Journal of Korea Multimedia Society Vol. 9, No. 11, November 2006(pp. 1483-1495)

가시성 검사를 이용한 3차원 게임에서의 효율적인 경로 탐색

김형일[†], 정동민[††], 엄기현[†††], 조형제[††††], 김준태[†††††]

요 약

이동망(Navigation Mesh)은 캐릭터가 이동할 수 있는 다각형(삼각형) 집합으로 지형을 표현한다. 이동망은 자동화된 생성이 가능하며, 유연하게 3차원 공간을 표현할 수 있다. 지형 구조에 따라 삼각형 수를 달리함으로써 다양한 표현이 가능하며, 캐릭터는 3차원 상태 공간을 2차원 평면 공간으로 표현한 이동망으로만 이동하기 때문에 효율적인 이동과 경로 계획을 보장받을 수 있다. 그러나 현실적인 캐릭터의 이동을 위해 보다 많은 삼각형을 이용하여 지형을 표현함으로 경로 계획 수립 시 많은 상태 공간(다각형)을 검색하여 효율적인 탐색이 이루어지지 않는다. 본 논문에서는 이동망으로 표현된 3차원 게임에서에서의 효율적인 경로 탐색 방법을 위한 가시성 검사 기법을 제안한다. 정교한 다각형으로 이루어진 세밀한 지형에 그래프 기반 탐색을 적용하면 탐색할 공간이 많기 때문에 효율적인 탐색이 이루어지지 않는다. 그래서 본 논문에서는 정교하게 구성되어 있는 3차원 지형에서 효율적인 탐색이 이루어질 수 있도록 가시성 검사(visibility test)를 이용하여 탐색 공간을 줄이는 방법을 사용하였다. 장애물의 정점을 찾고 추정 함수(heuristic function)를 이동망을 지나가는 거리로 정의함으로서, 직선거리를 추정 함수로 정의한 그래프 기반 탐색 방법보다 탐색 영역을 현저하게 줄일 수 있었다.

Efficient Path Finding in 3D Games by Using Visibility Tests

Hyungil Kim[†], Dongmin Jung[††], Kyhyun Um[†††], Hyungje Cho[††††], Juntae Kim[†††††]

ABSTRACT

The navigation mesh represents a terrain as a set of triangles on which characters may move around. The navigation mesh cab be generated automatically, and it is more flexible in representing 3D surface. The number of triangles to represent a terrain may vary according to the structure of the terrain. As characters are moving around on a navigation mesh, the path planning can be performed more easily by projecting the 3D surfaces into 2D space. However, when the terrain is represented with an elaborated mesh of large number of triangles to achieve more realistic movements, the path finding can be very inefficient because there are too many states (triangles) to be searched. In this paper, we propose an efficient method of path finding in 3D games where the terrain is represented by navigation meshes. Our method uses the visibility tests. When the graph-based search is applied to elaborated polygonal meshes for detailed terrain representation, the path finding can be very inefficient because there are too many states (polygons) to be searched. In our method, we reduce the search space by using visibility tests so that the search can be fast even on the detailed terrain with large number of polygons. First we find the visible vertices of the obstacles, and define the heuristic function as the distance to the goal through those vertices. By doing that, the number of states that the graph-based search visits can be substantially reduced compared to the plane search with straight-line distance heuristic.

Key words: Path Finding(경로 탐색), Visibility Graph(가시성 그래프), Graph Search(그래프 탐색), 3D games(3차원 게임)

※ 교신저자(Corresponding Author) : 김준태, 주소 : 서울특별시 중구 필동 3가 동국대학교 컴퓨터공학과 인공지능연구실(100-715), 전화 : 02)2285-3712, FAX : 02)2285-3712, E-mail : jkim@dongguk.edu
접수일 : 2006년 6월 15일, 완료일 : 2006년 10월 23일
[†] 정회원, 동국대학교 컴퓨터공학과
(E-mail : hikim@dongguk.edu)
[††] 준회원, 동국대학교 컴퓨터공학과
(E-mail : mjung@dongguk.edu)
[†††] 종신회원, 동국대학교 게임멀티미디어공학과
(E-mail : khum@dongguk.edu)
[††††] 정회원, 동국대학교 멀티미디어학과
(E-mail : chohj@dongguk.edu)
[†††††] 정회원, 동국대학교 컴퓨터공학과

1. 서 론

3차원 게임이나 가상현실에서는 현실감을 중요하게 여기며, 이러한 현실감의 묘사는 복잡한 알고리즘과 높은 하드웨어 성능을 요구한다[1-2]. 현실감을 나타내기 위한 세밀한 지형 공간의 묘사는 메모리 부담으로 작용하며, 이와 같은 게임 상황에서 게임 캐릭터 이동은 많은 탐색 시간을 요구한다[3-4]. 메모리의 부담을 줄이고 게임 진행 속도를 저하시키지 않기 위해서는 지형 공간 표현을 단순화하여야 하며, 게임 캐릭터의 이동경로도 단순화되어야 한다. 그러나 사실적인 묘사를 위해서는 세밀한 지형 표현이 요구된다.

3차원 게임이나 가상현실에서 지형을 표현하는 대표적인 방법은 격자(grid) 방법이다. 격자 방법은 지형을 동일한 크기의 격자로 분할하는 방식으로 이동경로 탐색에 편리한 방법으로, 이동경로 탐색 시 표현된 격자를 따라 탐색하기 때문에 경로 탐색이 수월하다. 그러나 격자 공간내의 지형이나 객체들은 격자 모양에 의해 표현되어야 한다. 그래서 지형이나 객체를 정교하게 표현하기 위해서는 격자를 매우 작은 단위의 격자로 분할해야 하기 때문에 탐색 영역이 증가되며, 탐색 영역의 증가는 캐릭터의 탐색 속도를 저하시키는 원인으로 작용한다.

격자 방법의 문제점을 해결하기 위해 나온 방법이 경로점(waypoint) 방법이다. 경로점 방법은 캐릭터가 이동할 수 있는 몇 개의 경로점을 사전에 정의하고, 사전에 정의된 경로점만을 이용하여 캐릭터를 이동시킨다. 경로점 방법은 사전에 정의된 적은 수의 경로점만을 탐색하여 이동하기 때문에 메모리에 부담을 주지 않으면서 캐릭터를 쉽고 간단하게 이동시킬 수 있다. 경로점 방법을 활용할 경우, 지형 표현 시 캐릭터의 경로 탐색을 고려하지 않아도 되기 때문에 경로 탐색과 독립적으로 지형을 표현할 수 있는 장점이 있다. 그러나 게임 캐릭터의 이동경로를 단순화하여 현실성이 떨어지는 문제점을 발생시킨다. 현실성을 잘 나타내면서 캐릭터의 이동을 자연스럽게 하기 위해서는 3차원 게임에 적합한 효율적인 경로 탐색 기법이 필요하다.

