Locomotion Skills for Reconfigurable Hexapod Robots

Tomson Qu

Electrical Engineering and Computer Sciences
University of California, Berkeley

Technical Report No. UCB/EECS-2025-150
http://www2.eecs.berkeley.edu/Pubs/TechRpts/2025/EECS-2025-150.html

August 11, 2025

Locomotion Skills for Reconfigurable Hexapod Robots

by

Tomson Qu

A thesis submitted in partial satisfaction of the

requirements for the degree of

Master of Science

in

Electrical Engineering and Computer Science

in the

Graduate Division

of the

University of California, Berkeley

Committee in charge:

Professor Avideh Zakhor, Chair
Professor Ken Goldberg

Spring 2025

Locomotion Skills for Reconfigurable Hexapod Robots

by Tomson Qu

Research Project

Submitted to the Department of Electrical Engineering and Computer Sciences, University of California at Berkeley, in partial satisfaction of the requirements for the degree of **Master of Science, Plan II**.

Approval for the Report and Comprehensive Examination:

Committee:

Professor Avideh Zakhor
Research Advisor

7/28/2025

(Date)

* * * * * * *

DocuSigned by:
32B42AC2D1E3498...

Professor Ken Goldberg
Second Reader

8/11/2025

(Date)

Locomotion Skills for Reconfigurable Hexapod Robots

Abstract

Locomotion Skills for Reconfigurable Hexapod Robots

by

Tomson Qu

Master of Science in Electrical Engineering and Computer Science

University of California, Berkeley

Professor Avideh Zakhor, Chair

Hexapod robots are useful for carrying out tasks in cluttered environments since they are stable, compact, and lightweight. They also have multi-joint legs and variable-height bodies, making them suitable candidates for tasks such as stair climbing and squeezing under objects in a typical home environment or an attic. Expanding on previous work on joist climbing in attics, we train a legged hexapod equipped with a depth camera and visual-inertial odometry (VIO) to perform four key locomotion tasks: (1) climbing stairs, (2) avoiding obstacles, (3) squeezing under obstacles such as a table, and (4) squeezing between narrow vertical gaps such as between furniture.

Our policies are trained with simulation data only and are deployed on low-cost hardware that does not require real-time joint state feedback. We train a teacher-student model in two phases: In phase 1, we use reinforcement learning with access to privileged information such as height maps and joint feedback. In phase 2, we use supervised learning to distill the model into one with access to only onboard observations, consisting of egocentric depth images and robot pose captured by a tracking VIO camera. By manipulating available privileged information, constructing simulation terrains, and refining reward functions during phase 1 training, we are able to train the robot with skills that are robust in non-ideal physical environments. We demonstrate successful sim-to-real transfer and achieve high success rates across all four tasks in physical experiments.

We have integrated an adjustable-height camera mount that enables the robot to squeeze under low obstacles such as a table or a sofa. This additional degree of freedom allows our policies to automatically reduce the robot's height by lowering the camera making it more compact during squeezing tasks. This enables the robot to squeeze under realistic furniture at home, such as a table or sofa.

Contents

List of Figures

List of Tables

Acknowledgments

I want to express my gratitude to my advisor Prof. Avideh Zakhor for her support through out the years. I'm also very fortunate to receive guidance from Wenhao Yu from Google Deepmind. I want to thank Jason Zang's for his previous inspiring work in hexapod joist climbing. Aarav's work in hardware design is also invaluable.

Chapter 1

Introduction

In this chapter, we discuss the background and motivation of our work in Section 1.1 and the related work in Section 1.2.

1.1 Hexapod Robot and Locomotion Skills

Lightweight legged robots are ideal platforms for navigating cluttered home environments, where the robot must maneuver around large objects such as refrigerators, squeeze under low objects such as couches and beds, and climb staircases to move from one floor to the next. They can also be useful in rough environments such as attics, as shown in Figure 1.1, which are full of joists and are uncomfortable and potentially dangerous for human workers to

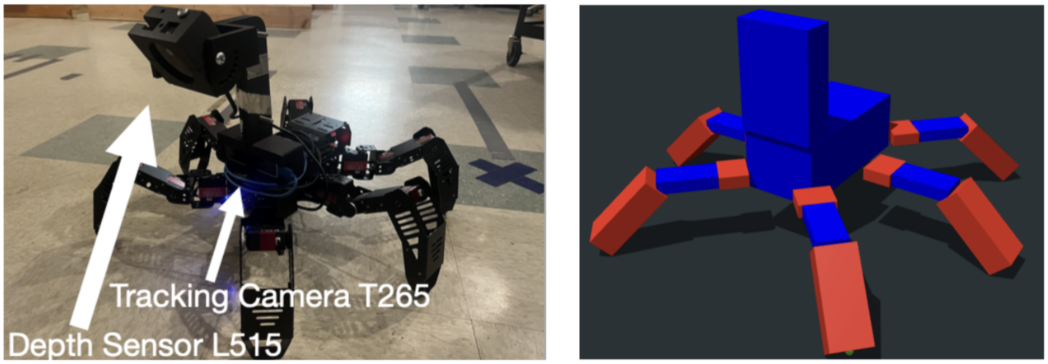
Figure 1.1: An example of a cluttered attic.

vacuum and air seal before adding insulation [49]. For example, since attics typically consist of multiple rows of joist structures, a human worker could easily fall through the attic floor and get seriously injured if they step on the sheetrock between two joists by mistake. Also, attics are typically low, which means human workers have to crawl on their stomachs to get around.

There has been a significant amount of work on quadrupeds and bipedal robot locomotion [1, 5, 15, 17, 20, 22, 23, 24, 25, 34, 36, 41, 42, 43, 46, 47, 54]. In our prior work [50], we developed methods to enable hexapods to climb joists in harsh attic environments. In this work, we extend that prior work to include stair climbing, obstacle avoidance, squeezing under and between objects for a hexapod robot. We focus on hexapods for two main reasons. First, hexapod robots can be more stable and lightweight than quadrupeds and humanoids of similar size. Second, bipedal or quadruped robots are often taller than hexapods and are therefore less suitable for traversing tight spaces such as the corners of attics.

To facilitate the practical usage of robots in the retrofit business, it is important for legged locomotion controllers to work with low-cost hardware. However, most existing legged locomotion systems require high-end robots capable of real-time sensing of joint states, which can ultimately result in expensive hardware costing thousands, if not tens of thousands, of dollars. For example, model predictive control methods [37] require powerful computational resources and real-time joint feedback from expensive robot platforms, and they often compromise real-time performance when incorporating more complex dynamics. Data-driven methods [2] can work with limited computational resources and are robust to a variety of perception failures but still require fast joint state feedback. Many low-cost robots are not equipped with powerful onboard computation or real-time feedback, such as joint torque and angle sensing, which are accessible on more expensive platforms. Meanwhile, humans without leg-sensing feedback, when equipped with prosthetics, can walk and even participate in competitive sports using only egocentric visual perception and a sense of body orientation [29].

In this thesis, we propose an end-to-end learning-based perceptive controller for low-cost, sub-thousand-dollar hexapods to autonomously climb staircases, avoid obstacles, and squeeze under and between objects, demonstrating zero-shot sim-to-real transfer in real environments. In addition to attics, these skill sets are useful for robots navigating inside homes filled with furniture, where the robot must maneuver around and squeeze under and between obstacles and climb staircases to move from one floor to the next. Our robot is a $600 SpiderPi robot by Hiwonder [14] with no real-time joint feedback, and shown in Figure 1.2, and equipped with an Intel L515 depth camera and an Intel T265 tracking camera with a customized camera mount. Similar to the approach proposed in [50], we use a two-stage teacher-student training procedure to learn models that can operate without real-time joint feedback: the first stage involves reinforcement learning (RL) with access to privileged observations, and the second stage uses supervised learning to distill the model using only onboard observations, including body pose and egocentric depth images. Since optimal stair climbing, squeezing, and obstacle avoidance motions are fundamentally different from walking, we train our controllers without human-defined prior gait knowledge, guiding the



(a) Physical Robot

(b) URDF

Figure 1.2: Physical and URDF of the robot. *(a)* The hexapod robot standing at the reset position - roughly 37 cm tall. *(b)* The URDF of the hexapod.

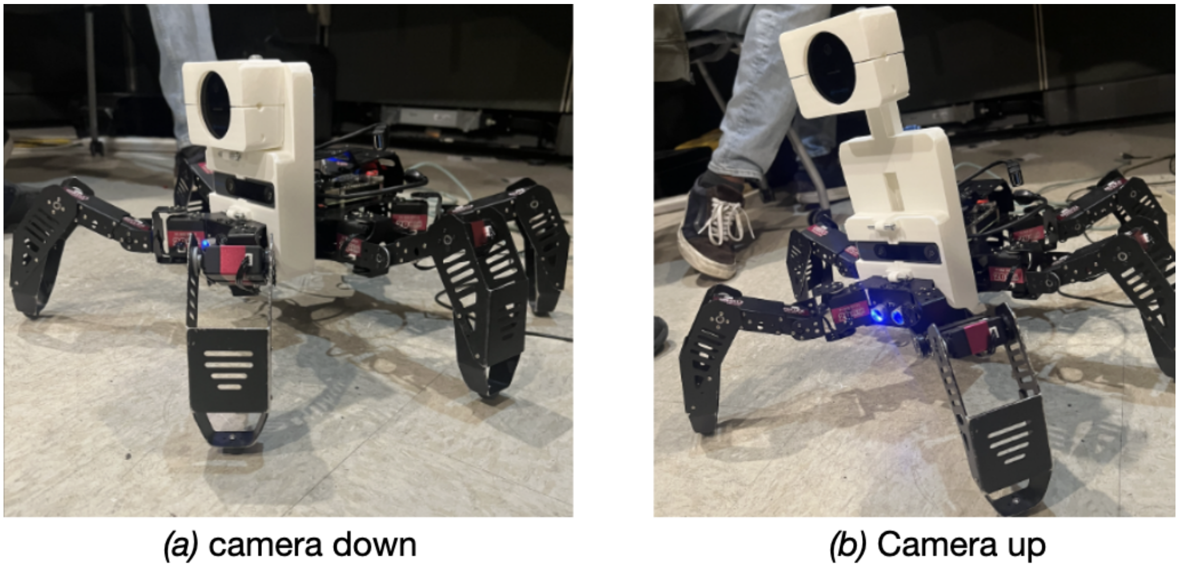
models to explore task-appropriate motions.

Through extensive simulations and physical experiments, we show that our low-cost robot successfully learns the following four skills and generalizes across different terrains: (a) climbing up and down staircases with as many as 15 steps, including a landing pad; (b) squeezing under low objects, regardless of whether the objects are long or short; (c) maneuvering right and left around multiple obstacles without scraping or touching them, and continuing in the same direction as before each obstacle; and (d) squeezing between tight spaces without getting stuck. With proper design of terrains, reward functions, and the choice of privileged information in the simulation environment, our model is able to learn a variety of skills. After training in simulation, the policy is deployed to the physical robot zero-shot, with physical experiments suggesting that the learned skills are robust.

We further enhance the robot's squeezing capability through the design of an adjustable camera mount that can dynamically change its height, as shown in Figure 1.3. This mechanism introduces an additional degree of freedom, allowing the robot to reduce its overall height by up to 12 centimeters by lowering the camera. This adjustment is critical for the robot to squeeze under realistic obstacles such as a bed or a chair. We demonstrate that our policy automates the camera's up-and-down decision-making process and operates robustly, even though the camera movements were not explicitly trained in simulation. This highlights the generalization capability of our learned squeezing policy.

1.2 Related Work

In this section, we review prior work in four areas. We begin by (1) examining the limitations of traditional obstacle avoidance algorithms when applied to complex terrains, (2) reviewing locomotion control strategies for legged robots, (3) exploring the application of reinforcement



(a) camera down

(b) Camera up

Figure 1.3: The hexapod robot standing at the reset position with camera *(a)* down - roughly 25 cm tall. *(b)* up - roughly 37 cm tall.

learning to challenging tasks, and (4) discussing reconfigurable robotic systems and adaptive perception mechanisms.

Traditional Obstacle Avoidance Methods in Complex Terrains

Traditional obstacle avoidance algorithms have provided foundational techniques for navigation-related tasks. Well-known algorithms have demonstrated strong performance in such tasks. The Dynamic Window Approach (DWA) operates by sampling the robot's velocity space, constrained by its dynamic capabilities, to select motions that are safe for navigation [8]. Another method, the Vector Field Histogram (VFH), constructs a histogram representing the density of nearby obstacles and guides the robot through available gaps [4]. While it performs well in many obstacle avoidance scenarios, it can fail in narrow spaces, making it suboptimal for use in cluttered environments. Bug algorithms such as Bug1 and Bug2 direct a robot to follow the boundaries of obstacles until a path becomes visible [7]. Finally, graph-based methods such as A* and Dijkstra rely on complete maps of the environment to plan optimal paths [13].

Although these traditional methods have shown strong performance in specific tasks, they exhibit significant drawbacks and inefficiencies when faced with complex terrains such as attics or staircases. For example, many of these algorithms require pre-existing maps and struggle to generalize to unfamiliar or dynamically changing environments. Tasks requiring adaptive body configurations—such as squeezing between furniture or under tables—are generally beyond the capabilities of these classical methods. These limitations make them less suitable for application in real-world complex environments. In contrast, reinforcement learning enables robots to acquire task-specific policies that integrate perception and control for effective navigation in such challenging settings.

Locomotion Control

Prior work has focused on specific control methods to achieve basic locomotion abilities in hexapod and quadruped robots. Researchers have employed two-layer Central Pattern Generator (CPG) networks and posture control strategies based on force distribution and compensation to enable robot locomotion across a variety of terrains [28]. In previous work, Zang et al. [51] successfully enabled a hexapod to climb over joist terrains using a teacher-student model and an actor-critic reinforcement learning strategy [50].

Deep Reinforcement Learning (DRL) has become a promising approach for developing autonomous and complex behaviors in real-world systems. Many researchers choose to test the performance of their systems in simulation environments before applying them to real-world applications. However, the sim-to-real gap poses significant challenges in transferring simulated learning to real-world scenarios [18, 30]. Solutions such as system identification, domain randomization, domain adaptation, imitation learning, meta-learning, transfer learning, and knowledge distillation have shown promise in narrowing this gap, enabling more effective deployment of robotic systems in physical environments [16, 18, 38, 53]. Rizzardo proposed a sim-to-real technique that trains a Soft Actor-Critic agent together with a decoupled feature extractor and a latent-space dynamics model, enabling transfer without retraining or fine-tuning [30]. Tiboni introduced DROPO, a novel method for estimating domain randomization distributions for safe sim-to-real transfer [39].

