Proceedings of KIIS Spring Conference 2008 Vol. 18, No. 1.

침입자 추적을 위한 군집 로봇의 분산 이동 알고리즘에 관한 연구

Study on Diversified Moving Algorithm of Swarm Robots to Track a Invader

이희재, 양현창, 심귀보

Hea-Jae Lee, Hyun-Chang Yang and Kwee-Bo Sim

중앙대학교 전자전기공학부
(E-mail: kbsim@cau.ac.kr)

요 약

군집 로봇(swarm robots)의 경우 작업을 수행하기 위해서는 여러 로봇의 협동 작업이 필요하게 된다. 따라서 로봇이 주어진 환경을 인식하고 모델링 하는 것, 주어진 복잡한 작업을 분석하고 단순한 작업으로 나누는 것, 로봇에게 주어진 작업을 효과적으로 이행하는 것 등이 연구의 핵심이 되고 있다. 침입자 발견시 군집 로봇이 침입자를 효과적으로 추적하기 위해서는 다양한 경로를 통해 침입자가 이동할 수 있는 경로를 예상하고 분산 이동해야 한다. 본 논문에서는 알려진 맵에서 군집 로봇이 침입자를 효과적으로 추적하기 위한 분산 이동 방법을 제안한다.

Key Words : 군집 로봇, 이동 알고리즘, 침입자 추적, 분산 이동

1. 서 론

군집 로봇에 관한 연구는 지난 몇 년간 현대 로보틱스 분야에서 활발히 연구되고 있는 분야 중 하나이다. 군집 로봇의 협조 행동은 주어진 임무를 좀 더 빠르고 정확하게 하는데 목적이 있다. 이러한 연구는 달 기지 건설 [1], 물체 옮기기 [2], 조사 [3][4], 침입자 또는 타켓 포위[5][6] 등 다양한 분야에서 진행되었다. 이러한 군집 로봇들이 협조 행동을 제어 하는 것은 2가지의 제어로 나눌 수 있는데, 모델에 기반한 제어와 행동에 기반한 제어이다. 모델에 기반한 제어는 우리가 알고있는 정확한 모델링에 의한 로봇의 kinematics와 dynamics를 말한다. 하지만 행동에 기반한 제어는 우리는 정확한 모델을 알고 있지 못하며, 어떤 행동을 특정 알고리즘으로서 나타내는 것이다. 우리는 침입자/타켓을 포위하는 이런 로봇들의 행동을 제어하는 이동 알고리즘에 초점을 두고 있다.

이러한 임무를 수행하기 위해, 우리는 먼저 이동 로봇들을 구성하는 로봇 시스템의 local input들과 global output을 생각해야 한다. 여기서 local input들은 각각 로봇들의 위치가 되며, global output은 로봇들이 침입자/타켓을 포위하였을 때의 최종 위치가 된다. 이후 위에서 찾아낸 global output까지 로봇들의 최적 이동 경로를 전략적으로 계획하여 주는 것 또한 침입자/타켓을 포위 하는 데 중요한 일이라 할 수 있다.

본 논문에서는 군집 로봇들의 효과적인 침입자의 포위를 위하여 알려진 맵에서 침입자의 예상 이동 경로를 테이블화한 확률로서 각각 로봇들의 포위 위치를 얻고, 이 포위 위치까지 로봇들의 분산 이동 경로를 계획하는 알고리즘을 제안한다.

감사의 글 : 본 연구는 지식경제부의 성장동력기술개발사업인 집단 로봇 기술을 이용한 사회안전로봇 개발(세부과제: 로봇통제 및 환경기술개발)에 의해 수행되었습니다. 연구비 지원에 감사드립니다.