현실감을 게임에 가미하기 위해서는 세밀하게 표현된 지형이 요구되지만, 정교하게 지형을 표현할수록 경로 탐색 시간이 증가하는 문제점이 발생한다. 그래서 경로 탐색 시간을 줄이기 위해 게임에서는 여러 가지 방법이 시도되고 있다[5-7]. 근래, 가장 효율적인 경로 탐색을 제공하는 지형 표현 방법은 이동망(navigation mesh)이다. 이동망은 캐릭터가 이동할 수 있는 삼각형 모양의 다각형으로 지형을 표현한다. 이동망은 자동화된 생성이 가능하며, 유연하게 3차원 공간을 표현할 수 있다. 지형 구조에 따라 다각형(삼각형) 수를 달리함으로써 다양한 표현이 가능하며, 캐릭터는 3차원 상태 공간을 2차원 평면 공간으로 표현한 이동망으로만 이동하기 때문에 효율적인 이동과 경로 계획을 보장받을 수 있다. 그러나 현실감을 가미하기 위해 세밀하게 지형을 표현하면 캐릭터의 이동을 위해 보다 많은 다각형을 이용하여야 하기 때문에 효율적인 경로 탐색이 이루어지지 않는다. 3차원 게임과 같이 지형이 방대한 게임에서는 탐색 공간이 넓기 때문에 캐릭터의 경로 탐색이 메모리 부하로 작용한다[2]. 3차원 게임과 같이 방대한 크기의 탐색 공간을 소유한 지형에서는 효율적인 경로 탐색을 위해 탐색 공간의 줄여야 한다.

경로 탐색에서 가장 많이 활용하는 방법은 그래프 기반 탐색으로 그래프 기반 탐색은 크게 맹목적 탐색과 경험적 탐색으로 나뉜다. 맹목적 탐색은 목표지점에 상관없이 무관한 순서로 탐색을 수행하므로 소모적인 탐색을 하지만, 경험적 탐색은 목표지점에 대한 관련 정보를 이용하여 탐색을 수행하므로 효율적으로 탐색을 수행할 수 있다. 경로 탐색에서 가장 많이 사용되는 방법은 경험적 탐색이다[8-9]. 경험적 탐색 알고리즘은 최단 경로를 보장하기 때문에 3차원 게임에서도 게임 캐릭터의 경로 탐색에 가장 널리 활용되는 알고리즘이다. 그러나 최단 경로를 보장하는 경험적 탐색 알고리즘도 탐색 공간이 커지게 되면 경로 탐색에 오랜 시간이 소요된다.

본 논문에서는 3차원 게임에서 효율적인 경로 탐색을 위한 방법으로 가시성 검사를 이용한 경로 탐색 기법을 제안한다. 본 논문에서 제안한 가시성 검사를 이용한 경로 탐색 기법은 3차원 게임이나 가상현실에서 경험적 탐색 기법보다 빠른 탐색 시간을 보장한다. 본 논문에서는 이동망으로 표현된 3차원 게임 지형에서의 효율적인 경로 탐색 방법을 제시한다. 기본적인 개념으로 가시성 그래프를 사용하였다[10-13]. 가시성 검사를 사용하여 탐색 공간을 축소시키면 많은 폴리곤을 사용하여 상세하게 지형을 표현했더라

도 캐릭터는 매우 빠르게 이동경로를 탐색할 수 있다. 본 논문에서 제안한 가시성 검사를 이용한 경로 탐색 기법은 먼저 3차원 지형에 나타난 장애물의 가시적인 정점을 찾고, 이 정점을 경유한 목적지까지의 거리를 추정 함수(heuristic function)로 정의한다. 이러한 방법을 경로 탐색에 이용하면 경험적 탐색 기법보다 방문한 상태 수가 현저하게 줄어든다. 본 논문에서 제안한 가시성 검사를 이용한 경로 탐색 기법을 경험적 탐색 기법과 몇 가지 비교 실험을 통해 탐색의 효율성을 확인하였다.

본 논문의 2장에서 이동망에서의 경로 탐색에 대해 설명하고, 3장에서는 본 논문에서 제안한 경로 탐색에서의 가시성 검사 개념과 가시성 검사를 이용한 탐색에 대해 설명한다. 4장에서 제안한 기법의 실험 결과를 분석하고, 5장에서는 결론과 향후 연구에 대해 기술한다.

2. 이동망에서의 경로 탐색

2.1 이동망

3차원 게임이나 가상현실에서 지형을 표현할 때에는 표현하려는 지형의 표면을 다각형의 집합으로 근사화(approximation)하여 나타낸다[14-16]. 과거부터 현재까지 다각형으로 지형을 표현할 때 가장 많이 활용하는 것은 사각형 모양의 격자이다[17-18]. 그러나 사각형은 곡면의 근사화에 사용되지만, 세밀하고 정확한 렌더링(rendering)을 기대하기가 어렵다. 하지만 삼각형은 세밀하고 정확한 렌더링이 가능하여 근래에 부각되고 있는 다각형이다.

이동망은 캐릭터가 장애물과 충돌하지 않으면서 자유롭게 이동할 수 있는 이동 공간이다. 이동망에서는 지형을 표현할 때 다각형 중에서도 삼각형을 이용하기 때문에 표현할 수 있는 영역이 넓다[19]. 이와 같은 이유로 근래에는 3차원 게임이나 가상현실에서 지형 표현에 이동망을 활용하고 있는 추세이다.

이동망은 다각형들의 집합으로 게임 캐릭터가 이동할 수 있는 3차원 공간을 표현한다. 이러한 이동망에 속한 다각형은 3가지 조건을 만족해야만 한다. 가장 단순하게 3차원 공간을 표현할 수 있는 다각형으로 구성되어야 하며, 모든 인접한 다각형들은 2개의 정점(vertices)과 하나의 공유 변을 가져야 한다. 그리고 서로 다른 두 다각형은 동일한 평면에 겹쳐져서는 안 된다.

그림 1은 이동망을 이용하여 지형을 표현한 예제이다. 그물과 같이 형성된 삼각형들은 캐릭터가 이동할 수 있는 지형이고, 장애물 1, 장애물 2, 장애물 3으로 표현된 구조물들은 캐릭터가 통과할 수 없는 장애물을 나타낸다. 시작지점은 캐릭터가 이동을 시작하기 위해 위치한 장소(상태)이며, 목표지점은 캐릭터가 시작지점에서 출발해서 도착해야 하는 장소를 나타낸다. 게임 캐릭터가 시작지점에서 출발하여 목표지점까지 이동을 할 경우, 장애물 1, 장애물 2, 장애물

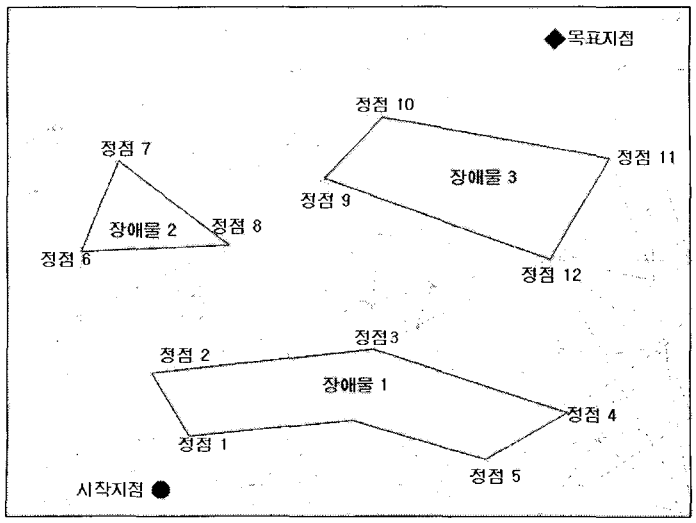


그림 1. 이동망으로 표현한 지형

3과 충돌하지 않으며, 이동망으로 구성된 지형만을 이용하여 목표지점에 도착하여야 한다.

