Online Path Planning by using Learned Latent Dynamics

André Brandenburger*, Diego Rodriguez* and Sven Behnke

***Abstract*— Efficient and collision-free navigation is an essential requirement for deploying robots in quotidian scenarios. In the robotics community, Reinforcement Learning (RL) approaches have increasingly gained popularity and have demonstrated their applicability on control tasks based on visual observations. In this paper, we propose a novel deep RL approach to address the mapless navigation problem, in which the actions are taken online based on the knowledge encoded in learned models. Planning happens by generating open-loop trajectories in a learned latent space that captures the dynamics of the environment. Our planner considers visual (RGB images) and non-visual observations (e.g., attitude estimations). This confers the agent upon awareness not only of the scenario, but also of its own state. In addition, we incorporate a termination likelihood predictor model as an auxiliary loss function of the control policy, which enables the agent to anticipate terminal states of success and failure. In this manner, the sample efficiency of the approach for episodic tasks is increased. Our model is evaluated on the NimbRo-OP2X humanoid robot that navigates in scenes avoiding collisions efficiently in simulation and with the real hardware.**

I. Introduction

Mobile robot navigation typically requires a robot to traverse a series of static and dynamic obstacles in the environment to reach desired target poses, e.g., by walking with pedestrians on sidewalks. Traditional methods tackle this problem by processing raw sensor information (e.g., RGB images or laser scans) in order to construct local maps for path planners [1–3]. Traditional approaches, however, lose expressivity with the increment of uncertainty and complexity of the environments mainly because of computational limitations associated with high-dimensional systems and real-time constraints. In the last decade, the rapid advances of learning methods have paved the path for an increasing development of robot learning approaches, which are a promising alternative to solve these issues by leveraging data [4–8].

In this paper, we address the problem of mapless navigation, in which the robot needs to reach a known relative target pose without constructing a map of the environment. The target pose is assumed to be given by higher level modules (e.g., object detection, semantic segmentation or Wi-Fi signal localization). Several DRL approaches have been proposed to solve this problem based on 3D data (e.g., laser scans) [9–11]. In our approach, however, the environment is perceived by RGB-only images which, in contrast to depth data, render a harder problem for planning, since no direct measurements to object distances are provided. Our learned path planner considers additionally non-visual observations such as IMU measurements. In this manner, the planner can act upon large instabilities of the robot posture in order to avoid falls.

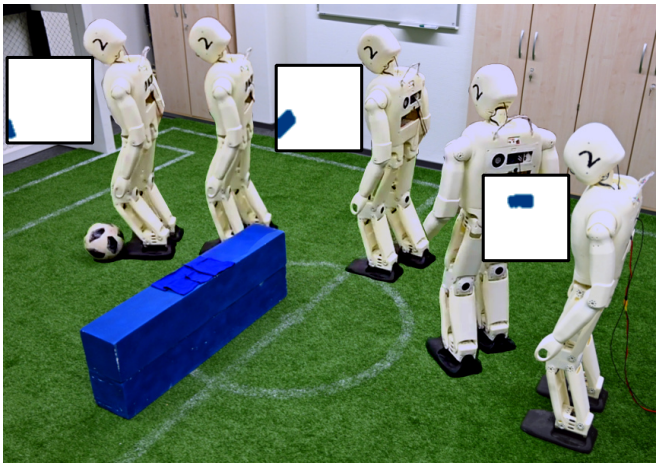

Fig. 1. The NimbRo-OP2X robot navigates to reach the goal (ball) while avoiding obstacles. The actions are inferred online by a control policy (at 10 Hz), given a segmented image (surrounded by black squares) and non-visual sensor data. For clarity, only three segmented images are shown.

Our approach is able to plan collision-free paths without local maps by learning a latent world model and by *imagining* possible future outcomes based on learned models. These open-loop (imagined) trajectories address the problem of lack of memory of Markov Decision Processes which are typically used to formulate RL tasks. Re-planning happens implicitly with each new inference step. This allows our approach to handle uncertainty present in scenarios with dynamic obstacles.

In order to handle episodic tasks, as the one discussed here, we incorporate a predictor model that infers a termination likelihood and provides this information to the control policy as an auxiliary loss. We explicitly differentiate between successful and failed terminal states; the former encourages the agent to finish the episode collecting a high reward, while the latter contributes to the sample efficiency and training time reduction by neglecting experiences collected during failed terminal states, e.g., when the robot is lying on the floor after falling.

We evaluate our approach on a real autonomous humanoid robot (Fig. 1). To handle the sim-to-real transfer, segmented images are employed for training the learned models, which in conjunction with noise injection and system identification allows to transfer the control policy to the real robot without retraining.

* Authors with equal contribution. All authors are with the Autonomous Intelligent Systems (AIS) Group, Computer Science Institute VI, University of Bonn, Germany rodriguez@ais.uni-bonn.de

In summary, the main contributions of this paper are:

- the formulation of a novel approach for online path planning that considers visual (RGB images) and non-visual observations to learn a control policy and an environment dynamics model;
- the introduction of a termination likelihood predictor to handle multiple terminal states specially relevant for episodic tasks;
- and the demonstration on a real humanoid robot of the learned policy for mapless navigation.

