Evolving a Vision-Based Predator-and-Prey System for Two Robots with a Learning Classifier System

Noah W. Smith

Computer Science Department
Colby College
5830 Waterville, ME 04901
nwsmith@colby.edu

Abstract: This paper describes an experiment to evolve a predator-and-prey system where the primary input for each robot is a linear camera. The method of learning is a Learning Classifier System. It builds on similar work implemented with an evolved neural network, in which both predator and prey behaviors were learned based mostly upon camera input. It is designed for implementation on two Khepera model robots with standard K213 Vision Turrets from K-Team [1], but the actual learning will be done in simulation.

1 Introduction

Imagine watching a game of tag in a small square room played by two children, both wearing helmets with only a slit of vision available to them. One is chasing the other, but both are limited by their restricted vision and are feeling with their hands to make sure they do not run into a wall. My experiment models this behavior on a smaller scale, using two Khepera robots in a one-meter square arena. They play very poorly at first, but through the use of a Learning Classifier System a complex set of behaviors is evolved for both contestants. The limited vision can be captured with a linear camera, while feeling for walls will be done with IR sensors.

2 Background

Learning Classifier Systems [2] (LCSs) utilize reinforcement learning and a genetic algorithm to evolve a set of condition-action pairs, or classifiers. The LCS has little inherent knowledge of the exterior world – the environment is a black box from which it receives input, and, after selecting an appropriate classifier, returns an output. In a robot system, these outputs are commonly motor actions. The LCS used in this experiment have no internal memory nor any predictive ability.

The predator-and-prey task emerged in 1986 [3], and was originally designed with four predators and a single prey. The world was divided into a grid, and movement was restricted to north, south, east, and west, and in addition, full knowledge of the world was known to all five agents. Recent work has limited the knowledge available to each participant, and also moved away from the rigid grid format as well: the result being a partial-knowledge system with a full range of motion. In particular, Bauson & Ziemke [4] evolved an optimal field of vision for each robot in a two-robot predator-and-prey system where each robot was equipped with a linear camera. All behavior was learned by means of an evolved neural network. Interestingly, in this experiment, the prey developed a strategy based on constant movement, circling the arena despite whatever the predator was doing. However, the predator developed a highly reactive strategy, searching the center of the arena for the prey, then attempting to track and catch the prey once it could see it. In this experiment, both predator and prey will develop reactive behavior.

3 System Design

3.1 Hardware

The experiments described in this paper are designed for a 50 mm Khepera robot, with eight IR sensors and a linear camera. This camera has a resolution of 64x1, returning essentially a string of 64 8-bit pixels. For the sake of simplicity, while the camera can detect 256 levels of gray, the output is abstracted to black or white. The threshold will be calibrated for each robot at the beginning of each experiment in order to best distinguish between light and dark. The cameras sit on top of the robot, and are positioned level, so all one camera can see is composed of the white background walls, or the other camera, if it happens to be in its field of vision. The range of the cameras is 5-500 mm. The view angle is 36 degrees.

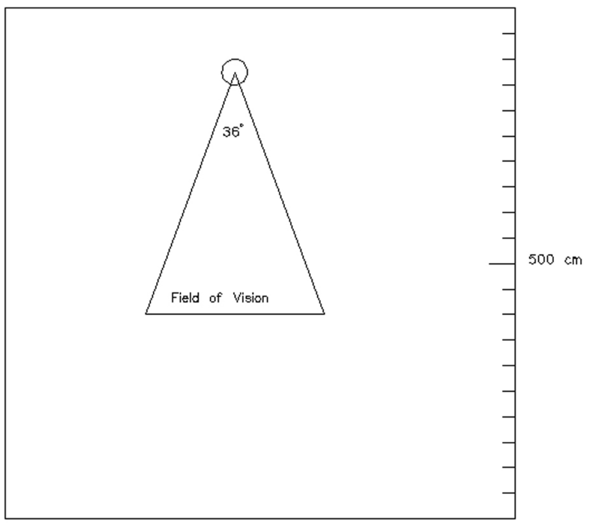

Fig. 1. A Khepera type robot in an arena (to scale). Note that almost the whole arena would be visible to a robot in the center of the area if it turned in place

3.2 Environment

The arena is a well-lit (no shadows) one by one meter square with white walls, with no internal obstacles, and the target behavior is that of predator and prey. Both the predator and the prey are relying solely on their cameras for tracking and following the opponent. The predator is considered to have won if it can touch the prey (detected by the IR sensors), while the prey is declared victor if it survives a preset time ceiling of 500 time steps. Simulation is performed in Webots[5], a 3D commercially available robot simulator available from Cyberbotics, but the code can and will be run on actual robots for demonstration purposes.

