Mitriakov, A., Papadakis, P., Nguyen, S. M., and Garlatti, S. (2020). Staircase Traversal via Reinforcement Learning for Active Reconfiguration of Assistive Robots, IEEE World Congress on Computational Intelligence

Staircase Traversal via Reinforcement Learning for Active Reconfiguration of Assistive Robots

Andrei Mitriakov
Lab-STICC, UMR 6285 F-29238, team RAMBO
IMT Atlantique
Brest, France
firstname.lastname@imt-atlantique.fr

Panagiotis Papadakis
Lab-STICC, UMR 6285 F-29238, team RAMBO
IMT Atlantique
Brest, France
firstname.lastname@imt-atlantique.fr

Sao Mai Nguyen
Lab-STICC, UMR 6285 F-29238, team RAMBO
IMT Atlantique
U2IS, ENSTA, IP Paris & Inria, FLOWERS team
nguyensmai@gmail.com

Serge Garlatti
Lab-STICC, UMR 6285 F-29238, team RAMBO
IMT Atlantique
Brest, France
firstname.lastname@imt-atlantique.fr

***Abstract*—Assistive robots introduce a new paradigm for developing advanced personalized services. At the same time, the variability and stochasticity of environments, hardware and unknown parameters of the interaction complicates their modelling, as in the case of staircase traversal. For this task, we propose to treat the problem of robot configuration control within a reinforcement learning framework, using policy gradient optimization. In particular, we examine the use of safety or traction measures as a means for endowing the learned policy with desired properties. Using the proposed framework, we present extensive qualitative and quantitative results where a simulated robot learns to negotiate staircases of variable size, while being subjected to different levels of sensing noise.**

***Index Terms*—Cognitive robotics, learning-based control, obstacle negotiation, active stability, reinforcement learning, neural networks**

Fig. 1: Example platform and application scope: *Scewo Bro* wheelchair traversing a staircase (https://scewo.ch/en/bro/).

I. Introduction

Among a number of applications for service robots, there is a growing interest towards assisting activities of daily living, for instance via assistive robots [1]. To be able to palliate the loss of autonomy of elderly or frail people, in the context of ambient assisted living (AAL), a robot is required to transport objects or humans (see Fig.1 for a representative example) in an environment populated with various 3D obstacles such as steps and stairs. To enable mobility in 3D environments, a 2D environment structure typically used in path planning is overly restrictive if the robot requires to undertake novel tasks. The first possible application relates to an autonomous mobile robot (agent) navigating in indoor, usually multi floor, environments while accomplishing different tasks provided by a human or an algorithm. Such a robot can be integrated into a smart home with connected objects and mobile robots such as presented in [2] in the context of the leadership (Chaire) research program M@D whose research goal concerns the creation of a living environment for elderly or frail people whose loss of autonomy would be palliated by the use of connected objects and mobile robots (https://chaire-mad.fr/la-chaire-md/).

A major challenge then amounts to autonomously and safely traversing (negotiating) staircases which constitutes the scope of this paper. Thanks to task similarity, once the first use-case has been successfully addressed, we could transfer the developed behavior to the second use-case of person transportation.

Earlier works addressing the stair traversal problem [3], [4] were largely based on solutions customized to specific, previously known robot kinematics and based on accurate stair parameter estimates. Broadly speaking, this complicates portability to platforms with different or unknown kinematics or poor environment observability, weakly taking into consideration the dynamics of physical interaction. On the other hand, learning-based control of flippers [5] allows to make less restrictive hypotheses with respect to variation in the environment or the robotic kinematics and focuses more on task constraints such as safety [6]. A more recent approach [7] for navigation in multi floor environment, that has shown good

performance on more than hundreds of staircase instances, relies on the passive adaptation to obstacles which cannot be generalized to assistive robots due to high movement stochasticity.

Following up on earlier work [8], we present in this article a learning-based control approach for the problem of staircase negotiation (see Fig. 2), applied to an articulated, tracked robotic vehicle. Employing a reinforcement learning-based (RL) pipeline, our focus here is on the constraints of the control policy that we wish to satisfy, such as safety, traction, execution speed. To demonstrate the generalization ability of the framework, the policy is learnt and tested for a variety of staircases of different step heights and angles, while evaluating its capacity to cope with different levels of noise in its sensory data. This initial work on simulation can serve as paradigm for transfer to different platforms but it can also provide insights when modelling control for newly encountered situations.

The remaining of the paper is organized as follows. Section II presents related works for the problem of stair negotiation together with stability assessment techniques. In section III the proposed reinforcement learning framework is presented. In section III-B we formulate the notion of safe stair traversal and design the reward function. Finally, in section IV the experimental setup is presented together with the obtained results.

II. Related work

Following various attempts for the kinematic modelling of tracked robots, the authors of [9] were among the first to propose an effective motion controller. To improve the capability of negotiating more complex obstacles, such robots are often equipped with articulated front and/or rear flippers. These flippers improve obstacle negotiation skills but they inevitably increase the complexity of robot control due to the addition of degrees of freedom while accounting for surface geometry. We can broadly distinguish two main approaches for the control of such robots, namely, learning-free and learning-based.

