwangbo, Karen Bodie, Peter Fankhauser, Michael Bloesch, et al. Anymal-a highly mobile and dynamic

quadrupedal robot. In *2016 IEEE/RSJ international conference on intelligent robots and systems*, pages 38–44. IEEE, 2016.

[21] Kyunam Kim, Patrick Spieler, Elena-Sorina Lupu, Alireza Ramezani, and Soon-Jo Chung. A bipedal walking robot that can fly, slackline, and skateboard. *Science Robotics*, 6(59):eabf8136, 2021. doi: 10.1126/scirobotics.abf8136. URL https://www.science.org/doi/abs/10.1126/scirobotics.abf8136.

[22] Sangbae Kim, Matthew Spenko, Salomon Trujillo, Barrett Heyneman, Daniel Santos, and Mark R Cutkosky. Smooth vertical surface climbing with directional adhesion. *IEEE Transactions on robotics*, 24(1):65–74, 2008.

[23] Korea Atomic Energy Research Institute. Korea atomic energy research institute (kaeri) — english website. https://www.kaeri.re.kr/eng/, 2025. Accessed: 31 Oct. 2025.

[24] Scott Kuindersma, Frank Permenter, and Russ Tedrake. An efficiently solvable quadratic program for stabilizing dynamic locomotion. In *2014 IEEE International Conference on Robotics and Automation (ICRA)*, pages 2589–2594, 2014. doi: 10.1109/ICRA.2014.6907230.

[25] Kenji Nagaoka, Hayato Minote, Kyohei Maruya, Yuki Shirai, Kazuya Yoshida, Takeshi Hakamada, Hirotaka Sawada, and Takashi Kubota. Passive spine gripper for free-climbing robot in extreme terrain. *IEEE Robotics and Automation Letters*, 3(3):1765–1770, 2018.

[26] Aaron Parness, Neil Abcouwer, Christine Fuller, Nicholas Wiltsie, Jeremy Nash, and Brett Kennedy. Lemur 3: A limbed climbing robot for extreme terrain mobility in space. In *2017 IEEE international conference on robotics and automation*, pages 5467–5473. IEEE, 2017.

[27] Giulia Piovan and Francesco Bullo. On coordinate-free rotation decomposition: Euler angles about arbitrary axes. *IEEE Transactions on Robotics*, 28(3):728–733, 2012. doi: 10.1109/TRO.2012.2184951.

[28] Max Polzin, Qinghua Guan, and Josie Hughes. Robotic locomotion through active and passive morphological adaptation in extreme outdoor environments. *Science Robotics*, 10(99):eadp6419, 2025. doi: 10.1126/scirobotics.adp6419. URL https://www.science.org/doi/abs/10.1126/scirobotics.adp6419.

[29] Marc H. Raibert. *Legged Robots That Balance*. Massachusetts Institute of Technology, USA, 1986. ISBN 0262181177.

[30] Marc H. Raibert, Jr H. Benjamin Brown, and Michael Chepponis. Experiments in balance with a 3d one-legged hopping machine. *The International Journal of Robotics Research*, 3(2):75–92, 1984. doi: 10.1177/027836498400300207. URL https://doi.org/10.1177/027836498400300207.

[31] Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. You only look once: Unified, real-time object detection. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 779–788, 2016. doi: 10.1109/CVPR.2016.91.

[32] Aaron Saunders, Daniel I Goldman, Robert J Full, and Martin Buehler. The rise climbing robot: body and leg design. In *Unmanned Systems Technology VIII*, volume 6230, pages 401–413. SPIE, 2006.

[33] Alexander Schperberg, Yuki Shirai, Xuan Lin, Yusuke Tanaka, and Dennis Hong. Adaptive force controller for contact-rich robotic systems using an unscented kalman filter. In *2023 IEEE-RAS 23rd International Conference on Humanoid Robots*, 2023.

[34] Alexander Schperberg, Yusuke Tanaka, Saviz Mowlavi, Feng Xu, Bharathan Balaji, and Dennis Hong. Optistate: State estimation of legged robots using gated networks with transformer-based vision and kalman filtering. *arXiv preprint arXiv:2401.16719*, 2024.

[35] Alexander Schperberg, Yeping Wang, and Stefano Di Cairano. Safe whole-body loco-manipulation via combined model and learning-based control, 2026. URL https://arxiv.org/abs/2603.02443.

[36] Sangok Seok, Albert Wang, Meng Yee Chuah, David Otten, Jeffrey Lang, and Sangbae Kim. Design principles for highly efficient quadrupeds and implementation on the mit cheetah robot. In *2013 IEEE International Conference on Robotics and Automation*, pages 3307–3312. IEEE, 2013.

[37] Fan Shi, Timon Homberger, Joonho Lee, Takahiro Miki, Moju Zhao, Farbod Farshidian, Kei Okada, Masayuki Inaba, and Marco Hutter. Circus anymal: A quadruped learning dexterous manipulation with its limbs. In *2021 IEEE International Conference on Robotics and Automation*, pages 2316–2323. IEEE, 2021.

[38] Manjia Su, Yu Qiu, Yisheng Guan, Haifei Zhu, and Zhi Liu. Climbot-Ω: A soft robot with novel grippers and rigid-compliantly constrained body for climbing on various poles. In *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 4975–4981. IEEE, 2021.

[39] Yusuke Tanaka, Yuki Shirai, Zachary Lacey, Xuan Lin, Jane Liu, and Dennis Hong. An under-actuated whipple-tree mechanism gripper based on multi-objective design optimization with auto-tuned weights. In *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 6139–6146, 2021. doi: 10.1109/IROS51168.2021.9635872.

[40] Yusuke Tanaka, Yuki Shirai, Xuan Lin, Alexander Schperberg, Hayato Kato, Alexander Swerdlow, Naoya Kumagai, and Dennis Hong. Scaler: A tough versatile quadruped free-climber robot. In *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 5632–5639, 2022. doi: 10.1109/IROS47612.2022.9981555.

[41] Emanuel Todorov, Tom Erez, and Yuval Tassa. Mujoco: A physics engine for model-based control. In *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 5026–5033. IEEE, 2012. doi: 10.1109/IROS.2012.6386109.

[42] Vassilios Tsounis, Mitja Alge, Joonho Lee, Farbod

Farshidian, and Marco Hutter. Deepgait: Planning and control of quadrupedal gaits using deep reinforcement learning. *IEEE Robotics and Automation Letters*, 5(2): 3699–3706, 2020. doi: 10.1109/LRA.2020.2979660.

[43] Kentaro Uno, Naomasa Takada, Taku Okawara, Keigo Haji, Arthur Candalot, Warley F. R. Ribeiro, Kenji Nagaoka, and Kazuya Yoshida. Hubrobo: A lightweight multi-limbed climbing robot for exploration in challenging terrain. In *2020 IEEE-RAS 20th International Conference on Humanoid Robots (Humanoids)*, pages 209–215, 2021. doi: 10.1109/HUMANOIDS47582.2021.9555799.

[44] Eric Vollenweider, Marko Bjelonic, Victor Klemm, Nikita Rudin, Joonho Lee, and Marco Hutter. Advanced skills through multiple adversarial motion priors in reinforcement learning. In *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pages 5120–5126, 2023. doi: 10.1109/ICRA48891.2023.10160751.

[45] Changyu Xu, Yajun Cao, Jingyang Zhao, Yujia Cao, Yi Huang, Yangyi Lin, Dong Wang, Zhuang Zhang, and Hanqing Jiang. Muscle-inspired elasto-electromagnetic mechanism in autonomous insect robots. 16(1):6813. ISSN 2041-1723. doi: 10.1038/s41467-025-62182-2. URL https://doi.org/10.1038/s41467-025-62182-2.

[46] Justin K. Yim and Ronald S. Fearing. Precision jumping limits from flight-phase control in salto-1p. pages 2229–2236, 2018. doi: 10.1109/IROS.2018.8594154.

[47] Chen Yu and Andre Rosendo. Multi-modal legged locomotion framework with automated residual reinforcement learning. *IEEE Robotics and Automation Letters*, 7(4):10312–10319, 2022. doi: 10.1109/LRA.2022.3191071.

