개선된 Floor Field 기반 보행 시뮬레이션 모델

(An Enhanced Floor Field based Pedestrian Simulation Model)

전 철 민

(Chulmin Jun)

요 약 실내 공간과 같이 마이크로한 스케일에서의 분석을 위해 다양한 보행시뮬레이션 모델들이 연구되어 왔으며, 이 중에는 social force 모델과 floor field 모델이 주목을 받는다. 이 중 연산이 복잡한 social force 모델보다는 CA 기반의 floor field 모델이 컴퓨터 시뮬레이션에 더 적합한 모델이라고 할 수 있다. 그러나 Kirchner 등이 제안한 floor field 모델에서는 dynamic field의 연산 시 자신의 dynamic 값에도 영향을 받을 수 밖에 없는 단점을 가지고 있으며, 본 연구에서는 이를 개선한 알고리즘을 제안한다. 본 연구에서는 dynamic field의 데이터 구조를 변경함으로써 자신의 dynamic 값은 배제한 다른 에이전트의 영향만을 받도록 하였으며, dynamic 값의 초기값을 할당하는 문제도 현실적으로 변경하였다. 본 연구에서 제시된 알고리즘을 테스트하는 데에는 공간 DBMS에 저장된 실제 3차원 건물모델을 사용하여 추후 실내 센서를 이용한 실시간 대피 시스템에 적용할 수 있는 기반이 되도록 하였다.

키워드 : 보행 시뮬레이션, floor field 모델, CA, multi-agent 시뮬레이션, 실내공간모델

Abstract Many pedestrian simulation models for micro-scale spaces as building indoor areas have been proposed for the last decade and two models - social force model and floor field model -are getting attention. Among these, CA-based floor field model is viewed more favourable for computer simulations than computationally complex social force model. However, Kirchner's floor field model has limitations in capturing the differences in dynamic values of different agents and this study proposes an enhanced algorithm. This study improved the floor field model in order for an agent to be able to exclude the influences of its own dynamic values by changing the data structure, and, also modified the initial dynamic value problem in order to fit more realistic environment. In the simulations, real 3D building data stored in a spatial DBMS were used considering future integration with indoor localization sensors and real time applications.

Keywords : Pedestrian simulation, Floor field model, CA, Multi-agent simulation, Indoor spatial model

1. 서론

실내 공간에서의 화재 대피나 건축물 안전 진단과 같은 분석을 위해 다양한 보행시뮬레이션 모델들이 연구되어 왔다[1]. Node-link 구조를 이용한 way-finding 방식의 매크로한 스케이일의 연구도 있으나, 실내 공간과 같이 마이크로한 스케일에서의 각 에이전트들의 동적인 특성을 모델링하는 데에는 social force model[2, 3]과 floor field model[4, 5]이 최근 주목을 받고 있다. Helbing 등에 의해 제시된 social force 모델은 엄격한 수학적 모델에 근거하여 공간내의 개체들간의 역학적인 힘을 표현하였으나 연산과정이 매우 복잡하고 보행자를 동질화시킨다는 단점을 가지고 있다[6, 7]. 이에 반해서 Kirchner 등에 의해 제시된 floor field 모델은 cellular 구조를 이용하고 인접 셀들에 대한 작용만으로 연산과정을 단순화 시켰음에도 불구하고 social force 모델과 유사한 결과를 보여서 컴퓨터 시뮬레이션에 유리한 모델로 주목받는다[7, 8].

Floor field 모델은 두 개의 field(static field와 dynamic field)를 이용하여 social force 모델의 역학적인 특성을 cellular automata 기반의 이동 규칙으로 변화시킨다. 이 두 개의 field 중 dynamic field는 생물학의 주화성(chemotaxis)과 유사한 특성을 나타내며 종종 개미가 짝짓기 때 분비하는 페로몬(pheromone)에 비유된다[7, 10]. Kirchner 등이 제시한 연구에서는 이 값이 다른 에이젼트로 향해서 움직이는 성향을 나타내며, static field 값과의 상대적인 크기에 따라 그 강도가 달라지게 된다[7, 8, 10].

본 연구에서는 이 dynamic field에 주목한다. 에이젼트들이 움직이면서 자신들의 주변에 dynamic field 값을 남기게 되고, 다른 에이젼트들이 이 값을 참조하여 주변 에이젼트들로의 이끌림을 형성하게 된다. 그러나 이전의 연구에서는 dynamic field 값을 셀에 할당할 때나 이 값을 참조할 때 어떤 에이젼트 자신이 남긴 값과 다른 에

† 이 논문은 2008년도 서울시립대학교 교내학술연구비에 의하여 연구되었습니다.

* 서울시립대학교 공간정보공학과 교수, cmjun@uos.ac.kr(교신저자)

논문접수 : 2010.02.15 수정일 : 2010.03.19 심사완료 : 2010.03.22

이전트의 값을 구분하지 않는다는 점과, dynamic 값의 초기화시 static 값과의 균형을 맞추지 않는다는 단점이 있었다. 이 때 단일 공간에서는 큰 문제가 없으나 실내공간이 분할되어 있고 좁은 입구를 통과할 때는 경우에 따라 흐름이 중단되어 버리는 문제가 있다는 것이 발견되었다.

본 연구에서는 이러한 dynamic field의 연산 문제가 해결된, 보다 개선된 floor field 기반의 보행 시뮬레이션 모델을 제시한다. 본 연구에서는 <key, value> 쌍의 컬렉션 (C#의 dictionary) 데이터 구조를 이용하여 각 에이전트의 dynamic value를 구분하여 저장하였으며, dynamic value의 초기값을 할당하는 문제에서도 보다 현실적으로 개선하였다. 본 연구에서 제시된 알고리즘을 테스트하는 데에는 공간 데이터베이스에 저장된 실제 3차원 건물모델을 사용하여 추후 실내 센서를 이용한 실시간 대피 시스템에 적용할 수 있는 기반이 되도록 하였다.

