An Autonomous Agent-Based Surveillance System

ARTICLE

2 AUTHORS, INCLUDING:

Yaser Khalifa
ABB

34 PUBLICATIONS   **106** CITATIONS

An Autonomous Agent-Based Surveillance System

Yaser Khalifa, and Ehi Okoene

Electrical and Computer Engineering Department, State University of New York at New Paltz
New Paltz, 75 South Manheim Boulevard,
New Paltz, NY 12561
yaserma@engr.newpaltz.edu

Abstract. In this paper, we present an evolutionary technique for an agent-based surveillance and target tracking system. The proposed system is composed of two stages. The first stage is the optimization of a network of ultrasonic sensors within a pre-specified coverage area. In phase two, the target tracking phase, sensors are activated based on need according to the tracking requirement of the moving object. Unneeded sensors are kept in a low-power stand-by state to optimize power usage and hence minimize human intervention for maintenance purpose. A centralized server communicates with sensor agents to power-up those needed for proper tracking of the moving object. The system is designed in C++ and simulated using MATLAB.

1 Introduction

The design and development of surveillance systems for military purposes is experiencing an increased growth. Such systems are intended to detect, locate and track moving targets, which include humans, trucks, tanks etc. The accuracy of these systems depends on a number of factors. Some of theses are 1) the integration of information from several sensors, 2) the optimization of the location of sensors in order to achieve complete coverage of the desired area, 3) the efficient use of the system's resources (e.g. maintenance of sensor's battery life for a longer life-span), and 4) reliable real-time communication in the midst of harsh environmental conditions and/or changes in the system. As a result of such a dynamic environment a lot of research has been done into the development of autonomous systems, capable of achieving the goals of surveillance (target detection, location, tracking etc) in the midst of constantly changing conditions. The work presented in this paper focuses on two aspects of surveillance: the optimization of the location of sensors (with the intent of maximizing the amount of area covered by the sensors) and target tracking. Genetic Algorithm is utilized to achieve an optimum solution for the location of sensors. The rest of this paper is organized as follows: Sections II provides the following information: a) a detailed description of the Optimization Program and how it is implemented in order to achieve goal of maximizing the area covered by these sensors, and b) a description of the techniques used to track objects that enter the surveillance area. The results obtained so far are presented in section III, while section IV gives a conclusion and future work.

2 Target Tracking

2.1 The Optimization Program

The Optimization Program utilizes Genetic Algorithm to maximize the area covered by sensors within a surveillance region. The sensors used (the SRF04 Ultra-Sonic Ranger) are directional sensors with a detection range from 3cm to 3m (see Fig. 1 for the beam pattern of the sensors). Within this program, the sensors can point in one of four directions: North, South, East or West.
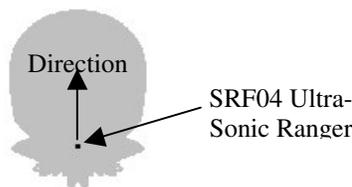
Fig. 1. Beam Pattern for the Ultra-Sonic Ranger

The area within which these sensors are placed is a rectangular grid, measuring 1000 cells by 1000 cells, with each cell representing an area of about 3cm by 3cm. Similar to any Genetic Algorithm, each solution is embedded in a chromosome, and a sample chromosome that is manipulated within this program is shown in Fig 2.
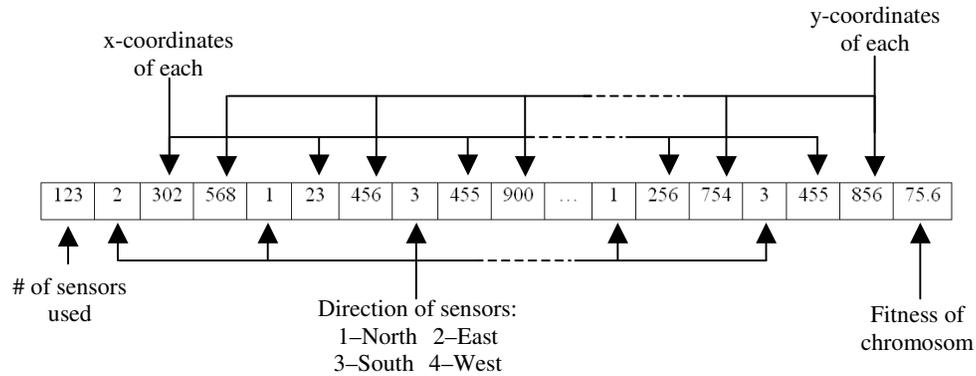
Fig. 2. Sample chromosome utilized within the Optimization Program

