Towards semi-episodic learning for robot damage recovery

Konstantinos Chatzilygeroudis, Antoine Cully, Jean-Baptiste Mouret

Towards semi-episodic learning for robot damage recovery

Konstantinos Chatzilygeroudis[1,2,3], Antoine Cully[4] and Jean-Baptiste Mouret[1,2,3]*

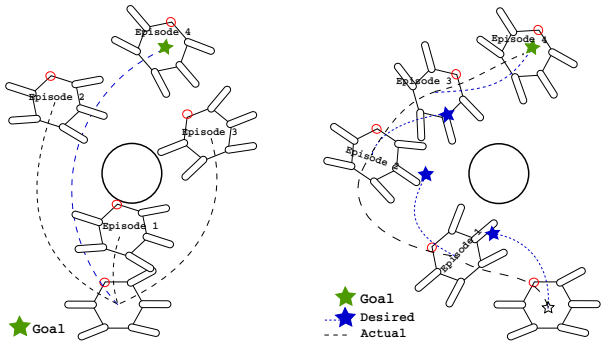
***Abstract*— The recently introduced Intelligent Trial and Error algorithm (IT&E) enables robots to creatively adapt to damage in a matter of minutes by combining an off-line evolutionary algorithm and an on-line learning algorithm based on Bayesian Optimization. We extend the IT&E algorithm to allow for robots to learn to compensate for damages while executing their task(s). This leads to a semi-episodic learning scheme that increases the robot's life-time autonomy and adaptivity. Preliminary experiments on a toy simulation and a 6-legged robot locomotion task show promising results.**

I. Introduction

Recent research on autonomous systems and robotics has achieved important progress in increasing the autonomy of robots, which makes it possible to operate robots for long periods of time in real-world scenarios. Nevertheless, as robots move from controlled and well-structured environments to more complex [1] and more natural ones [2], they must be able to react to unforeseen situations; in particular, they have to face the inevitable fact that they will be damaged [3], [4].

Current methods for robot damage recovery can be divided into two categories: (1) diagnosis-based approaches [5], and (2) learning methods — mostly Reinforcement Learning (RL) techniques [6], [7], [8]. Most of the techniques in the first category require to anticipate the situations that the robot may have to face; an issue can be diagnosed only if the right sensors are present in the right place. These requirements make diagnosis-based techniques difficult to use in complex robotics systems/scenarios — typically they are only used in the lowest levels of control. Nevertheless, the state-of-the-art RL approaches are also difficult to use for damage recovery because they require many iterations to converge. For example, many RL approaches require tens if not hundreds or thousands of iterations to learn problems with low-dimensional state spaces and fairly benign dynamics, like the mountain car [9]. The data efficiency of RL approaches is a critical aspect that limits their application in real-world robotics scenarios [10].

A promising approach is the *Intelligent Trial and Error algorithm* (IT&E), a recently introduced algorithm [8]. The intuition behind IT&E is that, before the mission, an off-line and computationally expensive evolutionary algorithm can be used to create a behavior-performance map that predicts the performance of thousands of different behaviors. While in mission, this map, guides a fast and on-line search, based on Bayesian Optimization [11], to find a compensatory behavior. An important idea is that the behavior-performance map is created using a simulated intact robot, but the algorithm is able to find a working behavior on the damaged real robot because some behaviors from the map perform similarly on the intact and the damaged robot (typically, the behaviors that do not rely on the broken part). The most recent results showed that IT&E can allow various types of robots (a 6-legged robot and an 8-DOF manipulator) to compensate for many different types of injuries in a matter of minutes [8], [12].



(a) (b)

Fig. 1: Episodic vs Semi-episodic learning for robot damage recovery. (a) **Episodic Learning**: The robot learns in episodes how to get to a single target starting from the same initial state. (b) **Semi-episodic Learning**: The robot learns the outcome of its atomic behaviors while executing the task.

