P-11

VR기반 출구 탈출 시뮬레이션 설계에 관한 연구

이범종, 박종승, 이동호*
*인천대학교 컴퓨터공학과 ,인천대학교 안전공학과**

A study on VR based simulation design of exit evacuation

Bum-Jong Lee, Jong-Seung Park, Dong-Ho Rie*
Department of Computer Science Engineering, University of Incheon
*Department of Safety Engineering, University of Incheon**

1. 서 론

초고층 및 초대형 건물은 특히 방대한 규모로 인하여 필연적으로 많은 거주자 및 방문자들이 발생한다. 따라서 초고층 및 초대형 건물의 건설에 있어서, 그 계획 단계에서 해당 건물의 잠재적 사용자의 공간이용 패턴에 대한 예측 평가는 핵심적이고 긴요한 계획 요소이다. 특히 화재 등의 재난이 발생하였을 때에 해당 건물 이용자의 피난이 얼마나 신속하고 용이하게 이루어질 수 있는지에 대한 문제는 초고층 및 초대형 건물의 방재 계획과도 직결되는 매우 중대한 계획 요소이다.

초고층 및 초대형 건물의 대피 계획을 수립하는데 있어서, 피난 성능에 대한 평가 도구는 화재안전 규정이 '사양규정'에서 '성능규정'으로 바뀌는 전 세계적 추세에 맞게 여러 이론과 방법론의 기반하여 다양하게 개발되고 있다. 이들 중 대부분은, 화재 발생 시에 사람들의 이동형태를 시뮬레이션하고, 그 결과로 피난 성능을 평가하는 방식의 도구들이다. 기존의 피난 시뮬레이션 도구들로는 SIMULEX[1], EXODUS[2], EXITT[3], BuildingEXODUS[4] 등이 대표적이다. 이 도구들은 대부분 최종 대피 완료 시간을 중심으로 대피 성능을 평가한다. 이러한 평가 방식은 해당 건축물의 전반적인 피난 성능을 판단할 수 있는 지표로 활용할 수는 있으나 실제 환경은 3차원으로 이루어져 있기 때문에 이러한 3차원 공간에 대한 피난 효율의 지표로 활용할 수는 없는 담점이 있다. 이는 기존의 피난 시뮬레이션 도구들이 재난 발생 시 이용자 및 거주자의 형태 재현에 초점을 맞추고, 이에 대한 공간 구조적 영향에 대해서는 따로 다루고 있기 때문이다.

본 연구에서는 이러한 문제점들을 해결하기 위해 3차원 그래픽 렌더링 기술을 활용하여 화재 발생 시 사람들이 제한된 탈출구를 통해 효율적이고, 안전하게 탈출할 수 있는 경로를 찾을 수 있는 시뮬레이션 시스템을 제안한다. 경로를 찾기 위한 알고리즘으로는 Best-first search 알고리즘[5]과 Dijkstra 알고리즘[6], Johnson 알고리즘[7], Bellman-Ford 알고리즘[8] 등이 있으며 본 연구에서는 사람들의 현재 위치에서 탈출구까지의 경로를 찾기 위해 Hart 등에 의해 처음 소개되어진 A* 알고리즘을 사용하였다[9]. 실제 대피환경과 동일한 대피조건을 구현하기 위하여 문에 도달한 사람은 무조건

탈출한 것으로 간주하지 않고, flow rate에 의해 문을 통과할 수 있는 사람의 수와 문의 크기에 따라 설정하였다.

2. 대피 시뮬레이션 시스템 설계

시스템 설계 순서는 셀 단위로 공간에 대해서 편집하고, 남성, 여성, 노인, 아이 등에 대한 사람을 입력한 후, 계단과 출구를 편집한다. 편집한 후에 시뮬레이션을 시작하게 되며 각 사람에 대해 경로를 찾게 되고, 찾은 경로를 따라 출구로 진행하게 된다. 제안된 시뮬레이션 시스템의 전체 흐름도는 Fig.1과 같다.