Model-free reinforcement learning has emerged as a pivotal approach for achieving different locomotion tasks, such as hopping and crawling, in various environments—both terrestrial and aquatic, with barriers or gaps [3, 19, 32, 35, 40, 48, 55]. Special network designs have been introduced to facilitate the training of the locomotion policy. [40] proposed learning low-level motion from a biological dog first and then learning high-level tasks in order to save training time. All these policies were combined into a single framework, allowing the robot to autonomously select and execute the appropriate policy. Kareer et al. [19] developed a two-layer architecture consisting of a visual navigation layer, which outputs angular velocity commands, and a visual locomotion layer, enabling a quadruped robot to step over scattered terrain toward a target destination. [48] proposed a hierarchical structure with a high-level vision policy and a low-level motion controller, enabling a quadrupedal robot to traverse uneven environments effectively.

Reinforcement learning

In order to show the efficacy of the policy employed on the robot, physical experiments are imperative. Traditional methods such as the Dynamic Window Approach (DWA) [8] are effective in obstacle avoidance for robotics. Actor-critic reinforcement learning-based avoidance methods have been used to enable robots to avoid scattered obstacles [6, 12]. [21] proposed training ANYmal robots with reinforcement learning in simulation and deployed the policy to operate in challenging natural environments. [20] presents an approach to teach a quadruped robot to adapt and conquer unseen environments with a base policy and an adaptation module. Researchers introduced a general DRL framework for obstacle avoidance and incorporated a manipulability index into the reward function in order to avoid joint singularity while executing tasks [31]. The FAM-HGNN framework, which relies on an attention mechanism within a heterogeneous graph neural network, presents a novel solution for the obstacle avoidance problem in RL. This approach surpasses the performance of both multi-layer perceptron-based and existing GNN-based RL methods [52].

With regards to squeezing, researchers enabled the reconfigurable robot RSTAR to squeeze through two adjacent obstacles, duck underneath an obstacle, and climb over an obstacle using the Q-learning algorithm [44].

In tasks involving climbing stairs, both Deep Deterministic Policy Gradients (DDPG) and Trust Region Policy Optimization (TRPO) were evaluated, with the latter showing superior performance [9]. Researchers used sim-to-real RL to achieve climbing by modifying an existing flat-terrain training framework to include stair-like terrain randomization, without any changes to the reward function [33]. Researchers also conducted experiments to enable different articulated, tracked robots [26] and assistive robots [27] to climb on slopes or stairs using machine learning techniques.

Reconfigurable Robots and Adaptive Perception Systems

The development of reconfigurable robots has emerged as a promising solution for enhancing versatility and agility in dynamically changing environments. Platforms such as Snapbot V2 and PolyBot showcase the capacity to alter reconfigurations on-the-fly to better suit task-specific locomotion requirements [10, 45]. Extending on those works, Hefty [11], an agricultural robot, demonstrates how reconfiguration can solve realistic tasks in the real world.

Inspired by those works, our system integrates a height-adjustable depth camera onto a hexapod platform. This design choice enables the robot to automatically lower its height when detecting obstacles, thereby achieving optimal performance in squeezing under them. This automatic reconfiguration introduces a critical degree of freedom, significantly enhancing the robot's ability to achieve compact traversal in cluttered environments such as houses and attics.

Chapter 2

Methodolgy and experiements

The outline of this chapter is as follows. In Section 2.1, we discuss the methodology for training stair climbing, obstacle avoidance, and squeezing under and between obstacles policies. In Section 2.2, we discuss the corresponding physical experiments.

2.1 Methodology

We train the robot to climb up/down stairs, avoid obstacles, squeeze under objects, and squeeze between objects. For each task, we train the corresponding policy in simulation using Isaac Gym, then directly deploy the trained policy onto the robot, which is equipped with an Intel RealSense Tracking Camera T265 for pose estimation and an L515 for depth estimation. Each task requires a different terrain construction, reward function, and camera angle, as shown in Tables 2.1 and 2.2. To further understand the reward terms, they can be grouped into two categories: (1) action-related terms in Table 2.3, and (2) environment-related reward terms in Table 2.4. We adopt a 2-stage student-teacher model, shown in Figure 2.1, similar to [50], in which the teacher policy is trained with privileged information such as joint feedback and a height map shown as the yellow pad in Figure 2.2. The teacher policy is then transferred to a student policy that takes in estimated pose and depth images as visual input using supervised learning. In the teacher model training, even though different tasks are trained with different reward terms, height map sizes, and terrain constructions, they are each distilled into a policy that takes a depth map of size 32×24 as visual input. During training, we control our robot by directly predicting the joint angles and applying them to the corresponding joints. The angles range from [-120, 120] degrees. Each joint is initialized to a resting position provided by the manufacturer, as shown in Figure 1.2.

The entire training process takes approximately 10 hours for the teacher policy and 5 hours for the student policy on an RTX TITAN GPU. The reward terms and weights for each of the five tasks—joist climbing, stair climbing, obstacle avoidance, squeezing under objects, and squeezing between objects—are shown in Tables 2.3 and 2.4. Different tasks require different number of training iterations to converge, as seen in the reward convergence

Reward Term	**Expression (Definition)**
Linear velocity in global x (forward)	$\text{clip}(v_x, \text{min} = -0.4, \text{max} = 0.4)$
Linear velocity in body y (left/right)	v_y^2
Global heading	θ^2
Angular velocity: yaw	ω^2
Ground impact	$\|f_t - f_{t-1}\|^2$
Collision penalty	$\mathbf{1}\{\text{coxa, femur, or base contacting terrain}\}$
Action rate	$\|a_t - a_{t-1}\|^2$
Action magnitude	$\|a_t\|^2$
Torques	$\|\tau\|^2$
Joint acceleration	$\hat{\ddot{q}}^2 = \left(\frac{\dot{q}_t - \dot{q}_{t-1}}{\Delta t}\right)^2$
Joint limit penalty	$\text{clip}(q_t - q_{\text{min}}, \text{max} = 0) + \text{clip}(q_t - q_{\text{max}}, \text{min} = 0)$
End effector height	$\|z_{\text{end\_effector}}\|$
Global y deviation	$\|y_{\text{current}} - y_{\text{start}}\|^2$
Distance to obstacle (front)	$f(H)$
Distance to obstacle (above)	$f(H)$
Distance to obstacle (to joints)	$f(H)$

Table 2.1: Reward Term Definitions

Task	**Camera Angle (degrees)**	**Height map size (m)**	**Height map location (m)**
Joist Climbing	30	0.6×0.8	0.3
Stairs Climbing	30	0.6×0.8	0.3
Obstacle Avoidance	30	0.6×1.0	0.6
Squeezing under Obstacles	0	0.6×0.8	0
Squeezing between Obstacles	0	1.2×0.8	-0.3

Table 2.2: Optimal camera angle and height map for each task.

curve shown in Figure 2.3. During model deployment, we take the depth image and pose estimate from our visual odometry system as input to our policy and output the joint angles for each of the 18 joints. We then send signals to each servo to set them to the desired angles.

The optimal camera angle for different tasks is shown in Table 2.2. Other than squeezing, which requires the optical axis of the camera to be parallel to the horizon, all other tasks work well with a 30-degree downward-looking inclination. This is expected, since the robot needs to look “up” rather than “down” to see the obstacle above it when squeezing under an object.

Reward Term	Joist	Stair	Obstacle	Squeeze under	Squeeze between
Action rate	-5^{-1}	-5^{-1}	-5^{-1}	-5^{-1}	-5^{-1}
Action magnitude	-1^{-2}	0	-1^{-2}	-1^{-2}	-1^{-2}
Torques	-1^{-3}	0	-1^{-3}	-1^{-2}	-1^{-2}
Joint acceleration	-1^{-5}	-1^{-5}	-1^{-5}	-1^{-5}	-1^{-5}
Joint limit penalty	-1^{0}	-1^{0}	-1^{0}	-1^{0}	-1^{0}
Ground impact	-1^{-1}	0	0	0	0
Collision penalty	-1^{0}	0	-3^{0}	-1^{0}	-1^{0}
Linear velocity (global x)	1^{2}	1^{2}	1^{2}	1^{2}	1^{2}
Linear velocity (body y)	-1^{1}	-1^{1}	-1^{1}	-1^{1}	-1^{1}
Global heading	-3^{1}	0	0	0	0
Angular velocity (yaw)	-1^{0}	0	-1^{0}	-1^{0}	-1^{0}
End effector height	-1^{-1}	0	0	0	0
Global y deviation	0	-1^{2}	-1^{0}	-1^{0}	-1^{0}

Table 2.3: Action-related reward terms and weights.

Reward Term	Joist	Stair	Obstacle	Squeeze Under	Squeeze between
Distance to obstacle (front)	0	0	-1^{-1}	0	0
Distance to obstacle (above)	0	0	0	-1^{-1}	0
Distance to obstacle (to joints)	0	0	0	0	2

Table 2.4: Environment-related and task-specific reward terms and weights.

The height map size and location for different tasks are also shown in Table 2.2 and visualized in 2.4. We have empirically found that obstacle avoidance requires a larger height map positioned farther from the robot than joist climbing and stair climbing.

Stair Climbing

We trained the robot to climb up and down staircases using the two-step teacher-student method described earlier. We deployed curriculum training to help the robot climb more challenging staircases by first learning easier ones. In particular, we let the riser heights increase from 4.5 cm to 18 cm and the tread depth decrease from 30 cm to 18 cm as the level of difficulty increased in curriculum training. In addition, we randomized the tread depth within a given staircase in simulation to improve the generalization of our policy.

The weight of the reward term 'Global y deviation' in Table 2.1 is an order of magnitude larger for stair climbing than for the other tasks. The main motivation for this is to prevent the robot from unnecessarily deviating to the right or left as it climbs a staircase. Intuitively, this term minimizes the lateral deviation of the robot from the direction of the axis it was

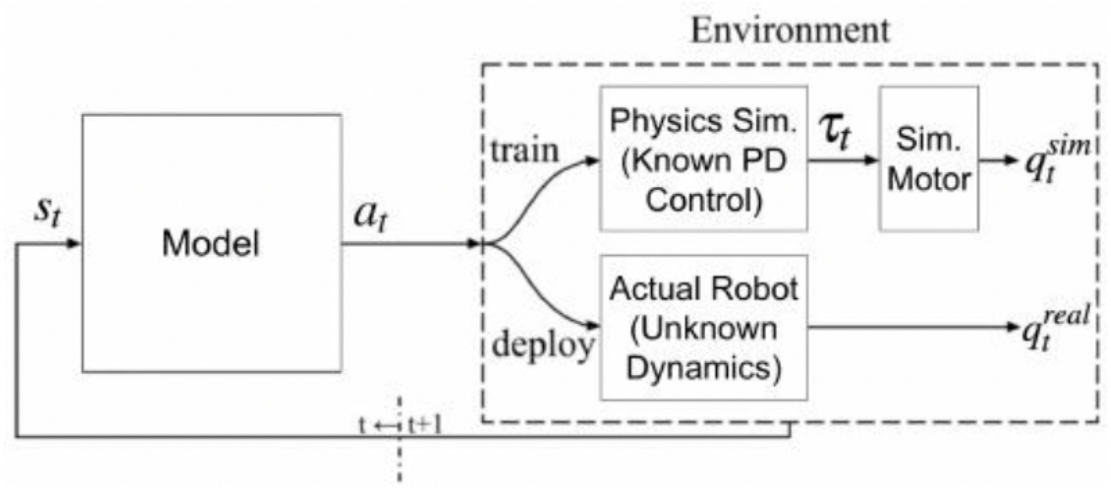


Figure 2.1: High-level overview of training methodology [50]

pointed toward before it starts climbing. We empirically found that a 30° tilt angle for the depth camera works well for both climbing up and down staircases.

Obstacle Avoidance

Even though traditional methods such as DWA [8] work well for obstacle avoidance, since our eventual goal is to combine the different RL skills into one, we have also developed an RL-based policy for obstacle avoidance. The most straightforward way to teach the robot to avoid collisions is by minimizing collisions between the robot's femur, coxa, and body during simulation, assigning large negative rewards when such collisions occur. This is shown as the "Collision Penalty" in Table 2.1 and indicates the number of collisions, where a collision is defined as an event with contact force larger than a force threshold of 0.1 N. We have empirically found that such a method results in the robot scraping by and touching obstacles as it tries to avoid them. To circumvent this, we use a reward term shown in the third-to-last row of Table 2.1, given by:

$$R(H) = -\sum_{i=1}^{M}\left(\sum_{j=1}^{N} h'_{ij}\cdot \overrightarrow{w_{1j}}\right)\cdot \overrightarrow{w_{2i}}$$

where $h'_{i,j}$ is the $(i,j)^{th}$ element of the binarized height map $M \times N$ matrix H' given by:

$$h'_{ij} = \begin{cases} 1 & \text{if } h_{ij} > 0 \\ 0 & \text{otherwise} \end{cases}$$