Although the IT&E approach is promising, its main limitation is the pure episodic approach it has adopted: for each trial (episode), the robot has to begin in the same starting state (Figure 1a). This is limiting because learning a compensatory behavior has to be achieved in two steps, first learn a compensatory behavior, and then use it to complete the task. On the contrary, a wounded animal, for example, can perform trial and error "episodes" to learn how to walk again, while going back to its nest for protection.

In this paper, we extend the IT&E algorithm by (1) using a *generic reward* of the outcome of each atomic behavior of the robot in the adaptation part, and by (2) adding a *specialized reward selection layer* that selects a specialized reward function at each episode. These additions allow for a semi-episodic learning scheme that improves the robot's long-term autonomy by allowing to recover while attempting to achieve its task(s) (Figure 1b).

*Corresponding author: jean-baptiste.mouret@inria.fr
[1]Inria, Villers-lès-Nancy, F-54600, France
[2]CNRS, Loria, UMR 7503, Vandœuvre-lès-Nancy, F-54500, France
[3]Université de Lorraine, Loria, UMR 7503, Vandœuvre-lès-Nancy, F-54500, France
[4]Personal Robotics Lab, Department of Electrical and Electronic Engineering, Imperial College London, UK

II. Background

A. Bayesian Optimization with Gaussian Processes

Bayesian Optimization (BO) is a well-established strategy for finding the extrema of functions that are expensive to evaluate [11], [13]. It is applicable in cases where one does not have a closed-form expression for the objective function (the function is a "black-box"), but where one can obtain observations (possibly noisy) of this function. One of the distinctive features of BO is that it constructs a probabilistic model for the objective function and then exploits this model to make decisions about which point to evaluate next, while taking into account the uncertainty.

There are two major choices that must be made when performing BO. First, one must select a prior over functions that will express assumptions about the function being optimized. Second, one must choose an acquisition function, $u(\boldsymbol{x}|D_{1:t})$, which is used to construct a utility function from the model posterior, allowing us to determine the next point to evaluate.

Many models could be used for the BO prior, but Gaussian Process (GP) priors are the most common choice [11]. A GP is an extension of the multivariate Gaussian distribution to an infinite-dimension stochastic process for which any finite combination of dimensions will be a Gaussian distribution [11]. A GP is a distribution over functions, completely specified by its mean function, $m(\cdot)$ and covariance function, $k(\cdot,\cdot)$:

$$f(\boldsymbol{x}) \sim GP(m(\boldsymbol{x}), k(\boldsymbol{x}, \boldsymbol{x}'))$$

Assuming $D_{1:t} = \{(\boldsymbol{x_1}, f(\boldsymbol{x_1})), \cdots, (\boldsymbol{x_t}, f(\boldsymbol{x_t}))\}$ is a set of observations and σ^2_{noise} the sampling noise, the GP is computed as follows:

$$p(f(\boldsymbol{x})|D_{1:t}, \boldsymbol{x}) = \mathcal{N}(m_t(\boldsymbol{x}), \sigma_t^2(\boldsymbol{x}))$$

where:

$$m_t(\boldsymbol{x}) = \boldsymbol{k}^\top \boldsymbol{K}^{-1} D_{1:t}$$

$$\sigma_t^2(\boldsymbol{x}) = k(\boldsymbol{x}, \boldsymbol{x}) - \boldsymbol{k}^\top \boldsymbol{K}^{-1} \boldsymbol{k}$$

$$\boldsymbol{K} = \begin{bmatrix} k(\boldsymbol{x_1}, \boldsymbol{x_1}) & \cdots & k(\boldsymbol{x_1}, \boldsymbol{x_t}) \\ \vdots & \ddots & \vdots \\ k(\boldsymbol{x_t}, \boldsymbol{x_1}) & \cdots & k(\boldsymbol{x_t}, \boldsymbol{x_t}) \end{bmatrix} + \sigma^2_{noise} I$$

$$\boldsymbol{k} = \begin{bmatrix} k(\boldsymbol{x}, \boldsymbol{x_1}) & \cdots & k(\boldsymbol{x}, \boldsymbol{x_t}) \end{bmatrix}$$

We used *Upper Confidence Bound* (UCB) as the acquisition function. We refer the reader to *Brochu et al.* [11] for a more detailed explanation.