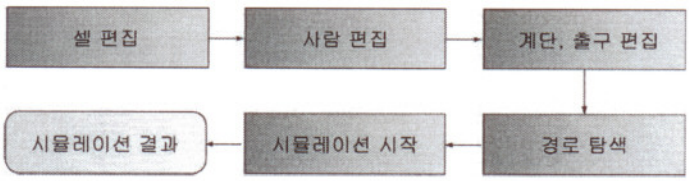

Fig. 1 제안된 시뮬레이션 시스템의 전체 흐름도

2.1 A* 알고리즘

사람의 대피 상황을 시뮬레이션하기 위해서는 각 사람의 대피 경로를 계산하는 알고리즘이 필요하다. 본 연구에서는 인공지능 이론에서 흔히 사용되는 A* 알고리즘을 최적 경로 계산에 도입할 수 있다. A* 알고리즘는 두 지점 사이의 최적 경로를 신속히 찾을 수 있는 장점이 있다. A* 알고리즘은 가장 적절한 탐색 방향을 평가하며, 뒤로 돌아와서 다른 수단을 강구하기도 한다.

A* 알고리즘의 적용을 위해서는 맵 상의 위치를 표현하는 자료구조인 노드(node)를 정의해야 한다. 또한 탐색되는 노드의 적합성을 평가하기 위해 거리(distance)와 휴리스틱(heuristic)을 정의해야 한다. 거리와 휴리스틱의 정의를 위해서는 노드의 비용(cost)을 목적에 맞게 추정할 수 있도록 해야 한다. A* 알고리즘의 목표는 비용이 가장 낮은 경로를 찾는 것이다. 비용의 정의는 응용에 따라 다르게 정의될 수 있으므로 어떻게 비용을 정의하는지가 중요하게 작용한다. 본 연구에서는 노드의 속성을 f, g, h로 정의한다. g는 목표(goal) 노드를 의미하고 시작 노드에서 현재 노드까지 오는데 드는 비용에 해당한다. 여러 경로 중에서 특정한 한 경로의 비용이다. h는 휴리스틱(heuristic)을 의미하고 현재 노드에서 목표 노드까지 가는데 드는 추정된 비용을 의미한다. 아직 최단 경로를 모르고 있기 때문에 실제 비용은 아직 알 수 없다. f는 적합도(fitness)를 의미하고 g+h이다. 현 노드를 거쳐 진행하는 경로의 비용에 대한 최선의 추측이다. 각 노드는 속성 f, g, h를 가지도록 하고 노드들을 위한 두 개의 목록을 열린 목록(OL: open list)과 닫힌 목록(CL: closed list)로 관리하며 다음과 같은 알고리즘으로 경로를 탐색하며 Fig. 2와 같이 계산된다.

2.2 사람의 탈출 속도 계산

A* 알고리즘을 이용해 사람의 탈출 경로를 찾은 후에는 그 탈출 경로를 통해 사람이 문을 통해 빠져나가야 한다. 사람이 출구를 향해 갈 때에는 정해진 속도에 의해 탈출

하므로서 사람이 밀집되어 있는 장소에서는 사람의 속도가 느려지게 되는데, 사람의 밀집도에 따른 사람의 속도는 식 (1)에 의해 계산되어 진다.

$$S = k - ak\frac{1}{d^2\pi} \tag{1}$$

S는 속도를 의미하고, k는 일정탈출 속도이며, a는 0.266의 상수로 고정하며, d는 가장 가까운 사람과의 거리이다. Fig.3은 사람의 탈출 속도를 계산하는 과정의 전체 흐름도를 나타낸다.