A. Learning-free approaches

Among learning-free approaches, a terrain traversability (we refer the reader to [10] for an extensive review on the notion of traversability) study was performed in [11] as well as a motion control algorithm for a tracked robot with two front flippers. This work assumes knowledge of the exact robot kinematics which is not always feasible for commercial robots, while motion control is only applied to the traversal of a palette. The authors of [12] elaborate the idea of tangential orientation of flippers during obstacle traversing along with a motion planning framework. They suggest that this allows to negotiate any uneven structure within the limits of robot capabilities, yet based on exhaustive laser sensory data about the environment. In [13] the authors introduce a framework capable of dealing with obstacles exceeding obstacle negotiation capabilities of [12]. They endow a tracked robot with passive flippers, which apply force against the traversing obstacle, and a warning system based on the normalized energy stability margin (NESM). This stability criterion, which estimates deviation from the most stable position, became widely used in robotics while keeping comprehensibility and will be discussed later in detail. The previous framework considers precise knowledge of robot dynamics and supposes that a human expert controls the robot.

To the best of our knowledge, the most elaborate and recent learning-free method was proposed in [7] and delegates the problem of stair negotiation to passive adaptation of the mechanical platform. The latter comprises six arc-shaped legs rotating under a certain control algorithm used both in outdoor and indoor multi floor navigation. This platform is more suitable for search and rescue missions for what it was initially devoted to [14]. Obviously, its utility is limited in the context of assistive robotics, due to the stochasticity of its movement that could cause considerable slipping and shaking.

B. Learning-based approaches

Employing learning-based control becomes more relevant in cases where the true kinematics of such robots are harder to obtain or approximated in simulation [15]. The authors of [5] were among the first who used the learning-based approach and applied deep deterministic policy gradient (DDPG) to the staircase negotiation problem in an end-to-end fashion. They mapped input data from an inertial measurement unit, front and back cameras with flipper commands using a convolutional neural network (CNN). The main shortcoming of this work is the high processing cost induced from the employed CNN and slow convergence to the optimal policy. One of the most elaborate approaches for flipper control in the scenario of palette traversal is proposed by [16]. The main contribution concerns the use of contextual relative entropy policy search [17] by introducing safety constraints into the optimization problem. A small number of episodes was deemed sufficient for learning to negotiate a palette using a RL algorithm, estimating safety by a physics-based simulator. Often, knowledge of forward/inverse robot kinematics cannot be obtained for commercial robots. In such cases, a learning-based approach appears more suited for obtaining the policy of flipper control for the task of obstacle negotiation.

C. Safety assessment

Moreover, the policy used for traversing terrains has to ensure safety of the robot motion in some manner. For example, safety of actions could be estimated and a robot could be programmed to avoid such unsafe actions [13]. Another method is to incorporate a safety criterion inside a RL algorithm [16]. However, this does not generalize well across all RL algorithms. Another alternative considers integration of penalty terms into the reward function [18] allowing to obtain a policy with desired properties with a wider generalization across RL algorithms. In this paper, we consider this method and incorporate stability and traction criteria into the reward function and study the learning performance along with policy safety and traction.

D. Reinforcement learning preliminaries

In this section, we concisely recall the main components of reinforcement-based learning.

In a typical RL framework [19], at each discrete time step t and a given state $\boldsymbol{s} \in S$, an agent selects and applies an action $\boldsymbol{a} \in A$ with respect to its policy $\pi : S \longrightarrow A$, obtains a scalar reward $r \in \mathbb{R}$ and observes a new state $\boldsymbol{s'} \in S$. Although A can be discrete or continuous, the latter choice seems more adapted to use-cases requiring variable accuracy depending on the state. For instance, in our use-case, the robot may encounter situations where it is necessary to adapt flippers drastically avoiding step-wise approaching to a desired flipper angle as it would happen with the use of discrete action space. The trajectory $\tau = \{\boldsymbol{s}_0, \boldsymbol{a}_0, .., \boldsymbol{s}_{T-1}, \boldsymbol{a}_{T-1}\}$, called the *roll-out* where T is the number of time steps in the roll-out, defines the set of consecutive state-action pairs. A policy can either be stochastic $\boldsymbol{a}_t \sim \pi_\theta(\boldsymbol{a}_t|\boldsymbol{s}_t)$ or deterministic $\boldsymbol{a} = \pi(\boldsymbol{s})$, depending on the problem.

RL methods could be divided into two groups which are value-based and Policy Search (PS) methods. The first makes an agent evaluate the quality of an action taken in some state by calculation of a value function in accordance with the expected cumulative reward.

Methods of the second group [19] tend to be more advantageous in robotic applications [20]. The reasons for this are ease of implementation thanks to reduced complexity of policy approximation and the fact that small changes in policy do not lead to a drastic changes in behaviour in contrast to value-based methods where an agent can start to exploit a significantly less favorable policy after policy update.

In previous years, policy gradient (PG) algorithms attracted significant attention because of their direct policy optimization in contrast to value-based methods where the policy is obtained via its relation with the learnt approximator. The main idea behind these algorithms is the gradient ascent over the policy approximator parameters $\boldsymbol{\theta}$ using the policy gradient $\nabla_\theta J(\pi_\theta)$,

$$\boldsymbol{\theta}_{k+1} = \boldsymbol{\theta}_k + \alpha \nabla_\theta J(\pi_{\theta_k}) \tag{1}$$