이동망은 다양한 형태의 지형 구조에 적용할 수 있으며, 정적인 장애물과 상호 배타적으로 모델링이 된다는 장점이 있다. 또한 캐릭터와 장애물 간의 충돌은 이동망에 속하는 다각형 영역을 벗어났는지 검사하면 되기 때문에 손쉽게 충돌을 발견할 수 있으며, 충돌 검사에 걸리는 시간이 적게 소요되어 상대적으로 경로 탐색 시간을 줄일 수 있다.

캐릭터가 이동망에서 이동할 때, 이동 벡터는 대응되는 삼각형 평면으로 투영되기 때문에 삼각형과 이동 벡터의 교차를 확인할 수 있다. 만약에 이동 벡터가 공유하지 않는 다른 삼각형의 변에 교차한다면 캐릭터는 장애물과 충돌한 것으로 인식한다. 이와 같은 방법을 통해 캐릭터는 이동망에서 자유로운 이동을 보장 받을 수 있다.

2.2 이동망에서의 경험적 탐색

이동망에서 경험적 탐색 기법을 이용하여 경로 탐색을 수행하면 상태 공간에서 각각의 삼각형은 특정 상태에 이웃한 인접 상태가 대응된다. 특정 상태에서 인접 상태까지의 경로 탐색 비용은 두 공유 변의 중간 지점을 경유하는 두 삼각형의 중심 간의 거리로 정의한다. 그림 2에서는 시작지점(상태)과 3개의 인접 상태를 나타내고 있다. 점을 포함하는 삼각형이 시작지점이고, 음영 삼각형이 인접 상태이다. 점선은 두 상태 간에 거리를 나타낸다.

그림 3은 이동망에서 경험적 탐색 결과에 의해 생성된 경로를 나타낸다. 경험적 탐색은 직선 방향으로 표현된다. 상태 S에 대한 평가함수 f(s)는 시작 상태에서부터 목표 위치까지의 추정 비용을 위한 거리로 3차원 유클리드 거리를 적용하였다. 탐색 결과는 연속적인 인접 삼각형으로 나타나며, 이 삼각형의 중심과 공유 변의 중점을 통해 실제적인 이동경로를 생성한다. 최종적인 이동경로는 간단한 기법을 이용하여 자연스러운 직선으로 표시되지만, 그림 3에서는 이동망에서의 이동경로를 명확히 보이기 위해 기본적인 경험적 탐색 진행 경로만을 나타내었다.

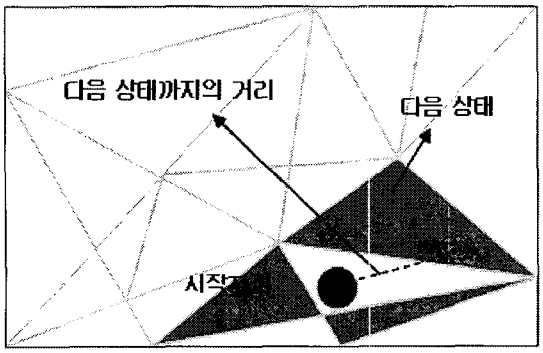


그림 2. 경험적 탐색을 위한 상태 표현

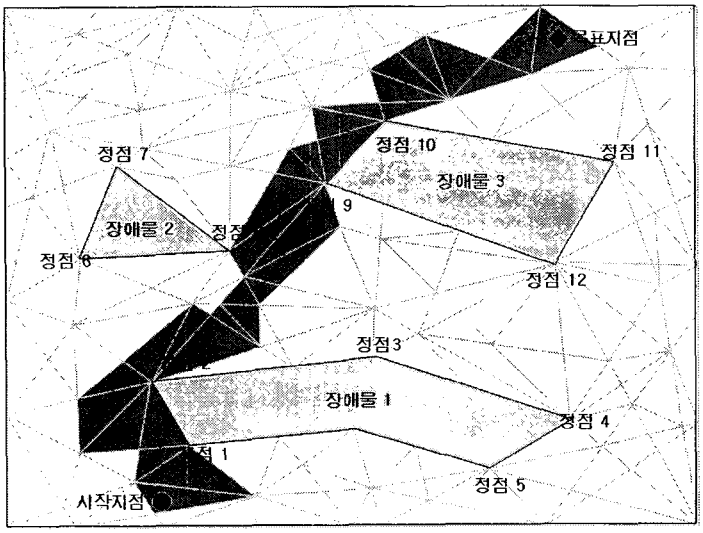


그림 3. 이동망에서의 경험적 탐색 결과

3. 경로 탐색에서의 가시성 검사(Visibility Test)

이동망은 캐릭터의 이동과 경로 탐색에 대해서 간단한 방법을 제공한다. 그러나 정밀하게 지형을 표현하기 위해 많은 수의 삼각형을 사용할 경우에는 효율적인 경로 탐색이 이루어지지 않는다. 세밀한 지형 표현으로 탐색 공간이 많아지고 장애물이 있을 경우, 캐릭터는 불필요한 상태 공간까지 탐색한다.

그림 4에서 빨간색 음영으로 표시된 상태(삼각형)는 경로 탐색을 수행하는 동안 방문한 상태이다. 그러나 시작지점과 목표지점 사이의 장애물로 인해 대부분이 불필요하게 탐색된 영역이다. 그래서 시작지점에서 가시적인 영역에 있는 장애물의 두 정점까지의 경로를 찾을 수 있는 알고리즘이 있다면 불필요한 공간을 탐색하지 않을 수 있다. 3장에서는 가시성을 이용하여 불필요한 탐색 공간을 줄일 수 있는 방법에 대해서 설명한다.

3.1 가시성 검사

가시성 그래프 알고리즘은 2차원 상태 공간에서 로봇 동작 계획에 광범위하게 사용되었다[7,10,13]. 이러한 가시성 그래프는 다음과 같이 정의된다.

가시성 그래프 VG = (V, E)
V: 장애물의 정점, 시작점, 목표점
E: 가시적인 노드의 쌍
w(u, v): u와 v 사이의 거리(weight of edge(u, v))

가시성 그래프는 시작지점에서 목표지점까지 충돌을 배제한 경로들(시작지점에서 목표지점까지의 경로 집합)을 나타낸다. 각각의 경로들은 연속적인 직선으로 구성되며, 이를 기반으로 생성된 그래프는 시작지점부터 목표지점까지의 최단 경로를 포함하고 있다. 이러한 가시성 그래프는 장애물의 정점과 시작지점, 목표지점의 모든 가시적인 연결 쌍과 이들 간의 거리 가중치를 이용한 연결정보를 가지고 구성한다. 두 정점 v와 w가 서로 가시적이면 v와 w를 연결했을 때 어떤 장애물도 통과하지 않는다. 가시성을 결정하기 위해서 평면상의 모든 정점들을 대상으로 가시성 검사를 수행한다. 만약에 정점 v에서 w 방향으로 가시선을 생성했을 때, 이 가시선이 정점 w에 이를 때까지 장애물의 내부와 교차하지 않는다면 정점 v와 w 사이에 가시적인 연결이 생성된다. n개의 정점들에 대해 가시적인 정점들을 찾는 것은 $O(n^2 \log n)$내에 수행된다. 가시성 그래프가 생성이 되면, 시작지점과 목표지점까지의 최적화된 경로는 경험적 탐색 기법을 이용하여 찾게 된다. 본 논문에서도 유사한 개념을 사용했다. 각각의 상태에 대해서 가시성을 테스트하고 두 정점 간의 가중치 정보는 경험적 탐색 기법의 추정 함수를 이용하여 계산하였다.