2. 관련 연구

대피관련 모델은 화재, 교통, 건축 등 다양한 학문적 영역에서 연구되어 왔으며 서로 다른 문제 영역과 목적을 위해 연구되었다. 이들은 대체로 매크로와 마이크로한 모델로 나누어 볼 수 있다[11]. 매크로 모델들은 보통 network flow나 traffic assignment 문제들에 적용된 연구들에서 나타나며, 최적화 접근방법을 사용하고 데이터 구조로는 node-link 기반의 graph 구조를 사용한다[12, 13, 14, 15]. 그들은 보행자들을 node나 link에 할당할 수 있는 동질한(homogeneous) 그룹으로 간주하며, 이동 중 개인간의 상호 작용은 고려하지 않는다. 그러나 보행자들이 자기 자신들간 또는 물리적인 환경과의 작용(예: Jamming, lane formation, oscillation 등)이 보행 흐름에 영향을 미치게 되며 실내 공간과 같은 공간을 분석하기 위해서는 매크로한 접근 방법은 적절치 않다.

마이크로 모델들은 개개 보행자들의 움직임과 그들이 다른 보행자나 물리적인 환경(벽이나 장애물과 같은)과 어떻게 상호작용을 하는지에 중점을 둔다. 이 모델들은 대체로 시뮬레이션 방법을 사용하며 시뮬레이션을 위한 기반 데이터 형식으로는 대상 공간을 세분한 그리드셀을 사용한다. 이들 모델들은 건축 디자인과 같은 영역에서 화재와 같은 비상시 건축구조가 사람들의 움직임에 미치는 영향을 분석하는데 흔히 사용되어 왔다.

지난 수십년간 다양한 마이크로 시뮬레이션 모델들이 연구되어 왔는데[16], 대체로 두 가지 모델, 즉, social force model과 floor field model이 주목을 받는다[5]. 전자는 주로 Helbing과 그의 동료들에 의해 연구되었으며[2, 3], 에이전트가 목적지(예를 들어 출구)를 찾아가는 과정을 계산하는데 매우 엄격한 수학적인 모델을 사용한다. Helbing의 모델은 각 에이전트가 공간 내의 다른 모든 에이전트들 및 물리적 환경과의 작용을 고려한다. 즉, 인접한 에이전트 뿐 아니라 원거리(long-range)의 모든 개체(보행자 및 기타 물리적 환경)와의 다대다 관계에 대해 모델을 하고 몇 겹의 편미분 방정식을 풀어야 하며 알고리즘 복잡도로서는 $O(n^2)$의 복잡도를 갖는다. 이는 많은 에이전트를 이용한 컴퓨터 시뮬레이션을 매우 불리하게 만드는 요소가 된다[6, 7, 8]. Helbing의 social force 모델에 대한 자세한 이론은 그의 연구를 참고하기 바란다.

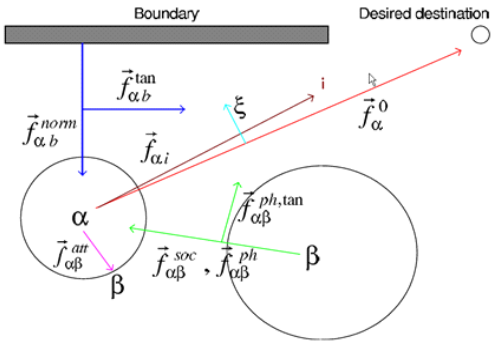


그림 1. Helbing의 social force model 개념

최근에는 마이크로 보행 시뮬레이션의 기반으로서 cellular automata를 이용하는 경향이 증가하고 있다[17, 18]. Kirchner와 그의 동료들은 CA 기반의 floor field 모델을 제시하였다[5]. 이 모델은 Helbing 등의 연구에서의 에이전트간의 원거리 상호작용(long-range interaction)을 로컬 상호작용으로 해석하여 변환하는데에 static field와 dynamic field라는 두 가지 field를 사용한다. 이렇게 로컬 작용만을 고려하는데도 불구하고, 결과로 나타나는 global한 현상은 social force 모델에서 보여주는 특징들, 즉, lane formation, oscillations at bottlenecks, faster-is-slower effects 등을 그대로 보여준다. Floor field 모델은 그리드 셀을 기반 데이터 구조로 이용하며, Helbing 연구와 같이 모든 에이전트의 다대다의 계산이 아닌, 에이전트의 인접한 셀들 중 다음 시간 단계에서의 에이전트들의 움직임만을 계산하므로 컴퓨터 시뮬레이션에 적용하는 데에 매우 효과적이다. 본 연구에서는 floor field 모델을 기반 모델로 하여 이 모델에서 사용하는 dynamic field의 연산과정을 개선한 알고리즘을 제시한다. 이에 대해서는 다음에 소개된다.

3. Floor Field 모델과 Dynamic field의 개선

3.1 Floor Field 모델에서의 두 가지 field

본 연구에서는 새로운 CA기반 모델을 개발하는 대신, 앞에서 소개한 Kirchner 등이 제시한 floor field 모델[5, 10]을 기반으로 하여 이를 개선하고자 한다. 이들이 제시한 floor field 모델은 앞서 소개한 여러 가지 보행자들의 특성을 적절히 반영한다는 것으로 보고되고 있으며, 연산면에서도 우수한 것으로 평가되고 있다. 먼저 floor field 모델의 기본적인 이론을 소개하고 본 연구에서 개선한

부분을 제시하고자 한다.

Floor field model은 기본적으로 multi-agent 기반의 시뮬레이션 모델이라고 할 수 있다. 보행자 하나 하나는 환경이나 다른 보행자와 상호 작용을 하는 에이전트이다. 이렇게 에이전트들은 다수가 모여 multi-agentsystem (MAS)을 이룬다. Multi-agent 시스템에서의 각 에이전트들은 다음과 같은 특징이 있다[19].

- Autonomy: 자율적이다. 스스로 다른 에이전트나 환경에 대해 인지하고 반응(움직이거나 정지)한다.
- Local view: 어떤 에이전트도 전체 시스템에 대한 global한 지식이 없다. 즉, 각 에이전트가 출구를 향해 어떻게 움직일 것인가에 대한 전역적인 룰이 있는 것이 아니라 로컬한 룰만으로 움직인다.
- Decentralization: 모든 에이전트들이 동등하며 어느 에이전트도 조정을 하지 않는다[20].