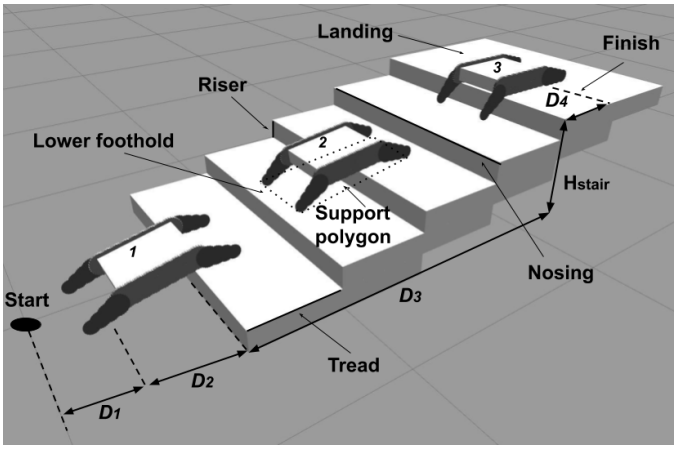


Fig. 2: Stair traversal task illustration

where α is the learning rate and $J(\pi_\theta)$ is the expected return that could be written as follows:

$$J(\pi_\theta) = E_{\tau\sim\pi_\theta}[R(\tau)] \tag{2}$$

$R(\tau)$ is the cumulative reward over a trajectory τ.

$$R(\tau) = \sum_{t=0}^{T-1} r_t \tag{3}$$

Finally, r_t is the reward obtained by the robot during a time step t, its value depending on the reward function design and will be discussed in the following section III-B.

Among various possible policy gradient algorithms, we have selected Proximal Policy Optimization (PPO) [21], [22] as a state-of-the-art algorithm that exhibits good trade-off between ease of tuning, sample complexity, ease of implementation and good performance [23]. This algorithm updates the policy with stochastic gradient ascent while keeping a new policy close to the previous one that could be completed either by penalisation of Kullback–Leibler divergence or specialized clipping in the objective function.

III. Reinforcement Learning Framework

A. Flipper Control Problem Formalisation

Robot control could be divided into reactive main track and flipper control [13], [16]. Reactivity implies that the controller directly maps sensor input to desired actions. Hereafter, we assume an independently developed main track controller (i.e. a global path planner) and focus only on the development of flipper control whose objective is to perform a mapping between the pose of a robot on a staircase and the appropriate flipper commands.

In relation to the addressed problem, we consider the robot to select two rotation angles $\psi_{front}, \psi_{rear}$ forming an action vector $\boldsymbol{a} = (\psi_{front}, \psi_{rear}) \in [\psi_{min}, \psi_{max}]^2$ for the front and rear flippers while constant effort is applied to each track moving it forward. The state vector $\boldsymbol{s} = (p_x, p_y, \phi_{front}, \phi_{rear}) \in S$ consists of the distances p_x, p_y of the robot centroid to the next step nosing along the axes X and Y (see Fig. 3) where the origin corresponds to the step center. It further includes the flipper angles ϕ_{front} and ϕ_{rear} in each flipper frame which belong to $[\phi_{min}, \phi_{max}]$ where $\phi_{min} < \psi_{min} < \psi_{max} < \phi_{max}$.

In accordance to the reactive control architecture, the task to be learned consists of flipper reconfiguring so as: (i) to enable the robot to reach the top of the staircase and (ii) to fulfil certain policy constraints such as traction and stability.

This is performed with application of PPO where the agent's goal is to maximize the cumulative return (Eq.3). Learning efficiency is sensible to the reward function design as well as learning policy that can incorporate desired or unpleasant characteristics by mistake. We introduce such reward functions which bring safety characteristics to the policy in the next section and, afterwards, analyze their impact on stability in comparison with the "default" reward function elaborated for stair negotiation within discussed problem formulation.

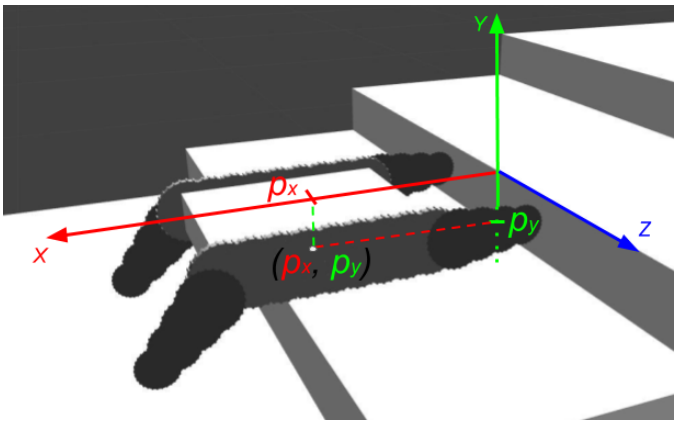


Fig. 3: Relative distance of the robot centroid to the next step edge/nosing

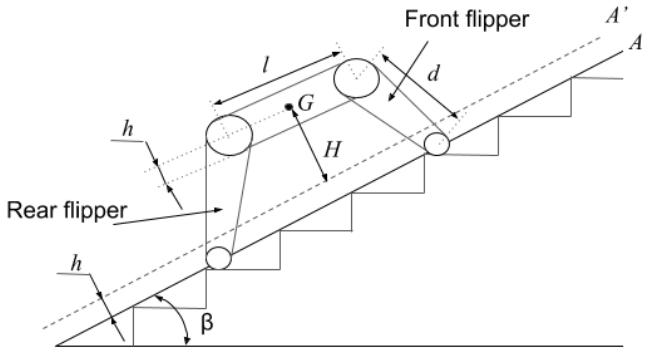


Fig. 4: NESM notations

Our contribution of this paper is summarized as follows: (i) we apply Proximal Policy Optimization (PPO) to learn flipper control for the stair traversal task in simulation (see Fig. 2) and (ii) we examine the use of stability and traction measures during policy learning and (iii) we show the generalisation capability of our policy by varying the terrain parameters and simulating sensory noise.

