

Alexander Kazeka

Learning Generic Action Patterns for Mobile Robots

1. Problem Statement and Evaluation Criteria

The problem I'd like to address next semester is learning generic action patterns for mobile robots. Generic action patterns can be defined as nonspecific (flexible) and reusable sequences of simple actions. For example a command to rotate robotic wheels one revolution forward may be regarded as a simple action. Another simple action can be the command to turn robotic wheels 30 degrees to the right. Then, a generic action pattern can be the sequence turn-rotate-rotate, which allows the robot to move right and forward; a variation of this generic action pattern is the sequence turn-rotate-rotate-rotate, which moves the robot farther right and forward. The goal of this project is development and evaluation of a method which can be used to control simple actions and to learn relevant patterns of such actions. The patterns learned have to be generic to be useful in a wide range of different scenarios. Relevance of each action pattern is determined in the context of the scenario it's aimed to accomplish.

If successful, such method would make it possible for a robot to learn to control its simple actions and compose generic patterns in a laboratory setting, then when the robot is shipped, it would be able to re-compose the patterns it learned in the laboratory to accomplish new tasks. This would produce a system capable of solving novel problems in terms of simple actions it is familiar with, which is the main motivation for my research.

A valid solution to learning generic action patterns requires reusability in novel scenarios. The solution is envisioned to transfer competence in completing training tasks to their modified derivatives. Quantification of allowed problem modification is one of the evaluation criteria for the proposed solution. More specifically, the quantification should consider the change in error that the solution incurs when problem definition changes according to some measurable factor.

Another important evaluation criteria for the solution is its comparison to alternative methods. Although comparison among machine learning algorithms is often difficult, it is nonetheless an important test [1].

Yet another evaluation parameter for the proposed solution is its generated pattern complexity in terms of the number of simple actions involved. This will allow both evaluating the optimality of learned patterns and also the maximum complexity of the problems the solution can be applied to.

2. Project Objectives

The goal of this project is to develop a method for learning generic action patterns for mobile robots and to evaluate the proposed method with respect to three criteria: an error metric, performance compared to alternative methods, and learned pattern complexity. The problem itself is important because of a very simple observation - it is desirable to build robots which can be deployed in unfamiliar environments and can use knowledge about their capabilities (simple actions) to solve novel problems there.

3. Proposed Solution and Testing Environment

The proposed solution is inspired by [2] and [3]. Similarly to [2], this problem can be reformulated as learning a strategy to solve a puzzle. The main difference is that [2] did not address the question of controlling simple actions of robots. Tani et al., on the other hand, addressed the question of decomposing complex actions into their reusable sub-components, but did show how to combine simple actions into sequences necessary to solve novel problems.

It has been demonstrated that *liquid state machines* (LSMs) can be used to learn to control simple actions [4] and also learn sequences of simple actions [4, 5]. The learning, however, was not shown to generalize in the sense that LSM-based action controllers were not evaluated with respect to significant environment and problem changes. More specifically, only adaptation to training scenarios was reported [4, 5].

Controlling simple robot actions is accomplished by training LSM readouts filters corresponding to robot actuators. In [4], this approach was applied to controlling a standard two-joint robotic arm and an alternative biological model in two dimensional plane. Similarly to [3], authors used motion prediction generated by LSM to improve the performance of the controller.

The proposed solution applies *reinforcement learning* (RL) [2], a biologically sound learning algorithm, to a set of LSM readout filters to automatically learn to compose generic action patterns for a mobile robot. Although LSM and RL are biologically plausible computational models, such implications are beyond the scope of this project. An important detail about the proposed RL implementation, is that it will use LSM to approximate Q-function (LSM-RL, liquid state machine for reinforcement learning).

To evaluate the feasibility of the proposed approach, LSM-RL will be first applied to a mountain car problem. This problem is often used to test new RL methods. If the results are positive, further testing will be done using Unified System for Automation and Robot Simulation (USARSim).

USARSim is a high-fidelity simulation environment based on Unreal's 3D game engine. In addition to realistic robot and sensor models, it also provides a model of National Institute of Standards' (NIST) Reference Test Facility for Autonomous Mobile Robots for Urban Search and Rescue (USAR) [6]. The USAR tasks are concentrated on robot behaviors in rubble-filled environments which makes it an appropriate testbed for the proposed approach.

The following images from [6] demonstrate the correspondence between real and simulated environments in USARSim:



NIST Reference Arena



Simulated NIST Reference Arena

Communication with and feedback from USARSim robots is done via TCP/IP-based protocol documented in [6]. Although certain abstractions are also implemented, at the very basic level, robot actuator commands can be sent and sensory feedback received via TCP/IP. This information can be used to train LSM readout banks for simple actions, and input into LSM-RL to approximate the Q-function and learn simple action patterns.

In addition to high-fidelity testing, USARSim also provides a way to visualize the performance of the proposed method.

4. Expected Results

Expected results should demonstrate whether the proposed approach is viable. Personally, my expectations are high for this project and expected results include developing a LSM-RL (liquid state machine for reinforcement learning) which to my knowledge has not been described in literature. Another expected result is development and testing of a framework for learning action patterns in which simple actions are represented by LSM readout banks and LSM-RL is used to compose such sequences. This also, to my knowledge has not been described in literature. If the results are positive, this work can potentially be applied to develop a method to automatically solve novel problems in terms of known system capabilities (simple actions).

5. Milestones

February 2:

Mountain car experiments report: involves implementing LSM-RL (using publicly available Matlab LSM implementation), running mountain car experiments (and possibly comparing to tabular RL), and reporting the results.

February 16:

USARSim testing environment ready for the experiments: involves installation of Unreal Tournament 2004, USARSim, and required packages; also compiling publicly available C++ implementation of LSM and setting up TCP/IP communication between the LSM and the simulator.

March 2:

Simple USARSim experiments done: involves implementing LSM-RL in C++ and running experiments on simple tasks such as finding a lit area.

March 30:

More complex (for example search and rescue) experiments done, as well as comparison with alternative methods. Progress report.

April 15:

Rough draft of final report done.

April 27:

Final report turned-in.

6. References

[1] D. Whitley, J. P. Watson, A. Howe, L. Barbulescu. *Testing, evaluation and performance of optimization and learning systems*. In Proc. 5th Evolutionary / Adaptive Computing in Design and Manufacture conference, 2002.

[2] C. Anderson. *Learning the Solution to a Puzzle*. Ph.D. Dissertation, Chapter 7. 1986.

[3] J. Tani, R. Nishimoto, R. W. Paine. *Achieving "Organic Compositionality" through Self-organization: Review on Brain-inspired Robotics Experiments*. Neural Networks, vol. 21, 2008.

[4] P. Joshi, W. Maas. *Movement Generation with Circuits of Spiking Neurons*. Neural Computation, vol. 17, 2005.

[5] A. F. Morse. *Scale Invariant Associationism, Liquid State Machines, And Ontogenetic Learning In Robotics*. AAI Technical Report, In Proc. DevRob05 the Developmental Robotics Symposium, 2005.

[6] J. Wang. *USARSim. A Game-based Simulation of the NIST Reference Arenas*.