Learning-based modelling of physical interaction for assistive robots

Andrei Mitriakov, Panagiotis Papadakis, Sao Mai Nguyen, Serge Garlatti

Learning-based modelling of physical interaction for assistive robots

Andrei Mitriakov*, Panagiotis Papadakis, Mai Nguyen, Serge Garlatti

IMT Atlantique, Lab-STICC, UMR 6285
F-29238 Brest, France

*andrei.mitriakov@imt-atlantique.fr

Abstract

Deploying companion robots for assisting humans requires safe and robust interaction with the environment, both in terms of mobility and object manipulation. To extend a robot's workspace, we are here concerned with multifloor operation and staircase traversal, as a building block for the development of object fetching services. In this article, we advocate a life-long learning treatment of this problem within a reinforcement learning (RL) framework. In view of sparse earlier work for the scenario of interest, we hereby identify relevant methodological aspects and report our preliminary developments.

Keywords

Assistive robotics, Control learning, Autonomous flipper control, Policy search methods

1 Introduction

Among various application areas, service robots can be particularly useful in assisting daily human activities, for example via companion robots [6]. As far as our application of interest is concerned, such robots are assumed to operate within an indoor environment. In such scenarios, robot operation is most often presumed as 2D although, in practice, the environment can be 3D and highly variable. To compensate motor impairments of elderly or frail people with limited autonomy, it becomes reasonable to develop robots with improved autonomy in navigation and object manipulation. Relatedly, for the task of stair traversal, this can be addressed by hard-coded behaviours that make use of the exact robot kinematics and the staircase characteristics [12, 13]. In general terms, this approach is not efficient as it is not easily transferable to different robots. Likewise, manipulation with objects or movement on flat or uneven surfaces separately have been receiving considerable attention during recent years [22, 32, 4]. On the contrary, transporting potentially sensitive objects while performing staircase traversal has not been studied, especially in the context of a personal assistance scenario. Conventionally, stair traversal is performed by methods presented in the work [1] that usually make some artificial hypotheses used in control that makes the robot less adaptable to new situations, i.e. new robot configurations and stair shapes. Finally, reinforcement learning could constitute a more suitable approach for avoiding ad hoc solutions that are customized to particular platforms but it has not widely been studied.

A number of questions are posed in order to accommodate such scenarios. How does the task of safe stair traversal is formulated as a reinforcement learning problem while taking into consideration the presence of an arm carrying a sensitive object? How is it possible to generate actions which respect robot safety? How can novel (unseen) environment states or robot configurations be accommodated? How can we account both for robot safety and other potentially conflicting criteria of performance such as traction? In view of these questions, our research goal is to develop robust algorithms that would allow safe operation for a service robot capable of (i) indoor, multifloor navigation via obstacle negotiation such as stairs or steps and (ii) combination with object-centric assistive services. In this context, the use of approaches that are not based on learning is overly restrictive because of their inferior generalization capability. Therefore, the key challenge of the doctoral research concerns the elaboration of methods for 3D mobility and/or manipulation where the role of learning is eminent, allowing them to be less dependent on a particular platform and to require less expert supervision.

The rest of the paper is structured as follows. Section 2 presents related works of the problem of interest. In section 3 we formulate the problem of stair traversal using notions of reinforcement learning. Section 4 provides preliminary results related to machine vision algorithms for stair detection in color and depth images. Finally, section 5 is focused on future works in navigation and reinforcement learning.

2 Related work

2.1 Learning-free approaches

The task of stair traversal has been predominantly studied in learning-free frameworks in search and rescue applications where robots do not generally face the same constraints as in service robotics, namely, object and environment safety is usually less important relatively to assistive robot applications where it is inadmissible to dam-

age human property. Related robot navigation problems were studied outside the context of machine learning. For example, the authors of [23] propose a framework for autonomous 3D path and motion planning including flipper control for tracked robots where the main idea is to keep flippers tangentially in contact with the surface. Authors show that their solution can face any kind of structure within the limits of the robot's obstacle negotiation abilities, however they only demonstrate it with two trials. In [30] the authors present an approach where a tracked robot with passive sub-crawlers faces obstacles that exceed obstacle negotiation capabilities of [23], a warning system based on normalized energy stability margin (NESM) and a traversal algorithm that is able to apply force against an obstacle in order to climb it. The NESM-based approaches for stability assessment became widespread in robotics while at the same time being easy to understand because the stability criterion simply supposes the calculation of vertical deviation from the lowest stable position of the main robotic chassis [14].