B. Reward function design

1) Baseline/default reward function: The first and most simple examined reward function remunerates the robot according to the travelled distance on the staircase. It receives the positive reward $r_t = \Delta x_t / \|D_{max}\|$ where Δx_t is the travelled distance to the goal on the ground, stair or upper landing at time step t, and $\|D_{max}\|$ is the total distance from the starting position to the goal. We do not assign the positive reward twice if the robot slips back and re-travels an already seen position. The robot starts to receive this positive reward when it comes closer than $D_2 = d + 0.5l$ on the ground to the first stair step (see Fig. 2, 4), while the episode ends when the robot crosses D_4 on the stair landing. Thus, the cumulative episode reward $R(\tau)$ becomes:

$$R(\tau) = \frac{\sum_{i=0}^{T-1} \Delta x_t}{\|D_{max}\|} \tag{4}$$

That total reward hence varies from 0 to 1 accordingly to an applied policy. If the robot gets closer than D_2 without appropriate flipper actions, it could be stuck (cumulative reward 0), thus, we start to provide the reward to evaluate policy performance once the robot is close enough to the first step. The episode is considered terminated if the distance D_4 is crossed on the upper landing (cumulative reward 1). D_4 is chosen to be equal to $d + \frac{l}{2}$, because we want to assure that the robot loses the contact with the stair at the end of the episode, and the traversal is successfully accomplished.

2) Normalized Energy Stability Margin (NESM): Following the idea of applying negative rewards to unsafe states in Markov Decision Processes [18], we incorporate a negative penalty term by introducing a stability criterion into the reward function. In detail, we employ the Normalized Energy Stability Margin (NESM) [24], [25] that is based on the fact that the robot rotates around a support line when tumbling/tipping over. This margin E is usually deduced as the difference between the maximum centroid height and its current height [13].

$$E = H_{max} - H \tag{5}$$

Based on this measure, we associate robot instability I as equal to the current height H over the surface A' (see Fig. 4). Thus, when $I = 0$ the robot is considered as maximally safe, whereas when $I = H_{max} = \cos(\beta)\sqrt{d^2 + (\frac{l}{2})^2 - ld\cos(\pi - |\phi_{min}|)} - h$ where β is the stair inclination, the robot is maximally unsafe. The value H_{max} is the distance of the robot centroid to the surface A' when the robot is pivoting over the lower support polygon foothold and $\phi_{rear} = \phi_{min}$. We consider the negative reward based on NESM to be $-I/1m$ where division by $1m$ ensures independence from the chosen units of measurement. Thus, the reward per time step $r_t = \Delta x_t / \|D_{max}\| - I$. Every time step the robot can obtain a positive reward along with a negative one if its gravity center G deviates from the surface A', whenever the robot centroid projection is located within the first and last nosings on the surface A'.

3) Support polygon (SP): The idea of maximizing the contact of flippers to the traversing surface is commonly employed as a measure of traversability. Authors of [12] suggested to set the orientation of flippers tangential with respect to the traversed surface during obstacle negotiation in order to maximize traction. In [13] flippers are even pushed against the traversed surface in order to enforce contact.

It is reasonable to assume that a minimal contact surface area should be maintained, between the flippers and the obstacle being negotiated, otherwise the exerted control actions might not have the desired effect. On the other hand, depending on the geometry of the robot and the staircase, maximizing surface contact might drive the robot to poses from which it would be impossible to successfully traverse the staircase.

Knowing the actual size proportions between a conventional staircase and commercial tracked robotic vehicles, maximizing traction while traversing a staircase indeed amounts to moving at an angle equal to the inclination of the staircase. Therefore, we assume that the maximum traction with the surface is achieved when the robot maximizes its SP. The maximum SP surface area P_{max} is then simply the projected surface area of the robot on the surface A when the robot is located on the staircase with a flat configuration, namely, with flat flippers.

Letting the actual SP surface area be P, in order to favor a control policy where the robot has increased traction, a negative reward is considered at each time step $r_p = -(P_{max} - P)/(P_{max})$. Thus, the overall reward per time step is $r_t = \Delta x_t / \|D_{max}\| + r_p$.

IV. Experiments

In this section, we present the experiments that we conducted to evaluate the RL control pipeline as was described earlier in Section III, quantitatively as well as qualitatively, by varying the form of staircases and in the presence of different levels of simulated noise.

Main experimentation protocol: We simulate the control-learning pipeline of the robot in the Gazebo environment (gazebosim.org). The tracked robot is endowed with front and rear flippers and its task is to learn how to mount a five-step staircase whose tread and rise are randomly chosen. Among various alternatives for imitating mobile robot tracks, we employ the common approximation of representing tracks by an ensemble of overlapping, non self-colliding, tracked wheels (cf. [6]).

Experiments were performed repeatedly for a given configuration of a staircase. At each time step the robot observes the state vector $\boldsymbol{s}$. The state vector comprises relative distances to the next step nosing p_x and p_y. The former varies from the minimum distance $p_x^{min} = 0$ to the horizon of step observation $p_x^{max} = 1m$. Relative distance along Y axis has as its maximum value the biggest height p_y^{max} which is assumed to be the negotiation robot capability $H_{critical}$, thus, $p_y^{max} = H_{critical} = h + d \cdot \sin\frac{\pi}{4}$ and its minimum $p_y^{min} = -d \cdot \sin\frac{\pi}{4}$.

