Neurocomputing 271 (2018) 95–103

Contents lists available at ScienceDirect

Neurocomputing

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

Making physical proofs of concept of reinforcement learning control in single robot hose transport task complete

Jose Manuel Lopez-Guede[a,b,*], Julian Estevez[b,c], Asier Garmendia[b,c], Manuel Graña[b,d]

[a] Department of Systems Engineering and Automatic Control, Faculty of Engineering of Vitoria, Basque Country University (UPV/EHU), Nieves Cano 12, Vitoria, 01006, Spain
[b] Computational Intelligence Group, The Basque Country University (UPV/EHU), Spain
[c] Department of Mechanical Engineering, Faculty of Engineering of Gipuzkoa, Basque Country University (UPV/EHU), Plaza Europa 1, San Sebastian, 20018, Spain
[d] Department of Computer Science and Artificial Intelligence, Faculty of Informatics, Basque Country University (UPV/EHU), Paseo Manuel de Lardizabal 1, San Sebastian, 20018, Spain

ARTICLE INFO

Article history:
Received 2 May 2016
Revised 13 September 2016
Accepted 29 January 2017
Available online 8 July 2017

Keywords:
Reinforcement learning
Linked multicomponent robotic systems
LMCRS
Hose transport
Proof of concept

ABSTRACT

This paper deals with the realization of physical proof of concept experiments in the paradigm of Linked Multi-Component Robotic Systems (LMCRS). The main objective is to demonstrate that the controllers learned through Reinforcement Learning (RL) algorithms with different state space formalizations and different spatial discretizations in a simulator are reliable in a real world configuration of the task of transporting a hose by a single robot. This one is a prototypical example of LMCRS task (extendable to much more complex tasks). We describe how the complete system has been designed and implemented. Two different previously learned RL controllers have been tested solving two different LMCRS control problems, using different state space modeling and discretization step in each case. The physical realizations validate previously published simulation based results, giving a strong argument in favor of the suitability of RL techniques to deal with LMCRS systems.

1. Introduction

The autonomous learning of optimal policies to carry out tasks with Linked Multi-Component Robotic Systems (LMCRS) [1] is an open research field. These systems are composed of a collection of autonomous robots linked by a flexible one-dimensional link introducing additional non-linearities and uncertainties when designing the control of the robots to accomplish a given task, often related to the non-rigid link itself. A paradigmatic task example is the transportation of a hose-like object by the robots (or only one robot in its simplest form). The first attempts to deal with this problem modeled the task as a cooperative control problem [2], however that too low level approach lacked the intended autonomous learning. The work reported in [3,4] gave a breakthrough contribution: a powerful modeling and simulation tool based on Geometrically Exact Dynamic Splines (GEDS) [5–7] to execute accurate simulations of LMCRS, allowing to assess the dynamical effects of the linking element (i.e., the hose) of the LMCRS on the active elements (i.e., the robots) [8]. Using that tool, the work focused on the autonomous learning of optimal policies by Reinforcement Learning (RL) [9] reformulating the task as a Markov Decision Process (MDP) [9–11]. Within the RL paradigm, the Q-Learning algorithm [9,12,13] was implemented because it allows the learner agent to learn from its experience with the environment, without any previous knowledge. Several works have been reported [14–19] using Q-Learning showing optimal results. Following the philosophy of using RL techniques that learn only from the experience, the TRQ-Learning algorithm was introduced in [20] reaching better results with boosted convergence. However, these results were always demonstrated in computer simulations.

The main objective of the paper is to report the execution of two proof of concept physical experiments of the task in the simplest instance of a LMCRS (with only one robot) to demonstrate that the computational simulation results are transferable to real physical world systems, even when the controllers have been obtained departing from different space state formalizations and different space discretization steps. To achieve this objective first it is necessary to build a complete physical system composed of the hose to transport, the robot that transports the hose, the RL controller that controls the execution of the task, the communications interface connecting the RL controller and the robot, and, finally, a perception system to monitor the evolution of the task execution.

* Corresponding author.
E-mail address: jm.lopez@ehu.es (J.M. Lopez-Guede).

http://dx.doi.org/10.1016/j.neucom.2017.01.110
0925-2312/

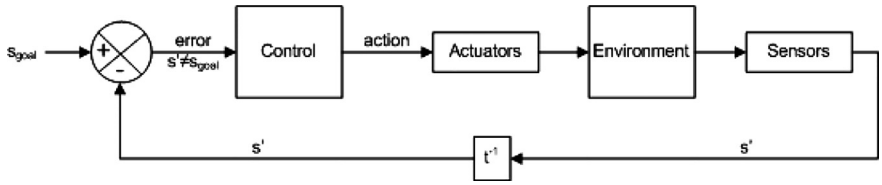

Fig. 1. Generic closed loop of the general system.