Appendix: Supplementary Material for MOBIUS: A Multi-Modal Bipedal Robot that can Walk, Crawl, Climb, and Roll

APPENDIX

A. Details on Hardware

1) Limb Kinematics and Jacobian Properties: In bipedal mode, MOBIUS exhibits a $19\,\%$ increase in velocity ellipsoid volume and a $24\,\%$ reduction in Jacobian condition number compared to the mechanism design from Tanaka et al. [13], resulting in more isotropic swing-leg motion. This improves tracking of cubic swing trajectories at the expense of reduced force isotropy, which remains sufficient for walking and crawling.

2) Actuator Selection and Fatigue Analysis: MOBIUS uses high gear-ratio spur gear actuators for compactness and weight efficiency. However, repeated impact loading during bipedal walking introduces fatigue risks. Gear bending stress is estimated using AGMA formulations (Kumar et al. [4]):

$$\sigma_b = \frac{F_t}{mJ} \frac{K_a K_m}{K_v} K_S K_B K_I \tag{1}$$

At peak bipedal torques of 2.5 Nm, the estimated bending stress is 496 MPa, corresponding to a fatigue life of approximately 2.8×10^5 cycles. Smaller-module gears used in earlier designs exhibited significantly reduced fatigue life (Fig. 1 of Appendix). Additional bearings were introduced to mitigate out-of-plane loading and reduce stress concentration (Maitra [5]), preventing observed gear failures during experiments.

3) End-Effector Design and Force Characteristics: The gripper, as shown in Fig. 7 serves as both a grasping hand and a load-bearing foot. While underactuated grasping introduces compliance undesirable for locomotion, the gripper reaches a kinematic singularity when fully closed, eliminating compliance during ground contact. Force measurements on aluminum bars show that spine-based engagement dominates load support, producing up to 124.2 N in the normal direction with minimal reliance on friction. This allows a single gripper to support the full weight of MOBIUS during climbing and pull-up tasks.

B. Details on Reinforcement Learning for Locomotion

1) Domain Randomization and Noise: Employing an RL policy trained in simulation successfully on hardware typically demands sufficient domain randomization (Chen et al. [1], Park et al. [7], Jiang et al. [3], Shakerimov et al. [12], Schwartzwald and Papanikolopoulos [11]). To this end, we randomize several parameters as shown in Table III of Appendix. For each time step during training, we add a uniform noise on our state estimate for orientation, angular, and linear velocity. During the reset stage of each episode, we also add uniform noise to the initial starting state of the robot. We also introduce randomness to the observation state of our RL-agent, as shown in Algorithm 1. The objective of this randomization is to simulate potential communication delay of our observations. Additionally, by having a potentially erroneous current observation, the RL-agent may emphasize finding patterns on the change of observations from previous time steps (as our observation space includes a history of N number of past observations). To further robustify against disturbances, we simulate a kick against the robot by adding

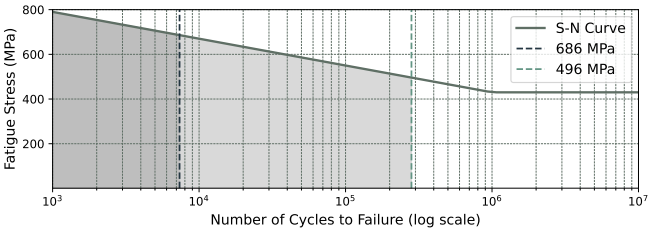


Fig. 1: Estimated gear fatigue life based on S–N curves for 1045 steel (Moćko [6]).

Parameter Name	Value
Mujoco solver time step	0.005
Linear search iteration	3
Constrain iteration	3
Network parameters	(128,128,128,128)
Batch size	256
Learning rate	3e-4
Discounting	0.97
Entropy cost	1e-2
Minibatches	32
Update per batch	4
Num of time steps (flat)	300,000,000
Num of time steps (rough)	100,000,000
Max time step in episode	2000
Max motor RPM	50 (rpm)
Action scale (a_{scale})	0.3

TABLE I: Hyperparameters and settings for RL training

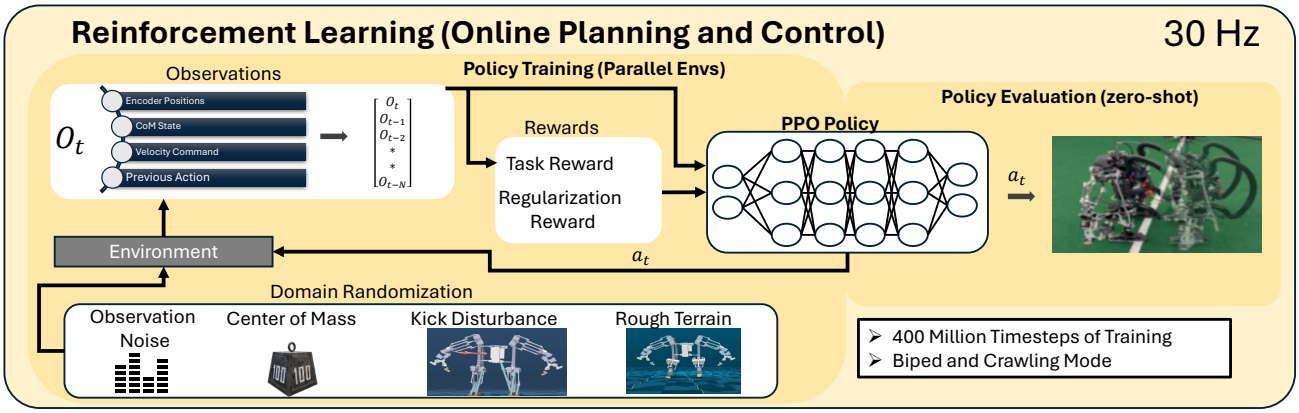


Fig. 2: Our reinforcement learning flowchart is illustrated. It includes the observations used, rewards, and domain randomization, to achieve biped and crawling locomotion. We trained 300 million time steps on flat terrain, and another 100 million time steps on rough terrain to achieve robustness.

Reward/Penalty Description	Reward Equation
Penalize Vertical Velocity (R_{v_z})	${v_z}^2$
Penalize Angular Velocity ($R_{\omega_{xy}}$)	$\|\|\boldsymbol{\omega}_{xy}\|\|^2$
Penalize Joint Torques (R_τ)	$\sum_{i=1}^{N_j} \|\boldsymbol{\tau}_i \|\boldsymbol{\omega}_i\|$
Penalize Change in Actions (R_a)	$\sum_{i=1}^{N_j} (\mathbf{a}_t^i - \mathbf{a}_{t-1}^i)$
Reward Linear Velocity ($R_{v_{xy}}$)	$\exp\left(-\frac{\|\|\mathbf{v}_{xy}^{des} - \mathbf{v}_{xy}\|\|^2}{\sigma}\right)$
Reward Angular Velocity (R_{ω_z})	$\exp\left(-\frac{\|\|\boldsymbol{\omega}_z^{des} - \boldsymbol{\omega}_z\|\|^2}{\sigma}\right)$
Reward Standing Still (R_{stand})	$\sum_{i=1}^{N_j} \left\|\boldsymbol{\theta}_i - \boldsymbol{\theta}_i^{\text{ref}}\right\| \left(\|\|\mathbf{v}^{\text{des}}\|\| < 0.1\right)$
Reward Air Time (R_{air})	$\sum_{i=1}^{N_f} \|\Delta T_{air} \mathbf{C}_i\| \left(\|\|\mathbf{v}^{\text{des}}\|\| > 0.05\right)$
Reward Contact (R_{contact})	$\sum_{i=1}^{N_f} \mathbf{C}_i \left(T_{\text{contact}} \geq 0.10\right)$
Penalize Foot Slip (R_{slip})	$\sum_{i=1}^{N_f} \|\mathbf{p}_i \mathbf{C}_i\|^2 \left(\|\|\mathbf{v}^{\text{des}}\|\| > 0.05\right)$
Penalize Termination (R_{term})	$\text{Done}_{term}(t < 500)$

TABLE II: Reward and penalty formulation for the RL planning and control pipeline.

