가시성을 포함한 개선된 실내 보행자 모델[1)]

An Enhanced Indoor Pedestrian Model Incorporating the Visibility

곽수영* · 남현우 · 전철민**

Suyeong Kwak* · Hyunwoo Nam · Chulmin Jun**

서울시립대학교 공간정보공학과

{ksykk0* · nhw612 · cmjun**}@uos.ac.kr

요 약

현재 대규모 실내공간이 증가하고 있고, 이에 따른 화재 등의 사고가 발생함에 따라 화재대피시스템과 같은 실시간 실내 응용에 대한 관심이 증대되고 있다. 그러나 학문적 연구 또는 상업용 시스템에서 사용되는 화재모델이나 보행자 모델은 대부분 2D 기반의 CAD와 같은 파일구조를 기반으로 하고 있으며 가상데이터를 이용한 시뮬레이션에 초점을 두고 있다. 따라서 이들을 실내 센서 등을 이용한 실시간 시스템으로 구축하기 위해서는 몇 가지 문제점들이 해결되어야 한다. 우선, 실내공간의 의미 있는 관계정보를 포함하는 위상적인 3D 모델이 필요하다. 또한, 실내 센서들에 의해 감지된 보행자의 이동을 저장하고, 저장된 데이터를 이용하여 실시간 안내를 위해서는 실내데이터 구축에 공간 DBMS를 이용해야 한다. 본 연구에서는 실내보행자 모델에서 두 가지 개선점을 제시하고자 한다. 첫째는, 간단한 3D 실내 모델구축과 공간 DBMS와 연계된 보행자 시뮬레이터를 구현하는 과정을 제시한다. 둘째는, 보행자의 가시성(visibility)에 대한 영향이 반영된, 개선된 floor field 모델을 제안한다. 이와 같은 과정을 캠퍼스 건물에 적용하고 시뮬레이션을 수행하는 과정을 예시하였다.

1. 서론

지난 수십 년 동안 많은 micro-scale 보행자 모델이 제시되어 왔으며, 이들 대부분은 실세계 응용보다는 알고리즘 개선에 초점이 맞추어져 있다. 최근 3D 모델과 실내 센서들의 개발로 인해 실외뿐만 아니라 실내에서의 길 찾기나 화재 대피와 같은 실시간 응용에 대해 관심이 높아지고 있다. 그러나 기존의 보행 모델을 실시간 상황에 적용하기 위해서는 다음과 같은 제약들이 해결되어야 한다. 첫째로, 실험적인 2D 데이터를 이용하는 대신에 [3, 11] 복층과 계단을 포함하는 실제 실내 구조를 반영하는 3D 데이터 모델을 이용해야 한다.

둘째로, 기존의 학문적 연구나 상업적 응용에서 사용된 접근 방법들은 CAD와 같이 지리적인 좌표체계가 참조되지 않은 파일 기반의 데이터를 이용하였다. 이러한 접근 방법들은 시뮬레이션 목적으로는 적합하지만, 실시간 응용에 적용하기에는 한계가 있다. 실시간 응용에 필요한 client-server간의 데이터 연동을 위해서는 파일이 아닌, DBMS를 이용해야 한다.

이러한 제약조건들을 해결하기 위해, 본 연구에서는 DBMS를 이용한 3D 보행

1) 본 연구는 국토해양부 첨단도시기술개발사업 - 지능형국토정보기술혁신 사업과제의 연구비지원(07국토정보C04)에 의해 수행되었습니다.

** 교신저자

자 모델의 구축 방안을 제안하고자 한다. 실제건물의 실내 3D 공간모델을 구축하고, 이를 공간 DBMS에 저장하여 대피시뮬레이션에 적용한다. 저장된 실내 공간요소들을 시뮬레이션에 이용 가능하게 하고 3차원 가시화가 가능한 데이터로 변환하는 방법을 제시하고자 한다. 또한, 본 연구에서는 기존의 floor field 모델을 개선한 보행 알고리즘을 제안한다. 기존의 floor field 모델에 기반하면서, 보행자의 위치로부터 출구까지 가시성의 정도에 따라 보행속도를 조절할 수 있는, 개선된 floor field 모델을 제시하고자 한다. 본 연구에서는 이와 같이 DBMS 기반의 개선된 보행 모델의 구축과정과 시뮬레이션 수행과정을 캠퍼스건물에 적용하여 예시하였다.

