

# Learning to Walk over Structured Terrains by Imitating MPC

Chao Ni, Alexander Reske, Takahiro Miki, Jan Carius, Ruben Grandia, Marco Hutter

**Abstract**— Walking over structured terrain requires perception awareness of the environment. The robot needs to extract information from exteroceptive sensors and generate control signals by either model-based approaches, such as Model Predictive Control (MPC), or inferring a neural network, trained with Reinforcement Learning (RL) or Imitation Learning (IL). While MPC can produce accurate solutions, it relies on an accurate perceptual sensor and is constrained by computational costs. RL solves such problems by designing rewards for a successful traverse. But designing rewards can be tedious and training an RL often takes days. IL trains the policy by learning from the expert demonstration and requires less time if good demonstrations are available. MPC-Net is an IL approach that learns from MPC expert. It minimizes the control Hamiltonian of the Optimal Control (OC) problem instead of imitating the observation-action mapping directly as Behavioral Cloning (BC). In this work, we add perception to MPC-Net, using demonstrations from perceptive MPC, which can walk over structured obstacles. To train the policy, we use a teacher-student learning framework to incorporate noisy exteroceptive information.

We benchmark our MPC-Net teacher and validate our student policy on the hardware. We show that our student policy outperforms MPC expert under noisy perception inputs. Moreover, we compare the performance of the robot when walking over a high step with Perceptive MPC-Net against blind MPC-Net policy.

## I. INTRODUCTION

Legged robots, such as ANYmal [1], Laikago, Cassie, and MIT cheetah [2], have recently shown great ability to cope with challenging terrains, such as slopes [3], [4], stairs [5], [6], and stepping stones [7], [8]. The agility of the leg makes it suitable for conducting tasks in human-orientated environments, such as industrial inspection, environment exploration, and autonomous navigation. One common challenge of these tasks is to traverse structured obstacles, such as stairs, steps, and gaps. This requires the robot to be aware of the obstacle by onboard sensors, such as cameras or LiDAR. With visual information, the robot can traverse without compromising its speed, avoid collisions with obstacles, and achieve smooth motions. However, incorporating visual information into the robot locomotion control remains an active area of research.

Model-based methods have been used to cope with such scenarios. The control task can be formulated as an Optimal Control (OC) problem and walking over obstacles can be abstracted as selecting feasible foothold locations directly [9], [10] or encoded as constraints that the end-effectors satisfy [7]. Winkler et al. presented a framework for dynamic quadrupedal locomotion over challenging terrain and used



Fig. 1: ANYmal [1] walking over an obstacle.

a terrain cost map to select the foothold locations [11]. Kalakrishnan et al. proposed to decompose the control into sub-systems and used expert demonstrations to learn the footholds [12]. Instead of selecting footholds separately, Grandia et al. added the end-effector constraints into the optimization objective [7]. While Model Predictive Control (MPC) can produce accurate solutions if the obstacle can be perfectly detected and depicted, it is computationally intense, which enforces extra constraints to the perception module. Furthermore, the exteroceptive sensors are not perfect and have noises, which may be crucial to the modeling of the problem.

On the other hand, learning-based approaches have seen a rapid development in the past years [13]–[19]. Among them Reinforcement Learning (RL) [20] and Imitation Learning (IL) [21] are the two main paradigms. RL encodes the problem as a markov decision process (MDP) and guides the agent with task-specific rewards. Lee et al. trained a policy that can walk on steps by gradually increasing the terrain difficulty during training [22]. Miki et al. incorporated the exteroceptive input explicitly and used a Recurrent Neural Network (RNN) encoder to combine multi-modal information to handle the noisy exteroceptive information [23]. While RL is capable of overcoming various structured terrains, it requires engineering efforts in reward design and usually takes days to train the policy. In contrast, IL learns the policy from expert demonstrations and can train the policy in a short time. It either seeks to replicate the behavior of the expert through learning the observation-action mapping directly, which is called Behavioral Cloning (BC) or learn the demonstrator's reward function for RL purpose, as known as Inverse Reinforcement Learning (IRL). When good experts are available, sample efficiency can be significantly improved compared to RL [24]. However, expert demonstrations for quadrupedal locomotion are often difficult to obtain. Peng et.al proposed to learn quadrupedal locomotion by imitating natural animals based on motion capture data [17]. Carius

All authors are with the Robotic Systems Lab, ETH Zürich, Switzerland.  
Email: chaoni@ethz.ch

et al. proposed to learn locomotion from the MPC expert by MPC-Net [18], which is a variant of IL approaches. Instead of imitating the action directly, it minimizes the control Hamiltonian, which also encodes the underlying constraints of the OC problem, and therefore improves constraint satisfaction practically [18]. The effectiveness of MPC-Net has also been shown in [19], where a robust multi-gait policy is learned. Nevertheless, most imitation approaches for locomotion are limited on flat terrain [17]–[19], where the terrain perception is not available and walking over structured obstacles often leads to failure.

In this work, we propose to traverse structured obstacles with IL via leveraging the knowledge from MPC. MPC can provide accurate solution given the perfect exteroceptive information in simulation, and we benefit from sample efficiency and improved constraint satisfaction by extending MPC-Net. We use a teacher-student learning framework [25] to incorporate the noisy exteroceptive information. To the best of our knowledge, this is the first approach that achieves perceptive IL for overcoming structured obstacles and transfers to the real robot. Our work is built upon the theoretical principle of a Hamiltonian loss for policy search [18] and the application in robust multi-gait locomotion [19] and contributes the following advances:

- We add exteroception to MPC-Net to traverse structured obstacles.
- We show improved convergence by introducing an adaptive data generation for IL.
- Benchmarking experiments confirm that teacher policy trained with MPC-Net leads to better performance compared to BC.
- Simulation results show that Perceptive MPC-Net achieves better performance than MPC expert under noisy environments.
- Sim-to-real transfer shows that Perceptive MPC-Net outperforms blind MPC-Net over a 10 cm step setup.