B. Intelligent Trial & Error Algorithm

IT&E proposed a novel approach for robot damage recovery that consists of a 2-step process. An off-line evolutionary algorithm, **MAP-Elites** [14][8], that generates many thousands of potential good behaviors is followed by a trial and error on-line adaptation part, based on BO (**M-BOA**), in order to find a compensatory behavior.

MAP-Elites is an evolutionary *illumination* algorithm: instead of searching for a single, best solution, like optimization algorithms, MAP-Elites searches for the highest-performing individual for each point in a user-defined space. This user-defined space is often called the **behavior space**, because the dimensions of variation (*behavior descriptors*) usually measure behavioral characteristics.

In IT&E, the authors made a slight modification to the classical BO scheme. Their BO variation, called *Map-Based BO Algorithm* (M-BOA), models the difference between a *mean* function and the actual performance, instead of directly modeling the objective function ($\mathbf{P}(\cdot)$ is the *mean* function):

$$m_t(\boldsymbol{x}) = \mathbf{P}(\boldsymbol{x}) + \boldsymbol{k}^\top \boldsymbol{K}^{-1}(D_{1:t} - \mathbf{P}(\boldsymbol{x_{1:t}}))$$

In the original work, the *mean* function was the prediction of the performance in the map generated from MAP-Elites. Algorithm 1 shows the pseudo-code for M-BOA.

Algorithm 1 M-BOA (Map-Based BO Algorithm)

procedure M-BOA
  $\forall \boldsymbol{x} \in map$:
    $p(f(\boldsymbol{x})|\boldsymbol{x}) = \mathcal{N}(\mathbf{P}(\boldsymbol{x}), k(\boldsymbol{x}, \boldsymbol{x}))$
  while stopping criteria not met **do**
    $\boldsymbol{x_{t+1}} = argmax_{\boldsymbol{x}} u(\boldsymbol{x}|D_{1:t})$ ▷ Next test point
    $Y_{t+1} = performance(execute\_behavior(\boldsymbol{x_{t+1}}))$
    $D_{1:t+1} = \{D_{1:t}, (\boldsymbol{x_{t+1}}, Y_{t+1})\}$
    Update GP

III. Approach

A. Generic Reward

In the original IT&E paper, the GP modeled the performance of each atomic behavior given a task. In this paper, we suggest learning a mapping from the atomic behaviors to the resulting relative outcomes. We call it a *Generic Reward* (GR) of the outcome of each atomic behavior of the robot. We use one GP for each dimension of the GR.

For example, imagine we have a robot moving in 2D space using an 1D continuous atomic behavior (direction to move 0.1-step). A GR could be the relative position of the robot after executing a behavior - (x, y). Thus, we need 2 GPs: $GP_x(\theta), GP_y(\theta)$. If we query the GPs at the point θ_0, then we get a position, $p_1 = (GP_x(\theta_0), GP_y(\theta_0))$, as the prediction. In that way, we can now compute *specialized rewards* for different locomotion tasks, like the distance to different target points.

Put differently, the GR is a description of the outcome of each atomic behavior of the robot that it is generic-enough to be independent from one task to another, but specific-enough so that the performance of the outcome of one atomic behavior given a task can be computed.

The changes for *M-BOA* to work are:

- define a *Reward* function that takes the GPs' prediction as input and returns the expected task performance;
- define an *Aggregator* function (*afun*) that takes as input the execution of an atomic behavior and returns the GR.

B. Specialized Reward Selection Layer

We, also, augment the proposed algorithm, by adding a layer responsible for selecting the *Reward* function, defined above. We call it *Specialized Reward Selection Layer* (RSL). Since we are modeling a GR of the outcome of each behavior of the robot and not the actual performance (given a task), we can change the *Reward* function as often as needed. This is true, because only the acquisition function needs an actual reward to select a new test point. We suggest updating or selecting the *Reward* function at each iteration of M-BOA.

