Journal of The Korea Society of Computer and Information
Vol. 20, No. 1, January 2015
www.ksci.re.kr
http://dx.doi.org/10.9708/jksci.2015.20.1.085

그리드 기반 맵에서 꼭지점 정보를 이용한 휴리스틱의 설계

김 지 혜*, 정 예 원*, 유 견 아*

Design of Heuristics Using Vertex Information in a Grid-based Map

Ji-Hyui Kim*, Ye-Won Jung*, Kyeon-Ah Yu*

요 약

컴퓨터 게임 배경이 정교하게 표현되면서 그리드 기반으로 표현된 게임 맵에서 A* 알고리즘을 이용한 경로 찾기는 전체 게임 성능을 저해하는 요인이 되고 있다. 셀 단위의 세밀한 표현으로 상태 공간이 커져 탐색 시간이 증가하기 때문이다. 본 논문에서는 정규 그리드로 표현된 컴퓨터 게임 배경을 꼭지점 리스트로 된 다각형 기반 맵으로 변환하고 다각형의 꼭지점에 대한 가시성 정보를 이용하여 효율적인 경로 찾기가 가능하게 하는 방법을 제안한다. 다각형 기반 맵으로의 변환은 오프라인으로 전처리하여 실시간 쿼리에는 영향을 미치지 않도록 하며 꼭지점의 가시성 정보를 이용하는 휴리스틱을 설계함으로서 추정의 정확도를 높여 경로 탐색 시에 방문하는 노드수를 획기적으로 감소시키도록 한다. 시뮬레이션에서는 제안한 방법들이 그리드 기반 방식의 장점을 유지하면서 탐색 공간과 탐색 시간을 효율적으로 감소시킴을 확인한다.

▸ Keywords : 경로 찾기, A* 알고리즘의 휴리스틱, 그리드 기반 맵

Abstract

As computer game maps get more elaborate, path-finding by using A* algorithm in grid-based game maps becomes bottlenecks of the overall game performance. It is because the search space becomes large as the number of nodes increases with detailed representation in cells. In this paper we propose an efficient pathfinding method in which the computer game maps in a regular grid is converted into the polygon-based representation of the list of vertices and then the visibility information about vertices of polygons can be utilized. The conversion to the polygon-based map does not give any effect to the real-time query process because it is preprocessed offline. The number of visited nodes during search

•제1저자 : 김지혜 •교신저자 : 유견아
•투고일 : 2014. 10. 8, 심사일 : 2014. 11. 26, 게재확정일 : 2014. 12. 1.
* 덕성여자대학교 컴퓨터학과(Dept. of Computer Science, Duksung Women's University)
※ 본 연구는 2013년도 덕성여자대학교 교내연구비 지원에 의해 수행되었다.

can be reduced dramatically by designing heuristics using visibility information of vertices that make the accuracy of the estimation enhanced. Through simulations, we show that the proposed methods reduce the search space and the search time effectively while maintaining the advantages of the grid-based method.

▸ Keywords : Path-finding, Heuristics for A* algorithm, Grid-based maps

I. 서 론

그리드 맵은 간단하고 쉽게 게임 배경을 만들 수 있다는 장점 때문에 컴퓨터 게임에서 가장 많이 사용되는 배경표현 방식이다. 오늘날에는 높은 해상도로 정교한 배경 표현도 가능해졌는데 이와 같이 고해상도 그리드 맵에서 A* 알고리즘을 이용하여 경로를 탐색하는 경우의 문제점은 상태 공간의 크기가 너무 커서 탐색 시간이 많이 걸린다는 것이다. 이를 해결하기 위한 기존의 접근 방식은 2가지로 나뉜다. 첫째는 상위의 추상 단계에서 대략적인 경로를 찾고 하위 단계에서 세부적인 경로 계획을 실행하는 계층적 방법 [1][2]이고 둘째는 정확한 휴리스틱 함수를 고안하여 빠른 탐색을 유도하는 방법이다 [3][4]. 본 논문에서는 이 두 가지 방법을 복합한 새로운 해결 방식을 제안한다. 우선 그리드 맵을 다각형 기반 맵으로 변환하여 다각형의 꼭지점을 경로 찾기에 이용하는 추상화 단계와 꼭지점에 대한 가시성 정보를 이용한 휴리스틱을 설계를 복합하는 것이다.