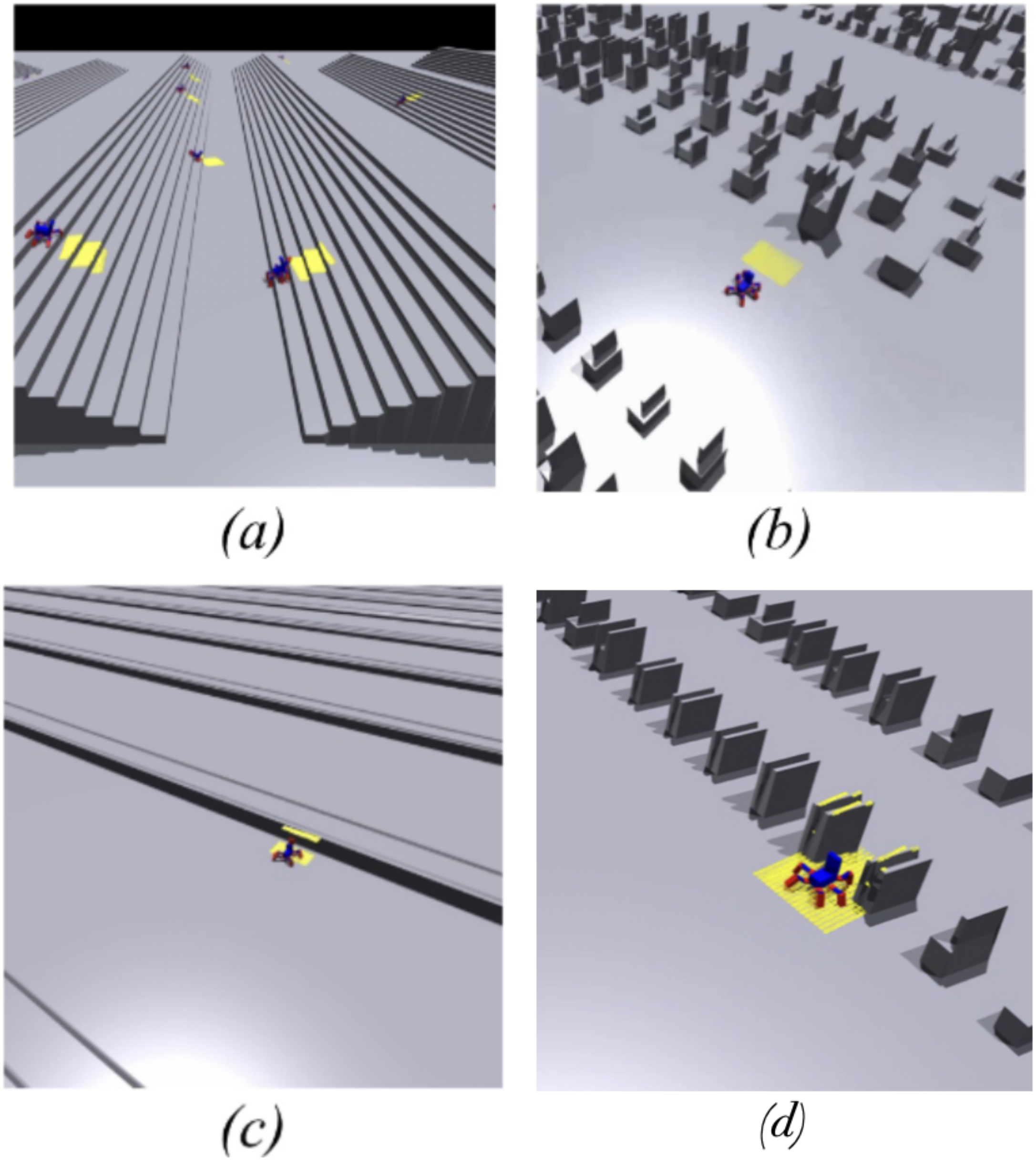


Figure 2.2: Simulation environment in Isaac Gym. *(a)* Stair climbing. *(b)* Obstacle avoidance. *(c)* Squeezing under obstacles. *(d)* Squeezing between obstacles.

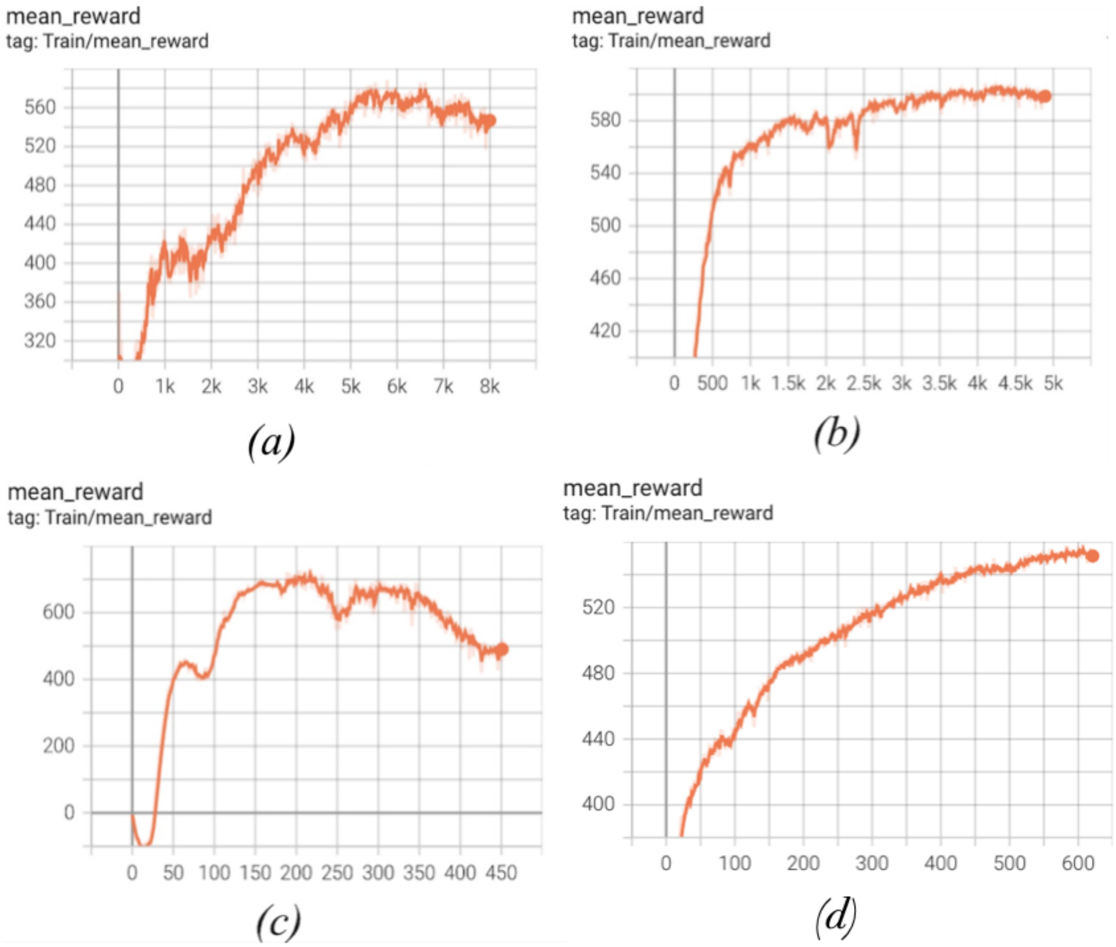


Figure 2.3: Reward vs. episode convergence curve for tasks. *(a)* Stairs Climbing. *(b)* Obstacle avoidance. *(c)* Squeezing under obstacles. *(d)* Squeezing between obstacles.

h_{ij} is the $(i,j)^{th}$ element of the $M \times N$ matrix height map matrix H, and $\overrightarrow{w_1}$ is an N dimensional weight vector of dimension N of a triangular shape given by:

$$\overrightarrow{w_1} = \left[1, 1+\frac{2}{N}, 1+\frac{4}{N}, \ldots, 2, \ldots, 1+\frac{4}{N}, 1+\frac{2}{N}, 1\right]$$

corresponding to the perpendicular distance of the points in the binarized height map to the axis parallel to the direction of the movement of the robot passing through the middle of the robot. A visualization of w_1 can be seen in Figure 2.5. w_2 is an M dimensional weight vector of the shape of a ramp ranging from 1 to 2, providing more weight to the obstacle points closer to the center front of the robot in the walking direction given by:

$$\overrightarrow{w_2} = \left[1, 1+\frac{1}{M}, 1+\frac{2}{M}, \ldots, 2\right]$$

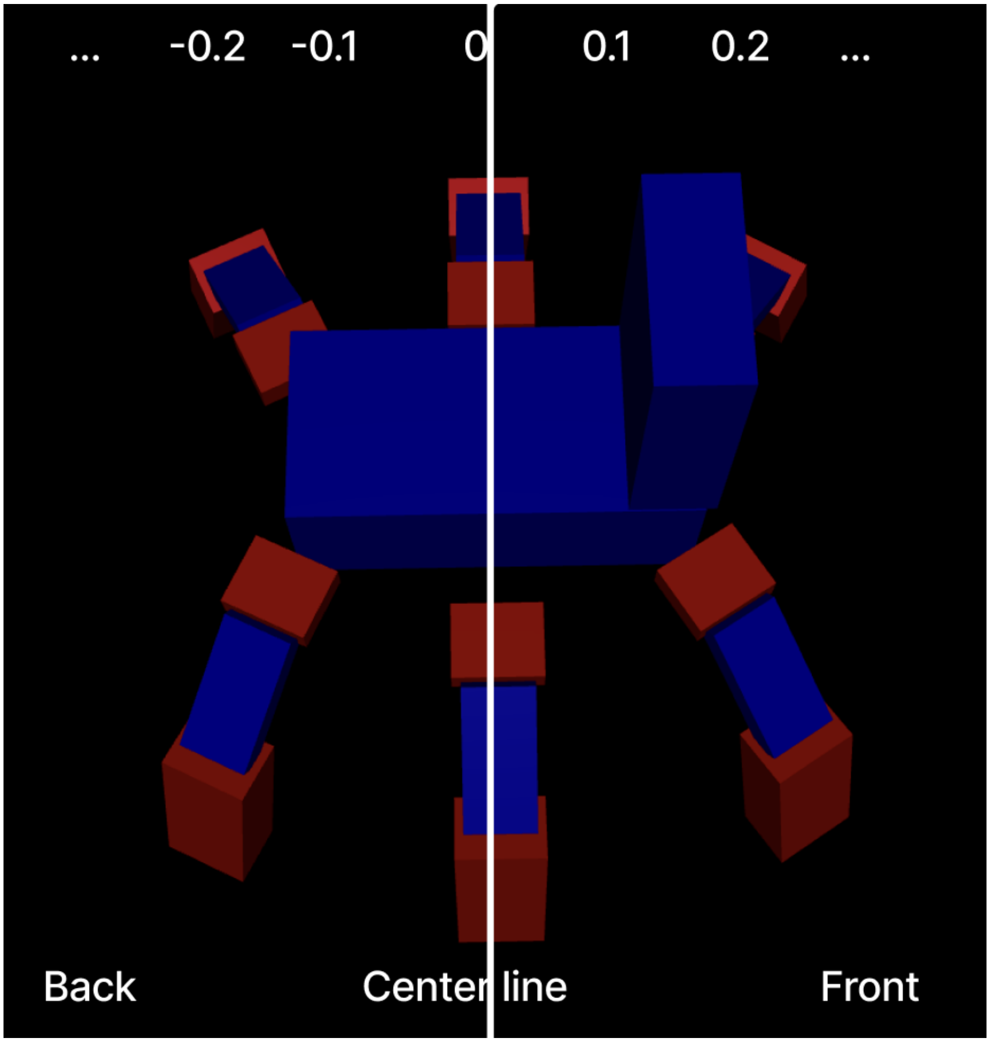


Figure 2.4: Visualization of heightmap center and heightmap location value as shown in Table 2.2.

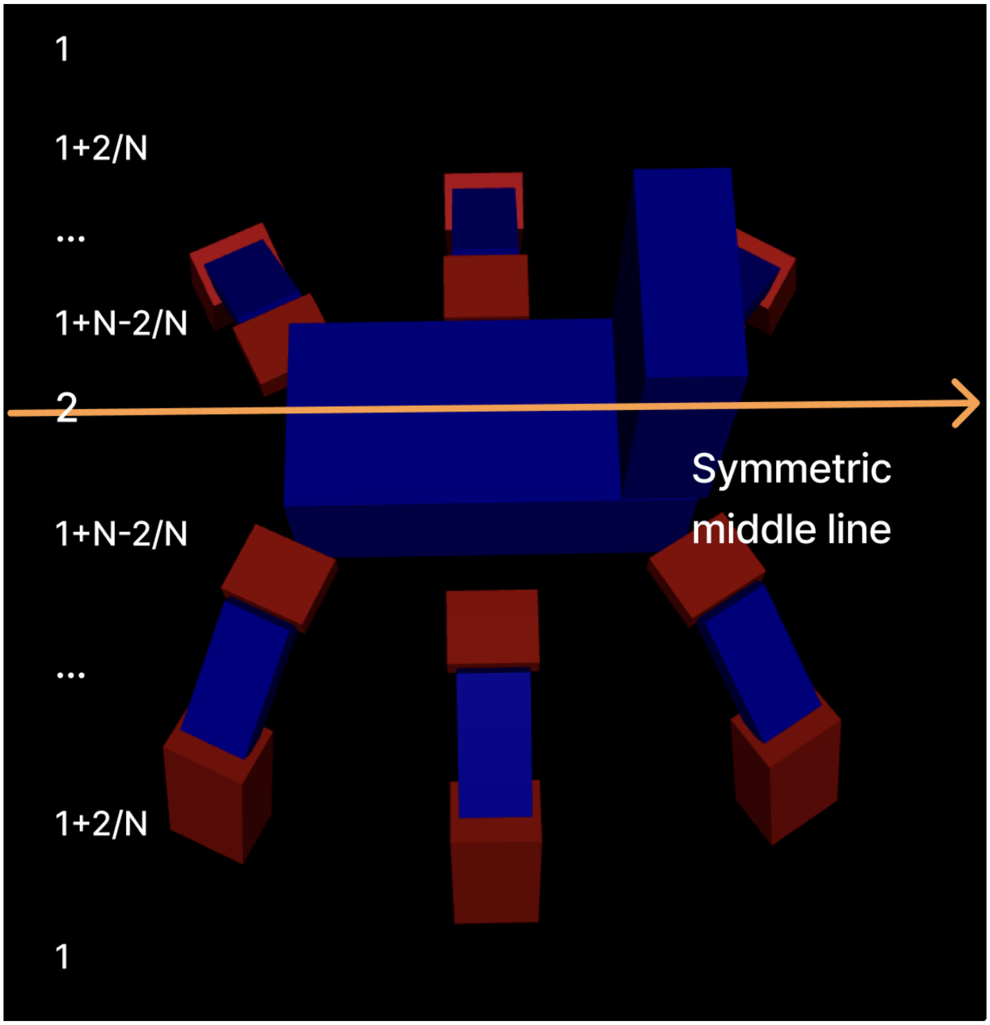


Figure 2.5: Visualization of symmetric middle line and the vector structure.

For the terrain in simulation, we included a variety of obstacle shapes in the map. We also note that if the obstacles are densely placed on the map at the beginning stages of training, the policy is likely to converge to a local maximum reward and terminate early. This is due to the sudden large penalty generated by the distance to obstacle reward term. To address this, we incorporate curriculum training, where the robot first learns to walk without any obstacles, then to avoid sparsely placed obstacles, followed by navigating a more densely populated obstacle terrain. The density range at each level of difficulty is given by $(2 \times Level/Total\_Levels) \times Density_{final}$, where $\text{Density}_{\text{final}}$ is defined as the density of obstacle spacing at the final difficulty level.