In every episode, irrespective of the results of the previous episode, a new staircase of 5 steps is generated in the following way. First, a random rise size $H_{rise} \in [H_{rise}^{min}, H_{rise}^{max}]$ is sampled where $H_{rise}^{min} = 0.5 \cdot H_{critical}$ and $H_{rise}^{max} = 0.8 \cdot H_{critical}$. Second, the stair angle $\beta \in [\beta_{min}, \beta_{max}]$, where $\beta_{min} = 20°$ and $\beta_{max} = 30°$, is randomly sampled. Last, the tread size is calculated as $D_{tread} = H_{rise} / \tan(\beta)$.

The robot starts at a front-parallel position with respect to the staircase at a distance of $D_1 + D_2$. Once the robot travels D_1, upon application of an action the robot obtains a reward r proportional to the travelled distance and potentially penalized depending on the chosen reward function (see section III-B). A traversal experiment is deemed as successful if the robot succeeds in traversing over D_4 on the upper landing from the last nosing. The finite-horizon undiscounted return of one trajectory represents the proportion of travelled distance and could vary from 0 to 1 if we do not model safety. This sequence is repeatedly executed until the end of an episode that is set to occur after a maximum number of 100 time steps. If the standard deviation of 25 most recent robot position estimations along the X-axis drops under 0.01 m, the episode stops assuming that the robot has been "stuck". All experiments have 10000 time steps that compares with 250 episodes while remaining parameters were chosen in accordance with [26].

Employed policy representation: We wish to map the robot state represented by a vector $\boldsymbol{s}$ to a vector of actions $\boldsymbol{a}$ under the policy π, which requires a good policy function approximation. Artificial neural networks have demonstrated remarkable results in machine learning as well as in reinforcement learning in particular, and could be well suited for nonlinear function approximation [5], [19], [27]. Having small input and output vectors of estimated robot state and corresponding actions, we approximate the policy with a multilayer perceptron consisting of two layers where each of them has 32 neurons with $tanh$ activation function, in contrast to [5] which employs a complex CNN because of necessity to treat complex image input data. Judging from the obtained qualitative data, this policy function approximator provided satisfactory results for the task under consideration.

Policy testing: A learnt policy is evaluated on stair configurations that do not necessarily appear during learning. We perform this after the first policy update and, then, every 8 policy updates on three stair configurations with parameters presented in Table I. During this assessment, we calculate mean cumulative reward, stability and traction over 3 cases which correspond to *small*, *medium* and *big* staircases also called configurations.

TABLE I: Stair configurations used in evaluation

Type	***Step height***	***Stair angle***
Small	H_{rise}^{min}	β_{min}
Medium	$0.5(H_{rise}^{min} + H_{rise}^{max})$	$0.5(\beta_{min} + \beta_{max})$
Big	H_{rise}^{max}	β_{max}

A. Reward function evaluation

Along with the quantitative results presented later in this section, we bring a video that provides representative results of the learnt policies for staircases of varying difficulty, accessible in (https://partage.imt.fr/index.php/s/pBSzKaoeDnqFSyA/download).

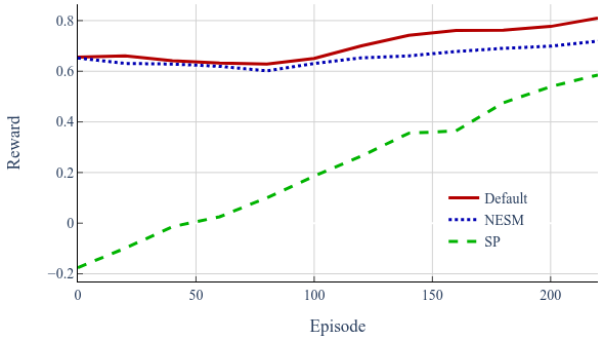
For every type of reward function, we performed 3 training repetitions. During experiments, cumulative reward, NESM-based instability measure I and estimated robot projection ratio P/P_{max} were measured every 20 episodes.

Fig. 5 presents the smoothed average cumulative reward per learning episode and testing, using min-max bands. Hereinafter referred smoothing is accomplished using an exponential moving average (EMA) with the coefficient 0.95 and 0.8 for learning and testing curves respectively.

With respect to learning performance (Fig. 5 (a)), the NESM-based and SP policies converge as early as the unsafe policy (around 175 episodes) and their mean episode returns vary from 0.5 at the beginning up to 1 at the end. We can observe the presence of non-zero episode cumulative rewards in the beginning that could be provoked by random actions that the robot makes in the beginning of training which allows it to reach half of its chassis length on the staircase. On the other hand, the optimal reward values per episode are slightly lower for policies that incorporate safety, because of unavoidable necessity to perform less safe actions. We can further observe that max bands of learning curves reach the maximum return after 100 episodes for all types of reward function. This means that policies are yet capable to perform appropriate flipper control, but min bands are constantly increasing due to behaviour improvement on unseen staircases. By 200 episodes, the mean curves converge and this could be considered as the end of learning.

With respect to testing performance (Fig. 5 (b)) for stair configurations of varying difficulty (cf. Table I), we observe that the mean reward tends to become higher during learning, which suggests a successful learning progress, independently of the staircase. Moreover, the mean test learning curve for SP policies constantly increases the cumulative reward $R\tau$ that also confirms steady learning of a policy that satisfies safety constraints.