2. 군집 로봇 제어 시스템의 구조

우리는 알려진 2차원 맵에서 군집 로봇의 협조 행동을 통해 침입자를 포위하는 알고리즘을 위해서 그림 1과 같은 군집 로봇 제어 시스템 구조를 설계하였다. 먼저 우리는 군집 로봇은 모두 같은 능력을 가지고 있다고 가정하였다. 모든 로봇들은 먼저 자신의 현재 위치의 정보를 전송하여야한다. 이 전송된 맵 정보와 침입자의 위치를 가지고 Formation controller는 침

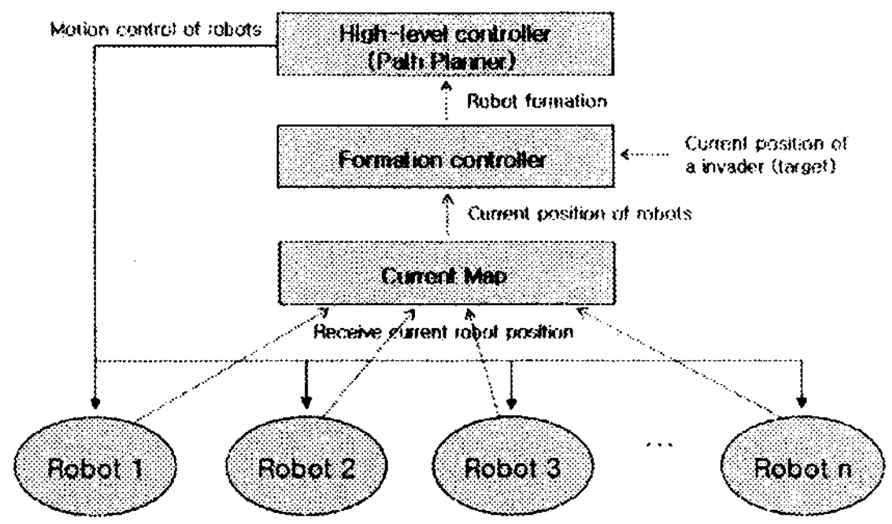

그림 1. 군집 로봇 제어 시스템의 구조

입자를 포위하는 로봇들의 최종 위치를 계산하며, 이 정보를 High-level controller(Path planner)에 전송한다. High-level controller는 현재 로봇의 위치 정보와 계산된 로봇의 포메이션을 가지고, 로봇의 이동을 제어 한다.

여기서 로봇이 전송하는 현재 로봇의 위치는 $[x_i, y_i]^T$라고 할 수 있으며, 알고리즘에 의해 전송되는 로봇의 다음 위치는 다음과 같다.

$$\begin{bmatrix} x_{i+1} \\ y_{i+1} \end{bmatrix} = \begin{bmatrix} x_i \\ y_i \end{bmatrix} + \begin{bmatrix} 1 \\ 0 \end{bmatrix} a + \begin{bmatrix} 0 \\ 1 \end{bmatrix} b \tag{1}$$

3. Formation Controller의 설계

이번 3장에서는 Formation controller에 적용할 알고리즘을 제안한다. 그림 2와 같은 이차원 맵에서 침입자가 발견 되었을 때, 우리는 침입자가 이동할 방향을 확률로서 나타내기 위해 문이나 창문 등 탈출이 가능한 장소에 다음과 같은 초기 적합도를 주었다.

Door = 5
if *there is a stair outside the door*
    Door += 1
Window = 1
if *the window is on the first floor*
    Window += 7

