한국시뮬레이션학회 논문지
Vol. 18, No. 4, pp. 49-58 (2009. 12)

집단 로봇 제어를 위한 수정된 플로킹 알고리즘의 시뮬레이션 검증

이은복[1†] · 신석훈[1] · 유용준[1] · 지승도[1] · 김재익[2]

Verification of Modified Flocking Algorithm for Group Robot Control

Eun-Bok Lee · Suk-Hoon Shin · Yong-Jun You · Sung-Do Chi · Jae-Ick Kim

ABSTRACT

Top-down approach in the intelligent robot research has focused on the single object intelligence however, it has two weaknesses. One is that has a high cost and a long spending time of sensing, calculating and communications. The other is the difficulty of responding to react changes in the unpredictable environment. we propose the collective intelligence algorithm based on Bottom-up approach for improving these weaknesses and the applied agent model and verify by simulation. The Modified Flocking Algorithm proposed in this research is the algorithm which is modified version of the concept of the Flocking (Craig Reynolds) which is used to model the flocks, herds, and schools in the graphics or games, and simplified the operation of conventional Flocking algorithm to make it easy to apply for the number of group robots. We modeled the Boid agent and verified possibility collectivization of the Modified Flocking Algorithm by simulation. And We validated by the actual multiple mobile robot experiment.

Key words : Bottom-up approach, Artificial Life, Flocking, Boid, Group robot, Agent, Simulation

요 약

로봇의 지능화에서 기존의 하향식 접근 방식은 단일 개체 지능화에 중점을 두어 왔으나 이러한 접근은 첫째, 센싱, 연산, 통신에 소모되는 비용과 시간이 크다는 것 그리고 둘째, 예측 불가능한 환경변화에 민감하게 대응하는데 어려움이 있다. 본 연구는 이러한 단점을 극복하는 상향식 접근 방식의 집단적 지능화를 위한 알고리즘과 이를 적용한 에이전트 모델을 제안하고 시뮬레이션을 통해 검증하였다. 본 연구에서 제안한 수정된 플로킹 알고리즘은 그래픽이나 게임에서 집단이동을 보이는 생명체를 모델링 하는데 주로 사용되어온 플로킹(Flocking, Craig Reynolds)의 개념을 단순화시킴으로써 기존 플로킹의 연산과정을 단순화하여 보다 많은 수의 집단 로봇에 적용하기 용이 하도록 수정한 알고리즘이다. 시뮬레이션을 통해 수정된 플로킹 알고리즘의 집단화 적용 가능성을 검증하였고, 이를 위한 보이드 에이전트를 모델링 하였다. 또한 실질적 검증을 위하여 실제 집단 로봇에 대한 사례 연구를 진행하였다.

주요어 : 상향식 접근 방식, 인공생명, 플로킹, 보이드, 집단 로봇, 에이전트, 시뮬레이션

1. 서 론

인간이 직접 수행하기에 어려운 작업이 많아짐에 따라 지능형 로봇에 대한 관심이 높아지고 있으며 다수의 지능형 로봇들이 통신 등을 통해 집단을 구성하여 작업을 수행하는 집단 로봇에 대한 연구도 활발하게 이루어지고 있다[1].

그러나 과거 하향식 접근 방식과 중앙 집중식 제어를 통한 집단 로봇 제어는 실세계의 변화에 즉각적으로 반응하기 위해 복잡한 행동제어 알고리즘이 요구되었고, 단일 개체의 지능화 접근에 중점을 두고 있어 이를 위한 고성능의 센서와 높은 연산성능이 필요했다. 또한 집단 구성을 위해 센싱, 연산, 통신에 소모되는 비용과 시간적 부하가 크고 환경의 변화에 따른 위치 재설정에 한계가 있다

* 이 논문은 2008년도 한국항공대학교 교비지원 연구비와 방위산업청과 국방과학 연구소의 지원(계약번호 UD 030000AD) 으로 수행되었음.
2009년 7월 20일 접수, 2009년 10월 19일 채택
[1] 한국항공대학교 컴퓨터공학과
[2] 국방과학연구소 함정전투체계개발단
주 저 자 : 이은복
교신저자 : 이은복
E-mail; danalee@kau.ac.kr

[2]. 이에 비해 상향식 접근 방식은 실세계와 즉각적으로 반응하고 단순한 알고리즘을 사용하며 집단 로봇의 집단적 지능화를 목표로 한다는 점에서 하향식 패러다임과 차별화 되고 집단 로봇의 제어에 더 효율적이다[3-5].

이에 따라 상향식 접근 방식을 적용시킨 집단 로봇 연구에 대한 관심이 높아졌으며 현재 MIT의 SwarmBot, 유럽 3개국 4개 대학이 연합하여 만든 swarm-bot, 캘리포니아 대학의 Pioneer 2 mobile robot을 이용한 집단로봇 연구, 카네기 멜롯 대학의 Millibot 등 수많은 연구가 이루어지고 있다[6-8].

이 중 가장 대표적인 SwarmBot은 MIT 인공지능 연구실과 IROBOT의 연계로 진행 중이며 연구의 목적은 1000대 이상의 로봇으로 구성된 로봇집단을 효율적으로 제어하여 다양한 작업을 수행하는 것이다. SwarmBot의 수행 작업으로는 목표 지시 장비를 이용한 길 찾기, 무리의 중심에서부터 최대한 고르게 펼쳐지는 자유분열, 지정된 범위 내에서의 최대한 고르게 펼쳐지는 제한 분열, 무리의 리더로 모이는 응집, 각각의 그룹별로 그룹리더에게 모이는 그룹응집, 움직이는 리더를 따라가기 등이 있다[9-12]. SwarmBot은 본 연구에서 사용한 로봇인 FloBot보다 훨씬 고사양, 고비용의 로봇이며 수많은 센서를 가지고 환경을 인지한다. 본 논문의 4절에서 SwarmBot과 FloBot에 관한 제원을 비교할 것이다.