2.2 Learning-based approaches

In this section we begin by presenting works that are more relevant to learning-based staircase traversal and we conclude by focusing on the most significant algorithms. The authors of [25] were among the first to test the deep deterministic policy gradient (DDPG) algorithm for the stair traversal task following an end-to-end fashion. The idea is to get actions from input data received from the Inertial measurement unit (IMU) sensor, front and back cameras. Q-values, which are the quality measure of a state-action combination, and policy parameters were approximated using sophisticated multilayer convolutional neural networks (CNN), which inevitably induce significant processing costs. Another limitation of this work is due to the sole use of time-demanding simulations as the result of the chosen function approximations that require a large number of trials. In turn, this results in lower generalization as a result of training on a single stair type.

Another more recent and elaborate approach is proposed by authors of [26]. The authors' contribution amounts to the development of RL algorithms for the scenario of stair traversal. They implement constraints in the contextual relative entropy policy search (Contextual REPS) algorithm [20] based in turn on [28] and [10]. That extension showed that a small number of iterations is sufficient to learn stair traversal of a previously unknown obstacle where the safety of a rollout is computed by a physics-based simulator. The latter represents software simulating the main interactions and influence of the real-world system and determining as the safety of the rollout as 1 if it is safe or 0 otherwise. However, the authors did not investigate the impact of context represented by variables like the height of an obstacle that do not change during the task execution but might change from task to task. We consider this paper [26] as reference work with respect to the approach that we envision to advance.

3 Reinforcement learning problem

To exploit the advantages of learning-based staircase traversal, we intend a reinforcement learning approach applied to flipper control. Two families of reinforcement learning methods exist in robotics, the first one is the policy search (PS) which learns the policy, and the second one is value-based where one wishes to estimate the quality of the value function in conformity with the expected cumulative reward. PS [31] provides more advantages over the value-based approach in robotics because the former allows expert knowledge integration, domain appropriate prestructuring of the policy, and reduced complexity of policy approximation relative to value function approximation. Finally, small changes in policy do not consequently lead to a large change in a value function and again in the policy [17] like in value-based methods.

3.1 Problem description

We briefly recall associated RL notions following the description given in [19]. We denote the state of the robot as $\boldsymbol{x}$. The vector of control $\boldsymbol{u} \in \mathbb{R}^d$ is generated by the lower-level policy $\pi_l(\boldsymbol{u}|\boldsymbol{x}, \boldsymbol{\omega})$ that is usually parameterized with feedback controllers, movement primitives or torque profiles [24, 19, 15] by $\boldsymbol{\omega} \in \Omega^d$ where Ω is the space of the lower-level policy parameters and d denotes the total number of degrees of freedom. We decouple policy into the lower and upper-level policies in accordance with [26, 20, 19]. The upper-level policy $\pi_u(\boldsymbol{\omega}|\boldsymbol{s})$ provides the parametrization $\boldsymbol{\omega}$ of the lower-level policy and can be generalized to different shapes of stairs. Let $\boldsymbol{s}$ the context containing information about the environment such as number, height and depth of steps has to be used in the upper-level policy distribution in order enable the algorithm to choose from different parametrization parameters $\boldsymbol{\omega}$ according to each context.

We consider an episode-based policy search framework with length T. The trajectory $\boldsymbol{\tau} = \{\boldsymbol{x_1}, \boldsymbol{u_1}, .., \boldsymbol{x_T}, \boldsymbol{u_T}\}$ defines the set of state-action pairs. We assume that the context vector $\boldsymbol{s}$ is drawn from an unknown distribution $\mu(\boldsymbol{s})$ and our goal is to learn the optimal upper-level policy $\pi_u^*(\boldsymbol{\omega}|\boldsymbol{s})$ which yields the parametrization of the lower-level policy given the context $\boldsymbol{s}$. The trajectory $\boldsymbol{\tau}$ has the probability $p(\boldsymbol{\tau}|\boldsymbol{s}, \boldsymbol{\omega})$ given the context $\boldsymbol{s}$ and the reward function $R(\boldsymbol{\tau}, \boldsymbol{s})$ depends on the context. In the case of staircase traversal it should further incorporate safety constraints, because the robot has to generate trajectories that are as safe as possible, and potentially additional criteria such as total travelled distance and platform slipping that we estimate using the model proposed in [8]. During learning N rollouts will be generated. In the beginning of every episode the low level parameters $\boldsymbol{\omega}$ are drawn from the upper-level policy $\pi_u(\boldsymbol{\omega}|\boldsymbol{s})$ given the observed context, then the control vector is retrieved from the lower-level policy $\pi_l(\boldsymbol{u}|\boldsymbol{x}, \boldsymbol{\omega})$. Lately, the policy parameters are evaluated

on the simulated and real robot, the reward is collected and, lastly, the policy is updated.