그리드 기반 맵에서 장애물의 꼭지점을 찾아내는 과정은 오프라인으로 전처리되는데 이를 위해 벽타기 알고리즘으로 알려진 우수법(right hand rule)과 유사한 방식의 알고리즘을 개발한다. 일단 다각형 기반 맵으로 변환되면 로봇 경로계획에서 오랜 기간 동안 사용된 많은 방법들을 적용할 수 있지만 본 논문에서는 최단 경로를 찾는 것이 보장되어 있는 가시성 그래프 개념을 이용하도록 한다. 다각형 기반 환경에서 A* 알고리즘에 의한 최단 경로는 장애물의 꼭지점을 지난다는 점에 착안하여 기존의 그리드 기반 맵에서 A* 탐색을 할 때 사용하던 직선 거리 휴리스틱에 비해 많은 정보를 포함하는 휴리스틱을 설계한다.

본 논문에서 제안하는 계층화 방법은 그리드를 특정한 방법으로 구획화하여 추상화하는 기존 연구와는 달리 가시 정보를 이용하여 2중 탐색을 수행한다는 점이 차별화되며 그리드 기반 맵과 그래프 기반 맵의 장점을 동시에 취할 수 있다는 장점이 있다 [5]. 또한 제안한 휴리스틱의 허용성을 증명함으로서 가시 정보를 이용한 세부적인 경로 찾기를 통해 개개의 셀이 갖는 정보를 고려한 경로 찾기가 되는 동시에 탐색의 최적성도 보장함을 보여준다. 시뮬레이션을 통해 제안한 방식이 기존의 그리드 기반 A* 탐색에 비해 방문 노드수를 크게 줄이면서 효율적인 경로 탐색이 됨을 확인한다.

본 논문의 구성은 다음과 같다. 2장에서는 그리드 기반 맵에서의 경로 찾기에 관한 연구를 조사하고 3장에서는 그리드 기반 맵을 다각형 기반 맵으로 변환하는 알고리즘을 소개하고 이 과정에서 고려해야 할 세부적인 문제에 대해 살펴본다. 4장에서는 다각형 기반으로 표현된 맵에서 꼭지점의 가시성을 이용하는 휴리스틱과 실시간 쿼리 단계를 소개한다. 5장에서는 구현 방법과 시뮬레이션 결과에 대해 살펴보고 6장에서 결론을 맺는다.

II. 관련 연구

게임 개발 분야에서 가장 일반적인 경로 계획 방법은 배경을 그리드 기반으로 나누고 빈 그리드에 대해 A* 탐색 방법을 적용하는 것이다. 이 방법은 경로가 존재하면 반드시 찾고, 찾은 경로는 최적이라는 장점이 있지만 세밀한 표현을 위해 그리드의 수가 많아지면 상태 공간이 커지고 탐색 시간이 오래 걸린다는 단점이 있다[6][7]. 그리드에서 경로 찾기의 성능 향상을 위한 노력은 매우 다양하게 이루어지며 [8]에서는 이러한 개별 연구들에 대한 객관적인 성능 평가를 위해 벤치마크 지도 집합을 제안하고 이를 이용한 대회도 개최하였다. 경로 찾기 성능 향상을 위한 전형적인 방법으로는 계층적 경로 찾기와 정확한 휴리스틱 설계 방법이 있다. 두 단계 계층으로 나눈 경로 찾기의 제안은 컴퓨터 게임의 초창기에 [2]에서 제안되었고 [1][9]에서는 두 단계 이상의 다단계화 방법으로 확장되었다. [2]에서는 배경 맵을 건물이나 야외 배경을

블록 단위로 추상화하여 경로 찾기를 수행한 후, 각 블록의 중간 지점을 지나는 세부 경로 찾기를 수행하도록 하여 최적성을 희생하지만 빠른 연산을 한다. [1]에서는 [2]과 유사하지만 2단계로 고정하지 않고 필요에 따라 다단계로 추상화하며 블록 수준의 길찾기를 전처리하고 결과를 캐시하여 온라인 연산 비용을 줄였으며 [9]에서는 나누어진 구역을 다시 도달 가능한 영역으로 나누어 각 영역을 잇는 그래프를 생성하여 메모리 효율적인 추상화를 제안하였다. 본 논문에서는 그래프를 생성하여 추상화하는 방법은 유사하지만 그래프를 가시성 기반으로 생성하는 것이 차별화되어 있다.