MAS에서 이러한 에이전트의 특징은 종종 cellular automata (CA) 기반의 시뮬레이션 모델을 써서 구현되며, Kirchner의 모델도 CA 기반으로 되어 있다. CA에 대해서는 다수의 문헌에 소개되고 있으므로 여기에서는 생략한다.

Kirchner 모델의 기본적인 데이터 구조는 그리드셀이며, 각 셀은 에이전트가 움직이는 위치이면서 값(value)을 저장하는 데이터 구조이다. 이 값은 두 개의 레이어로 분류하는데, 하나는 static field이고 다른 하나는 dynamic field이다. Static field는 출구로부터 각 cell까지의 최단거리를 가지는 field이다. 각 에이전트는 자기와 인접된 cell의static field 값으로서 출구로의 최단 방향을 알게 된다. Static field는 출구로부터 거리라는 고정된 (static) 값을 가지는 반면, dynamic field는 각 에이전트의 움직임에 따라 달라지는 값을 나타낸다.

Dynamic field는 개미가 짝짓기를 할 때 분비하는 페로몬(pheromone)과 같이[3] 각 에이전트가 자신의 영향권을 확산(diffuse)하고 소멸(decay)시키면서 움직인다. 한 에이전트는 근처의 다른 에이전트의 위치는 모르면서도 에이전트가 남긴 dynamic value를 참고하여 그 방향으로 따라가려는 성향을 갖게 된다. 에이전트의 static과 dynamic field에 대한 반응 정도를 변경함으로써 다양한 시뮬레이션이 가능하게 된다. 예를 들어, dynamic field에 대한 민감도를 높임으로써 공포(panic) 상황에서 다른 사람을 무조건 따라가는 행동을 실험할 수도 있다.

3.2 Floor field 모델의 Update rule

Floor field 모델에서 각 에이전트는 cell에 부여된 점수(score)에 따라 동시에 움직인다. Cell에 부여된 점수는 곧 그 곳으로의 유인력(attraction) 또는 이동 가능성을 나타내며, cell i의 점수는 다음과 같이 정의된다[8].

$$Score(i) = exp(k_d D_i) \times exp(k_s S_i) \times \xi_i \times \eta_i \qquad (1)$$

여기에서,

D_i: 셀 i의 dynamic field 값

S_i: 셀 i의 static field 값

k_d,, k_s: 에이전트가 D_i 또는 S_i값 중 어느 쪽에 더 민감하게 반응할지를 조정하는 조정 계수

ξ_i: 이동할 인접 셀의 벽, 장애물과 같은 물리적 요인을 나타냄. 1인 경우 이동가능하며 0인 경우 이동 불가함

η_i: 인접 셀의 다른 보행자의 점유 상태를 나타냄. 1인 경우 이동 가능하며 0인 경우 이동 불가함

한 에이전트의 cell-i에서 인접된 셀(j)로의 움직임은 다음과 같은 p_{ij} 확률값으로 결정하며, 이는 모든 아홉개 cell에 대한 Score(i)의 normalized된 값이다. 하지만 score(i)나 p(i)는 인접한 아홉 cell에 대해서는 항상 서로 비례하므로 어떤 것을 사용하던 동일한 효과를 보인다.

$$P_{ij} = Score(j)(\Sigma_{a \in N(i)} Score(a))^{-1} \qquad (2)$$

출구로부터 거리를 기반으로 한 static field가 먼저 결정된다. 이는 Dijkstra나 A* 알고리즘과 같은 최단거리 알고리즘을 사용하는 것으로 해결할 수 있다. 다음으로는 모든 에이전트들이 자신이 움직여 갈 cell을 결정하게 되며, 그 후 공간 내의 에이전트들이 동시에 움직이게 된다. 이 과정은 그림 2와 같이 수도코드(pseudocode)로 나타낼 수 있다.

```
O = Φ                  // Set the open list to empty set
P(i) = null            // Set the parent node of the beginning node to null
s(i) = 0               // Set the score of the beginning node to 0
O = (b)                // Add the beginning node to the open list
d(b) = 0               // Set the dynamic value of i to 0
i = b                  // Set i to b
While (i ≠ E)          // Iterate while node-i is not the destination node
{
   // Choose the maximum score node among the open list.
   Let  i ∈ O  be a node for which
   p(i) = min{ s(i) : i ∈ O } and s(i) > 0

   // If the agent- i has moved to a node other than itself
   if( i ≠ P(i) )
   {
      // increase the dynamic value of the previous node by one
      d(p(i)) = d(p(i)) + 1
      for each ( p(i), j ) ∈ A(p(i))
      // j: Available adjacent nodes of p(i)
      {
         // define the dynamic diffuse amount of j
         // as the α-portion of the dynamic value of the previous
         // node divided by the number of the adjacent nodes
         d_diff = α*d(p(i))/N (A(p(i)))
           d(j) = d(j) + d_diff
      }
      // to decay, the previous node discards β portion
      // of the dynamic value
      d(p(i)) = d(p(i)) - β*d(p(i))
      O = Φ        // Reset the open list to empty set
      // For each of searchable adjacent nodes of i (i.e. including
      // i and excluding those obstacles as walls and furniture)
      // set parent, and add to the open list
      for each (i, j) ∈  A(i)     // j: Adjacent nodes of i including i
      {
         // If not in the open list, add j to it
         if  j∉O then O = O ∪ (j)
         P(j) = i       // Set the parent node of j to i
      }
   }
}
```