The Optimization Program is coded in MATLAB (Matrix Laboratory), which is a mathematical programming language and environment, optimized for matrix operations [1]; as such, the chromosome is represented in a row vector. There are a total of 371 cells within this chromosome and a maximum of 123 sensors are needed to cover the entire area. The first cell of this vector specifies how many sensors are used within the region. The second cell (and every third cell after this) specifies the direction of each sensor (1-North, 2-East, 3-South and 4-West). The third and fourth cells (and every third cell after each one) specify the Cartesian coordinates of each sensor, while the last cell indicates the fitness of the chromosome. The values that each cell within the chromosome can hold are given in Table 1. The first 34 sensors within the chromosome are used as boundary sensors; these are always turned-on, while the other sensors are internal sensors and are turned-on as needed.

Table 1. Values that can be held within each cell of the chromosome.

| Cell	Values cell can hold
1	0 to 123
2, 5, 8, … 365, 368	1, 2, 3 or 4
3, 6, 9, … 366, 369	50 to 950
4, 7, 10, … 367, 370	50 to 950
371	0 to 100

One unique fact about the Optimization Program is its ability to either generate a new population of chromosomes or work on a pre-existing population, where each population contains 100 chromosomes. The operators used within the optimization program are Reproduction, Crossover and Mutation. The reproduction operator is based on the Roulette Wheel technique. In addition, the four most-fit chromosomes are always reproduced to the next generations. The Crossover operator performs a variable-length two-point crossover; the points are chosen so as to only affect the internal sensors. For the Mutation operator, a mutation rate of 5% is utilized and only the coordinates of the internal sensors are mutated. This is done in such a way as to minimize the amount of change in its location. Lastly, the fitness of each chromosome is based of two factors: A) how much of the area is covered and B) how many sensors are used. The formula that defines the fitness is shown in Equation 1.

$$Fitness = A*B = \left[\left(\frac{x}{1000000}\right)*100\right]*\left[\left(\frac{35.05}{35}\right)-\left(\left[\frac{0.03}{35}\right]*z\right)\right]. \quad\quad (1)$$

where x is the number of cells covered by all the sensors (in terms of MATLAB) and z is the number of sensors used within the current chromosome.

2.2 The Tracker

The second part of this research is the tracking program. It is designed to track the movement of objects that enter the surveillance area. As mentioned in the previous section, there are two types of sensors within the surveillance region: the boundary sensors and the internal sensors (see Fig. 3). The boundary sensors are always turned-on and are able to detect objects as they enter the region, while the inner sensors (which are on stand-by) are turned-on as needed. Communication within the tracking system is carried out between the sensors and the central processing system; there is no communication amongst sensors.
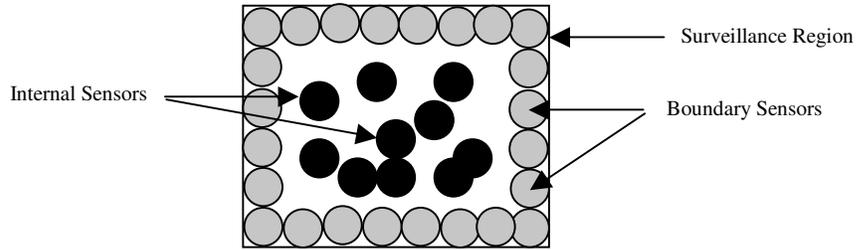