2. 관련연구

보행모델은 network flow, traffic assignment, simulation 등 다양한 분야에서 연구되어 왔으며[1], 대체로 macroscopic 모델과 microscopic 모델로 나누어진다[4]. Macroscopic 모델은 일반적으로 실외에서의 교통흐름 최적화나 길 찾기 등에 적용되며, 기본 데이터 구조로서 node-link 구조를 갖는다. Macroscopic 모델이 개개인들의 개별성을 고려하지 않고 노드나 링크에 할당될 수 있는 동질적인 그룹으로 보는 반면에, microscopic 모델은 개별적인 파라미터(예, 보행속도, 반응속도, 개인성향 등)와 다른 사람이나 물리적인 환경(예, 벽, 장애물, 연기 등)과의 상호작용을 고려한다. 최근에는 microscopic 모델에서 social force 모델과 floor field 모델이 주목을 받고 있다. Helbing 등에 의해 제안된 social force 모델은 목적지까지(예, 출구) 이동할 때, 보행자에게 작용되는 모든 힘을 수학적으로 모델링한다[5, 6, 7]. Helbing의 모델에서는 모든 보행자 각각이 받는 영향과 물리적인 환경요인(예, 어깨폭, 기대속도, 목표점 등)이 고려된다. 이 모델은 $O(n^2)$의 연산 복잡도를 갖는데, 이는 컴퓨터 기반의 시뮬레이션 수행에는 매우 불리하게 작용한다[8, 9].

반면에, Kirchner와 동료들에 의해 제안된 floor field 모델은 cellular automata 기반의 접근방법을 사용한다. 대상공간에 있는 한 보행자가 모든 보행자들의 영향을 받는 대신에, 주위에 있는 보행자와의 상호작용만을 고려한다. 매 time step 마다 각각의 보행자의 이동을 계산하고, 인접한 셀 중에서 다음에 이동할 셀이 결정된다. 이렇게 한 순간에 주변 셀들만을 고려하는 연산은 컴퓨터 기반의 시뮬레이션에 유리하게 작용한다. 본 연구에서는 floor field 모델을 기본 모델로 이용하여 알고리즘 개선에 적용하였다.

Kirchner의 floor field 모델은 static field와 dynamic field, 두 가지의 field를 이용한다. 이들은 보행자의 이동에 관련된 값들을 저장한 레이어들이다. Static field에 저장된 셀 값은 출구까지의 최단거리를 나타낸다. 보행자는 주변 셀의 값들을 통하여 가장 가까운 출구의 위치를 알게 되고, 이는 수치화되어 셀 값으로 저장된다. Static field가 물리적인 거리로 계산된 고정된 값인 반면에, dynamic field는 동적으로 변화되는 값이다. Dynamic field는 보행자가 움직이는 동안에 주변 셀에 미치는 영향을 계산한 값이며 움직임에 따라 확산(diffuse)하고 소멸(decay)한다. 민감도를 통해 static field와 dynamic field의 상대적인 영향의 정도를 조절할 수 있으며, 다양한 시나리오(예, 공포의 정도)를 실험할 수 있다. 이와 관련된 자세한 내용은 [10, 12]에 제시되어 있다.

본 연구에서는 세 번째 field로서 가시성(visibility)을 추가하여 floor field 모델을 개선하였다. Visibility field는 가장 가까이 있는 출구에 대해서 보행자의 시야의 제약에 따라 대피의 유리함 정도를 나타낸 값이다. 이와 관련된 세부적인 사항

은 다음 장에서 자세하게 설명된다.

