2008 International Symposium on Nonlinear Theory and its Applications
NOLTA'08, Budapest, Hungary, September 7-10, 2008

Sheepdog Problem Simulation by Using Cellular Automata

Yoshinobu Adachi and Hiroshi Igarashi

Faculty of Engineering, Tokyo Denki University
2-2, Nishiki-cho, Kanda, Chiyoda-ku, Tokyo 101-8457
Email: adachi@isl.d.dendai.ac.jp, igarashi@isl.d.dendai.ac.jp

Abstract—We propose the simulation framework for complex path planning problems with multiagent systems focuses on the sheepdog problem for handling distributed autonomous robot systems. The sheepdog problem is an extension of the pursuit problem for handling one prey robot and multiple predator robot. The sheepdog problem involves a more complex issue in which multiple dog robot chase and herd multiple sheep robot. We use the Boids model and cellular automata to model sheep flocking and chase and herd behavior for dog robots. We conduct experiments using a Sheepdog problem simulator and study cooperative behavior.

1. Introduction

Recently, technological improvements in robot hardware and related software application are increasing mobile robot utilization, making solutions for intelligent and efficient control of multiple mobile robot systems very hot issues. Simulation study and development for such systems is particularly relevant and important at this stage. The key issue is cooperative behavior between distributed autonomous robot systems.

The problem of cooperation between multiple robot systems is a subject of distributed artificial intelligence (DAI), which covers a vast, impressive number of studies [1][2].Examples include cooperative search, hunting, and capture problems. Cellular automata are used to model DAI problems and as a powerful problem solver for multiagent systems. The approach is applied to model the complex behavior of particle elements such as gases, liquid materials, and biological cells [3][4][5][6].

The Boids model is another very practical, useful tool used to model cooperative movement such as birds flocking, swarming, fish schooling, and other distributed behavior [7][8]. We use boids and cellular automata to model cooperative multiple mobile robot behavior in solving the sheepdog problem. In our previous work, we studied the sheepdog simulation performance with robot density and sheepdog robot diversity[9].

In this paper, we focus on dog robot sight range to compare sheepdog performance. We present the definitions underlying the sheepdog problem in the next section.

2. Problem Definition

The sheepdog problem is an extension of the pursuit problem that handles a single prey robot and multiple predator robots. In the sheepdog problem, multiple agents (dog robots) seek to herd multiple agents (sheep robots), each agent moving vertically or horizontally on a two-dimensional (2D) lattice shown in **Fig. 1**.

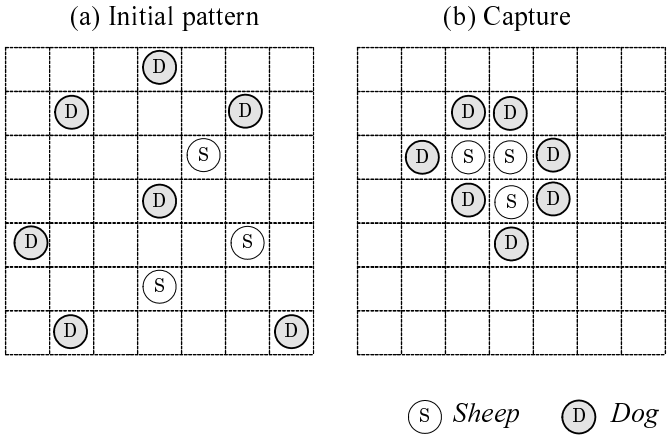

Figure 1: Definition of sheepdog problem. (a) Initial pattern, (b) Capture.

The field defined by the 2D lattice is an infinite toroidal plane. Robot moving to the right for enough over the right edge of the lattice appears at the opposite side up from the left edge.

Sheep completely surrounded by dogs are captured. The number of sheep and dogs ranges from 0 to 1000 and the filed size is 100x100 grids.

The sheepdog problem requires two different cooperative behavioral models: First, a flocking model for sheep. Second, a chase and herd (capture) model for dogs.

Robots detect the presence of neighbor robots within a specific range of sight, recognizing the robot type. Robots moves individually based on local information and neighborhood robot interaction.

Performance is measured by the number or ratio of captured sheep. The ratio is the proportion of captured to total number of sheep.

3. Robot Models

To simulate the sheepdog problem, we use the Boids algorithm and cellular automata to model sheep and dog robot behaviors, proposing a simulation framework based on a Boids algorithm within 2D cellular automata.

The basic Boids flocking model consists of three types of simple steering forces (separation, cohesion, and alignment) describing how individual robots maneuver based on the location and speed of neighborhood robots:

1. *Separation*: The boid avoids collisions with neighborhood boids.
2. *Cohesion*: The boid moves toward neighborhood boids.
3. *Alignment*: The boid steers itself in the same direction as neighborhood boids.