II. Related Work

Previous research on *visual control* problems, in which an agent takes actions based on image observations, has led to multiple analytical open-loop approaches [12–14]. To address the typical shortcomings of analytical solutions, especially related to the curse of dimensionality, novel learning-based methods have called the attention of the community due to their generalization capabilities to uncertainty and due to the inference time that enables their usage in real-world tasks [15–17].

Particularly, Reinforcement Learning (RL) approaches have gained increased popularity in robotics, where policies are learned by interaction with the environment. Popular model-free RL methods, such as DQN [18], aim to construct a state-action value function (Q-value) that quantifies the quality of state-action pairs to maximize an accumulative reward in the long term [4–6]. Other model-free RL approaches, called policy gradient methods, construct a policy by optimizing a cost function directly, such as D4PG [7] and PPO [19]. Although these RL methods have been successfully implemented in robotics applications [11, 20] including visual control tasks [15, 21], the training with raw images requires a large amount of data — due to the absence of a learned dynamics model, which could encode the state evolution effectively.

While model-free RL approaches are often straightforward to employ, model-based methods can be more sample-efficient by exploiting a learned dynamics model. One of the first attempts to learn a control policy in conjunction with a dynamics model is Dyna-Q [22]. Recent approaches such as [8], [23] and [24] are able to process raw image observations directly by using self-supervised representation techniques, i.e., autoencoders. Inspired by these works, in this paper, we present a novel model-based RL approach for mapless navigation.

Mapless navigation using RL have been previously addressed [9–11]. Khan *et al.* [11] proposed a two stage architecture consisting of local planners defined by value iteration networks and differentiable memory networks that provide past information. Zhelo *et al.* [9] do not define any memory component but they encourage curiosity-based exploration formulated in a secondary reward function, and consequently the agent is able to navigate in long corridors and dead corners. None of these approaches, however, are able to handle dynamic obstacles and require depth data as input. Moreover, these approaches were evaluated in known scenarios only, thus their generalization capabilities are questionable.

Few RL approaches for robot navigation based on RGB images have been demonstrated in real robots [15, 21]. Xie *et al.* [15] propose a depth prediction network based on monocular RGB images that infers a depth field and a Q-value function for controlling a mobile robot. Lobos-Tsunekawa *et al.* [21] investigate visual navigation on a bipedal platform and learned a control policy by using DDPG. None of these approaches, however, incorporate latent dynamics models and terminal states for episodic tasks are not explicitly handled.

III. Background

As common in RL, we model the environment as a Markov Decision Process (MDP) described by a tuple (S, A, P, R, γ) of environment states S, action space A, state transition probabilities $P : S \times A \times S \to [0, 1]$, reward function $R : S \times A \to \mathbb{R}$, and discounted factor $\gamma \in [0, 1]$. The goal of the agent is to take actions $a_t \in A$ that maximize the collected reward. Often, the agent only has access to partial observations $o_t \in O$ of the environment, which are provided according to state observation probabilities $\Omega : S \times O \to [0, 1]$. This results in a Partially Observable Markov Decision Process (POMDP) defined by $(S, A, P, R, \gamma, O, \Omega)$.

In domains where the observations are defined as images, policies are often expensive to train due to the high dimensionality of the observation space O. Thus, representation techniques such as autoencoders ($\mathcal{E} : O \to W$) are frequently incorporated to reduce the dimensionality of the image input and to define a prior latent state W of the environment model [8, 23, 24]. The latent state dynamics $\mathcal{D} : W \times A \to W$ can be learned effectively to resemble the unknown true state transition P of the environment. Both, the autoencoder and the latent state dynamics can be combined to form a non-linear Kalman Filter, where the state prediction $\tilde{w}$ is given by $\mathcal{D}$, while the filtering is done by the encoder $\mathcal{E}$ [23].

Hafner *et al.* [24] recently proposed a model-based RL approach that builds a latent space W and dynamics model $\mathcal{D}$ which are ultimately employed in an open-loop fashion to plan latent trajectories $\tilde{w}_{t_0} \dots \tilde{w}_{t_N}$. For each of the latent states w_{t_i}, a state value is calculated by use of the Bellman return:

$$v_{t_i} = \sum_{t=t_i}^{t_N} \gamma^{(t-t_i)} \tilde{r}_t \,, \tag{1}$$

given predicted rewards $\tilde{r}_t$ inferred by a predictor $\mathcal{R} : W \to \mathbb{R}$. Additionally, a value predictor model $\mathcal{V} : W \to \mathbb{R}$ is incorporated to optimize the Bellman consistency. The predicted rewards $\tilde{r}_t$, values $\tilde{v}_t$ and actions a_t are modeled stochastically. More precisely, the means of Gaussian distributions are dictated by the prediction models $\mathcal{R}, \mathcal{V}, \pi$. The standard deviations of the action distributions are also inferred by the actor π, while a unit standard deviation is chosen for the other predictors. In

