Robotics and Autonomous Systems 100 (2018) 236–250

Contents lists available at ScienceDirect

Robotics and Autonomous Systems

journal homepage: www.elsevier.com/locate/robot

Reset-free Trial-and-Error Learning for Robot Damage Recovery

Konstantinos Chatzilygeroudis, Vassilis Vassiliades, Jean-Baptiste Mouret *

Inria, Villers-lès-Nancy, F-54600, France
CNRS, Loria, UMR 7503, Vandœuvre-lès-Nancy, F-54500, France
Université de Lorraine, Loria, UMR 7503, Vandœuvre-lès-Nancy, F-54500, France

HIGHLIGHTS

- RTE makes it possible for complex robots to quickly recover from damage.
- It is a trial-and-error learning approach that considers the environment.
- It does not require the robot to be reset to the same state after each trial.
- It breaks the complexity by pre-generating hundreds of possible behaviors.
- It allowed a damaged 6-legged robot to learn to walk in an arena with obstacles.

ARTICLE INFO

Article history:
Received 15 April 2017
Received in revised form 13 October 2017
Accepted 21 November 2017
Available online 2 December 2017

Keywords:
Robot damage recovery
Autonomous systems
Robotics
Trial-and-Error learning
Reinforcement learning

ABSTRACT

The high probability of hardware failures prevents many advanced robots (e.g., legged robots) from being confidently deployed in real-world situations (e.g., post-disaster rescue). Instead of attempting to diagnose the failures, robots could adapt by trial-and-error in order to be able to complete their tasks. In this situation, damage recovery can be seen as a Reinforcement Learning (RL) problem. However, the best RL algorithms for robotics require the robot and the environment to be reset to an initial state after each episode, that is, the robot is not learning autonomously. In addition, most of the RL methods for robotics do not scale well with complex robots (e.g., walking robots) and either cannot be used at all or take too long to converge to a solution (e.g., hours of learning). In this paper, we introduce a novel learning algorithm called "Reset-free Trial-and-Error" (RTE) that (1) breaks the complexity by pre-generating hundreds of possible behaviors with a dynamics simulator of the intact robot, and (2) allows complex robots to quickly recover from damage while completing their tasks and taking the environment into account. We evaluate our algorithm on a simulated wheeled robot, a simulated six-legged robot, and a real six-legged walking robot that are damaged in several ways (e.g., a missing leg, a shortened leg, faulty motor, etc.) and whose objective is to reach a sequence of targets in an arena. Our experiments show that the robots can recover most of their locomotion abilities in an environment with obstacles, and without any human intervention.

1. Introduction

During the recent DARPA Robotics Challenge (2015), many robots had to be "rescued" by humans because of hardware failures [1], which is paradoxical for robots that were designed to operate in environments that are too risky for humans. While these robots could certainly have been more robust and some falls prevented, even the best robots will encounter unforeseen situations: hardware failures will always be a possibility, especially with highly complex robots in complex environments [2]. For instance, C. Atkeson et al. report that the Atlas robot they used in the DARPA Robotics challenge had a "mean time between failures of hours or, at most, days" [1,3].

The traditional method for damage recovery is to first diagnose the failure, then update the plans to bypass it [4–6]. Nevertheless, conceptually, the probability of failing grows exponentially with the complexity of the robot (e.g., a Roomba vs a humanoid) and the environment (e.g., an empty arena vs a post-earthquake building); accurate diagnosis, therefore, becomes much more challenging and requires many more internal sensors, which, in turn, increase the complexity and the cost of robots.

To overcome these challenges, robots can avoid the diagnosis step and directly learn a compensatory behavior by trial-and-error [7–9]. In this case, damage recovery is a reinforcement learning (RL) problem in which the robot has to maximize its performance for the task at hand *in spite of being damaged* [10]. The most successful traditional RL methods typically learn an action-value

* Corresponding author.
E-mail addresses: konstantinos.chatzilygeroudis@inria.fr (K. Chatzilygeroudis), vassilis.vassiliades@inria.fr (V. Vassiliades), jean-baptiste.mouret@inria.fr (J.-B. Mouret).

https://doi.org/10.1016/j.robot.2017.11.010

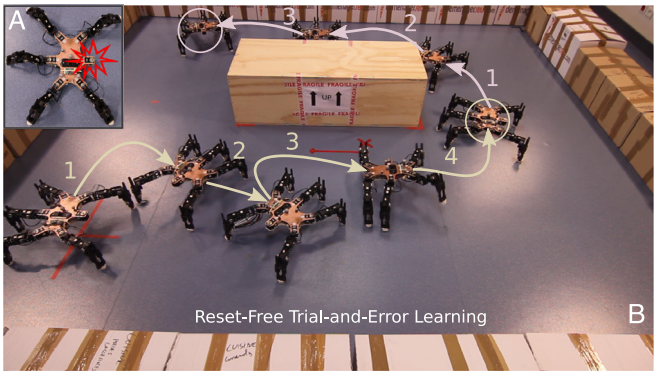

Fig. 1. A typical experiment with the Reset-free Trial-and-Error (RTE) algorithm. **A.** A 6-legged (hexapod) robot is damaged; i.e., missing a leg. **B.** The robot uses RTE to learn how to compensate while completing its task and taking into account the environment. As the robot moves, it improves its performance, i.e., it needs fewer episodes to reach the next target.

function that the agent consults to select the best action from each state (i.e., one that maximizes long-term reward) [11,12]. These methods work well in discrete action spaces (and even better when combined with discrete state spaces), but robots are typically controlled with continuous inputs and outputs (see [10,13] for detailed discussions on the issues of classic RL methods in robotics).