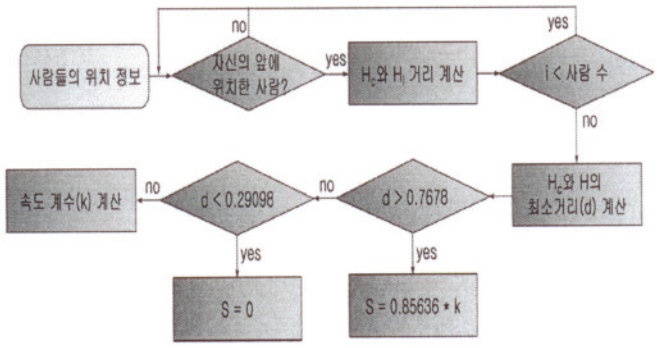

Fig. 2 사람의 탈출 속도계산 과정의 전체 흐름도

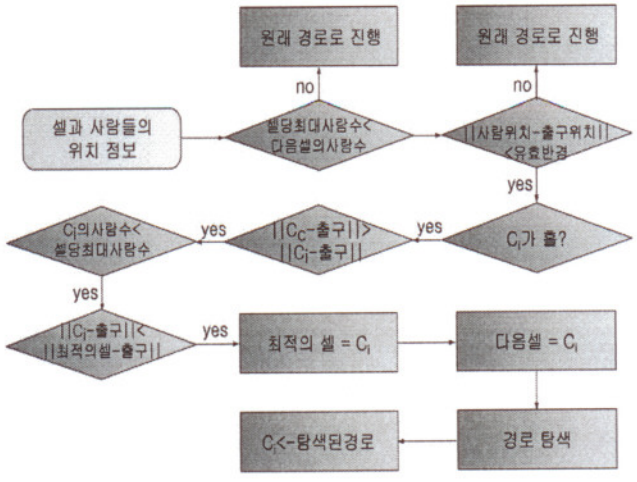

Fig.3 이동병목현상의 해결을 위한 전체 흐름도

첫 번째로 사람의 위치 정보를 입력하여 자신의 앞에 위치한 사람인지 판단하는 것은 사람의 진행하려는 방향 벡터인 V_p와 다름 사람의 위치인 H_i에서 현재 사람의 위치인 H_c로의 벡터 V_d의 내적으로 식 (2)와 같이 계산한다. 식(2)의 내적 값이 양수이면 H_i가 H_c의 앞에 있는 것으로 판단한다.

$$V_p \bullet (H_i - H_c) \tag{2}$$

두 번째 단계에서는 현재 사람의 위치에서 앞의 위치한 사람일 경우에 두 사람의 거리를 계산하고, 다음 단계로 현재 사람의 인덱스가 사람 수 보다 작으면 첫 번째 단계로 가서 다시 루프를 수행하고, 크다면 다음 단계로 진행한다.
그 다음 단계로는 현재 사람의 위치 H_c와 모든 사람의 위치와의 거리를 비교하여 최소 거리를 갖는 사람과의 거리 d를 계산한다. 최소속도와 최고속도의 제한을 두기 위한 방법은 계산된 d가 0.7678 보다 크다면 일정탈출 속도와 0.85636을 곱한 값을 사람의 탈출 속도로 설정하고, d가 0.29098 보다 작다면 속도를 0으로 설정한다. 만약 d가 0.7678과 0.29098 사이의 값을 가진다면 d의 값과 일정탈출 속도 k를 입력으로 식 (1)을 사용하여 사람의 탈출 속도를 계산한다. 사람의 일정탈출 속도는 SIMULEX에서 지정한대로 Table 1의 값들을 사용한다.