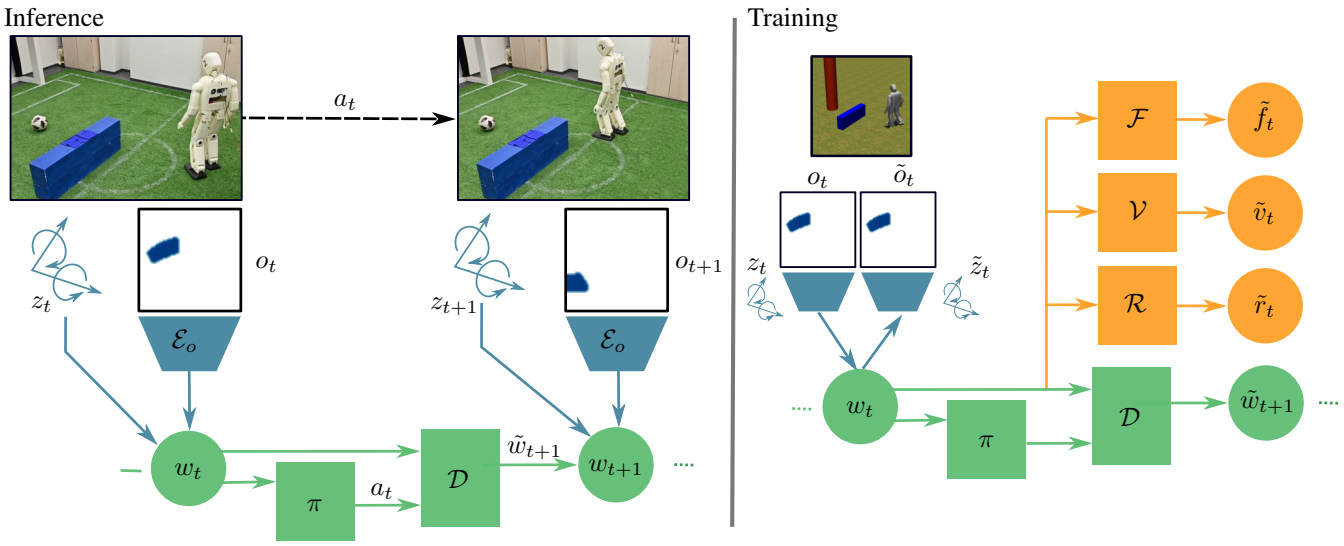

Fig. 2. Approach overview. Visual o_t and non-visual z_t observations are fused into a latent vector w_t which is used by the policy π to infer actions a_t. The dimensionality of the images is reduced by employing an autoencoder $\mathcal{E}_o$. The dynamics model $\mathcal{D}$ predicts the next latent state $\tilde{w}_{t+1}$ which is later filtered by sensor data observed in $t+1$, namely by z_{t+1} and by $\mathcal{E}_o(o_{t+1})$. During training, a decoder is also learned which aims to reconstruct an observed image o_t from w_t. Additionally, the state value $\mathcal{V}$, the reward $\mathcal{R}$, and the termination likelihood $\mathcal{F}$ predictors are learned, which are used in the loss function of the policy π (Eq. (6)).

contrast to the fully stochastic prediction models, the latent state is constructed using the Recurrent State Space Model (RSSM), which represents the latent space by a mixture of deterministic and stochastic states [23, 25].

The autoencoder $\mathcal{E}$ as well as the prediction model $\mathcal{R}$ are trained using the negative log likelihood of the true data from an experience replay buffer. In addition, the latent state dynamics loss is based on the Kullback-Leibler divergence between the open-loop 1-step prediction and the closed-loop 1-step prediction. The loss is based on the Information Bottleneck objective [26], defined as:

$$\mathcal{L}_{\mathcal{E},\mathcal{R},\mathcal{D}} = -\mathbb{E}\Big[\sum_t \ln \mathcal{E}\left(o_t|w_t\right) + \ln \mathcal{R}\left(r_t|w_t\right) - \beta\, \mathrm{KL}\left[\mathcal{E}\left(w_t|\mathcal{D}\left(\tilde{w}_t|w_{t-1}, a_{t-1}\right), o_t\right) || \mathcal{D}\left(\tilde{w}_t|w_{t-1}, a_{t-1}\right)\right]\Big]. \tag{2}$$

In contrast to the $\mathcal{E}, \mathcal{R}$ and $\mathcal{D}$ networks, the value model $\mathcal{V}$ and the actor π are not trained on the recorded episodes, but on state trajectories that are generated through consecutive inference of the learned latent state dynamics $\mathcal{D}$ in conjunction with the actor π on a single filtered state posterior. This results in a tuple $(w_{t_i}, \ldots, w_{t_{i+H}})$ of unfiltered states that are used to train the value and actor models over a horizon of length H. The loss of the value prediction network minimizes the regression error of the state value that is calculated via reward predictions:

$$\mathcal{L}_{\mathcal{V}} = -\mathbb{E}\left[\sum_{t=t_i}^{t_{i+H}} \left\|\mathcal{V}(w_t) - v_t\right\|^2\right], \tag{3}$$

while the loss of the actor π maximizes the value of the generated state tuples:

$$\mathcal{L}_{\pi} = \mathbb{E}\left[\sum_{t=t_i}^{t_{i+H}} v_t\right]. \tag{4}$$

IV. Method

Our approach aims to solve the mapless navigation problem. We fuse visual o_t and non-visual observations z_t into a latent state w_t. Actions a_t are taken by a learned policy π that is trained by open-loop (imagined) latent trajectories inferred by an environment dynamics model $\mathcal{D}$. State value $\mathcal{V}$, reward $\mathcal{R}$, and termination likelihood $\mathcal{F}$ predictors are also learned during training. The latter is employed to increase the sample efficiency of this episodic task. The high dimensionality of images is reduced by the incorporation of a variational autoencoder $\mathcal{E}_o$. Fig. 2 shows an overview of our approach.

