Policy Transfer in Mobile Robots using Neuro-Evolutionary Navigation

Matt Knudson
Carnegie Mellon University
matt.knudson@sv.cmu.edu

Kagan Tumer
Oregon State University
kagan.tumer@oregonstate.edu

ABSTRACT

In this paper, we first present a state/action representation that allows robots to learn good navigation policies, but also allows them to transfer the policy to new and more complex situations. In particular, we show how the evolved policies can transfer to situations with: (i) new tasks (different obstacle and target configurations and densities); and (ii) new sets of sensors (different resolution). Our results show that in all cases, policies evolved in simple environments and transferred to more complex situations outperform policies directly evolved in the complex situation both in terms of overall performance (up to 30%) and convergence speed (up to 90%).

Categories and Subject Descriptors

I.2.6 [**AI**]: Learning

Keywords

Neural Networks, Incremental Evolution, Navigation, Robotics

1. INTRODUCTION

Advances in mobile autonomous robots have provided solutions to complex tasks previously only considered achievable by humans. Such domains include planetary exploration and unmanned flight where autonomous navigation plays a key role in the success of the robots. One of the most popular ways to mitigate the complex nature of robotic tasks is to focus on simple tasks first and then transfer that knowledge into more complex tasks. Such an approach is termed *transfer learning* and is gaining more attention as of late [6]. The primary reason for the success of transfer learning is in the decomposition of a task into either stages [6] or into subtasks the knowledge of which is combined to accomplish the more complex task [5].

In this work, we explore a neuro-evolutionary approach whereby policies are incrementally evolved through tasks with increasing degrees of difficulty [4]. Neuro-evolutionary approaches fall in the *policy search* category where the aim is to search directly across policies. This search, interspersed with policy improvements through selection, allows for the discovery of new and robust navigation strategies. Neuro-evolutionary approaches have been successfully applied both to benchmark problems [2] and to real world control problems [1]. Often the policy (e.g. an artificial neural network) is simple in construction and therefore is inexpensive to modify and evaluate in practice, providing resource cost benefits as well [3].

GECCO'12 Companion, July 7–11, 2012, Philadelphia, PA, USA.
ACM 978-1-4503-1178-6/12/07.

2. ROBOT NAVIGATION

Robot navigation is a critical first step in many key mobile robot applications. Most current robot navigation algorithms are computation intensive. In cases where robots have limited resources, it is critical to focus on a simple mapping between incoming information and a navigation action. Regardless of resources, the robot must have the ability to choose safe and efficient paths through an environment to reach a specific destination. This includes the ability to avoid obstacles and maximize robot speed, while maintaining a level of robustness to inaccuracies and noise in sensor and actuator signals.

In this work, we selected a state space representation that encodes as much information as possible from the sensors, as simply as possible, using two state variables:

1. *Object distance:* Simulating ultra-sonic type sensors, for each vehicle relative potential path heading angle, a distance to the nearest object is provided.

2. *Destination Heading:* The difference between the potential path heading and the vehicle relative destination heading is provided. This indicates the correction required for the potential path.

To provide a space of actions that is as directly indicative of robot task needs as possible, but abstract enough to reduce the impact of non-determinism, the concept of *path quality* is introduced. This quality is calculated in varying ways dependent on the algorithm used, but represents the quality of a potential path for the robot to take next. In producing a distribution of quality for all possible paths at each time-step, the state of the environment is represented, and a path can be chosen either via the maximum quality, or by sampling to inject exploration behavior.

There are several ways to perform incremental evolution. For our case, when the population fitness has converged, the best network is chosen. The initial population for the new situation is then "seeded" with this network by creating mutants with the same procedure. The network remains in the population, but a random network is chosen for the first episode. The algorithm then progresses as usual, and ranking of the networks proceeds with the same equation, shown in Equation 1.

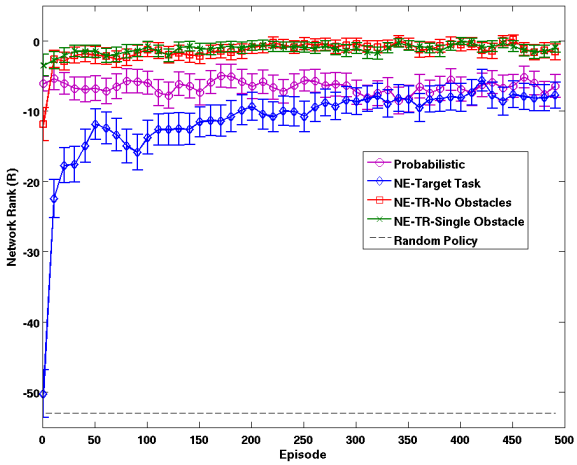

Figure 1: The results of evolution are shown for two source tasks; a) an open environment, and b) an environment with a single obstacle. The target task is an environment dense with obstacles. The probabilistic navigation and direct policy evolution on the target task are shown for comparison.