Description	Symbol	Range of Values
Orientation (x, y, z)	$\boldsymbol{\theta}$	$\boldsymbol{\theta} \sim \mathcal{U}(-\boldsymbol{\pi}, \boldsymbol{\pi})^\circ$
Angular velocity (yaw)	ω_{yaw}	$\omega_{\text{yaw}} \sim \mathcal{U}(-0.15, 0.15)$rad/s
Linear velocity (x, y, z)	$\mathbf{v}_{\text{linear}}$	$\mathbf{v}_{\text{linear}} \sim \mathcal{U}(-\mathbf{0.05}, \mathbf{0.05})$m/s
Initial starting state	$\delta\boldsymbol{\Theta}_{\text{default}}$	$\delta\boldsymbol{\Theta}_{\text{default}} \sim \mathcal{U}(\mathbf{0.01}, \mathbf{0.01})^\circ$
Selection of Past Observation	N_{delay}	$N_{\text{delay}} \sim \mathcal{U}(1, 3)$
Prior Selection Chance	p	0.3
Kick Interval	P_{int}	$P_{\text{int}} \sim \mathcal{U}(10, 30)$
Kick angle	θ_{kick}	$\theta_{\text{kick}} \sim \mathcal{U}(0, 2\pi)^\circ$
Body mass	m_{body}	$m_{\text{body}} \sim \mathcal{U}(-5.0, 5.0)$kg
Body location (x, y, z)	$\mathbf{l}_{\text{body}}$	$\mathbf{l}_{\text{body}} \sim \mathcal{U}(-\mathbf{0.1}, \mathbf{0.1})$m
Surface friction	$\delta\mu$	$\delta\mu \sim \mathcal{U}(-0.3, 0.5)$
Actuator gain	$\delta\mathbf{k}_{\text{act}}$	$\mathbf{k}_{\text{act}} \sim \mathcal{U}(-\mathbf{20}, \mathbf{20})$
Actuator bias	$\delta\mathbf{b}_{\text{act}}$	$\mathbf{b}_{\text{act}} \sim \mathcal{U}(-\mathbf{20}, \mathbf{20})$

TABLE III: Domain randomization parameters.

random linear velocities to the robot's trunk, $\mathbf{v}_{xy}$, at every P_{int} number of time steps and at random directions – see Algorithm 2. We also randomize the center of mass and its location, as well as surface friction and actuator gain/bias parameters. Finally, after our agent has finished training on flat terrain, we then continue the agent's training on a height map to ensure higher success during sim-to-real transfer.

2) Termination Conditions: For our RL training procedure, we impose the following termination conditions, which ends the episode:

$$x_z < x_z^{\min} \tag{2}$$

$$\dot{\boldsymbol{\Theta}} > \dot{\boldsymbol{\Theta}}_{\max} \tag{3}$$

$$|\boldsymbol{\Theta}_{\text{action}} - \boldsymbol{\Theta}_{\text{limit}}| < \epsilon_{\text{limit}} \tag{4}$$

$$n_{\text{steps}} < n_{\text{limit}}^{\text{steps}} \tag{5}$$

Where we want to ensure that the episode ends if the current estimated trunk height is below some specified limit (2), the current joint velocities reach beyond the limit set by the maximum RPM from out motors (3), the actions chosen by the policy is near the joint limits by some error ϵ_{limit} (4), or we reach the total number of time steps of our episode given by $n_{\text{limit}}^{\text{steps}}$ (5).

Algorithm 1 Introduce Randomness in Observations

Input: Current time step t, Number of past observations N, Max number of delay N_{delay}, Probability p of selecting a past observation, Current and past observations $\mathbf{O}_t = [\mathbf{O}_t, \mathbf{O}_{t-1}, \ldots, \mathbf{O}_{t-N+1}]$
Output: Current observation $\mathbf{O}_{\text{current}}$
Draw a random number $r \sim \mathcal{U}(0, 1)$
if $r \leq p$ **then**
    Draw a delay $D \sim \mathcal{U}(1, N_{\text{delay}})$
    Set $\mathbf{O}_{\text{current}} \leftarrow \mathbf{O}_{t-D}$
else
    Set $\mathbf{O}_{\text{current}} \leftarrow \mathbf{O}_t$
end if
return $\mathbf{o}_{\text{current}}$

C. Details on Model Predictive Control Baseline for Locomotion

The baseline locomotion employs an online planner (Sec. C1) to generate reference trajectories given a desired trunk

Algorithm 2 Update Velocity with Kick

$\theta_{\text{kick}} \sim \mathcal{U}(0, 2\pi)$
$\mathbf{k}_{\text{kick}} \leftarrow \begin{bmatrix} \cos(\theta_{\text{kick}}) \\ \sin(\theta_{\text{kick}}) \end{bmatrix}$
$\mathbf{k} \leftarrow \mathbf{k}_{\text{kick}} \left((n_{\text{steps}} \bmod P_{\text{int}}) == 0\right)$
$\mathbf{v}'_{x,y} \leftarrow \mathbf{v}_{x,y} + \mathbf{k} \cdot \text{kick\_vel}$
$\mathbf{v}_{x,y} \leftarrow \mathbf{v}'_{x,y}$

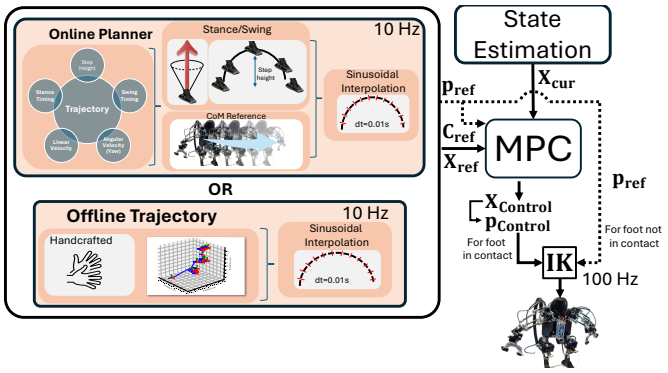


Fig. 3: We show different planning approaches depending on the secondary mode of our robot. For the climbing mode, we hand craft the trajectories offline, and run them primarily open-loop, with some minor correction of grasping hold positioning using the visual-servo controller described in Sec. H. For the locomotion tasks (e.g., biped walking or crawling crawling), we use an online planner to generate trunk and footstep trajectories given a desired trunk linear and angular velocity, and track them via MPC.

linear and angular velocity, and which is then tracked by a Model Predictive Controller (MPC) (Sec. C2).

1) Online Planner: The first two equations represent the Raibert feedback controllers for the x and y directions:

$$\text{raibert}_x = k_x(\dot{x}_{\text{cur}} - \dot{x}_{\text{des}}) \tag{6}$$

$$\text{raibert}_y = k_y(\dot{y}_{\text{cur}} - \dot{y}_{\text{des}}) \tag{7}$$

These equations calculate the corrective terms, raibert_x and raibert_y, in the x and y directions based on the difference between the current velocity $\dot{x}_{\text{cur}}$, $\dot{y}_{\text{cur}}$ and the desired velocity $\dot{x}_{\text{des}}$, $\dot{y}_{\text{des}}$. Here, $\dot{x}_{\text{cur}}$ and $\dot{y}_{\text{cur}}$ are the current velocities in the x and y directions, while $\dot{x}_{\text{des}}$ and $\dot{y}_{\text{des}}$ are the desired velocities. The feedback gains k_x and k_y are defined as $k_x = \sqrt{\frac{\bar{p}^z}{g}}$ and $k_y = \sqrt{\frac{\bar{p}^z}{g}}$, where $\bar{p}^z$ represents a reference height or step height and g is the gravitational constant.

The following equation considers the effects of angular velocity on the footstep position in the x and y directions:

$$\begin{bmatrix} p^x_{\text{angular}} \\ p^y_{\text{angular}} \end{bmatrix} = R(\theta) \begin{bmatrix} \bar{p}^x \\ \bar{p}^y \end{bmatrix} \tag{8}$$

This equation rotates the nominal footstep positions $\bar{p}^x$ and $\bar{p}^y$ by the angle θ, which is related to the desired angular velocity. This accounts for the changes in footstep positions due to body rotation. Here, $\bar{p}^x$ and $\bar{p}^y$ are the nominal footstep positions in the x and y directions, while p^x_{angular} and p^y_{angular} are the adjusted footstep positions accounting for angular velocity. The rotation matrix $R(\theta)$ is based on the angle θ, which is calculated as:

$$R(\theta) = \begin{bmatrix} \cos(\theta) & -\sin(\theta) \\ \sin(\theta) & \cos(\theta) \end{bmatrix} \tag{9}$$

with $\theta = \dot{\theta}_{\text{des}} 0.5 \Delta T$, where $\dot{\theta}_{\text{des}}$ is the desired angular velocity and ΔT is the footstep timing.