For instance, if we consider the previous mobile robot example, at each iteration a planner algorithm chooses the next best point to reach. This point can then be used by the RSL in order to update the *Reward* function so that it outputs the Euclidean distance between the point selected by the planner and the prediction of the GPs.

C. Semi-Episodic Learning Algorithm

Using the two proposed additions, we can now have a non purely episodic version of the IT&E algorithm. We call it **Semi-Episodic Learning Algorithm** (SELA). The pseudo-code is shown in Algorithm 2.

Algorithm 2 Semi-Episodic Learning Algorithm

procedure SELA
  Before mission (in simulation with intact robot):
    Create Behavior-Performance Map using MAP-Elites
  while in mission do
    if significant performance drop **then**
      Adaptation-Step (using **SELA-ADAPT**)
procedure SELA-ADAPT
  $\forall \boldsymbol{x} \in map$:
    $p(f(\boldsymbol{x})|\boldsymbol{x}) = \mathcal{N}(\mathbf{P}(\boldsymbol{x}), k(\boldsymbol{x}, \boldsymbol{x}))$
  while stopping criteria not met **do**
    Update *Reward* function
    $\boldsymbol{x_{t+1}} = argmax_{\boldsymbol{x}} u(Reward(GPs(\boldsymbol{x}))|D_{1:t})$
    $\boldsymbol{Y_{t+1}} = afun(execute\_behavior(\boldsymbol{x_{t+1}}))$
    $D_{1:t+1} = \{D_{1:t}, (\boldsymbol{x_{t+1}}, \boldsymbol{Y_{t+1}})\}$
    Update GPs

IV. Preliminary Experiments

A. Toy Simulation

As a toy example, we consider the mobile robot example introduced previously. This mobile robot is a point (no dimensions, no orientation) and can take a 0.1-long step in any direction. We represent each atomic behavior by a scalar value, θ: the direction of the corresponding move. This environment was inspired by *Engel et al.* [15]. The task of the robot is to reach a target point despite some damage.

Because the example is too simple, but also to show the effectiveness of our method without relying on simulated data, we did not generate any behavior-performance map. We used the exact model of the intact robot as the mean function. Also, for the GR, we used the (x, y) relative end position of each behavior, for the *Reward* function the Euclidean distance between the next target and the prediction of the GPs and for the reward selection layer an A* path planner.

To evaluate our technique we used the following two control experiments:

- learn the model of the robot (using GPs) via random babbling and then use it to complete the task;
- solve the problem with the classic IT&E approach: we first learn with IT&E how to walk in 4 major directions (up, down, right, left) and then use these behaviors to reach the target.

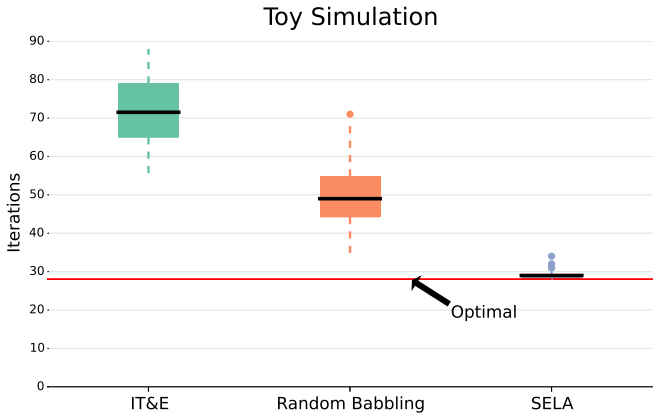


Fig. 2: **Toy Simulation Evaluation**. Comparison between the baseline approaches and **SELA** for the toy simulation. For both of the baseline approaches, we measure the number of iterations required to learn and the number of steps that they take to complete the task. We ran 50 replicates of each approach.