국내 집단 로봇 통제 및 환경 기술 개발 연구는 집단 로봇에 SVM을 적용한 행동학습 및 진화, Q-Learning 알고리즘을 이용한 센싱 및 집단화제어, 협조행동을 위한 강화 학습 기반의 진화 등이 있으나 현재까지 인공 생명 개념을 적용한 집단 제어 실험이라고 보기 어렵다[13-15].

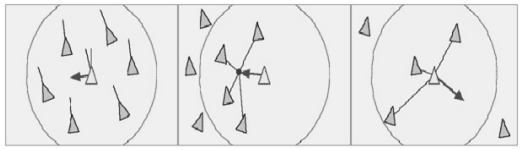
이러한 상향식 접근 방식을 적용한 이론 중에서 인공생명 개념의 플로킹 이론이 있다. 그러나 기존의 플로킹 알고리즘을 이용한 집단 로봇 제어 연구는 이에 사용되는 행동 제어 알고리즘의 수학적 표현을 저사양의 로봇이 처리하기에는 어려운 점이 있고 집단 구성을 위해 센싱과 수식 계산에 들어가는 시간적인 부하가 크다는 문제점이 있다. 본 논문에서는 이를 극복하기 위해 수정된 플로킹 알고리즘을 제안하며, 보다 단순해진 알고리즘을 처리함에 따라 저사양의 처리기로도 기존의 플로킹 알고리즘과 매우 비슷한 로봇의 집단적 창발현상을 보일 수 있음을 시뮬레이션 하였고 이를 토대로 실제 집단 로봇에 적용한 검증을 시도하였다.

본 논문의 구성은 다음과 같다. 2절에서는 기존의 플로킹 개념과 수정된 플로킹 개념을 비교하고 수정된 플로킹 개념을 기반으로 한 행동제어 알고리즘에 대해 소개하며 3절에서는 알고리즘의 타당성을 검증하기 위해 기존의 플로킹 개념과 수정된 플로킹 개념을 탑재한 시뮬레이션의 개요와 그 결과를 비교한다. 4절에서는 시뮬레이션에서 검증된 알고리즘을 실제 로봇을 이용하여 실험하고 그 결과를 분석하고 5절에서 결론을 맺는다.

2. 수정된 플로킹 알고리즘

2.1 레이놀즈의 플로킹 이론

플로킹 이론은 인공생명 개념의 하나로써 일종의 무리를 모델링하는 기법으로 1987년 플로킹의 아버지로 불리는 레이놀즈가 처음으로 발표하였다. 레이놀즈는 조종 행동(Steer)이라 불리는 그림 1의 세 가지 기본 조종힘(Steering Force)을 이용한 시뮬레이션을 통해 생태계에 존재하는 집단 생물들과 유사한 집단행동 패턴을 취하게 됨을 보여주었다. 이 때 각각의 개체를 보이드라 하며 이 보이드들의 행위를 결정하는 세 가지 간단한 기본 규칙은 다음과 같다[16,17].



(a) 정렬 (b) 응집 (c) 분리

그림 1. 플로킹 기본 규칙[16]

표 1. 보이드의 위치를 결정하는 수학적 모델[17]

$B=\{b_i, i=0,1,\ldots,n-1\}$: 보이드 b_i의 집합
$V_i=\{b_i\in B, \forall b_j: \lvert b_i-b_j\rvert<e, j=0,1,\ldots,m-1\}$: b_i의 시야 범위에 속한 모든 보이드
p_i : b_i의 위치
$\overrightarrow{v_i}$: b_i의 진행방향
$\overrightarrow{u_i}$: b_i의 3차원 상향벡터
$\overrightarrow{S_i}=-\sum_{\forall b_j\in V_i}(p_i-p_j)$: b_i의 응집(Cohesion)
$\overrightarrow{k_i}=\left(\sum_{\forall b_j\in V_i}\frac{p_j}{m}\right)-p_i$: b_i의 분리(Separation)
$\overrightarrow{m_i}=\sum_{\forall b_j\in V_i}\frac{\overrightarrow{v_i}}{m}$: b_i의 정렬(Alignment)
(보이드의 위치를 결정하기 위한 힘)
$p_i'=p_i+\overrightarrow{v_i}+S.\overrightarrow{s_i}+K.\overrightarrow{k_i}+M\overrightarrow{m_i}$: 시간t가 지난 후 보이드 b_i의 위치 p_i'
(S,K,M은 각각 속성 값에 작용하는 상수, 다음 위치를 결정함과 동시에 적정거리를 결정하는 데 사용)

표 2. 수정된 플로킹 모델의 의사코드

```
 1) Bst = SF
 2) while(!Goalin()){
 3)     if(Bst == SF){
 4)         Search(Bnt);
 5)         if(Bnt){
 6)             if(dBnt > dstd){
 7)                 SetDirection(θBnt);    }
 8)             else if(dBnt < dstd){
 9)                 SetDirection(2 × θI - θBnt);    }
10)             else{
11)                 SetDirection(θI);    }
12)             GoAhead();}
13)         else{
14)             Bst = SL;    }    }
15)     else if(Bst == SL){
16)         SetDirection(θG);
17)         GoAhead();
18)         While(Bst == SL){
19)             Search(Bnt);
20)             if(Bnt){
21)                 Bst = SF;    }    }    }    }
22) Stop();
```

	B_{st}	: 보이드의 상태
	B_{nt}	: 가장 가까운 보이드
	d_{Bnt}	: 가장 가까운 보이드와의
	d_{std}	: 적정거리
범례	θ_{Bnt}	: 가장 가까운 보이드의 방
	θ_I	: 현재 자신의 방향
	θ_G	: 목표의 방향
	S_L	: 보이드의 'leader' 상태
	S_F	: 보이드의 'follower' 상태

- 정렬 : 주변 보이드들과 같은 방향을 가리키려는 힘
- 응집 : 주변 보이드들과 가까운 쪽으로 가려는 힘
- 분리 : 주변 보이드들과 먼 쪽으로 가려는 힘

표 1은 보이드의 위치를 결정하는 수학적 모델이다. 이러한 규칙들에 의한 플로킹은 벡터 수식의 계산과 자신의 시야에 있는 360도 범위의 모든 보이드들을 고려해야 하므로 그에 따른 시간과 메모리 소모가 문제가 된다.