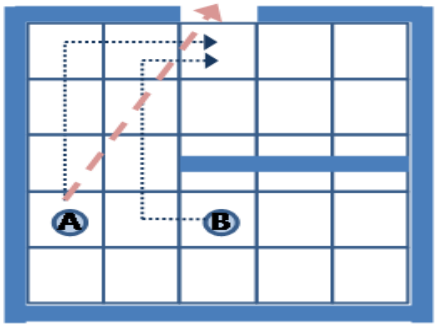

그림 1. 출구에 대해 서로 다른 가시성을 갖는 두 보행자의 예시

3. 가시성을 반영한 floor field 모델

Floor field 모델에서의 static field는 시뮬레이션이 시작되기 전에 출구와의 최단거리로 계산된 값이며, 시뮬레이션이 구동되는 동안 변하지 않는 값이다. 같은 공간에서 다른 위치에 있으나 출구까지 같은 거리에 있으면 static field의 값은 같은 값을 가지게 된다. 그러나 이것은 가구나 벽과 같은 장애물이 존재하여 출구까지의 시야가 방해받는 경우에 대해서는 고려하지 않고 있다. 출구까지의 시야가 제한될수록 보행자는 대피에 어려움을 겪게 될 것이다. 예를 들면, (그림 1)에서, A와 B는 출구로부터 같은 거리에 있다. 이때, 보행자 B는 장애물 뒤에 존재하기 때문에 출구를 바로 볼 수 없다. 즉, A보다 B가 출구까지 가는데 시간이 더 소요될 것으로 예상할 수 있으며, 특히 긴급한 상황에 낯선 환경이라면 탈출시간이 더 길어질 것이다. 이에 착안하여 본 연구에서는 기존 floor field 모델에 visibility field를 추가한 모델을 소개한다. 이를 구현하기 위해서는 우선, 출구에 대한 가시성에 따라 대상 공간을 몇 개의 세부공간으로 분할해야 한다. 그리고 시뮬레이션 중에는 분할된 공간에 보행자들을 배치시키고 이들 공간별로 보행자들이 서로 다른 이동속도를 갖게 한다. 이는 다음 절에서 설명된다.

3.1 보행자의 가시성을 고려한 공간 분할

보행자의 가시성을 고려한 공간 분할 기법으로 우리는 공간구문(space syntax) 이론을 응용하였다[2]. 공간구문론은 도시나 건축공간의 연결성, 접근성을 측정하는데 이용되는 기법이다[13]. 공간구문론에서는 '깊이(depth)'만 고려하며 물리적인 거리는 고려하지 않는다. 이 이론은 거리나 복도를 시축(visual path)으로 구성된 그래프(graph)로 변환한다. 앞서 말한 깊이는 시축의 수 또는 시야의 굴곡성에 의해 결정된다. 깊이가 클수록 대상공간의 구조적인 연결성이 좋지 않음을 의미한다. 우리는 visibility field를 구현하는데 이러한 깊이 기반의 접근 방법을 사용하였다. 우리는 대상공간을 시야의 깊이에 따라 몇 개의 세부공간으로 나누었다. (그림 2)는 출구까지의 시야를 고려하여 깊이를 부여하고, 이에 따라 공간을 분할하는 모습을 보여준다. (그림 2)에서 보는 바와 같이 방 안에 두 개의 장애물이 있다고 가정하면, 출구가 바로 보이는 지역을 공간깊이 ①로 설정한다. 다음으로 공간깊이 ①이 보이는 지역을 공간깊이 ②로 설정한다. 같은 과정으로, 공간깊이 ②

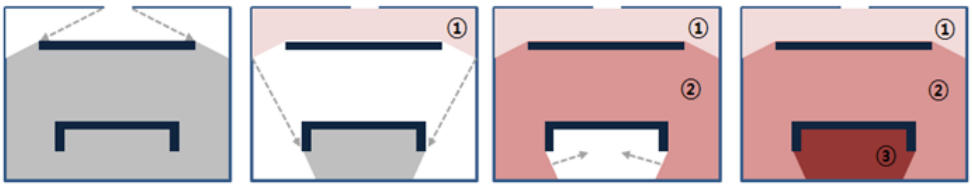

그림 2. 가시성의 정도에 따른 공간 분할

지역이 보이는 지역을 공간깊이 ③으로 설정한다. 이 과정은 전체공간이 세부공간으로 모두 분할될 때까지 반복된다. 최종적으로, 분할된 지역들은 1, 2, 3...으로 공간깊이가 설정되고, 이는 출구까지의 가시성의 정도를 나타낸다.