The obstacle avoidance functionality provided by the manufacturer of our robot uses ultrasound to sense obstacles and change direction accordingly, but it does not revert to

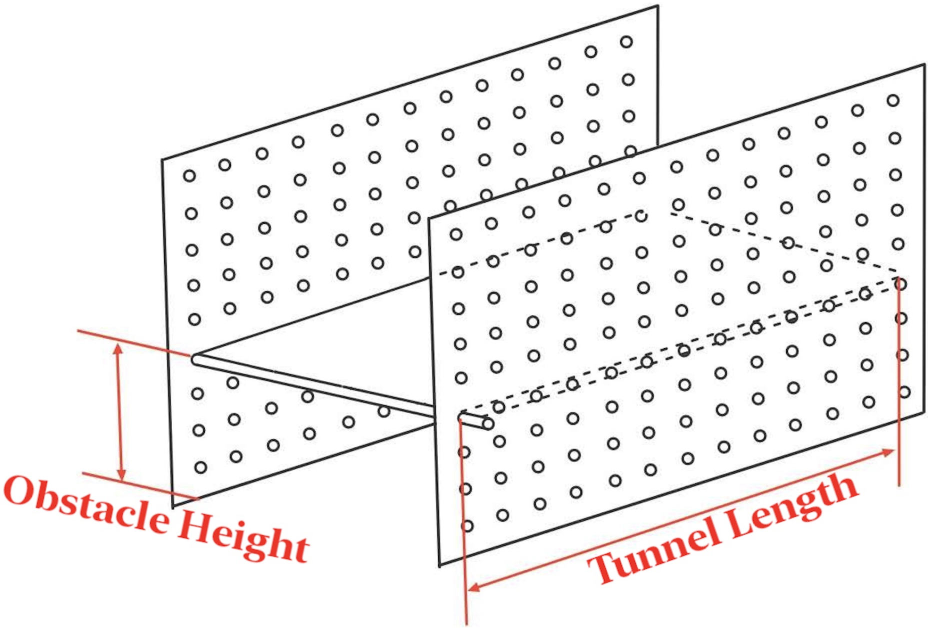


Figure 2.6: Physical design of a squeezing environment.

its original orientation afterward, as seen in this video **link**. In contrast, we ideally want our obstacle avoidance functionality to have the robot return to its original orientation after avoiding a given obstacle. To achieve this, we use a combination of a 'global y deviation' term and a 'local y velocity' term, where the y-axis is perpendicular to the direction the robot was moving along before reacting to the obstacle. The 'global y deviation' term, shown in the third-to-last row of Table 2.1, indicates the distance between the robot and the y-axis. The local y velocity term, shown in row 2, refers to the squared velocity along the y direction. The latter term prevents the robot from returning to its original direction of motion too soon, which can occur if the obstacle is no longer within the robot's field of view after the initial turn.

Squeezing Under Obstacles

Squeezing under objects is an important skill for traversing environments with low-clearance obstacles such as beds, couches, and tables. We mainly focus on two aspects in the design of the squeezing terrain: obstacle height and tunnel length, as defined in Figure 2.6. Our goal is for the robot to learn this skill and generalize to obstacles of varied shapes and tunnels of varied lengths.

Since Isaac Gym does not support floating terrains and requires all elements to be grounded, we modified the code to construct terrains on vertices "in the air." In the simulation environment, we created a floating obstacle above the basic terrain shown in Figure 2.2(c) to mimic the effect of a physical table, bed, or couch. The height map used in phase 1 of training in simulation takes on the ground-level height when there is no obstacle above, and the obstacle's height otherwise. We then define a single reward term, shown in the second-to-last row of Table 2.1, to maximize the distance between the robot and the obstacle when there is an obstacle above, and maximize the distance between the robot's base and the ground otherwise. Concretely, the reward is defined as follows:

$$R(H) = -\sum_{i=1}^{N} \left(\sum_{j=1}^{M} h''_{ij} \cdot \overrightarrow{w_{3j}} \right) \cdot \overrightarrow{w_{4i}}$$

where h''_{ij} is a function of the height map h_{ij} and the distance of the base robot to the ground, b, as follows:

$$h''_{ij} = \begin{cases} 1 & \text{if } (h_{ij} - b) > 0 \\ -2 \cdot |h_{ij} - b| & \text{otherwise} \end{cases}$$

$\overrightarrow{w_3}$ is a N dimensional vector of all ones, and $\overrightarrow{w_4}$ is an M dimensional ramp vector ranging from 1 to 2 to assign a larger weight to the obstacle points closer to the center front of the robot in the walking direction given by:

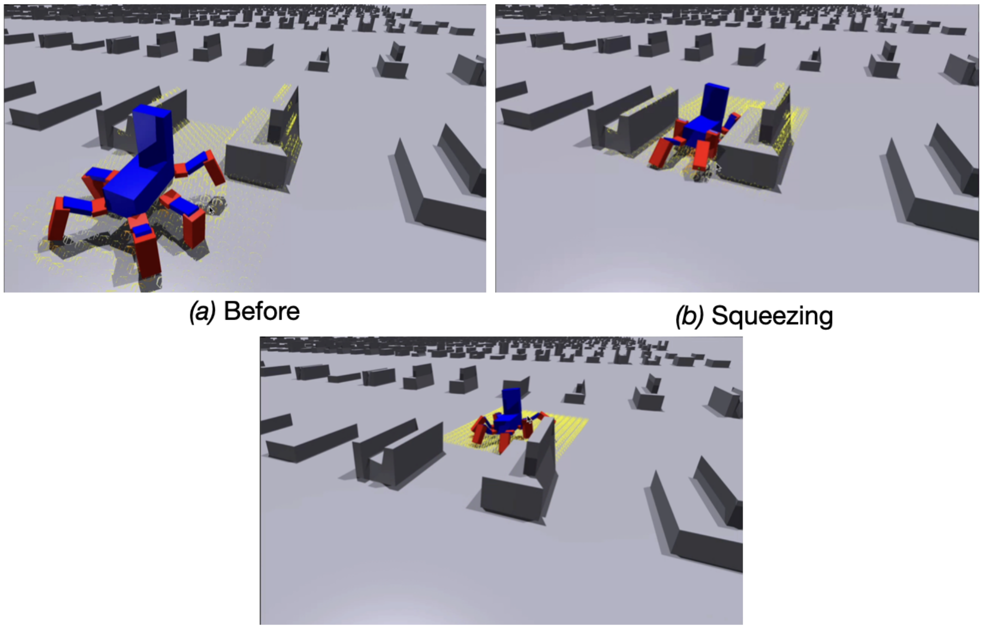
$$\overrightarrow{w_4} = \left[1, 1 + \frac{1}{N}, 1 + \frac{2}{N}, \ldots, 2 \right]$$

We use curriculum learning so that the robot first learns to walk and then to squeeze under obstacles. The level of difficulty increases as we decrease the object's distance to the ground from 37 to 35, 33, and finally to 31 cm. The obstacle itself can have variable height in the vertical direction, as well as variable length or width. The longer the obstacle, the longer the robot has to remain in a squeezed posture to avoid hitting it while passing underneath. We also randomize the origin so that, in simulation, the robot encounters the obstacle after walking different distances. Finally, the optical axis of the depth camera is parallel to the ground so that it can clearly see the beginning and end of the "tunnel" created by the obstacle.

Squeezing Between Obstacles

Squeezing between obstacles is a crucial skill for traversing cluttered environments. It enables the hexapod robot to navigate through narrow hallways and narrow spaces between furnitures, and further enhances its agility.

The width of the robot in the rest position is 52 cm, as shown in Figure 1.2, and it becomes wider while in motion. In simulation, we construct terrains consisting of obstacles with various shapes and gaps to mimic a cluttered room. To improve the generalization of the policy, we vary the shapes of the obstacles and randomize the gaps between them to values in [45, 50, 55, 60] cm. During policy execution, the robot squeezes its body to reduce

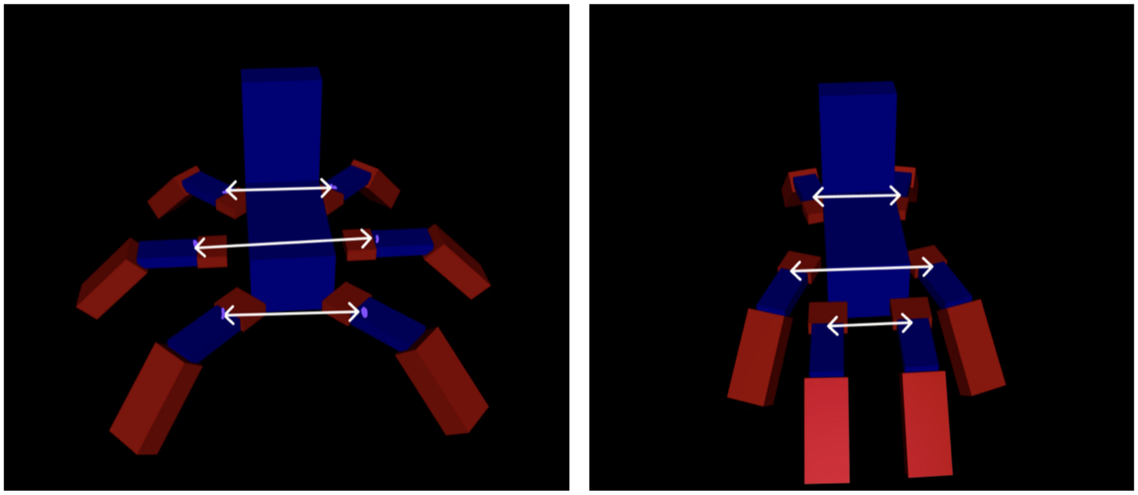


(a) Before

(b) Squeezing

(c) After

Figure 2.7: Squeezing between process in simulation.



(a) Rest

(b) Squeeze

Figure 2.8: Visualization of the distance between symmetric legs.

	Up (Steps completed/total)			Down (Steps completed/total)		
Method	Cory	Soda	Sud.	Cory	Soda	Sud.
Perceptive (Ours)	6.0/7.0	7.0/7.0	7.6/8.0	6.6/7.0	7.0/7.0	7.5/8.0

Table 2.5: Stair climbing performance across three test locations (Cory Stairs, Soda Stairs, and Sudardja).

its width when surrounded by obstacles and expands after successfully exiting the tunnel. A visualization of this process can be seen in Figure 2.7.

A simple reward that encourages distance between the joints and obstacles can result in an undesired gait and may be difficult to train. To address this issue, we introduce a new reward to encourage the squeezing behavior when the robot is surrounded by obstacles. We define a single reward term, shown in the last row of Table 2.1, to minimize the distance between the symmetric legs of its body as shown in Figure 2.8. This is designed to deal with situations where there are obstacles on both the left and right sides of the robot. Concretely, the reward is defined as follows:

$$R(H) = I * \frac{1}{3}\sum_{i=1}^{3}(1 - (d_i - t) * c)$$

where H stands for the heightmap, d_i denote the distance between symmetric outer joint of the left and right legs in each row of the legs, t refers to a target width, and c is a constant scalar value. In our design, we select $t = 0.3$ and $c = 4$. I is an indicator defined as follows:

$$I = \begin{cases} 1 & \text{if } H_{il} > 2 \text{ and } H_{ir} > 2 \\ 0 & \text{otherwise} \end{cases}$$

where H_{il} and H_{ir} represent the number of height points greater than 0 on the left and right sides of the center of the robot of row i of the heightmap, respectively. In this design, if any single row satisfies this condition, the indicator I is set to 1; otherwise, it is set to 0.

2.2 Physical Experiments

We run the policies trained in simulation on a Raspberry Pi processor mounted on the physical robot. We conducted experiments on all four tasks to verify and characterize the performance of our policies in real-world environments. For the staircase task, we used three staircases on the campus of U.C. Berkeley. For the other three tasks, we constructed several terrains to evaluate the robot's performance by measuring its success rate in completing the tasks.

	Single Box			Box and Bag			Person Standing			Person Moving		
Method	success	scrape	collision	success	scrape	collision	success	scrape	collision	success	scrape	collision
Perceptive (Ours)	7/10	2/10	1/10	9/10	1/10	0/10	7/10	2/10	1/10	8/10	1/10	1/10

Table 2.6: Object avoidance performance.

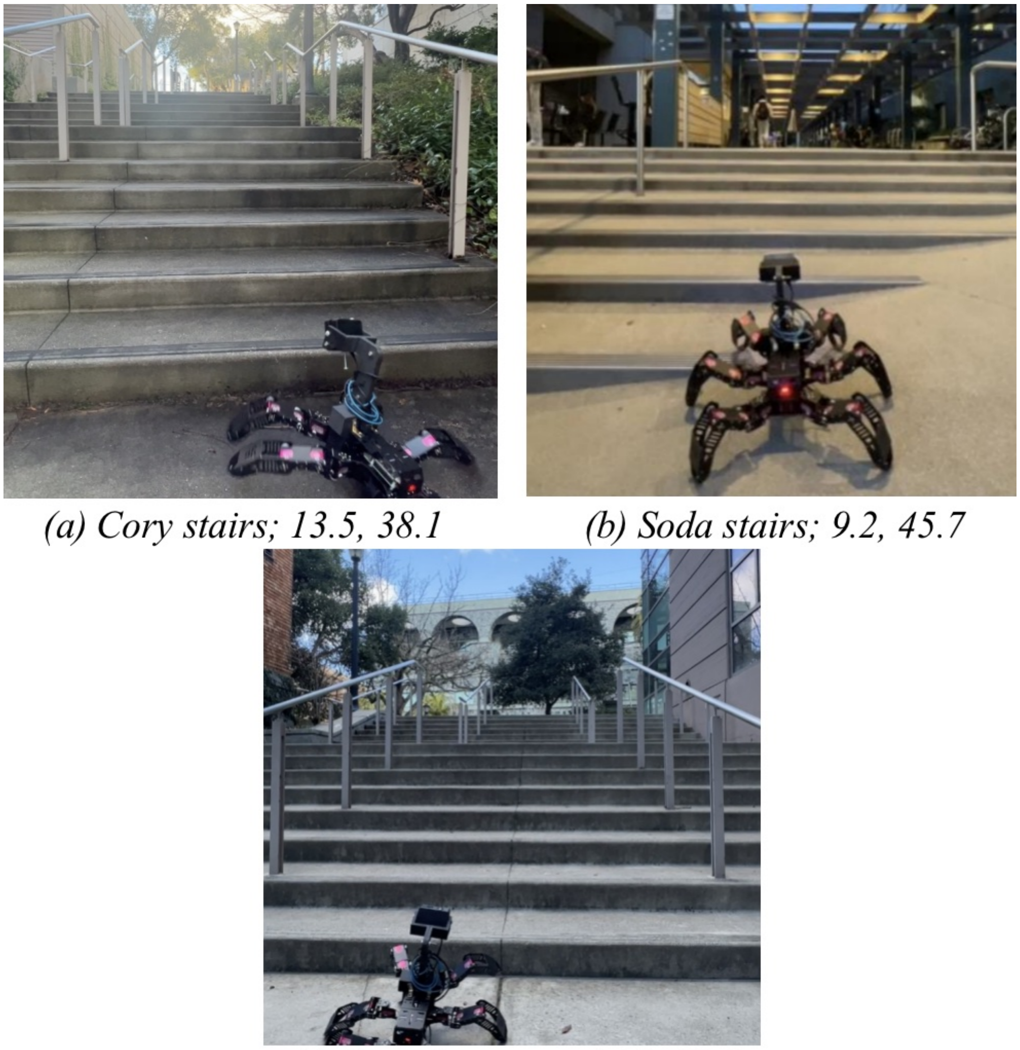
Stair Climbing

We chose three staircases on the U.C. Berkeley campus, as shown in Figure 2.9, to evaluate the robot's performance in both climbing and descending tasks. In Figure 2.9, Cory Hall and Soda Hall each have 7 steps, while Sutardjai Hall has 8 steps. At the start of each experiment, the robot is placed 20 cm in front of the stairs and is reset to its default standing position, as shown in Figure 1.2. We intervene only if the robot falls over or becomes stuck on a stair.

