Legged Locomotion in Challenging Terrains using Egocentric Vision

Anonymous Author(s)
Affiliation
Address
email

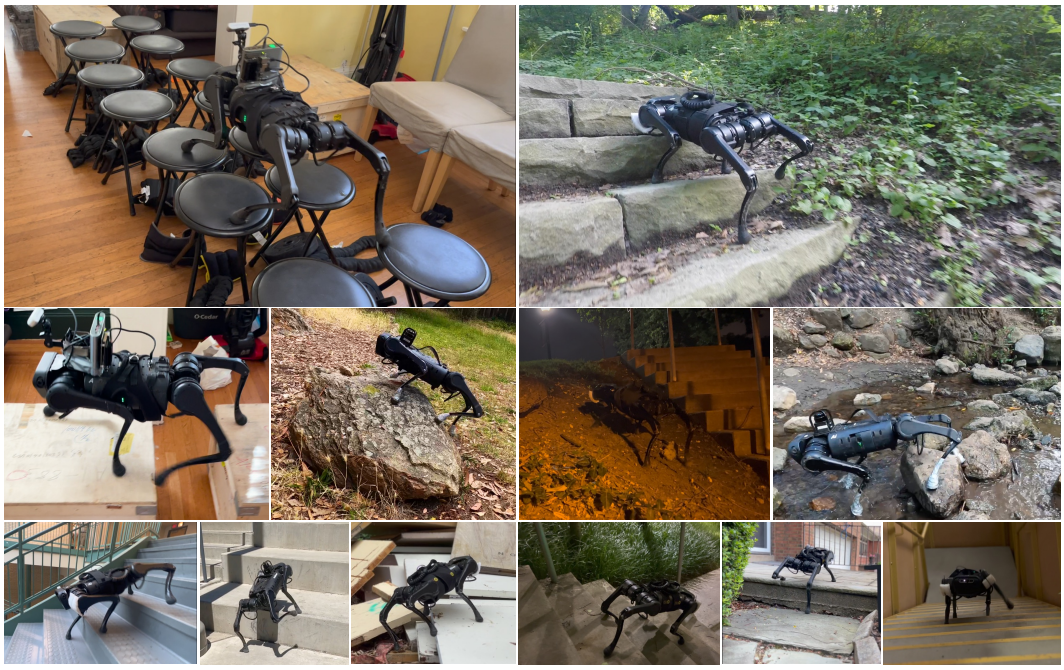


Figure 1: Our robot can traverse a variety of challenging terrain in indoor and outdoor environments, urban and natural settings during day and night using a single front-facing depth camera. The robot can traverse curbs, stairs and moderately rocky terrain. Despite being much smaller than other commonly used legged robots, it is able to climb stairs and curbs of a similar height. Videos at https://blindsupp.github.io/visual-walking/

Abstract: Animals are capable of precise and agile locomotion using vision. Replicating this ability has been a long-standing goal in robotics. The traditional approach has been to decompose this problem into elevation mapping and foothold planning phases. The elevation mapping, however, is susceptible to failure and large noise artifacts, requires specialized hardware, and is biologically implausible. In this paper, we present the first end-to-end locomotion system capable of traversing stairs, curbs, stepping stones, and gaps. We show this result on a medium-sized quadruped robot using a single front-facing depth camera. The small size of the robot necessitates discovering specialized gait patterns not seen elsewhere. The egocentric camera requires the policy to remember past information to estimate the terrain under its hind feet. We train our policy in simulation. Training has two phases - first, we train a policy using reinforcement learning with a cheap-to-compute variant of depth image and then in phase 2 distill it into the final policy that uses depth using supervised learning. The resulting policy transfers to the real world without any fine-tuning and can traverse a large variety of terrain while being robust to perturbations like pushes, slippery surfaces, and rocky terrain. Videos are at https://blindsupp.github.io/visual-walking/

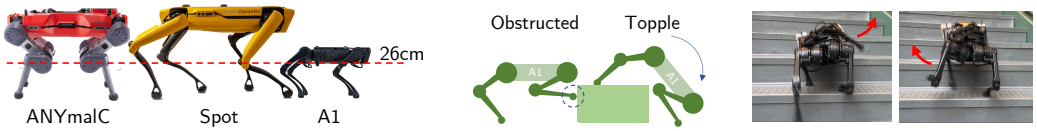
Submitted to the 6th Conference on Robot Learning (CoRL 2022). Do not distribute.

1 Introduction

Of what use is vision during locomotion? Clearly, there is a role of vision in navigation – using maps or landmarks to find a trajectory in the 2D plane to a distant goal while avoiding obstacles. But given a local direction in which to move, it turns out that both humans [1] and robots [2, 3] can do remarkably well at blind walking. Where vision becomes necessary is for locomotion in challenging terrains. In an urban environment, staircases are the most obvious example. In the outdoors, we can deal with rugged terrain such as scrambling over rocks, or stepping from stone to stone to cross a stream of water. There is a fair amount of scientific work studying this human capability and showing tight coupling of motor control with vision [4, 5, 6]. In this paper, we will develop this capability for a quadrupedal walking robot equipped with egocentric depth vision. We use a reinforcement learning approach trained in simulation, which we are directly able to transfer to the real world. Figure 1 and the accompanying videos shows some examples of our robot walking guided by vision.

Humans receive an egocentric stream of vision which is used to control feet placement, typically without conscious planning. As children we acquire it through trial and error [7] but for adults it is an automatized skill. Its unconscious execution should not take away from its remarkable sophistication. The footsteps being placed now are based on information collected some time ago. Typically, we don't look at the ground underneath our feet, rather at the upcoming piece of ground in front of us a few steps away[1, 4, 5, 6]. A short term memory is being created which persists long enough to guide foot placement when we are actually over that piece of ground. Finally, note that we learn to walk through bouts of steps, not by executing pre-programmed gaits [7].

We take these observations about human walking as design principles for the visually-based walking controller for an A1 robot. The walking policy is trained by reinforcement learning with a recurrent neural network being used as a short term memory of recent egocentric views, proprioceptive states, and action history. Such a policy can maintain memory of recent visual information to retrieve characteristics of the terrain under the robot or below the rear feet, which might no longer be directly visible in the egocentric view. Competing approaches which rely on metric localization to construct elevation maps either assume access to perfect elevation map in the real world, or learn a system capable of handling noise which then is forced to rely primarily on proprioception while using vision for only some additional robustness [8, 9, 10]. This hinders the ability of such systems to perform reliably on gaps and stepping stones. We use vision as a first class citizen and show all the uneven terrain capabilities along with a high success rate on crossing gaps and stepping stones.