To implement the sheepdog problem simulator, we propose the following framework for controlling the sheep and sheepdog behavior with cellular automata:

- *Robot activity*: Robots move within a 2D lattice of n x n grids in a toroidal world.
- *Robot range of sight*: Robot sensing detects the location of other robots, the robot type, and relative speed within a certain 24-cell range of sight (**Fig. 2**). The robot detects other sheep and dog robots within the range of sight.
- *Robot action rules*: A robot in the lattice field has two parameters (eq. (1)). $R(t)$ represents status information of the robot at step time t. Parameter 1 is the x-y location in lattice as $P(t)$. Parameter 2 is the force that affects robot movement, $F(t)$.
- *Robot Movement*: The robot takes a step forward in the von Neumann neighborhood at step time $t + 1$. The von Neumann neighborhood consists of four direct neighbors, i.e., directly to the right, directly to the left, directly above, and directly below the robot. The cell of next step $P(t+1)$ is decided by force vector $F(t)$ of the robot (eq. (4)). Function $rand(n)$ returns a random number between 0 and n. If, for example, force vector $F(t)$ equals (4, 1), the robot moves one cell to the right with a probability of 80%, or one cell down with a probability of 20%.

Robot status is defined as in eq. (1):

$$R(t) = \{P(t) = (x, y), F(t) = (f_x, f_y)\} \tag{1}$$

Force at step time $t + 1$ is defined as in eqs. (2) and (3):

$$F(t + 1) = F_s(t) + F_d(t) + F(t) \tag{2}$$

where $F_s(t)$ is the force affected by the same type of robot and $F_d(t)$ is the force affected by a different type of robot.

$$\begin{aligned} F_s(t) &= a_s F_{s_{sep}}(t) + b_s F_{s_{coh}}(t) + c_s F_{s_{ali}}(t) \\ F_d(t) &= a_d F_{d_{sep}}(t) + b_d F_{d_{coh}}(t) + c_d F_{d_{ali}}(t) \end{aligned} \tag{3}$$

where $F_{s_{sep}}$, $F_{s_{coh}}$, and $F_{s_{ali}}$ are separation, cohesion, and alignment forces using the same type of robot. Parameters a_s, b_s, and c_s are the weights of each force. $F_{d_{sep}}$, $F_{d_{coh}}$, and $F_{d_{ali}}$ are separation, cohesion, and alignment forces using different type of robots. a_d, b_d, and c_d are the weights of each force.

$$P(t + 1) = \begin{cases} (x + sign(f_x), y) & when \quad d = 1 \\ (x, y + sign(f_y)) & when \quad d = 0 \end{cases} \tag{4}$$

$$rand(|f_x| + |f_y|) \le |f_x| \rightarrow d = 1$$
$$rand(|f_x| + |f_y|) > |f_x| \rightarrow d = 0$$

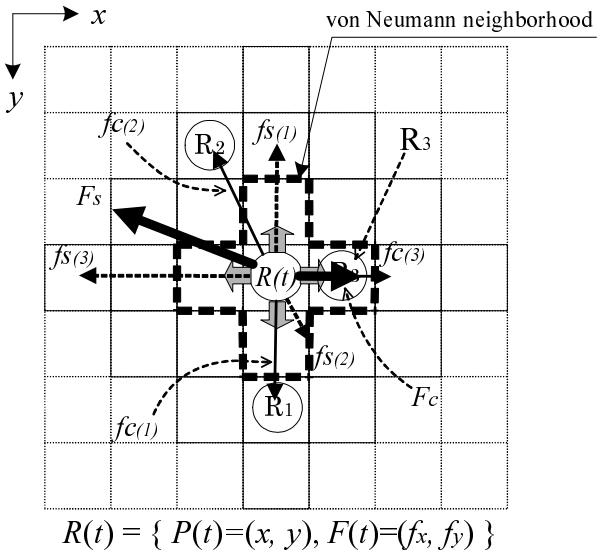

Figure 2: Robot range of sight with three Manhattan distance.

Separation force $F_s(t)$, cohesion force $F_c(t)$, and alignment force $F_a(t)$ at step in time t are defined by

$$F_s(t) = \sum_i f_s(i), \quad F_c(t) = \sum_i f_c(i), \quad F_a(t) = \sum_i F(i)$$

where $f_s(i)$ is separation force elements affected by robots, $f_c(i)$ is cohesion force elements affected by robots, $F(i)$ is one of force elements owned by a neighborhood robot, and i is the index of robots within the range of sight. Force F_a acts on the robot to steer it towards the average heading of other robots within the range of sight.