A. Observation Model

We enrich the observation space by considering both, image inputs and non-visual sensory data. Consequently, we propose two separate autoencoders, $\mathcal{E}_o$ and $\mathcal{E}_z$, where $\mathcal{E}_o$ represents the convolutional autoencoder for image inputs and $\mathcal{E}_z$ processes non-visual sensor information. Since z_t is low dimensional, we forego the encoder part of $\mathcal{E}_z$ and only utilize the decoder to predict the measurements from the latent state. The autoencoders $\mathcal{E}_o$ and $\mathcal{E}_z$ are trained by optimizing the negative log likelihood of the true observations under the observation models:

$$\mathcal{L}_{\mathcal{E}_z,\mathcal{E}_o} = -\mathbb{E}\left[\sum_t \ln \mathcal{E}_o(o_t|w_t) + \ln \mathcal{E}_z(z_t|w_t)\right]. \tag{5}$$

B. Termination Likelihood Predictor

A terminal state can represent either success or failure in episodic tasks. Typical episodic tasks define a successful

terminal state that indicates the achievement of the task's goal. Moreover, early termination is an established strategy for improving sample efficiency, such that an episode is terminated when certain states are reached whose contribution is considered negligible for the overall task, e.g., states that represent a biped robot lying on the floor in a navigation task. In this manner, the sample acquisition time and the corresponding gradient propagation are avoided for these terminal states that do not contribute to reaching the task's goal.

Having multiple terminal states f_i poses a challenge to the design of the reward function, as it is no longer possible to reward or to penalize termination per se. Due to the different nature of each terminal state, success and failure need to be addressed separately. In episodic tasks, one issue with successful termination emerges when the agent prefers to collect rewards instead of terminating because it continues accumulating reward. While termination rewards at the end of an episode promise a fast and straightforward solution to this issue, their inherent discontinuity makes them hard to predict. Thus, we introduce a termination likelihood model, which predicts a continuous indicator $f_{i,t}$ for reaching a terminal state. In contrast to $\mathcal{R}, \mathcal{V}$ and π, the termination likelihood is modeled as beta distributed. The inferred termination likelihood is weighed and passed as a smooth learning signal to the actor model, enabling the agent to anticipate success and failure states. The actor loss is then reformulated as:

$$\mathcal{L}_\pi = -\mathbb{E}\left[\sum_t \left(v_t + \sum_i \lambda_i \mathcal{F}(f_{i,t}|w_t)\right)\right]. \tag{6}$$

C. Task Definition

The goal of the agent is to reach a desired 2D pose on a flat ground plane without collisions with obstacles in the environment.

The agent perceives the environment through RGB images and additional non-visual sensors. The images are taken from an ego perspective of a walking humanoid robot and, hence, contain much walking-induced motion. They are passed through a semantic segmentation module that classifies obstacles pixelwise. Unnecessary textural information and background pixels are therefore removed. This image segmentation facilitates the image prediction and the real-world transfer. The resulting segmented image (resolution 64×64 in our experiments) defines the visual observations o_t of our approach.

In addition, the non-visual observation is defined as $z_t = [\mathbf{V}_t, h_t, d_t, \theta_t, \mathbf{R}_t]^T$, where $\mathbf{V}_t$ is the current gait velocity, h_t is the yaw joint position of the head, $[d_t, \theta_t]^T$ is the relative target position expressed in polar coordinates, and $\mathbf{R}_t$ contains the pitch and roll rotation of the robot base link. Note that in real world applications, the relative target position is often determined by high level-task planners or perception modules.

In each time step, the agent selects an action $a_t = [\Delta\mathbf{V}_t, \Delta h_t]^T$, where $\Delta\mathbf{V}_t$ is an increment of the gait velocity, i.e., $\mathbf{V}_{t+1} = \mathbf{V}_t + \Delta\mathbf{V}_t$, and Δh_t represents an increment of the yaw head position. Note that the incremental action representation is introduced to guide the agent learning process, especially at the beginning of training where exploration of the action space might lead to oscillating motions that saturate the low-level joint controllers. The velocity vector $\mathbf{V}_t = [v_x, v_y, \omega_z]^T$ consists of the translational x- and y-velocities, as well as a rotational velocity around the z-axis of the robot. Overall, a 4D action space is defined.

D. Terminal States

We propose two different termination criteria. The robot arrives into a successful terminal state when the distance d_t to the target is below a certain threshold, whereas the failure terminal state is reached when the sum of the absolute roll and pitch rotations $|\mathbf{R}_{0,t}| + |\mathbf{R}_{1,t}|$ of the robot surpasses limit values that indicate an imminent robot fall. Both error values, i.e., the distance and orientation errors, are passed through an exponential decay to yield a continuous signal $f_{i,t}$ that indicates termination whenever $f_{i,t} = 1$. Note that the causality $f_{i,t} = 1 \implies \forall \Delta t > 0 : f_{i,t+\Delta t} = 1$ holds, which can be incorporated into the latent world model.

E. Reward Function

We define the task reward at time t, $r_t = \sum_{i=0}^{N} \eta_i r_{i,t}$ as the weighted sum of N sub-reward terms. For brevity, the dependence of time will be dropped in the equations.

