Applied Soft Computing 34 (2015) 678–692

Contents lists available at ScienceDirect

Applied Soft Computing

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

Motion planning with adaptive motion primitives for modular robots

Vojtěch Vonásek*, Martin Saska, Karel Košnar, Libor Přeučil

Department of Cybernetics, Faculty of Electrical Engineering, Czech Technical University in Prague, Technicka 2, Prague 6 166 27, Czech Republic

ARTICLE INFO

Article history:
Received 13 November 2013
Received in revised form 3 May 2015
Accepted 4 May 2015
Available online 14 May 2015

Keywords:
Motion planning
Modular robots
Particle swarm optimization

ABSTRACT

This paper presents a novel motion planning algorithm for modular robots moving in environments with diverse terrain conditions. This requires the planner to generate a suitable control signal for all actuators, which can be computationally intensive. To decrease the complexity of the planning task, the concept of motion primitives is used. The motion primitives generate simple motions like 'crawl-forward' or 'turn-left' and the motion planner constructs a plan using these primitives. To preserve the efficiency and robustness of the planner on varying terrains, a novel schema called RRT-AMP (Rapidly Exploring Random Trees with Adaptive Motion Primitives) for adapting the motion primitives is introduced. The adaptation procedure is integrated into the planning process, which allows the planner simultaneously to adapt the primitives and to use them to obtain the final plan. Besides adaptation in changing environments, RRT-AMP can adapt motion primitives if some module fails. The methods is experimentally verified with robots of different morphologies to show its adaptation and planning abilities in complex environments.

1. Introduction and motivation

Modular robots consist of many interconnected mechatronic modules. Depending on the hardware architecture, the modules can be equipped with various motion actuators, basic sensors, a communication bus and a simple processing unit [1]. In comparison to conventional fixed-shape robots, which are usually built for performing a specific task, modular robots are more flexible, as they can be reconfigured to various shapes according to the current application. Moreover, they are fault tolerant and able to self-repair. Modular robots may bring additional abilities in various applications, e.g. in search & rescue missions [2], space exploration [3] or object manipulation [4]. For example, in the search & rescue scenario, where the task is to find a victim after a disaster, a quadruped modular robot can be used for efficient motion over a complex terrain. When the robot needs to pass a narrow space, it can reconfigure to a snake-like robot. Beside the ability to form robots of various shapes, the advantage of modular robots stands in the possibility of changing failed modules, as they can be simply disconnected from the complex organism and replaced by new ones.

The individual modules can move using various actuators like wheels, belts [5] or even screw-drives [6]. The modular robots utilize two different concepts of movements: a joint-control and a self-reconfiguration. In the joint-control approach, which is considered in this paper, the locomotion of the robot is achieved by controlling the joints connecting the modules [7]. In the concept of self-reconfiguration, the locomotion is achieved by disconnection and reconnection of the modules [8–12]. This presents completely new type of motion which cannot be seen in the nature. It can be applied in situations where the joint-control approach cannot be used.

The joint-control locomotion can be realized using a Central Pattern Generator (CPG) [13], which produces periodic control signals for the actuators. Although the CPGs can be very effective to produce a desired gait, a global navigation over a large scene with complex obstacles and varying terrain usually require multiple gaits that are switched using a motion planner [14]. To ensure that the robot can move in environments with altering terrain conditions or even when some modules fail, the utilized locomotion gaits have to be adapted.

In this paper, we address the problem of finding a trajectory of a modular robot between different locations in a complex environment. To find the trajectory, the RRT-MP [14] algorithm is utilized. The original RRT-MP employs a vocabulary of fixed motion primitives. The main contribution of this paper is the novel approach for adaptation of the motion primitives, called RRT-AMP (RRT with Adaptive Motion Primitives). Beside the ability to design motion plans in environments with varying terrain types, the novel adaptation mechanism allows the robot to recover after a module fails.

---

* Corresponding author. Tel.: +420 22435 7384.
E-mail address: vonasek@labe.felk.cvut.cz (V. Vonásek).

http://dx.doi.org/10.1016/j.asoc.2015.05.002
1568-4946/

The paper is organized as follows. In the next section, the related work is summarized. The notation and description of motion primitives used in the rest of the paper is described in Section 3. Section 4 describes the method for optimization of motion primitives. The utilization of these primitives in the RRT-MP planner is described in Section 5 and the novel adaptation mechanism in Section 6. Section 7 describes implementation details. Experimental verification of the proposed methods is introduced in Section 8.

2. Related work

The concept of motion primitives, i.e., signals providing basic motions of a system, has been widely used in robotics. This approach is motivated by observations in nature, where complex behavior of invertebrates, and also vertebrates, including humans, can be achieved as a flexible combination of basic motion primitives [15,16]. In robotics, three types of motion primitives can be identified according to their preparation: (a) hand-coded primitives, (b) primitives learned by imitation; and (c) primitives learned through interaction with the environment.