3.1. Sheep Robot

To implement sheep robot behavior, we set the following action rules for sheep robot: First, sheep robots execute the simulated flocking. Second, sheep robots attempt to

avoid colliding with dog robots. Sheep robot action rules based on $R(t)$ defined in the previous section, consist of the following forces:

1. *Separation*: The separation force $f_s(i)$ element is generated by a sheep robot or dog robot within the range of sight.
2. *Cohesion*: The cohesion force $f_c(i)$ element is generated by a sheep robot within the range of sight.
3. *Alignment*: The alignment force $F(i)$ element is generated by a sheep robot within the range of sight.

3.2. Dog Robot

To implement dog robot behavior, we set the following action rules for dog robots: First, dog robots have no separation force in relation to sheep robots. Second, dog robots develop strong cohesion in relation to sheep robots. Third, dog robots develop strong alignment force in relation to sheep robots.

Dog robot action rules based on $R(t)$ defined in the previous section, consist of the following forces:

1. *Separation*: The separation force $f_s(i)$ element is generated by a dog robot within the range of sight.
2. *Cohesion*: The cohesion force $f_c(i)$ element is generated by a sheep or a dog robot within the range of sight.
3. *Alignment*: The alignment force $F(i)$ element is generated by a sheep or a dog robot with in the range of sight.

Fig. 3 shows the status of the sheepdog cellular automata in progress. Dog robots have captured sheep robots forming a cluster.

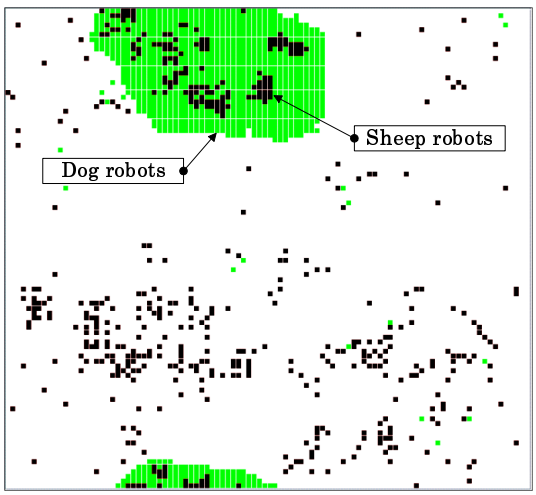

Figure 3: Dog robots cluster to herd sheep robots.

4. Dog Robot Sight Range Simulation

4.1. Robot Parameters

Simulation parameters are as follows: Number of dog robots is 1000, and number of sheep robots 500, the field size is 100x100 grids. Each dog robot has same robot parameters. Five different sight ranges of dog robots (Case 1 to 5), Manhattan distance 1, 2, 3, 4 and 5 were chosen to compare sheepdog simulation performance (**Fig. 4**).

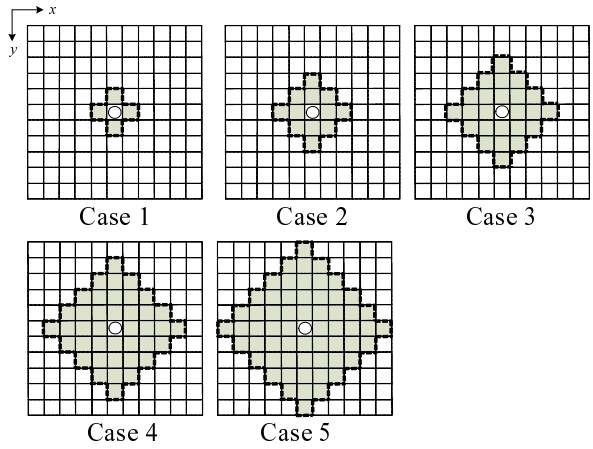

Figure 4: Sheepdog sight range patterns.

4.2. Simulation Results

In simulation results (**Fig. 5**), the x axis is the number of steps over time and the y axis is the number of captured sheep robots calculated by averaging the results of ten simulations.

Note the following: Case 4 and 5 shows the best performance and case 1 shows the worst. The overall result shows that the dog robot sight range area improves sheepdog performance.

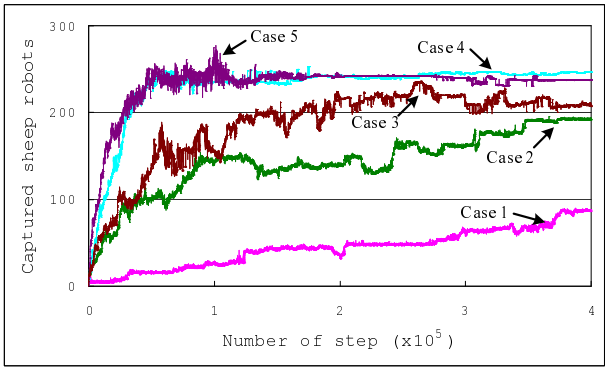

Figure 5: Captured sheep robot number with five dog robot sight range patterns.