3.2 Visibility field의 구현

전술한 바에 따라 공간깊이에 의해 visibility field가 구현되면, 보행자들은 static field값과 dynamic field 값을 참고함과 동시에 공간 깊이, 즉, visibility field 값에 따라 다른 보행속도를 가지면서 이동하게 된다. 우리는 프로그램 언어에서 제공하는 time tick을 이용하여 보행속도를 조절하였다. Time tick은 보행자가 매 셀마다 이동할 때 대기하는 시간을 조정할 수 있는 값이다. 기존 모델에서는 매 time step 마다 모든 보행자가 동시에 이동하는 반면에, 개선된 모델에서는 가시성에 따라 할당된 time tick에 따라 이동속도가 달라진다. 따라서 출구에 대해 가시성이 떨어지는 지역에 있는 보행자들은 대피 시간이 더 오래 걸리게 된다.

(그림 3)은 기존 floor field모델과 visibility field 추가를 통해 개선된 모델에서의 보행자 이동을 비교한 것이다. 이것은 시뮬레이션이 가동된 후, 같은 시간에 캡처한 그림이다. Static field만을 고려한 (그림 3)의 왼쪽 그림을 보면 장애물의 양쪽 끝에서 병목현상이 나타난다. 이것은 장애물 뒤에 있는 보행자들이 출구로부터 동등하게 떨어져있는 위치일 때, 가시성이 고려되지 않은 채 같은 속도로 이동하기 때문이다. 반면에, (그림 3)의 오른쪽 그림은 공간깊이가 큰 지역에 있는 보행자들이 공간깊이가 작은 지역에 있는 보행자들보다 더 천천히 이동하는 것을 알 수 있다.

그림 3. 기존 floor field 모델과 visibility field가 추가된 모델의 비교

4. 시뮬레이터의 구현

본 연구에서는 공간 DBMS 기반의 3차원 실내 대피 시뮬레이터를 구현하였다. 서론에서 언급한 바와 같이, 우리는 3D 건물 모델과 공간 DBMS의 사용에 중점을 두었다. 구현과정을 보면 우선, 층과 계단의 공간데이터와 속성데이터를 생성하고 이를 공간 DBMS에 저장하였다. 그리고 저장된 데이터를 시뮬레이션에서 요구하는 데이터형식인 grid cell 데이터로 변환하였다. Grid cell 데이터는 보행자의 대피

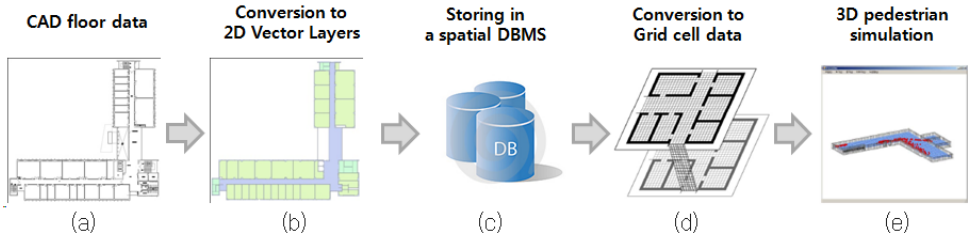

그림 4. 데이터 변환 및 시뮬레이션 구현 과정

경로 연산과 3D 모델링에 이용된다. 시뮬레이터는 grid cell 데이터를 이용하여 보행자들의 대피시뮬레이션을 수행하고, 대피과정을 3차원으로 보여준다. 위의 처리과정은 (그림 4)에 요약되어 있다.