The main sub-rewards encourage the agent to reach the target pose and are formulated as:

$$r_d = 1 - \frac{d}{d_0} \quad \in (-\infty, 1], \tag{7}$$

$$r_\theta = -\left|\frac{\theta}{\pi}\right| \quad \in [-1, 0], \tag{8}$$

where d is the distance to the target position, d_0 is the distance from the initial pose to the target, and $\theta \in (-\pi, \pi]$ is the relative orientation of the robot to the target position, for example, $\theta = 0$ means the robot is directly facing the target position. The former sub-reward encourages the agent to walk towards the target by reducing the distance d, while the latter penalizes the robot when it is not facing the target.

In addition, we define a sub-reward based on the location of the target inside the robot's camera image. This target attention reward is generated by the multiplication of an importance map $\mathbf{L} \in [0, 1]^{64\times64}$ with a binary segmented image $\mathbf{I} \in \{0, 1\}^{64\times64}$ showing only the target:

$$r_a = \frac{\sum_{i,j} \mathbf{L}_{i,j}\mathbf{I}_{i,j}}{\sum_{i,j} \mathbf{I}_{i,j}} \in [0, 1]. \tag{9}$$

We set $r_a = 0$ if the target is not visible in the image, i.e., $\sum_{i,j} \mathbf{I}_{i,j} = 0$. To ensure that the agent prefers to keep the target in the center of the observed egocentric images, we set $\mathbf{L}_{i,j} = 1$ for pixels at the center while we quadratically discount the values towards $\mathbf{L}_{i,j} = 0$ at the borders of the image.

Consequently, the agent will try to keep the target in the center of its field of view mainly by controlling the head yaw joint, whose motion relates directly with the

relative movement of the target in the observed images. Moreover, this relative position of the target is also affected by the gait and contacts with the floor. In order to avoid oscillating motions of the head, we penalize the normalized head position h and the normalized head control action Δh quadratically:

$$r_h = -(\Delta h)^2 \in [-1, 0], \tag{10}$$

$$r_H = -(h)^2 \in [-1, 0]\,. \tag{11}$$

In addition, we encourage the agent to maintain a safe distance to obstacles by penalizing its distance towards the closest obstacle ρ:

$$r_\rho = \text{clip}\,(-(1-\rho), -1, 0) \in [-1, 0]\,. \tag{12}$$

Finally, we penalize the current gait velocity using a sigmoid kernel $k(x) = 1/[1 + \exp(-\alpha x - c)]$ to limit the maximum gait velocity of the agent due to difference between the simulated and the real gait. This penalization is formulated as:

$$r_v = 1 - k(||\mathbf{V}||_2)\,. \tag{13}$$

V. Evaluation

We evaluate our approach on the NimbRo-OP2X humanoid robot [27]. All training is done using experience collected only in simulation employing MuJoCo as multi-body simulator. Eight environments are executed in parallel to speed up the data acquisition. The policy frequency is 10 Hz, while the simulation runs at 1 kHz. The robot incorporates a bipedal gait engine, which generates leg motions based on a target gait velocity $\mathbf{V}_t$ [27, 28]. The gait runs at 100 Hz. For collision checking operations, the robot links are approximated by geometrical primitives.

Each episode starts with the humanoid robot standing without any obstacle in its direct vicinity. The target position, the number of obstacles, their poses and geometries are drawn uniformly at random. To encourage the development of robot skills to circumvent obstacles, each episode places an obstacle between the initial position of the robot and the target pose with probability $p_{block} = 0.5$. The feasibility of reaching the target is checked by an A* planner; if no path is found, a new environment is generated. The agent has a maximum time of 60 s to complete the task before the episode ends.

The agent captures a monocular color image (64×64), which is later processed to get the segmented input image. The inferred actions are bounded to $\Delta\mathbf{V} \in [-0.06, 0.06]^3$ and $\Delta h \in [-0.012, 0.012]$. The weights of the reward functions are: $\eta_d = 1.0$, $\eta_\theta = 0.2$, $\eta_a = 0.2$, $\eta_h = 0.08$, $\eta_H = 0.08$, $\eta_\rho = 0.2$, and $\eta_v = 0.35$, and the weights of the terminal states are: $\lambda_s = -1150$ and $\lambda_f = 3250$. The models are trained every 4,000 recorded steps by batches containing 50 samples of length 50. The learning rates take the following values: 6×10^{-4} for the observation model and 8×10^{-5} for the value and actor networks. The horizon of the open-loop trajectories is set to 15. The policy is trained for 2 million simulation steps, resulting in a total training time of around 2 days on a computer with an Intel i9-9990K CPU, 64GB of RAM and an nVidia GeForce 2080 Ti with 12GB of VRAM. This model is able to solve simple scenes after 200.000 steps, i.e., less than five hours of training.