#### A. Related Work

This section covers a subset of model-based and learning-based approaches on robot locomotion over uneven obstacles and IL in robotics tasks.

##### 1) Locomotion over Uneven Terrain

A large portion of model-based approaches focuses on the selection of foothold location, which is incorporated into the optimization problem. Jenelten et al. defined a grid map centered at the nominal foothold, where each cell is scored based on manually selected features [26]. Prediction of the foothold can also be done by a pretrained convolutional neural network (CNN) [9], [10]. Meduri et al. proposed to encode the foothold selection on uneven terrain as a MDP and trained a stepper via deep RL [27]. Besides predicting foothold locations, there are also works on learning to walk over uneven terrains directly through RL. An end-to-end planner and controller are proposed in [28] where the perception-aware robot learns to walk on difficult terrains through deep RL.

##### 2) Imitation Learning

Learning from expert demonstration has also been widely used in multiple autonomous tasks. Johns proposed a coarse-to-fine process to imitate manipulation from a single demonstration [29]. Qin et al. built an IL platform for manipulation from human videos [30]. Cao et al. investigated the imperfect demonstrations and score the demonstrations by their feasibility and optimality [31]. An adaptive locomotion skill is learned from multiple motion clips via Generative Adversarial Imitation Learning (GAIL) in [32]. Despite the agility it achieves, it is still limited to the area of animation. Chen et al. proposed a “learning by cheating” schedule for IL, where a privileged agent is trained with ground truth observation in the first stage, and in the second stage, the sensorimotor agent, which has no access to ground truth, learns to imitate the privileged agent [25].

## II. PRELIMINARY

In this section, we give a brief background on MPC and the minimum-principle guided policy search approach: MPC-Net [18].

### A. Model Predictive Control

We consider the following OC problem

$$\underset{\mathbf{u}(\cdot)}{\text{minimize}} \quad \Phi(\mathbf{x}(t_f)) + \int_0^{t_f} l(\mathbf{x}(t), \mathbf{u}(t), t) dt, \quad (1)$$

$$\text{subject to} \quad \mathbf{x}(0) = \mathbf{x}_0, \quad (2a)$$

$$\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}, \mathbf{u}, t), \quad (2b)$$

$$\mathbf{g}(\mathbf{x}, \mathbf{u}, t) = \mathbf{0}, \quad (2c)$$

$$\mathbf{h}(\mathbf{x}, \mathbf{u}, t) \geq \mathbf{0}, \quad (2d)$$

where  $\mathbf{x}(t)$  and  $\mathbf{u}(t)$  are the state and input at time  $t$ ,  $t_f$  the time horizon,  $\mathbf{x}_0$  the initial state,  $\Phi(\cdot)$  the final cost and the  $l(\cdot)$  the intermediate cost. The system dynamics is defined by  $\mathbf{f}(\cdot)$  and has vectorized equality constraints  $\mathbf{g}(\cdot)$  and inequality constraints  $\mathbf{h}(\cdot)$ .

To solve the optimization problem, we use a Sequential Linear-Quadratic (SLQ) algorithm [33], which is a variant of the Differential Dynamic Programming (DDP) algorithm. The equality constraints are incorporated via Lagrange multipliers  $\boldsymbol{\nu}$  [33] and inequality constraints are handled by a barrier function  $b(\cdot)$  [34]. The full Lagrangian of the OC problem (1, 2) is given by

$$\begin{aligned} \mathcal{L}(\mathbf{x}, \mathbf{u}, t) &= l(\mathbf{x}, \mathbf{u}, t) + \boldsymbol{\nu}(\mathbf{x}, t)^\top \mathbf{g}(\mathbf{x}, \mathbf{u}, t) \\ &\quad + \sum_i b(h_i(\mathbf{x}, \mathbf{u}, t)). \end{aligned} \quad (3)$$

The solution to the OC problem (1,2) consists of a nominal state-input trajectory  $\{\mathbf{x}_{\text{nom}}(\cdot), \mathbf{u}_{\text{nom}}(\cdot)\}$  and a time-dependent feedback gain  $\mathbf{K}(t)$ , leading to the control policy

$$\pi_{\text{mpc}}(\mathbf{x}, t) = \mathbf{u}_{\text{nom}}(t) + \mathbf{K}(t)(\mathbf{x} - \mathbf{x}_{\text{nom}}(t)). \quad (4)$$

The OC problem (1, 2) has an associate optimal cost-to-go function

$$V(\mathbf{x}, t) = \min_{\substack{\mathbf{u}(\cdot) \\ \text{s.t. (2)}}} \left\{ \Phi(\mathbf{x}(t_f)) + \int_t^{t_f} l(\mathbf{x}(\tau), \mathbf{u}(\tau), \tau) d\tau \right\}, \quad (5)$$

and the control Hamiltonian is given by

$$\mathcal{H}(\mathbf{x}, \mathbf{u}, t) = \mathcal{L}(\mathbf{x}, \mathbf{u}, t) + \partial_{\mathbf{x}} V(\mathbf{x}, t)^{\top} \mathbf{f}(\mathbf{x}, \mathbf{u}, t), \quad (6)$$

which for all  $t$  and  $\mathbf{x}$  satisfies the Hamilton-Jacobi-Bellman (HJB) equation

$$0 = \min_{\mathbf{u}(\cdot)} \{ \mathcal{H}(\mathbf{x}, \mathbf{u}, t) + \partial_t V(\mathbf{x}, t) \}. \quad (7)$$