As a result, the most promising approaches to RL for robot control do not rely on value functions; instead, they are *policy search* methods that learn parameters of a controller, called the policy, that maps sensor inputs to joint positions/torque [13]. These methods make it possible to use policies that are well-suited for robot control such as dynamic movement primitives [14] or general-purpose neural networks [15]. In *direct* policy search, the algorithms view learning as an optimization problem that can be solved with gradient-based or black-box optimization algorithms [16]. As they are not modeling the robot itself, these algorithms scale well with the dimensionality of the state space. They still encounter difficulties, however, as the number of parameters which define a policy, and thus the dimensionality of the search space, increases [13]. In *model-based policy search*, the algorithms typically alternate between learning a model of the robot and learning a policy using the learned model [17,18]. As they optimize policies without interacting with the robot, these algorithms not only scale well with the number of parameters, but can also be very *data efficient*, requiring few trials on the robot itself to develop a policy. They do not scale well with the dimensionality of the state space, however, as the complexity of the dynamics tends to scale exponentially with the number of moving components.

In addition to scaling, another limitation of most of the current RL methods used in robotics is that after each trial, the robot needs to be reset to the same state [10,19]. While this reset is often not a problem for a manipulator, it prevents mobile robots (e.g., a stranded mobile manipulator or a legged robot) from exploiting this kind of algorithms to recover from damage in real-world situations. The robot cannot ignore its environment while learning, which is usually the case, as it may be further damaged if it makes a wrong decision. For example, if the robot is in front of a wall and needs to try a new way to move, it should not try to go forward, but it should select actions that would make it more likely to move backwards in order to avoid hitting the wall.

An ideal damage recovery algorithm should therefore (1) not need any reset between episodes, (2) scale well enough with respect to the dimensionality of the state/action space of the robot, so that it can be used for "complex" robots (e.g., legged robots) with the computing resources that are typically embedded in modern robots, and (3) explicitly take into account the environment. *The objective of the present paper is to introduce a reinforcement learning algorithm that fits these three requirements by exploiting specific features of the damage recovery problem.*

More precisely, we investigate a simplified scenario that captures these challenges: a waypoint-controlled robot is damaged in a way that is unknown to its operator (e.g., a leg is partially cut or a motor working at half speed); to get out of the building, the robot must recover its locomotion abilities so that it can reach the waypoints fixed by its operator. Our objective is to have the robot recover its locomotive abilities to the maximum extent possible in the shortest amount of time (Fig. 1). We assume that no diagnosis is available or that the diagnosis failed, either because the robot lacks the right sensor or because the damage is so out of the ordinary that it cannot be properly diagnosed. For simplicity, we also assume that the environment is known to the robot; we will discuss possible extensions of our approach when the environment is unknown in the discussion section.

Our first source of inspiration is the recently introduced Intelligent Trial and Error (IT&E) algorithm [7]. This algorithm is an episodic policy search algorithm that is specifically designed for damage recovery. It addresses the scaling challenge by assuming that some high-performing policies for the intact robot still work on the damaged robot. While this assumption does not always hold, empirical experiments show that it often holds with highly redundant robots (e.g., legged robots or humanoids) [7,8] because (1) there are often many ways to perform a task, and (2) the outcomes of behaviors that do not use the damaged parts are similar between the intact and the damaged robot. Using this assumption, IT&E searches for a diverse set of high-performing policies *before the mission* (offline), then performs the *online* search, that is, the adaptation to damage, by searching solely in this lower-dimensional set of pre-selected policies (using Bayesian optimization) [7]. As a result, most of the trials required for the policy search are transferred from the real damaged robot, which can perform only a few trials, to simulations with the intact robot, which can perform many more trials, especially on modern computing clusters. For instance, IT&E allows an 18-DOF hexapod robot to learn to walk after several injuries within a dozen episodes [7] and only two minutes of combined interaction and computation time.

A second source of inspiration is the recent AlphaGo algorithm that succeeded in beating the European and World champions in the game of Go [20]. Essentially, the authors use deep learning to pre-compute default policies and initial values for a Monte Carlo Tree Search (MCTS) [21] algorithm that plans (approximately) the best next action to take. We can draw an analogy in robotics and pre-compute actions or policies, learn the model of the robot on-line (the physical robot is damaged) and use MCTS to select the most promising action. Interestingly, MCTS can also take into account the uncertainty of the prediction of the model of the environment (e.g., when using Gaussian processes for models [22]). Unfortunately, it seems unrealistic to learn a probabilistic model of the full dynamics of a walking robot (like in [23]) within a few seconds (or minutes) of interaction time and the on-board computational power of a typical robot; more importantly, a probabilistic planner that would plan in the full controller space is even more computationally demanding.

Our main idea is to adapt the pre-computing part of IT&E, so that it can be used by a MCTS-based planner to select the next trial, in place of the Bayesian optimization used in IT&E. In addition, we utilize a probabilistic model to learn how to correct the outcome of each action on the damaged robot and use the MCTS-based planner in a similar way as in AlphaGo [20] and the TEXPLORE algorithm [23], but also incorporating the uncertainty of the model prediction in the search. This allows us to propose a trial-and-error learning algorithm for damage recovery that can work on a real hexapod robot, within reasonable computation time (less than