(a) Robot size comparison (b) Challenges due to size (c) Emergent hip abduction

Figure 2: A smaller robot (a) faces challenges in climbing stairs and curbs due to the stair obstructing its feet while going up and a tendency to topple over when coming down (b). Our robot deals with this by climbing using a large hip abduction that automatically emerges during training (c).

The design principle of not having pre-programmed gait priors turns out to be quite advantageous for our relatively small robot [1] (see Figure 2). Predefined gait priors or reference motions fail to generalize to obstacles of even a reasonable height because of the relatively small size of the quadruped. The emergent behaviors for traversing complex terrains without any priors enable our robot with a standing height of 28cm to traverse the stairs of height upto 25cm, 89% relative to its height, which is significantly higher than any existing methods which typically rely on gait priors. Overall, this use of learning "all the way" and the tight coupling of egocentric vision with motor control are the distinguishing aspects of our approach.

[1] A1 standing height is 40cm as measured by us. Spot, ANYmalC both are 70cm tall reported here and here.

2 Method: Legged Locomotion from Egocentric Vision

We train in two phases, RL in phase 1 and supervised learning in phase 2. During phase 1, we use a low resolution elevation map located in front of the robot. This captures information about the terrain the robot sees from the depth camera while being cheap to compute. In phase 2, we use imitation learning to distil the phase one policy into a policy that has access to a depth map instead of the elevation map (Fig. 3). Since supervised learning is more sample efficient than RL and requires smaller batch sizes, we can train a policy with a manageable number of depth images. We choose observation spaces such that phase 1 policy does not have access to any privileged information not available to phase 2. This allows distillation loss to be low implying that the phase 2 policy's value function will be close to optimal. We derive concrete formulation for this bound in Theorem 2.1.

Theorem 2.1. *$\mathcal{M} = (\mathcal{S}, \mathcal{A}, P, R, \gamma)$ be an MDP with state space $\mathcal{S}$, action space $\mathcal{A}$, transition function $P : \mathcal{S} \times \mathcal{A} \to \mathcal{S}$, reward function $R : \mathcal{A} \times \mathcal{S} \to \mathbb{R}$ and discount factor γ. Let $V^1(s)$ be the value function of the phase 1 policy that is trained to be close to optimal value function $V^*(s)$, i.e., $\left|V^*(s) - V^1(s)\right| < \epsilon$ for all $s \in \mathcal{S}$, and $\pi^1(s)$ be the greedy phase 1 policy obtained from $V^1(s)$. If the phase 2 policy is close to phase 1 $\left|\pi^1(s) - \pi^2(s)\right| < \eta \; \forall \; s$ and R, P are Lipschitz continuous, then the return of phase 2 policy is close to optimal everywhere, i.e., for all s,*

$$\left|V^*(s) - V^{\pi^2}(s)\right| < \frac{2\epsilon\gamma + \eta c}{1 - \gamma}$$

where $c \propto \sum_{s \in \mathcal{S}} V^*(s)$ is a large but bounded constant. (the proof is in appendix)

Interestingly, the above theorem proves that the upper bound on return of phase 2 is proportional to how close it is to the phase 1 policy multiplied by a large factor c. In our method, since the phase 1 and phase 2 policies have the same information contained in their observation space, we can expect distillation loss to be low. This implies that our phase 2 policy will achieve close to optimal performance, and hence, the reward return will be close to optimal too. We now describe each of the phases of training in detail. The coordinate frame we use is fixed to the robot with the x-axis pointing out along its head and z axis pointing vertically upwards.

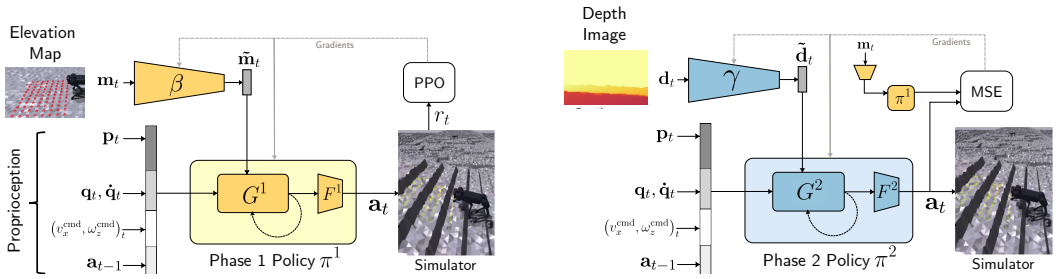
2.1 Phase 1: Reinforcement Learning from Local Elevation Map

Given the local elevation map in front of the robot, proprioception and commanded action, we learn a policy using PPO without gait priors and with biomechanics inspired reward functions to walk on a variety of terrains. We denote the observation space by $\mathbf{o}_t^1 = \left(\mathbf{p}_t, \mathbf{q}_t, \dot{\mathbf{q}}_t, \mathbf{a}_{t-1}, v_x^{\text{cmd}}, w_z^{\text{cmd}}, \mathbf{m}_t\right)$:

- *Proprioception* contains (1) IMU information $\mathbf{p}_t = (\boldsymbol{\omega}_t, \boldsymbol{\theta}_t)$ i.e. the angular velocity $\boldsymbol{\omega}_t \in \mathbb{R}^3$ and roll, pitch values $\boldsymbol{\theta}_t \in \mathbb{R}^2$ of the robot base (2) joint angles $\mathbf{q}_t \in \mathbb{R}^{12}$ and velocities $\dot{\mathbf{q}}_t \in \mathbb{R}^{12}$ from the servo motors. We also include the last action taken $\mathbf{a}_{t-1} \in \mathbb{R}^{12}$ as this leads to smoother policies. Actions and joint positions are normalized to lie in the range $[-1, 1]$ with zero being the mean standing position of the robot.
- *Command* contains target linear velocity in x direction $\left(v_x^{\text{cmd}}\right)_t$ and yaw angular velocity $\left(\omega_z^{\text{cmd}}\right)_t$.
- *Elevation map* is an ego-centric map of the terrain in front of the robot that is close to the field of view of the camera. In particular, it consists of the height values $\mathbf{m}_t = \{h(x, y) \mid (x, y) \in \mathcal{P}\}$ at 88 points $\mathcal{P} = \{0.3, 0.2 \ldots 1.0\} \times \{-0.5, -0.4, \ldots, 0.5\}$. Note that we add small i.i.d. gaussian noise to all input observations (except $\mathbf{a}_{t-1}$) as specified in appendix.