### B. Perceptive MPC

Perceptive MPC [7] handles obstacles by first extracting a convex segmentation from the elevation map either via LiDAR point cloud or perfect terrain in simulation, and forms state-only constraints

$$h_i^{\text{st}}(\mathbf{x}) = \mathbf{A}_i \cdot \mathbf{p}_{E_i}(\mathbf{x}) + \mathbf{c}_i \geq 0, \quad (8)$$

where  $\mathbf{p}_{E_i} : \mathbb{R}^{24} \rightarrow \mathbb{R}^3$  maps the robot state to the foot position and the matrix  $\mathbf{A}_i$  and  $\mathbf{c}_i$  project the position of the foot  $i$  onto the target segmentation and form a set of half-space constraints. These state-only constraints are then incorporated into (2) and integrated into the objective by barrier functions [34].

### C. MPC-Net

MPC-Net [18] imitates MPC and learns a policy by minimizing the Hamiltonian, which encodes the constraints of the OC problem including obstacle-related constraints (8). The policy  $\pi(\mathbf{x}, t; \theta)$  is parametrized by a mixture-of-experts network (MEN), each expert specializing for different modes, where each mode represents a contact configuration. Accordingly, the policy can be written as follows

$$\pi(\mathbf{x}, t; \theta) = \sum_{i=1}^E p_i(\mathbf{x}, t; \theta) \pi_i(\mathbf{x}, t; \theta), \quad (9)$$

where  $\mathbf{p} = (p_1, \dots, p_E)$  is the weight for each expert and is computed by a gating network.  $E$  is the number of experts. To improve expert specialization, the cross-entropy is used that allows the incorporation of domain knowledge [19]:

$$CE(\tilde{\mathbf{p}}, \mathbf{p}) = - \sum_{i=1}^E \tilde{p}_i(t) \log(p_i(\mathbf{x}, t; \theta)), \quad (10)$$

where  $\tilde{p}_i(t)$  is the probability to observe mode  $i$  at time  $t$  and is defined as  $\tilde{p}_i(t) = \mathbb{1}_{\{m(t)=i\}}$ , where  $m(t)$  is the commanded mode schedule. This leads to the following loss function for MPC-Net

$$\text{loss} = \mathcal{H}(\mathbf{x}, \pi, t) + CE(\tilde{\mathbf{p}}, \mathbf{p}), \quad (11)$$

where we define  $\pi := \pi(\mathbf{x}, t; \theta)$ .

One common problem in IL is the violation of i.i.d. assumption when the learner's actions effects its future observations and states [35]. MPC-Net uses a behavioral policy to address the mismatch:

$$\pi_b(\mathbf{x}, t; \theta) = \alpha \pi_{\text{mpc}}(\mathbf{x}, t) + (1 - \alpha) \pi(\mathbf{x}, t; \theta), \quad (12)$$

where  $\alpha$  linearly decreasing from one to zero in the training. The behavioral policy executes the rollout in the data generation and the collected data is later used for training.

Reske et al. proposed to replace the absolute state with the so-called relative state to achieve better tracking, and introduced a generalized time to guide the gait based on the commanded mode schedule [19].

## III. APPROACH

Perceptive MPC-Net empowers the autonomous agent with visual ability to traverse structured terrain. Apart from the proprioceptive state estimation, the agent receives exteroceptive information and makes steps based on the observation.

We utilize a two-stage “learning by cheating” framework [25]. First, a privileged teacher is trained via imitating the MPC expert. Then, we train a student to imitate the behavior of the privileged teacher. The teacher is confined to simulation, while the student can be deployed on the hardware. Similar to [25], we provide the teacher with expert demonstration. We use Perceptive MPC [7] as the MPC expert. To avoid confusion of the usage of teacher in two stages in the rest of the paper, we refer to MPC as **expert**, privileged agent as **teacher** and the deployable agent as **student**.

In the following subsections, we explain each training component in detail.

### A. Adaptive Data Generation

While the linear adaption for the mixture ratio has been proven practically effective in previous works [18], [19], the mismatch between the expert policy and learner's policy still exists given a more complex observed environment. To address this problem, we propose a two-step data generation approach, where in the first step, we linearly decrease the mixture ratio  $\alpha$  and in the second step, we roll out the trajectory starting with  $\alpha = 0$  and actively adapt  $\alpha$  based on how well the learner's policy minimizes the Hamiltonian. As hinted by (7), we actively adapt the ratio at every contact configuration transition by

$$\alpha(t_i) = 1 - \exp \left( -\frac{\beta}{\Delta t} \int_{t_{i-1}}^{t_i} (\mathcal{H} + \partial_t V) dt \right), \quad (13)$$

where  $\beta$  is a constant,  $t_i$  is the  $i$ -th contact configuration transition time extracted from the mode schedule.  $\Delta t = t_i - t_{i-1}$ ,  $\mathcal{H} = \mathcal{H}(\mathbf{x}, \pi, t)$  and  $\partial_t V = \partial_t V(\mathbf{x}, t)$ . We update mixture ratio at each transition to avoid rapid change and have a better estimate of the quality of the learner's policy.

### B. Privileged Teacher

The privileged teacher has access to the ground truth elevation map and privilege information regarding its contact status. The teacher observes a list of scan points around the foothold as its exteroceptive observation.