Fig. 3. The types of sensors within the surveillance region

Once the program has generated an optimal solution, the direction and distance of each sensor with respect to the others is generated. Next, all the boundary sensors are turned-on and checked to see if any of them has detected an object. When a boundary sensor detects an object, it then alerts the central processing system, which in turn alerts all sensors (boundary and internal) that are within a 300-cell radius of the detecting sensor of the presence of an object (Fig. 4). A 300-cell radius was chosen in order to ensure that all neighboring sensors (of the sensor that detected the object) are chosen. When the object is detected by another sensor (from the group of "alerted" sensors), its direction is determined with the help of geometry and all other currently alerted sensors are turned-off. With the presumed direction of the object determined, every sensor that lies within a radius of 300-cells and 180° of the presumed direction of the object are alerted and turned-on (the choice of 180° was to ensure the detection of the object if it traveled in a direction other than what was assumed, as shown in Fig. 5). The object is then detected and the whole process continues until the object exits the region.
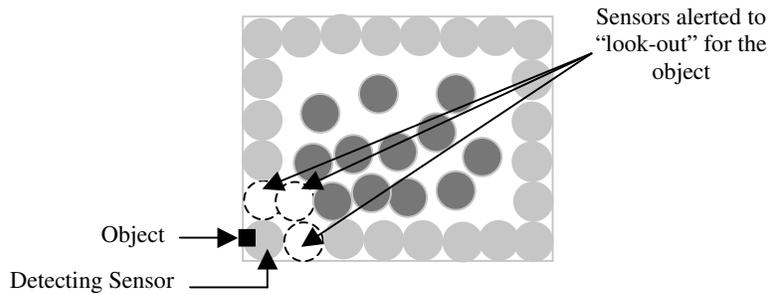
**Fig. 4.** Detection of an object by a boundary sensor.
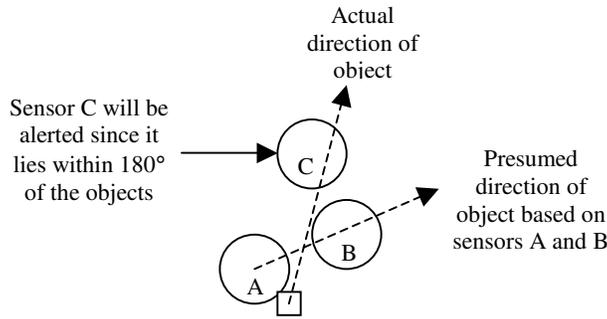
Fig. 5. Reason for the choice of 180°

3   Results

3.1   Results of the Optimization Program

The program was executed numerous times over a period of about two months, with an average of 250 generations produced per run. It took about an hour to generate a generation from its predecessor (i.e. through the process of reproduction, crossover and mutation). After producing about 5000 generations, the maximum fitness that was achieved was about 76.12 % (starting from 60.05%; see Fig. 6). This was achieved using 122 sensors. Fig. 7 gives a comparison of the amount of area covered by the sensors within two chromosomes (one with a fitness of 60.05% and the other with a fitness of 76.12%).
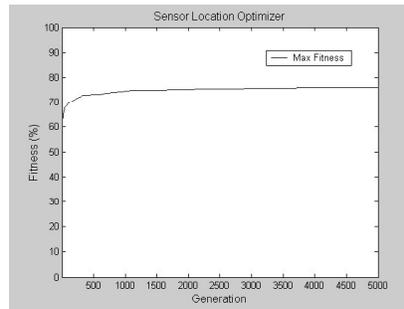

Fig. 6. Resulting maximum fitness of each population, from one generation to the next
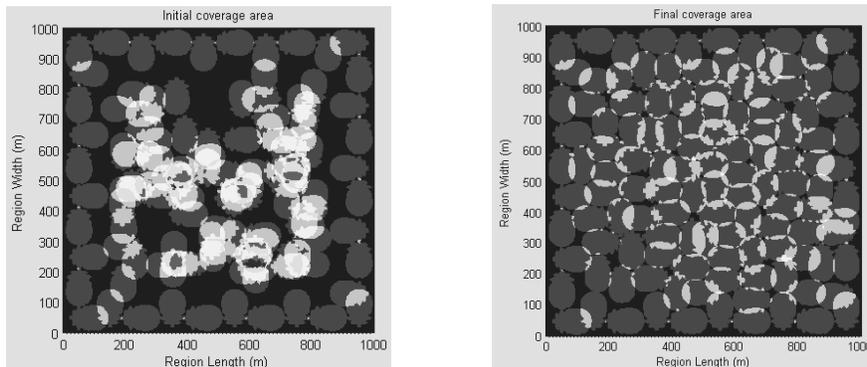
Fig. 7. A pictorial representation of the amount of area covered by two groups of sensors (encoded in two chromosomes), one with a fitness of **a)** 60.05% and the other **b)** 76.12%. The brighter regions indicate more overlap