그림 2. Floor field 모델에서의 에이전트의 움직임을 나타내는 pseudocode

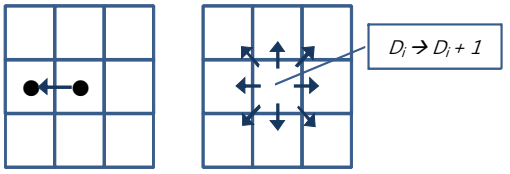


그림 3. Dynamic value의 diffuse & decay

그림 2의 pseudocode와 그림 3에 나타낸 바와 같이 에이전트가 어떤 셀에서 다른 셀로 움직인 후에는 원래의 셀 i의 dynamic value를 1만큼 증가시킨다[4, 10]. 그 다음에는 그 셀의 dynamic value를 인접셀로 일정 부분(α) 나누어서 인접셀들의 dynamic 값들에 더한다. 다음으로는 Di의 값을 일정 부분(β) 감소시키게 된다. 이는 마치 개미가 움직이면서 자신의 냄새를 확산시킨후 점차 묽어져서 사라질 때까지 흔적을 길게 남기는 효과를 준다. 에이전트들은 이와 같은dynamic 값을 static값과 더불어 함께 참조한다. 이 두 개의 값 중에 더 강한 값에 움직이는 확률을 높이기 위해 위에서 언급한 scaling factor(k_d와 k_s)를 사용한다. 이 두 factor의 비율, k_d/k_s은 공포감(panic)의 강도로 해석된다. 즉, 이 비율이 클수록 한 에이전트가 다른 에이전트로 향하는 정도가 강하게 된다.

4. Dynamic field의 개선

4.1 Dynamic field 저장구조의 개선

Kircher 등[5]의 floor field 모델에서의 dynamic field는 인접 셀만이 아닌 Helbing 모델에서의 원거리(long-range)의 에이전트들에 대한 영향을 포함시킬 수 있는 효과적인 수단이라고 할 수 있다. 그러나 위와 같이 dynamic 값을 정하게 되면 어떤 셀의 dynamic 값은 자신이 남긴 값과 다른 에이전트들이 남긴 값들이 섞여서 더해지게 되는 문제가 있다. 개미의 페로몬(pheromone)과 같이 다른 에이전트들의 영향에 반응하여 그 쪽으로의 인력을 작용시키기 위해서는 자신의 영향을 배제해야 하는 것이 당연하다. 왜냐하면 자신의 냄새에 반응하지는 않을 것이기 때문이다. 하지만 Kirchner의 모델에서는 자신과 다른 에이전트의 dynamic 값을 구분하지 않고 자신의 dynamic 값을 다른 에이전트의 값에 누적해서 더하는 방식을 쓰고 있다. 이렇게 되면, 어떤 에이전트가 한 방향으로만 움직이게 된다면 문제가 없지만, 그림 4와 같이 제 자리로 돌아오는 등의 움직임이 발생했을 때는 결국 자신이 남긴 dynamic value의 영향을 받을 수 밖에 없게 된다. 이는 셀별로 에이전트들의 dynamic 값을 구분하지 않아서 생기는 문제이다. 그림 5에서는 이를 간단한 숫자를 이용하여 예시하였다. 여기에서는 예시를 단순화시키기 위해 static 값은 배제하였고, diffuse와 decay 계수를 모두 0.1로 하였다.

본 연구에서는 dynamic field를 자신의 영향을 제거한 페로몬 효과를 부여할 수 있도록 개선시켰다. 이를 위해

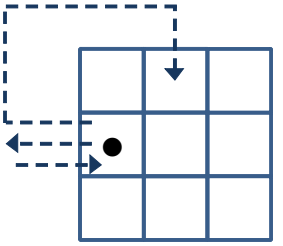


그림 4. 자신의 dynamic value에 영향을 받는 문제

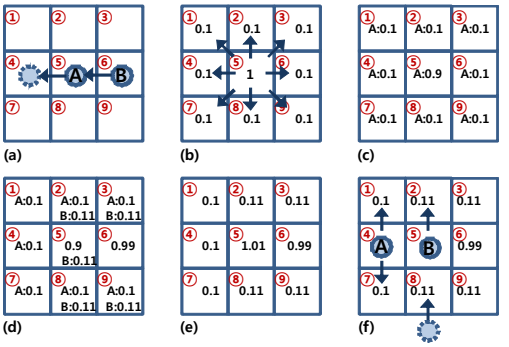


그림 5. 자신의 dynamic value에 영향을 받는 문제의 예시

((a) Agent A와 B가 그림과 같이 이동. (b) 이동 후 ⑤위치에 D값이 1증가하고 α 만큼씩 diffuse(이전 D값은 없고 α=β=0.1이라고 가정함) (c) ⑤셀의 D는 β만큼 decay (d) Agent B도 동일하게 diffuse & decay함 (e) 움직임이 끝난 후 최종 D값 (f) Agent B가 ②셀을 택했고, 또다른 제3의 Agent가 ⑧셀을 택했을 경우를 가정하면, Agent A는 ① 또는 ②번 셀을 택할 수 밖에 없음)

서는 각 셀에서 에이전트 마다의 dynamic value를 보관하는 구조를 가져야 한다. 이 때 셀 k에서의 에이전트 p의 dynamic value는 d(p, k)라고 하고 이들 값의 집합을 D(k)라고 한다.

$$D(k) = \{d(p, k) : p=1, 2, \cdots, n\} \quad (3)$$

여기에서 n은 셀 k에 dynamic이 유지되고 있는 에이전트의 개수이다. 식 (3)에서 D(k)를 유지하기 위해 모든 셀이 모든 에이전트에 대해 값을 유지하는 O(n^2)의 복잡도를 가진다고 생각하기 쉬우나, 실제로는 움직임 과정 중 셀 k에 dynamic 값의 영향을 준 에이전트 들의 리스트만 유지하므로 각 셀은 매우 적은 개수의 값들만 유지하게 되어, 연산과정에도 무리가 없게 된다. 실제로 구현시에는 .NET의 Dictionary라는 데이터구조를 이용했는데, Dictionary는 (key, value) 쌍으로 된 리스트이다. 여기에서 key는 에이전트 p를 의미하고 value는 dynamic value가 된다.