#### 1) Architecture

The exteroceptive observation and privilege information are processed with a Multilayer Perceptron (MLP) and form the latent representation  $\bar{z}$ . The downstream MEN is similar to [19] and we extend it such that it also receives the latent representation of the environment. The complete architecture is shown in Fig. (2).

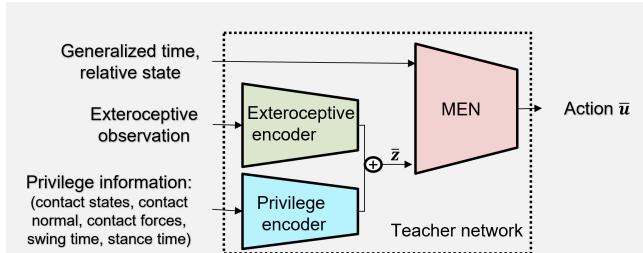


Fig. 2: Teacher network. Inputs include 1) generalized time and relative state; 2) exteroceptive observation; 3) privilege information (contact states, contact normals, contact forces, as well as swing and stance time). Privilege information are obtained from the RaiSim [36] simulator. The encoded exteroceptive observation and privilege information form the latent  $\bar{z}$ , which together with generalized time and relative state [19], is the input to the mixture-of-experts network (MEN) policy.

### 2) Objective

The teacher training is supervised in the MPC-Net fashion and the loss function is defined by (11), where the components are provided by the MPC solver.

### 3) Training

The data generation and policy search run asynchronously in a multi-thread scheme as in [19]. The data is generated by a behavioral policy (12), where the mixture ratio  $\alpha$  is determined in a two-step fashion as described in Sec. III-A.

### C. Student

After the teacher policy is trained, we copy the MEN and the exteroceptive encoder from the teacher's network and freeze them in the course of student training. A sequence of proprioceptive observation is received. The key assumption is that a sequence of proprioceptive observation helps to reconstruct the latent representation of the environment and the contact status of the agent.

#### 1) Architecture

The architecture of the student network is shown in Fig. (3) leveraged from [23]. We use a Gated Recurrent Network (GRU) [37] to encode the exteroceptive observation as well as proprioceptive information. The noise of the exteroceptive observation can be eased through the averaging effect of history and GRU, and the contact status is related to the phase of locomotion. The GRU outputs a latent representation  $z$  supervised by the teacher's latent feature  $\bar{z}$ . This representation  $z$  is further fed into the MEN with generalized time and relative state, and final action is generated.

#### 2) Objective

The student is trained with BC and the loss is defined as

$$\text{loss} = \|\mathbf{u} - \bar{\mathbf{u}}\|_{\mathbf{R}} + \|z - \bar{z}\|_2 + \lambda \|z - \bar{z}\|_1, \quad (14)$$

where  $\bar{\mathbf{u}}$  and  $\bar{z}$  are obtained by evaluating the teacher policy with privileged information given by the RaiSim [36] simulator and  $\mathbf{R}$  is the cost matrix for balancing different input dimensions. We use a combination of 2-norm and 1-norm loss with a regularizer coefficient  $\lambda$  for the reconstruction of the latent representation.

### 3) Training

During the training, we use a similar two-step data generation approach as in the teacher training with a shortened stage one. However, in the second step, we only use student's

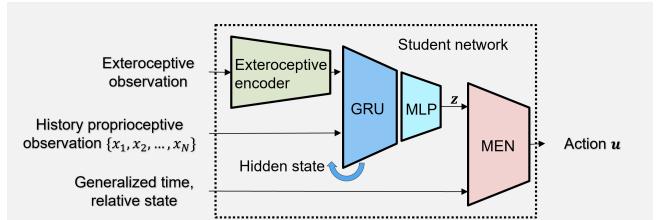


Fig. 3: Student network. The exteroceptive encoder and the multi-expert policy are copied from the teacher network and are frozen in the student training. The exteroceptive observation and a sequence of proprioceptive observation are used to reconstruct the robot's consensus about the environment with a Gated Recurrent Network (GRU) [37].

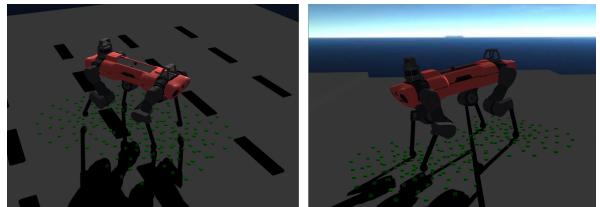


Fig. 4: Simulation terrains: gaps and steps. The width and depth of the gap as well as the shape and height of the step are randomized during the course of training.

policy and keep  $\alpha = 0$ . With frozen MEN and exteroceptive encoder, the student policy is able to roll out successfully within a few iterations.

## IV. IMPLEMENTATION

This section introduces the setup of our quadrupedal robot and the structured terrain it traverses. We also explain some training details and how we deploy on a real robot.

### A. ANYmal Control

We verified our approach on the quadrupedal robot ANYmal [1]. The kinodynamic model used by the MPC expert has a 24-dimensional state (base pose, base twist, joint angles) and 24-dimensional inputs (contact forces, joint velocities). The intermediate cost and final cost for the OC problem (1, 2) are formed as follows

$$\Phi(\mathbf{x}) = (\mathbf{x} - \mathbf{x}_d(t_f))^T \mathbf{Q}_f (\mathbf{x} - \mathbf{x}_d(t_f)), \quad (15)$$

$$l(\mathbf{x}, \mathbf{u}, t) = (\mathbf{x} - \mathbf{x}_d(t))^T \mathbf{Q} (\mathbf{x} - \mathbf{x}_d(t)) + \mathbf{u}^T \mathbf{R} \mathbf{u}, \quad (16)$$

where  $\mathbf{x}_d(\cdot)$  is the desired state given by a user-defined reference trajectory.  $\mathbf{Q}_f$ ,  $\mathbf{Q}$  and  $\mathbf{R}$  are cost matrices.