Phase 1 policy The elevation map $\mathbf{m}_t$ is passed through a two-layer MLP β with a $\tanh$ non-linearity on the output to get $\tilde{\mathbf{m}}_t \in \mathbb{R}^{32}$. This information bottleneck forces the network to learn a low dimensional representation of terrain that is less susceptible to noise. $\tilde{\mathbf{m}}_t$ is concatenated along with the rest of the observations and fed to a recurrent policy to obtain actions $\mathbf{a}_t$.

$$\begin{aligned}
\tilde{\mathbf{m}}_t &= \beta\left(\mathbf{m}_t\right) \\
\mathbf{h}_t, \mathbf{c}_t &= G^1\left(\mathbf{p}_t, \mathbf{q}_t, \dot{\mathbf{q}}_t, \mathbf{a}_{t-1}, v_x^{\text{cmd}}, w_z^{\text{cmd}}, \tilde{\mathbf{m}}_t \mid \mathbf{h}_{t-1}, \mathbf{c}_{t-1}\right) \\
\mathbf{a}_t &= F^1\left(\mathbf{h}_t\right)
\end{aligned}$$



(a) Phase 1: Reinforcement Learning

(b) Phase 2: Supervised Learning

Figure 3: We train our locomotion policy in two phases to avoid rendering depth for too many samples. (a) In phase 1, we use RL to train a policy π^1 that has access to a low-resolution elevation map that is cheap to compute. (b) In phase 2, we use π^1 to provide ground truth actions which another policy π^2 is trained to imitate. This student has access to depth map from the front camera.

where G^1 is a GRU with hidden and cell state $\mathbf{h}_t, \mathbf{c}_t$ respectively and F^1 is a two layer MLP with $\tanh$ output non-linearity. The recurrent policy is trained using PPO [11] with backpropagation through time [12] truncated at 24 timesteps. See supplementary for pseudocode.

Rewards Previous work [3, 13] has shown that task agnostic energy minimization based rewards can lead to the emergence of stable and natural gaits that obey high-level commands. We use this same basic reward structure along with penalties to prevent behavior that can damage the robot on complex terrain. Now onwards, we omit the time subscript t for simplicity.

- *Absolute work penalty* $-|\boldsymbol{\tau} \cdot \mathbf{q}|$ where $\boldsymbol{\tau}$ are the joint torques. We use the absolute value so that the policy does not learn to get positive reward by exploiting inaccuracies in contact simulation.
- *Command tracking* $v_x^{\text{cmd}} - \left|v_x^{\text{cmd}} - v_x\right| - |\omega_z^{\text{cmd}} - \omega_z|$ where v_x is velocity of robot in forward direction and ω_z is yaw angular velocity (x, z are coordinate axes fixed to the robot).
- *Foot jerk penalty* $\sum_{i \in \mathcal{F}} \|\mathbf{f}_t^i - \mathbf{f}_{t-1}^i\|$ where $\mathbf{f}_t^i$ is the force at time t on the ith rigid body and $\mathcal{F}$ is the set of feet indices. This prevents large motor backlash.
- *Feet drag penalty* $\sum_{i \in \mathcal{F}} \mathbb{I}\left[f_z^i \geq 1\text{N}\right] \cdot \left(\left|v_x^i\right| + \left|v_y^i\right|\right)$ where $\mathbb{I}$ is the indicator function, and v_x^i, v_y^i is velocity of ith rigid body. This penalizes velocity of feet in the horizontal plane if in contact with the ground preventing feet dragging on the ground which can damage them.
- *Collision penalty* $\sum_{i \in \mathcal{C} \cup \mathcal{T}} \mathbb{I}\left[\mathbf{f}^i \geq 0.1\text{N}\right]$ where $\mathcal{C}, \mathcal{T}$ are the set of calf and thigh indices. This penalizes contacts at the thighs and calves of the robot which would otherwise graze against edges of stairs and discrete obstacles.
- *Survival bonus* constant value 1 at each time step to prioritize survival over following commands in challenging situations.

Notice that these reward functions do not define any gait priors and the optimal gait is allowed emerge via RL. Target heading values h_t^{cmd} are sampled and commanded angular velocities is computed as $\left(\omega_z^{\text{cmd}}\right)_t = 0.5 \cdot \left(h_t^{\text{cmd}} - h_t\right)$ where h_t is the current heading value. When walking on terrain, $\left(v_x^{\text{cmd}}\right)_t = 0.35$m/s and heading is varied in $h_t^{\text{cmd}} \in [-10°, 10°]$. On flat ground, one of three sampling modes are chosen uniformly at random - curve following, in-place turning and complete stop. In the curve following regime $\left(v_x^{\text{cmd}}\right)_t \in [0.2\text{m/s}, 0.75\text{m/s}]$ while $h_t^{\text{cmd}} \in [-60°, 60°]$. For in-place turning, $\left(v_x^{\text{cmd}}\right)_t = 0$ and $h_t^{\text{cmd}} \in [-180°, 180°]$. In complete stop, $\left(v_x^{\text{cmd}}\right)_t = 0, h_t^{\text{cmd}} = 0$. This scheme is designed to mimic the distribution of commands the robot sees during operation. We terminate if the pitch exceeds 90° or if the base or head of the robot collides with an object.

Training environment Similar to [14] we generate different sets of terrain (see Figure in appendix) of varying difficulty level. Following [3], we generate fractal variations over each of the terrains to get robust walking behaviour. At training time, the environments are arranged in a 6×10 matrix with each row having terrain of the same type and difficulty increasing from left to right. We train with a curriculum over terrain [14] where robots are first initialized on easy terrain and promoted to

harder terrain if they traverse more than half its length. They are demoted to easier terrain if they fail to travel atleast half the commanded distance $v_x^{\text{cmd}}T$ where T is maximum episode length.