이 때, d(p, k)는 p 에이전트 자신의 dynamic value 중 항상 최대값 하나로만 유지하도록 한다. 개미 주변에 초기에 가장 강한 pheromone이 발생한다고 한다면, 위치

를 옮겨감에 따라 원래 위치의 pheromone이 약해져 가다가도 다시 그 위치로 돌아올 때 강한 pheromone이 다시 발생하는 것과 같은 원리라고 할 수 있겠다.

$$d(p, k) = max\{d(p, k): p=1, 2, \cdots, m\} \quad (4)$$

여기에서 m은 에이전트 p가 cell k에 dynamic 값을 남긴 회수이다. 실제로 구현시에는 자신의 m개의 값을 보관하는 것이 아니라 항상 최대값으로 update되도록 한다. 또한 score(k)를 계산할 때는 D(k) 중 에이전트 q 자신의 값은 제외한 값 중 최대값을 취해서 자신의 dynamic 값에 영향을 받지 않도록 한다.

$$D(k)_q = max\{d(p, k): p \neq q\} \quad (5)$$

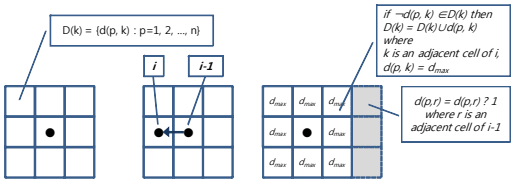


그림 6. Cell k의 dynamic 값의 리스트(a), 에이전트의 움직임(b), 움직임 후의 dmax 값의 할당(diffusion) 및 이전 셀들의 쇠퇴(decay)과정(c)

4.2 Dynamic field 초기값 연산의 개선

본 연구에서는 에이전트 p가 움직였을 때 초기의 p의 dynamic value 값을 정하는 방법도 Kirchner의 모델과는 다른 방법을 사용했다. 그림 6-(c)를 보면, 에이전트가 움직였을 때 가장 강한 dynamic value를인접셀에 할당하고 이전 셀에서 에이전트 p가 가지고 있는 dynamic 값들을 1씩 줄인다.

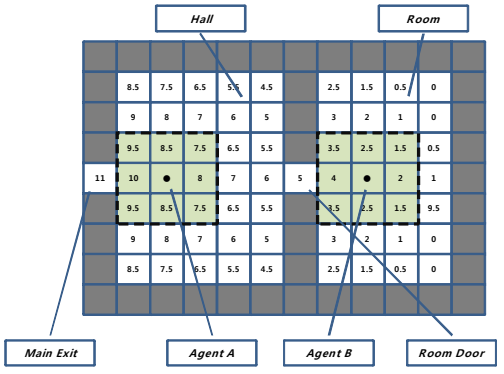


그림 7. 분할된 공간을 가진 건물 평면에서의 Dynamic 초기값 설정의 문제

그림 7은 건물의 외부로 나가는 주 출입문이 있고, 그 안에 방과, 방을 출입하는 문이 있는 것을 단순화 시킨 그림이다. 이 때, 에이전트 A와 에이전트 B가 그림과 같이 위치해 있다고 가정한다. 셀에 부여된 숫자는 주 출입구로부터의 거리, 즉, static field 값의 예시를 위해 단순한 값을 사용해서 나타낸 것이다. 이 때 초기에 부여되는 d_{max} 값을 정하는 것이 문제가 된다. Kirchner 모델에서는 dynamic value가 static value에 비해 상대적으로 지나치게 커질 수 있다는 점이 발견되었다. 저자의 실험에서는 단일 공간에서의 시뮬레이션에서는 큰 문제가 없었으나, 그림과 같이 내부 공간이 분할되어 있고, 좁은 출입구(단일 셀로 되어 있는)가 내부 공간 내에 위치해 있는 세팅에서는 문제가 발생되었다.

Static value는 주 출입구에서부터 점진적으로 역으로 부여되기 때문에 건물의 크기에 따라서는 내부에 있는 방들은 상대적으로 매우 작은 static 값들을 가질 수 있다. 예를 들어, 주 출입구에서 멀리 떨어져 있는 셀에서는, 에이전트 A 주변의 static 값에 비해, 에이전트 B 주위의 static value는 작은 값을 갖게 된다. 단일 공간이라면, dynamic 초기값을 모든 셀에 일정하게 할당하더라도 점진적인 static value로 인해서 출구로의 방향성을 갖는다. 하지만, 이렇게 분할된 공간에서는 일차적으로 내부의 방에서 빠져나오는 것이 목적인데, 방 출입구 주변의 static 값이 작기 때문에, 이 값보다 dynamic 값이 지나치게 커지게 되면 출입구를 빠져 나오지 못하고 dynamic 값만을 쫓는 상황도 발생하게 된다.

그림 8은 이렇게 흐름이 중단되어 버리는 문제를 예시한 것이다. Dynamic 값은 diffuse와 decay 계수, α, β를 대체로 [0, 1] 범위에서 부여한다. 매우 작은 값이라 하더라도 이 값이 누적이 되어 전방의 Static값과의 합(S+D)보다 후방의 (S+D)값이 커지게 되면, 그림 8의 Agent A는 그림과 같은 상황에서 전방으로 빠져나가지 못하고 제자리에 있던지 후방 셀과 자신의 자리를 반복해서 맴돌게 된다.

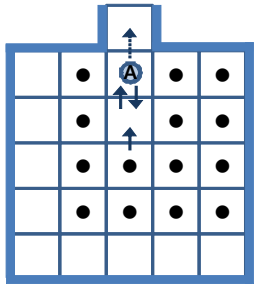


그림 8. 좁은 출입구(1개의 셀로된)를 통과할 때의 흐름 정지문제

본 연구에서는 이를 개선하기 위해서 에이전트가 움직였을 때 갖는 초기 dynamic 값을 인접 셀의 static value 중 최대 값을 갖도록 하였다. 이렇게 함으로써 공간의 크기나 분할공간 유무에 상관없이 dynamic value는 static 값에 대체로 비례하는 값을 갖게 된다. 개미의 pheromone에 비유한 것만 고려한다면, 어떻게 개미의 냄

새가 출구에 가까워 질수록 비례해서 강해질 수 있느냐는 의문을 가질 수도 있다. 그러나, 식 (1)에서 보면, 어차피 S와 D는 서로 다른 단위의 값을 exponential 함수의 지수로 사용한 것을 보면 이해할 수 있다. 각 에이전트는 global한 상황에 대해서는 모르는 채, 로컬 룰에 따라서만 움직이게 되어 있고, S와 D가 현재 로컬 영역에서의 에이전트에 대한 "인력"이라고 볼 때, 서로 비슷한 값을 사용하고, 이를 scaling parameter로 민감도를 조절하는 것이 타당하다고 할 수 있다.