The following equations determine the progression terms in the x and y directions:

$$\Delta p^x_{\text{prog}} = \frac{1}{2}\dot{p}^x + p^x_{\text{angular}} + \text{raibert}_x \tag{10}$$

$$\Delta p^y_{\text{prog}} = \frac{1}{2}\dot{p}^y + p^y_{\text{angular}} + \text{raibert}_y \tag{11}$$

These equations calculate the progression terms Δp^x_{prog} and Δp^y_{prog}, which define the adjustment needed in the footstep positions in the x and y directions. They combine the effects of linear velocity, angular velocity, and the Raibert feedback. Here, $\dot{p}^x$ and $\dot{p}^y$ represent the linear velocities in the x and y directions, while p^x_{angular} and p^y_{angular} are the adjusted footstep positions due to angular velocity, and raibert_x and raibert_y are the Raibert feedback terms calculated earlier.

The reference trajectory from the current footstep to the next is generated using a sinusoidal implementation, as detailed in the following equations:

$$\bar{a}_z = p_{\text{step}} \frac{1}{2} \left(\frac{2\pi}{\Delta T}\right)^2 \tag{12a}$$

$$a_{z,t} = \bar{a}_z \cos\left(\frac{2\pi}{\Delta T} t\right) \tag{12b}$$

$$v_{z,t} = \bar{a}_z \frac{\Delta T}{2\pi} \sin\left(\frac{2\pi}{\Delta T} t\right) \tag{12c}$$

$$p_{z,t} = \bar{a}_z \left(\frac{\Delta T}{2\pi}\right)^2 \left(1 - \cos\left(\frac{2\pi}{\Delta T} t\right)\right) \tag{12d}$$

These equations define the vertical motion of the foot using a sinusoidal function. The motion is defined by maximum acceleration $\bar{a}_z$, velocity $v_{z,t}$, and position $p_{z,t}$, all functions of time t and step height p_{step}. Here, p_{step} represents the step height, ΔT is the step timing interval, and $\bar{a}_z$, $a_{z,t}$, $v_{z,t}$, and $p_{z,t}$ are the vertical acceleration, instantaneous acceleration, velocity, and position, respectively.

The equations for horizontal accelerations in the x and y directions are:

$$\bar{a}_x = \Delta p^x_{\text{prog}} \frac{1}{2} \left(\frac{\pi}{\Delta T}\right)^2 \tag{13}$$

$$\bar{a}_y = \Delta p^y_{\text{prog}} \frac{1}{2} \left(\frac{\pi}{\Delta T}\right)^2 \tag{14}$$

These equations determine the maximum horizontal accelerations $\bar{a}_x$ and $\bar{a}_y$ required to achieve the desired footstep positions. Here, Δp^x_{prog} and Δp^y_{prog} are the progression terms calculated earlier.

The horizontal motion of the foot is defined as:

$$a^{xy}(t) = \bar{a}_{xy} \cos\left(\frac{\pi}{\Delta T} t\right), \tag{15}$$

$$v^{xy}(t) = \bar{a}_{xy} \frac{\Delta T}{\pi} \sin\left(\frac{\pi}{\Delta T} t\right), \tag{16}$$

$$p_{xy}(t) = \bar{a}_{xy} \left(\frac{\Delta T}{\pi}\right)^2 \left(1 - \cos\left(\frac{\pi}{\Delta T} t\right)\right). \tag{17}$$

In these equations, $a^{xy}(t)$ and $v^{xy}(t)$ represent the time-dependent acceleration and velocity of the foot in the x and y directions, respectively. $p_{xy}(t)$ indicates the position of the foot. $\bar{a}_{xy}$ is the maximum horizontal acceleration, derived from the progression terms Δp^x_{prog} and Δp^y_{prog}. The motion follows a sinusoidal trajectory to ensure smooth transitions between footsteps.

2) Model-Predictive Control: To track the reference trajectories, either from an online planner as proposed in Sec. C1, or from offline trajectories, we employ a Model Predictive Controller (MPC) to calculate the ground reaction forces (for end-effectors in contact with the ground or object) to track the reference trajectory of the body in world frame or $\mathbf{X}_{\text{ref}} \in \mathbb{R}^{12}$, where $\mathbf{X}_{\text{ref}} = [\boldsymbol{\Theta}^\top, \mathbf{r}^\top, \boldsymbol{\omega}^\top, \mathbf{v}^\top]^\top$, $\boldsymbol{\Theta}$ is the robot's orientation, $\mathbf{r}$ is the CoM base position, $\boldsymbol{\omega}$ is the angular velocity, and $\mathbf{v}$ is the linear velocity, each with x, y, and z components While details of this controller can be found in various sources such as Di Carlo et al. [2], because our motors are position-controlled, we employ the approach taken in Tanaka et al. [13] instead. Specifically, instead of using the ground reaction forces and the inverse Jacobian to calculate the required joint torques, we instead track the ground reaction forces directly using the admittance controller, which outputs end-effector positions. These end-effector positions can then be converted to desired joint angles through inverse kinematics.

D. Details on the Maximal Output Admissibility Set (MOAS)

1) MOAS Computation Details: The computation of the MOAS is an iterative process that evaluates whether a given state and reference input remain admissible over a finite time horizon. We calculate the MOAS using second-order Euler integration together with the control law given by the admittance controller. We first initialize the position, velocity, and wrench constraints, along with controller gains and discretization parameters. We then generate grids over position, velocity, reference position, wrench, and reference wrench to sample system trajectories.

For each combination of sampled $(\mathbf{x}_{\text{cur}}, \mathbf{v}_{\text{cur}}, \mathbf{x}_{\text{ref}}, \mathbf{W}_{\text{cur}}, \mathbf{W}_{\text{ref}})$, we reset integral terms, simulate the closed-loop dynamics over discrete time steps, and check whether position, velocity, or wrench constraints are violated. If any constraint is violated, the trajectory is discarded. Otherwise, the sample is added to $\mathcal{O}_\infty$. In practice, we compute MOAS independently along the x, y, and z axes for position, velocity, and force, and include torque about the axis normal to the gripper base.

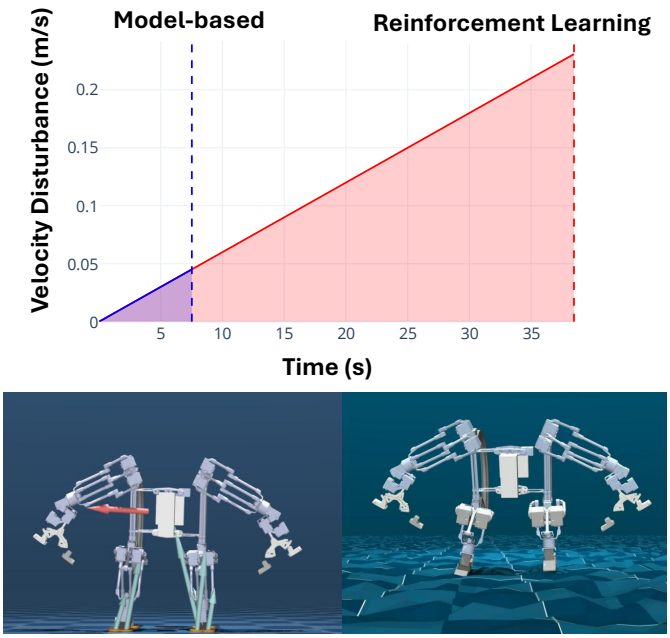


Fig. 4: We compare the model-based baseline approach with RL, for handling velocity disturbances in the top part of the figure, applied to the base of the robot (red arrow in bottom left figure). The bottom right figure shows the height map environment used to robustify our RL controller.