그림 5는 기본적인 가시성 검사에 대한 개념을 보여주고 있으며, 직선으로 되어 있는 화살표는 시작지점에서의 가시성을 나타내고 있으며, 점선으로 되어 있는 화살표는 정점 v2에서의 가시성을 표현한 것이다.

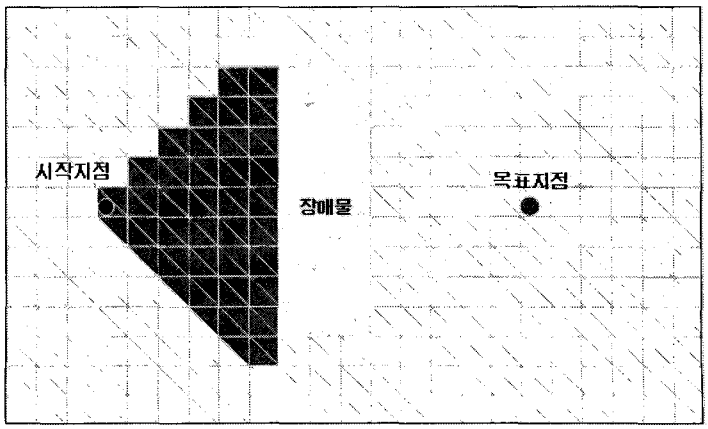


그림 4. 일반적인 탐색의 비효율성

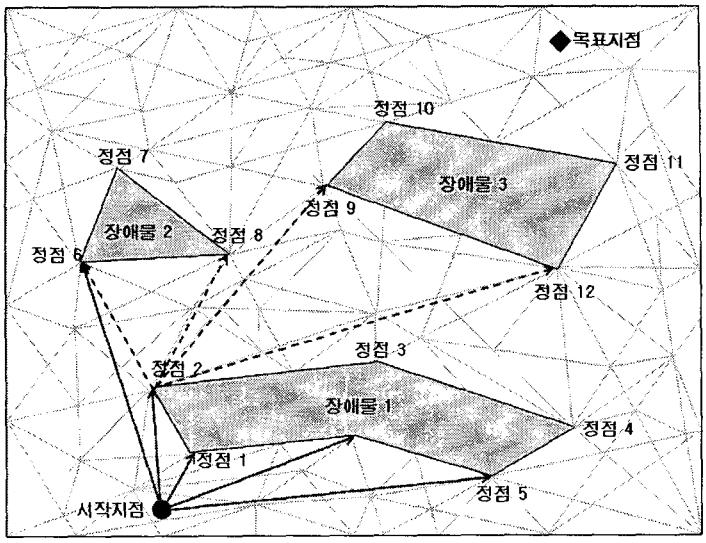


그림 5. 가시성 검사

3차원 지형에서 가시성 검사는 볼록 다각형을 2차원으로 투영함으로써 가능하나, 이와 같은 경우 정점들은 실제적인 가시성은 찾을 수 없게 된다. 3차원 지형에 속한 각 정점들은 이동 방향의 변경 없이 가시성 검사를 수행하나, 이와 같은 경우에서의 이동은 실제적인 최적화된 경로가 아닐 수 있다.

그림 6에서 3차원 지형 표면에 가시성에 해당하는 선을 그려서 가시성 검사 결과를 나타내었다.

3.2 가시성 정보

가시성 그래프는 가시적인 정보를 이용하기 때문에 3차원 지형에서는 가시성 그래프를 생성하기 어렵다. 이와 같은 문제점 때문에 가시성 그래프는 2차원 지형에서 주로 활용되었다. 그러나 본 논문에서 제안한 알고리즘은 3차원 지형에서 경로 탐색을 수행할 수 있다. 본 논문에서 제시한 알고리즘은 기존의 가시성 그래프를 기초로 하고 있지만, 다양한 3차

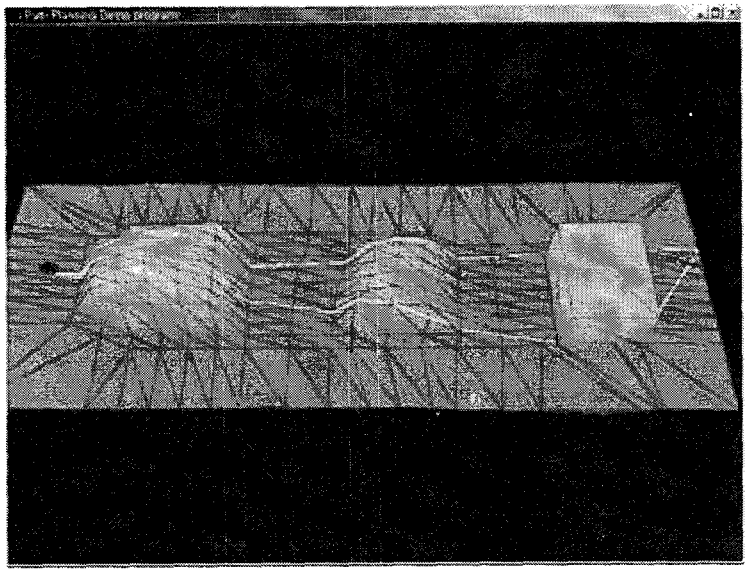


그림 6. 3차원 지형에서의 가시성 검사

원 지형에 적용하기 위해서 기존의 방법처럼 간단하게 가시성 그래프를 생성하지 않는다. 또한 본 논문에서 제시한 알고리즘은 다양한 3차원 장애물 지형에서 경로 탐색을 수행하여 경로 계획을 수립한다.

경로 계획 수립에서 중요한 요소는 가시성 정보의 최대지점을 어디로 정할 것인가에 있다. 본 논문에서 제안한 경로 계획 방법은 다음과 같다. 캐릭터가 시작지점에서 목표지점까지 장애물을 포함한 지형에서 가시성 검사를 수행하며, 장애물의 정점들을 이용하여 가시선을 생성한다. 가시성 검사는 장애물의 정점을 찾아 현재의 위치를 기반으로 목표지점에 가장 효율적으로 이동할 수 있는 위치를 탐색하는 것을 목적으로 한다. 가시성 검사에 의해 효율적인 정점이라고 판단된 위치는 현재의 위치에 연결선을 생성하여 가시선을 만든다. 현재의 지형 정보로 판단할 경우, 생성된 가시선은 캐릭터가 현재 위치에서 목표지점에 가장 빠르게 도착할 수 있는 직선을 의미한다. 이와 같은 가시선을 기준으로 캐릭터는 최적화된 경험적 경로 탐색을 수행한다. 캐릭터가 가시선의 종점에 위치했을 때, 목표지점까지 최단 경로에 나타난 가장 가까운 다음 장애물의 정점을 찾아 가시선을 확장하고 가시성 검사를 수행하여 최적화된 경험적 경로 탐색을 수행한다. 이렇게 목표지점까지 가시성 검사와 가시선 생성 및 경로 탐색이라는 행위를 반복함으로써 캐릭터는 시작지점에서 목표지점까지 빠르게 이동할 수 있다.