2.2 수정된 플로킹 알고리즘

수정된 플로킹 알고리즘은 기존 플로킹 알고리즘의 벡터 연산의 복잡성을 줄이기 위하여 시야에 존재하는 보이드 중 가장 가까운 보이드와의 거리만을 측정하여 다음의 위치를 결정한다.

시야 범위를 진행방향 기준으로 전방 180도로 한정하여 거리측정의 대상이 되는 보이드의 수를 줄임으로서 연산과정의 복잡성을 낮추었다. 이 때, 보이드 간의 적정거리는 보이드 길이의 두배로 설정한다.

또, 기존 플로킹 알고리즘의 응집, 정렬, 분리의 조종힘을 각각 하나의 행동으로 규정하고, 가장 가까운 보이드와의 거리가 적정 거리보다 멀면 가까워지는 응집행동을, 가까우면 멀어지려는 분리행동을 통해 자신의 위치를 결정함으로써 위치 벡터들의 연산, 합력의 계산 과정을 한 번의 행동 선택 과정으로 단축하였다. 정렬행동은 시야에 존재하는 보이드가 없을 경우 스스로를 리더로 인식하고, 거리계산 없이 정해진 목표방향을 향해 가는 행동으로 정의하였다.

수정된 플로킹 알고리즘을 통해 구축된 모델의 의사코드는 표 2와 같다. 이 행동 모델링 규칙에서 보이드의 상태는 'leader' 상태, 'follower' 상태의 두 가지를 갖는다. 보이드의 초기상태는 'follower' 상태이며(표 2에서 Line 1), 목표에 도착할 때 까지 다음의 행동을 계속한다(Line 2). 'follower' 상태일 때 보이드는 가장 가까운 보이드를 찾고(Line 2,3,4) 이를 자신의 리더로 인식하고(Line 5) 둘 사이의 거리를 측정하여(Line 6) 적정거리보다 멀면 응집행동을(Line 7), 가까우면 분리행동을 한다(Line 8,9). 둘 사이의 거리가 적정거리이면 현재 방향을 유지하고(Line 10,11) 진행한다(Line 12). 만약, 다른 보이드를 발견하지 못하면, 'leader' 상태가 된다(Line 13,14).

'leader' 상태일 때 보이드는 목표점을 향해 나아가는 행동인 정렬행동을 하게 되며(Line 15,16,17), 일정시간이 지나면 자신보다 앞서있는 보이드가 있는지 주변을 탐색한다(Line 18,19). 이때 다른 보이드를 발견하게 되면 'follower' 상태가 된다(Line 20,21). 목표점에 도달하면 알고리즘이 종료된다(Line 22).

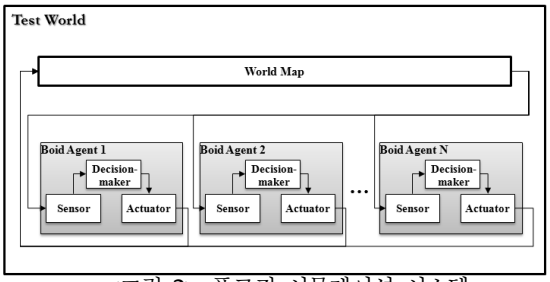
3. 시뮬레이션 및 성능 비교

플로킹 시뮬레이션은 집단 로봇 모델들이 출발지점에서 출발하여 대각선 경로로 이동하여 도착지점까지 도착하는 시뮬레이션이다.

3.1 플로킹 시뮬레이션의 구조

제안된 알고리즘의 타당성 검토를 위해 그림 5와 같은 구조를 가지는 플로킹 시뮬레이션 시스템에서 집단 보이드 에이전트(Agent) 모델(Model)을 이용한 시뮬레이션을 수행하였다.

시뮬레이션 시스템은 보이드 에이전트(Boid Agent)와



<그림 2> 플로킹 시뮬레이션 시스템

그림 2. 플로킹 시뮬레이션 시스템

시뮬레이션 환경(World Map)으로 구성되어 있다. 보이드 에이전트는 보이드 개체 모델로 집단의 최소 단위인 3개부터 n개까지 증가가 가능하며, 각각 센서모델, 의사결정모델, 실행모델로 구성된다.

센서(Sensor)모델 : 주변 보이드 에이전트와의 거리를 측정하여 의사결정모델에게 전달한다.
의사결정(Decision maker)모델 : 센서모델에서 전달된 거리 값을 기준을 표 2의 행동모델링 의사코드를 기반으로 행동을 결정하며 이 행동을 실행모델로 전달한다.
실행(Actuator)모델 : 의사결정모델에서 결정된 행동을 실행하는 모델이다.

World Map은 보이드 에이전트들에게 시뮬레이션 환경의 정보를 전달하는 모델로 시작점의 위치 정보(start position), 목표점의 위치 정보(goal position), 장애물의 위치 정보(obstacle position), 보이드 에이전트들의 위치 정보(boid position)를 좌표형태(x,y)로 저장하고 있다. 이 위치 정보들을 보이드 에이전트로 전달하고, 보이드 에이전트의 다음 위치 정보를 전달받아 갱신하는 역할을 한다.

3.2 시뮬레이션의 초기조건과 시나리오

1024×768 픽셀의 범위 내에서 집단 제어 현상을 가장 잘 표현하기 위하여 보이드 3개부터 100개까지 실험해본 결과, 이들 중 30개체가 가장 적절한 것으로 나타나는 바, 이를 중심으로 결과를 분석하였다.