4.1 공간데이터의 생성과 저장

건물의 층 데이터는 CAD 파일을 통해서 얻을 수 있다. 하지만 CAD 데이터들은 기본적으로 지리참조가 되어있지 않고, 위상정보가 존재하지 않는다. 따라서 실시간으로 건물의 인원을 파악하고 대피경로를 안내하는 등의 실시간 응용에 적용하기에 한계가 있다. 위상적인 다층 데이터를 이용하기 위해서, 본 연구에서는 새로운 형식을 개발하기보다는 기존에 존재하는 shapefile 데이터를 사용하였다. Shapefile은 많은 GIS 어플리케이션에서 처리할 수 있고, 활용성이 넓기 때문이다. 우리는 QuantumGIS[16]를 이용하여 데이터변환을 수행하였다. QuantumGIS에서 CAD 파일을 불러온 후, 문, 벽, 방, 복도, 출구 등의 정보를 폴리곤과 라인의 형태로 추출한다. 추출된 데이터는 shapefile 형태로 저장된다.

데이터 저장에 이용한 공간 DBMS는 PostgreSQL/PostGIS이다[14]. 본 연구에서 데이터 저장에 사용한 DBMS 테이블의 구조는 (그림 5)에 나타나 있다. 방, 복도, 벽, 문, 출구 등의 공간 요소들은 각각에 대응하는 테이블에 저장된다. 또한, 건물의 층, 방 번호와 같은 속성데이터도 테이블에 저장된다. 위의 과정을 거쳐 모든 층과 계단 데이터를 DBMS에 저장한다.

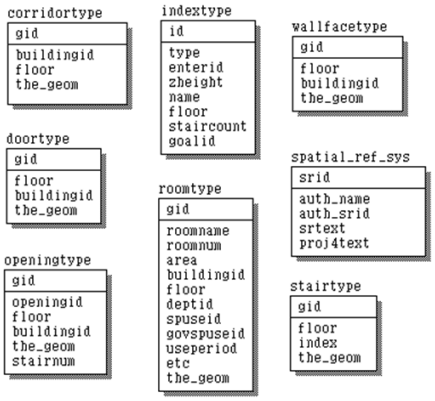

그림 5. 공간 DBMS 테이블의 ERD

4.2 Grid cell 데이터로 변환

본 연구는 전술한 바와 같이 floor field 모델을 기반 모델로 이용하였으며, 이 모델은 grid cell 데이터를 통해 시뮬레이션을 수행한다. 본 연구에서는 시뮬레이션 시스템은 C#을 이용하여 구현되었다. 또한, 공간데이터를 C#으로 불러오기 위해서 PostgreSQL에서 지원하는 npgsql library[15]를 이용하였다. Grid cell 데이터로 변환하는 과정은 (그림 6)에 나타나있다. 우선, DBMS에 저장된 건물데이터와 속성데이터를 시뮬레이터에서 읽어온다. 그리고 불러온 건물데이터에서 층과 계단 데이터를 SharpMap library[17]를 이용하여 bitmap 형식으로 변환한다. 이 과정은 (그림 6-(a))에 나타나 있다. SharpMap library는 공간 DBMS에 저장된 공간데이터를 2차원으로 가시화하는 기능을 제공한다. Bitmap 형식으로 변환하는 이유는

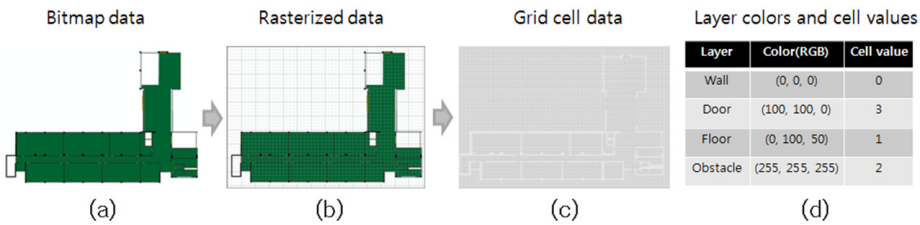

그림 6. Grid cell 데이터 구축과정

raster 구조를 가지고 있기 때문인데, raster 구조도 격자 구조의 형태이므로 grid cell로 변환이 용이하다. 또한, 각각의 레이어를 서로 다른 RGB값을 부여하여 구분할 수 있기 때문이다. Bitmap 데이터에서 문, 방, 벽, 외부공간 등의 레이어들은 각각 다른 색으로 지정되고, 부여된 RGB값을 기준으로 셀 값이 정해진다. (그림 6-(d))는 레이어별로 부여된 RGB값과 그에 따른 셀 값을 나타낸다.