We randomize parameters of the simulation and add small i.i.d. gaussian noise to observations to bridge the sim2real gap and make the policy robust (see Tables in appendix).

2.2 Phase 2: Supervised Learning

Having learnt a useful visuomotor policy in phase 1, we can now use supervised learning to distil these into a phase 2 policy that receives depth input $\mathbf{d}_t$. We can use much smaller batch sizes and samples to train this since the phase 1 policy provides high-quality supervision at each time-step.

Phase 2 policy We create a copy of the recurrent base policy $G^2 \leftarrow G^1, F^2 \leftarrow F^1$. The elevation map compression MLP β is replaced with a convolutional depth backbone γ which processes depth map $\mathbf{d}_t \in \mathbb{R}^{58\times 87}$ to produce a depth latent $\tilde{\mathbf{d}}_t \in \mathbb{R}^{32}$, predicted actions $\hat{\mathbf{a}}_t$ are then computed:

$$\begin{aligned}
\tilde{\mathbf{d}}_t &= \gamma\left(\mathbf{d}_t\right) \\
\mathbf{h}_t, \mathbf{c}_t &= G^2\left(\mathbf{p}_t, \mathbf{q}_t, \dot{\mathbf{q}}_t, \mathbf{a}_{t-1}, v_x^{\text{cmd}}, w_z^{\text{cmd}}, \tilde{\mathbf{d}}_t \mid \mathbf{h}_{t-1}, \mathbf{c}_{t-1}\right) \\
\hat{\mathbf{a}}_t &= F^2\left(\mathbf{h}_t\right)
\end{aligned}$$

Phase 2 training We use DAgger [15] with truncated backpropagation through time. In particular, we unroll the student inside the simulator for $N = 24$ timesteps and then label each of the states encountered with the ground truth action $\mathbf{a}_t$ from phase 1. We minimize mean squared error between phase 2 actions $\hat{\mathbf{a}}_t$ and $\mathbf{a}_t$ (see supp.)

$$\underset{\gamma, G^2, F^2}{\operatorname{argmin}} \underset{s_\tau \sim \rho(\pi^2)}{\mathbb{E}} \left[\sum_{t=\tau}^{\tau+N-1} \left\|\hat{\mathbf{a}}_t - \mathbf{a}_t\right\|^2\right]$$

where $\mathbf{a}_t = \pi^1\left(\mathbf{s}_t\right), \hat{\mathbf{a}}_t = \pi^2\left(\mathbf{s}_t\right), \mathbf{s}_{t+1} = \pi^2\left(\mathbf{s}_t\right)$ and $\rho(\pi^2)$ is the state-visitation probability of π^2.

Depth map is obtained from a Intel RealSense camera inside the head of the robot. The camera captures images every 100ms $\pm$ 20ms at a resolution of 480×848. We preprocess the image by cropping 200 white pixels from the left, applying nearest neighbor hole-filling and downsampling to 58×87. This is passed through the depth backbone γ to obtain the latent $\tilde{\mathbf{d}}_t$ which is sent over a UDP socket to the base policy. This has a latency of 10 ± 10ms which we account for during phase 1.

Deployment The student can be deployed as-is on the hardware using only the available onboard compute. It is able to handle camera failures and the asynchronous nature of depth due to the randomizations we apply during phase 1. It is robust to pushes, slippery surfaces and large rocky surfaces and can climb stairs, curbs, and cross gaps and stepping stones.

3 Experimental Setup

We use the Unitree A1 robot pictured in Fig. 2. The robot has 12 actuated joints. The robot has a front-facing Intel RealSense depth camera in its head. The onboard compute consists of the UPboard and a Jetson NX. The policy operates at 50Hz and sends joint position commands which are converted to torques by a low-level PD controller running at 400Hz. More details in the appendix. For simulation, we use the IsaacGym (IG) simulator with the legged_gym library [14] to develop walking policies. We construct a large elevation map with 100 sub-terrains arranged in a 20×10 grid. Each row has the same type of terrain arranged in increasing difficulty while different rows have different terrain.

Baselines We compare against three baselines, each of which uses the same number of learning samples for both RL phase and supervised learning phase.

- **Blind policy** trained with the elevation map observations $\mathbf{m}_t$ masked with zeros. This policy must rely on proprioception to traverse terrain like stairs and gaps. This baseline helps quantify the benefit of vision for walking.

Terrain	Average Distance (↑)				Mean Time to Fall (s)			
	Blind	Privileged	Noisy	Ours	Blind	Privileged	Noisy	Ours
Slopes	29.1	18.0	26.0	**31.5**	155.5	13.3	18.0	89.2
Stepping Stones	0.5	2.2	5.3	**18.1**	3.8	1.9	4.3	46.3
Stairs	11.1	7.0	12.5	**23.8**	70.6	5.8	9.6	60.5
Discrete Obstacles	22.0	18.3	26.0	**30.0**	124.1	12.7	17.7	80.6

Table 1: We measure the average distance travelled and mean time to fall for all methods on different terrains in simulation. For each method, we train a single policy for all terrains and use that for evaluation. We see that our method outperforms the baselines on all terrains for average distanced traveled. On stairs, slopes and discrete terrain, we see that the blind policy survives for the longest, despite the lower average distance recorded implying that it simply gets stuck. However, in the more risky stepping stones task, it falls and is unable to make progress, or survive. Although the blind policy makes non trivial progress on stairs, discrete terrain and slopes, using vision can potentially double the average distanced travelled before getting stuck or falling. The privileged and noisy baselines get a similar performance and don't perform significantly better than the blind baseline because they both suffer from information asymmetry between teacher and student. This empirically validates Theorem 2.1 that it is indeed beneficial to train without a privileged teacher.