휴리스틱 함수의 정확도를 높여서 빠른 탐색이 가능하도록 하는 방법에서는 A* 알고리즘에서 보편적으로 사용되는 허용 가능한 휴리스틱인 유클리디언 거리 대신, 실제 거리에 가까운 휴리스틱으로 개선하는 시도들이 주를 이룬다. [10]에서는 현재 질의와는 무관한 영역을 미리 찾아 탐색할 후보 공간으로부터 제거하는 데드 엔드(dead end) 휴리스틱과 그리드 맵을 분할하고 분할된 각 영역으로 통하는 게이트웨이를 이용하여 휴리스틱을 계산하는 게이트웨이 휴리스틱 두 가지를 제안하였다. 이 방법은 방과 복도들로 이루어진 환경에서만 잘 동작하도록 특화되어 있다는 단점이 있다. 모든 노드 쌍의 실제 거리를 저장하여 휴리스틱(true distance heuristics, TDH)으로 이용하는 것을 제안하는 [4]에서는 모든 값을 저장하는데 소요되는 메모리를 효율적으로 개선하는 방법인 차등(differential) 휴리스틱과 정규형 휴리스틱(canonical heuristics)을 제안하였다. [11]에서는 모든 노드쌍의 거리를 저장하고자 하는 TDH를 일정 지역으로 분할하고 분할된 영역으로 들어가는 문들 사이의 거리를 저장하는 포털 기반 TDH 휴리스틱의 사용을 제안하였다.

계층적 경로찾기의 단점은 서브최적(suboptimal) 경로를 찾게 된다는 것인데 [12][13]에서는 이를 보완하는 방법들도 제안되었다. [12]에서는 자유공간을 직사각형 공간으로 분할한 후, 내부 셀은 제거하고 둘레만 남겨 둘레를 연결한 매크로 에지를 소개하여 노드수와 에지수를 줄이는 방법을 제안했으며 [13]에서는 점프 포인트라는 특정한 노드를 결정하고 그 노드 사이에 있는 노드들은 확장하지 않아도 됨을 증명하여 전처리 과정 없이도 빠른 경로 탐색이 가능하게 하였다.

반드시 통과하여야 하는 지점을 거치는 휴리스틱을 기존의 직선거리 휴리스틱에 비해 더 정확한 휴리스틱이라고 할 수 있다. [14]에서는 네비게이션 메쉬에서 가시성 정보를 이용하여 반드시 거치는 지점의 후보를 정하고 이를 이용한 휴리스틱의 사용을 제안하였다.

III. 전처리 단계: 장애물 꼭지점 리스트와 가시성 그래프 생성

고해상도 그리드 맵에서 효율적인 경로 찾기 수행을 위한 전처리 단계에서는 우선 그리드 맵을 구성하는 장애물들의 꼭지점을 찾아 순서 리스트로 표현하고 그를 이용한 축소형 가시성 그래프(reduced visibility graph)를 구축한다. 전처리 과정은 실제 경로 찾기 쿼리에 앞서 오프라인으로 처리된다.

그리드에서 장애물의 꼭지점 리스트(vertex list)를 찾는 알고리즘은 벽타기 알고리즘으로 알려진 우수법과 유사한 방식으로 진행한다. 그리드의 타일들을 좌측 최상단부터 차례로 스캐닝 하여 최초로 나온 장애물 타일이 '시작' 타일이 된다. 기본적인 원리는 한 칸씩 진행하면서 진행 방향에 있는 타일이 장애물이 아니면 우측으로 방향으로 90도 회전하여 한 칸씩 진행하는 것인데, 진행 방향에 있는 타일을 확인하기 전에 좌측 타일이 장애물이 아님을 확인해야 한다. 좌측 타일이 장애물인 경우에는 좌측 방향으로 90도 회전하여 진행해야 한다. 이 과정은 그림 1의 순서도로 정리할 수 있다. 순서도의 입력은 타일의 정보가 저장된 그리드 맵과 '시작' 타일(최상단 좌측에 있는 장애물 타일)이며 출력은 입력된 '시작' 타일을 시작 꼭지점으로 하는 장애물의 꼭지점 리스트가 된다. 장애물 내부에 구멍이 있는 경우에는 플러드 필(flood-fill) 알고리즘 등을 적용하여 내부를 채우고 시작한다. 순서도의 반복 구조는 다시 '시작' 타일에 이를 때까지의 순환 구조이며 위에서 설명한 조건에 부합하는 타일을 만나면 자동으로 진행 방향을 수정하면서 장애물의 경계를 구축하게 된다. 여러 장애물에 대한 처리 방법은 꼭지점 리스트가 구해진 장애물을 빈 타일로 처리하고 위 알고리즘을 다시 적용하면 된다. n개의 장애물이 존재하면 n개의 꼭지점 리스트가 얻어진다.