경로 탐색에서 고려할 사항은 경로 이동에 따른 가중치이다. 예를 들어 캐릭터가 3차원 지형에서 이동할 경우, 위험 지역이나 오르기 힘든 언덕과 같은 곳은 위험성과 에너지 문제로 회피하여야 한다. 진흙으로 이루어진 지형에서 캐릭터가 이동할 때 소모되는 에너지는 단단한 흙으로 이루어진 지형에서 캐릭터가 이동할 때 소모되는 에너지보다 많다. 캐릭터가 평원에서 이동을 할 때보다 경사가 급한 협곡에서 이동할 때가 위험성이 높다. 이러한 여러 가지 문제들을 캐릭터의 경로 계획에 고려하여야 최적화된 경로 탐색을 수행할 수 있다. 그러므로 에너지와 안전성은 경험적 탐색의 추정 함수에 반영되어야 현실적인 캐릭터의 움직임을 발생시킬 수 있다.

경로 계획에 이용되는 추정 값의 생성은 다음과 같다. 가시성 정보를 고려하지 않는 경우, 경로 탐색에 대한 간단한 추정은 현재지점(s)에서 목표지점(G)까지의 유클리디안 거리(d)이며, 이것을 d(s,G)로 표현한다. 이 경우, 가시성 정보를 다음과 같이 정의한다. 현재지점 s에서 가시적인 점들 $v_1, v_2, \ldots, v_n$이라 가정한다. 목표지점까지의 경로는 장애물을 회피하기 위해 가시적인 점들에서 하나의 점은 통과해야 하기 때문에 추정 함수는 식 (1)과 같이 정의한다.

$$h(s)=\min[d(s, v_1)+d(v_1, G), d(s, v_2)+d(v_2, G), \cdots, d(s, v_n)+d(v_n, G)] \quad (1)$$

다시 말하면, 현재지점 s에서 목표지점 G까지의 측정 거리는 현재지점 s에서 가시적인 점들 중에 한 개 점까지의 거리와 그 점에서 목표지점 G까지의 거리를 합한 것이다. 이 새로운 h(s)는 항상 목표지점까지의 최단 거리의 경로를 찾아 줄 수는 있지만, 직선 거리보다는 길다. 그러므로 식 (2)와 같이 나타낼 수 있다.

$$d(s, G) \leq h(s) \leq h^*(s) \quad (2)$$

공식 (1)과 같이 추정 함수 정의에 의해 탐색 알고리즘은 장애물 근처에서 불필요한 상태를 방문하지 않고 목표지점까지 경로 탐색을 수행하기 때문에 효과적으로 탐색 공간을 줄여 준다. 또한 모든 상태에 대해 가시성 체크가 필요하지 않아, 경로 탐색에 소요되는 탐색 시간을 줄여 준다.

본 논문에서 제안한 가시성 검사를 이용한 탐색 기법을 적용한 캐릭터가 3차원 장애물 지형에서 이동할 때 경로 탐색 방법은 다음과 같다. 가시적인 점들 중 인접한 다음 상태에 도달하지 않는 점에서는 후속 상태(자식 상태)로 분기하지 않으며, 가시성 검사도 수행하지 않는다. 만약 현재 상태에서 가시적인 점들 중에 하나에 도달한다면, 도달한 점에서 가시성은 다시 테스트 되어지고, 새로운 가시성 점들의 리스트를 구성한다. 새로운 가시성 점들에 대한 리스트는 새로운 추정 함수를 계산하기 위해 사용한다.

가시성 검사를 이용한 탐색 알고리즘을 그림 7에 나타내었다.

4. 실 험

본 논문에서 제안한 알고리즘을 실험하기 위해 C++와 OpenGL을 사용하여 데모 프로그램을 구현하였다. 데모 프로그램에서 3차원 지형과 장애물은 3DMax를 이용하여 생성하였고, 지형 공간은 다각형

```
Procedure A*-with-visibility-test (Start, Goal, obstacles)
  check visibility from Start
  // P is a list of visible points - one of which must be visited on the path to the Goal
  P ← visible vertices of obstacles or Goal = (v1, .., vn)
  g ← 0
  h ← min [ d(Start, v1)+d(v1, Goal), ··· , d(Start, vn)+d(vn, Goal) ]
  // OPEN is a priority queue sorted by g+h
  OPEN ← {(Start, P, g, h)}

  While OPEN ≠ ∅
        (s, P, g, h) ← first of OPEN
        if (s = Goal) return success
        else
        generate children ci
        // if current position is adjacent to one of the points in P, find new P
        if ((s ∈ P) and (all ci ∈ P)) then
              check visibility from s
              P ← visible vertices of obstacles or Goal
        // compute heuristic using currently visible points
        vi, .., vk ← elements of current P
        for each child ci
              gi ← g + d(s, ci)
              hi ← min [ d(ci, v1)+d(v1, Goal), .. , d(ci, vk)+d(vk, Goal) ]
              insert (ci, P, gi, hi) into OPEN
  return fail
```

그림 7. 가시성 검사를 이용한 탐색 알고리즘

집합으로 표현하였다. 3차원 지형과 장애물들은 3DMax로 생성한 후, OpenGL상의 좌표 체계에 맞게 변환하였으며, 이렇게 생성된 3차원 지형을 이동망으로 활용하여 실험을 수행하였다.

실험은 크게 두 가지로 나뉜다. 첫 번째 실험은 여러 3차원 장애물 지형들에서 경험적 탐색 알고리즘과 본 논문에서 제안한 가시성 검사를 이용한 탐색 알고리즘의 경로 탐색량에 대한 비교 실험이다. 첫 번째 실험에서 장애물 지형을 생성할 때 다각형의 수량을 서로 다르게 하여 다각형의 수량에 따른 탐색 성능을 비교 실험한다. 두 번째 실험에서는 지형을 생성할 때 사용하는 장애물의 사용 수량을 서로 다르게 하여 장애물 사용량에 따른 탐색 성능을 비교 실험한다.