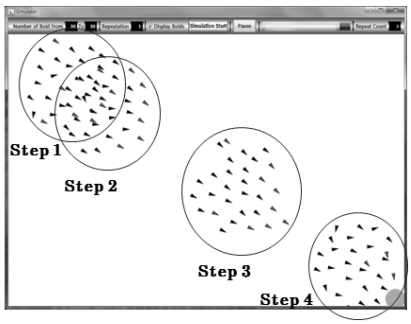
기존 플로킹 알고리즘은 거의 360도 전방위, 수정된 플로킹 알고리즘은 전방 180도를 탐색하며, 보이드 에이전트의 인식거리는 40픽셀이다. 1회 의사결정 후 모든 보이드 에이전트가 동일하게 1픽셀을 이동하며 출발상태, 정상(주행)상태, 도착상태로 나뉜다. 출발상태는 시뮬레이션 시작부터 보이드 에이전트들이 자신의 위치를 잡았을 때까지이며, 정상상태는 첫 번째 보이드 에이전트가 도착지점에 도착했을 때까지, 도착상태는 시뮬레이션 종료까지이다.

모든 보이드들은 시뮬레이션이 시작되면, 다른 보이드를 탐색하고, 시야내의 보이드가 없는 경우 리더가 된다. 모든 보이드가 플로킹 알고리즘에 따라 이동하며 도착지점에 도착하면 시뮬레이션이 종료된다.

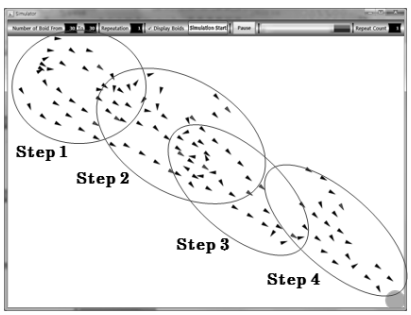
3.3 시뮬레이션의 결과 및 분석

그림 3에서 보는 바와 같이 시뮬레이션 결과 중 이동 모습 면에서 기존 플로킹 알고리즘과 수정된 플로킹 알고리즘이 집단을 이루어 움직인다는 점에서 거의 흡사한 모습을 보였지만 집단의 밀집형태면에서 차이를 보였다.

기존 플로킹 알고리즘(그림 3(a))은 대체로 원형의 대형을 이루고 목표까지 큰 변화 없이 도달하는 것을 볼 수 있고, 수정된 플로킹 알고리즘(그림 3(b))은 기존 플로킹 알고리즘에 비해 타원에 가까운 대형을 이루는 것을 볼 수 있다(Step 2~4). 이 이유는 기존 플로킹 알고리즘은 시야에 감지되는 모든 이웃 보이드와의 벡터 연산을 통해 최적의 위치를 찾는데 그 과정에서 이웃 보이드 하나가 여러 보이드들에게 영향을 미치는 경우가 많고, 그로 인



(a) 기존 플로킹 알고리즘



(b) 수정된 플로킹 알고리즘

그림 3. 시뮬레이션 결과

해 하나의 보이드의 응집력이 시야에 존재하지 않는 다른 보이드까지 영향을 미치게 되고, 장애물이 없는 경우 좀처럼 분기 효과가 보이지 않게 되며, 대형은 원형에 가까워지게 된다.

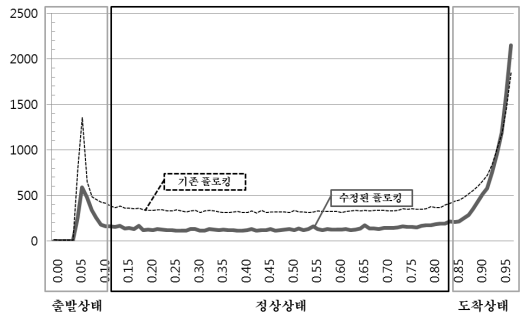
반면에 수정된 플로킹 알고리즘은 시야에 존재하는 가장 가까운 이웃 보이드 하나와의 거리만을 이용하게 됨으로 기존 플로킹에 비해 감지된 이웃 보이드의 공유가 줄어든다. 이는 장애물이 없는 경우에도 분기가 일어나는 원인이 된다.

그럼에도 불구하고 집단유지가 가능한 이유는 'leader' 상태의 보이드가 정렬과정을 수행할 때 도착지점를 향해 방향을 수정하는데 그 과정에서 분리되었던 다른 보이드를 만날 확률이 매우 높고, 이 경우 두 집단이 하나의 집단으로 결합하게 된다. 이 과정이 빈번하게 일어남을 시뮬레이션 결과에서 볼 수 있었다. 또한, 재결합 과정에서 도착지점에서 먼 집단이 도착지점에서 가까운 집단으로 흡수되는데 이 과정의 반복이 원형보다는 타원형을 보이게 된다.

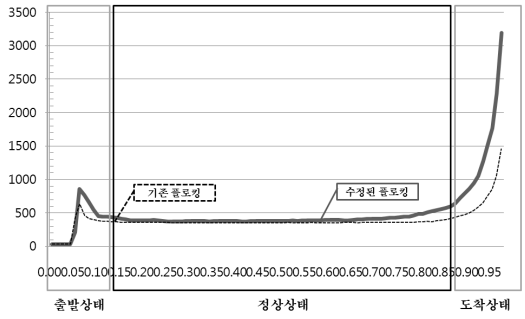
그림 4는 상세한 시뮬레이션 결과를 그래프로 표현한 것으로 출발지점에서 도착지점까지의 거리를 100으로 봤을 때 전체 보이드들의 무게 중심 위치가 매 1의 거리를 이동했을 때 걸린 판단시간, 판단횟수, 다른 보이드들과의 거리를 기록한 것이다. 각 그래프의 수치는 30대의 보이드 집단을 이용한 시뮬레이션 100회의 평균을 나타낸다. 그림 4에서 보는 바와 같이 출발상태와 도착상태는 두 알고리즘 모두 과도현상을 보이기 때문에 본 연구에서는 정상(주행)상태를 위주로 분석하였다.