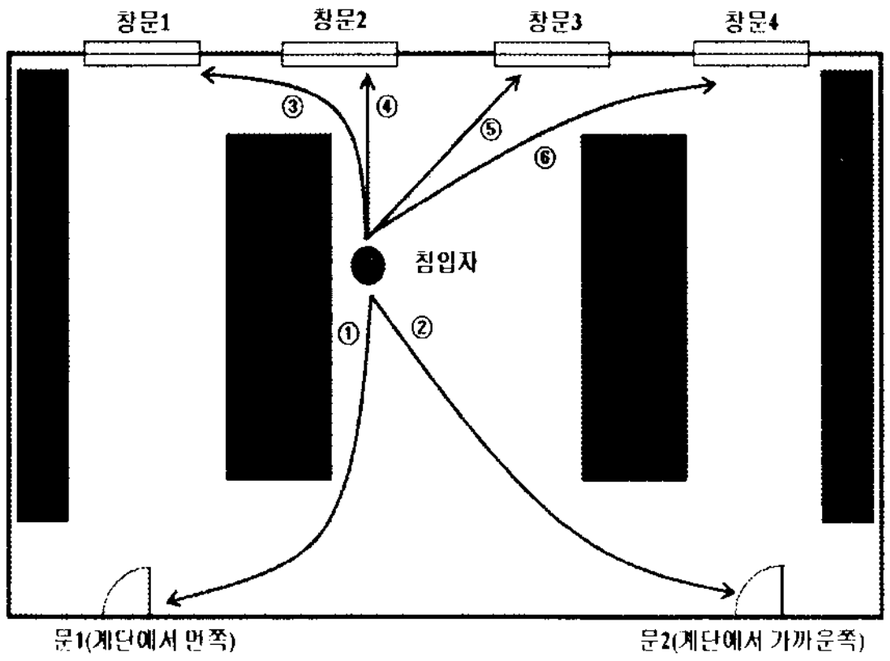

그림 2. 침입자의 예상 이동 경로

이와 같은 초기 적합도로서 침입자의 이동 방향을 확률로서 나타내는 식은 다음과 같다.

$$P(x_i) = x_i / \sum_{n=1}^{m} x_m \tag{2}$$

식(2)로부터 그림 2의 각 장소로 이동할 확률을 구해보면 다음과 같다.

1층이 아니라고 가정하자.
$P(door1) = 5/5+5+1+1+1+1+1+1 = 33.33\%$
$P(door2) = 5+1/5+5+1+1+1+1+1+1 = 40\%$
$P(window1) = P(window2)$
$= P(window3) = P(window4) = 6.67\%$

또한 문이나 창문의 개수가 n이라고 가정하면, 우리는 침입자에서 문이나 창문으로의 방향 $[x_{m(n)}, y_{m(n)}]^T$ 을 각각의 문과 창문의 위치 좌표를 가지고 식(3)을 통해 얻을 수 있다.

$$\begin{bmatrix} x'_{m(n)} \\ y'_{m(n)} \end{bmatrix} = \begin{bmatrix} x_{site(n)} \\ y_{site(n)} \end{bmatrix} - \begin{bmatrix} x_{invader} \\ y_{invader} \end{bmatrix}$$
$$\begin{bmatrix} x'_{m(n)} \\ y'_{m(n)} \end{bmatrix} = c \begin{bmatrix} x_{m(n)} \\ y_{m(n)} \end{bmatrix},$$
$$where\ c\ is\ \ c \in N,\ c > 0 \tag{3}$$

여기서 m대의 로봇이 침입자를 포위하고 l $(a < l < b)$은 사용자가 지정한 포위시 침입자와의 거리라고 가정했을 때, 각각 로봇의 위치를 구하기 위해서, 위에서 얻은 $[x_{m(n)}, y_{m(n)}]^T$을 가지고 식(4)를 계산한다.

```
for i = 1 to n
  k1 = 2, k2 = 0.5, m1 = 0.5, m2 = 0.25
  while(1)
    if a < ||[x_m(n), y_m(n)]^T|| < b,
      [x_location(n); y_location(n)] = [x_invader; y_invader] - [x_m(n); y_m(n)]
      break
    elseif ||[x_m(n), y_m(n)]^T|| < a,
      x_m(n) = [k1 x_m(n)], y_m(n) = [k1 y_m(n)]
    elseif ||[x_m(n), y_m(n)]^T|| > b,
      x_m(n) = [k2 x_m(n)], y_m(n) = [k2 y_m(n)],
    k1 = k1 - m1, k2 = k2 + m2
    m1 = 0.5 m1, m2 = 0.5 m2
end
where [ • ] is the number
that is smaller than • .
```
(4)

위 식으로부터 얻은 $[x_{location(n)}, y_{location(n)}]^T$는 장애물의 위치를 고려하지 않은 상태의 좌표이다. 그러므로 본 논문에서는 만약 위에서 얻은

좌표가 장애물이 위치하고 있는 좌표라면 식(5)를 반복하며 장애물의 좌표인지 확인해 좌표를 다시 정해주었다.