그림 8은 가시성 검사를 이용한 탐색 알고리즘과 경험적 탐색 알고리즘을 3차원 장애물 지형에서 비교 실험한 결과 화면이다. 그림 8(a)는 두 개의 일자형 장애물이 설치된 3차원 장애물 지형에서 경험적 탐색 알고리즘의 탐색 결과이다. 좌측에 작은 원은 시작지점을 나타내고, 우측의 작은 원은 목표지점을 나타낸다. 지형에서 음영이 나타난 부분은 경험적 탐색 알고리즘이 탐색을 수행한 다각형(삼각형)들을 나타낸다. 경험적 탐색 알고리즘은 두 개의 일자형 장애물로 인해 시작지점에서 목표지점까지 많은 다각형들을 탐색하고 나서야 목표지점까지 도달하였다. 그림 8(b)는 두 개의 일자형 장애물이 설치된 3차원 장애물 지형에서 가시성 검사를 이용한 탐색 알고리즘의 탐색 결과이다. 좌측의 작은 원으로 나타난 시작지점에서 우측의 작은 원으로 나타난 목표지점까지 탐색을 수행한 다각형들에는 음영을 나타내었다. 그림 8(b)에서 음영으로 나타난 부분을 보면 그림 8(a)보다 매우 적은 수의 다각형들에서만 음영이 나타난다. 이와 같이 적은 수량의 다각형들에서만 음영이 나타난다는 것은 가시성 검사를 이용한 탐색 알고리즘이 경험적 알고리즘보다 적은 탐색량으로 시작지점에서 목표지점까지 캐릭터를 이동 시킬 수 있다는 것을 나타낸다. 이와 같은 결과가 발생한 이유는 가시성 검사를 이용한 탐색 알고리즘은 3차원 장

(a) (b)

그림 8. 경로 탐색 시 방문 상태 비교 : (a) 경험적 탐색, (b) 가시성 검사를 이용한 탐색

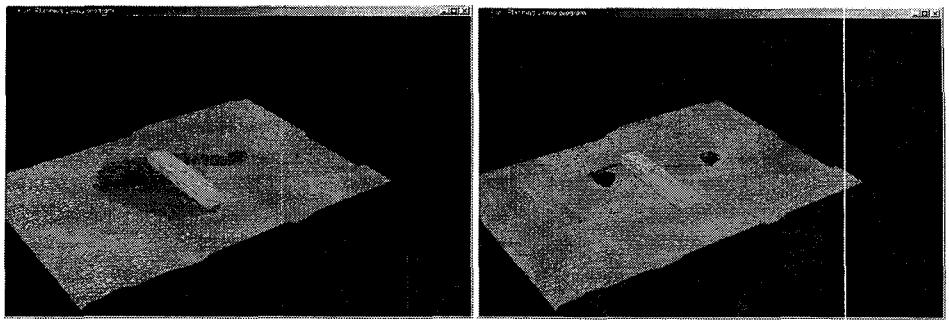
애물 지형에서 모든 장애물들과 목표지점에 대해 가시성 검사를 수행하면서 탐색을 진행함으로 경험적 탐색 알고리즘에 비해 많은 수의 불필요한 상태들을 탐색하지 않고 경로 탐색을 수행하기 때문이다.

그림 9는 다양한 형태의 3차원 장애물 지형에서 가시성 검사를 이용한 탐색 알고리즘과 경험적 탐색 알고리즘을 비교 실험한 결과 화면이다. 그림 9의 결과를 보더라도 본 논문에서 제안한 가시성 검사를 이용한 탐색 알고리즘이 경험적 탐색 알고리즘보다 다양한 형태의 장애물 지형들에서 많은 수의 불필요한 탐색을 줄여준다는 것을 알 수 있다.

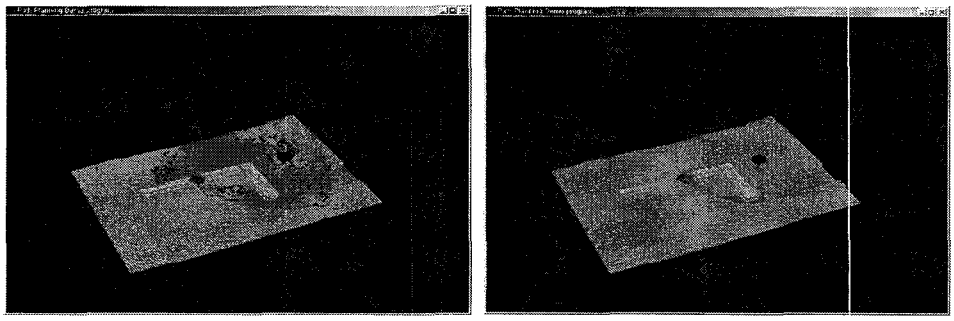
일반적으로 게임이나 가상현실에서 자세한 형태로 지형을 표현하기 위해서는 지형을 구성하는 요소인 격자를 많이 사용하여야 한다. 격자는 캐릭터가 이동을 할 때 활용하는 요소이기 때문에, 많은 수의 격자가 활용되었다는 것은 캐릭터의 이동에 많은 탐색 시간이 요구한다는 것을 의미한다. 지형 생성의 효율성에 의해 사용되는 이동망도 자세한 지형 표현에 따른 탐색 시간의 증가에 자유롭지 못하다. 이동망에서도 지형을 세밀하게 표현하였다면 많은 수의 다각형들을 사용한 것이기 때문에 캐릭터가 경로 탐색을 수행할 때 많은 수의 다각형들을 탐색해야 한다. 자세한 지형 생성에 따른 탐색 시간의 증가는 지형 생성에 강한 제약 조건으로 작용한다. 본 논문에서 제안한 가시성 검사를 이용한 탐색 알고리즘은 장애물이 설치된 3차원 장애물 지형에서 경험적 탐색에 비해 적은 탐색량으로 경로 탐색을 수행한다는 것을 본 실험으로 확인하였다.

표 1은 서로 다른 상태 총수를 나타내는 장애물들에서 경험적 탐색과 가시성 검사를 이용한 탐색의 방문 횟수를 비교 실험한 결과이다. 지형-1을 생성할 때 사용된 다각형의 수량은 940개이며, 지형-1에서 경험적 탐색 알고리즘이 경로 탐색에 이용한 다각형 수량은 243개로 가시성 검사를 이용한 탐색 알고리즘보다 221개가 더 많았다. 즉, 가시성 검사를 이용한 탐색 알고리즘은 경험적 탐색 알고리즘에서 탐색한 상태의 9.05%만을 활용하여도 목표지점까지 도달한다는 것을 나타낸다. 7개의 모든 지형에서 본 논문에서 제안한 가시성 검사를 이용한 탐색 알고리즘이 경험적 탐색 알고리즘보다 우수하게 나타났다.

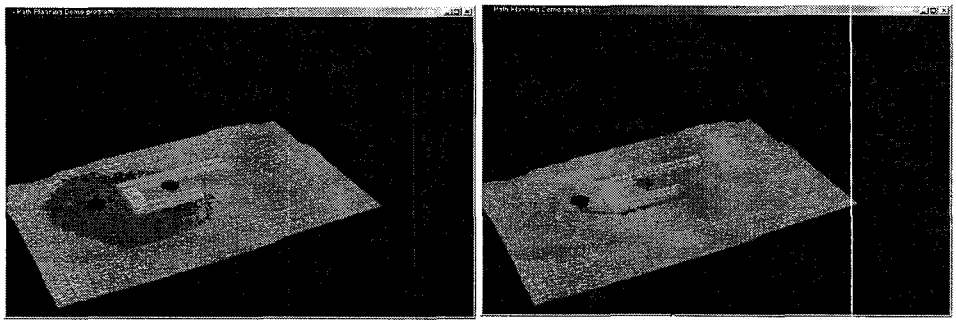
그림 10은 3차원 장애물 지형을 구성하는 다각형들의 총수가 증가함에 따른 탐색량 증가율을 나타낸 실험 결과이다. 지형-1은 가장 적은 수량의 다각형들을 사용하였고, 지형-7은 가장 많은 수량의 다각형을 사용하여 생성하였다. 지형-1에서 경로 탐색을 수행할 경우에 경험적 탐색 알고리즘은 전체 다각형 총수의 25.85%를 활용하였고, 가시성 검사를 이용한 탐색 알고리즘은 전체 다각형 총수의 2.34%를 활용하였다. 탐색량은 메모리 사용과 탐색 시간에 많은 영향을 미치는 요소로, 탐색량은 7개의 모든 지형에서 가시성 검사를 이용한 탐색 알고리즘과 경험적 탐색 알고리즘은 극심한 차이를 나타냈다. 본 논문에서 제안한 가시성 검사를 이용한 탐색 알고리즘은 다각형의 사용 수량이 증가하여도 탐색에 활용하는 다각형