We use trotting gait in this work. The robot is controlled by torques, which are generated by inverse dynamics and PD control:

$$\boldsymbol{\tau} = \boldsymbol{\tau}_{id} + \mathbf{K}_d \cdot (\dot{\mathbf{q}}_{j,d} - \dot{\mathbf{q}}_{j,m}), \quad (17)$$

where  $\boldsymbol{\tau}_{id}$  is the inverse dynamics torque and  $\dot{\mathbf{q}}_{j,d}$ ,  $\dot{\mathbf{q}}_{j,m}$  are desired and measured joint velocities, respectively.

### B. ANYmal Perception

We evaluate our approach mainly on two different terrains: gaps and steps, as is shown in Fig. (4). A curriculum factor is set for each terrain and represents the level of traversability. During training, we randomize the parameter of the terrains and the initial position of the robot to increase the variability of exteroceptive observations.

The robot is equipped with two RS-Bpearl LiDARs and an elevation map is constructed from the point clouds. A list

TABLE I: Scan configuration for each leg

Radius [m]	0.08	0.16	0.26	0.36	0.48
Number of Points	6	8	10	12	16

TABLE II: Hyperparameters

time step $\Delta t$	0.0025 s	rollout length $T$	8 s
number of threads $n_t$	5	adaption value $\beta$	0.1
number of jobs $n_j$	10	batch size $B_t$	256
horizon $n_s$	10	batch size $B_s$	256
data reuse $n_b$	2	learning rate $\eta$	1e-3
data decimation $d_s$	4	step one $i_1$	25k
maximum capacity $N$	5000	step two $i_2$	15k
number of experts $E$	3	regularizer $\lambda$	0.2

of scan points around the foothold is observed and the scan points are circled uniformly around the foothold projection onto the terrain in different radii. For each point, we take its vertical distance to the end-effector as scan input. The scan configuration is shown in Table. I [23].

### C. Training Details

In this subsection, we give a detailed training schedule and summarize the parameters we use for training.

#### 1) Hyper Parameters

Each trajectory lasts for a duration  $T$  and is forwarded every time step  $\Delta t$ . Note that we use a twist command to roll out the trajectory, instead of setting a random desired state as [18], [19]. This allows us a longer horizon and a complete cross over the obstacle. A rollout of trajectory is considered as failure, if the pitch or roll angle exceeds  $30^\circ$ , or if the height deviates more than 20 cm from the default value [18], [19]. When evaluating the policy, survival time defined as the earliest time of failure is used.

In the teacher training, data is generated on CPU by  $n_t$  threads that work on  $n_j$  jobs per run. For each job, a trajectory is generated with a random initialization at  $x_0$  and twist command. Each trajectory is down-sampled by a factor  $d_s$  before being pushed into a circular buffer. We use batch size  $b_t$  for the teacher training. Moreover, the buffer has a maximum capacity  $N$  in the number of trajectories. As mentioned in Sec. III-B.3, we train the step one for  $i_1$  gradient steps, and step two for  $i_2$  steps.

In the student training, each data generation run gives a batch of  $B_s$  trajectories. To train the GRU, we forward the trajectory for  $n_s$  steps and sum up the loss, followed by one gradient step. Each batch is discarded after being forwarded  $n_b$  times.

We use Adam optimizer [38] for both teacher and student with a learning rate  $\eta$ . The mentioned hyperparameters are listed in Table II.

#### 2) Dimensions

The exteroceptive encoder and the privilege encoder are both a MLP with two hidden layers, with each layer having 64 neurons. Each expert of MEN, as well as the gating network of MEN, is a two-layer MLP with 128 neurons. A softmax [39] function is applied to the top of the gating network. All activation functions as set to LeakyReLU [40].

#### D. Deployment

In the deployment, user controls the robot with a twist command. The robot receives the scan points from its LiDAR sensor and proprioceptive observation from the state

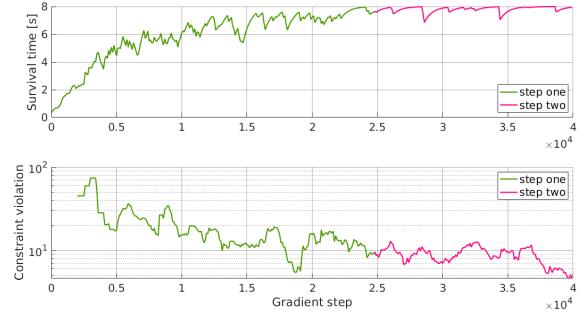


Fig. 5: The graph illustrates the improvement of survival time and constraint satisfaction with an adaptive data generation. The first 25k is trained with linearly decreasing mixture ratio, and the rest 15k is trained with adaptive ratio as defined in (13).

TABLE III: Comparison between our two step adaptive data generation and previous work [19]. We report the average survival time and the standard deviation by running each cases for 50 times. We evaluate this experiment on a 8 cm wide gap and 14 cm high step with a forward twist command 0.2 m/s.

	Our Approach	Linear Adaption
Gap	<b>25.59±6.59</b>	21.18 ± 6.31
Step	<b>26.65±6.99</b>	23.90 ± 7.59

estimator. The input  $u$  is inferred at 400 Hz from the policy, which is evaluated via ONNX Runtime [41]. In contrast to previous works [18], [19], where a whole-body controller (WBC) is used to track the input and computes the torques for the robot, we turn to inverse dynamics and PD control as used in the simulation. We find that such a change facilitates the sim-to-real transfer.