그림 9는 이렇게 개선된 Dynamic 값의 연산을 포함하여 수정한 pseudocode이다. 코드의 주석은 변경된 부분에 대해서만 추가하였다.

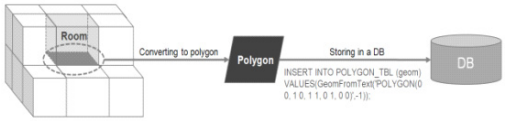
```
O=Φ
D(k)=Φ  for k ∈ N  // Set the Dynamic list for each node to empty set
P(b) = null
s(b) = 0
O = (b)
i = b
While (i ≠ E)
{
    Let  i ∈ O  be a node for which
    s(i) = max{ s(i) : i ∈ O } and s(i) > 0
    iff( i ≠ P(i) )
    {
        // For each node in the open list, if the node contains
        // the agent p's dynamic value, decrease it by one
        for each k ∈ O
            if d(p, k) ∈ D(k) then d(p,k) = d(p,k) - 1

        O=Φ
        for each ( i, j ) ∈ A(i)
        {
            if j∉O then O = O∪(j)
            P(j) = i
        }
        // Get the maximum static value of among those of
        // open list nodes
        dmax = max{ t(k) : k ∈ O }
        // For each node in the open list, if the dynamic list does
        // not contain  the dynamic value of node k, add it to D(k)
        for each k ∈ O
        {
            d(p, k) = dmax
            if d(p, k)∉D(k) then D(k) = D(k)∪d(p, k)
        }
    }
}
```

그림 9. 개선된 dynamic field 연산 규칙을 적용한 pseudocode

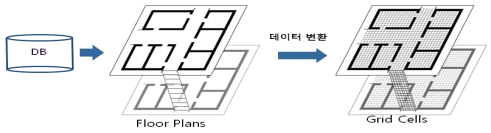
5. 적용 테스트

5.1 DBMS 기반의 3차원 모델 적용

본 연구에서는 시뮬레이션 테스트를 위해 공간 DBMS에 저장된 실제 3차원 건물을 사용하였다. 이는 본 연구가 시뮬레이션 차원에만 머무는 것이 아니라 추후에 실내 센서와 연동이 될 경우 실시간으로 파악된 실내 거주 보행자에 따라 시뮬레이션 결과를 이용하여 구조나 대피에 활용할 수 있도록 함이다. 이는 기존의 이론적인 연구들에서 보여지는 단순한 가상의 사각형 공간을 이용한 실험과는 차별이 되는 부분이다. 그러나 알고리즘의 개선에 초점을 둔 본 연구에서는 이와 같은 데이터 구축과정과 이론적 배경에 대해서는 간단하게만 언급하도록 하겠다.



(a) 각 3차원 공간의 floor surface를 이용하여 DB에 저장



(b) 저장된 데이터의 추출과 grid cell로의 변환

그림 10. 공간 DBMS를 이용한 3차원 건물 데이터 저장(a)과 시뮬레이션을 위한 grid cell로의 변환(b)

그림 10에 나타난 바와 같이 본 연구에서는 각 공간의 바닥면(floor surface)을 공간 DBMS에 저장하였으며 DBMS로는PostGIS를 사용하였다. 보행자의 이동은 바닥면을 통해서만 이루어지므로 바닥면만을 이용하더라도 시뮬레이션에는 문제가 없다. 또한 이 바닥면 정보를 이용하여 공간간의 토폴로지와 semantic 정보를 저장할 수 있다. 계단 부분은 몇 개의 연속된 폴리곤의 집합으로 정의하여 저장하였다. 시뮬레이션 실행 시에는 이 저장된 정보를 쿼리로 얻어온 후 그리드 셀로 변환한다. 본 연구에서는 실제 사람의 어깨 폭을 고려하여 40㎝×40㎝의 셀들로 이루어지도록 하였다.

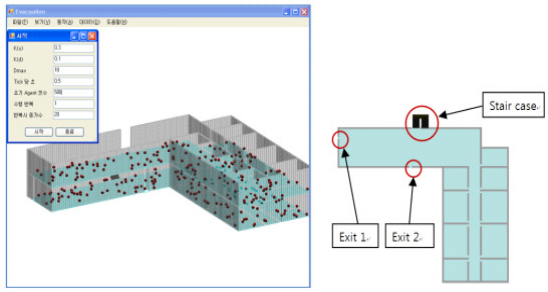
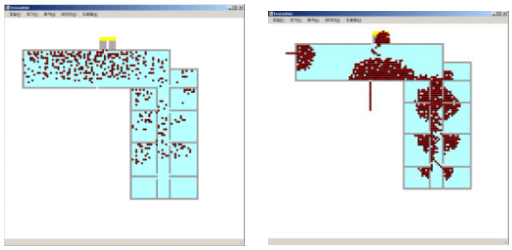


그림 11. 시뮬레이션에 사용된 건물 데이터

그림 11은 본 연구에서 사용된 2층 구조의 캠퍼스 건물이다. 그림에서와 같이 두 개의 주 출입구와 하나의 계단으로 구성되어 있다. 본 연구에서는 시뮬레이션을 위해 C# 언어와 OpenGL을 이용하여 간단한 인터페이스를 개발하였다. 시스템에서는 먼저 PostGIS로부터 데이터를 읽어들여서 그리드 셀로 변환하게 된다. 이 때 2차원 및 3차원으로 가시화가 가능하며 k_s, k_d와 같은 scaling factor 및 시간 스텝, 반복 회수, 에이전트의 개수 및 반복당 증가수 등의 파라미터들을 조정할 수 있다.