$$\begin{bmatrix} x_{o(n)} \\ y_{o(n)} \end{bmatrix} = \begin{bmatrix} x_{location(n)} \\ y_{location(n)} \end{bmatrix} - \begin{bmatrix} x_{invader} \\ y_{invader} \end{bmatrix}$$

if $x_{o(n)} > 0$ *then* $x_{location} - 1$

else $x_{o(n)} \leq 0$ *then* $x_{location} + 1$

$x_{location(n)} = x_{o(n)}$

if *the coordinate don't existed at the obstacle*

break

if $y_{o(n)} > 0$ *then* $y_{location} - 1$

else $y_{o(n)} \leq 0$ *then* $y_{location} + 1$

$y_{location(n)} = y_{o(n)}$

if *the coordinate don't existed on the obstacle*

break (5)

마지막으로 m대의 로봇에 각각 위치를 지정해주어야 한다. 먼저 침입자를 기준으로 $\pm x$, $\pm y$방향으로 각각의 이동장소에 대한 확률을 나눈다. 그림 2로서 예를 들어보면,

$$P(+x) = P(door2) + P(window3) + P(window4)$$
$$P(-x) = P(door1) + P(window1) + P(window2)$$

$$P(+y) = \sum_{n=1} P(window(n))$$
$$P(-y) = P(door1) + P(door2)$$

위 결과에서 y방향을 기준이 되어, $\pm x$방향의 확률을 고려해서 $P(\pm x, \pm y)$의 확률을 계산할 수 있다.

$$P(+x,+y) = \frac{P(window1) + P(window2)}{P(+y)}$$
$$P(-x,+y) = \frac{P(window3) + P(window4)}{P(+y)}$$
$$P(+x,-y) = \frac{P(door2)}{P(-y)}, \quad P(-x,-y) = \frac{P(door1)}{P(-y)}$$

m대의 로봇 중에서 $\pm x$, $\pm y$방향에 위치할 로봇의 수는 $mP(\pm y)P(\pm x, \pm y)$가 된다. 마지막으로 각각의 로봇의 위치를 정해주는 방법은 식(5)로부터 얻은 침입자로부터 $(\pm x, \pm y)$방향의 위치 좌표의 개수가 $mP(\pm y)P(\pm x, \pm y)$보다 크면 그림 3(a)와 같이 위치 좌표의 개수와 로봇의 대수가 같아질 때까지 위치를 줄인다. 만약 작으면 그림 3(b)와 같이 새로운 위치를 얻는다.

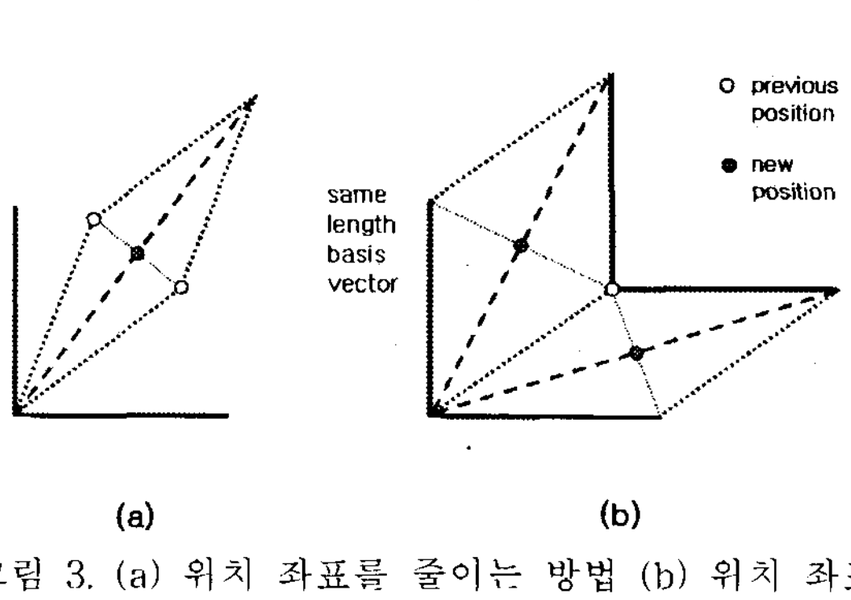

그림 3. (a) 위치 좌표를 줄이는 방법 (b) 위치 좌표를 추가하는 방법

4. High-level Controller의 설계

High-level Controller는 Formation Controller로부터 로봇의 위치를 받아서, 로봇의 이동 경로를 계획해준다. 이동 경로를 지정해주는 알고리즘은 그림 4과 같이 주어진 맵을 각 영역으로 구분함으로써 TSP(Travel Salesman Problem)문제처럼 만들어주어 GA를 통하여 최적화된 로봇들의 이동 경로를 찾아낸다. 각 맵을 구분한 영역의 적합도는 영역에서 바로 갈 수 있는 장소의 적합도와 다른 영역을 두 번 이내에 거쳐서 갈 수 있는 장소의 적합도의 합으로 정해주었다. 밑의 예는 *Area*1의 적합도를 구한 것이다.