## V. RESULTS

We evaluate our approach with thorough comparison results in simulation and hardware tests. For simulation results reported, our runs are executed on a single thread of the machine: Intel® Xeon(R) CPU E3-1280 v5 @ 3.70GHz x8.

### A. Adaptive Data Generation

Using an adaptive data generation improves the convergence of the training and leads to a higher survival time and better constraint satisfaction as Fig. (5) shows. We also compared against policy that is trained with only step one [18], [19] with same network structure and gradient steps, and our policy survives longer with a similar standard deviation as shown in Table III.

### B. Teacher Benchmarking

We benchmark our teacher policy with BC and MPC expert. In the case of BC, we replace the loss in (11) with

$$\text{loss} = \|\pi_{\text{mpc}} - \pi(t, \mathbf{x}; \boldsymbol{\theta})\|_{\mathbf{R}} + CE(\tilde{p}, p), \quad (18)$$

where we use the cost matrix  $\mathbf{R}$  to normalize different dimensions. Perceptive MPC-Net achieves better performance than BC policy in both obstacles. The detailed results are shown in Table IV.

### C. Comparison with MPC expert

We compared the performance of our student policy with MPC expert under noisy environments. We setup the step experiments on Gazebo and created a noisy elevation map for both MPC experts and our student policy. The performance is

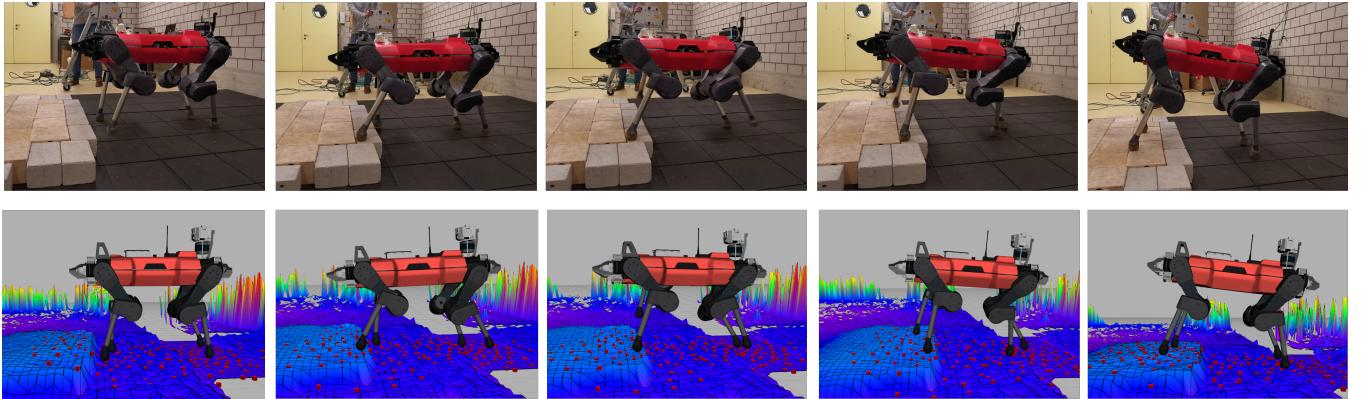


Fig. 6: Stepping on a 10 cm step. We create the obstacle with bricks and wooden boxes. The red points are selected scan points based on the scan configuration in Table I. Top: Hardware experiments, Bottom: A replay visualization. From left to right: 1). The robot observes the step and prepares for the lift of its left front (LF) leg; 2). Finishes LF step; 3). Prepares right front (RF) leg; 4). Finishes RF leg; 5). Stabilizes and prepares to walk forward. The full transverse of the obstacles is in the supplementary video.

TABLE IV: Survival time comparison against MPC expert and teacher policy trained with BC. The evaluation is on a 8 cm wide gap and {10, 12, 14} cm high step terrain respectively. We collected 50 episodes, each with maximum 30 seconds.

	MPC	Perceptive MPC-Net		BC
Step	10 cm	<b>30.00±0.00</b>	<b>30.00±0.00</b>	29.06 ± 4.61
	12 cm	<b>30.00±0.00</b>	29.10 ± 4.39	25.05 ± 9.61
	14 cm	<b>30.00±0.00</b>	26.65 ± 6.99	23.25 ± 9.83
Gap	8 cm	<b>27.41±4.56</b>	26.21 ± 5.98	24.09 ± 6.67

TABLE V: Success rate comparison against MPC expert under noisy environment. For each case, we forward the robot with same twist command and tried 50 attempts. Note that our policy is only trained with a maximum height 14 cm.

$\sigma$	10 cm		12 cm		14 cm		16 cm	
	MPC	Ours	MPC	Ours	MPC	Ours	MPC	Ours
0.000	<b>1.00</b>	<b>1.00</b>	<b>1.00</b>	0.98	<b>0.98</b>	0.90	<b>1.00</b>	0.08
0.030	<b>1.00</b>	<b>1.00</b>	<b>1.00</b>	0.96	0.30	<b>0.86</b>	<b>0.32</b>	0.06
0.035	0.48	<b>1.00</b>	0.32	<b>0.96</b>	0.16	<b>0.82</b>	0.04	0.04
0.040	0.14	<b>1.00</b>	0.18	<b>0.90</b>	0.08	<b>0.80</b>	0.00	<b>0.04</b>
0.100	0.00	<b>1.00</b>	0.00	<b>0.80</b>	0.00	<b>0.76</b>	0.00	0.00

evaluated by the success rate, which is defined as the number of successful traversal over the step divided by total attempts. The noise level of the elevation map is controlled by the standard deviation  $\sigma$ , and the noisy elevation map is defined as:

$$\tilde{p}_{x,y} = p_{x,y} + \mathcal{N}(0, \sigma), \quad (19)$$

where  $p_{x,y}$  is the true elevation value and  $\tilde{p}_{x,y}$  is the noisy value at the coordinate  $(x, y)$  and  $\mathcal{N}(0, \sigma)$  is a zero-mean normal distribution. When the noise level or step height increases, MPC fails to find the solution because of incorrect segmentation. On the other hand, our student policy outperforms under the noisy elevation map. The detailed results are in Table. V.