그림 4(a)는 판단시간의 비교 그래프이다. 기존 플로킹의 경우 정상상태에서의 평균 판단시간은 350.25초가 걸렸고, 수정된 플로킹은 152.85초가 걸렸다. (여기서 '초'란 .NET 기반 TickTimer로 측정한 1tick을 말한다.) 기존 플로킹 알고리즘이 수정된 플로킹 알고리즘에 비해 약 2.3배 오래 걸렸음을 알 수 있는데 그 이유는 기존 플로킹 알고리즘의 경우 감지된 이웃 보이드들과의 응집, 분리, 정렬을 구하는 벡터 연산 과정 모두가 필요하며, 더 많은 시간을 요구하기 때문이다. 수정된 플로킹 알고리즘은 가장 가까운 보이드 1기와 거리 판단을 통해 행동이 결정됨으로 기존 플로킹 알고리즘에 비해 짧은 시간을 요구한다.

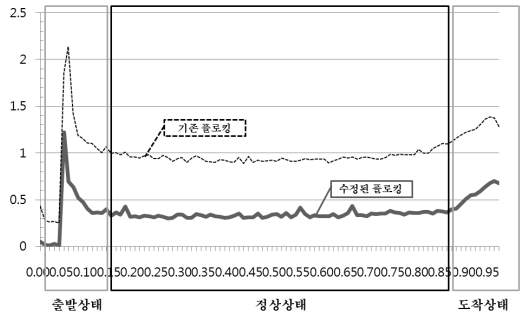
그림 4(b)는 판단횟수의 비교 그래프이다. 기존 플로킹 알고리즘의 경우 평균 360회, 수정된 플로킹 알고리즘의 경우 452회로 수정된 플로킹 알고리즘이 약 1.2배가량 많이 판단함을 알 수 있고 그 이유는 기존 플로킹 알고리즘은 매 순간 집단에서의 최적의 위치를 찾기 위해 많은 연



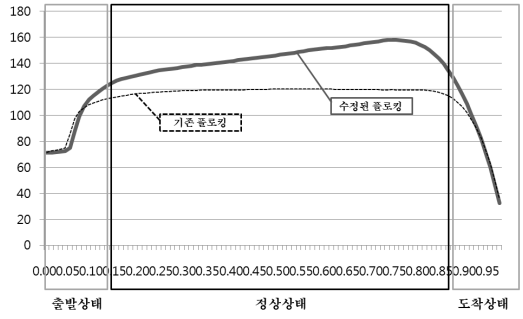
(a) 판단시간



(b) 판단횟수



(c) 판단비용



(d) 집단 분산도

그림 4. 시뮬레이션 결과 그래프

산과정을 거쳐 최소의 판단을 하는 경향이 있고, 수정된 플로킹 알고리즘은 가장 가까운 보이드 1기만을 고려하여 다음 위치를 판단하기 때문에 연산시간이 짧은 반면 판단 횟수는 다소 많을 수 있다는 것을 보여준다.

그림 4(c)는 판단비용의 그래프이다. 1회 판단에 걸리는 시간(판단시간(a)/판단횟수(b))을 표현한 그래프이다. 기존 플로킹 알고리즘의 경우 0.96초, 수정된 플로킹 알고리즘이 0.34초로 수정된 플로킹 알고리즘이 기존 플로킹 알고리즘에비해 약 2.8배 이상 빠르다.

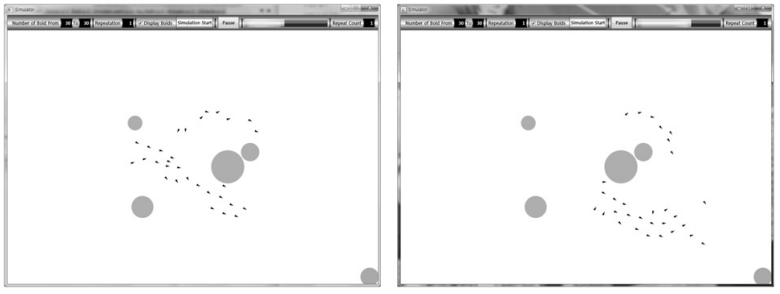
그림 4(d)는 집단 분산도의 그래프로 집단 밀집도의 차이를 나타낸다. 매 1 거리마다 다른 모든 보이드들과의 평균 거리 값을 나타낸다. 기존 플로킹 알고리즘은 정상상태에서 평균 118픽셀로 거의 일정한 거리를 계속 유지하는 것을 볼 수 있고, 이는 그림 3(a)의 모습과 일치한다. 수정된 플로킹 알고리즘은 평균 142픽셀로 기존 플로킹 알고리즘에 비해 느슨한 모습을 보이며, 120픽셀에서 157픽셀까지 밀집도가 분산됨을 보인다. 위의 결과는 수정된 플로킹 알고리즘의 경우 집단이 분기와 재결합하는 과정이 반복되면서 점차 타원형을 이루기 때문이다(그림 3(b)).

그림 4에서 보았듯이 기존의 플로킹 알고리즘은 각 보이드들이 자신의 최적의 위치를 찾아감에 따라 집단구성이 원활하고 집단구성원의 분포가 균일함을 알 수 있다. 그러나 이를 위해 필요한 센싱 데이터가 많고, 연산이 복잡하다는 단점이 있다. 그에 비해, 수정된 플로킹 알고리즘은 판단에 소요되는 시간과 비용이 적다는 장점이 있지만, 상대적으로 집단 분산도가 균일하지 못하고 판단 횟수가 많다는 단점이 있다.

두 번째로 장애물이 존재하는 시뮬레이션을 추가로 진행하였다. 장애물 모델은 보이드 에이전트와 동일한 상태와 구조를 가지고 있으며, 보이드 에이전트와 달리 특정 위치에 고정되어 있고, 부피 값이 크게 설정된다.

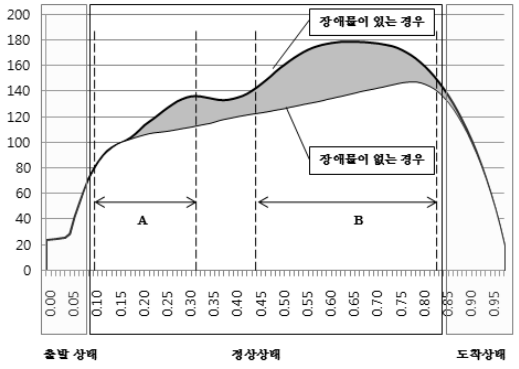
그림 5(a)는 장애물이 존재하는 환경에서 수정된 플로킹 알고리즘을 적용한 보이드 에이전트들이 장애물을 피해 두 그룹으로 나눠지는 것을 볼 수 있다. 또한 그림 5(b,c)는 장애물이 존재하는 경우와 존재하지 않는 경우의 시뮬레이션 결과를 분석한 그래프이다. 그림 5(b)는 판단시간의 비교 그래프로, 정상상태에서 장애물이 없는 경우 평균 판단시간은 42.05초가 걸렸고, 장애물이 있는 경우는 51.70초가 걸렸다.