3.3 Classifier Structure

The LCS used to evolve the predator and the prey receives binary encodings of the state of the environment as input and returns motor actions to the robot as output. Based on the input string, a classifier is chosen, and its corresponding action dictates the output string. Sets of classifiers (each robot has exactly one set) are evolved as an entire string by the software package GENESIS [6].

Input. The input string is composed of the following elements: proximity to opponent, angle to opponent, and proximity of walls.

- The proximity to opponent can be gleaned from the number of pixels that are black (this should be one contiguous band representing the other robot, if it is in the field of vision). There are four different zones: more than 32 pixels, between 20 and 31, between 8 and 19, and less than 8. Four bits, one for each zone, represent this information.
- Angle to opponent can similarly be obtained from the camera data as the distance (in pixels) from the center of the field of vision to the center of the band. With 36 degrees in the field of vision, five zones seems appropriate: 12 to 18 degrees to the left, 6 to 12 degrees to the left, 6 to 12 degrees to the right, 12 to 18 degrees to the right, and the center, or 6 degrees left to 6 degrees right. This information fits in 5 bits.
- Proximity to the walls is calculated for each side of the robot, and is based on IR input. For each side (front, right, back, and left) of the robot, there exists three possibilities: a wall within 5 cm, between 5 and 15 cm, and farther than 15 cm. Four sets of 3 bits encode this information.

For example, the 21-bit input string 0010-00100-001-001-001-010 would mean that the robot was relatively close to its opponent, the opponent is in the central vision sector, and that walls are over 16 cm front, right, and behind, while there is a wall between 6 and 15 cm to the left of the current position.

Output. The output string specifies only two things: a number of motor steps for the left motor to perform before the next cycle, followed by the same information for the right motor. The number of steps can be specified in six bits: a sign bit (1 for forward, 0 for backward) followed by a five-bit binary number.

As another example, the 12-bit output string 1-01000-0-01001 would tell the left motor to move forward eight steps, while the right motor should move backward nine steps.

Classifiers. The classifiers are composed of a 21-bit condition followed by a 12-bit action, matching the input and output string lengths. The format for a condition is that a 1 in the input must match a 1 in the condition, while a 0 in the input can match a 0 or a 1 in the condition. For example, the condition 01100 may match either 01000 or 00100.

To illustrate a full classifier, the string 0110-01110-001-111-111-111 / 1-01000-1-01000 would mean that if the target is mid-range, and in one of the three central sectors, and there is no close wall straight ahead, that the robot should move both motors ahead by 8 steps. Note that the "111" sequence for the left, right, and back IR sensors is very vague – as a result, the whole classifier is very general.

Feedback. The feedback is applied after a match is over, with the positive feedback given to the classifiers of predators that catch their prey quickly as well as prey that evade the predator for a long period of time, while negative feedback is applied to classifiers of predators that fail to catch the prey as well as prey that are caught quickly. The degree of success is taken into consideration when selecting which classifier populations to evolve into the next generation. For example, a predator that caught its prey in a short amount of time is more likely to be selected as a parent for the next generation of predators than one which failed to catch the prey, or one which took a very long time.

4 Expected Results

While we have not yet completed our experiments, the following results are expected. Specific actions expected for the predator to learn include centering the opponent in its field of vision so as to face it directly, moving quickly once facing it, and trying to maximize the width of the band. Similarly, the prey would try to minimize the width of the band, while perhaps keeping it off to one side so as to turn away from it. We hope that both will develop fully reactive strategies relying strongly upon camera input. To encourage this, perhaps penalizing prey strategies which rely on staying out of sight of the predator will be necessary.

5 Future Work

An interesting modification to the environment would be to allow obstacles inside the arena, either blocking vision, IR, or both. This would greatly increase the learning space, and increase the relative real-world applications of the problem as well. In the same vein, allowing more than black and white vision in a world with shadows or objects of varying shades of color would add to the realism of the experiment.

References

1. K-Team: www.k-team.com .

2. Holland, J.H. Adaptation in natural and artificial systems. University of Michigan Press, Ann Arbor, MI (1975).

3. M. Benda, V. Jagannathan, and R. Dodhiawala. On optimal cooperation of knowledge sources - and empirical investigation. Technical Report BCS-G2010-28, Boeing Advanced Technology Center, Boeing Computer Services, Seattle, Washington, July 1986.

4. Bauson & Ziemke. Co-evolving Task-Dependent Visual Morphologies in Predator and Prey Experiments. In GECCO 2003, pp 458-469, Chicago, July 12-16 2003.

5. Webots: www.cyberbotics.com .

6. Grefenstette, John J. Technical Report CS-83-11. Computer Science Dept., Vanderbilt Univ., 1984.