TABLE IV: MIQCP Constraints (1)-(20)

Visiting Grid Constraints	
(1)	$z(t,i,j) \in \{0,1\}$
(2)	$z_{\text{sum}}(i,j) = \sum_{t=0}^{T-1} z(t,i,j), \quad \forall i,j \in \{0,1,\ldots,N_{\text{grid}}-1\}$
(3)	$V(i,j) \leq z_{\text{sum}}(i,j), \quad \forall i,j$
(4)	$V(i,j) \leq T \cdot V(i,j), \quad \forall i,j$
(5)	$x(t,0) - i \geq -M(1 - z(t,i,j))$
(6)	$x(t,1) - j \geq -M(1 - z(t,i,j))$
Propagation Update	
(7)	$x(t+1) = x(t) + d(t)$
Mode Types Constraints	
(8)	$m_1(t) = 1 \implies d(t) \in \{-1,1\}$
(9)	$m_2(t) = 1 \implies d(t) \in \{-2,-1,1,2\}$
(10)	$m_3(t) = 1 \implies d(t) \in \{-3,3\}$
(11)	$m_1(t) + m_2(t) + m_3(t) = 1$
Terrain Type Constraints	
(12)	$d_{\text{circle}} = (x(t,0) - x_{\text{center}})^2 + (x(t,1) - y_{\text{center}})^2$
(13)	$d_{\text{circle}} \leq r^2_{\text{circle}} + M(1 - b^k_{\text{circle}}(t))$
(14)	$d_{\text{circle}} \geq r^2_{\text{circle}} + \epsilon - M b^k_{\text{circle}}(t)$
(15)	$m_k(t) \geq b^k_{\text{circle}}(t), \quad b^k_{\text{circle}} \in \{0,1\}$
Obstacle Constraints	
(16)	$x(t,0) \geq x_{\min} + M(1 - b^{\text{rect}}_1(t))$
(17)	$x(t,0) \leq x_{\max} - M(1 - b^{\text{rect}}_2(t))$
(18)	$x(t,1) \geq y_{\min} + M(1 - b^{\text{rect}}_3(t))$
(19)	$x(t,1) \leq y_{\max} - M(1 - b^{\text{rect}}_4(t))$
(20)	$b^{\text{rect}}_1(t) + b^{\text{rect}}_2(t) + b^{\text{rect}}_3(t) + b^{\text{rect}}_4(t) \leq 3$

Algorithm 3 MOAS Calculation

Initialize parameters: $\mathbf{x}_{\min}$, $\mathbf{x}_{\max}$, $\mathbf{v}_{\min}$, $\mathbf{v}_{\max}$, $\mathbf{w}_{\min}$, $\mathbf{w}_{\max}$, T, dt, N
Define control gains: $\mathbf{D}_d$, $\mathbf{K}_d$, $\mathbf{K}_f$, $\mathbf{K}_{I_x}$, $\mathbf{K}_{I_f}$
Generate state and reference grids: $\mathbf{x}_{\text{samples}}$, $\mathbf{v}_{\text{samples}}$, $\mathbf{x}_{\text{ref\_samples}}$, $\mathbf{W}_{\text{samples}}$, $\mathbf{W}_{\text{ref\_samples}}$
Create empty list $\mathcal{O}_\infty = []$
for each state $\mathbf{x}_{\text{cur}}, \mathbf{v}_{\text{cur}}, \mathbf{x}_{\text{ref}}, \mathbf{W}_{\text{cur}}, \mathbf{W}_{\text{ref}}$ **do**
    Set valid = **True**
    Set $\mathbf{z_x} = 0$, $\mathbf{z_f} = 0$
    for each time step $t = 0$ to $T/\delta t$ **do**
        Compute control acceleration $\mathbf{u}_t$ using Eq. (4) of main text.
        Update current position and velocity:

$$\mathbf{x}_{\text{cur}} \leftarrow \mathbf{x}_{\text{cur}} + \mathbf{v}_{\text{cur}} \cdot \delta t + 0.5 \cdot \mathbf{u}(t) \cdot \delta t^2$$

$$\mathbf{v}_{\text{cur}} \leftarrow \mathbf{v}_{\text{cur}} + \mathbf{u}(t) \cdot \delta t$$

        Check constraints:

        if $\mathbf{x}_{\text{cur}} \notin [\mathbf{x}_{\min}, \mathbf{x}_{\max}]$ or $\mathbf{v}_{\text{cur}} \notin [\mathbf{v}_{\min}, \mathbf{v}_{\max}]$, set valid = **False**

        if valid == **False then**
            break
        end if
    end for
    if valid == **True then**
        Add $(\mathbf{x}_{\text{cur}}, \mathbf{v}_{\text{cur}}, \mathbf{W}_{\text{cur}}, \mathbf{x}_{\text{ref}}, \mathbf{W}_{\text{ref}})$ to $\mathcal{O}_\infty$
    end if
end for
return $\mathcal{O}_\infty$

E. Details on MIQCP, High-Level Planner

We define exploration over a discretized $N_{\text{grid}} \times N_{\text{grid}}$ map using binary variables $z(t, i, j)$ that equal 1 if cell (i, j) is visited at time t. Constraints (2)–(4) in Table IV of Appendix aggregate these visits into $V(i, j)$, a binary indicator of whether a cell is visited at any point in the horizon.

The robot's planar path is represented by continuous variables $x(t, 0)$ and $x(t, 1)$ for the x and y grid coordinates. Constraints (5)–(6) enforce consistency between $z(t, i, j)$ and position using a big-M formulation (Schouwenaars et al. [9]), where M is chosen sufficiently large. For example, if $z(t = 5, i = 1, j = 2) = 1$ then $x(t = 5, 0) = 1$ and $x(t = 5, 1) = 2$.

State propagation is given by constraint (7): $x(t + 1) = x(t) + d(t)$, where the step $d(t)$ depends on the active mode. Mode constraints (8)–(10) define allowable step sets: $m_1(t)$ for biped motion with $d(t) \in \{-1, 1\}$, $m_2(t)$ for crawling with $d(t) \in \{-2, -1, 1, 2\}$, and $m_3(t)$ for rolling with $d(t) \in \{-3, 3\}$ to model fixed-distance rolls. Constraint (11) enforces a one-hot mode choice, so only one mode is active at any time. Since rolling induces unreliable state estimation, we do not count intermediate cells between the roll start and end as "seen."

1) Terrain and Obstacle Constraints: Terrain regions are modeled as circles. Constraint (12) computes squared distance to each circle center. Constraints (13)–(14) use a binary indicator $b^k_{\text{circle}}(t)$ to activate when inside the circle, with $\epsilon > 0$ enforcing strict separation. Constraint (15) links terrain to mode admissibility, for example requiring crawling ($m_2(t) = 1$) on rough terrain or biped ($m_1(t) = 1$) in elevated-observation regions such as near tables.

Obstacles are modeled as rectangles distinct from terrain circles for clarity. Constraints (16)–(20) ensure the robot never enters these forbidden regions, using binary indicator $b^{\text{rect}}_k(t)$. Where $x_{\min/\max}$ and $y_{\min/\max}$ define the rectangle boundaries. Circular obstacles can also be represented, but we separate shapes here to distinguish terrain from obstacles.

An analysis of optimal hyper-parameter selection is given in Table V of Appendix.

F. Additional Force Control Pull-Up Results and Description

In Fig. 5a of Appendix, we present the force manipulability ellipsoid of the robot arm at the initial grasp configuration, corresponding to the hanging phase depicted in panel 3 of (E). This ellipsoid, defined in task space, spans three axes: torque about the surface normal (z-axis), pulling force perpendicular to the surface (x-axis), and tangential force along the handle (y-axis). It characterizes the set of wrenches (forces and moments) that the end-effector can produce under unit joint torques. The pronounced elongation of the ellipsoid along the x-axis indicates a high capability to generate pulling forces—critical for initiating and sustaining upward motion during climbing. In contrast, the shorter extent along the z-axis reflects a reduced ability to apply torque about the surface normal. This geometry reveals that the arm's configuration is optimized for exerting strong pulling forces rather than rotational control, aligning well with the functional requirements of the climbing maneuver. Figs. 5b-d of Appendix show the end-effector wrench and position response during the pull-up maneuver illustrated in Fig. 5e of Appendix. Specifically, Fig. 5b of Appendix plots the moment about the gripper normal (z-axis), which varies from approximately $-3\,\text{N}\,\text{m}$ to $5\,\text{N}\,\text{m}$, while moments in the orthogonal axes remain near zero. Fig. 5c of Appendix shows the end-effector Cartesian position with reference trajectories: $0.3\,\text{m}$ in y (vertical), $-0.15\,\text{m}$ in x, and $-0.12\,\text{m}$ in z. The y-axis exhibits a characteristic down–up motion as the robot is first pushed downwards by an external force, and then actively pulls itself upward, returning to the target pose. A small offset in z is also observed, caused by the admittance controller balancing between position tracking and a zero-force command along that axis, which it cannot satisfy simultaneously. Fig. 5d of Appendix plots the end-effector forces, where the external disturbance manifests as a downward force of approximately $15\,\text{N}$ in y. As the robot completes the pull-up and stabilizes at the reference position, the measured force in all directions approaches zero. This experiment highlights the robot's ability to absorb external perturbations through passive compliance

Optimal Hyper-Parameter Selection (Top 2) using map from Fig. 7 of main text.