The problem formalization may directly serve for a stair climbing learning. The lower-level policy is parametrized by movement primitives whose parameters correspond to the parameter vector $\boldsymbol{\omega}$ of the upper-level policy. This policy defines the command vector $\boldsymbol{u}$, which contains 6 elements which are 2 torques applied to tracks and 4 flipper angles. The robot state $\boldsymbol{x}$ presents actual applied torques to tracks, flipper angles, robot orientation and the position on the stair. The context $\boldsymbol{s}$ comprises information about the environment, e.g. size, number of steps and stair inclination. Every rollout generates a trajectory $\boldsymbol{\tau}$ which has the maximum length T.

Various PS methods can be applied to the described RL problem. We consider that the relative entropy policy search (REPS) algorithm family is the state-of-the-art [26] which we seek to improve. In the next section, we briefly describe algorithms that we have interested in and consider to use as a baseline.

3.2 Policy Search algorithms

REPS. An interesting PS algorithm was developed in [27] based on a conventional RL setting [31]. The REPS keystone is simply the optimization problem where we attempt to find optimal policies that maximize the expected reward while satisfying constraints (cf. eq. 5 - 8 [27]). It binds the loss of information measured between observed data distribution and the data distribution generated by the new policy in order to prevent aggressive policy update steps.

Contextual REPS. The authors of the Contextual REPS [16] were inspired by the recent work [27], they add a task-dependent context s, from which changing features of the environment can be integrated. This also requires to divide the policy into two levels. The lower-level policy is used to generate control commands and typically parametrized with a small number of parameters $\boldsymbol{\omega}$, controllers like linear feedback controllers, movement primitives [15], torque profiles [24] or, even, neural network controllers, that are harder to learn in comparison with others, are usually used [19]. The high-level policy $\pi(\boldsymbol{\omega}|\boldsymbol{s})$ searches for parameters of the lower-level policy $\boldsymbol{\omega}$ maximizing the expected reward (cf. [16]). After sampling the parameter vector $\boldsymbol{\omega}$ from the high level-policy given the context $\boldsymbol{s}$, the lower-level policy $\pi(\boldsymbol{u}|\boldsymbol{x}, \boldsymbol{\omega})$ defines the control vector $\boldsymbol{u}$ during the episode based the state $\boldsymbol{x}$ of the robot. Contextual REPS searches over the joint distribution $p(\boldsymbol{s}, \boldsymbol{\omega})$ of contexts and parameters to maximize the expected reward J bounding the relative entropy to the previously observed distribution $q(\boldsymbol{s}, \boldsymbol{\omega})$. The Contextual REPS optimization problem aims to maximize the expected reward J while satisfying bounding constraints (cf. eq. 6 [19]).

Gaussian Process REPS. The Contextual REPS is further developed by the authors of [19]. Inspired by the recent works [28, 20] they developped a model-based contextual policy search algorithm named gaussian process relative entropy policy search (GPREPS) that learns a representation of the robot dynamics and the reward function giving equivalent learning results 100 times faster than the original REPS. Its main motivation is to improve the data-efficiency of the model-free REPS using artificial rollouts. This algorithm simply adds a Gaussian Process (GP) in the loop, which learns the representation of the system dynamics. At each iteration of the GPREPS algorithm, N trajectories are generated by observing the context and executing the policy on the real system, then the learned models are updated and based on them M artificial samples are created. In the following, the algorithm samples L trajectories and averages over the trajectory rewards getting the expected reward. Finally, these artificial samples are used for optimization of the dual function by updating the upper-level policy.