- **Privileged Teacher** teacher policy is trained with a noiseless elevation map completely surrounding the robot (see supp). This elevation map is updated at each time step (50Hz) and has no latency. The teacher policy also has no execution lag. The privileged teacher is then distilled to a student with laggy, low frequency front camera depth and execution lag. The key difference between this baseline and our method is that we add lag, latency and limited view of the terrain to both the teacher and the student whereas here it is only present in the student and we must rely on supervised learning in phase 2 to estimate privileged information from limited data.
- **Noisy** Methods which rely on elevation maps constructed using depth and tracking cameras often have large noise due to sensor deficiencies. As a result, large noise must be added during simulation as well. We train a privileged teacher as in the previous baseline and distil it to a student with large noise added to the elevation map. We use the noise model in [9]. This baseline is a close (but not exact) instantiation of Miki et al. [9] and helps us to show the benefits of directly using the depth image which has lower noise and not relying on a tracking camera.

4 Results and Analysis

Simulation Results We report mean time to fall and mean distance travelled before crashing for different terrain and baselines in Table 1. For each method, we train a single policy for all terrains and use that for evaluation. Although the blind policy makes non trivial progress on stairs, discrete terrain and slopes, it is significantly less efficient at traversing these terrains. On slopes our method travels a greater distance in nearly half of the time implying that the blind baseline gets stuck often. Similarly, on stairs and discrete obstacles the distance travelled by the vision baseline is much greater in a shorter amount of time. The noisy and privileged baselines get similar average distances and mean time to fall and both are worse than our policy. This is empirically validates our Theorem 2.1 which formalizes the intuition that information asymmetry leads to degradation in performance during distillation. On slopes, stairs and discrete obstacles our policy consistently travels a longer distance, outperforming the baselines. This trend is even more significant on the stepping stones terrain where all baselines barely make any progress. The blind policy has no way of estimating the position of the stone and crashes as soon as it steps into the gap. For the noisy policy, the large amount of added noise makes it impossible for the student to reliably ascertain the location of the stones since it cannot rely on proprioception any more. The privileged teacher distilled into student with egocentric vision doesn't work well due to information asymmetry.

Real World Comparisons We compare the performance of our method to the blind baseline in the real world. In particular we have 4 testing setups as shows in Figure 4: Upstairs, Downstairs, Gaps and Stepping stones. We use a separate policy for stairs, which is distilled into the front camera,

	Success	#Stairs
Ours	100%	13
Blind	0%	2.2

	Success	#Stairs
Ours	100%	13
Blind	100%	13

	Success	#Stones
Ours	94%	9.4
Blind	0%	0

	Success
Ours	100%
Blind	0%

Figure 4: We show success rates and time-to-failure (TTF) for our method and the blind baseline on curbs, stairs, stepping stones and gaps. We use a separate policy for stairs which is distilled to front camera, and use a separate policy trained on stepping stones distilled to the top camera which we use for gaps and stepping stones. We observe that our method solves all the tasks perfectly except for the stepping stone task in which the robot achieves 94% success. The blind baseline fails completely on gaps and stepping stones. For upstairs, it makes some progress, but fails to complete the entire staircase even once, which is expected given the small size of the robot. The blind policy completes the downstairs task 100% success, although it learns a very high impact falling gait to solve the task. In our experiments, the robot dislocates its real right leg during the blind downstairs trials.

and a different one for gaps and stepping stones, which is distilled to a top camera. We see that the blind baseline is incapable of walking upstairs beyond a few steps and fails to complete the staircase even once. Although existing methods have shown stairs for blind robots, we note that our robot is relatively smaller making it a more challenging task for a blind robot. On downstairs, we observe that the blind baseline achieves 100% success, although it learns to fall on every step and stabilize leading to a very high impact gait which led to the detaching of the rear right hip of the robot during our experiments. We additionally show results in stepping stones and gaps, where the blind robot fails completely establishing the hardness of these setups and the necessity of vision to solve them. We show a 100% success on all tasks except for stepping stone on which we achieve 94% success, which is very high given the challenging setup.

Urban Environments We experiment on stairs, ramps and curbs (Fig. 1). The robot was successfully able to go upstairs as well as downstairs for stairs of height upto 24cm in height and 28cm as the lowest width. Since the robot has to remember the terrain under its body from recent visual history, it sometimes misses a step, but shows impressive recovery behaviour and continues climbing or descending. The robot is able to climb curbs and obstacles as high as $26cm$ which is almost as high as the robot 2. This requires an emergent hip abduction movement which is necessary because the small size of the robot doesn't leave any space between the body and stair for the leg to step up. Hip abduction behavior emerges because of our tabula rasa approach to learning gaits without reliance on priors or datasets of natural motion.

Gaps and Stepping Stones We construct an obstacle course consisting of gaps and stepping stones out of tables and stools (Fig. 4). For this set of experiments we use a policy trained on stepping stones on gaps, and distilled onto the top camera instead of the front camera. The robot achieves a 100% success rate on gaps of upto 26cm from egocentric depth and 94% on difficult stepping stones. The stepping stones experiment shows that our visual policy can learn safe foothold placement behavior even without an explicit elevation map or foothold optimization objectives. The blind baseline achieves zero success rate on both tasks and falls as soon as any gap is encountered.

Natural Environments We also deploy our policy on outdoor hikes and rocky terrains next to river beds (Fig. 1). We see that the robot is able to successfully traverse rugged stairs covered with dirt, small pebbles and some large rocks. It also avoids stumbling over large tree roots on the hiking trail. On the beach, we see that the robot is able to successfully navigate the terrain despite several slips

and unstable footholds given the nature of the terrain. We see that the robot sometimes get stuck in the crevices and in some cases shows impressive recovery behavior as well.

5 Related Work

Legged locomotion Legged locomotion an important problem which has been studied for decades. Several classical works use model based techniques, or define heuristic reactive controllers to achieve the task of walking [16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28].This method has led to several promising results in the real world, although they still lack the generality needed to deploy them in the real world. This has motivated work in using RL for learning to walk in simulation [11, 29, 30, 31], and then successfully deploy them in a diverse set of real world scenarios [32, 33, 34, 35, 36, 37, 32, 38]. Alternatively, a policy learned in simulation can be adapted at test-time to work well in real environments [39, 40, 41, 42, 43, 44, 45, 46, 3, 47, 48, 49]. However, most of these methods are blind, and only use proprioceptive signal to walk.