$$Fitness(Area1) = 5 \times 4 + 1 \times 2 + (1+1+6) \times 1 = 32$$

유전알고리즘의 해들의 구성은 각각의 로봇의 초기 위치정보에서 이동 가능한 영역을 찾아내고, 이 후 연결된 영역으로 랜덤하게 구성된다. 연결된 영역이란 그림 2를 예로 들면 *Area*1에서는 *Area*2 또는 *Area*10으로만 이동할 수 있으며 다른 영역은 갈 수 없다는 것을 의미한다. 해들의 적합도는 영역들이 가지는 적합도를 이용해서 아래 과정을 통해 부여된다.

if *any Robot*$_n$ *preempted Area*$_n$

$$Fitness(Area_n) -= \frac{num\ of\ preempting\ Robot}{num\ of\ all\ Robot}$$

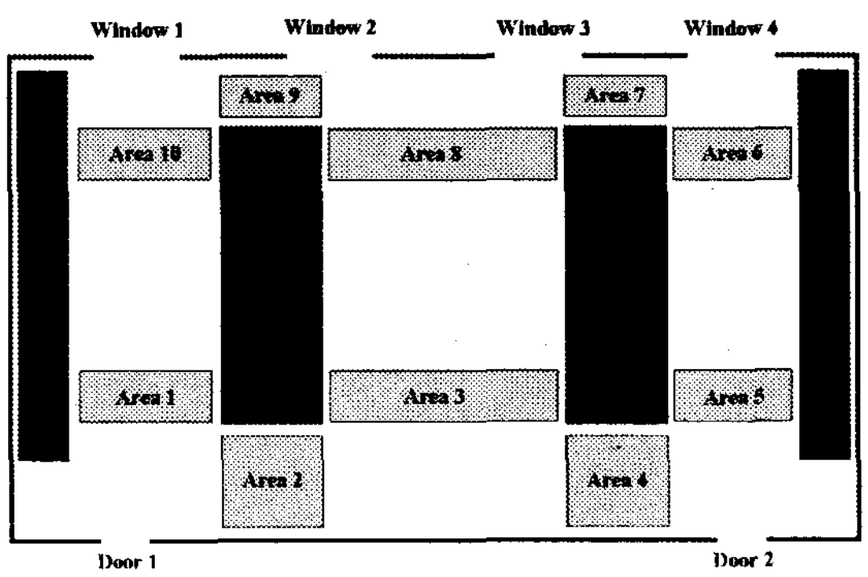

그림 4. 맵 영역 구분의 예

$$Fitness(Robot_n) = Fitness(1st\ Area_n) \times 1 + Fitness(2nd\ Area_n) \times \frac{1}{2} + \ldots$$

$$Fitness(individual_i) = \sum_{n=1}^{m} Fitness(Robot_n)$$

위의 과정으로부터 얻어낸 적합도를 가지고 GA를 통해 최적해를 찾아낸다. High-level Controller는 이 이동 경로로서 로봇에게 움직일 방향을 전송하게 된다.

5. 시뮬레이션

본 논문에서는 컴퓨터 시뮬레이션을 통해 제안된 알고리즘을 평가하였다. 로봇은 10대로 가정하였으며, 각각의 로봇과 침입자의 초기 위치를 그림 5와 같이 가정하였으며, High-level Controller의 GA의 종료 세대 수는 200세대로 설정하고 시뮬레이션을 하였다. 그림 5는 로봇들이 이 알고리즘을 통해 좀 더 전략적으로 이동한 것을 보여준다. 비슷한 위치에서 침입자를 포위하기위해 출발을 하였더라도 서로 다른 경로를 선택하여 협조적으로 탐색하면서 침입자에게 접근한 것을 볼 수 있다.