4.3 시뮬레이션 수행과 3차원 가시화

변환된 grid cell 데이터를 이용하여 대피 시뮬레이션을 수행하였다. 시뮬레이션 모델은 본 연구에서 제안하는 visibility field를 반영한 개선된 floor field 모델을 사용하였다. 우선, 설정된 수의 보행자들을 대상 건물에 임의의 위치에 배치한 후, 대피 과정을 연산하였고, 그 결과를 3차원으로 가시화하였다. 대상건물과 보행자를 OpenGL Library를 사용하여 3차원으로 가시화 하였다. 벽과 3차원 모양의 계단의 모델링은 DBMS에 저장된 높이 값을 이용하였다. 계단은 기울기를 계산하여 계단의 형태가 되도록 하였고, 이를 3차원으로 가시화하였다.

시뮬레이션의 결과는 text형식의 log 파일로 저장된다. 보행자가 빠져나간 출구의 좌표와 보행자의 ID, 탈출시간 등의 정보가 기록되며, 이를 통해 전체 인원이 건물에서 탈출한 시간도 알 수 있다. 본 시뮬레이션 시스템은 visibility field의 적용 여부를 포함한 다양한 파라미터들을 미리 설정할 수 있으며, 설정된 값들을 통해 시뮬레이션이 수행하게 된다.

5. 시스템 테스트

우리는 대학 캠퍼스 건물을 이용하여 시뮬레이션 시스템을 테스트 하였다. 전술한 처리과정에 따라서, CAD 형식인 건물 데이터를 shapefile로 변환하였고, 이를

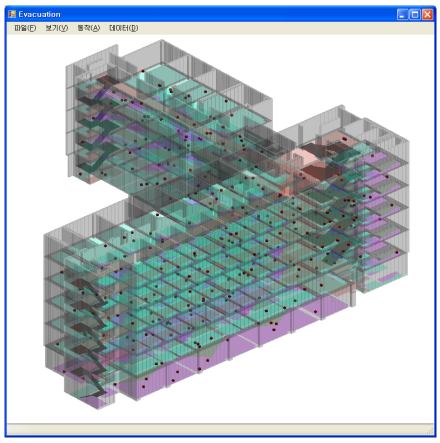

그림 7. 3D로 가시화된 시뮬레이션 화면

공간 DBMS에 저장하였다. 시뮬레이션 시스템은 DBMS에 저장된 데이터를 불러와 grid cell 형식으로 변환한다. 본 시스템은 (그림 7)과 같이 3차원 가시화를 지원한다. (그림 8)은 시뮬레이션의 결과를 저장한 log file을 나타낸다. 이를 통해 우리는 특정 보행자가 어떤 출구로, 어느 시간에 나갔는지를 알 수 있다. 그림에 나와 있는 'TIMETICK'은 보행자가 대피에 소요된 시간을 나타낸다. 'AGENT ID'는 보행자의 ID번호를 나타내고, 'EXIT POS'는 보행자가 대피한 출구의 좌표를 나타낸다. 또한, 우리는 전체 보행자의 대피 시간도 계산할 수 있다. 이 log 파일은 다시 DBMS에 저장되어 관리된다. 이 결과파일을 통해 보다 다양한 분석을 할 수 있는데, 예를 들면, 건물 내에서 어떤 공간이 대피가 어려운지 알 수 있다. 아직 개발되지는 않았지만, 만약 추후에 본 시스템이 실내 센서와 통합이 이루어진다면, 건물 내 각각의 공간에 실시간으로 사람들이 몇 명이 있는지 알아내어, 이 데이터를 이용한 대피시뮬레이션을 수행하고 대피 경로를 안내하는 실시간 인명 구조에 활용될 수 있을 것이다.