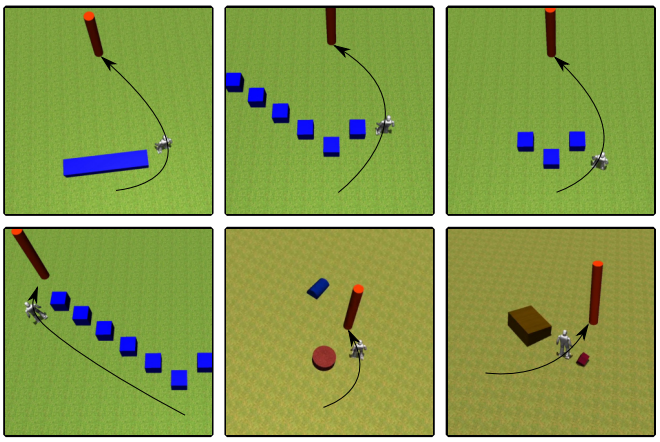

Fig. 3. The learned policy successfully navigates in different scenarios where a direct path to the goal pose is blocked by obstacles. Observe that the agent is able to circumvent small and large single obstacles. Finer control is evidenced in scenarios where the robot is required to go through a passage of obstacles (bottom right)

After training, the control policy is able to command the robot to reach target poses avoiding obstacles. Figure 3 shows sample scenarios that the robot is able to navigate collision-free with our learned control policy. As anticipated, the robot is able to circumvent obstacles and to go through narrow passages without falling. All the environments are presented to the robot for the first time.

We compare our approach against two ablated versions of our method. The first ablation (M_f) does not include the termination likelihood predictor and the second one (M_{nv}) does not consider non-visual observations. The return and success rate are presented in Fig. 4. Note that although the M_{nv} model accumulates more reward compared to our model, it is not able to reach the target indicated by its low success rate. The larger reward is attributed to longer sequences that do not reach the goal, where, for example, the robot might stand in front of the target. The main contribution in the performance increment is clearly attributed to the introduction of non-visual information, which is rather to be expected because the agent is not forced to obtain information from the images

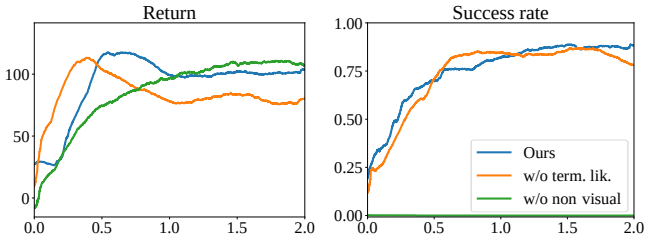

Fig. 4. Performance over the number of collected episodes. We compare our approach against two ablations by removing the non-visual observations and the termination likelihood predictor.

TABLE I. Success rate, accumulated reward and episode length of our model and ablations over 100 samples.

	w/o non-visual	w/o P_{term} f_i	Ours
Success rate	0.0	0.7	0.86
Acc. reward	108.50	104.37	111.59
Ep. length	585.36	401.15	419.77

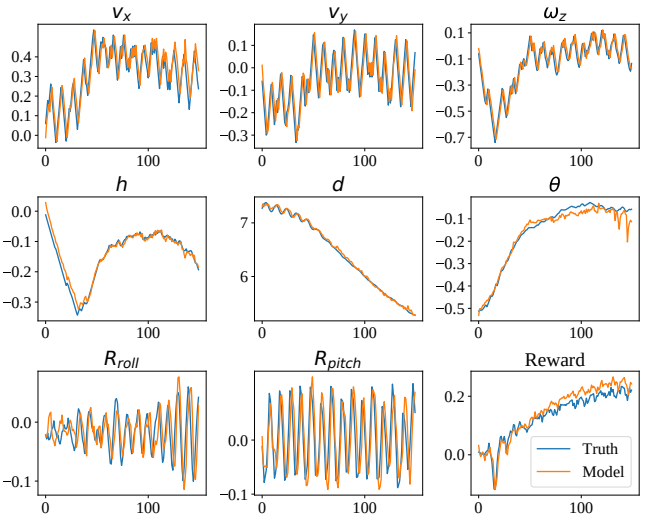

Fig. 5. Reconstructed non-visual observations and reward signal for a sampled sequence inferred by our models. The better the reconstruction, the more realistic are the imagined trajectories used for training.

but it is given directly. In other words, the incorporation of non-visual information improves the data-efficiency of the overall approach.

We evaluate the performance of each of the learned models compared above. We generate a set of 100 random scenes sampled as during training. The results are presented in Table I. After 2 million steps, the M_{nv} model is not able to successfully complete the task. The accumulated reward and the long episodes indicate that the agent does not fall but does not find the target in the given time per episode (60 s). The higher success rate and reward of our approach compared against the model without termination predictor $\mathcal{F}$ demonstrates the improvement in the sample efficiency by incorporating a predictor for f_i.

Since the performance of the agent strongly depends on the ability of the model to reproduce the actual observations and reward values from the latent state, we evaluate the quality of their reconstruction. Figure 5 shows the reconstructed non-visual observations and reward for a random sequence. For clarity in the figure, only one sequence is shown, but other sequences present a similar behavior. Note how well the model tracks the ground truth signals, which is to be expected once the models have converged. The accuracy of the reconstruction is an indicator of the quality of the inferred open-loop trajectories used for training the value and actor networks.