Hand-coded primitives contain a predefined sequence of control signals of the robot. Primitives of this kind are usually used in industry, e.g. to program the movements of welding manipulators. The main disadvantage of this approach lies in the difficulty to embrace all possible situations and responses to them, which limits its application to tasks with low uncertainty. Moreover, the preparation of a hand-coded control become difficult for robots with many degrees of freedom (DOF) [17]. In the concept of learning by imitation (also referred to as learning by demonstration) a supervisor provides demonstration data, which are then processed using a machine learning method to automatically derive the desired motion primitives. Learning by imitation has been applied to many systems including arm manipulators [18], climbing robots [19], and cars [20]. In the concept of learning through interaction with the environment, a robot performs a given task without a supervisor. The most used method is Reinforcement Learning (RL), where a robot performs a given task while it adapts itself according to a received experience [21]. RL is an optimization technique aiming to find an optimal policy describing which action has to be performed in every state. RL has attracted research interest during the last decade resulting in many applications in mobile robotics [22], unmanned aerial vehicles [23] and humanoids [24]. As both states and actions need to be considered in RL, it can be memory consuming when applied to systems with many DOF, unless the state space is simplified [25].

The above mentioned methods are useful for finding motions for systems with few degrees of freedom. In modular robotic systems, which usually have many degrees of freedom, other approaches have been developed to generate locomotion. Locomotion generators can be based on gait control tables [26], role-based control [27], phase automata [28] or hormone-based control [29]. A widely-used concept for locomotion generation is based on Central Pattern Generators (CPGs). This concept is inspired by evidence from nature that motion is generated by coupled neuro-oscillators providing periodic signals [30]. CPGs are widely used in robotics to produce various rhythmic motions, e.g. walking or crawling [31]; we refer to [13] for a comprehensive review. The behavior of a CPG is controlled using several parameters, which need to be adjusted to achieve a desired locomotion. The parameters can be found e.g. using bio-inspired optimization tools [32–34,14,35].

Locomotion generators usually provide movements suitable for uniform environments, where the utilization of periodic actions is satisfactory. To move a robot to a distant location in a complex environment with obstacles or with diverse surfaces, more types of locomotion patterns are needed and a mechanism for switching between them is required.

This can be solved using motion planning, where the task is to find a feasible trajectory between an initial state and a goal state. The motion planning problem has been well studied in the literature; we refer to book [36] for a comprehensive survey of motion planning techniques. Modular robots are systems with many DOF, therefore sampling-based motion planning methods are most suitable candidates for computing feasible trajectories. The idea of sampling-based methods is to create a roadmap describing the configuration space of the robot. The configuration space is a space that covers all possible configurations of a robot, hence its dimension is equal to the number of DOFs. A path in the roadmap, which can be found using a graph-search method, then corresponds to a motion of the robot. Several sampling-based motion planning algorithms have been designed, such as Probabilistic Roadmaps [37], Rapidly Exploring Random Trees [38] and their variants. These methods have been utilized in many applications [39–41] including modular robotics [42,14,43].

Despite the ability of sampling-based methods to cope with many-DOF systems, locomotion planning remains a challenging task as many actuators need to be controlled by the planning method. To reduce the complexity of the planning process, the concept of motion primitives modeled using CPGs can be used [44,45,14]. As the CPGs provide control signals for the actuators, the signals do not need to be computed by a planner. Instead, the planners can derive a high-level plan utilizing the primitives. Therefore, the number of possible actions considered in the planner is reduced to the number of primitives that are utilized, which can significantly decrease the planning time.

The success of motion planning with motion primitives depends strongly on the quality of the primitives. Their performance can however decrease if the environment changes (e.g. when the robot enters an area with a different type of surface) or they may even become useless e.g. when some module fails. To succeed in such situations, the motion primitives need to be adapted. In recent work [25], an on-line RL-based adaptation strategy was proposed. To reduce the state-space, each module can perform a limited set of discrete actions, and these modules optimize their own policies independently. The reward, which is transmitted to all modules using a global message, is based on the achieved speed of the organism. This approach requires the robot to physically perform actions to learn a new control policy. While this may be useful in simple environments without obstacles, the presence of obstacles may lead to a situation, where the actual ‘moving’ policy, despite its quality, needs to be re-learned, when the robot gets stuck due to an obstacle. Once the robot is released, the previous ‘escaping’ policy cannot be used and a new ‘moving’ policy needs to be learned again.

In this work, we propose a novel adaptation schema for the motion primitives used in motion planning. In our concept, physical simulation is utilized to reason about possible actions in the environment. Therefore, knowledge about the environment in the form of a 3D map is required. Beside the ability of our method to adapt the motion primitives, they are simultaneously used to construct motion plans, which allows the robot to reach a distant location.

3. Preliminaries

The modular robots used in this work are CoSMO modules [6], which were developed in Symbrion EU FP7 project [5]. The CoSMO module is a cube 10 cm in size and ∼1.2 kg in weight equipped with four docking mechanisms. The modules provide 2D locomotion using a pair of screw-drives. 3D locomotion can be achieved using a main joint. The connected modules share communication