지역 우선 탐색 기법을 이용한 속도 지향 Terrain Following 알고리즘

김 종혁[0]* 고 명철* 최 윤철* 고 견**

*연세대학교 컴퓨터과학과, **청주대학교 컴퓨터정보학과

{iwannado, zoo, ycchoy, kyunkoh}@rainbow.yonsei.ac.kr

A Local Search Algorithm For Fast Terrain Following

Jong-Hyuk Kim[0]* Myeong-Cheol Ko* Yoon-Chul Choy* Kyun Koh**

*Dep. of Computer Science, Yonsei University

**Dep. of Computer and Information Engineering, Chongju Univ.

요 약

Terrain Following 이란 가상환경 내에서 지형의 표면을 이동할 때 지형의 모양에 따라 자연스럽게 지표 위를 이동할 수 있게 하는 기술로서, 이것을 위해서는 지표면상에서 움직이는 물체가 갖는 위치와 높이 정보를 매시간 알아내야 한다. 전통적인 접근 방법은 지표면의 높이(z)가 일정하다고 가정하거나, 일반적인 충돌 감지 알고리즘을 이용하는 것이다. 그러나 충돌 감지 알고리즘은 복잡하게 모델링 된 물체들간의 충돌을 감지하기 위해 고안된 알고리즘으로서 비교적 단순한 Terrain Following 에 적용할 경우 불필요한 연산을 많이 하게 되어 오히려 시간적인 손실을 초래할 수 있다. 또한 Quadtree 를 지형의 표현을 위한 기본 자료구조로 이용하는 연구도 진행되고 있지만, Quadtree 는 기본적으로 래스터 데이터의 표현에 적합하며 지형과 같은 벡터 데이터의 표현에는 적합하지 않다. 본 연구에서는 네비게이션시 발생하는 인접성에 기반한 지역 탐색 방법을 사용하여 실시간 계산을 좀 더 빠르게 처리할 수 있는 알고리즘과 자료구조를 제시한다. 이 방법을 사용하면 탐색 과정에서 발생하는 비교 횟수를 크게 줄일 수 있어 실시간 네비게이션시의 탐색 성능을 향상시킬 수 있다.

1. 서론

Terrain Following 혹은 Terrain Avoidance란 네비게이터나 오브젝트가 가상환경 내에서 지형의 표면을 이동할 때 지형의 모양에 따라 자연스럽게 지표 위를 이동할 수 있게 하는 관련 기술을 말한다. 이것을 위해서는 현재 움직이는 물체가 지형의 어느 부분에 위치하는지를 알아야 한다. 즉, 현재 네비게이터가 지형의 어느 위치(x, y)에 있으며, 그 곳 지형의 높이(z)가 얼마인지를 네비게이터의 이동에 따라 매시간 계산에 의해 알아내야 한다.

Terrain Following을 함으로써 얻을 수 있는 장점으로는 현실감을 꼽을 수 있다. 현실과 가까운 지형 위를 움직일 때 참여자는 더욱 임장감과 몰입감을 느낄 수 있으며 따라서 참여자가 느끼는 현실감을 보다 높일 수 있는 것이다[1].

Terrain Following은 군사 시뮬레이션이나 원격 탐사와 같은 시뮬레이션 분야에서 널리 쓰일 수 있다. 또한, Terrain Following 이외의 분야에서도 활용될 수 있다. 가령 지형 데이터 내에서 접근 불가능한 지역(호수, 벽)에 대한 처리에도 이용할 수 있다. 미리 접근 불가능한 지역을 명시하여 참여자의 접근을 막음으로써 별도의 다른 방법을 사용하지 않더라도 제어를 할 수 있다. 구역을 나누어 여러 개의 서버가 관리하는 분산 가상환경 등에서도 이용될 수 있는데, 참여자가 어느 구역에 속해있는지에 대한 정보를 Terrain Following 알고리즘을 이용하여 감지할 수 있다.