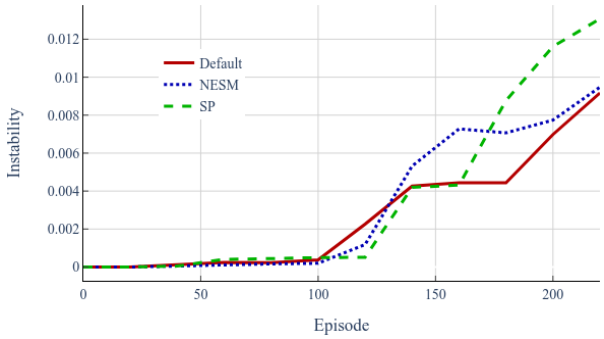
(a) Learning



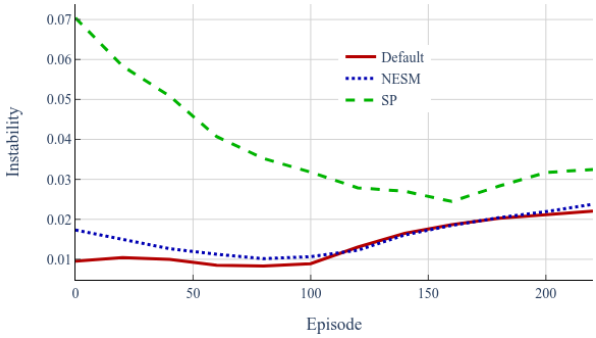
(b) Testing

Fig. 5: Smoothed cumulative reward $R(\tau)$ during (a) learning and (b) testing

With respect to the results shown in Fig. 6 (a), we can deduce that the policy safety does not change too much in the beginning of the learning due to possible difficulties to negotiate the stair. Instability for all 3 policy types starts to increase after 100 episodes which correlates to learning curves whose max bands reach the maximum cumulative reward 1. We can argue that a safety drop is unavoidable to perform in order to accomplish the main task of stair traversal. By the end of the training, the lowest instability corresponds to NESM-based and default (unsafe) policies, while the instability of SP tends to be higher. Test instability curves (see Fig. 6 (b)) show the same behaviour, the SP policy is less safe by the end of training, while the NESM-based and unsafe policies have the same instability values. One notable difference on test learning curves (Fig. 6 (b)) in the beginning of learning, namely, is that the SP-based policy is considerably more unstable compared to the two other policy types.



(a) Learning

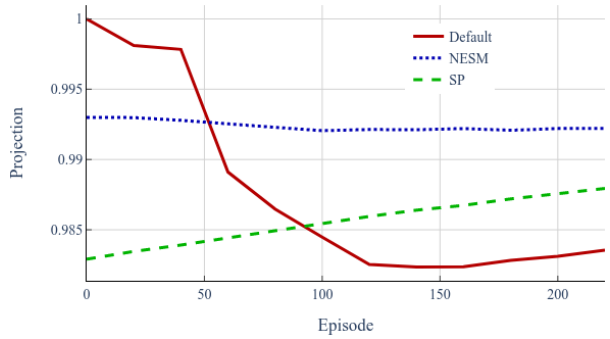


(b) Testing

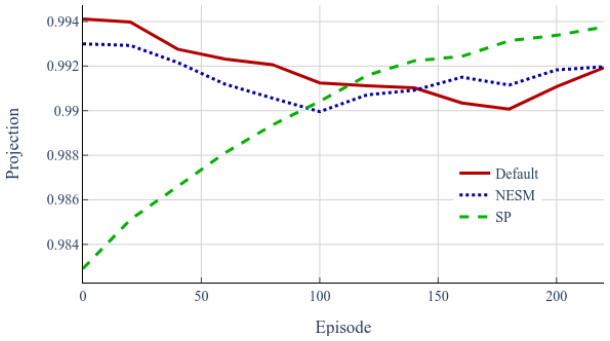
Fig. 6: Smoothed instability I during learning and tests

The projection assessment curves (see Fig. 7 (a)) show that the NESM and SP-based policies tend to better ensure a flat robot configuration compared to the default policy. The NESM and SP policies perform better in most cases than the unsafe policy after episodes, NESM-based and projection penalisation terms could be considered as improving traction. The SP policy tends to increase its projection meanwhile the NESM-based policy decreases it. Thus, the SP policy learns desired behaviour of flippers. In accordance with the tests in Fig. 7 (b), the SP-based policy tends to increase traction with the stair, two other policy types decrease their projection that

correlates to learning projection curves. It seems surprising at first inspection that a safety-based NESM and unsafe policies exhibit the same behaviour (see 6 (b), 7 (b)) in terms of instability and traction. However, we penalize elevation of the robot over the stair and at the same time the robot has to be relatively flat in order to negotiate a staircase. Thus, the robot does not visit states which may drive it to unsafe state and this constraint is automatically satisfied.



(a) Learning



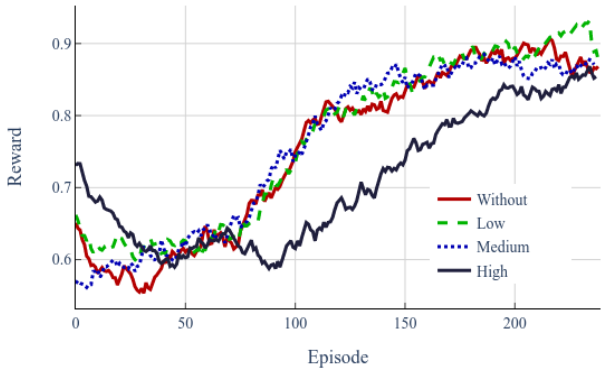
(b) Testing

Fig. 7: Smoothed projection P/P_{max} during learning and tests

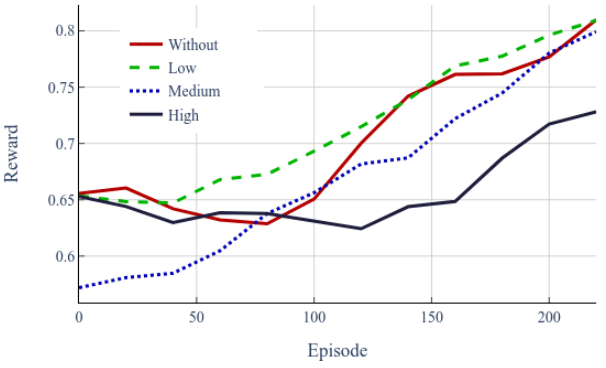
B. Resilience of policy learning to noise