꼭지점 리스트를 구하는 목적은 가시성 정보를 휴리스틱에 반영하기 위해서이므로 위에서 설명한 기본적인 알고리즘에 고려해야 하는 사항이 두 가지 있다. 첫째는 가시성 정보를 그래프로 구축하는 데에는 볼록(convex) 꼭지점만 의미가 있기 때문에 오목(concave) 꼭지점을 제거한 볼록 꼭지점 리스트를 구하는 것이며 둘째는 그리드 기반에서는 사선의 표현이 계단식으로 나타내져 사선 표현 부분에 과도하게 나타나는 꼭지점들을 제거하는 것이다. 오목 꼭지점을 제거하는 것은 간단하다. 그림 1의 순서도에는 꼭지점을 추가하는 처리 상자가 2개가 있는데 시계방향으로 회전할 때 볼록 꼭지점이, 시계반대방향으로 회전할 때 오목 꼭지점이 생기는 것이므로 꼭지점

리스트에 추가할 때 이 정보를 이용하면 쉽게 분리해 낼 수 있다. 사선의 표현을 단순하게 하는 방법은 꼭지점을 찾을 때 이웃한 볼록 꼭지점 사이의 기울기를 구해 기울기가 같은 구간을 직선으로 처리하고 이 구간의 첫 점과 끝점만을 꼭지점으로 분류하는 것이다. 이 작업은 볼록 꼭지점을 찾을 때 동시에 체크할 수 있어 추가 시간을 요구하지 않는다.

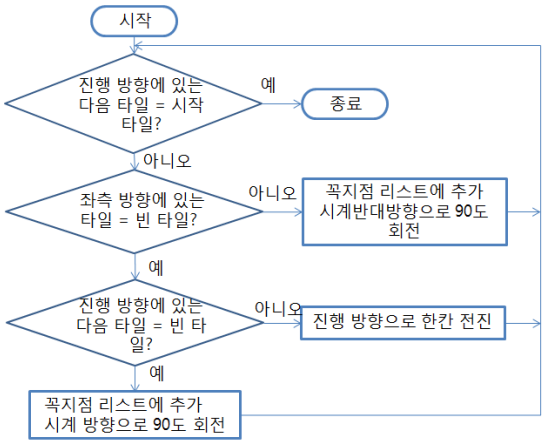


그림 1. 그리드 맵에서 장애물의 꼭지점 리스트 생성
Fig. 1. Generating a vertex list of the obstacles in the grid map

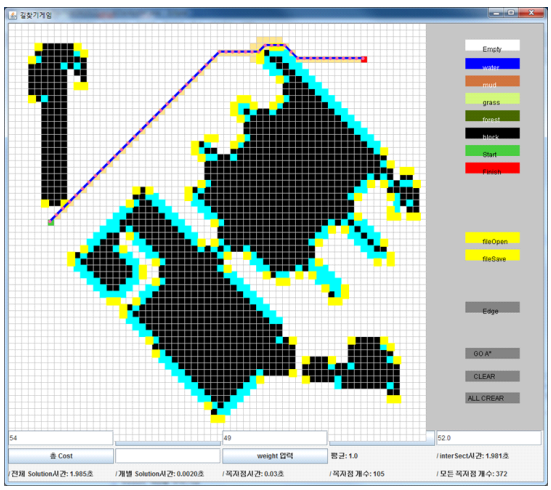


그림 2. 장애물의 꼭지점 표시
Fig. 2. Indicating the vertices of obstacles

그림 2의 예에서 검은색 타일이 장애물을 나타낼 때 구해진 꼭지점을 밝은 색으로 표시하였는데 이 예에서는 전체 꼭지점 수는 372개이며 사선 작업을 통해 105개로 감소하였다. 전처리의 두 번째 단계에서는 실시간 쿼리 단계에서 가시성 정보를 이용한 휴리스틱을 사용할 수 있게 하기 위해 가시성 정보를 모두 담고 있는 가시성 그래프를 구축해 놓는다. 자유 공간에서 경로를 찾기 위해 최단 경로 탐색이 보장되어 있는 가시성 그래프는 이동의 시작 위치와 목표 위치, 장애물의 꼭지점들을 그래프의 노드로 하고 노드와 노드를 연결하는 선분 중에 장애물과 교차하지 않는 선분을 그래프의 링크로 하여 형성된다. 최단 경로를 찾기 위한 로드맵으로 사용되는 가시성 그래프는 일반적으로 축소형 가시성 그래프이다. 축소형 가시성 그래프란 가시성 그래프의 링크 중에서 오목 꼭지점을 끝점(endpoint)으로 하는 링크를 제거한 그래프를 말하는데 오목 꼭지점을 포함하는 링크는 최단 경로를 구성하지 않기 때문에 최단경로를 찾기 위해서는 축소형 가시성 그래프를 이용하는 것이 충분하다고 알려져 있다[15]. 전처리 과정에서의 출력은 시작과 목표 위치를 제외한 장애물의 볼록 꼭지점들로만 구성된 가시성 그래프가 된다.