Locomotion from Elevation Maps To achieve visual control of walking, classical methods decouples the perception and control aspects, assuming a perfect output from perception, such as an elevation map, and then using it for planning and control [50, 51, 52, 53, 54]. The control part can be further decoupled into searching for feasible footholds on the elevation map and then execute it with a low-level policy Chestnutt [55]. The foothold feasibility scores can either be estimated heuristically [56, 57, 8, 58, 59, 10, 60] or learned [61, 62, 63, 64, 65]. Other methods forgo explicit foothold optimization and learn traversibility maps instead [66, 67, 68, 69]. Recent methods skip foothold planning and directly train a deep RL policy that takes the elevation map as input and outputs either low-level motor primitives [9, 70] or raw joint angles [14, 71, 72, 73]. Elevation maps can be noisy or incorrect and dealing with imperfect maps is a major challenge to building robust locomotion systems. Solutions to this include incorporating uncertainty in the elevation map [50, 74, 75] and simulating errors at training time to make the walking policy robust to them [9].

Locomotion from Egocentric Depth Closest to our method is the line of work that does not construct explicit elevation maps and predicts actions directly from depth frames. [49] learn a policy that for obstacle avoidance from egocentric depth on flat terrain, [76] train a hierarchical policy which uses depth to traverse curved cliffs and mazes in simulation, [77] use lidar scans to show zero-shot generalization to difficult terrains. Yu et al. [78] train a policy to step over gaps by predicting high-level actions from egocentric depth from the head and below the torso. Relatedly, Margolis et al. [79] train a high-level policy to jump over gaps from egocentric depth using a whole body impulse controller. In contrast, we directly predict target joint angles from egocentric depth without constructing metric elevation maps.

Teacher-Student Training To accelerate simulation training, several methods use privileged terrain information and then distill it to a student policy which resembles the deployment system. This technique has been fruitfully applied to self-driving [80, 81], autonomous drone flight [82, 83], among others. Closest to our work, [9, 79] train a policy with access to ground truth elevation map to predict either high-level actions or residual over predefined gait priors. We follow a similar two phase training, but differ from prior work in avoiding the use of privileged information in phase 1 training. We formally show the benefit of such a separation, and empirically show superior performance.

6 Discussion and Limitations

In this work, we show an end-to-end approach to walking with egocentric depth that can traverse a large variety of terrains including stairs, gaps and stepping stones. However, there can be certain instances where the robot fails because of a visual or terrain mismatch between the simulation and the real world. The only solution to this problem under the current paradigm is to engineer the situation back into simulation and retrain. This poses a fundamental limitation to this approach and in future, we would like to leverage the data collected in the real world to continue improving both the visual and the motor performance.

References

[1] J. M. Loomis, J. A. Da Silva, N. Fujita, and S. S. Fukusima. Visual space perception and visually directed action. *Journal of experimental psychology: Human Perception and Performance*, 18 (4):906, 1992.

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

[3] A. Kumar, Z. Fu, D. Pathak, and J. Malik. RMA: Rapid Motor Adaptation for Legged Robots. In *RSS*, 2021.

[4] J. S. Matthis and B. R. Fajen. Visual control of foot placement when walking over complex terrain. *Journal of experimental psychology: human perception and performance*, 40(1):106, 2014.

[5] A. E. Patla. Understanding the roles of vision in the control of human locomotion. *Gait & posture*, 5(1):54–69, 1997.

[6] A. A. Mohagheghi, R. Moraes, and A. E. Patla. The effects of distant and on-line visual information on the control of approach phase and step over an obstacle during locomotion. *Experimental brain research*, 155(4):459–468, 2004.

[7] K. E. Adolph, W. G. Cole, M. Komati, J. S. Garciaguirre, D. Badaly, J. M. Lingeman, G. L. Chan, and R. B. Sotsky. How do you learn to walk? thousands of steps and dozens of falls per day. *Psychological science*, 2012.

[8] F. Jenelten, T. Miki, A. E. Vijayan, M. Bjelonic, and M. Hutter. Perceptive locomotion in rough terrain–online foothold optimization. *RA-L*, 2020.

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

[10] D. Kim, D. Carballo, J. Di Carlo, B. Katz, G. Bledt, B. Lim, and S. Kim. Vision aided dynamic exploration of unstructured terrain with a small-scale quadruped robot. In *ICRA*, 2020.

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

[12] P. J. Werbos. Backpropagation through time: what it does and how to do it. *Proceedings of the IEEE*, 78(10):1550–1560, 1990.

[13] Z. Fu, A. Kumar, A. Agarwal, H. Qi, J. Malik, and D. Pathak. Coupling vision and proprioception for navigation of legged robots. *arXiv preprint arXiv:2112.02094*, 2021.

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

[15] S. Ross, G. Gordon, and D. Bagnell. A reduction of imitation learning and structured prediction to no-regret online learning. In *Proceedings of the fourteenth international conference on artificial intelligence and statistics*, 2011.

[16] H. Miura and I. Shimoyama. Dynamic walk of a biped. *IJRR*, 1984.

[17] M. H. Raibert. Hopping in legged systems—modeling and simulation for the two-dimensional one-legged case. *IEEE Transactions on Systems, Man, and Cybernetics*, 1984.

[18] H. Geyer, A. Seyfarth, and R. Blickhan. Positive force feedback in bouncing gaits? *Proceedings of the Royal Society of London. Series B: Biological Sciences*, 2003.

[19] K. Yin, K. Loken, and M. Van de Panne. Simbicon: Simple biped locomotion control. *ACM Transactions on Graphics*, 2007.

[20] K. Sreenath, H.-W. Park, I. Poulakakis, and J. W. Grizzle. A compliant hybrid zero dynamics controller for stable, efficient and fast bipedal walking on mabel. *IJRR*, 2011.

[21] A. M. Johnson, T. Libby, E. Chang-Siu, M. Tomizuka, R. J. Full, and D. E. Koditschek. Tail assisted dynamic self righting. In *Adaptive Mobile Robotics*. World Scientific, 2012.

[22] M. Khoramshahi, H. J. Bidgoly, S. Shafiee, A. Asaei, A. J. Ijspeert, and M. N. Ahmadabadi. Piecewise linear spine for speed–energy efficiency trade-off in quadruped robots. *Robotics and Autonomous Systems*, 2013.

[23] A. D. Ames, K. Galloway, K. Sreenath, and J. W. Grizzle. Rapidly exponentially stabilizing control lyapunov functions and hybrid zero dynamics. *IEEE Transactions on Automatic Control*, 2014.