Our targeted application in the context of AAL where the robot is integrated within a smart space/house [28], suggests a relatively well-controlled environment. The part of the state of the robot concerning its relative position with respect to the traversed steps could be obtained by fusion of on-board sensors with ambient sensors. Nevertheless, due to potential slipping of the platform while traversing the staircase it is important to assess the robustness of policy learning in the presence of varying levels of noise, since erroneous state estimation could lead to wrong actions and influence the robot performance even making it impossible to attain the desired goal.

Thus, noise should be accounted for during learning at the moment when we provide the robot with estimation of its relative position to the next step. We simulate corrupted estimation of p_x, p_y via a Gaussian error model, namely, $\widehat{p_i} \sim \mathcal{N}(p_i, \sigma_x)$ where $\widehat{p_i}$ is the noisy position, p_i is the true position given by the simulation environment, $\sigma_i = \xi \cdot V_{max}$



(a) Learning



(b) Testing

Fig. 8: Smoothed cumulative reward $R(\tau)$ during learning and tests, noising

is the chosen standard deviation, i denotes either X or Y axes, V_{max} stands for the maximum possible observation value and ξ is a parameter modelling different noise levels. We experiment with three levels of noise, abbreviated as *low*, *medium* and *high*, that correspond to $\xi = \{0.1, 0.3, 1.0\}$.

Firstly, three sets of experiments were conducted for each noise level using the baseline/default reward function. We can observe (see Fig. 8 a)) that policies learned with low or medium noise converge as the default policy. On the other hand, high noise in state estimation decreases the learning convergence rate, but eventually, the same cumulative reward is attained after 250 episodes. During testing (see Fig. 8 (b)), high noise levels seemed to have a more notable impact on the final cumulative reward, yet, we do not expect such extreme noise levels to be representative of real conditions of our AAL scenario. For reference, the results provided in Tables II, III demonstrate that noisy state estimation does not have an noticeable impact on safety constraints.

V. Conclusion

In this paper we have presented a RL framework for the staircase negotiation. Within 200 learned trajectories which correspond to 60 policy updates, the robot learned how to safely traverse staircases consisting of 5 steps with varying rise and stair angle in representative real-world ranges. We have designed and compared three alternative reward functions. The

TABLE II: Resilience to noise for safety constrains during learning

Measure type	Level of noise			
	Without	Low	Medium	High
I	0.017 ± 0.021	0.015 ± 0.017	0.013 ± 0.025	0.005 ± 0.009
P/P_{max}	0.962 ± 0.047	0.986 ± 0.007	0.966 ± 0.029	0.967 ± 0.026

TABLE III: Resilience to noise for safety constrains during testing

Measure type	Level of noise			
	Without	Low	Medium	High
I	0.018 ± 0.01	0.022 ± 0.008	0.017 ± 0.01	0.015 ± 0.01
P/P_{max}	0.991 ± 0.003	0.99 ± 0.003	0.992 ± 0.002	0.99 ± 0.003

NESM-based policy did not behave significantly differently with respect to the unsafe policy. Incorporation of the projection maximization into the reward function produced a desired behaviour in terms of flipper sticking to the stair. A policy learnt without safety criteria presented a good generalization capability when applied to newly encountered stair parameters. Finally, while noisy sensory data may decrease convergence rate the final control policy attains the maximum reward in most cases. In our subsequent work, we plan to couple these results with [8], and apply it on a real robot allowing it to autonomously navigate in 3D indoor environments.

VI. Acknowledgements

The present work is performed in the context of the leadership program M@D (Chaire Maintien@Domicile), project REACT (Physical Interaction Models of Companion Robots) financed by Brest Métropole and the region of Brittany (France) and supported by project VITAAL (Vaincre l'Isolement par les TIC pour l'Ambient Assisted Living), cofinanced by European Regional Fund (FEDER).

References

[1] L. D. Riek, "Healthcare robotics," *Communications of the ACM*, vol. 60, no. 11, pp. 68–78, 2017.

[2] S. M. Nguyen, C. Lohr, P. Tanguy, and Y. Chen, "Plug and play your robot into your smart home: Illustration of a new framework," *KI - Künstliche Intelligenz*, pp. 1–7, 2017.

[3] D. M. Helmick, S. I. Roumeliotis, M. C. McHenry, and L. Matthies, "Multi-sensor, high speed autonomous stair climbing," in *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, 2002.

[4] J. A. Hesch, G. L. Mariottini, and S. I. Roumeliotis, "Descending-stair detection, approach, and traversal with an autonomous tracked vehicle," in *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, 2010.

[5] G. Paolo, L. Tai, and M. Liu, "Towards continuous control of flippers for a multi-terrain robot using deep reinforcement learning," *CoRR*, vol. abs/1709.08430, 2017.