5.2 시뮬레이션

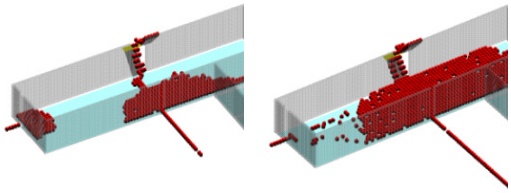
시뮬레이션이 시작되면 우선 두 개의 주 출입구로부터 각 셀들까지의 최단 거리값에 근거한 static field가 계산된다. 본 연구에서는 k_s와 k_d의 비율을 다르게 하면서 실험하였다. 그림 12는 두 가지의 극단적인 경우, 즉, k_s=0과 k_d=0인 경우를 예시한다. 쉽게 예상할 수 있듯이 k_s=0은 출구로의 방향성이 전혀 없이 주변의 에이전트들만을 쫓아서(herding behavior) 공간을 방황하는 결과를 낳는다. 한편, k_d=0은 다른 에이전트를 쫓는 효과는 전혀 없이 곧장 출구를 향해 움직여 가는 효과를 가져온다.



(a) (b)

그림 12. 두 가지 극단적인 경우에서의 에이전트들의 움직임 k_s = 0(a) and k_d = 0(b)

그림 13은 k_d값을 변화시키는 데에 따른 효과를 보여준다. k_d=0은 에이전트들 자신들 위치의 static 값에 따라 해당되는 근접한 출구로 직접 유도하게 되며, k_d>0은 쫓는 행태(herding behavior)를 보이기 시작한다. 그림 12-(b)에서와 같이 주 출입구를 향해 몰려 있던 에이전트들 중 몇 명이 그룹을 이탈하게 되면, 이 에이전트들을 쫓아서 다른 에이전트들도 그룹을 이탈하는 것이 관찰되었다. 이들은 왼쪽에 보이는 부 출입구를 통해 건물을 빠져나간다.



(a) (b)

그림 13. k_d값의 변화에 따른 효과 k_d = 0(a)과 k_d/k_s = 0.5(b)

표 1은 이렇게 k_d값의 변화에 따라 그룹의 herding behavior 정도를 측정한 결과이다. 표에서는 k_d값의 변화에 따라 왼쪽 출입구(Exit 1)를 사용한 탈출 인원수와 공간 내의 전체 에이전트들이 모두 대피하는 데에 걸린 시간을 보여준다. 실험에는 2000명의 에이전트들이 사용되었다. 예상대로 k_d값의 증가에 따라 Exit 1의 사용이 증가했으며, 두 개의 출입구를 사용하는 정도가 증가한 만큼 전체 대피 시간도 줄어드는 것이 관찰되었다. 그러나 이는 출구의 폭이나 공간 구조와도 상관이 깊다. 예를 들어 주 출입구(Exit 2)의 폭이 부 출입구(Exit 1)보다 상당히 클 경우에는 이와 같이 대피시간을 줄이는 효과는 상대적으로 떨어지게 된다.

표 1. k_d의 변화에 따른 herding behavior 및 대피시간 감축 효과

	k_d=0	k_d=0.05	k_d=0.1	k_d=0.25	k_d=0.5	k_d=1.0
Exit1	120	351	422	484	566	689
Exit2	1880	1649	1578	1516	1434	1311
evactime	945	723	702	688	670	632

또 다른 실험은 에이전트들을 증가시키고, k_d/k_s비율을 증가시키면서 실시하였다. 그림 14에서 보여주는 것과 같이, 실험에서는 여섯 개의 (k_d, k_s) 쌍, (0, 0.3), (1, 0.1), (0.25, 0.5), (0.1, 0.1), (0.05, 0.1), (0.1, 1)이 이용되었다. 또한 에이전트들은 500~5000의 범위에서 증가시키면서 실시하였다. 실험에서는 k_d=0의 경우 시간당 대피 인원이 거의 비례하여 증가하였으며, k_d>0인 값들에 대해서는 상호 큰 차이를 보이지는 않았다. 그러나 결과에서 보여주는 바와 같이 k_d>0인 값들은 확실하게 일부 에이전트들을 대안 출구로 유도하게 되어 시간당 대피 인원을 줄이는 효과를 가져오는 것을 알 수 있다.

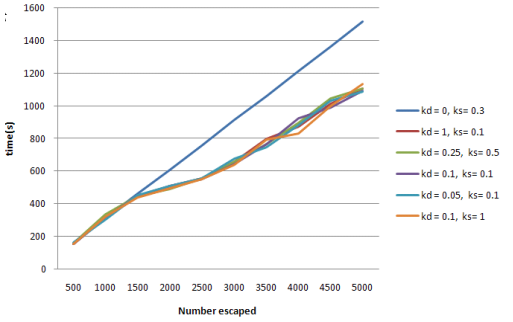


그림 14. 몇 가지 다른 k_d/k_s들을 이용한 시간당 대피 인원수

5. 결론

본 연구에서는 floor field를 기반으로 한 multi-agent 보행 시뮬레이션 모델을 개선시키는데 중점을 두었다. Kirchner의 모델은 에이전트를 구분하지 않고 로컬 룰에 따라 자율적으로 움직인다는 기본 원칙에는 충실하지만, 자신의 dynamic 값에도 영향을 받을 수 밖에 없는 단점을 가지고 있다.

본 연구에서는 dynamic field의 데이터 구조를 변경함으로써 개미의 pheromone과 같이 자신의 dynamic 값은 배제한 다른 에이전트의 영향만을 받도록 하였다. 또한

자신의 dynamic 값의 최대값을 static 값에 비례하게 줌으로써, 공간의 복잡도에 상관없이 두 값의 영향이 항상 유사하도록 하고, scaling factor로 조절이 용이하도록 하였다.