장애물이 있는 경우, 보이드 에이전트들이 첫 번째 장애물을 만나 우회하는 경우 판단시간이 늘어나고, 우회를



(a) 장애물이 있는 시뮬레이션

(b) 판단시간의 비교



(c) 집단 분산도의 비교

그림 5. 장애물이 있는 시뮬레이션 결과

마치면 다시 판단시간이 줄어듦을 알 수 있다. 두 번째 장애물을 만난 경우도 첫 번째와 유사한 결과를 보이며, 장애물을 통과하면 장애물이 없는 경우와 거의 일치하는 판단시간을 보이는 것을 알 수 있다. 그림 5(c)는 집단 분산도의 그래프이며 이 경우도 판단시간의 그래프와 유사한 변화를 볼 수 있다.

수정된 플로킹 알고리즘은 추가적인 회피 동작을 구현하지 않았음에도 불구하고 자연스럽게 장애물을 회피하는 모습을 볼 수 있었으나, 기존 플로킹 알고리즘의 경우 벡터연산의 특성상 정지해있는 장애물의 벡터가 다른 보이드들에게 영향을 미쳐 장애물 근처에서 모두 정지하는 모습을 보였다. 보편적인 플로킹 알고리즘의 응용 사례의 경우 장애물의 회피를 위한 추가적인 조종힘을 구하는 계산을 추가하여 이를 해결한다[18]. 이는 결과적으로 판단시간이 증가되는 단점을 보인다.

4. 사례연구 : 수정된 플로킹 알고리즘을 탑재한 집단 로봇 실험

수정된 플로킹 알고리즘을 탑재한 플로킹 시뮬레이션의 결과를 검증하기 위해 실내형 GPS를 설치한 실험 환경을 구축하고, 6대의 FloBot이라고 부르는 로봇들을 이용하여 목표점까지 집단을 이루어 이동하는 모습을 확인하였다.

4.1 FloBot의 소개

FloBot은 센서모듈, 의사결정모듈, 실행모듈 구성되어 있는 모바일 형태의 로봇으로 그림 6과 같다. 센서모듈은 그림 7과 같이 적외선 거리센서와 실내형 GPS센서로 이루어져 있고, 머리처럼 보이는 부분이 180도 범위의 시야 확보를 위해 좌/우로 회전 가능하게 만든 적외선 센서부이고, 몸 부분 위쪽에 장착된 것이 실내형 GPS 센서부이다. 이들 센서에서는 응집, 분리 과정을 위한 상대 FloBot과의 거리를 측정하고, 정렬 과정을 위한 목표지점을 전송받는다. 이 외부의 정보를 의사결정모듈로 보내서 플로킹 시뮬레이션과 동일한 수정된 플로킹 알고리즘에 의해 행동을 결정하고 실행모듈을 통해 이동한다. 그림 8은 FloBot에 탑재된 프로그램의 구조를 나타낸다. 이 구조는 크게 Sensor, Decision -Maker, Actuator, Global로 구성되며 Sensor 는 센서모듈, Actuator는 실행모듈을 제어하며, Decision Maker는 수정된 플로킹 알고리즘을 이용하여 의사결정을 담당한다. Global에는 각 모듈에서 사용하는 변수들이 저장된다.

4.2 집단 로봇 실험 및 결과

이 실험은 플로킹 시뮬레이션을 통해 증명한 수정된 플로킹 알고리즘이 실제 로봇에 얼만큼 잘 적용되는가를 검증하기 위한 실험이다.

3m×3m의 정방형 실험 공간에서 6대의 FloBot이 도착지점을 향해 전진 하게 된다. 실험은 78초 동안 진행 되었고, 평균 약 3.95m를 이동하였다. 그림 9에서는 초기상태부터 78초 경과 후 실험이 종료된 후의 모습을 볼 수 있다.

FloBot들은 초기상태에서는 하나의 집단을 이루어 도착지점을 향해 전진하다가 42초에서 장애물을 만나 두 개의 집단으로 분기하여 장애물을 빠져나가서 다시 하나의 집단을 형성한 뒤 도착지점에 도달한 것을 볼 수 있다. 본 실험 결과에서는 FloBot들이 분리, 응집, 정렬 등의 집단 제어 능력을 보이며, 다양한 집단적 지능을 나타냄을 확인 할 수 있었다.

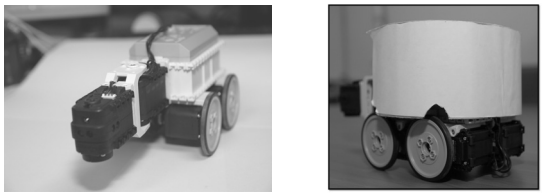
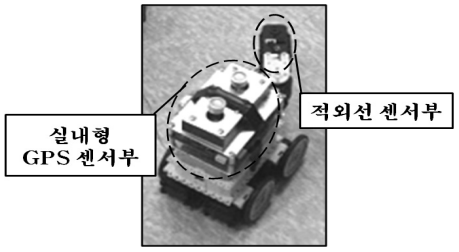
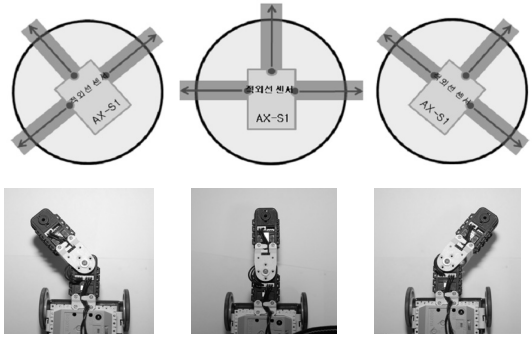