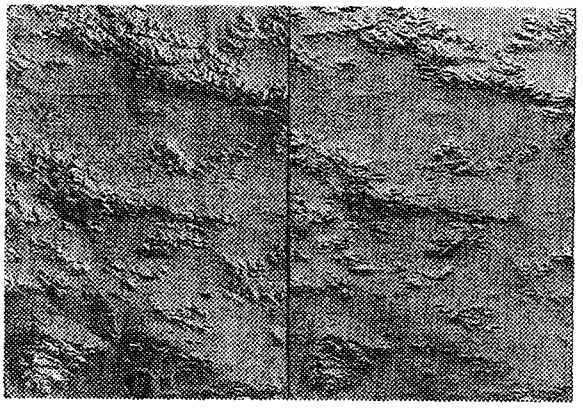
Terrain Following을 하기 위한 전통적인 접근 방법은 충돌 감지(Collision Detection) 알고리즘을 이용하는 것이다. 그러나 일반적인 충돌 감지 알고리즘을 Terrain Following에 적용시키는 방법은 계산에 드는 시간 비용이 상대적으로 크다. 충돌 감지 알고리즘은 복잡하게 모델링 된 물체들간의 충돌을 감지하기 위해 고안된 알고리즘으로 비교적 단순한 Terrain Following에 이용할 경우 불필요한 연산을 많이 하게 되어 시간적 손실을 초래할 수 있다[1]. 이러한 문제들을 해결하기 위해 QOTA(Quick Oriented Terrain Algorithm) 에서는 Quadtree를 지형의 표현을 위한 기본 자료구조로서 이용하고 있지만, Quadtree는 기본적으로 래스터 데이터의 표현에 적합한 자료구조이기 때문에 지형과 같은 벡터 데이터의 표현에는 적합하지 않다. 또한, 지형 데이터의 skew 현상이 클 때 Quadtree 방법은 저장구조나 탐색 속도적인 측면에 있어 비효율적인 면을 가지고 있다. 이러한 이유로 대부분의 가상환경 및 시뮬레이션 분야에서는 높이(z)가 일정한 지형을 사용함으로써 문제들을 회피하고 있는 실정이다.

본 연구에서는 네비게이션시 발생하는 인접성에 기반한 지역 탐색 방법(Local Search)을 사용하여 실시간 계산을 좀더 빠르게 처리할 수 있는 알고리즘과 자료구조를 제시한다.

2. 관련연구

2.1 지형의 표현: Regular grid 와 Irregular grid

Terrain Following을 하기 위해서는 크게 세 가지 기술이 요구된다. 첫 번째로 지형의 표면을 근사 시키기 위한 방법을 들 수 있다. 이는 지형의 표면을 데이터화 하는 것으로 수학적 방법과 이미지 방법으로 나뉜다. 수학적 방법으로는 multiquadratic polynomials 방법을 사용하는 Global-Fourier series와 irregular patches 방법을 사용하는 Local-Regular patches를 가장 많이 사용한다. 수학적 방법의 장점으로는 지형의 표면을 나타내는데 우수한 성능을 보이지만, 현실적으로 모든 지표면을 간략하게 표현할 수 있는 수학식은 존재하지 않는다는 단점을 갖고 있다. 이미지 방법은 크게 점(vertex) 데이터를 이용하는 방법과 선(line) 데이터를 이용하는 방법으로 나뉜다. 점 데이터를 이용하는 방법은 지형의 표면을 나타낼 때 점을 이용하여 표현하는 것으로 regular grid(DEM), irregular grid (TIN)등의 방식이 존재한다[그림 1]. 일반적으로 이 방법들을 가장 많이 사용하고 있는 실정이다. 선 데이터를 이용하는 방법은 지형의 표면을 나타낼 때 선을 이용하여 표현하는 것으로 horizontal slices (contours), vertical slices(profiles), Critical Lines(ridges, stream, slope)등을 많이 사용하고 있다.



DEM (4 million polygons) TIN (90,000 polygons)

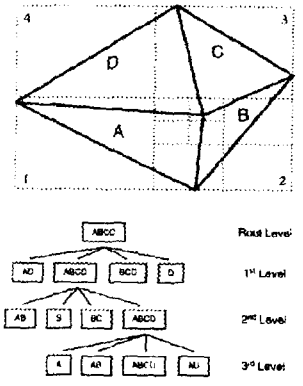
[그림 1] DEM과 TIN의 비교

2.2 지형 데이터를 위한 자료구조

Terrain Following을 하기 위해서 요구되는 두 번째 기술은 지형 데이터를 표현하기 위한 자료 구조로 현재 이용되고 있는 기술은 Quadtree를 이용하여 구역을 분할하는 방식을 이용하고 있다. 마지막으로 빠른 Terrain Following을 위한 알고리즘을 들 수 있는데 현재까지는 Terrain Following을 위해 특화된 알고리즘은 존재하지 않는 실정이다.