[6] M. Pecka, K. Zimmermann, and T. Svoboda, "Fast simulation of vehicles with non-deformable tracks," *CoRR*, vol. abs/1703.04316, 2017.

[7] B. D. Ilhan, A. M. Johnson, and D. E. Koditschek, "Autonomous stairwell ascent," *Robotica*, vol. 38, no. 1, p. 159–170, 2020.

[8] A. Mitriakov, P. Papadakis, M. Nguyen, and S. Garlatti, "Learning-based modelling of physical interaction for assistive robots," Journées Francophones sur la Planification, la Décision et l'Apprentissage pour la conduite de systèmes, 2019.

[9] J. L. Martínez, A. Mandow, J. Morales, S. Pedraza, and A. García-Cerezo, "Approximating kinematics for tracked mobile robots," *Int. J. Robotics Res.*, vol. 24, pp. 867–878, 2005.

[10] P. Papadakis, "Terrain traversability analysis methods for unmanned ground vehicles: A survey," *Engineering Applications of Artificial Intelligence*, vol. 26, no. 4, pp. 1373 – 1385, 2013.

[11] Q. Xie, B. Su, J. Guo, H. Zhao, and W. Lan, "Motion control technology study on tracked robot with swing arms," in *IEEE Int. Conf. on Unmanned Systems*, 2017, pp. 620–623.

[12] Keiji Nagatani, Ayato Yamasaki, Kazuya Yoshida, Tomoaki Yoshida, and Eiji Koyanagi, "Semi-autonomous traversal on uneven terrain for a tracked vehicle using autonomous control of active flippers," in *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, 2008, pp. 2667–2672.

[13] S. Suzuki, S. Hasegawa, and M. Okugawa, "Remote control system of disaster response robot with passive sub-crawlers considering falling down avoidance," *ROBOMECH J.*, vol. 1, p. 20, 2014.

[14] U. Saranli, M. Buehler, and D. E. Koditschek, "Rhex: A simple and highly mobile hexapod robot," *The Int. J. of Robotics Res.*, vol. 20, no. 7, pp. 616–631, 2001.

[15] P. Papadakis and F. Pirri, "3D Mobility Learning and Regression of Articulated, Tracked Robotic Vehicles by Physics-based Optimization," in *Workshop on Virtual Reality Interaction and Physical Simulation*. The Eurographics Association, 2012.

[16] M. Pecka, V. Šalanský, K. Zimmermann, and T. Svoboda, "Autonomous flipper control with safety constraints," in *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, 2016, pp. 2889–2894.

[17] A. Kupcsik, M. P. Deisenroth, J. Peters, A. P. Loh, P. Vadakkepat, and G. Neumann, "Model-based contextual policy search for data-efficient generalization of robot skills," *Artificial Intelligence*, vol. 247, pp. 415 – 439, 2017, special Issue on AI and Robotics.

[18] S. P. Coraluppi and S. I. Marcus, "Brief risk-sensitive and minimax control of discrete-time, finite-state markov decision processes," *Automatica*, vol. 35, no. 2, pp. 301–309, 1999.

[19] R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*. MIT Press, 1998.

[20] M. P. Deisenroth, G. Neumann, and J. Peters, "A survey on policy search for robotics," *Found. Trends Robot*, vol. 2, pp. 1–142, 2013.

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

[22] A. Hill, A. Raffin, M. Ernestus, A. Gleave, A. Kanervisto, R. Traore, P. Dhariwal, C. Hesse, O. Klimov, A. Nichol, M. Plappert, A. Radford, J. Schulman, S. Sidor, and Y. Wu, "Stable baselines," https://github.com/hill-a/stable-baselines, 2018.

[23] Ł. Kidziński, S. P. Mohanty, C. F. Ong, Z. Huang, S. Zhou, A. Pechenko, A. Stelmaszczyk, P. Jarosik, M. Pavlov, S. Kolesnikov, S. Plis, Z. Chen, Z. Zhang, J. Chen, J. Shi, Z. Zheng, C. Yuan, Z. Lin, H. Michalewski, P. Milos, B. Osinski, A. Melnik, M. Schilling, H. Ritter, S. F. Carroll, J. Hicks, S. Levine, M. Salathé, and S. Delp, "Learning to run challenge solutions: Adapting reinforcement learning methods for neuromusculoskeletal environments," in *The NIPS '17 Competition: Building Intelligent Systems*, S. Escalera and M. Weimer, Eds. Cham: Springer Int. Publishing, 2018, pp. 121–153.

[24] S. Hirose, H. Tsukagoshi, and K. Yoneda, "Normalized energy stability margin and its contour of walking vehicles on rough terrain," in *Proceedings Int. Conf. on Robotics and Automation.*, vol. 1, 2001, pp. 181–186.

[25] E. Garcia, J. Estremera, and P. Gonzalez-de Santos, "A comparative study of stability margins for walking machines," *Robotica*, vol. 20, 2002.

[26] S. Fujimoto, H. van Hoof, and D. Meger, "Addressing function approximation error in actor-critic methods," *CoRR*, vol. abs/1802.09477, 2018.

[27] J. Kober, A. J. Bagnell, and J. Peters, "Reinforcement learning in robotics: A survey," *Int. J. Rob. Res.*, vol. 32, no. 11, p. 1238–1274, 2013.

[28] P. Papadakis, C. Lohr, M. Lujak, A. Karami, I. Kanellos, G. Lozenguez, and A. Fleury, "System design for coordinated multi-robot assistance deployment in smart spaces," in *IEEE Int. Conf. on Robotic Computing*, 2018.