Constrained REPS. Another improvement is made in [26] that extends the Contextual REPS algorithm with constraints and replaces the GP by a cautious physics-based simulator, which evaluates generated policy in the safety simulator, constrains the upper-level policy distribution reducing the required number of iterations and generates safe trajectories. It was made by extending the original Contextual REPS constraints with an additional one. The upper-level distribution $p(\boldsymbol{s}, \boldsymbol{\omega})$ is forced to have the expected safety higher then the vector of bounds $\boldsymbol{\delta}$:

$$\sum_{\boldsymbol{s},\boldsymbol{\omega}} p(\boldsymbol{s}, \boldsymbol{\omega})(\mathbf{1} - \boldsymbol{C}_{\boldsymbol{s}\boldsymbol{\omega}}) \leq \boldsymbol{\delta}, \tag{1}$$

where $\boldsymbol{C}_{\boldsymbol{s}\boldsymbol{\omega}}$ is a collection of evaluated quantities which are safety and mechanical constraints, and $\mathbf{1}$ is a vector with all-ones of a corresponding dimension. The optimization problem of the Contextual REPS (cf. eq. 6 [19]) is enriched only with one condition (cf. 1). As was shown in [26] Constrained REPS takes about 40 iterations to learn how to traverse previously unknown obstacles represented by one step. Nonetheless, this is still overly high if we intend to adress learning of unknown stair traversal during the robot operation. Two possible extensions of this algorithm concern learning of the expected return models along with stronger prior knowledge on the policy structure than a prior knowledge of physical limits.

4 Preliminary work

We consider the Jaguar V4 (Fig. 1, `jaguar.drrobot.com`) as the experimental platform, designed for traversing 3D terrain via its active components (flippers). As we consider an indoor deployment of the robot, conventional RGB-D sensors will be used for its situation awareness. Low-level robot control is performed within Robot Operating System (ROS) [29], while the robot is simulated in the Gazebo environment (`gazebosim.org`). Preliminary results concern perception as the achieved performance of this stage will have an influence on all subsequent stages related to learning and execution.

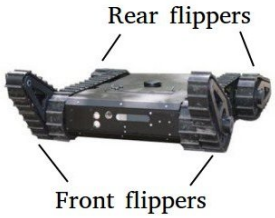


Figure 1: Jaguar V4

(a) Dominant horizontal lines obtained after applying Hough transform

(b) Detected stair edges

Figure 2: Line and step detection

4.1 Perception

The main perception skill required by the robot is to detect and estimate the state of a given staircase. Treating the problem as a sequential estimation problem wherein the robot observes the staircase from multiple viewpoints and two sensor modalities (color (RGB) and depth), a Kalman filter-based fusion scheme is applicable. At this point, we assume that a preliminary algorithm enables the robot to discriminate a staircase from other similar looking objects.

Staircase detection in the RGB image. We hereby assume a right-handed image coordinate system. A staircase is initially detected in the RGB image following ideas borrowed from [5] which consists in estimating a 2D line for each step edge. In detail, the image is converted to grayscale and undergoes erosion and dilation operations to filter out small or noisy line segments. Subsequently, the Canny edge detector together with Hough transform are used to estimate the residual lines that do not intersect with the 2D ground plane (see Fig. 2a) and extract the corresponding line coefficients in 2D image plane. In order to isolate lines corresponding to stairs, the algorithm operates in Hough space where every line is represented by a point (cf. Fig. 3) which coordinates correspond to slope and intercept values of a line.

In example, we can distinguish a multitude of lines that have been detected whose majority clusters around a single line slope, that is assumed to correspond to the line slope of the step edges of the staircase. To extract the lines belonging to the steps of the staircase, the Hough space is uniformly discritized along the slope axis into a fixed number of sections, each one corresponding to a histogram bar. Counting in each histogram bin the total number of point entries, we can thereby determine the dominant line.

Not surprisingly, Hough transform detects multiple line segments with varying length along a step edge, so these lines should be regrouped and filtered by length (cf Fig. 2a).

Regrouping could be done with respect to the fact that segment lines of different step edges have the same slope but different intercept values whereas lines from one step edge have approximately the same intercept value. In order to associate every small segment to corresponding edges, points are grouped in the following way. If the distance between two consecutive points in Hough space is less than the average distance between all consecutive points, these two points belong to one line, if not, this is the next step edge. Grouped line segments are finally filtered by length, so that short segments are discarded (Fig. 2b). Eventually, the coordinates of the staircase are obtained through a bounding box in the RGB image that contains all the grouped lines. These coordinates are then jointly considered with the estimated staircase coordinates obtained by processing the Depth image.