W_{exp}	W_{goal}	ϵ	M	T	Comp. Time (s)	Mode Switch Rate	Energy/Visited Cells (J/cell)
20	-20	0.0001	1000	20	10.67	0.27	34.91
200	-200	0.0100	16000	20	11.20	0.22	35.41

Optimal Hyper-Parameter per Mode Selection using simplified version of the map from Fig. 7 of main text.

Mode	W_{exp}	W_{goal}	ϵ	M	T	Comp. Time (s)	Energy/Visited Cells (J/cell)	Failure Count
Biped	100	-50	0.001	8000	40	78.86	26.89	270/324
Crawl	50	-1	0.001	8000	20	4.16	5.11	160/324
Roll	1	-1	0.001	8000	15	1.73	21.96	135/324

TABLE V: Top table shows the optimal parameter selection, from a combination of parameters (324 in total) shown in Table III of main text, for the top 2 performing combinations using the example map shown in Fig. 7 of main text. Performance is evaluated by lowest Energy/Visited Cell. In this test, the planner is able to choose between biped, crawling, and rolling mode. In the bottom table, we remove all terrain from the map, but still include the obstacles. We then force the solver to be either in only Biped, Crawling, or Rolling mode to reach the goal state and show the optimal parameters that achieve the lowest Energy/Visited cell. Note, failure count means how many times the solver failed when exploring the different combination of hyper-parameters per mode.

TABLE VI: Energy comparison per meter of travel. COM heights are 0.35 m (biped) and 0.20 m (crawling). Biped incurs higher potential energy costs due to its elevated posture, while crawling distributes effort across more limbs. Rolling takes a lot of initial effort as it has to go kick back with as much force as possible to roll backwards.

Locomotion Type	**KE (J)**	**PE (J)**	**Energy per 1 m (J/m)**
Biped	1.35	102.82	104.17
Crawling	6.00	58.86	64.86
Rolling	70.00	0.00	70.00
Pull-up	3.00	58.86	61.86
Climbing	0.60	294.30	294.90

and then execute a coordinated whole-body recovery using force-aware position control.

G. Details on State Estimation

We have two sources of state estimation. The first comes from a T265 IntelRealsense camera, attached to the back of the robot and provides on-board state estimation at 200 Hz using visual-inertial odometry SLAM (i.e., VIO-SLAM). The second source comes from using OptiState (Schperberg et al. [10]), which uses a Kalman filter that leverages joint encoder and IMU measurements, enhanced by a single-rigid body model incorporating ground reaction force outputs from convex Model Predictive Control (MPC). The state is further refined using Gated Recurrent Unites, integrating semantic insights and robot height from a Vision Transformer autoencoder applied to depth images using a IntelRealsense D435i camera, attached to the head of the robot. The estimations from the T265 camera and OptiState are fused using a simple complementary filter. Note, the performance of VIO-SLAM from the T265 camera, as well as the details of OptiState are demonstrated in Schperberg et al. [10].

H. Visual-Servo Controller

The visual-servo component, as shown in Fig. 6 of Appendix, is used primarily for positioning the arms of the robot to grasp the handles of the slide, to help initiate the climbing trajectory. To accomplish this, we first annotate approximately 200 images with bounding boxes for both side handles of the slide. We then train a YOLOv8 (Redmon et al. [8]) model on these annotations. During inference, the bounding boxes of each slide handle and their center positions, or $\mathbf{p}^{\text{cam}}$, are passed into a PID controller running at 300 Hz, with the goal of reducing the error between the current gripper position, $\mathbf{p}_{\text{cur}}$ and $\mathbf{p}^{\text{cam}}$. To reduce jerk, and promote smooth motions, we apply a low-pass filter on the output of our PID controller, before running inverse kinematics to pass the joint angles to our actuators. The objective is that distance between the robot's two grippers match the distance of the handle bar slide, and ensuring the gripper center position is centered at the position of $\mathbf{p}^{\text{cam}}$.

1) Visual Servoing for Kids Slide: In Fig. 8 of Appendix, we show the pre-manipulation phase, where the robot has already transitioned into biped mode and must actively position its grippers at the center point of the ladder bar (left and right ladder handles). The bounding boxes (green) and the left and right center points from the handle segmentation (in purple) are obtained using YOLOv8. These handle center points serve as reference targets for the end-effector gripper positions. A PID controller, combined with a low-pass filter for smoother control, is used to track these references, as illustrated in Fig. 6 of Appendix.

To demonstrate the robustness of the controller, we manually moved the slide back and forth. The centroid positions from the vision pipeline are shown in green for the right ladder handle and in blue for the left ladder handle. The measured end-effector positions, computed using joint angles and forward kinematics, are shown in purple for limb 0 (right

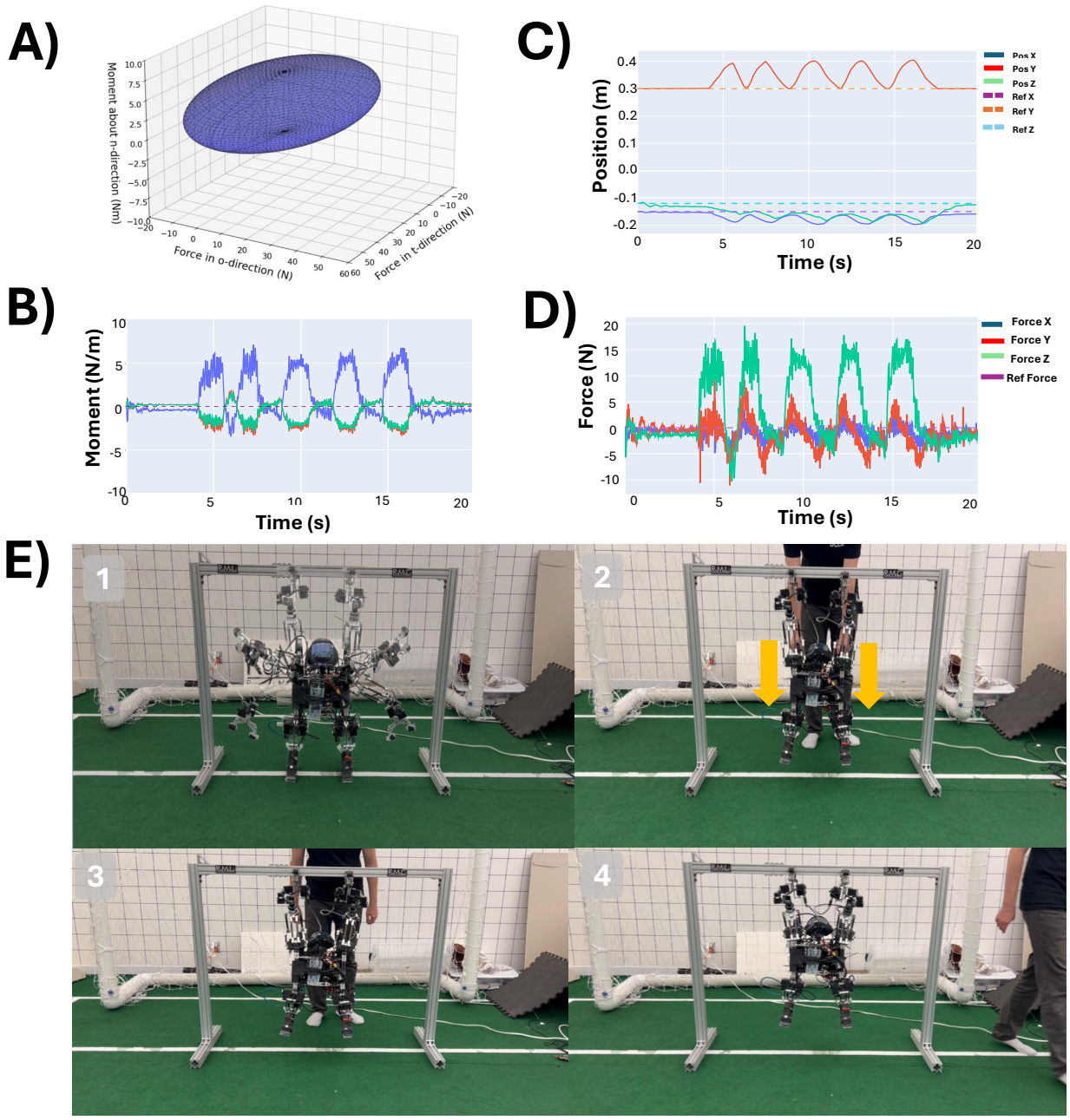


Fig. 5: In (A) we show the force manipulability ellipsoid at the initial grasp position shown in (E)-4 (frame definition given in Fig. 7 of Appendix). The moment about surface normal in (B), position of the end-effector relative to the reference (or initial pose) in (C) given to the admittance controller, and the forces in (D). In (E) we show the entire motion, along with the disturbance from a human pushing the robot downwards as seen in (2).