IV. 실시간 쿼리 단계: 가시성 휴리스틱을 이용한 경로 탐색

실시간 쿼리 단계는 시작 위치와 목표 위치가 주어지면 이들을 전처리 과정에서 구축된 가시성 그래프에 연결함으로서 시작된다. 이 과정부터는 경로 탐색 소요 시간에 포함되므로 실시간 효율성이 중요한데 실제로는 새로 추가되는 노드들과 기존의 노드 사이의 가시성(visibility)이 존재하는지 판단하기 위해 일일이 장애물과의 교차 여부를 체크해야 하는 시간 소모적인 과정이다. 이 과정을 효율적으로 구현하기 위하여 계산 기하학 분야의 회전 평면 스윕(rotational plane sweep) 알고리즘을 이용하여 소모적인 교차 확인 대신에 선행되는 계산에서 얻어진 정보를 다음 순서의 계산에서 이용하여 필요한 계산을 줄이도록 한다. 이렇게 시작과 목표 노드를 포함하는 가시성 그래프가 완성된다.

가시성 그래프에서 구한 최단 경로를 그리드 환경에서 바로 구현하기에는 적합지 않다. 그리드에서는 상하좌우와 대각선 움직임만 허용되므로 최단으로 구해진 링크들을 그리드 위에서 구현하기 불가능한 경우가 있을 뿐 아니라 그리드의 각 셀이 포함하는 지형 종류 등의 정보를 고려한 경로 탐색을 해야 하기 때문이다. 그러므로 본 연구에서는 가시성 그래프에 저장된 가시성 정보를 휴리스틱에 포함하여 활용하는 방법을 제안한다.

A* 알고리즘 탐색에서 가장 많이 사용되는 허용 가능한 휴리스틱은 현재 노드 n에서 목표 노드 n_g까지의 직선거리

$d(n,n_g)$이다. 현재 노드 n에서 보이는 노드(visible node)들을 $v_1,v_2,\cdots,v_m$이라고 하면 장애물을 피하는 경로는 보이는 노드중의 하나를 지나가야만 한다 [14]. 그러므로 제안하는 가시성 휴리스틱 $h^v(n)$는 식 (1)과 같이 정의한다.

$$h^v(n)=\min[d(n,v_1)+d(v_1,n_g), \quad d(n,v_2)+d(v_2,n_g),\cdots,d(n,v_m)+d(v_m,n_g)] \tag{1}$$

최단 경로는 장애물의 보이는 꼭지점(visible vertices) 중에 하나를 반드시 지나가며 [15] 식 (1)은 이 값들 중에 최소값을 취하므로 실제 비용에 비해 과소추정됨을 알 수 있으며 식 (2)에서와 같이 기존의 직선거리 휴리스틱에 비해 정보가 많은(more informed) 휴리스틱이다.

$$d(n,n_g) \le h^v(n) \le h^*(n) \tag{2}$$

가시성 정보는 가시 노드 중 하나에 도달할 때까지 다음 상태에 전달되어 모든 상태에서 가시성 체크를 하지 않는다. 가시 노드 중 하나에 도달하면 그 노드로부터 보이는 노드들은 이미 전처리 과정에서 저장해 놓았으므로 추가 시간이 소요되지 않는다. 이와 같이 제안하는 휴리스틱을 이용하여 수정된 A* 알고리즘은 그림 3에 의사 코드(pseudo codes)로 정리되어 있다.

modified_A*

```
{ V ← Get visible nodes v1,v2,... vn;
  g=0;
  h=min{d(start,v1)+d(v1,goal),...,  d(start,vn)+d(vn,goal)};
  OPEN ← {(start, V, g. h)};
  while (OPEN ≠∅) {
      x= leftmost state from OPEN;
      remove x from OPEN and put x on CLOSED;
      if (x==goal) then return(success);
      else if (x∈V)
        V ← Get visible nodes v1,v2,... vk;
      for each child ci of children_of(x) {
        if (ci∈CLOSED) then continue;
        gi=g+d(x,ci);
        hi ← min{d(ci,v1)+d(v1,goal),...,d(ci,vk)+d(vk,goal)};
        if (ci∈OPEN) and (gi<old_gi)
        then update(f_value of ci)
        else if (ci∉OPEN) add (ci, V, gi, hi) to OPEN;
        else continue; }
      reorder OPEN by g+h; }
  return(failure); }
```

그림 3. 가시성 휴리스틱을 이용한 수정된 A* 알고리즘
Fig. 3. Modified A* algorithm with Visibility heuristic

그림 4의 예에서 시작 셀(start)에서 보이는 노드들을 화살표로 나타내었다. 6개의 보이는 노드들 중에 최단 경로를 보장하는 가시 노드는 v_1이고 이에 도달할 때까지는 가시성 체크없이 휴리스틱 함수에 이용될 거리만 갱신한다. v_1에 도달하면 v_1에서 보이는 노드들은 전처리 과정에서 가시성 그래프에 이미 저장해 놓았으므로 단순히 가시성 정보를 가져오면 된다. v_1에서 보이는 노드들은 점선의 화살표로 표시하였다. 가시 노드들에 대해 위 과정을 반복하여 구해진 경로가 시작 셀에서 목표 셀(goal)까지의 점선으로 표시되어 있다.