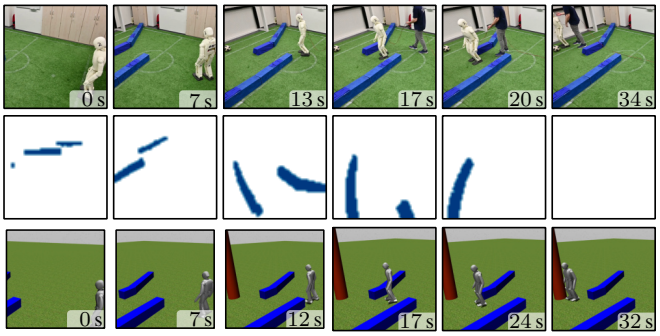

Fig. 6. The top row shows the real robot navigating a scene by taking a right turn, followed by rotating left to walk through a tight corridor between the obstacles. At the end, the robot walks into the target (soccer ball) by stepping laterally. The corresponding segmented image observations of the real experiment are shown in the middle row. At the bottom rows, a simulated scene is presented, where the agent chooses a similar path as the real-world one.

A. *Real-World Transfer*

The simulated robot is equipped with the same sensor systems as the real robot. Furthermore, the visual complexity of the camera images is reduced through semantic segmentation. In conjunction, this allows a real-world transfer with low additional effort and no retraining. Due to sim-to-real dissimilarities in the robot model, joint controllers and contact properties, the simulated gait does not behave as the one on the real robot. To facilitate the sim-to-real transfer, we inject Gaussian noise, $\mathcal{N}(0, 0.3)$, on the inferred actions during training. In addition, the gaits are tuned by introducing scaling factors to the inferred actions in order to obtain a similar response in simulation and with the real hardware. Figure 6 shows a real and a simulated robot performing the same scenario consisting of traversing a narrow passage. The row in the middle presents the segmented images captured with the real robot, whereas the bottom row shows the task performed in simulation. This is a challenging scenario that requires precise actions from the agent to avoid collisions. The temporal differences between the real and the simulated trajectories are attributed mainly to contact parameters such as frictions.

Finally, the control policy is tested with dynamic obstacles with the real robot. Figure 7 shows snapshots of the robot avoiding a moving obstacle which is blocking the direct path to the target pose.

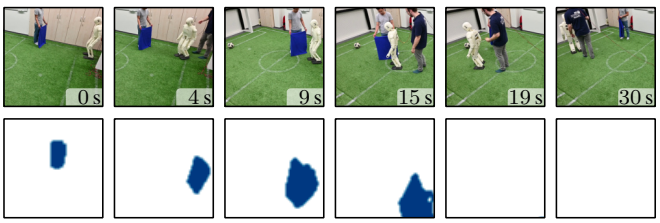

Fig. 7. The real NimbRo-OP2X robot avoids a moving obstacle that is constantly blocking the path to the target pose (top row). The segmented images taken from a first-person view are shown in the bottom row.

VI. Conclusion

In this paper, we have proposed a novel approach for learning mapless navigation around obstacles based on visual and non-visual observations. We have demonstrated that the incorporation of a termination likelihood predictor increases the data-efficiency of the approach. In addition, we have shown that our model produces a robust policy that can be successfully transferred to a real humanoid robot.

In the future, we would like to extend our approach to incorporate hierarchies. Multiple consistent policies are envisioned to solve more complex tasks that require long-term planning. Additionally, learning local and global maps seems to be a promising alternative to provide the agent with more sophisticated navigation skills, such as remembering dead ends. More dynamic scenarios where multiple objects move simultaneously require the agent to track and estimate the velocities of the moving bodies, which also states an interesting problem to enrich our approach.

References

[1] T. Klamt, M. Schwarz, C. Lenz, *et al.*, "Remote mobile manipulation with the Centauro robot: Full-body telepresence and autonomous operator assistance," *Journal of Field Robotics (JFR)*, vol. 37, no. 5, pp. 889–919, 2019.

[2] G. E. Jan, K. Y. Chang, and I. Parberry, "Optimal path planning for mobile robot navigation," *IEEE/ASME Transactions on Mechatronics*, vol. 13, no. 4, pp. 451–460, 2008.

[3] D. Rodriguez, H. Farazi, G. Ficht, D. Pavlichenko, A. Brandenburger, M. Hosseini, O. Kosenko, M. Schreiber, M. Missura, and S. Behnke, "RoboCup 2019 AdultSize Winner NimbRo: Deep Learning Perception, In-Walk Kick, Push Recovery, and Team Play Capabilities," in *RoboCup 2019: Robot World Cup XXIII*, ser. Lecture Notes in Computer Science, vol. 11531, 2019, pp. 631–645.

[4] D. Silver, A. Huang, C. J. Maddison, A. Guez, L. Sifre, G. van den Driessche, J. Schrittwieser, I. Antonoglou, V. Panneershelvam, M. Lanctot, S. Dieleman, D. Grewe, J. Nham, N. Kalchbrenner, I. Sutskever, T. Lillicrap, M. Leach, K. Kavukcuoglu, T. Graepel, and D. Hassabis, "Mastering the game of Go with deep neural networks and tree search," *Nature*, vol. 529, pp. 484–503, 2016.

[5] D. Silver, T. Hubert, J. Schrittwieser, I. Antonoglou, M. Lai, A. Guez, M. Lanctot, L. Sifre, D. Kumaran, T. Graepel, T. Lillicrap, K. Simonyan, and D. Hassabis, "Mastering Chess and Shogi by Self-Play with a General Reinforcement Learning Algorithm," *arXiv*, 2017.