arm) and orange for limb 3 (left arm). As observed, the end-effectors were able to track the ladder handles even under external disturbances. Notably, large disturbance spikes—such as the one shown in green at approximately 1.2 seconds—were successfully suppressed due to the low-pass filter.

On average, response time during rise (going from 10 % to approx. 90 %) is ≈0.5 seconds, and settling time is about 1.75 seconds, which is confirmed by calculating a damping ratio of 0.7 (calculated using our PID parameters, where $K_p = 3.0$, $K_i = 4.5$, and $K_d = 1.0$). Note, that these response times are naturally also affected by the values set by the low-pass filter, which we purposefully calibrated to encourage smoothness (i.e., non-jerky movements) over perfect tracking a moving reference and fast response times, as our objective is to track a largely stationary object.

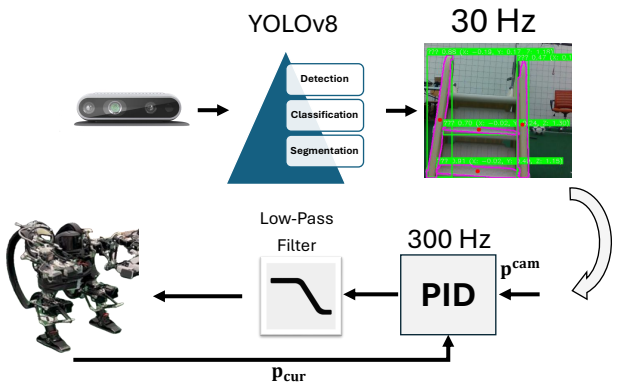


Fig. 6: A visual servo pipeline is used to track the handle bars of a slide before initiating the climbing task. It consist of the YOLOv8 algorithm along with a PID controller for tracking the centroids of the bars (red dots) and low pass filter for smoothness.

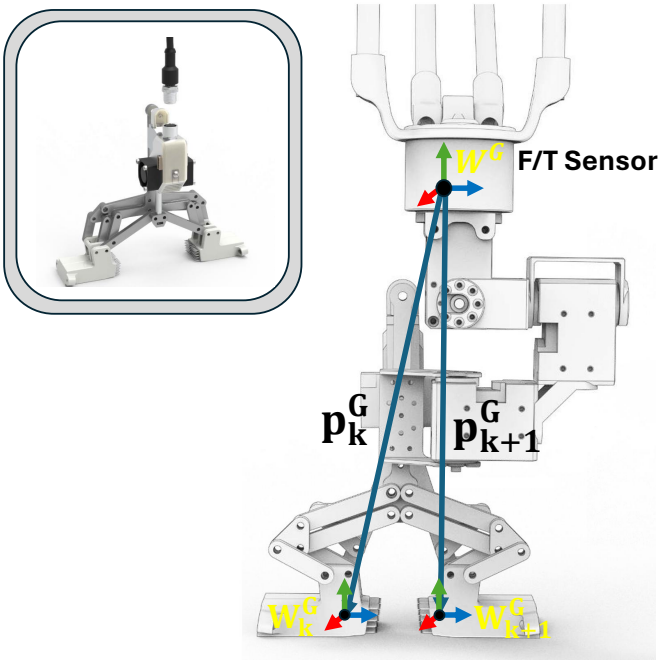


Fig. 7: We provide an illustration of the gripper, along with our frame definitions of the wrench at the wrist, $\mathbf{W}^G$, and the wrenches of the finger tips, $\mathbf{W}^G_k$ and their respective positions $\mathbf{p}^G_k$. for finger tip k.

I. Mode Transition: Standup Maneuvers

MOBIUS's standup motions are vital since they are used for transit among modes as well as fall recovery. From the prone pose, i.e., when the robot falls forward in the bipedal mode, MOBIUS can go into the crawling pose using the arms as shown in Fig. 9a of Appendix From the crawling to the bipedal mode, MOBIUS uses the arms to lift the body in Fig.

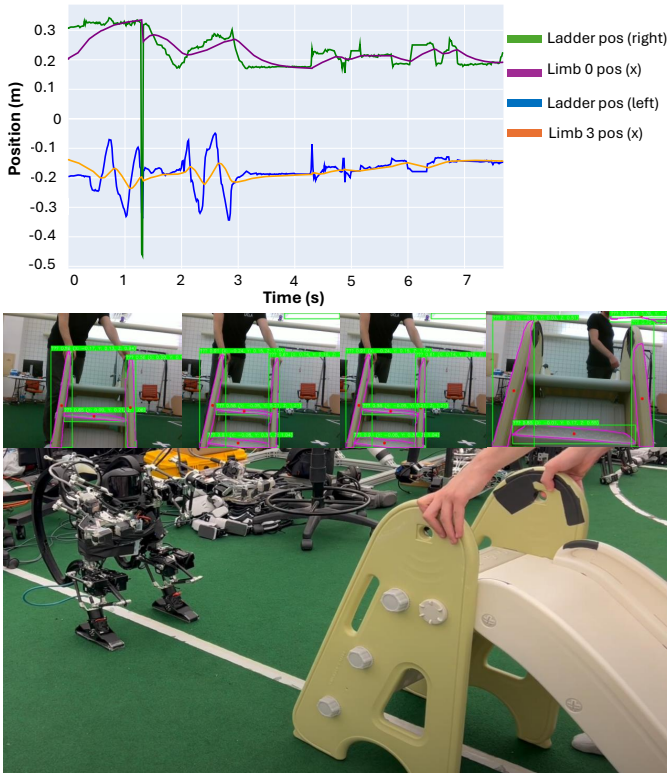
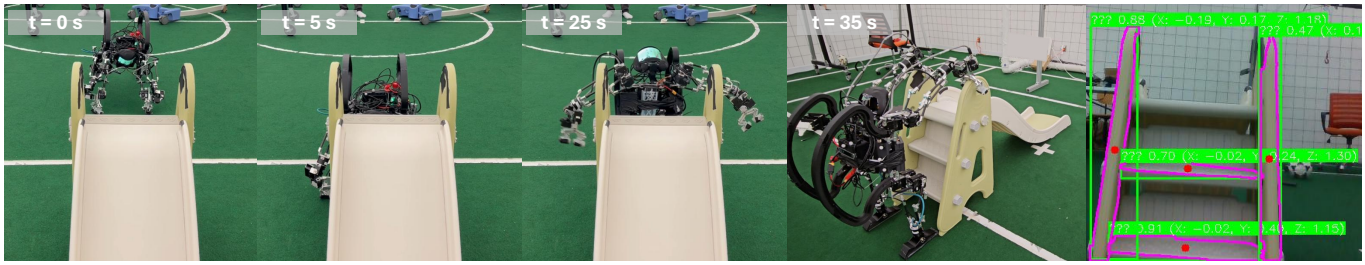