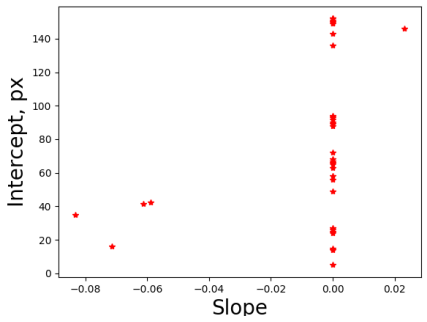
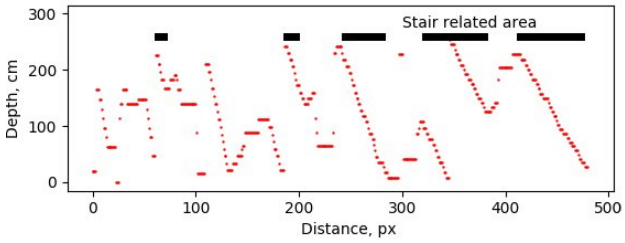


Figure 3: Lines in Hough space

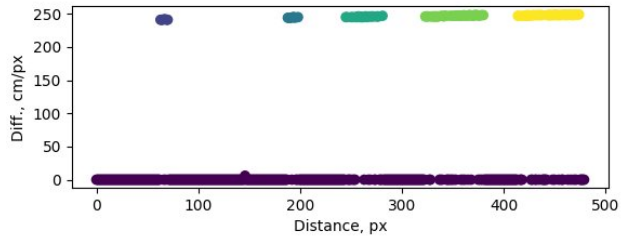
Staircase detection in the depth image. We begin by extracting N uniformly spaced vertical depth profiles of one pixel from this image starting from 0 X-axis position (see Fig. 4a). Let $p_{i,j}$ be image pixels where $i \in [1, .., w]$, $j \in [1, .., h]$, w and h are the image width and height. Depth profile at horizontal position k is defined as $dp_{k,j} = p_{i=k,j}$. Each depth profile is differentiated along itself and inverted as it is presented in (2) where Δ_j means distance between $j-1$ and j depth points (Fig. 4a), H_{max} is the maximum depth in cm.

$$dp^{diff}_{k,j} = \frac{H_{max}}{\Delta_j} - \frac{dp_{k,j} - dp_{k,j-1}}{\Delta_j} \tag{2}$$

Afterwards, we filter outliers in the differentiated profile by eliminating points which do not have necessary number of neighbors in its vicinity. Finally, we apply the DBSCAN clustering algorithm [7] to the pre-filtered differentiated profile and obtain clusters that correspond to different steps. Figure 4b illustrates clustering results where



(a)



(b)

Figure 4: (a) Depth profile at 160 px. (b) Differentiated, filtered and clustered depth profile at 220 px (0 corresponds to the top and 400 to the bottom of the image)

different clusters are associated with different colors. Accordingly, every cluster is analyzed and individual steps are distinguished from one another. Finally, the set of marked profiles fully represents the stair region and the lower step position (Fig. 5), we note that figure 5 visualizes the staircase in an uncommon way in order to increase contrast and comprehensibility of the depth image. In this way, we can obtain the position of the lowermost step which we choose to assign as the reference pose of the entire staircase. Moreover, the normal vector to the front lowermost step surface representing staircase orientation is calculated. Stair position on the 2D depth image is retrieved in the form of bounding box and is fused with the results of the color image stair detection algorithm. Finally, the staircase state parameters that are retained are the total number of steps, the minimum, maximum and average step height and depth.

To retain these parameters, we obtain the point cloud in the camera frame by transforming color and depth images with calibration parameters. Knowing camera orientation relatively to the main robot chassis and assuming that the robot is in the vertical position along gravity, we transform the point cloud coordinates from the camera frame to the main robot coordinate frame. The step depth is simply the mean distance between corresponding step edge point depths. The height is the mean distance between corresponding step edge points along gravity.

After obtaining two hypotheses on the presence of a staircase, one from RGB and the other from depth sensory data, the final decision is based on the surface area overlap of bounding boxes. The staircase estimation process continues by acquiring uniformly distributed observations in 2D space, determined by the robot's location.