We evaluate performance by counting the number of steps the robot could complete in each trial. We then average that number across 10 trials, as shown in Table 2.5. As observed, Soda Hall has the highest success rate because it has the lowest rise and the highest tread. Few of the climbing failure cases resulted from the robot falling over. Others occurred when the robot became stuck on the last stair for too long, believing it was walking on flat ground because the camera image did not show any stairs. For stair descent, failure cases resulted from the robot losing control and falling. We present a video **link** showing the robot climbing two sets of stairs separated by a platform, with 8 and 7 steps in the first and second sets, respectively. As shown in the video, the robot is able to traverse in a relatively straight line down the center without significant lateral shift to the right or left. The video also demonstrates the robot descending stairs using our policy, compared to a baseline walking policy trained on flat terrain. As expected, applying the "walk" policy to the staircase results in the robot crashing down the stairs.

Avoiding Obstacles

We construct a variety of obstacle scenarios to evaluate the robot's obstacle avoidance performance. We test the robot's generalization ability to detect arbitrarily shaped obstacles and to avoid them robustly. We design a total of four environments. The first consists of a single box-shaped obstacle placed in front of the robot to test its ability to avoid from the left or the right. The second consists of two obstacles: a box-shaped obstacle placed in front and a deformable plastic bag placed behind it. The third consists of a person standing or sitting in front of the robot. The fourth consists of a person walking toward the robot. We test each environment 10 times, with results shown in Table 2.6. We group the outcomes into three categories: success, scrape, and collision. Success means the robot navigates around the obstacle without any collision; scrape means the robot is able to go around the obstacle



(a) Cory stairs; 13.5, 38.1

(b) Soda stairs; 9.2, 45.7

(c) Sutardja stairs; 14.0, 38.1

Figure 2.9: Experimental scene for stairs climbing. The first number is the riser height and the second number is the tread depth both in centimeters.

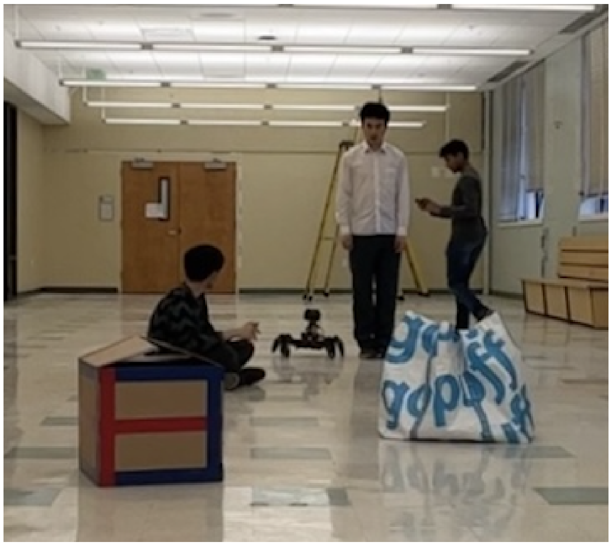


Figure 2.10: Obstacle Avoidance Environment.

but some part of its leg scrapes the object or person; collision means the robot runs into the object.

In failure cases, the robot often detects the object but reacts too slowly, resulting in the back leg scraping the obstacle after the front part has successfully avoided it. In successful cases, the robot exhibits avoidance behavior to either the left or right, depending on its distance to each side of the obstacle. This indicates that our policy did not simply memorize to avoid obstacles by always veering in one direction. After passing the obstacle, the robot rotates back to its original orientation and continues moving forward. Finally, we conducted a comprehensive test of the robot's ability to avoid all six of our tested obstacles in a row, as shown in Figure 2.10 and visualized in this video **link**.

Squeezing Under Obstacles

The squeezing setup shown in Figure 2.11 aims to mimic obstacles such as a couch, a bed, or a table. At the beginning of the experiment, the robot is positioned 30 cm in front of the obstacle. It is tasked with squeezing under the object and walking in the squeezed mode until it exits the "tunnel" created by the overhead obstacle, at which point it is expected to raise its body to the default height and continue walking. The experiment is considered a success if the robot is able to pass through the tunnel without any collisions[1].

Our experimental setup for squeezing is shown in Figure 2.11, with dimensions superimposed on the images. In Figures 2.11(a), 2.11(c), and 2.11(d), there is a pair of metal rods

[1] Collisions are visually detected and defined as events where contact between the robot and obstacles results in a trajectory change.

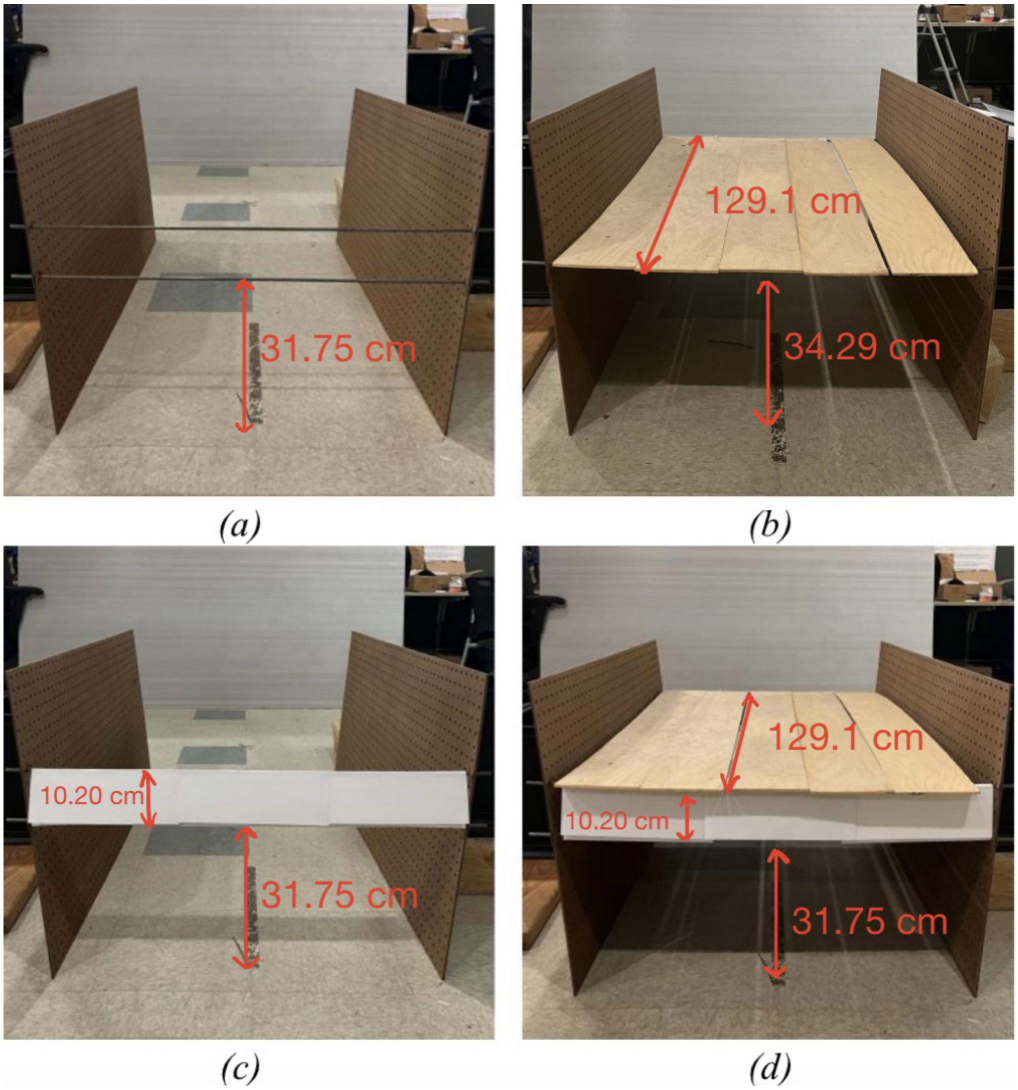


Figure 2.11: Experimental scene for squeezing. *(a)* Metal rod as obstacle without tunnel. *(b)* Lengthy tunnel. *(c)* Paper block as obstacle without tunnel. *(d)* Paper block as obstacle with tunnel.

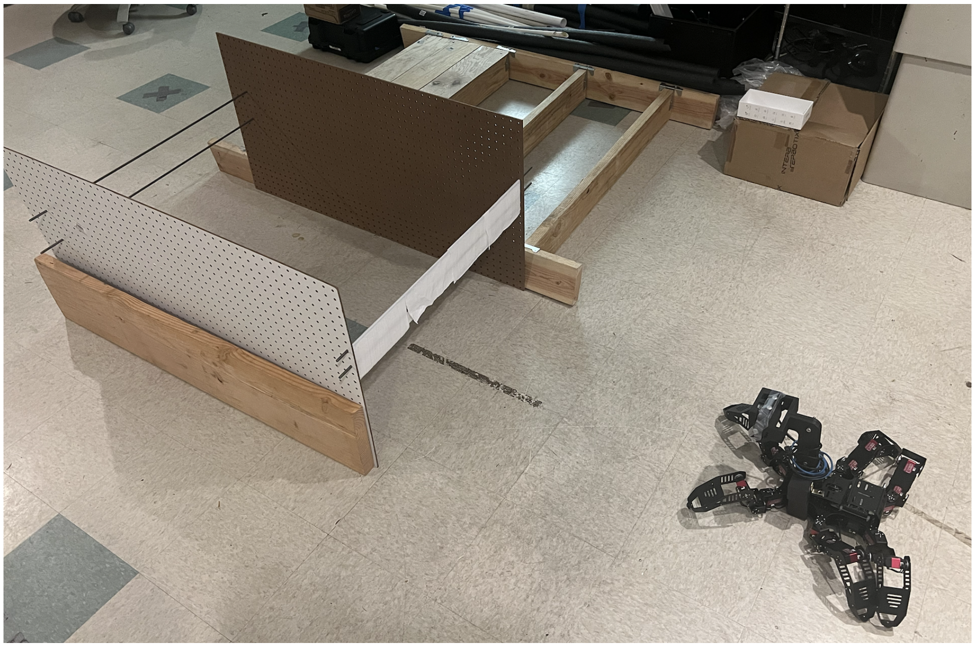


Figure 2.12: Squeezing under two consecutive obstacles.

placed 4 inches (10.20 cm) apart. The ends of the rods are inserted into two boards with holes on a one-inch grid. In Figures 2.11(c) and 2.11(d), paper pieces connect the two rods, creating a different depth image than in Figure 2.11(a) during inference and thus testing the generalizability of our policy. Finally, a wooden board in Figures 2.11(b) and 2.11(d) creates a tunnel of length 129.1 cm to ensure the robot remains in the squeezed position for an extended period.

As a reference, the robot is 28.75 cm high when lying flat and 37 cm high when standing in the rest standing position shown in Figure 1.2. While we acknowledge that the height of the robot in the squeezing position is still relatively large, our main goal is to demonstrate the concept of teaching the robot to squeeze when needed. For each of the four settings shown in Figure 2.11, we performed 10 to 20 trials and present the success rate in Table 2.7.

Terrain	**Success Rate**	**% Success**
Metal rod w/o tunnel (a)	17/20	85%
Lengthy tunnel (b)	10/10	100%
Block w/o tunnel (c)	18/20	90%
Block with tunnel (d)	9/10	90%

Table 2.7: Squeezing performance with corresponding construction in Figure 2.11.

As seen, the success rate is high[2]. Even though, we would intuitively expect the success rate for metal rod and block without tunnel experiments in rows (a) and (c) of Table 2.7 to be higher than that of block with tunnel in row (d), since the total number of trails for (a) and (c) were 20 and for (d) was 10, in practice, the battery in the robot would run out after 20 trials and result in more failures. While the robot can momentarily squeeze as low as 31.75 cm to pass under a thin metal rod, it cannot sustain that height over an extended distance—such as 129 cm—without hitting the tunnel roof. However, it can maintain the squeezed posture for a short while through the 34.29 cm high tunnel shown in Figure 2.11(b). The support boards on the sides have grid hole spacing of one inch; thus, we can only adjust the rod height in one-inch increments, resulting in a "jump" from 31.75 cm to 34.29 cm. We speculate that the robot could successfully pass through a tunnel with a height of 33 cm without scraping the top.

In one experiment, shown in Figure 2.12, where there are two obstacles each 31.75 cm high and no tunnel between them, we observed that the robot squeezes under the first obstacle, briefly rises in the middle, then squeezes again under the second obstacle upon sensing it, and finally stands up to continue walking. We present a video demonstration of this experiment in this **link**. As seen in the video, the robot's "belly" comes close to the ground as it squeezes under the obstacles and then rises afterward. The plot in Figure 2.13 shows the change in the height of the robot's base relative to the ground, as captured in simulation.