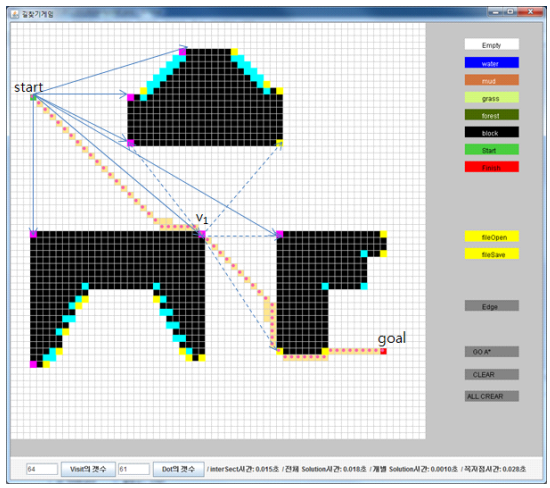
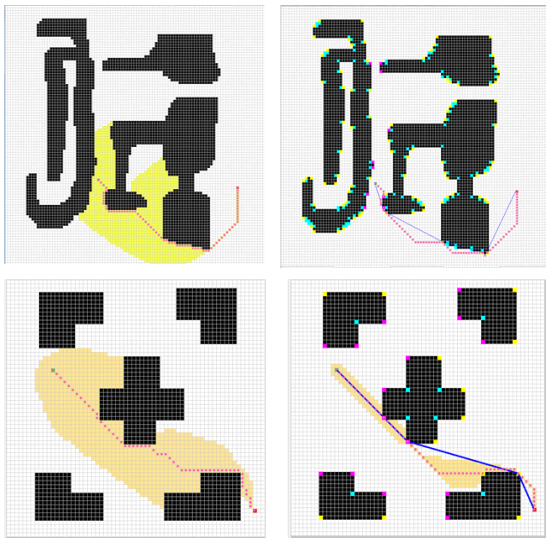


그림 4. 가시성 테스트와 경로 찾기
Fig. 4. Visibility tests and path-finding

V. 구현 및 시뮬레이션 결과

50x50~100x100 사이의 14종류의 그리드 기반 배경 화면을 대상으로 각각에 대해 1~3개의 다른 시작점과 목표점을 실험하여 평균 방문 노드수와 탐색 소요시간을 비교하였다. 사용한 배경 화면중의 일부는 그리드 기반 경로 찾기에서 성능 비교를 위해 벤치마크로 사용하도록 [8]에서 제안된 자료 중, 장애물이 다각형으로 이루어진 것을 64x64 그리드로 변환한 것이다.



(a) 직선거리 휴리스틱 이용 (b) 가시성 휴리스틱 이용

그림 5. 직선거리 휴리스틱 A* 탐색 결과와 가시성 휴리스틱을 이용한 경로 찾기 결과의 비교

Fig. 5. Comparison of results of path-finding using A* searches with Euclidean distance heuristic and Visibility heuristic

그림 5의 첫 번째 예제가 이들 예 중 하나이다. 그림 5(a)는 그리드 기반에서 직선거리 휴리스틱을 이용한 A* 탐색을 실행하여 경로를 찾은 결과이며 그림 5(b)는 꼭지점을 찾아 가시성 휴리스틱을 이용하여 A* 탐색을 실행한 결과이다. 최종 경로는 점으로 표시되어 있으며 노란색 부분이 방문 노드를 나타내는데 방문 노드를 나타내는 영역이 가시성 휴리스틱을 이용한 경우에 기존의 A*탐색의 경우에 비해 현저히 줄어듦을 알 수 있다. 이 예에서 실제 방문 노드 수는 각각 1026개에서 197개로, 825개에서 193개로 줄었다.