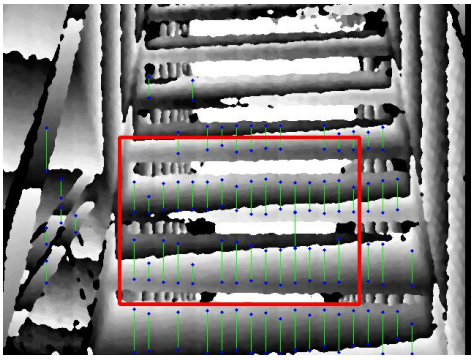


Figure 5: Detected stair in depth image; the red box is the stair related area

5 Future work

5.1 Navigation

We distinguish two mobility modes, namely, floor-based and staircase-based modes. The former is used only in the case of path execution on 2D surfaces. For this case, we will employ ideas borrowed from [9] where commands are continuously computed for track motors using the robot's kinematic model and flippers are raised up in order to decrease friction. The latter mode is applied when the robot faces the staircase. Switching between modes happens when the robot decides to traverse a staircase and approaches it at a certain distance (Fig. 6, transition state 1). When the mode changes to staircase traversal, motor commands start being computed by the trained algorithm (Fig. 6, transition states 2-4) and the robot returns to the flat mode once the traversal is accomplished (Fig. 6, transition state 5).

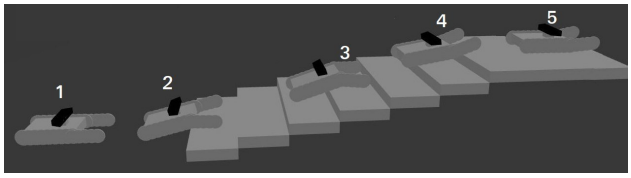


Figure 6: Robot stair climbing

We consider that the robot may traverse stairs up and down. When the robot finishes climbing, the robot saves its position on the next floor as the staircase position. Staircase descending is more challenging. Knowing staircase position after climbing up, the robot will keep in mind its location. Once the wished staircase position to descend is approached, the robot turns to the stair-traversal mode, but with another behaviour that should be learned as well as . If the robot wishes to find an unknown stair to descend, we will detect it as in [13] where authors use optical flow to detect descending stairs and extract the leading stair edge. Having a predominantly flat surface, we can rely on state-of-the-art 2D Simultaneous Localization and Mapping (SLAM) such as the one proposed in [11]. Stairs will

be localized by the perception module with their position associated to a certain location on the map as stated earlier. There are two navigable types of surfaces, namely, floor and stairs. We consider 2D SLAM, for the case when the robot resides on a flat surface. Staircases will serve as bridges shifting from one floor to another where the final staircase position is hence continuously estimated by using a Kalman filter. Eventually, this implies that we expect multiple 2D floor maps that are interconnected by staircases. The question arises about how the robot localizes itself while traversing a staircase. To address this question we consider using triangulation with odometry whereby regular patterns such as corner points could be used as landmarks whose position is calculated from 3D input depth data. To compensate for landmark disappearance along traversal, the localization of the robot will further rely on odometric data obtained either from the IMU or the track encoders that should be sufficiently robust because of small size of stairs and their regular shape. Once traversed, the robot triggers a new 2D SLAM operation corresponding to the level wherein the robot is situated.

To perform 2D path planning while navigating in floor-mode, D* Lite algorithm presented in [18] was designed to create optimal paths in dynamic scenarios by focusing replanning on the affected area. It is an incremental heuristic search algorithm based on the Lifelong Planning A*. It allows to plan paths on the occuppancy grid maps in reasonbale time and to avoid dynamic obstacles recalculating only influenced parts of the map graph. Like A* and LPA*, D* Lite uses a heuristic, that limits the cost of the path from a given node to the start. We refer the reader to the original paper [18] for more details and its pseudocode. Two flat maps connect each other by a node representing stair, hence, path planning problem is conveniently addressed in a lower-dimensional state space. The robot will have a module that tracks stair position and, if it decides to traverse stairs, the robot switches between flat surface movement mode and staircase movement mode.

5.2 Reinforcement learning-based control

Algorithms from section 3.2 may be used in application to different robotic problems. They were mostly tested in activities such as hockey [20], stair traversal [26] or table tennis [28, 19] where they showed results for the safety constraints and learning time. Thus, this renders this algorithm family particularly attractive in the case of stair traversal in order to accommodate the need for safe actions and learning in a handful of trials.

The future work comprises elaboration of Constrained REPS algorithm proposed in [26]. First, it demands a reasonably small amount of needed iterations to learn how to traverse a previously unknown obstacle. Second, its cautious physics-based simulator is important for applications in assistive robotics where we wish to avoid any damage of the environment and the robot. Lastly, being based on the Contextual REPS, the Constrained REPS enables the robot to use the context which corresponds to different, even not seen before, situations that would be new stair geometries. In our case, the context relating to staircase configurations will be employed and safety constraints will be implemented using a safety simulator, finally, results will be compared.