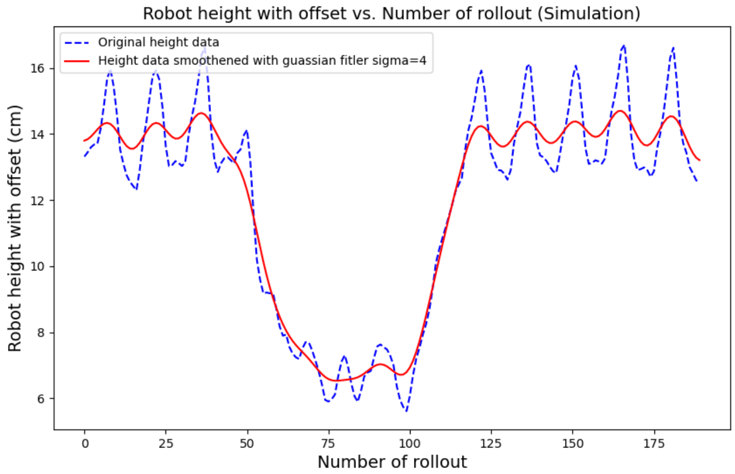


Figure 2.13: Height of robot body vs. rollout step in Isaac Gym.

[2] We have empirically found that the failure rate for all three tasks increases when the battery is low and the supplied voltage is around 10 volts rather than the nominal 12 volts.

Squeezing Between Obstacles

Squeezing between occurs when two objects are placed close to each other. In such scenarios, the robot is required to tighten its body to reduce its width and pass through the obstacles. To complete the process, the robot should stop squeezing and expand its legs into a walking gait after exiting the "tunnel" formed by the obstacles.

The experimental setup for squeezing between is shown in Figure 2.14. We tested the policy with two different groups of obstacles and various gap widths. The robot has a width of 52 cm in the resting position and becomes even wider while moving or executing other skills. Therefore, terrain construction focused primarily on the gap width between obstacles. As shown in Figure 2.14(a), we placed a tall fan on the left and a black box on the right, with two different gap widths of 35 cm and 40 cm, where the distance is measured from the bottom of the fan to the bottom of the box. In the second setup, shown in Figure 2.14(b), we placed two black boxes on either side and tested the robot's ability to pass through obstacles where the narrowest gap between them is 47 cm.

At the beginning of each experiment, the robot is placed 30 cm in front of the obstacles in a resting position. It is tasked with walking toward the tunnel, initiating squeezing behavior when surrounded by obstacles, and expanding its legs to resume walking after exiting, as shown in Figure 2.15. The experiment is considered a success if the robot exhibits squeezing behavior and exits the tunnel without getting stuck.

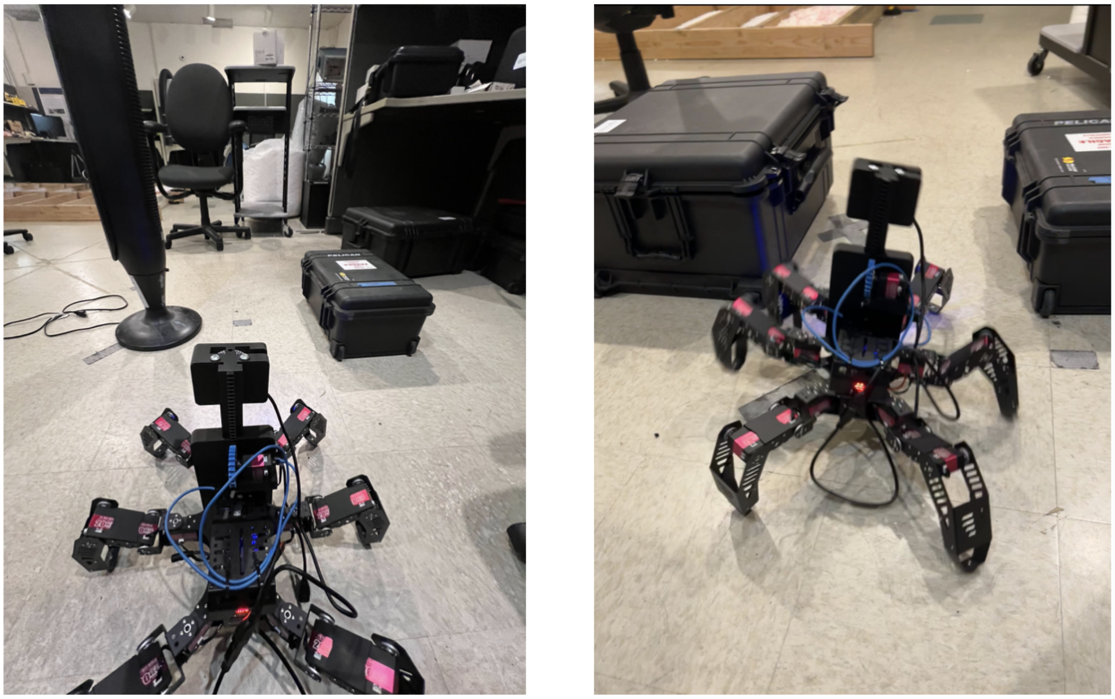
Terrain	**Success Rate**	**% Success**
two boxes (47 cm)	4/5	80%
box & fan (40 cm)	5/5	100%
box & fan (35 cm)	4/5	80%

Table 2.8: Squeezing between obstacles results

The results are shown in Table 2.9. We observed that, in failure cases, the robot often runs into the obstacle before initiating the squeezing behavior. In successful cases, the robot squeezes while surrounded by the obstacles, then expands and resumes walking after exiting. As a visual illustration, we present a video of the robot navigating each terrain at the following **link**.

2.3 Squeeze with adjustable camera

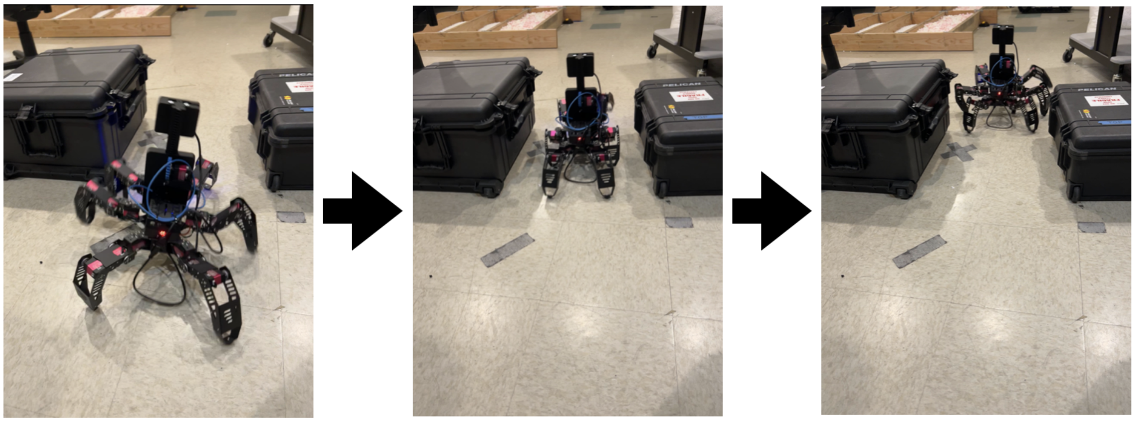
In the previous section, we demonstrated that the squeezing policy allows the robot to squeeze under obstacles of varying heights and depths. In that experiment, we showed that the robot is capable of squeezing under an obstacle of 31.75 cm with a high success rate. One limitation, however, is that the robot is still too tall to traverse realistic obstacles such as a low table or a chair. To enhance performance, we designed an adjustable camera mount to



(a) A box and a fan

(b) Two boxes

Figure 2.14: Squeezing between physical experiments setup.



(a) Before

(b) Squeezing

(c) After

Figure 2.15: Squeezing between process in physical experiments.

replace the rigid camera used in the previous design. We used the same type of servo as those controlling the legs to enable up-and-down movement of the camera. The servo operates within a range of 0 to 240 degrees, corresponding to a height adjustment of 0 to 12 cm. As shown in Figure 1.3, the new camera mount reduces the overall height of the robot from 37 cm to 25 cm in its resting position. This additional degree of freedom enables the hexapod robot to handle more challenging and realistic terrains, such as passing under a table or a chair. Different methods were considered for this task. A reinforcement learning (RL)–based method with the adjustable camera as an extra degree of freedom was trained in Isaac Gym. The "distance to above" reward used in the previous iteration remained unchanged, as it encourages maintaining distance between the obstacle and the camera, causing the camera to lower whenever an overhead obstacle is detected. However, we observed that this method suffers from a significant sim-to-real transfer gap.

We analyze the joint angles during the execution of the squeezing policy. A joint angle vs. rollout step graph is shown in Figure 2.16, where we plot the joint angles of all 18 joints during a squeezing task. We observe that a few joints exhibit clear patterns that can be used as thresholds to control camera movement. We select joint 3 as the reference for thresholding the camera. Specifically, when the joint angle reaches 90 degrees, the camera is set to the low position; when the camera is already low and the joint angle decreases to 70 degrees, the camera is set to the high position. Using this thresholding logic, performance remains stable across different obstacles.

We experiment with two different obstacles found in the office—a 20 cm tall table and a 25 cm tall sofa—as shown in Figures 2.17(a) and 2.17(b). During each experiment, the robot is placed in front of the obstacle and tasked with passing through while demonstrating the desired body and camera movement. We also placed the robot at a tilted angle facing the desk to test the robustness of the policy.

Terrain	**Success Rate**	**% Success**
Table (straight)	10/10	100%
Table (tilted)	10/10	100%
Sofa	9/10	90%

Table 2.9: Squeezing under obstacles with adjustable camera performance with corresponding construction

As shown in Table 2.9, the success rate is high. The only failure case occurred in the sofa experiment, where the robot's leg became stuck on the bottom metal rod of the sofa. In all other experiments, the robot demonstrated the desired behavior: lowering the camera and body during squeezing, and maintaining a high camera and body position otherwise. We performed a comprehensive test involving both the sofa and the table. The robot's movement during this process is visualized in Figure 2.18. The complete video can be viewed at the following **link**.

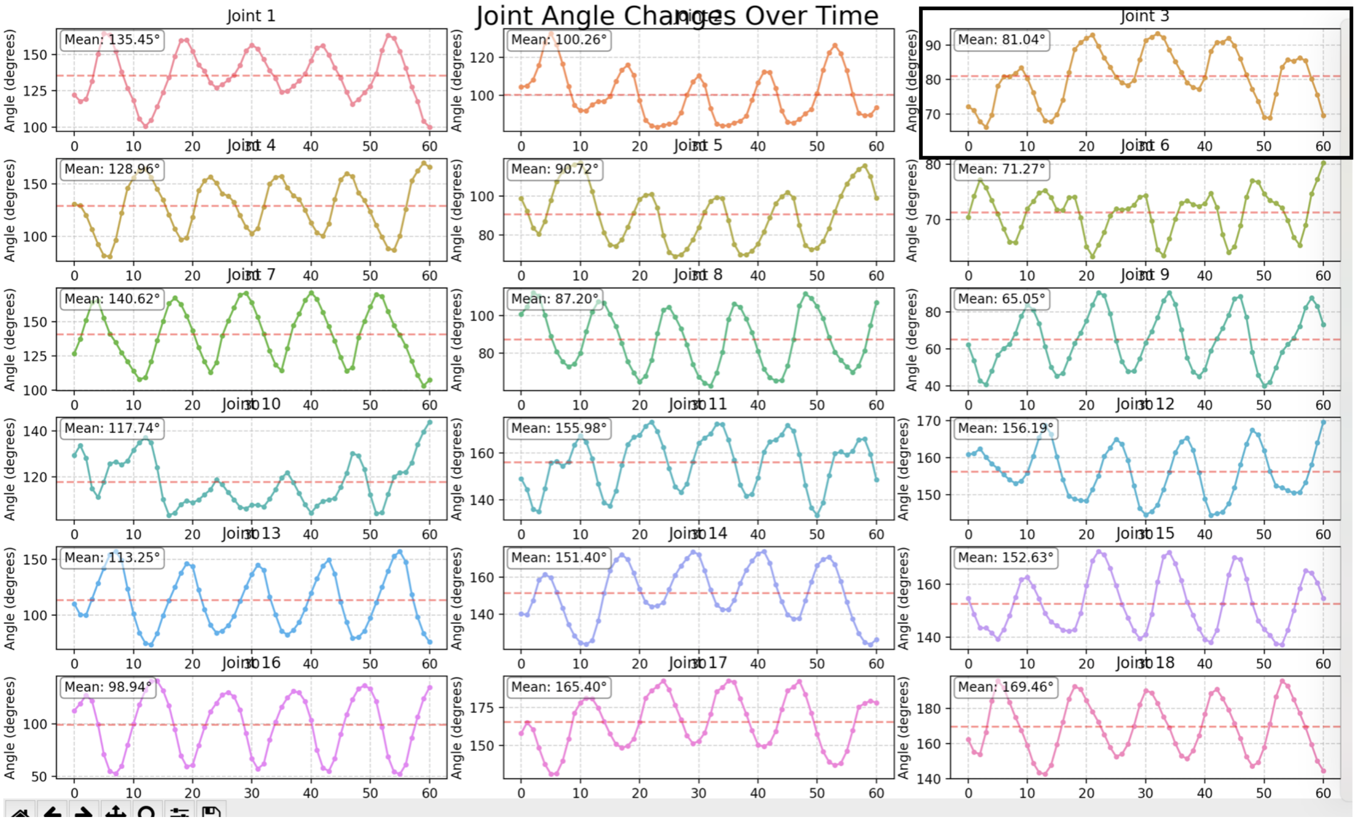
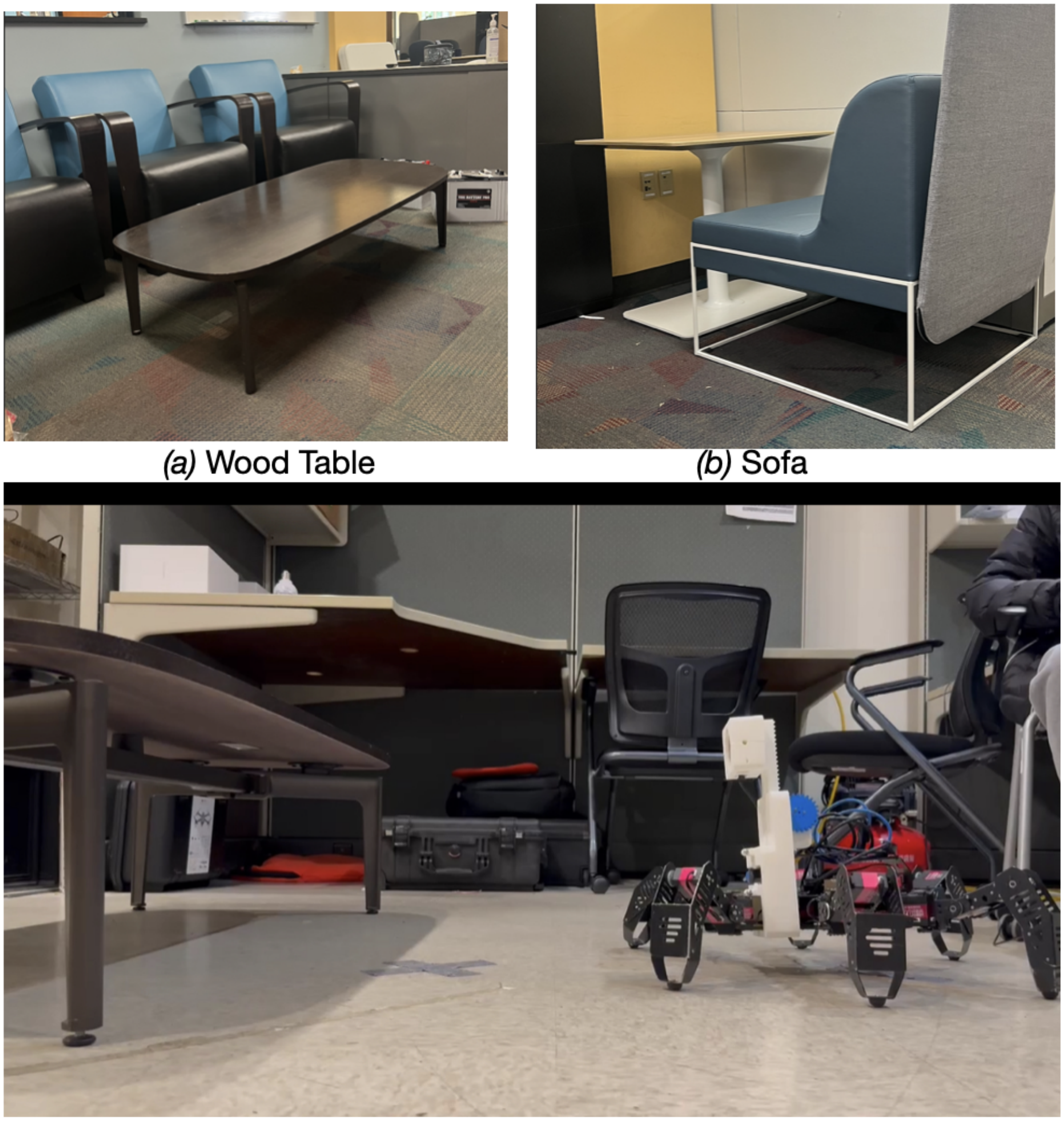