We ran 50 replicates of each approach for the scenario: "Reach the target point $(2.0, 2.0)$ starting from the origin despite a 0.5 radians angle offset in the range direction $\theta > 0$". To make the task a little more realistic we added a small Gaussian noise ($\mu = 0, \sigma^2 = 0.01$) to the position observations. Figure 2 shows the resulting performance (number of atomic behaviors taken to reach the target) for the different approaches. Our algorithm is able to reach the target with almost the optimal number of steps (i.e. if we perfectly knew the model), that is in much fewer steps than the other approaches.

B. 6-Legged Simulated Robot locomotion task

As a more realistic example, we consider a simulated 6-legged (hexapod) robot moving in space with the same task as in the toy simulation. See Figure 1b for the scenario and [8] for more details on the simulated hexapod. We evolved different atomic behaviors using the MAP-Elites algorithm with an 8D behavior descriptor (2 dimensions for space diversity + 6 dimensions for walking diversity), inspired by [16], [12]. The number of atomic behaviors evolved were approximately *1 million*. We used this behavior-performance map as the mean function. All the other parameters were the same as in the toy simulation experiment.

To evaluate our technique we used similar control experiments as in the toy simulation experiment:

- IT&E variant #1: we learn the outcome of the atomic behaviors (using GPs) via selecting the most uncertain behavior for $N = 15$ iterations. This can be considered as a uniform sampling of the behavior space. We then use what we learned to reach the target.
- IT&E variant #2: we first learn with IT&E how to walk in 4 major directions (forward, backward, turn cw, turn ccw) and then use these behaviors to reach the target.

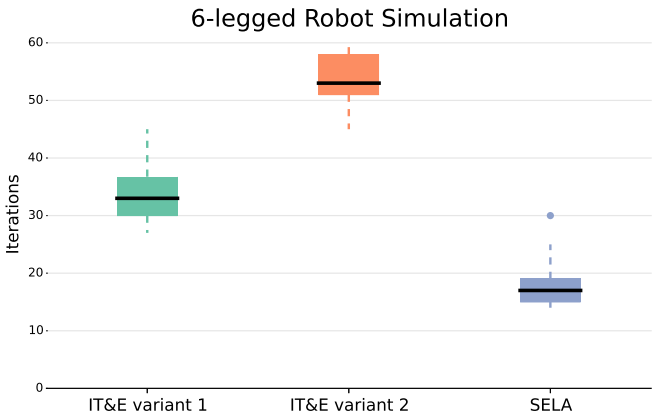


Fig. 3: **6-legged Robot Simulation Evaluation**: Comparison between the baseline approaches and **SELA** for the 6-legged robot simulation. For both of the baseline approaches, we measure the number of the iterations required to learn and the number of steps that they take to complete the task. We ran 50 replicates of each approach.

We ran 50 replicates of each approach for the scenario: "Reach the target point $(2.0, 2.0)$ despite the middle right leg being removed". We, also, added a small Gaussian noise $(\mu = 0, \sigma^2 = 0.01)$ to the position observations. Figure 3 shows the resulting performance (number of atomic behaviors taken to reach the target) for the different approaches. Our algorithm is able to find solutions in fewer steps than the other approaches.

C. 6-Legged Robot locomotion task

We, also, applied our technique on a real 6-legged robot. Preliminary experiments show promising results[1].

V. Conclusion and Future Work

We have introduced a semi-episodic learning scheme for robot damage recovery and a novel algorithm in this direction: **Semi-Episodic Learning Algorithm**. The intuition behind this scheme is that the robot can learn in a *data-efficient* way how to compensate for damages *while completing its task(s)*. This is achieved by (1) shrinking the search space, using simulated or computed data as prior knowledge, and by (2) using a *generic reward* of the outcome of the atomic behaviors of the robot instead of their performance given a task.

[1] https://www.youtube.com/watch?v=Gpf5h07pJFA

Future work includes performing more experiments with the real robot as well as experiments with different robots. In addition, BO can be replaced by other techniques that scale better. What is more, we used a naive *reward selection layer*, but more efficient/sophisticated methods can be used. We are currently investigating in this direction. Additionally, theoretical guarantees and analysis should be investigated in detail. Overall, this work is a first step towards semi-episodic and life-long learning for robot damage recovery.