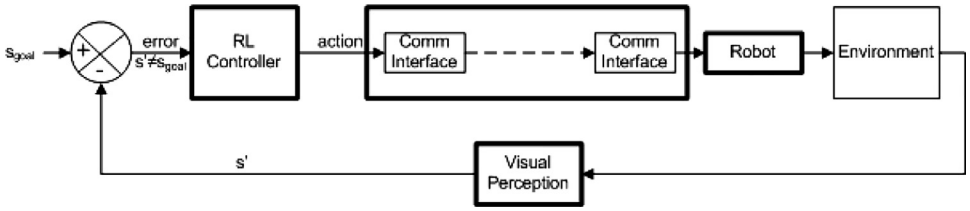

Fig. 2. Specific closed loop of the general system.

The paper is structured as follows. Section 2 details the design of the different parts of the system and their implementation to carry out the proof of concept experiments. The specific experimental design is given in Section 3, while Section 4 discusses the results obtained in the experimental realizations. Finally, Section 5 summarizes the obtained conclusions.

2. System design and implementation

In this section, we introduce the global system design and implementation by means of which the proof of concept has been carried out. First we describe the global control closed loop scheme through block diagrams. Later, the processes represented by the main blocks are explained with more detail.

2.1. Global scheme control

A generic representation of a closed loop control system is shown in Fig. 1. This abstract representation highlights that the perception and recognition of the global system goal is modeled through the concept of *state*. The global system tries to reach a desired state (s_{goal}) from the actual state (s') reached by the system. If these states do not match, the controller generates an action taking into account the actual system state. This action is carried out by the agent producing a change in the environment leading to a new state (s'), which is again perceived by the sensors.

Fig. 2 clarifies the previous generic schema showing the main specific modules composing the system in this proof of concept, whose boxes are highlighted with a thicker trace:

- A control module, built according to the optimal policy π learned by means of a RL algorithm.
- A communications module in charge of the transmission of the action to be executed by the robot by means of a wireless interface built for the occasion.
- The actuator that exerts the action on the environment, i.e., the robot.
- The perception module sensing and monitoring the environment after the actions have been executed to build the new system' s state representation.

2.2. RL control module

This module is the responsible for determining the optimal action to be carried out at each moment by the agent in order to reach the goal state (s_{goal}) knowing the actual state (s'). It is desirable to clarify that the RL controller has been previously learned by means of anyone of the available RL algorithms, and at this point it is executed without any retraining process. Therefore, at this stage the controller is in the *exploitation* phase, so that it is neither able to learn new knowledge nor to improve its performance. The previous required learning phase has been carried out using Q-Learning or TRQ-Learning algorithms, reaching a performance of 92% successful goals in the validation based on simulated processes. Anyway, that training phase is carefully described in previous papers [14–20] and it is beyond the scope of this paper.

Once that the agent has been trained, it is said that it implements a *policy* π (which is our best approximation to an optimal policy π^* to reach a given objective).

Although the training phase is not the main issue of this paper, it determines several aspects of the control module of this proof of concept. Since the learning algorithms have been the well known Q-Learning algorithm and the TRQ-Leaning [20], the knowledge of the agent has been implemented through a Q matrix representation, where a state-action value function is used. That Q matrix representation has as many rows as different states have been visited during the learning process, and as many different columns as actions are available to the agent to execute in the environment.

2.3. Robot manager

This part corresponds to the *actuator* block of the generic closed loop control system shown in Fig. 1. In this case, the actions that the agent can execute in the environment are movements of the SR1 robot. That robot model is quite simple and cheap, and we have adapted it to our purpose. We have divided the software functions that we have built into two groups according to their abstraction level.

2.3.1. Low level functions

This set of low abstraction level functions includes all the settings and functions that are necessary to carry out our purpose, lying below the interface that the robot offers to the RL controller. The first operation that we had to carry out was the calibration of the two servo motors to determine the width of pulses necessary to reach a given velocity. This operation was performed only once, but there are other supporting functions executed by demand of the high abstraction level functions:

- Managing and sending pulses to each servo motor.
- Serial port I2C management.
- Orientation measurement using the internal robot compass.