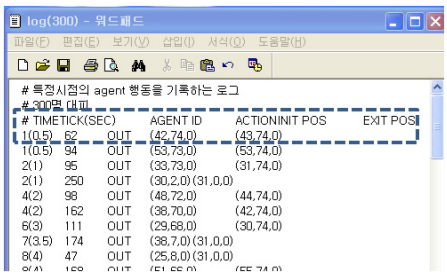

그림 8. 시뮬레이션 결과를 기록한 log file

6. 결론

본 연구에서는 공간 DBMS를 이용한 3차원 실내 대피 시뮬레이터를 구현하였으며, 2가지가 중점적으로 다루어졌다. 첫째, 건물의 데이터를 생성하여 공간 DBMS에 저장하고, 이를 시뮬레이션 시스템에서 불러와 grid cell 형식으로 변환하여 시뮬레이션을 수행하는 과정이 연속적으로 이루어지도록 개선한 것이다. 둘째, 기존의 floor field 모델에 visibility factor를 추가하여 개선하였다. 출구에 대한 시야의 제한에 따라서 공간을 분할하였고, 분할된 공간에 따라 대피자의 이동속도를 조절하였다. 본 연구에서 수행한 시뮬레이션의 결과를 추후에 실시간 시스템에 적용하기 위해서, 현재 본 시스템을 실내 센서와 연계할 수 있도록 개선하고 있다. 또한, 본 연구에서 제안한 visibility 알고리즘의 검증(validation)과정을 수행하고 있다.

참고문헌

[1] Ahuja, R. K., Magnate, T. L., Orlin, J. B. 1993. Network Flows, Theory, Algorithms, and Applications, Prentice Hall.

[2] Cho, D. 1999. A Study on the Tools of Analysis in the Space Syntax Theory, Architectural Institute of Korea, 156, 71-76.

[3] Gwynne, S., Galea, E.R., Owen, M., Lawrence, P.J., Filippidis, L.L. 2005. A systematic comparison of building EXODUS predictions with experimental data from the Stapelfeldt trials and the Milburn House evacuation, Applied Mathematical Modeling, 29, 9(Sep.2005), 818-851.

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

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

[6] Helbing, D., Farkas, I., Vicsek, T. 2000. Simulating dynamical features of escape panic, Nature 407(Sep.2000), 487-490.

[7] Helbing, D., Molnár, P. 1997. Self-organization phenomena in pedestrian crowds, In F. Schweitzer (ed.), Self-Organization of Complex Structures: From Individual to Collective Dynamics, Gordon & Beach, London, UK.

[8] Henein, C., White, T. 2005. Agent-based modeling of forces in crowds, In P. Davidsson, B. Logan and K. Takadama (Eds.), Multi-agent and Multi-agent-based Simulation, Lecture Notes in Computer Science, 3415, Springer, New York, 173-184.

[9] Henein, C., White, T. 2007. Macroscopic effects of microscopic

forces between agents in crowd models, Physica A, 373, 694-712.
[10] Kretz, T., Schrekenberg, M. 2006. Floor field- and Agent-based Simulation Tool, International Symposium of Transport Simulation 2006, Lausanne, Switzerland, 4 - 6.
[11] Lo, S.M., Fang, Z., Lin, P., Zhi, G.S. 2004. An evacuation model: the SGEM package, Fire Safety Journal, 39, 3(Apr.2004), 169-190.
[12] Nishinari, K., Kirchner, A., Namazi, A., Schadschneider, A. 2005. Simulations of Evacuation by an Extended Floor Field CA Model, Traffic and Granular Flow ' 03.
[13] Penn, A., B. Hillier, D. Banister, and Xu, J. 1998. Configurational modeling of urban movement networks, Environment and Planning B-Planning & Design 25, 1, 59-84.
[14] PostgreSQL, http://www.postgresql.org/.
[15] PostgreSQL/npgsql, http://pgfoundry.org/projects/npgsql/.
[16] QuantumGIS, http://www.qgis.org/.
[17] SharpMap, http://www.codeplex.com/SharpMap/.