Figure 2.16: Joint angle vs. rollout step during squeezing.

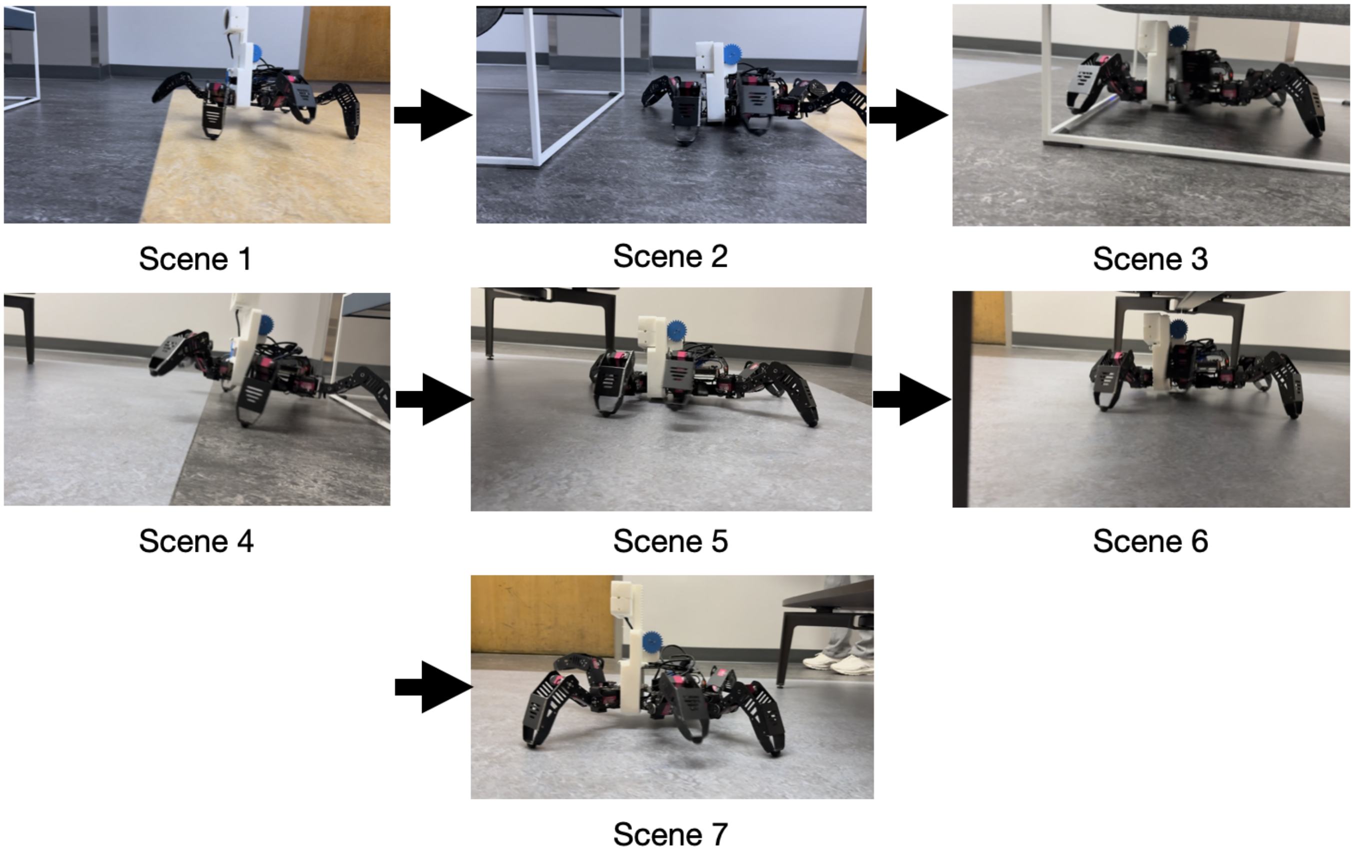


(a) Wood Table

(b) Sofa

(c) Start of the experiments

Figure 2.17: Squeezing under obstacles with adjustable camera setup in physical experiments.



Scene 1

Scene 2

Scene 3

Scene 4

Scene 5

Scene 6

Scene 7

Figure 2.18: Squeezing under obstacles with adjustable camera process in physical experiments

Chapter 3

Conclusions and Future work

In this work, we trained a hexapod robot with multiple locomotion skills—including climbing up/down stairs, avoiding obstacles, squeezing under objects, and squeezing between objects—using a depth camera and a visual-inertial odometry sensor. Using a 2 stage training strategy, we trained the robot entirely in simulation and demonstrated successful transfer to the physical robot. We further enhanced the robot's agility by mounting an adjustable camera, allowing it to become more compact during tasks that require squeezing under obstacles. This additional design enables the robot to pass under common furniture such as chairs and sofas. The ability to automatically adjust the camera height allows the robot to benefit from both compact and upright configurations. We demonstrated the robot's ability to perform these tasks effectively through rigorous physical experiments.

For future work, we consider it beneficial to enhance the system by combining locomotion skills with tools such as visual language models. As shown in this thesis, the skills acquired through training are well-suited for specific tasks. It may be advantageous to introduce a higher-level agent capable of combining and executing these skills in various scenarios. For example, given an instruction like "approach the blue can in the room," the agent should be able to decompose the task into subproblems. With the aid of visual reasoning, the agent can then select the appropriate locomotion skill for each subproblem. This would enable the hexapod robot to accomplish more complex tasks in real-world settings.

Adding extra mounts could also be useful in many scenarios. For instance, executing a command such as "pick up the ball under the bed" requires more than just locomotion—it necessitates manipulation capabilities. In such cases, a gripper or a hand would be instrumental in completing the task. With a more sophisticated system, the inexpensive hexapod robot has the potential to be both affordable and practical for household use.

We present preliminary results on skill combination using a vision-language model (VLM). We designed an agentic flow to solve planning tasks, as shown in Figure 3.1. The system receives a task in the form of a natural language instruction, and the agents within the system collaborate by performing their respective roles and communicating with each other to complete the task.

During execution, the vision agent processes depth information and RGB images. It

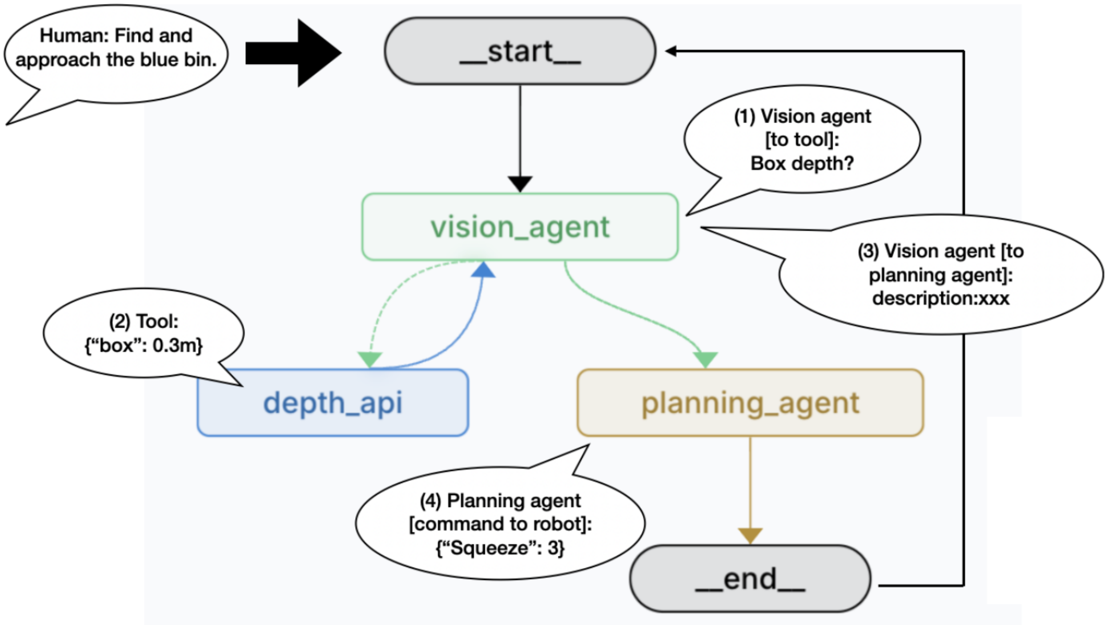

Human: Find and approach the blue bin.
__start__
(1) Vision agent [to tool]: Box depth?
vision_agent
(3) Vision agent [to planning agent]: description:xxx
(2) Tool: {"box": 0.3m}
depth_api
planning_agent
(4) Planning agent [command to robot]: {"Squeeze": 3}
__end__

Figure 3.1: Overview of the VLM flow with example.

sends requests to tools such as the depth API to obtain accurate environmental information. The vision agent then sends a scene description to the planning agent, which generates the most appropriate action for the current context. These actions include rotating by a certain angle or executing one of the predefined locomotion skills. The robot receives the action command and performs the action. After execution, the feedback agent stores the RGB image and action in the system state and initiates the next iteration. We observed that the current system only works reliably when the goal is visible at the beginning of the experiment. One challenge is that the robot lacks a reliable tracking system, making it difficult to estimate information such as distance traveled. Another limitation is the robot's lack of precise control—it cannot reliably rotate to a specific angle or maintain a constant moving speed. Addressing these challenges would be beneficial for future improvements.

Bibliography

[1] Ananye Agarwal et al. *Legged Locomotion in Challenging Terrains using Egocentric Vision*. 2022. arXiv: 2211.07638 [cs.RO]. URL: https://arxiv.org/abs/2211.07638.

[2] Ananye Agarwal et al. *Legged Locomotion in Challenging Terrains using Egocentric Vision*. 2022. arXiv: 2211.07638 [cs.RO].

[3] Miroslav Bogdanovic, Majid Khadiv, and Ludovic Righetti. "Model-free reinforcement learning for robust locomotion using demonstrations from trajectory optimization". en. In: *Frontiers in Robotics and AI* 9 (Aug. 2022), p. 854212. ISSN: 2296-9144. DOI: 10.3389/frobt.2022.854212. URL: https://www.frontiersin.org/articles/10.3389/frobt.2022.854212/full (visited on 02/14/2024).

[4] Johann Borenstein and Yoram Koren. "The vector field histogram—fast obstacle avoidance for mobile robots". In: *IEEE Transactions on Robotics and Automation* 7.3 (1991), pp. 278–288.

[5] Xuxin Cheng et al. *Extreme Parkour with Legged Robots*. 2023. arXiv: 2309.14341 [cs.RO]. URL: https://arxiv.org/abs/2309.14341.

[6] Jaewan Choi, Geonhee Lee, and Chibum Lee. "Reinforcement learning-based dynamic obstacle avoidance and integration of path planning". en. In: *Intelligent Service Robotics* 14.5 (Nov. 2021), pp. 663–677. ISSN: 1861-2776, 1861-2784. DOI: 10.1007/s11370-021-00387-2. URL: https://link.springer.com/10.1007/s11370-021-00387-2 (visited on 02/14/2024).

[7] Howie Choset et al. *Principles of robot motion: theory, algorithms, and implementation*. MIT press, 2005.