The GPREPS learns the robot dynamics finally decreasing the total amount of required real robot trials, however, this dynamics corresponds only to one robot configuration. The Constrained REPS does not use the Gaussian Process because, as authors say, it is difficult to provide guarantees of the safety of the generated trajectories based on data-driven models without any prior knowledge about underlying physics. The cautious physics-based simulator should be used to overcome this limitation. Nevertheless, it was shown that model-based approaches are more promising for learning in a handful of trials [2]. The GPREPS and the Constrained REPS have not yet been compared and the latter was not applied to different contexts which motivates our interest for further investigation. It should be possible to fuse these two algorithms to exploit the advantages of each of them and seek an implementation with more complex constraints.

Wishing to decrease the need of human expertise, we also desire to transfer the learned robot dynamics to different robot configurations of the spatial shape. It appears that knowledge about the robot's configuration could be added into the context vector or, possibly, it should learn its geometry using sensorimotor invariants as proposed in [21].

The next step concerns robotic arm integration. It will influence robot dynamics by translation of the mass center, therefore it has to be taken into account in the context vector and reward function. At the same time, the robot could move its arm improving capability of staircase traversal with respect to safety constraints, such movements should be also received like output of the lower-level policy. The last intention is to enable the robot to execute learning during operation in the real world in a handful of following the example of [3].

6 Conclusion

This paper presents the problem addressed in the context of a doctoral thesis, associated preliminary developments and workplan along with whose goal is to provide a tracked robot capable of safe navigation in 3D indoor environments. Contemporary robotic solutions of tracked robot stair climbing were presented and proposals for advancing the state-of-the-art were considered. Two main perspectives can be distinguished for addressing this problem. Relatedly, most contemporary approaches rely on over-customized solutions with poor generalization to different contexts. As an alternative, we advocate a reinforcement-learning, policy search based approach to reduce the amount of expert supervision and allow more flexibility to varying conditions. Subsequent work concerns development and evaluation of these modules in real-

istic conditions, system integration and scaling up learning complexity as the result of inclusion of the robotic arm.

7 Acknowledgments

The present work is performed in the context of the project M@D (chaire Maintien@Domicile) and the project VITAAL (Vaincre l'Isolement par les TIC pour l'Ambient Assisted Living) and is financed by Brest Metropole, the region of Brittany (France) and the European Regional Fund (FEDER).

References

[1] M. Brunner, T. Fiolka, D. Schulz, and C. M. Schlick. Design and comparative evaluation of an iterative contact point estimation method for static stability estimation of mobile actively reconfigurable robots. *Robotics and Autonomous Systems*, 63:89 – 107, 2015.

[2] K. I. Chatzilygeroudis, V. Vassiliades, F. Stulp, S. Calinon, and J.-B. Mouret. A survey on policy search algorithms for learning robot controllers in a handful of trials. *CoRR*, abs/1807.02303, 2018.

[3] K. Chua, R. Calandra, R. McAllister, and S. Levine. Deep reinforcement learning in a handful of trials using probabilistic dynamics models. *CoRR*, abs/1805.12114, 2018.

[4] H. Chung, C. Hou, Y. Chen, and C. Chao. An intelligent service robot for transporting object. In *2013 IEEE Int. Symp. on Industrial Electronics*, pages 1–6.

[5] Y. Cong, X. Li, J. Liu, and Y. Tang. A stairway detection algorithm based on vision for ugv stair climbing. In *2008 IEEE Int. Conf. on Networking, Sensing and Control*.

[6] K. Doelling, J. Shin, and D. O. Popa. Service robotics for the home: A state of the art review. In *Int. Conf. on Pervasive Technologies Related to Assistive Environments*, New York, USA, 2014. ACM.

[7] M. Ester, H.-P. Kriegel, J. Sander, and X. Xu. A density-based algorithm for discovering clusters a density-based algorithm for discovering clusters in large spatial databases with noise. In *Proceedings of the Second International Conference on Knowledge Discovery and Data Mining*, KDD'96, pages 226–231. AAAI Press, 1996.

[8] G. Genki, K. Nagatani, T. Hashimot, and K. Fujino. Slip-compensated odometry for tracked vehicle on loose and weak slope. *ROBOMECH Journal*, 4(1):27, Nov 2017.