6. 결 론

본 논문에서는 침입자 추적을 위한 군집 로봇의 분산 이동 알고리즘에 대해 제안하였고, 시뮬레이션을 통해 알고리즘을 평가하였다. 침입자 추적을 위해 군집 로봇의 이동 목적지를 침입자의 예상 이동 경로를 확률로서 평가하여 계산하고, 목적지까지의 군집 로봇의 이동 경로 또한 확률로서 결정하여 로봇의 전략적인 분산 이동을 계획하였다. 시뮬레이션을 통해 본 알고리즘의 성능을 평가하였지만, 이것이 최적의 알고리즘이라고는 할 수 없다. 현실에서는 여러 가지 변수들이 존재하기 때문에 실제 로봇을 통한 시뮬레이션과 여러 환경 변수들에 대해 학습을 통하여 알고리즘의 성능을 좀 더 높일 수 있는 방법을 추후 연구하여야 하겠다.

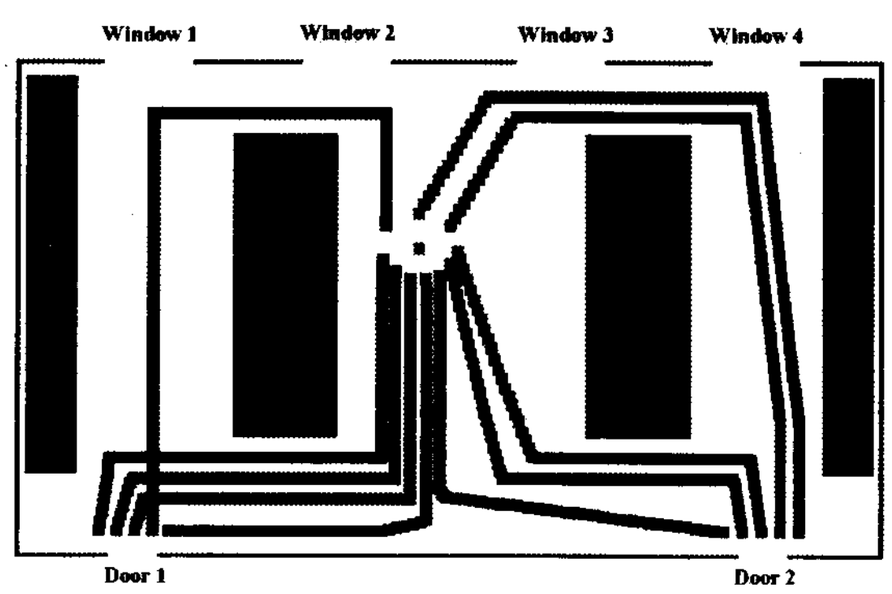

그림 5. 시뮬레이션 결과

참 고 문 헌

[1] R. A. Brooks, P Maes, M. J. Mataric and G. More, "Lunar base construction robots," *Proc. 1990 IEEE/RSJ Int. Workshop on Intelligent Robots and Systems(IROS'90)*, Tokyo, Japan, pp. 389-392, 1990.

[2] W. Iida and K. Ohnishi, "Mobile Robot Teamwork for Cooperated Task," *Proceeding of the 2002 IEEE/RSJ Int. Conference on Intelligent Robots and Systems*, pp. 1561-1566, 2002.

[3] W. Burgard, D. Fox, M. Moors, R. Simmons, and S. Thrun, "Collaborative multi-robot exploration," *Proc. 2000 IEEE International Conference on Robotics and Automation*, San Francisco, CA 2000.

[4] W. Kowalczyk, "Multi-Robot Coordination," *Proc. 2001 IEEE, 2nd Workshop on Robot Motion and Control*, pp. 219-223, 2001.

[5] H. Yamaguchi, "A Cooperative Hunting Behavior by Mobile Robot Troops," *Proc. 1998 IEEE*, Leuven, Belgium, pp. 3204-3209, 1998.

[6] D. J. Pack and B. E. Mullins, "Toward Finding an Universal Search Algorithm for Swarm Robots," *FProceeding of the 2003 IEEE/RSJ Int. Conference on Intelligent Robots and Systems*, Las Vegas, Nevada, pp. 1945-1950, 2003.