(a)



(b)



(c)

그림 9. 다양한 형태의 3차원 장애물 지형에서의 탐색 영역 비교 : (a) 일자형 장애물에서의 탐색 영역 비교, (b) 'ㄱ'자 장애물 지형에서의 탐색 영역 비교, (c) 'ㄷ'자 장애물 지형에서의 탐색 영역 비교

표 1. 상태 총수에 따른 방문 횟수 비교

지형	상태 총수	상태 방문 총수		경험적 탐색에 대한 가시성 검사 탐색 비율
		경험적 탐색	가시성 검사를 이용한 탐색	
지형-1	940	243	22	9.05 %
지형-2	1160	456	43	9.43 %
지형-3	1404	520	47	9.04 %
지형-4	1672	530	52	9.81 %
지형-5	1964	612	58	9.48 %
지형-6	2279	732	68	9.29 %
지형-7	2628	824	81	9.83 %

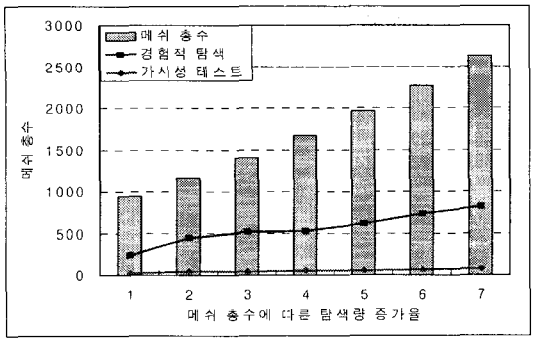


그림 10. 다각형 총수에 따른 탐색량 증가율

의 수량은 경험적 탐색 알고리즘에 비해 매우 적었다. 그림 10에 나타난 바와 같이 다각형 총수가 증가하면 경험적 탐색 알고리즘의 탐색량도 급격히 증가한다. 그러나 가시성 검사를 이용한 탐색 알고리즘은 다각형 총수가 급속도록 증가하여도 거의 비슷한 탐색량을 나타냈다.

표 2는 복합 장애물 지형에서 경험적 탐색과 가시성 검사를 이용한 탐색의 방문 횟수 비교에 대한 실험 결과이다. 복합 장애물 지형 실험에서는 장애물의 수량을 변경하면서 경험적 탐색과 가시성 검사 탐색에 대한 성능을 측정하였다. 지형-1을 생성할 때 사용된 다각형은 2,276개이고 장애물은 1개이다. 지형-1에서 경험적 탐색 알고리즘에서 탐색에 사용한 다각형의 총수는 1,242개이고, 가시성 검사를 이용한 탐색 알고리즘에서 탐색에 사용한 다각형의 총수는 126개로 가시성 검사를 이용한 탐색이 우수한 성능을 나타냈다. 지형-1에서 경험적 탐색에 대한 가시성 검사 탐색 비율은 10.14%로 나타났으며, 지형-6과 지형-7에서 경험적 탐색에 대한 가시성 검사 탐색 비율은 각각 9.29%와 9.83%를 나타내었다. 장애물의 수량이 증가되어도 가시성 검사를 이용한 탐색은 경험적 탐색에 비해 높은 탐색 효율을 나타낸다는 것을 복합 장애물 실험을 통해 알 수 있었다.

5. 결 론

이동망에서는 지형을 표현할 때 다각형을 이용하여 지형을 표현하기 때문에 표현할 수 있는 영역이 넓다. 이와 같은 이유로 3차원 게임이나 가상현실에서 지형 표현 시, 근래에는 이동망을 활용하고 있는 추세이다.

3차원 게임이나 가상현실에서는 비현실적인 장애물이나 지형은 사용자들에게 흥미를 유발시킬 수 없기 때문에, 사용자들에게 현실감을 제공하기 위해 지형을 사실적으로 묘사하는 경우가 많다. 세밀한 지형 표현은 캐릭터가 탐색할 영역들이 증가한다는 것을 의미함으로, 메모리 사용과 탐색 시간에 직접적인 악영향을 미치는 요소로 작용한다. 이와 같은 문제는 이동망에서도 발생한다.

본 논문에서는 이와 같은 문제를 해결하기 위해 이동망으로 표현된 3차원 장애물 지형에서 효율적인 경로 탐색을 수행할 수 있는 가시성 검사를 이용한 탐색 알고리즘을 제안한다. 가시성 검사를 이용한 탐색 알고리즘은 3차원 장애물 지형에 적합한 가시성 검사를 수행하여 3차원 장애물 지형의 탐색 공간을 축소시키며, 3차원 장애물 지형에서 캐릭터의 경로 탐색 속도를 향상시키고 빠른 탐색 연산을 위해 3차원 지형을 2차원으로 표현할 수 있도록 이동망을 구성한다. 또한, 시작지점에서 목표지점까지 효율적인 경로 탐색을 수행할 수 있도록 가시성 검사를 이용한 탐색에서는 지형에 대한 정보를 추정 함수에 적용하였으며, 지형 정보를 추정 함수에 적용시킨 가시성 검사를 이용한 탐색 알고리즘은 경험적 탐색 알고리

표 2. 복합 장애물 지형에서의 방문 횟수 비교

실험	상태 총수	장애물 총수	상태 방문 총수		경험적 탐색에 대한 가시성 검사 탐색 비율
			경험적 탐색	가시성 검사를 이용한 탐색	
지형-1	2267	1	1242	126	10.14 %
지형-2	2278	2	1201	114	9.49 %
지형-3	2307	3	1420	134	9.44 %
지형-4	2212	4	1325	123	9.28 %
지형-5	2187	5	1462	134	9.17 %
지형-6	2104	6	1310	119	9.08 %
지형-7	2234	7	1274	116	9.11 %

즘보다 탐색 영역을 대폭 축소하면서 목표지점까지 효율적인 경로 탐색을 수행한다.

상태 총수에 따른 방문 횟수 비교 실험에서 가시성 검사를 이용한 탐색은 경험적 탐색에 비해 적은 상태 방문으로 목표지점에 도달하였으며, 경험적 탐색에 대한 가시성 검사를 이용한 탐색의 비율은 평균 9.41%를 나타냈다. 상태 총수에 따른 방문 횟수 비교 실험에서 상태 총수가 증가하면 경험적 탐색은 상태 방문 횟수가 매우 빠르게 증가하지만, 가시성 검사를 이용한 탐색은 완만하게 나타났다. 복합 장애물 지형에서의 방문 횟수 비교 실험에서는 경험적 탐색 알고리즘에 대한 가시성 검사를 이용한 탐색 알고리즘의 탐색량 비율은 평균 9.38%를 나타냈으며, 장애물의 개수를 증가하여도 가시성 검사를 이용한 탐색 알고리즘은 효과적인 탐색 성능을 나타낸다는 것을 복합 장애물 실험에서 확인하였다.