#### D. Sim-to-real Transfer

In this subsection, we validate the student policy on the quadruped ANYmal and show the benefit of using two-stage learning with practical evidence. Since gap terrain is not available in the lab environment and may cause danger to the robot, we limit the hardware experiment to the step obstacles. As shown in Fig. (6), the robot detects the obstacle through scan points and lifts the legs high enough to clear the obstacle. It is able to stabilize after stepping on the obstacle and sometimes even compensate for the missed step.

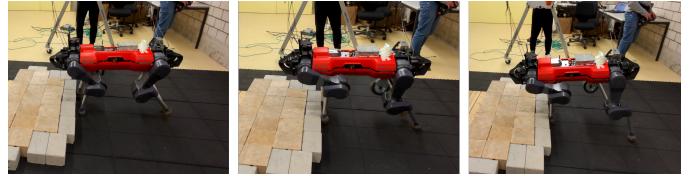


Fig. 7: Blind MPC-Net fails to step on high obstacles. From left to right, the robot is not aware of the existence of the obstacle, and therefore keeps the same trotting height. It hit the steps and fails to move forward.

#### 1) Comparison with Blind MPC-Net

To show the visual ability of our policy, we compared to the previous work [19], where no perception input is given. We refer to it as blind MPC-Net. As shown in Fig. (7), since the robot has no perception inputs, it was unable to walk over high steps and failed the task.

#### 2) Necessity of the Privileged Teacher

In the end, we investigated the necessity of using a two-stage learning approach. We trained a teacher without using privilege information and only used MEN to generate the input. However, it was unable to step over obstacles and much sensitive to noises.

## VI. CONCLUSION

In this work, we added perception to MPC-Net to walk over structured obstacles by learning from MPC experts. A teacher-student framework was used to handle the noisy exteroceptive information and performed better than a single-stage method. We compared the performance under noisy environment against MPC expert and showed our policy is more robust to the noisy elevation map. Moreover, we proposed an adaptive data generation for MPC-Net and it leads to better domain transfer. The simulation results showed the benefit of using MPC-Net comparing against BC. Finally, we validated the viability of the approach on hardware by demonstrating a successful traverse over the structured obstacle.

This work proposed to learn to walk over obstacles by leveraging knowledge from MPC and opens the door to a wider range of traversable terrains. Future work includes training a single policy for different types of obstacles and in an uncertain dynamical environment.