그림 6. Flobot



(a) 적외선 센서부와 실내형 GPS 센서부



(b) 적외선 센서부의 회전운동

그림 7. Flobot의 센서 모듈

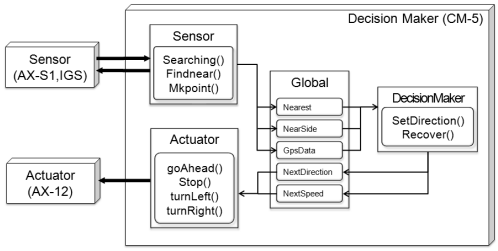


그림 8. FloBot에 탑재된 프로그램 구조

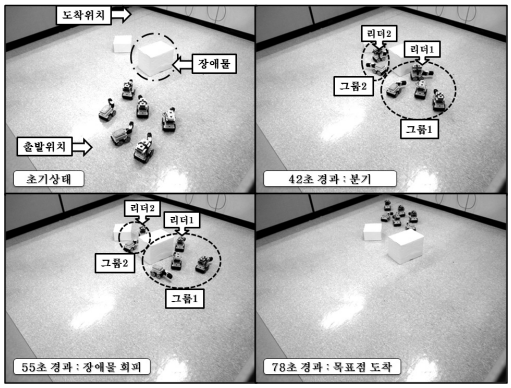


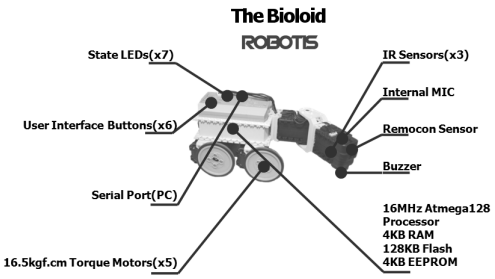
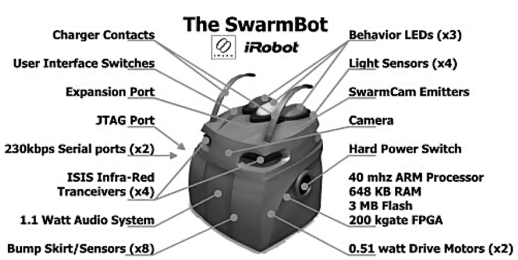
그림 9. 로봇 실험 결과

또한, 표 3는 FloBot과 MIT의 SwarmBot을 비교한 것이다. SwarmBot은 개체간의 통신을 통해 작업명령을 전달하며 이동에 필요한 수식에 들어가는 연산을 처리하기 위한 고사양의 로봇이 필요하며, 이를 위해 고성능 센서를 사용하고 있지만, 이에 비해 FloBot들은 간단한 알고리즘과 저사양의 로봇으로도 SwarmBot과 유사한 집단적 지능을 나타냄을 확인할 수 있다.

5. 결 론

본 논문에서는 수정된 플로킹 알고리즘을 제안하였다. 이를 토대로 상향식 접근 방식을 이용하여 집단 로봇을 제어함으로써 복잡한 연산과정을 줄일 수 있으며 로봇 간 통신에 사용되는 많은 양의 데이터가 불필요함을 보여줄 수 있었다. 기존의 플로킹 알고리즘과 수정된 플로킹 알고리즘을 비교한 시뮬레이션 결과를 보면 첫째, 수정된 플로킹 알고리즘만으로도 로봇의 지능적 집단행동을 충분히 보여줄 수 있었으며 둘째, 시간적, 경제적으로 더 적은 비용만으로도 기존의 플로킹 알고리즘과 대등한 결과

표 3. FloBot과 SwarmBot의 비교

	FloBot	SwarmBot [8]
제원		
개체간의 통신방법	개체간 통신이 없음 각 개체가 서로 통신 없이 자신의 목표로 이동	Random choice, Extream-comm, Card-Dealer's, Tree-Recolor 알고리즘을 상황에 따라 선택, 센서네트워크와 적외선 통신을 이용하여 집단 내 작업 전달
행동 법칙	플로킹 시뮬레이션과 적외선 센서로 들어온 가장 가까운 FloBot 1기와의 거리와 몇 가지 간단한 규칙만으로 행동을 결정 · 응집 - 로봇개체가 상대방과의 거리를 좁히는 행동 · 분리 - 로봇개체가 상대방과의 거리를 늘리는 행동 · 정렬 - 리더상태인 로봇 개체가 자신의 목표점을 향해 이동하는 행동	여러 가지 룰과 수식을 이용하여 일정 집단 내에서 행동을 결정 집단 형성 방법 · 항상 리더로봇 (Reference Robot)의 반지름 내부에 존재 vt=kf(rd−ra) (ra≥ rd) 0 (ra≥ rd) · 항상 리더로봇 (Reference Robot)을 바라봄 · 보이드로봇(Active Robot)은 최단경로를 통해 등속으로 최종위치까지 이동, 최종위치는 위의 두 가지 제약조건을 만족하는 임의의 자세
인식 방법	적외선 감지 (전방 180도), 실내형 GPS (목표점 인식)	적외선 감지 (전방위), 충격감지(전방위), 초음파 감지(전방위), 무선통신, 영상인식(정면) 등

를 확인할 수 있었다.

또한 이를 실제 로봇에 적용한 실험을 통해 알고리즘의 실질적 활용 가능성을 확인할 수 있었다. 이 알고리즘의 효율성은 앞으로 더 많은 수의 개체로 구성된 로봇집단을 제어할 때 더욱 극명한 시간적 차이를 보이게 될 것으로 기대된다.

본 연구결과는 국방 분야의 무인 지뢰 탐사 로봇이나, 무인 정찰 로봇, 대형사고현장에서의 생존자 수색, 우주의 행성 탐사 등 인간이 직접 개입하기에는 위험부담이 크고, 소모적인 자원을 사용하는 분야에서 실질적 활용효과가 클 것으로 기대된다.