Table 1 사람의 종별 일정탈출속도

구분	k
성인남성	1.581481438
성인여성	1.357187892
아이	1.054320959
노인	0.9341865571

Table 2 사람의 평균 크기

구분	어깨 폭	가슴 폭
성인남성	0.50	0.32
성인여성	0.44	0.28
아이	0.38	0.24
노인	0.46	0.30

2.3 사람의 탈출 시 병목현상 해결

대피자가 한정된 출구를 향해 건물을 탈출할 때에, 시뮬레이션 시스템이 셀 단위로 계산을 하기 때문에 병목현상이 발생할 수 있다. 이동병목현상을 해결하기 위하여 현재 사람이 위치한 셀을 중심으로 주변 셀 8노드를 검사하여 그 노드로부터 새로운 경로를 탐색함으로써 병목현상을 해결하는 방법을 제안한다. 본 연구에서 제안한 병목현상의 해결을 위한 방법의 전체 흐름도를 Fig. 4에 나타내며 다음의 순서로 진행된다.
(1) 셀과 사람들의 위치정보를 입력하여 셀당 최대사람 수와 사람이 이동해야할 다음 셀의 사람 수를 비교한다. 셀당 최대 사람의 수는 한 셀의 실제 크기와 Table 2의 사람의 표준 크기를 이용하여 계산한다. 셀당 최대 사람 수보다 다음 셀의 사람 수가 많으면 다음 단계로 진행하고 그렇지 않으면 원래의 경로로 진행한다.
(2) 사람의 위치와 문의 거리를 계산하여 그 거리가 병목현상 해결을 위한 옆걸음을 적용할 유효반경 안에 들어 있는지 판단한다.
(3) 현재 위치해 있는 셀의 8개의 이웃한 셀인 C_i가 홀인지 검사하여, 홀이라면 8개의 이웃한 셀들 중에 출구와 가장 가까운 셀을 찾기 위해 현재 셀에서의 출구의 위치와의 거리와 8개의 이웃한 셀인 C_i에서의 출구의 위치와의 거리를 비교한다.
(4) 다음 셀의 위치를 시작점으로 출구까지의 경로를 A* 알고리즘을 사용하여 재탐색한다. 그런 후에 다음 셀의 노드 뒤에 A* 알고리즘을 사용하여 탐색된 노드로부터 새로운 경로를 생성하게 된다.

3. 실험 결과

본 장은 건물 탈출시 병목현상을 해결하는 것에 대한 성능을 입증하기 위해 윈도우 플랫폼에서 C++와 DirectX 9.0을 사용하여 시뮬레이션 시스템을 구현하였다. PC는

2.4GHz 코어2 프로세서로 1GB DDR RAM의 메모리를 장착하고 512MB DRAM의 GeForce 6800 GPU의 그래픽 카드를 사용하였다.

Fig. 5는 제안된 화재시 대피예측 시뮬레이션 시스템의 모습이다. 출구는 파란색의 블록으로 표시하였으며 높이는 현재 층의 높이로 렌더링하였다. 셀은 막힌 부분에 대해서만 출구와 마찬가지로 현재 층의 높이로 블록으로 표시하였다.Fig.5는 각 대파자가 최종 탈출구인 파란색의 출구에 대한 경로탐색을 종료하고 시뮬레이션을 시작하기 위한 대기되어있는 상태를 나타낸다.

Fig. 6은 대피초기의 모습을 나타내며 대피동선의 이해를 돕기 위해서 각 사람별 탈출경로에 대해 선으로 표시하여 나타내었다. 붉은색 선은 앞으로 진행해야할 경로를 의미하고, 푸른색 선은 각 사람이 지나온 경로를 표시한다. 각 사람은 본 연구의 2.2절에서 설명한 대로 각 사람의 속도를 계산하여 그 속도에 따라 시뮬레이션을 진행한다. 현재까지는 병목현상 해결을 위한 출구와의 유효반경 내에 들지 않았기 때문에 옆걸음은 적용되지 않고 있음을 확인 할 수 있다.

Fig. 7은 출구 근처에서의 병목현상에 따른 사람들의 탈출 모습을 나타낸 다. 출구 근방에서 사람들은 병목현상 해결을 위한 출구와의 유효반경 내에 들어가는 조건하에 놓인다. 따라서 3.3절에서 제안한 병목 현상해결 방법에 기초하여 새로운 경로를 찾아 출구를 향해 진행하게 된다.

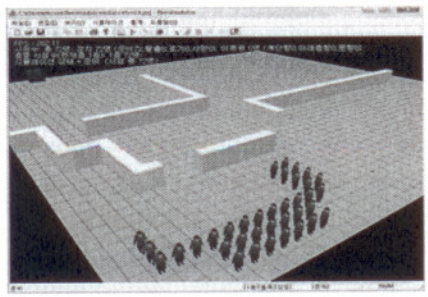

Fig. 4 제안한 화재시 대피예측 시뮬레이션 시스템의 모습.

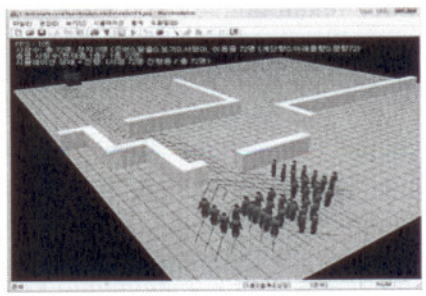

Fig. 5 사람의 대피 초기의 모습.