## REFERENCES

- [1] 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 *2016 IEEE/RSJ international conference on intelligent robots and systems (IROS)*. IEEE, 2016, pp. 38–44.
- [2] 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 *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2018, pp. 2245–2252.
- [3] J. Ding, Y. Wang, M. Yang, and X. Xiao, “Walking stabilization control for humanoid robots on unknown slope based on walking sequences adjustment,” *Journal of Intelligent & Robotic Systems*, vol. 90, no. 3, pp. 323–338, 2018.
- [4] M. Bando, M. Murooka, S. Nozawa, K. Okada, and M. Inaba, “Walking on a steep slope using a rope by a life-size humanoid robot,” in *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2018, pp. 705–712.
- [5] J. Siekmann, K. Green, J. Warila, A. Fern, and J. Hurst, “Blind bipedal stair traversal via sim-to-real reinforcement learning,” *arXiv preprint arXiv:2105.08328*, 2021.
- [6] E. Moore and M. Buehler, “Stable stair climbing in a simple hexapod robot,” MCGILL RESEARCH CENTRE FOR INTELLIGENT MACHINES MONTREAL (QUEBEC), Tech. Rep., 2001.
- [7] R. Grandia, A. J. Taylor, A. D. Ames, and M. Hutter, “Multi-layered safety for legged robots via control barrier functions and model predictive control,” in *International Conference on Robotics and Automation (ICRA 2021)*, 2021, p. 3969.
- [8] Q. Nguyen, A. Hereid, J. W. Grizzle, A. D. Ames, and K. Sreenath, “3d dynamic walking on stepping stones with control barrier functions,” in *2016 IEEE 55th Conference on Decision and Control (CDC)*. IEEE, 2016, pp. 827–834.
- [9] O. Villarreal, V. Barasuol, P. M. Wensing, D. G. Caldwell, and C. Semini, “Mpc-based controller with terrain insight for dynamic legged locomotion,” in *2020 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2020, pp. 2436–2442.
- [10] 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,” *IEEE Robotics and Automation Letters*, vol. 4, no. 2, pp. 2140–2147, 2019.
- [11] A. W. Winkler, C. Mastalli, I. Havoutis, M. Focchi, D. G. Caldwell, and C. Semini, “Planning and execution of dynamic whole-body locomotion for a hydraulic quadruped on challenging terrain,” in *2015 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2015, pp. 5148–5154.
- [12] M. Kalakrishnan, J. Buchli, P. Pastor, M. Mistry, and S. Schaal, “Learning, planning, and control for quadruped locomotion over challenging terrain,” *The International Journal of Robotics Research*, vol. 30, no. 2, pp. 236–258, 2011.
- [13] 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*, vol. 4, no. 26, 2019.
- [14] J. Tan, T. Zhang, E. Coumans, A. Iscen, Y. Bai, D. Hafner, S. Bohem, and V. Vanhoucke, “Sim-to-real: Learning agile locomotion for quadruped robots.” in *Robotics: Science and Systems*, 2018.
- [15] S. Ha, P. Xu, Z. Tan, S. Levine, and J. Tan, “Learning to walk in the real world with minimal human effort,” in *Conference on Robot Learning*, 2020.
- [16] T. Haarnoja, S. Ha, A. Zhou, J. Tan, G. Tucker, and S. Levine, “Learning to walk via deep reinforcement learning.” in *Robotics: Science and Systems*, 2019.
- [17] 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 *Robotics: Science and Systems*, 07 2020.
- [18] J. Carius, F. Farshidian, and M. Hutter, “Mpc-net: A first principles guided policy search,” *IEEE Robotics and Automation Letters*, vol. 5, no. 2, pp. 2897–2904, 2020.
- [19] A. Reske, J. Carius, Y. Ma, F. Farshidian, and M. Hutter, “Imitation learning from mpc for quadrupedal multi-gait control,” in *2021 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2021.
- [20] R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*. MIT press, 2018.
- [21] T. Osa, J. Pajarinen, G. Neumann, J. Bagnell, P. Abbeel, and J. Peters, “An algorithmic perspective on imitation learning,” *Foundations and Trends in Robotics*, vol. 7, no. 1-2, pp. 1–179, 2018.
- [22] J. Lee, J. Hwangbo, L. Wellhausen, V. Koltun, and M. Hutter, “Learning quadrupedal locomotion over challenging terrain,” *Science robotics*, vol. 5, no. 47, 2020.
- [23] T. Miki, J. Lee, J. Hwanbo, L. Wellhausen, V. Koltun, and M. Hutter, “Wild anymal: Robust zero-shot perceptive locomotion,” *Under review*, 2021.
- [24] W. Sun, A. Venkatraman, G. J. Gordon, B. Boots, and J. A. Bagnell, “Deeply aggregated: Differentiable imitation learning for sequential prediction,” in *International Conference on Machine Learning*. PMLR, 2017, pp. 3309–3318.
- [25] D. Chen, B. Zhou, V. Koltun, and P. Krähenbühl, “Learning by cheating,” in *Conference on Robot Learning*. PMLR, 2020, pp. 66–75.
- [26] F. Jenelten, T. Miki, A. E. Vijayan, M. Bjelonic, and M. Hutter, “Perceptive locomotion in rough terrain—online foothold optimization,” *IEEE Robotics and Automation Letters*, vol. 5, no. 4, pp. 5370–5376, 2020.
- [27] A. Meduri, M. Khadiv, and L. Righetti, “Deepq stepper: A framework for reactive dynamic walking on uneven terrain,” *arXiv preprint arXiv:2010.14834*, 2020.
- [28] 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*, vol. 5, no. 2, pp. 3699–3706, 2020.
- [29] E. Johns, “Coarse-to-fine imitation learning: Robot manipulation from a single demonstration,” *arXiv preprint arXiv:2105.06411*, 2021.
- [30] Y. Qin, Y.-H. Wu, S. Liu, H. Jiang, R. Yang, Y. Fu, and X. Wang, “Dexmv: Imitation learning for dexterous manipulation from human videos,” *arXiv preprint arXiv:2108.05877*, 2021.
- [31] Z. Cao and D. Sadigh, “Learning from imperfect demonstrations from agents with varying dynamics,” *IEEE Robotics and Automation Letters*, vol. 6, no. 3, pp. 5231–5238, 2021.
- [32] X. B. Peng, Z. Ma, P. Abbeel, S. Levine, and A. Kanazawa, “Amp: Adversarial motion priors for stylized physics-based character control,” *arXiv preprint arXiv:2104.02180*, 2021.
- [33] F. Farshidian, M. Neunert, A. W. Winkler, G. Rey, and J. Buchli, “An efficient optimal planning and control framework for quadrupedal locomotion,” in *2017 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2017, pp. 93–100.
- [34] R. Grandia, F. Farshidian, R. Ranftl, and M. Hutter, “Feedback mpc for torque-controlled legged robots,” in *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2019, pp. 4730–4737.
- [35] 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*, vol. 15. PMLR, 11–13 Apr 2011, pp. 627–635.
- [36] J. Hwangbo, J. Lee, and M. Hutter, “Per-contact iteration method for solving contact dynamics,” *IEEE Robotics and Automation Letters*, vol. 3, no. 2, pp. 895–902, 2018. [Online]. Available: [www.raisim.com](http://www.raisim.com)
- [37] K. Cho, B. Merriënboer, C. Gulcehre, F. Bougares, H. Schwenk, and Y. Bengio, “Learning phrase representations using rnn encoder-decoder for statistical machine translation,” in *EMNLP*, 2014.
- [38] D. P. Kingma and J. L. Ba, “Adam: A Method for Stochastic Optimization,” in *Proceedings of the 3rd International Conference on Learning Representations (ICLR)*, 2015.
- [39] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. MIT Press, 2016.
- [40] A. L. Maas, A. Y. Hannun, A. Y. Ng, *et al.*, “Rectifier nonlinearities improve neural network acoustic models,” in *Proc. icml*, vol. 30, no. 1. Citeseer, 2013, p. 3.
- [41] “ONNX Runtime: Cross-platform, High Performance ML Inferencing and Training Accelerator.” [Online]. Available: <https://github.com/microsoft/onnxruntime>