[24] D. J. Hyun, J. Lee, S. Park, and S. Kim. Implementation of trot-to-gallop transition and subsequent gallop on the mit cheetah i. *IJRR*, 2016.

[25] M. Barragan, N. Flowers, and A. M. Johnson. MiniRHex: A small, open-source, fully programmable walking hexapod. In *RSS Workshop*, 2018.

[26] G. Bledt, M. J. Powell, B. Katz, J. Di Carlo, P. M. Wensing, and S. Kim. Mit cheetah 3: Design and control of a robust, dynamic quadruped robot. In *IROS*, 2018.

[27] M. Hutter, C. Gehring, D. Jud, A. Lauber, C. D. Bellicoso, V. Tsounis, J. Hwangbo, K. Bodie, P. Fankhauser, M. Bloesch, et al. Anymal-a highly mobile and dynamic quadrupedal robot. In *IROS*, 2016.

[28] C. S. Imai, M. Zhang, Y. Zhang, M. Kierebinski, R. Yang, Y. Qin, and X. Wang. Vision-guided quadrupedal locomotion in the wild with multi-modal delay randomization. *arXiv:2109.14549*, 2021.

[29] T. P. Lillicrap, J. J. Hunt, A. Pritzel, N. Heess, T. Erez, Y. Tassa, D. Silver, and D. Wierstra. Continuous control with deep reinforcement learning. In *ICLR*, 2016.

[30] V. Mnih, A. P. Badia, M. Mirza, A. Graves, T. Lillicrap, T. Harley, D. Silver, and K. Kavukcuoglu. Asynchronous methods for deep reinforcement learning. In *ICML*, 2016.

[31] S. Fujimoto, H. Hoof, and D. Meger. Addressing function approximation error in actor-critic methods. In *ICML*, 2018.

[32] J. Tan, T. Zhang, E. Coumans, A. Iscen, Y. Bai, D. Hafner, S. Bohez, and V. Vanhoucke. Sim-to-real: Learning agile locomotion for quadruped robots. In *RSS*, 2018.

[33] J. Tobin, R. Fong, A. Ray, J. Schneider, W. Zaremba, and P. Abbeel. Domain randomization for transferring deep neural networks from simulation to the real world. In *IROS*, 2017.

[34] X. B. Peng, M. Andrychowicz, W. Zaremba, and P. Abbeel. Sim-to-real transfer of robotic control with dynamics randomization. In *ICRA*, 2018.

[35] Z. Xie, X. Da, M. van de Panne, B. Babich, and A. Garg. Dynamics randomization revisited: A case study for quadrupedal locomotion. In *ICRA*, 2021.

[36] O. Nachum, M. Ahn, H. Ponte, S. S. Gu, and V. Kumar. Multi-agent manipulation via locomotion using hierarchical sim2real. In *CoRL*, 2020.

[37] J. Hwangbo, J. Lee, A. Dosovitskiy, D. Bellicoso, V. Tsounis, V. Koltun, and M. Hutter. Learning agile and dynamic motor skills for legged robots. *Science Robotics*, 2019.

[38] J. Hanna and P. Stone. Grounded action transformation for robot learning in simulation. In *AAAI*, 2017.

[39] W. Yu, J. Tan, C. K. Liu, and G. Turk. Preparing for the unknown: Learning a universal policy with online system identification. In *RSS*, 2017.

[40] W. Yu, C. K. Liu, and G. Turk. Policy transfer with strategy optimization. In *ICLR*, 2018.

[41] X. B. Peng, E. Coumans, T. Zhang, T.-W. E. Lee, J. Tan, and S. Levine. Learning agile robotic locomotion skills by imitating animals. In *RSS*, 2020.

[42] W. Zhou, L. Pinto, and A. Gupta. Environment probing interaction policies. In *ICLR*, 2019.

[43] W. Yu, V. C. V. Kumar, G. Turk, and C. K. Liu. Sim-to-real transfer for biped locomotion. In *IROS*, 2019.

[44] W. Yu, J. Tan, Y. Bai, E. Coumans, and S. Ha. Learning fast adaptation with meta strategy optimization. *RA-L*, 2020.

[45] X. Song, Y. Yang, K. Choromanski, K. Caluwaerts, W. Gao, C. Finn, and J. Tan. Rapidly adaptable legged robots via evolutionary meta-learning. In *IROS*, 2020.

[46] I. Clavera, A. Nagabandi, S. Liu, R. S. Fearing, P. Abbeel, S. Levine, and C. Finn. Learning to adapt in dynamic, real-world environments through meta-reinforcement learning. In *ICLR*, 2019.

[47] Z. Fu, A. Kumar, J. Malik, and D. Pathak. Minimizing energy consumption leads to the emergence of gaits in legged robots. In *CoRL*, 2021.

[48] L. Smith, J. C. Kew, X. B. Peng, S. Ha, J. Tan, and S. Levine. Legged robots that keep on learning: Fine-tuning locomotion policies in the real world. In *ICRA*, 2022.

[49] R. Yang, M. Zhang, N. Hansen, H. Xu, and X. Wang. Learning vision-guided quadrupedal locomotion end-to-end with cross-modal transformers. In *ICLR*, 2022.

[50] P. Fankhauser, M. Bloesch, and M. Hutter. Probabilistic terrain mapping for mobile robots with uncertain localization. *IEEE Robotics and Automation Letters*, 3(4):3019–3026, 2018.

[51] Y. Pan, X. Xu, Y. Wang, X. Ding, and R. Xiong. Gpu accelerated real-time traversability mapping. In *2019 IEEE International Conference on Robotics and Biomimetics (ROBIO)*, pages 734–740, 2019. doi:10.1109/ROBIO49542.2019.8961816.

[52] I.-S. Kweon, M. Hebert, E. Krotkov, and T. Kanade. Terrain mapping for a roving planetary explorer. In *IEEE International Conference on Robotics and Automation*, pages 997–1002. IEEE, 1989.

[53] I.-S. Kweon and T. Kanade. High-resolution terrain map from multiple sensor data. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 14(2):278–292, 1992.

[54] A. Kleiner and C. Dornhege. Real-time localization and elevation mapping within urban search and rescue scenarios. *Journal of Field Robotics*, 24(8-9):723–745, 2007.