Appendix

For all experiments the following parameters were used:
Error threshold for reaching goal: $\epsilon_{goal} = 0.1$

A. BO with GPs

Acquistion function: UCB with $\alpha = 0.05$
Kernel: Exponential kernel with $\sigma = 0.1$
GP noise: $\sigma^2_{noise} = 0.001$
Max iterations: $N = 10$ (Toy), $N = 15$ (Hexapod)

B. Learning with random babbling

Error threshold: $\epsilon_{model} = 0.01$
Max iterations: $N = 15$

Acknowledgments

This work was supported by the **ERC project "ResiBots"** (*grant agreement No 637972*), funded by the *European Research Council*.

References

[1] K. Nagatani, S. Kiribayashi, Y. Okada, K. Otake, K. Yoshida, S. Tadokoro, T. Nishimura, T. Yoshida, E. Koyanagi, M. Fukushima, *et al.*, "Emergency response to the nuclear accident at the Fukushima Daiichi Nuclear Power Plants using mobile rescue robots," *Journal of Field Robotics*, vol. 30, no. 1, pp. 44–63, 2013.

[2] M. Devy, R. Chatila, P. Fillatreau, S. Lacroix, and F. Nashashibi, "On autonomous navigation in a natural environment," *Robotics and Autonomous Systems*, vol. 16, no. 1, pp. 5 – 16, 1995.

[3] J. Carlson, R. R. Murphy, and A. Nelson, "Follow-up analysis of mobile robot failures," in *ICRA*. IEEE, 2004.

[4] J. Carlson and R. R. Murphy, "How UGVs physically fail in the field," *IEEE Transactions on Robotics*, vol. 21, no. 3, pp. 423–437, 2005.

[5] V. Verma, G. Gordon, R. Simmons, and S. Thrun, "Real-time fault diagnosis," *Robotics & Automation Magazine, IEEE*, vol. 11, no. 2, pp. 56–66, 2004.

[6] S. R. Ahmadzadeh, M. Leonetti, A. Carrera, M. Carreras, P. Kormushev, and D. G. Caldwell, "Online discovery of AUV control policies to overcome thruster failures," in *ICRA*. IEEE, 2014.

[7] M. S. Erden and K. Leblebicioğlu, "Free gait generation with reinforcement learning for a six-legged robot," *Robotics and Autonomous Systems*, vol. 56, no. 3, pp. 199–212, 2008.

[8] A. Cully, J. Clune, D. Tarapore, and J.-B. Mouret, "Robots that can adapt like animals," *Nature*, vol. 521, no. 7553, pp. 503–507, 2015.

[9] R. S. Sutton and A. G. Barto, *Reinforcement learning: An introduction*. MIT press, 1998.

[10] M. P. Deisenroth and C. E. Rasmussen, "PILCO: A model-based and data-efficient approach to policy search," in *ICML*, 2011.

[11] E. Brochu, V. M. Cora, and N. De Freitas, "A tutorial on Bayesian optimization of expensive cost functions, with application to active user modeling and hierarchical reinforcement learning," *arXiv preprint arXiv:1012.2599*, 2010.

[12] A. Cully, "Creative adaptation through learning," Ph.D. dissertation, Université Pierre et Marie Curie, 2015.

[13] J. Mockus, *Bayesian approach to global optimization: theory and applications*. Kluwer Academic, 2013.

[14] J.-B. Mouret and J. Clune, "Illuminating search spaces by mapping elites," *arXiv preprint arXiv:1504.04909*, 2015.

[15] Y. Engel, S. Mannor, and R. Meir, "Reinforcement learning with Gaussian processes," in *ICML*. ACM, 2005.

[16] A. Cully and J.-B. Mouret, "Behavioral repertoire learning in robotics," in *GECCO*. ACM, 2013.