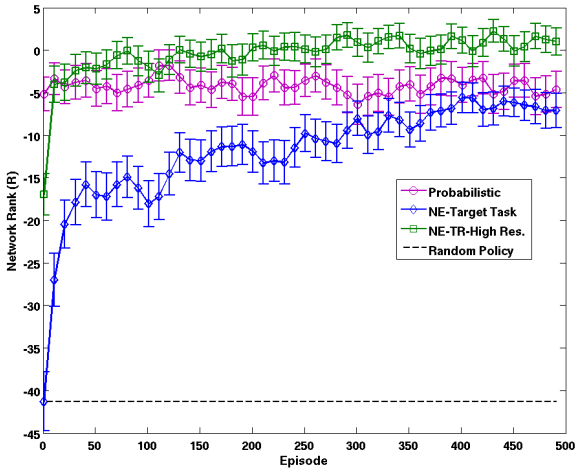

Figure 2: The results of evolution are shown for starting with 180 sectors and moving to 16. The probabilistic navigation and random initial population algorithms are shown for comparison.

The fitness function for the currently operating network at the end of an episode is given by:

$$R = \eta_1 \left(\frac{l}{\beta_l} \right) + \eta_2 \left(\frac{\tau_l}{\beta_\tau} \right) - \tau_c \qquad (1)$$

where τ_c is the number of time ticks the robot spends recovering from a collision with an obstacle or boundary wall, l and β_l are the actual total path length of the robot during the episode, and β_l is the best possible (straight line from start to destination). Similarly, τ_l is the *time* spent to execute that path, where β_τ is the shortest possible time. The ratios are inverted if l and τ_l becomes larger, keeping the range $[0.0, 1.0]$. η are tuning constants, currently 1.0 and 10.0 respectively. This fitness requires that the policy navigate the robot with the shortest possible path, as quickly as possible, and of course without hitting anything.

3. EXPERIMENTAL RESULTS

Figure 1 shows the results of transferring policies from two situations (initial task with no obstacles and initial task with one obstacle), along with results from the probabilistic navigation and a policy evolved directly for the dense obstacle environment. This process tests whether the agent can learn one portion of the state space first (path to destination) then learn to deal with obstacles. While seeding the population in an open environment (no obstacles) starts below -10, below a probabilistic algorithm, it still begins well above the random policy and climbs quickly to significantly exceed both. This is a significant result, indicating that the transfer of policies in this domain not only improves learning speed, but significantly improves overall performance. The population seeded from an environment with a single obstacle converges to a statistically similar performance, but does begin at higher performance, indicating that the policy did gain the ability to avoid obstacles in a specific situation.

Figure 2 show the results of transferring policies when the sensing capabilities change, dropping from 180 to 16 sectors. This experiment not only shows the potential for policy transfer, but also the robustness of the state/space representation. Indeed, in most cases, navigation algorithms are designed to exploit all possible sensing information, and suffer when that information is suddenly lost. These results show that the policy transfer outperforms both the probabilistic algorithm and policies evolved from scratch. This is a strong robustness result, showing the effectiveness of the state representation. While higher levels of information are beneficial and exploited, a dramatic change in the amount of that information does not significantly impact the algorithms performance, and it can easily adjust to the new situation.

4. REFERENCES

[1] A. K. Agogino and K. Tumer. Efficient evaluation functions for evolving coordination. *Evolutionary Computation*, 16(2):257–288, 2008.

[2] R. Chandra, M. Frean, and M. Zhang. An encoding scheme for cooperative coevolutionary feedforward neural networks. In *AI 2010: Advances in Artificial Intelligence*, volume 6464 of *Lecture Notes in Computer Science*. 2011.

[3] M. Knudson and K. Tumer. Adaptive navigation for autonomous robots. *Robotics and Autonomous Systems*, 59:410–420, June 2011.

[4] J.-B. Mouret and S. Doncieux. Incremental evolution of animats' behaviors as a multi-objective optimization. In *Proceedings of the 10th international conference on Simulation of Adaptive Behavior*, pages 210–219, 2008.

[5] M. E. Taylor, G. Kuhlmann, and P. Stone. Autonomous transfer for reinforcement learning. In *Autonomous Agents and Multiagent Systems (AAMAS)*, pages 283–290, 2008.

[6] M. E. Taylor, H. B. Suay, and S. Chernova. Integrating reinforcement learning with human demonstrations of varying ability. In *Autonomous Agents and Multiagent Systems (AAMAS)*, pages 617–624, 2011.