[9] M. Gianni, F. Ferri, M. Menna, and F. Pirri. Adaptive robust three-dimensional trajectory tracking for actively articulated tracked vehicles. *Journal of Field Robotics*, 33(7):901–930, 2016.

[10] G. Grisetti, C. Stachniss, and W. Burgard. Improved techniques for grid mapping with rao-blackwellized particle filters. *IEEE Transactions on Robotics*, 23(1):34–46, 2007.

[11] G. Grisetti, G. D. Tipaldi, C. Stachniss, W. Burgard, and D. Nardi. Fast and accurate slam with rao-blackwellized particle filters. *Robotics and Autonomous Systems*, 55:30–38, 2007.

[12] 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.

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

[14] S. Hirose, H. Tsukagoshi, and K. Yoneda. Normalized energy stability margin and its contour of walking vehicles on rough terrain. In *ICRA. IEEE Int. Conf. on Robotics and Automation*, 2001.

[15] J. A. Ijspeert, J. Nakanishi, and S. Schaal. Learning attractor landscapes for learning motor primitives. In *Proceedings of the 15th International Conference on Neural Information Processing Systems*, NIPS'02, pages 1547–1554, Cambridge, MA, USA, 2002. MIT Press.

[16] J.Kober and J. Peters. Policy search for motor primitives in robotics. *Mach. Learn.*, 84(1-2):171–203, July 2011.

[17] J. Kober and J. Peters. *Reinforcement Learning in Robotics: A Survey*, pages 9–67. Springer International Publishing, Cham, 2014.

[18] S. Koenig and M. Likhachev. D*lite. In *Eighteenth National Conference on Artificial Intelligence*, pages 476–483, Menlo Park, CA, USA, 2002. American Association for Artificial Intelligence.

[19] 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*, 247:415 – 439, 2017. Special Issue on AI and Robotics.

[20] A. G. Kupcsik, M. P. Deisenroth, J. Peters, and G. Neumann. Data-efficient generalization of robot skills with contextual policy search. In *Proceedings of the Twenty-Seventh AAAI Conference on Artificial Intelligence*, 2013.

[21] A. Laflaquière, J. K. O'Regan, S. Argentieri, B. Gas, and A. V. Terekhov. Learning agent's spatial configuration from sensorimotor invariants. *CoRR*, abs/1810.01872, 2018.

[22] M. Menna, M. Gianni, F. Ferri, and F. Pirri. Real-time autonomous 3d navigation for tracked vehicles in rescue environments. In *2014 IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*.

[23] K. Nagatani, A. Yamasaki, K. Yoshida, T. Yoshida, and . Koyanagi. Semi-autonomous traversal on uneven terrain for a tracked vehicle using autonomous control of active flippers. In *2008 IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*.

[24] G. Neumann, W. Maass, and J. Peters. Learning complex motions by sequencing simpler motion templates. volume 382, page 95, 01 2009.

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

[26] M. Pecka, S. Valansky, K. Zimmermann, and T. Svoboda. Autonomous flipper control with safety constraints. In *2016 IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*.

[27] J. Peters, K. Muelling, and Y. Altun. Relative entropy policy search. In *AAAI*, 2010.

[28] J. Peters, K. Mülling, , and Y. Altun. Reinforcement learning by relative entropy policy search. *30th International Workshop on Bayesian Inference and Maximum Entropy Methods in Science and Engineering (MaxEnt 2010)*, 30:69, 2010.

[29] M. Quigley, K. Conley, B. P. Gerkey, J. Faust, T. Foote, J. Leibs, R. Wheeler, and A. Y. Ng. Ros: an open-source robot operating system. In *ICRA Workshop on Open Source Software*, 2009.

[30] S. Soichiro, H. Satoshi, and O. Masayuki. Remote control system of disaster response robot with passive sub-crawlers considering falling down avoidance. *ROBOMECH Journal*, 1(1):20, Nov 2014.

[31] R. S. Sutton and A. G. Barto. *Introduction to Reinforcement Learning*. MIT Press, Cambridge, MA, USA, 1st edition, 1998.

[32] L. Zhang, K. Thurow, H. Liu, J. Huang, N. Stoll, and S. Junginger. Multi-floor laboratory transportation technologies based on intelligent mobile robots. *Transportation Safety and Environment*, 2019.