본 논문에서 제안한 가시성 검사를 이용한 탐색 알고리즘은 가시적인 영역에 있는 장애물의 정점을 먼저 탐색하기 때문에 불필요한 탐색을 줄여주는 장점이 있다. 가시성 검사를 이용한 탐색 알고리즘은 경로 탐색에 가장 많이 활용되는 경험적 탐색 알고리즘보다 효율적인 탐색 알고리즘이란 것을 몇 가지 실험을 통해 확인하였다.

향후에는 가시성 검사를 이용한 탐색 알고리즘을 다면 경사도에서 적용할 수 있는 연구와 효율적인 경로 탐색을 위해 경로 탐색 시 에너지 대사율을 적용할 수 있는 연구가 필요하다.

참 고 문 헌

[1] F. Durand, G. Drettakis, and C. Puech, "The 3D visibility complex," *ACM Transactions on Graphics*. Vol. 21, No. 2, pp. 176-206, 2002.

[2] W. Sterren, "Terrain reasoning for 3D action games," *Proceedings of the CGF-AI Conference*, pp. 307 - 323, 2001.

[3] T. Sederberg, J. Zheng, D. Sewell, and M. Sabin, "Non-uniform Subdivision Surface," *Proceedings of the ACM SIGGRAPH'98*, pp. 387-394, 1998.

[4] M. Yamamoto, Y. Isshiki, and A. Mohri, "Collision free minimum time trajectory planning for manipulators using global search and gradient method," *Proceedings of the IEEE International Conference on Advanced Robotic System and The Real World*, pp. 2184-2191, 1994.

[5] S. Rabin, "A* Speed Optimizations," *Game Programming Gems*, Charles River Media, 2000.

[6] G. Snook, "Simplified 3D Movement and Pathfinding Using Navigation Meshes," *Game Programming Gems*, Charles River Media, 2000.

[7] S. Wismath, "Computing the full visibility graph of a set of line segments," Information Processing Letters, No. 42, pp. 257-261, 1992.

[8] E. Angel, *Interactive Computer Graphics*, Addison Wesley, 2000.

[9] B. Stout, "The Basics of A* for Path Planning," *Game Programming Gems*, Charles River Media, pp. 254-263, 2002.

[10] M. Berg, M. Kreveld, M. Overmars, and O. Schwarzkopt, *Computational Geometry, Algorithms and Application*, Springer-Verlag, pp. 305-314, 1997.

[11] S. Ghosh and D. Mount, "An output sensitive algorithm for computing visibility graphs," *Proceedings of the 28th Annual IEEE Symposium on Foundations of Computer Science*, 1987.

[12] H. Overmars and E. Welzl, "New Methods for Computing Visibility Graphs," *Proceedings of the 4th annual Symposium on Computational Geometry*, pp. 164-171, 1988.

[13] M. Pocchiola, and G. Vegter, "Computing the Visibility Graph via Pseudo-Triangulations," *Proceedings of the 7th annual Symposium on Computational Geometry*, pp. 248-257, 1995.

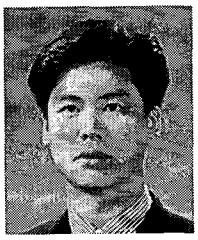
[14] H. Alt, M. Godau, and S. Whitesides, "Universal 3-dimensional visibility representations for graphs," *Computational Geometry: Theory and Applications*, Vol. 9, No. 1-2, pp. 111-125, 1998.

[15] J. Hancock, "Navigating Doors, Elevators, Ledges, and Other Obstacles," *AI Game Programming Wisdom*, Charles River Media, 2002.

[16] H. Brönnimann, M. Glisse, and D. Wood, "Cost-optimal quadtrees for ray shooting," *Proceedings of the 14th Canadian Conference on Computational Geometry*, pp. 109-112, 2002.

[17] H. Choset, I. Konukseven, and A. Rizzi, "Sensor Based Planning: A Control Law for Generating the Generalized Voronoi Graph," *Proceedings of the 8th International Conference on Advanced Robotics(ICAR'97)*, 1997.

[18] M. Eck and H. Hoppe, "Automatic Reconstruction of B-Spine Surfaces of Arbitrary Topological Type," *Proceedings of the 23rd International Conference on Computer Graphics and Interactive Techniques (SIGGRAPH '96)*, pp. 325-334, 1996.



김 형 일

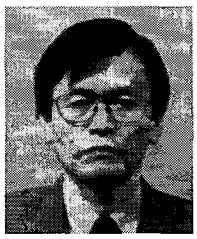
1996년 목원대학교 수학과 졸업(이학사)
1996년~1998년 (주)경기은행
2001년 동국대학교 대학원 컴퓨터공학과(공학석사)
2004년 동국대학교 대학원 컴퓨터공학과(공학박사)
관심분야 : 멀티미디어 정보처리, 게임, 지능형 에이전트, 기계학습, 정보검색, e-learning

정 동 민

2003년 동국대학교 컴퓨터멀티미디어공학과 졸업(공학사).
2005년 동국대학교 대학원 컴퓨터공학과(공학석사).
2005년~현재 동국대학교 대학원 컴퓨터공학과(박사과정).
관심분야 : 게임, 인공지능

엄 기 현

1975년 서울대학교 공과대학 응용수학과 공학사
1977년 한국과학기술원 전산학과 이학석사
1994년 서울대학교 대학원 컴퓨터공학과 공학박사
1978년~2006년 6월 동국대학교 컴퓨터멀티미디어공학과 정교수
2006년 7월~현재 동국대학교 영상미디어학부 게임멀티미디어공학과 정교수
1998년~현재 한국멀티미디어학회 부회장, 자문위원, 현재 수석 부회장
2004년~현재 한국게임학회 부회장, 현재 자문위원
관심분야 : 게임 시스템 설계, 멀티미디어 시스템 설계, 멀티미디어 정보처리, 멀티미디어 데이터베이스



조 형 제

1973년 부산대학교 전자공학과(학사)
1975년 한국과학기술원 전기·전자공학과(공학석사)
1986년 한국과학기술원 전기·전자공학과(공학박사)
1986년~현재 동국대학교 멀티미디어학과 교수
관심분야 : 컴퓨터 그래픽스, 게임, 디지털사운드처리, 컴퓨터비전

김 준 태

1986년 서울대학교 제어계측공학과 졸업(공학사)
1990년 미국 Univ. of Southern California, Electrical Engineering-Systems(M.S.)
1993년 미국 Univ. of Southern California, Computer Engineering(Ph.D.)
1994년~1995년 미국 Southern Methodist University, Computer Science and Engineering(Postdoc)
1995년~현재 동국대학교 컴퓨터공학과 교수
관심분야 : 기계학습, 데이터마이닝, 정보검색, 지능형 에이전트. e-learning