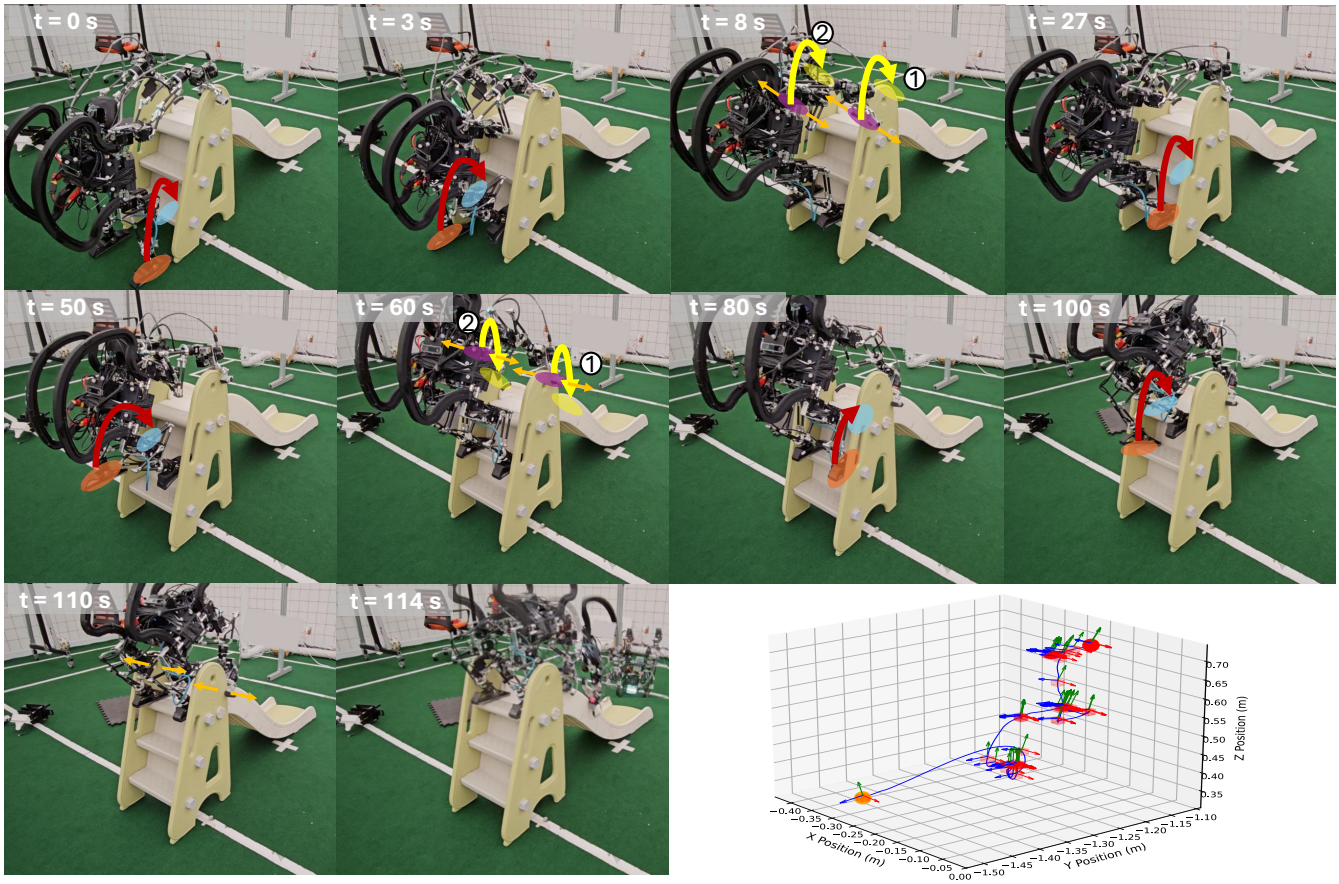
Fig. 8: Example results of the visual-servo controller described in Sec. H. Ladder pos is the centroid position marked by the red dot for the right and left ladder handles respectively. Limb 0 and Limb 1 are the left and right end-effector gripper positions (using x-axis as example).

9b of Appendix. However, the arms kinematically cannot reach the ground to push the body completely upward, and thus, the arms are lifted off the ground to move the center of gravity backward. The simulation in Fig. 9c of Appendix visualizes the change in the center of mass over the corresponding sequence of key points in Fig. 9b of Appendix. In the supine pose, MOBIUS lands on the feet naturally and can stand up without using arms due to the curved back rails. In another case, MOBIUS can land on the head, in which the arms are used to transit to the prone posture instead in the similar way as in the rolling shown in Fig. 9c of Appendix.

Fig. 9: We present the following motions generated by the offline trajectories visualized on the right of the figure, with limbs that contain grippers in blue and magenta, and limbs with flat-foot end-effectors in red and green. These trajectories include **(A)** falling to crawling mode; **(B)** biped to crawling mode; **(C)** crawling into rolling; and **(D)** crawling to biped mode.



(a) Pre-climbing phase: MOBIUS approaches the slide in crawling model, transitions to biped mode, and then uses the visual-servo controller to align its arms with the handle bars. Once aligned, MOBIUS can now initiate the climbing phase, illustrated in (b).



(b) MOBIUS now climbs the stair case, first grasping the top of the handle bars with its arms. It then lifts the right foot and then the left foot to the first step at t=3 sec, and releases one arm at a time, so it can grasp further to the handle bar of the slide at t=8 sec. The process repeats until at t=110 sec, where the robot releases both grippers and naturally can roll down the slide using the back rails to roll down.

Fig. 10: The climbing mode is described in (a) and (b). Where (a) is the pre-climbing phase, and (b) illustrates climbing the slide. In the bottom right figure, we also show a 3D plot of the CoM trajectory of the robot as it goes towards and traverse the slide.

References

[1] Xiaoyu Chen, Jiachen Hu, Chi Jin, Lihong Li, and Liwei Wang. Understanding domain randomization for sim-to-real transfer, 2022. URL https://arxiv.org/abs/2110.03239.

[2] Jared Di Carlo, Patrick M Wensing, Benjamin Katz, Gerardo Bledt, and Sangbae Kim. Dynamic locomotion in the mit cheetah 3 through convex model-predictive control. In *2018 IEEE/RSJ international conference on intelligent robots and systems*, pages 1–9. IEEE, 2018.

[3] Yuankun Jiang, Chenglin Li, Wenrui Dai, Junni Zou, and Hongkai Xiong. Variance reduced domain randomization for reinforcement learning with policy gradient. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 46(2):1031–1048, 2024. doi: 10.1109/TPAMI.2023.3330332.

[4] Paras Kumar, Harish Hirani, and Atul Agrawal. Fatigue failure prediction in spur gear pair using agma approach. *Materials Today: Proceedings*, 4(2):2470–2477, 2017.

[5] Gitin M Maitra. Handbook of gear design. 1994.

[6] W Moćko. The influence of stress-controlled tensile fatigue loading on the stress–strain characteristics of aisi 1045 steel. *Materials & Design*, 58:145–153, 2014.

[7] Sungyong Park, Jigang Kim, and H. Jin Kim. Zero-shot transfer learning of a throwing task via domain randomization. In *2020 20th International Conference on Control, Automation and Systems (ICCAS)*, pages 1026–1030, 2020. doi: 10.23919/ICCAS50221.2020.9268312.

[8] Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. You only look once: Unified, real-time object detection. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 779–788, 2016. doi: 10.1109/CVPR.2016.91.

[9] Tom Schouwenaars, Bart De Moor, Eric Feron, and Jonathan How. Mixed integer programming for multi-vehicle path planning. In *2001 European Control Conference (ECC)*, pages 2603–2608, 2001. doi: 10.23919/ECC.2001.7076321.

[10] Alexander Schperberg, Yusuke Tanaka, Saviz Mowlavi, Feng Xu, Bharathan Balaji, and Dennis Hong. Optistate: State estimation of legged robots using gated networks with transformer-based vision and kalman filtering. *arXiv preprint arXiv:2401.16719*, 2024.

[11] Andrew Schwartzwald and Nikolaos Papanikolopoulos. Sim-to-real with domain randomization for tumbling robot control. In *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 4411–4417, 2020. doi: 10.1109/IROS45743.2020.9341057.

[12] Aidar Shakerimov, Tohid Alizadeh, and Huseyin Atakan Varol. Efficient sim-to-real transfer in reinforcement learning through domain randomization and domain adaptation. *IEEE Access*, 11:136809–136824, 2023. doi: 10.1109/ACCESS.2023.3339568.

[13] Yusuke Tanaka, Yuki Shirai, Xuan Lin, Alexander Schperberg, Hayato Kato, Alexander Swerdlow, Naoya Kumagai, and Dennis Hong. Scaler: A tough versatile quadruped free-climber robot. In *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 5632–5639, 2022. doi: 10.1109/IROS47612.2022.9981555.