[6] D. Silver, J. Schrittwieser, K. Simonyan, I. Antonoglou, A. Huang, A. Guez, T. Hubert, L. Baker, M. Lai, A. Bolton, Y. Chen, T. Lillicrap, F. Hui, L. Sifre, G. van den Driessche, T. Graepel, and D. Hassabis, "Mastering the game of Go without human knowledge," *Nature*, vol. 550, pp. 354–359, Oct. 2017.

[7] G. Barth-Maron, M. W. Hoffman, D. Budden, W. Dabney, D. Horgan, D. TB, A. Muldal, N. Heess, and T. P. Lillicrap, "Distributed Distributional Deterministic Policy Gradients," in *6th International Conference on Learning Representations (ICLR)*, 2018.

[8] D. Ha and J. Schmidhuber, "Recurrent World Models Facilitate Policy Evolution," in *Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems (NeurIPS)*, 2018, pp. 2455–2467.

[9] O. Zhelo, J. Zhang, L. Tai, M. Liu, and W. Burgard, "Curiosity-driven exploration for mapless navigation with deep reinforcement learning," *IEEE International Conference on Robotics and Automation (ICRA), Brisbane, Australia*, 2018.

[10] L. Tai, G. Paolo, and M. Liu, "Virtual-to-real deep reinforcement learning: Continuous control of mobile robots for mapless navigation," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2017.

[11] A. Khan, V. Kumar, and A. Ribeiro, "Learning sample-efficient target reaching for mobile robots," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2018.

[12] G. López-Nicolás, C. Sagüés, J. J. Guerrero, D. Kragic, and P. Jensfelt, "Switching visual control based on epipoles for mobile robots," *Robotics Auton. Syst.*, vol. 56, no. 7, pp. 592–603, 2008.

[13] M. T. De Xu and Y. Li, "Visual control system for robotic welding," *Industrial Robotics*, p. 713, 2006.

[14] H. M. Becerra, G. López-Nicolás, and C. Sagüés, "Omnidirectional visual control of mobile robots based on the 1D trifocal tensor," *Robotics Auton. Syst.*, vol. 58, no. 6, pp. 796–808, 2010.

[15] L. Xie, S. Wang, A. Markham, and N. Trigoni, "Towards monocular vision based obstacle avoidance through deep reinforcement learning," *CoRR:1706.09829*, 2017.

[16] D. Pavlichenko, D. Rodriguez, C. Lenz, M. Schwarz, and S. Behnke, "Autonomous Bimanual Functional Regrasping of Novel Object Class Instances," *IEEE-RAS International Conference on Humanoid Robots (Humanoids)*, 2019.

[17] A. Devo, G. Mezzetti, G. Costante, M. L. Fravolini, and P. Valigi, "Towards generalization in target-driven visual navigation by using deep reinforcement learning," *IEEE Transactions on Robotics*, 2020.

[18] V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. A. Riedmiller, A. Fidjeland, G. Ostrovski, S. Petersen, C. Beattie, A. Sadik, I. Antonoglou, H. King, D. Kumaran, D. Wierstra, S. Legg, and D. Hassabis, "Human-level control through deep reinforcement learning," *Nature*, no. 7540, pp. 529–533, 2015.

[19] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal policy optimization algorithms," *CoRR abs/1707.06347*, 2017.

[20] D. Rodriguez and S. Behnke, "DeepWalk: Omnidirectional bipedal gait by deep reinforcement learning," *IEEE International Conference on Robotics and Automation (ICRA)*, 2021.

[21] K. Lobos-Tsunekawa, F. Leiva, and J. Ruiz-del-Solar, "Visual navigation for biped humanoid robots using deep reinforcement learning," *IEEE Robotics and Automation Letters (RA-L)*, 2018.

[22] R. S. Sutton and A. G. Barto, *Reinforcement learning: An introduction*. MIT press, 2018.

[23] D. Hafner, T. P. Lillicrap, I. Fischer, R. Villegas, D. Ha, H. Lee, and J. Davidson, "Learning latent dynamics for planning from pixels," in *36th International Conference on Machine Learning (ICML)*, 2019.

[24] D. Hafner, T. P. Lillicrap, J. Ba, and M. Norouzi, "Dream to control: Learning behaviors by latent imagination," in *8th International Conference on Learning Representations (ICLR)*, 2020.

[25] A. Doerr, C. Daniel, M. Schiegg, N.-T. Duy, S. Schaal, M. Toussaint, and T. Sebastian, "Probabilistic recurrent state-space models," in *Proceedings of Machine Learning Research*, 2018.

[26] N. Tishby, F. C. Pereira, and W. Bialek, "The information bottleneck method," in *37-th Annual Allerton Conference on Communication, Control and Computing*, 1999, pp. 368–377.

[27] G. Ficht, H. Farazi, A. Brandenburger, D. Rodriguez, D. Pavlichenko, P. Allgeuer, M. Hosseini, and S. Behnke, "NimbRo-OP2X: Adult-sized open-source 3D printed humanoid robot," in *18th IEEE-RAS International Conference on Humanoid Robots (Humanoids)*, 2018.

[28] P. Allgeuer and S. Behnke, "Bipedal walking with corrective actions in the tilt phase space," in *18th IEEE-RAS International Conference on Humanoid Robots (Humanoids)*, 2018.