[8] D. Fox, W. Burgard, and S. Thrun. "The dynamic window approach to collision avoidance". In: *IEEE Robotics Automation Magazine* 4.1 (1997), pp. 23–33. DOI: 10.1109/100.580977.

[9] Ujjawal Garg. "Virtual Robot Climbing using Reinforcement Learning". en. Master of Science. San Jose, CA, USA: San Jose State University, Dec. 2018. DOI: 10.31979/etd.u9xe-s6yw. URL: https://scholarworks.sjsu.edu/etd_projects/658 (visited on 02/14/2024).

[10] Kevin G. Gim and Joohyung Kim. “Snapbot V2: a Reconfigurable Legged Robot with a Camera for Self Configuration Recognition”. In: *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. 2020, pp. 4026–4031. DOI: 10.1109/IROS45743.2020.9341279.

[11] Dominic Guri et al. *Hefty: A Modular Reconfigurable Robot for Advancing Robot Manipulation in Agriculture*. 2024. arXiv: 2402.18710 [cs.RO]. URL: https://arxiv.org/abs/2402.18710.

[12] Fabian Hart and Ostap Okhrin. “Enhanced method for reinforcement learning based dynamic obstacle avoidance by assessment of collision risk”. en. In: *Neurocomputing* 568 (Feb. 2024), p. 127097. ISSN: 09252312. DOI: 10.1016/j.neucom.2023.127097. URL: https://linkinghub.elsevier.com/retrieve/pii/S0925231223012201 (visited on 02/14/2024).

[13] Peter E Hart, Nils J Nilsson, and Bertram Raphael. “A formal basis for the heuristic determination of minimum cost paths”. In: *IEEE transactions on Systems Science and Cybernetics* 4.2 (1968), pp. 100–107.

[14] “Hiwonder”. In: URL: https://hiwonder.hk/..

[15] David Hoeller et al. *Neural Scene Representation for Locomotion on Structured Terrain*. 2022. arXiv: 2206.08077 [cs.RO]. URL: https://arxiv.org/abs/2206.08077.

[16] Jiang Hua et al. “Learning for a Robot: Deep Reinforcement Learning, Imitation Learning, Transfer Learning”. en. In: *Sensors* 21.4 (Feb. 2021), p. 1278. ISSN: 1424-8220. DOI: 10.3390/s21041278. URL: https://www.mdpi.com/1424-8220/21/4/1278 (visited on 02/14/2024).

[17] Jemin Hwangbo et al. “Learning agile and dynamic motor skills for legged robots”. In: *Science Robotics* 4.26 (Jan. 2019). ISSN: 2470-9476. DOI: 10.1126/scirobotics.aau5872. URL: http://dx.doi.org/10.1126/scirobotics.aau5872.

[18] Julian Ibarz et al. “How to train your robot with deep reinforcement learning: lessons we have learned”. en. In: *The International Journal of Robotics Research* 40.4-5 (Apr. 2021), pp. 698–721. ISSN: 0278-3649, 1741-3176. DOI: 10.1177/0278364920987859. URL: http://journals.sagepub.com/doi/10.1177/0278364920987859 (visited on 02/14/2024).

[19] Simar Kareer et al. *ViNL: Visual Navigation and Locomotion Over Obstacles*. en. arXiv:2210.14791 [cs]. Oct. 2023. URL: http://arxiv.org/abs/2210.14791 (visited on 10/18/2023).

[20] Ashish Kumar et al. *RMA: Rapid Motor Adaptation for Legged Robots*. 2021. arXiv: 2107.04034 [cs.LG]. URL: https://arxiv.org/abs/2107.04034.

[21] Joonho Lee et al. “Learning quadrupedal locomotion over challenging terrain”. In: *Science Robotics* 5.47 (Oct. 2020). ISSN: 2470-9476. DOI: 10.1126/scirobotics.abc5986. URL: http://dx.doi.org/10.1126/scirobotics.abc5986.

[22] Minghuan Liu et al. *Visual Whole-Body Control for Legged Loco-Manipulation.* 2024. arXiv: 2403.16967 [cs.RO]. URL: https://arxiv.org/abs/2403.16967.

[23] Antonio Loquercio, Ashish Kumar, and Jitendra Malik. *Learning Visual Locomotion with Cross-Modal Supervision.* 2022. arXiv: 2211.03785 [cs.AI]. URL: https://arxiv.org/abs/2211.03785.

[24] Gabriel B Margolis et al. *Rapid Locomotion via Reinforcement Learning.* 2022. arXiv: 2205.02824 [cs.RO]. URL: https://arxiv.org/abs/2205.02824.

[25] Takahiro Miki et al. "Learning robust perceptive locomotion for quadrupedal robots in the wild". In: *Science Robotics* 7.62 (Jan. 2022). ISSN: 2470-9476. DOI: 10.1126/scirobotics.abk2822. URL: http://dx.doi.org/10.1126/scirobotics.abk2822.

[26] Andrei Mitriakov et al. "Reinforcement Learning Based, Staircase Negotiation Learning: Simulation and Transfer to Reality for Articulated Tracked Robots". en. In: *IEEE Robotics Automation Magazine* 28.4 (Dec. 2021), pp. 10–20. ISSN: 1070-9932, 1558-223X. DOI: 10.1109/MRA.2021.3114105. URL: https://ieeexplore.ieee.org/document/9583665/ (visited on 02/14/2024).

[27] Andrei Mitriakov et al. "Staircase Traversal via Reinforcement Learning for Active Reconfiguration of Assistive Robots". en. In: *2020 IEEE International Conference on Fuzzy Systems (FUZZ-IEEE).* Glasgow, United Kingdom: IEEE, July 2020, pp. 1–8. ISBN: 978-1-72816-932-3. DOI: 10.1109/FUZZ48607.2020.9177581. URL: https://ieeexplore.ieee.org/document/9177581/ (visited on 02/14/2024).

[28] Wenjuan Ouyang et al. "Adaptive Locomotion Control of a Hexapod Robot via Bio-Inspired Learning". en. In: *Frontiers in Neurorobotics* 15 (Jan. 2021), p. 627157. ISSN: 1662-5218. DOI: 10.3389/fnbot.2021.627157. URL: https://www.frontiersin.org/articles/10.3389/fnbot.2021.627157/full (visited on 02/14/2024).

[29] Kharunya Paramaguru. *Oscar Pistorius: The blade runner makes olympic history.* July 2012. URL: https://olympics.time.com/2012/07/05/oscar-pistorius-the-blade-runner-makes-olympic-history/.

[30] Carlo Rizzardo, Fei Chen, and Darwin Caldwell. "Sim-to-real via latent prediction: Transferring visual non-prehensile manipulation policies". en. In: *Frontiers in Robotics and AI* 9 (Jan. 2023), p. 1067502. ISSN: 2296-9144. DOI: 10.3389/frobt.2022.1067502. URL: https://www.frontiersin.org/articles/10.3389/frobt.2022.1067502/full (visited on 02/14/2024).

[31] Yue Shen et al. "Reinforcement Learning-Based Reactive Obstacle Avoidance Method for Redundant Manipulators". en. In: *Entropy* 24.2 (Feb. 2022), p. 279. ISSN: 1099-4300. DOI: 10.3390/e24020279. URL: https://www.mdpi.com/1099-4300/24/2/279 (visited on 02/14/2024).

[32] Junyao Shi, Tony Dear, and Scott David Kelly. “Deep Reinforcement Learning for Snake Robot Locomotion”. en. In: *IFAC-PapersOnLine* 53.2 (2020), pp. 9688–9695. ISSN: 24058963. DOI: 10.1016/j.ifacol.2020.12.2619. URL: https://linkinghub.elsevier.com/retrieve/pii/S2405896320333772 (visited on 02/14/2024).

[33] Jonah Siekmann et al. “Blind Bipedal Stair Traversal via Sim-to-Real Reinforcement Learning”. en. In: *Robotics: Science and Systems XVII*. Robotics: Science and Systems Foundation, July 2021. ISBN: 978-0-9923747-7-8. DOI: 10.15607/RSS.2021.XVII.061. URL: http://www.roboticsproceedings.org/rss17/p061.pdf (visited on 02/14/2024).

[34] Laura Smith et al. *Legged Robots that Keep on Learning: Fine-Tuning Locomotion Policies in the Real World*. 2021. arXiv: 2110.05457 [cs.RO]. URL: https://arxiv.org/abs/2110.05457.

[35] Seungmoon Song et al. “Deep reinforcement learning for modeling human locomotion control in neuromechanical simulation”. en. In: *Journal of NeuroEngineering and Rehabilitation* 18.1 (Aug. 2021), p. 126. ISSN: 1743-0003. DOI: 10.1186/s12984-021-00919-y. URL: https://jneuroengrehab.biomedcentral.com/articles/10.1186/s12984-021-00919-y (visited on 02/14/2024).

[36] Xingyou Song et al. *Rapidly Adaptable Legged Robots via Evolutionary Meta-Learning*. 2020. arXiv: 2003.01239 [cs.RO]. URL: https://arxiv.org/abs/2003.01239.

[37] Jie Tan et al. *Sim-to-Real: Learning Agile Locomotion For Quadruped Robots*. 2018. arXiv: 1804.10332 [cs.RO].

[38] Jie Tan et al. *Sim-to-Real: Learning Agile Locomotion For Quadruped Robots*. en. arXiv:1804.10332 [cs]. May 2018. URL: http://arxiv.org/abs/1804.10332 (visited on 02/14/2024).

[39] Gabriele Tiboni, Karol Arndt, and Ville Kyrki. “DROPO: Sim-to-real transfer with offline domain randomization”. en. In: *Robotics and Autonomous Systems* 166 (Aug. 2023), p. 104432. ISSN: 09218890. DOI: 10.1016/j.robot.2023.104432. URL: https://linkinghub.elsevier.com/retrieve/pii/S0921889023000714 (visited on 02/14/2024).

[40] Qifeng Wan et al. “Learning and Reusing Quadruped Robot Movement Skills from Biological Dogs for Higher-Level Tasks”. en. In: *Sensors* 24.1 (Dec. 2023), p. 28. ISSN: 1424-8220. DOI: 10.3390/s24010028. URL: https://www.mdpi.com/1424-8220/24/1/28 (visited on 02/14/2024).

[41] Zhaoming Xie et al. *Dynamics Randomization Revisited:A Case Study for Quadrupedal Locomotion*. 2021. arXiv: 2011.02404 [cs.RO]. URL: https://arxiv.org/abs/2011.02404.

[42] Ruihan Yang, Ge Yang, and Xiaolong Wang. *Neural Volumetric Memory for Visual Locomotion Control*. 2023. arXiv: 2304.01201 [cs.RO]. URL: https://arxiv.org/abs/2304.01201.

[43] Ruihan Yang et al. *Learning Vision-Guided Quadrupedal Locomotion End-to-End with Cross-Modal Transformers*. 2022. arXiv: `2107.03996 [cs.LG]`. URL: `https://arxiv.org/abs/2107.03996`.

[44] Liran Yehezkel, Sigal Berman, and David Zarrouk. "Overcoming Obstacles With a Reconfigurable Robot Using Reinforcement Learning". en. In: *IEEE Access* 8 (2020), pp. 217541–217553. ISSN: 2169-3536. DOI: `10.1109/ACCESS.2020.3040896`. URL: `https://ieeexplore.ieee.org/document/9272763/` (visited on 02/14/2024).

[45] Mark Yim, D.G. Duff, and K.D. Roufas. "PolyBot: a modular reconfigurable robot". In: vol. 1. Feb. 2000, 514–520 vol.1. ISBN: 0-7803-5886-4. DOI: `10.1109/ROBOT.2000.844106`.

[46] Wenhao Yu et al. *Sim-to-Real Transfer for Biped Locomotion*. 2019. arXiv: `1903.01390 [cs.RO]`. URL: `https://arxiv.org/abs/1903.01390`.

[47] Wenhao Yu et al. "Visual-Locomotion: Learning to Walk on Complex Terrains with Vision". In: *Proceedings of the 5th Conference on Robot Learning*. Ed. by Aleksandra Faust, David Hsu, and Gerhard Neumann. Vol. 164. Proceedings of Machine Learning Research. PMLR, Aug. 2022, pp. 1291–1302. URL: `https://proceedings.mlr.press/v164/yu22a.html`.

[48] Wenhao Yu et al. "Visual-Locomotion: Learning to Walk on Complex Terrains with Vision". en. In: ().

[49] Avideh Zakhor et al. URL: `https://www-video.eecs.berkeley.edu/papers/avz/revised-icra-2022.pdf`.

[50] Zixian Zang et al. "Perceptive Hexapod Legged Locomotion for Climbing Joist Environments". In: *2023 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. 2023, pp. 2738–2745. DOI: `10.1109/IROS55552.2023.10341957`.

[51] Zixian Zang et al. "Perceptive Hexapod Legged Locomotion for Climbing Joist Environments". en. In: *2023 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. Detroit, MI, USA: IEEE, Oct. 2023, pp. 2738–2745. ISBN: 978-1-66549-190-7. DOI: `10.1109/IROS55552.2023.10341957`. URL: `https://ieeexplore.ieee.org/document/10341957/` (visited on 03/04/2024).

[52] Feng Zhang, Chengbin Xuan, and Hak-Keung Lam. "An obstacle avoidance-specific reinforcement learning method based on fuzzy attention mechanism and heterogeneous graph neural networks". en. In: *Engineering Applications of Artificial Intelligence* 130 (Apr. 2024), p. 107764. ISSN: 09521976. DOI: `10.1016/j.engappai.2023.107764`. URL: `https://linkinghub.elsevier.com/retrieve/pii/S0952197623019486` (visited on 02/14/2024).

[53] Wenshuai Zhao, Jorge Pena Queralta, and Tomi Westerlund. "Sim-to-Real Transfer in Deep Reinforcement Learning for Robotics: a Survey". en. In: *2020 IEEE Symposium Series on Computational Intelligence (SSCI)*. Canberra, ACT, Australia: IEEE, Dec. 2020, pp. 737–744. ISBN: 978-1-72812-547-3. DOI: `10.1109/SSCI47803.2020.9308468`. URL: `https://ieeexplore.ieee.org/document/9308468/` (visited on 02/14/2024).

[54] Ziwen Zhuang et al. "Robot Parkour Learning". In: *Conference on Robot Learning (CoRL)*. 2023.

[55] Ziwen Zhuang et al. "Robot Parkour Learning". In: *Conference on Robot Learning (CoRL)*. 2023.