향후 시뮬레이션을 통해 고정된 장애물, 이동 장애물, 미로 등의 복잡한 환경에서의 실험을 통해 수정된 플로킹 알고리즘과 기존 플로킹 알고리즘과 비교하고, 각 개체에 여러 가지 작업을 효율적으로 협동하여 수행하는데 필요한 지능들을 검증할 것이며, 이를 토대로 실제 로봇에 적용시켜 실험할 예정이다.

참 고 문 헌

1. 백문홍, 백승호, 박재한 (2008), "정보화시대의 지능형로봇 연구동향", 정보과학회지, 제26권, 제1호, pp. 46-52.
2. Bekey, G.A (2006), "Co-existing with Robots", Intelligent Robots and Systems, 2006 IEEE/RSJ International Conference on.
3. Christopher Hartman and Bedrich Benes (2006), "Autonomous Boids", Journal of Comp. Anim. Virtual Worlds, Vol. 17, pp. 199-206.
4. Robin R. Murphy (1999), "Case Studies of Applying Gibson's Ecological Approach to Mobile Robots", IEEE TRANSACTIONS ON SYSTEMS, MAN, AND CYBERNETICS-PART A: SYSTEMS AND HUMANS, Vol. 29, No. 1.
5. Michael G. Hinchey, Loyola College in Maryland, Roy Sterritt, University of Ulster, Chris Rouff, Lockheed Martin Advanced Technology Laboratories (2007), "Swarms and Swarm Intelligence", Software Technologies, IEEE.
6. Roumeliotis S.I., Bekey G.A. (2002), "Distributed multirobot localization", Robotics and Automation, IEEE Transactions on, Vol. 18, Issue. 5, pp. 781- 795.
7. H. Benjamin Brown, J. Michael VANDE WEGHE, Curt A. BERERTON, and Pradeep K. KHOSLA (2002), "Millibot Trains for Enhanced Mobility", IEEE/ASME Transactions on Mechatronics, pp. 452-461.
8. Stefano Nolfi, Jean-Louis Denebourg, Dario Floreano, Luca Gambardella, Francesco Mondada and Marco Dorigo (2003), "Swarm-Bots: Swarm of Mobile Robots able to Self-Assemble and Self-Organize", ERCIM News No. 53, pp. 25.
9. James McLurkin (2008), "Measuring the Accuracy of Distributed Algorithms on Multi-Robot Systems with Dynamic Network Topologies", Proceedings of the The 9th International Symposium on Distributed Autonomous Robotic Systems.
10. M. Schwager, J. McLurkin, J. J. E. Slotine, D. Rus (2008), "From Theory to Practice: Distributed Coverage Control Experiments with Groups of Robots", Proceedings of International Symposium on Experimental Robotics.
11. James D. McLurkin (2008), Analysis and Implementation of Distributed Algorithms for Multi-Robot Systems, Ph.D, in Computer Science at the MASSACHUSETTS INSTITUTE OF TECHNOLOGY.
12. James McLurkin, Daniel Yamins "Dynamic Task Assignment in Robot Swarms", Proceedings of Robotics: Science and Systems.
13. 서상욱, 양현창, 심귀보 (2008), "SVM을 이용한 군집로봇의 행동학습 및 진화", 한국지능시스템학회 논문지, 제18권, 제5호 pp. 712-717.
14. 서상욱, 김호덕, 심귀보 (2007), 군집 로봇의 협조 행동을 위한 강화 학습 기반의 진화 및 학습 알고리즘, 퍼지 및 지능시스템학회 논문지, 제17권, 제5호, pp. 591-597.
15. 심귀보, 이동욱 (2006), "군집로봇의 군 행동을 위한 통신 모델과 이론적인 해석", 퍼지 및 지능시스템학회 논문지, 제16권, 제1호, pp. 8-17.
16. Craig Reynolds (1987), "Flocks, Herds, and Schools: A Distributed Behavioral Model", Computer Graphics, 21 (4) (SIGGRAPH '87 Conference Proceedings) pages 25-34.
17. Craig Reynolds (1999), "Steering Behaviors For Autonomous Characters", Proceedings of Game Developers Conference, pp. 763-782.
18. Programing Game AI by Exsample , Mat Buckland, Wordwave Pub, 2005.

이 은 복 (danalee@kau.ac.kr)
2008 한국항공대학교 컴퓨터공학과 학사
2008~ 현재 한국항공대학교 컴퓨터공학과 석사 과정

관심분야 : 모델링 및 시뮬레이션, 로봇 에이전트

신 석 훈 (ev4shin@naver.com)
2009 한국항공대학교 컴퓨터공학과 학사

관심분야 : 모델링 및 시뮬레이션, 로봇 에이전트

유 용 준 (ilog21c@kau.ac.kr)
2003 한국항공대학교 컴퓨터공학과 학사
2005 한국항공대학교 컴퓨터공학과 석사
2005～현재 한국항공대학교 컴퓨터공학과 박사 과정

관심분야 : 모델링 및 시뮬레이션, 워게임, 네트워크 보안

지 승 도 (sdchi@kau.ac.kr)
1982 연세대학교 전기공학과 학사
1984 연세대학교 전기공학과 석사
1985～1986 두산 컴퓨터 (현 한국 디지털) 근무
1991 미국 아리조나대학교 전기전산공학과 박사
1991～1992 미국 SIMEX Systems and S/W 회사 S/W 담당자로 근무
1992～현재 한국항공대학교 항공전자 및 정보통신공학부 교수

관심분야 : 이산사건 시스템 모델링 및 시뮬레이션, 컴퓨터 보안, 지능시스템 디자인 방법론, 시뮬레이션 기반 인공생명, 교통 모델링

김 재 익 (jaeick@add.re.kr)
1990 경북대학교 전자공학과 학사
1992 경북대학교 전자공학과 석사
1992~현재 국방과학연구소 선임연구원

관심분야 : 함정 전투체계 무장통제장치 개발, 전투체계 모델링 및 시뮬레이션, 전투체계 교전 효과도 분석