3.2 Tracking an object

Once an optimum solution had been generated (based on the fact that there was no change in the maximum fitness of the population over a long period), the tracking system was executed and tracked the movement of an object from the time it entered the region till it left. The objects movement was designed in such a way that it moved in one of three directions (based on its entry-point). For example, if the object entered from the Southeast direction, then it would move towards the North or the East or a combination of both. Fig. 8 shows the actual path of an object an the sensors that detected its movement.
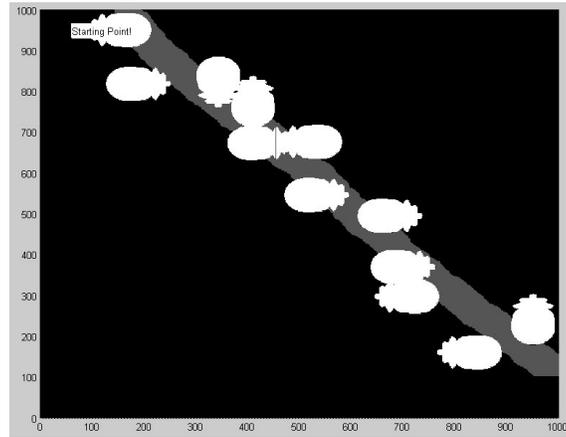
Fig. 8. Tracking of an object within the surveillance region. Note the starting point of the object

4 Conclusion and Future Work

An evolutionary based technique for sensor location optimization and target tracking has been presented. The system proposed is composed of two stages. In stage one the network of sensors and their locations are optimized in terms of area coverage and power consumption. In phase two, tracking of a moving target is achieved through inter agent communication and a centralized server to coordinate sensors activations. To save power, sensors within close proximity to the moving object only are powered. Other sensors stay in stand-by power saving mode until activated as target approaches them. Currently, we are investigating expanding the system to include different terrains and wooded areas. In addition to that, we are considering mobile sensors that would adapt to changes in environment, such as failing sensors within the network.

References

1. J. Harrison. *Matlab*. http://w3.arizona.edu/~cnl/matlab.htm.
2. C. P. Diehl, M. Saptharishi, J. B. Hampshire II and P. K. Khosla. *Collaborative Surveillance Using both Fixed and Mobile Unattended Ground Sensor Platforms*. SPIE's 13th Annual International Symposium on Aerospace/Defense Sensing Simulation and Controls, Orlando, FL, April 5-9, 1999.
3. L. Soh and C. Tsatsoulis. *Reflective Negotiating Agents for Real-Time Multi-sensor Target Tracking*. International Journal Conference on Artificial Intelligence, Seattle, Washington, pp. 1121-1127, 2001.
4. Azarbayejani and A. P. Pentland. *Recursive estimation of motion, structure and focal length*. IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. 17, No 6, pp. 562-575, June 1995.
5. J. Orwel, S. Massey, P. Remagnino, D. Greenhill and G. A. Jones. *A Multi-Agent Framework for Visual Surveillance*. IAPR International Conference on Image Analysis and Processing, Venice, pp. 1104-1107, September 27-29, 1999.
6. M. Piaggio and R. Zaccaria. *Autonomous Navigation Based on a Dynamic World Representation*. 2nd IEEE International Conference on Engineering of Complex Computer Systems, Montreal, Canada, pp. 152, October 21 – 25, 1996.

7. L. Buczak, Y. Jin, H. Darabi and Mohsen A. Jafari. *Genetic Algorithm Based Sensor Network Optimization for Target Tracking*. Intelligent Engineering Systems through Artificial Neural Networks, ASME Press, New York, Vol. 9, pp. 349-354, November 1999.

8. V. D. Gesù, B. Lenzitti, G. L. Bosco and D. Tegolo. *A Distributed Architecture for Autonomous Navigation of Robots*. Proceedings of the 5th IEEE International Workshop on Computer Architectures for Machine Perception, 2000.

9. J. Liu, M. A. Jafari and A. L Buczak. *Merging Redundant Sensors in Precision Tracking*. http://coewww.rutgers.edu/ie/research/working_paper/paper 03-102.pdf, 2004.