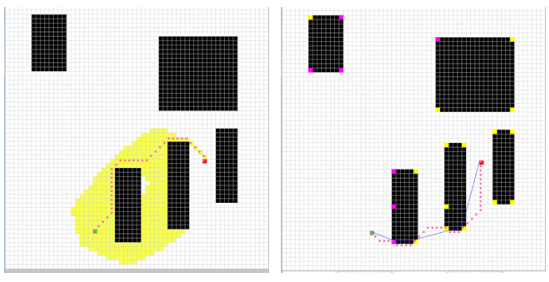


그림 6. 직선거리 휴리스틱과 가시성 휴리스틱을 이용한 경로 탐색 결과가 다른 예

Fig. 6. Example of different results of path-finding using Euclidean distance heuristic and Visibility heuristic

그림 6은 직선거리 휴리스틱을 이용하여 A* 탐색한 결과와 가시성 휴리스틱을 이용하여 A* 탐색한 결과가 다른 경우를 보여 준다. 두 경로의 길이는 39로 동일하며 실험 결과, 경로의 길이가 거의 같을 때 이런 현상이 종종 나타나는데 이는 기존의 그리드 탐색에서의 경로의 대칭성(path symmetry) 때문이다. 즉, 그리드에서는 자식노드를 생성하는 순서에 따라 경로가 다르게 구해지기 때문에 무수히 많은 경로가 가능한 것이다. 가시성 정보를 이용하면 이와 같은 불확실성이 제거되고 보이는 꼭지점 가운데 가까운 곳으로 먼저 이동하게 되므로 사람이 이동하는 것과 유사한 결과를 얻을 수 있다는 장점이 있다.

정보가 많은 휴리스틱을 이용함으로서 방문 노드수를 획기적으로 줄일 수 있으며 따라서 경로 탐색에 소요되는 시간도 줄일 있었다. 앞에서 소개한 다양한 배경에 대해 반복하여 30회를 테스트한 결과, 평균 방문 노드수와 탐색 소요시간은 표 1과 같았다. 따라서 방문 노드수는 기존의 방식에 비해 23%에 불과하였고 경로 탐색 소요시간은 3.4배 빨라졌으며 최종 경로의 길이는 두 가지의 경우가 유사하였다. 그리드의 복잡도에 따라 얻어지는 이득의 패턴을 보기 위해 애초 방문 노드수에 따라 분석하였다 (그림 7). 방문 노드수의 증가에 대해 완전한 비례 관계는 아니지만 대체적으로 경로 탐색이 간단한 경우에 비해 복잡한 경우에 방문 노드수와 탐색 시간이 더 향상됨을 확인할 수 있었다.

마지막으로 꼭지점을 찾고 가시성 정보를 저장하는 전처리 과정은 평균 707ms가 소요되어 탐색 시간에 비해 큰 단위이지만 실제 경로 탐색이 일어나기 전에 미리 처리하는 부분이기 때문에 탐색 성능에는 영향을 주지 않는다.

표 1. 직선거리 휴리스틱과 가시성 휴리스틱을 이용한 A* 탐색의 성능 비교

Table 1. Performance comparison of A* searches with Euclidean distance heuristic and Visibility heuristic

	직선거리휴리스틱	가시성 휴리스틱
방문 노드수	553	129
탐색 시간	12.7ms	3.7ms

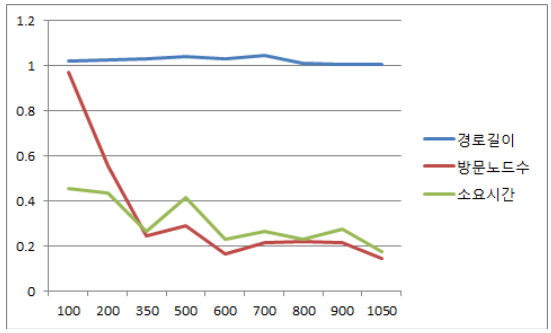


그림 7. 방문 노드수의 증가에 따른 탐색 성능 개선

Fig. 7. Improvement of search performance as the number of visited nodes increases

V. 결론

고해상도 그리드 맵에서 경로 찾기는 공간 소모적이고 비효율적인 과정이다. 본 논문에서는 효율적인 경로 찾기를 위해 그리드 맵을 구성하는 장애물들의 꼭지점을 찾아 순서 리스트로 표현하고 꼭지점에 대한 가시성 정보를 이용하는 휴리스틱을 제안하였다. 제안한 방법에 의해 전체적으로는 보이는 노드로 이동하면서 지형의 특성에 따라 세부 이동경로를 결정한다는 점에서 사람이 경로를 택하는 방법과 유사하게 구현되었다. 기존의 직선거리 휴리스틱보다 정보가 많은 휴리스틱의 사용으로 방문 노드수를 감소시킬 수 있었으며 이 결과는 시뮬레이션을 통해 확인하였는데 탐색하는 동안의 방문 노드수가 감소하여 탐색 시간의 향상 결과를 가져왔다. 배경이 단순한 경우보다 복잡한 경우의 성능 향상 효과가 더 뛰어남을 확인할 수 있었다.