Fig. 6 shows the simulation results with the x axis (log) is the dog robot sight range area and the y axis is the average number of captured sheep robots. In case 1, 2, 3 and 4, the number of captured sheep robots shows a linear increase by the increase of dog robot sight range area. In case 4 and 5, the number of captured sheep robots is almost same.

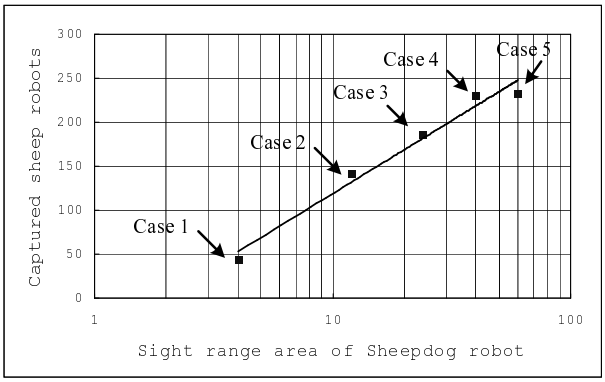

Figure 6: Captured sheep robot number and dog robot sight range.

Note the following: The robot density is 15%, sheep:dog ratio is 50%, and average number of robots within dog robot sight range is shown below.

Table 1: Dog robot sight range area and average number of robots within sight range.

Case	Dog robot sight range area	Average number of robots within sight range
1	4	0.6
2	12	1.8
3	24	3.6
4	40	6.0
5	60	9.0

5. Conclusion

In the simulation framework we have proposed for the sheepdog problem, Boids model and cellular automata realize two different behavior models. First, the sheep robots have the flocking behaviour. Secondly, the dog robots have the chasing and capturing behaviour.

Simulation results of dog robot sight ranges showed that sheepdog problem performance is affected by dog robot sight range. We chose the dog robot sight range from 1 to 5 Manhattan distance to compare sheepdog performance. Between 1 to 4 Manhattan distance, the number of captured sheep robots shows a linear increase. In 5 Manhattan distance, the number of captured sheep robots is almost the same as 4. It seems reasonable to conclude that the increase of dog robot sight range under particular sight range area, 40 in the simulation, improves sheepdog performance. The limitations of performance by sight range can be explained that excessive increase of average number of robots within dog robot sight range causes an inhibitory action against dog robot cooperative behavior.

In future work, we plan to proceed with this research targeting dynamic optimization of robot weighting parameters through sharing distributed knowledge, introducing other factors into robot models such as movement cost, and combinations with other algorithms for pursuit problems.

References

[1] H. M. Botee, E. Bonabeau, “Evloving And Colony Optimization”, Adv. Complex Systems (1998) 1, pp. 149-159

[2] K. Takahashi and M. Kakikura, ”Research on cooperative capture by multiple mobile robots - a proposition of cooperative capture strategies in the pursuit problem-”, Distributed Autonomous Robotic Systems, 2002, vol. 5, pp. 393-402

[3] U. Frisch, D. d’Humieres, B. Hasslacher, P. Lallemand, Y. Pomeau, and P. Rivet, “Lattice Gas Hydrodynamics in Two and Three Dimensions”, Complex Systems, 1, 1987, pp. 649-707

[4] G. W. Baxter, R. P. Behringer, “Cellular automata models of granular flow”, Physical Review A, Vol. 42, No 2, 1990, pp. 1017-1020

[5] M. S. Alber, M. A. Kiskowski, J. A. Glazier and Y. Jiang, “On Cellular Automaton Approaches to Modeling Biological Cells”, Mathematical Systems Theory in Biology, Communication, IMA 134, Springer-Verlag, New York, 2002, pp. 1-40

[6] T. Suzudo, “Spatial Pattern Formation in Asynchronous Cellular Automata with Mass Conservation”, Physica A, 2004 Vol. 343 C, pp.185-200

[7] C. W. Reynolds, “Flocks, Herds, and Schools: A Distributed Behavioral Model, in Computer Graphics”, SIGGRAPH 1987 Conference Proceedings, 21(4,) pp. 25-34

[8] W. J. Crowther, “Rule-based guidance for flight vehicle flocking”, submitted to the Journal of Guidance, Dynamics and Control, Sep. 2002

[9] Y. Adachi, M. Masayoshi, “Research on the Sheepdog Problem Using Cellular Automata”, *JACIII*, Vol.11, No.9, pp.1099–1106, 2007.