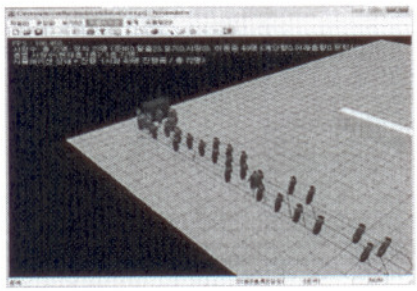

Fig. 6 출구 근처에서의 병목현상에 따른 사람들의 탈출

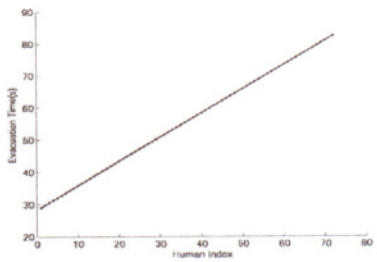

Fig. 7 출구에서 각 사람이 빠져나간 시간

실제 탈출 환경과 유사하게 구현하기 위한 방법으로 출구를 빠져나가는 속도를 flow rate로 제한하여 문의 물리적 크기에 따라 통과 대피자의 수를 연계함수로 지정하였다.

따라서, 탈출구를 빠져나가는 대피자의 수에 제한을 두기 때문에 출구 근처에서 사람들의 병목현상의 재현이 가능하게 되었으며 병목현상에 따른 대피지체시간의 계산이 가능하다.

4. 결 론

본 연구는 화재 발생 시 거주자의 초기탈출을 계획할 수 있는 시뮬레이션 시스템을 다음과 같이 제안하였다.

1.대피자는 가장 가까운 출구를 찾아 탈출하기 위한 최적경로계산으로 A*알고리즘의 적용이 가능하다.

2. 출구에서의 대피자의 출구속도는 대피출구 크기에 따라 flow rate로 정의되어 대피자의 수가 제한되므로 병목현상의 재현이 가능함을 확인하였다.

3. 병목현상을 포함한 대피이동현상을 구현하기 위하여 위해 현재 셀의 8개의 이웃하는 셀의 검사방식을 도입하고 A* 알고리즘을 사용하여 출구까지의 새로운 경로를 찾는 방식의 적용으로 실제대피현상과 매우 근접한 VR 시뮬레이터의 구현이 가능함을 확인하였다.

참고 문헌

[1] Thompson P. A., Marchant E. W., "Testing and Application of the Computer Model 'SIMULEX'," Fire Safety Journal, Vol. 24, No. 2, pp. 149-166, 1995.

[2] E. R. Galea and J. M. P. Galparsoro, "EXODUS: An Evacuation Model for Mass Transport Vehicles," Fire Safety Journal, Vol. 22, pp. 341-366, 1994.

[3] Levin, B. N., "EXITT - A Simulation Model of Occupant Decisions and Actions in Residential Fires: Users Guide and Program Description," US Department of Commerce, Gaithersburg, MD, 1987.

[4] Gwynne S., Galea E. R., Lawrence P., J., Filippidis L., "Modeling Occupant Interaction with Fire Conditions Using the BuildingEXODUS Evacuation Model," Fire Safety Journal, Vol. 36, No. 4, pp. 327-357, 2001.

[5] Russel S. J., Norvig P., "Artificial Intelligence: A Modem Approach. 2nd edition," Pearson Education, Inc, pp. 94-95, 2003.

[6] E. W. Dijkstra, "A Note on Two Problems in Connection with Graphs," Numerische Mathematik, pp. 269-271, 1959.

[7] Donald B. Johnson, "Efficient Algorithms for Shortest Paths in Sparse Networks," Journal of the ACM, Vol. 24, No. 1, pp. 1-13, 1977.

[8] R. Bellman, "On a Routing Problem," Quarterly of Applied Mathematics, Vol. 16, No. 1, pp. 87-90, 1958.

[9] P. E. Hart, N. J. Nilsson, B. Raphael, "A Formal Basis for the Heuristic Determination of Minimum Cost Paths," IEEE Transactions on Systems Science and Cybermetics, SSC4, No. 2, pp. 100-107, 1968.