시뮬레이션 결과에서 언급했듯이 본 연구는 경로 찾기 이전의 전처리 과정에 대한 효율성을 보장하지 못하는 단점이 있다. 비록 오프라인 계산이지만 효율적인 알고리즘을 고안하고 적용하여 전처리에 드는 시간을 단축하는 노력이 필요한데 두 부분에서 향상의 가능성이 있다. 우선 그리드 맵으로부터 다각형 맵으로 변환하는 부분에서는 결과로 얻은 꼭지점의 개수가 가시성 그래프의 생성이나 경로 찾기 모두에 영향을 주기 때문에, 연속적인 장애물의 경계를 그리드로 표현함으로서 불필요하게 나타나는 꼭지점들을 찾아 제거하는 것이 필요하다. 또 한 가지는 가시성 그래프의 생성 부분인데, 계산 기하학 분야에서는 O(n2) 시간 복잡도의 가시성 그래프 생성 알고리즘도 소개된 바 있으므로 [14] 이를 이용하여 효율적인 구현을 시도해 볼 수 있다. 이 부분이 효과적으로 구현되면 경로 탐색 실행시에 동적 장애물에 대한 처리도 가능해 질 것이다.

참고문헌

[1] A. Botea, M. Müller, and J. Schaeffer. Near Optimal Hierarchical Path-finding. In Journal of Game Development (Issue 1, Volume 1), 2004.

[2] Rabin, S. "A* speed optimizations and A* Aesthetic Optimizations," In: Deloura, M. (eds.): Game Programming Gems. Charles River Media, 264-287, 2000.

[3] A. V. Goldberg and C. Harrelson. Computing The Shortest Path: A* Search Meets Graph Theory. In SIAM Symposium on Discrete Algorithms (SODA), 2005.

[4] N.R. Sturtevant,, A. Felner, M, Barrer, J. Schaeer, and N. Burch. Memory-Based Heuristics for Explicit State Spaces. SOCS, In Proceedings of the Twenty-First International Joint Conference on Artificial Intelligence (IJCAI-09), 609-614, 2009.

[5] S. Thrun and A. Bucken. Integrating Grid-Based and Topological Maps for Mobile Robot Navigation. In Proceedings of the 13th National Conference on Artificial Intelligence, 1996.

[6] Pinter, M. "Towards more realistic pathfinding," Game Developer Magazine April, 2001.

[7] X. Cui and H. Shi. A*-based pathfinding in modern computer games. International Journal of Computer Science and Network Security, 11(1):125-130, 2011.

[8] N.R. Sturtevant, Benchmarks for Grid-Based Pathfinding, IEEE Transactions on Computational Intelligence and AI in Games, Volume:4(2), pp 144 - 148, 2012.

[9] N.R. Sturtevant, Memory-efficient abstractions for pathfinding. In Proceedings of the third conference on Artificial Intelligence and Interactive Digital Entertainment, pp. 31-36, 2007.

[10] Y. Bjornsson and K. Halldorsson. Improved heuristics for optimal path-nding on game maps. AIIDE, pp9-14, 2006.

[11] M. Goldenberg, A. Felner, N.R. Sturtevant, and J. Schaeer. Portal-Based True-Distance Heuristics for Path Finding. SOCS, In Symposium on Combinatorial Search, 39-45, 2010.

[12] D. Harabor and A. Botea. Breaking Path Symmetries in 4-connected Grid Maps. In AAAI Conference on Artificial Intelligence and

Interactive Digital Entertainment (AIIDE), 2010.

[13] D. Harabor and A. Grastien. Online Graph Pruning for Pathfinding on Grid Maps. In National Conference on Artificial Intelligence (AAAI), 2011.

[14] H. Kim, K. Yu and J. Kim, "Reducing the Search Space for Pathfinding in Navigation Meshes by Using Visibility Tests", Journal of Electrical Engineering & Technology, Vol. 6, No. 6, pp. 867~873, 2011.

[15] J.C. Latombe, "Robot Motion Planning", KAP, pp. 310-317, 1991.

저 자 소 개

김 지 혜

2014: 덕성여자대학교 컴퓨터학과 공학사.

관심분야: 게임 인공지능

Email : jihyuikim@naver.com

정 예 원

2014: 덕성여자대학교 컴퓨터학과 재학중

관심분야: 게임 인공지능

Email : jyewon92@hanmail.net

유 견 아

1986: 서울대학교 제어계측공학과 공학사.

1988: 서울대학교 제어계측공학과 공학 석사.

1995: Univ. of Southern California 컴퓨터학과 공학박사

현 재: 덕성여자대학교 컴퓨터학과 교수

관심분야: 인공지능, 경로계획 알고리즘

Email : kyeonah@duksung.ac.kr