본 연구는 다양한 에이젼트 속도를 다르게 준다던지, 출구로의 가시성(visibility)을 포함시키는 등 모델의 개선도 지속적으로 수행하고 있다. 또한 공간 DBMS를 적용한 실제 데이터와의 연동에도 목적을 두고 있다. 여기에서는 간단한 2층 구조의DBMS를 이용하여 랜덤 데이터를 이용한 시뮬레이션에 초점을 두었다. 그러나 추후에는 실내 위치센서와의 연동을 통해 건축 공간에서 실시간으로 파악된 사람들의 분포를 이용할 수 있다. 이렇게 되면 실시간으로 수집된 특정 공간내의 사람들이 대피하는 데에 지나치게 긴 시간이 예상된다면(이는 시뮬레이션 결과, 즉, 시간당 대피인원을 이용한다) 대안 대피로를 안내하거나 구조 활동 시 우선적인 구조 장소와 루트를 안내하는 데에 이용될 수 있다.

참 고 문 헌

[1] Schreckenberg, M and Sharma, S.D. (Eds.), Pedestrian and Evacuation Dynamics, Springer-Verlag, Berlin, 2001.

[2] Helbing, D., and Molnár, P., “Self-organization phenomena in pedestrian crowds,” In F. Schweitzer (Ed.), Self-Organisation of Complex Structures: From Individual to Collective Dynamics, Gordon & Beach, London, UK, 1997.

[3] Helbing, D., Farkas, I., Molnár , P., and Vicsek, T., “Simulation of pedestrian crowds in normal and evacuation situations,” In M. Schreckenberg and S. Sharma, (Eds.), Pedestrian and Evacuation Dynamics, Springer-Verlag, Berlin, 2001, pp. 21-58.

[4] Burstedde, C., Klauck, K., Schadschneider, A., and Zittartz, J., “Simulation of pedestrian dynamics using a two-dimensional cellular automaton,” Physica A, Vol.295, 2001, pp. 507 - 525.

[5] Kirchner, A., and Schadschneider, A., “Simulation of evacuation processes using a bionics-inspired cellular automaton model for pedestrian dynamics,” Physica A Vol.312, 2002, pp. 260-276.

[6] Apel, M., “Simulation of pedestrian flows based on the social force model using the Verlet Link Cell Algorithm,” Master Thesis, Poznan University of Technology, Poland, 2004.

[7] Henein, C. M., and White, T., “Macroscopic effects of microscopic forces between agents in crowd models,” Physica A, Vol.373, 2007, pp. 694-712.

[8] Henein, C. M., and White, T., “Agent-Based modelling of force in crowds,” MABS 2004, LNAI, 3415, 2005, pp. 173-184.

[9] Ben-Jacob. E., “From snowflake formation to growth of bacterial colonies. II: Cooperative formation of complex colonial patterns,” Contemporary physics, vol.38 no.3, 1997, pp. 205-241.

[10] Nishinari, K., Kirchner, A., Namazi, A, and Schadschneider, A., “Simulations of evacuation by an extended floor field CA model,” In Traffic and Granular Flow ’03, Spinger-Verlag, Berlin, 2005, pp. 405-410.

[11] Hamacher, H. W., and Tjandra, S. A., “Mathematical modelling of evacuation problems: a state of art,” In M. Schreckenberg and S. Sharma, (Eds.), Pedestrian and Evacuation Dynamics, Springer-Verlag, Berlin, 2001, pp. 227-266.

[12] 김종대, 원정임, 김상욱, "도로 네트워크에서 이동 객체의 과거 궤적 분석을 통한 미래 경로 예측," 한국공간정보시스템학회 논문지, 제8권 제2호, 2006, pp.109-120.

[13] 김영진, 김영창, 장재우, 심춘보, "공간 네트워크 상의 이동객체를 위한 시그니처 기반의 궤적 색인구조," 한국공간정보시스템학회 논문지, 제10권 제3호, 2008, pp.1-18.

[14] 김용기, 김아름, 장재우, "공간 네트워크 데이터베이스에서 공간 제약을 고려한 경로 내 최근접 질의처리 알고리즘," 한국공간정보시스템학회 논문지, 제10권 제3호, 2008, pp.19-30.

[15] 황정래, 강혜영, 이기준, "도로 네트워크 상의 이동 객체 궤적의 간략화," 한국공간정보시스템학회 논문지, 제9권 제3호, 2007, pp.51-65.

[16] Schadschneider, A., “Cellular automaton approach to pedestrian dynamics - Theory,” In M. Schreckenberg and S. Sharma, (Eds.), Pedestrian and Evacuation Dynamics, Springer-Verlag, Berlin, 2002, pp. 75-86.

[17] Blue, V. J., and Adler, J. L., “Using cellular automata microsimulation to model pedestrian movements,” In A. Ceder (Ed.), Proceedings of the 14th International Symposium on Transportation and Traffic Theory, Jerusalem, Israel, 1999, pp. 235-254.

[18] Klupfel, H., Konig, T., and Wahle, J., and Schreckenberg, M., “Microscopic simulation of evacuation processes on passenger ships,” In Proceedings of Fourth International Conference

on Cellular Automata for Research and Industry, Oct. 4-6, Karlsruhe, Germany, 2002.

[19] Wooldridge, M., An Introduction to Multi Agent Systems, John Wiley & Sons, 2002.

[20] Panait, L. and Luke, S., "Cooperative multi-agent learning: the state of the art," Autonomous Agents and Multi-Agent Systems, Vol.11 No.3, 2005, pp. 387-434.

[21] Bonabeau, E., Dorigo, M., and Theraulaz, G., Swarm intelligence: From natural to artificial systems. Oxford University Press, New York, 1999.

전 철 민

1988년 서울대학교 도시공학과 (공학사)
1990년 서울대학교 도시공학과(공학석사)
1997년 Texas A&M 대학교 도시및지역계획학 (공학박사)
1997년~1999년 North Carolina RTI GIS 전문요원
1999년~현재 서울시립대학교 공간정보공학과 교수
2008년~현재 서울시립대학교 공간정보연구센터 센터장
관심분야는 GIS, 공간 데이터베이스, 3차원 모델, network algorithm 등