[55] J. Chestnutt. *Navigation planning for legged robots*. Carnegie Mellon University, 2007.

[56] M. Wermelinger, P. Fankhauser, R. Diethelm, P. Krüsi, R. Siegwart, and M. Hutter. Navigation planning for legged robots in challenging terrain. In *IROS*, 2016.

[57] A. Chilian and H. Hirschmüller. Stereo camera based navigation of mobile robots on rough terrain. In *IROS*, 2009.

[58] C. Mastalli, I. Havoutis, A. W. Winkler, D. G. Caldwell, and C. Semini. On-line and on-board planning and perception for quadrupedal locomotion. In *2015 IEEE International Conference on Technologies for Practical Robot Applications*, 2015.

[59] P. Fankhauser, M. Bjelonic, C. D. Bellicoso, T. Miki, and M. Hutter. Robust rough-terrain locomotion with a quadrupedal robot. In *ICRA*, 2018.

[60] A. Agrawal, S. Chen, A. Rai, and K. Sreenath. Vision-aided dynamic quadrupedal locomotion on discrete terrain using motion libraries. *arXiv preprint arXiv:2110.00891*, 2021.

[61] J. Z. Kolter, M. P. Rodgers, and A. Y. Ng. A control architecture for quadruped locomotion over rough terrain. In *ICRA*, 2008.

[62] M. Kalakrishnan, J. Buchli, P. Pastor, and S. Schaal. Learning locomotion over rough terrain using terrain templates. In *IROS*, 2009.

[63] L. Wellhausen and M. Hutter. Rough terrain navigation for legged robots using reachability planning and template learning. In *IROS*, 2021.

[64] C. Mastalli, M. Focchi, I. Havoutis, A. Radulescu, S. Calinon, J. Buchli, D. G. Caldwell, and C. Semini. Trajectory and foothold optimization using low-dimensional models for rough terrain locomotion. In *ICRA*, 2017.

[65] O. A. V. Magana, V. Barasuol, M. Camurri, L. Franceschi, M. Focchi, M. Pontil, D. G. Caldwell, and C. Semini. Fast and continuous foothold adaptation for dynamic locomotion through cnns. *RA-L*, 2019.

[66] B. Yang, L. Wellhausen, T. Miki, M. Liu, and M. Hutter. Real-time optimal navigation planning using learned motion costs. In *ICRA*, 2021.

[67] R. O. Chavez-Garcia, J. Guzzi, L. M. Gambardella, and A. Giusti. Learning ground traversability from simulations. *RA-L*, 2018.

[68] J. Guzzi, R. O. Chavez-Garcia, M. Nava, L. M. Gambardella, and A. Giusti. Path planning with local motion estimations. *RA-L*, 2020.

[69] S. Gangapurwala, M. Geisert, R. Orsolino, M. Fallon, and I. Havoutis. Real-time trajectory adaptation for quadrupedal locomotion using deep reinforcement learning. In *International Conference on Robotics and Automation (ICRA)*, 2021.

[70] V. Tsounis, M. Alge, J. Lee, F. Farshidian, and M. Hutter. Deepgait: Planning and control of quadrupedal gaits using deep reinforcement learning. *IEEE Robotics and Automation Letters*, 5 (2):3699–3706, 2020.

[71] X. B. Peng, G. Berseth, and M. Van de Panne. Terrain-adaptive locomotion skills using deep reinforcement learning. *ACM Transactions on Graphics (TOG)*, 35(4):1–12, 2016.

[72] X. B. Peng, G. Berseth, K. Yin, and M. Van De Panne. Deeploco: Dynamic locomotion skills using hierarchical deep reinforcement learning. *ACM Transactions on Graphics (TOG)*, 36(4): 1–13, 2017.

[73] Z. Xie, H. Y. Ling, N. H. Kim, and M. van de Panne. Allsteps: Curriculum-driven learning of stepping stone skills. In *Computer Graphics Forum*, volume 39, pages 213–224. Wiley Online Library, 2020.

[74] D. Belter, P. Łabcki, and P. Skrzypczyński. Estimating terrain elevation maps from sparse and uncertain multi-sensor data. In *2012 IEEE International Conference on Robotics and Biomimetics (ROBIO)*, pages 715–722, 2012. doi:10.1109/ROBIO.2012.6491052.

[75] P. Fankhauser, M. Bloesch, C. Gehring, M. Hutter, and R. Siegwart. Robot-centric elevation mapping with uncertainty estimates. 09 2014. doi:10.1142/9789814623353_0051.

[76] D. Jain, A. Iscen, and K. Caluwaerts. From pixels to legs: Hierarchical learning of quadruped locomotion. *arXiv preprint arXiv:2011.11722*, 2020.

[77] A. Escontrela, G. Yu, P. Xu, A. Iscen, and J. Tan. Zero-shot terrain generalization for visual locomotion policies. *arXiv preprint arXiv:2011.05513*, 2020.

[78] W. Yu, D. Jain, A. Escontrela, A. Iscen, P. Xu, E. Coumans, S. Ha, J. Tan, and T. Zhang. Visual-locomotion: Learning to walk on complex terrains with vision. In *5th Annual Conference on Robot Learning*, 2021.

[79] G. B. Margolis, T. Chen, K. Paigwar, X. Fu, D. Kim, S. Kim, and P. Agrawal. Learning to jump from pixels. *arXiv preprint arXiv:2110.15344*, 2021.

[80] D. Chen, B. Zhou, V. Koltun, and P. Krähenbühl. Learning by cheating. In *Conference on Robot Learning*, pages 66–75. PMLR, 2020.

[81] J. Janai, F. Güney, A. Behl, A. Geiger, et al. Computer vision for autonomous vehicles: Problems, datasets and state of the art. *Foundations and Trends® in Computer Graphics and Vision*, 12(1–3):1–308, 2020.

[82] E. Kaufmann, A. Loquercio, R. Ranftl, A. Dosovitskiy, V. Koltun, and D. Scaramuzza. Deep drone racing: Learning agile flight in dynamic environments. In *Conference on Robot Learning*, pages 133–145. PMLR, 2018.

[83] E. Kaufmann, A. Loquercio, R. Ranftl, M. Müller, V. Koltun, and D. Scaramuzza. Deep drone acrobatics. *arXiv preprint arXiv:2006.05768*, 2020.