2.3 Terrain Following 알고리즘

현재까지 알려진 Terrain Following을 위한 알고리즘은 대부분 Quadtree를 지형의 표현을 위한 기본 자료구조로 이용하고 있다[그림 2]. 이 방법은 네비게이터가 이동하는 위치가 어느 정도 시간적인 locality를 갖는 특징을 이용하는 방식으로, 비교적 적절한 자료구조로서 평가될 수도 있지만, 기본적으로 래스터 데이터의 표현에 적절한 Quadtree를 벡터 데이터로 되어있는 지형의 표현을 위해 사용하기 때문에 그에 따른 문제들을 유발한다. 또한, 지형 데이터의 skew 현상이 클 경우 Quadtree 방법은 저장구조나 탐색 속도 적인 측면에 있어 비효율적인 면을 가지고 있다. 아울러 알고리즘적인 측면에 있어서도 Terrain Following 방식에 특화된 알고리즘을 사용하는 것이 아닌 자료구조에 따른 일반적인 탐색 방법을 그대로 사용하고 있다. 일 예로 Quadtree를 이용한 방법에서는 항상 root로부터 탐색하는 방식을 사용한다. 그러나 Terrain Following의 특성상 주변에 위치할 확률이 크기 때문에 주변을 먼저 찾아보는 지역 탐색(Local Search)기법을 사용하면 좀 더 빠른 탐색을 할 수 있다.

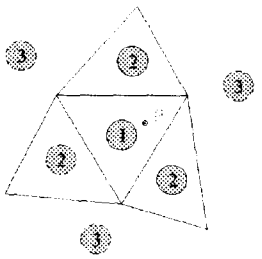


[그림 2] Quadtree 자료 구조

3. 지역 탐색 (Local Search) 알고리즘

3.1 탐색 알고리즘

지형 데이터는 방대한 자료를 가진다는 점과 갱신이 빈번하게 발생하지 않는다는 특성을 가지고 있다. 또한, Terrain Following은 실시간으로 발생한다는 특징이 있기 때문에 기존에 존재하는 알고리즘들을 그대로 이용하는 것으로는 효율성을 높일 수 없다. Terrain Following 시 특징을 살펴보면 현재 위치에서 다음 위치를 탐색할 경우 현재 위치 주변에 있을 확률이 크다는 것을 알 수 있다[그림 3]. 따라서 지역성에 기반한 지역 탐색 기법을 이용하면 좀더 빠른 시간에 탐색을 할 수 있다. 이러한 특성을 이용한 지역 탐색 알고리즘을 요약하면 다음과 같다[표 1].



[그림 3] Terrain Following 시 현재 위치 P 에서 다음 위치를 찾을 확률

[표 1] 지역 탐색 알고리즘

가상환경 내의 시작점에서 탐색 시작
1. 처음 시작되는 polygon 탐색
  2-1. 우선적으로 자신의 polygon 에서 탐색
    (1 번 구역 [그림 3] 탐색)
  2-2. 만약 존재하지 않으면, 다음 순위에 있는 3 개의 polygon 에서 탐색
    (2 번 구역 [그림 3] 탐색)
  2-3. 만약 존재하지 않으면, 마지막으로 나머지 연결된 polygon 에서 탐색
    (3 번 구역 [그림 3] 탐색)
3. 실제 높이(Z) 정보 추출
    Z = -(a/c)X-(b/c)Y-(d/c)
    (X, Y 는 현재 위치의 (x, y)좌표)
4. 2와 3의 과정을 반복하며 네비게이션

3.2 자료구조

지역 탐색을 위한 자료구조는 크게 Polygons 와 Points로 나뉜다. Polygons에서는 실제 point들에 대한 ID와 polygon 내부에 있는지 계산을 빠르게 하게 위한 Bounding Box, 인접하는 polygon들의 ID를 저장하고 있다[표 2]. Points 에서는 각각의 point 들의 ID에 따른 실제 (x, y, z)좌표를 저장하고 있다[표 3].

[표 2] Polygons 자료구조

ID	Points	Bounding Box	Neighbors	Etc
	1, 2, 3	(Ax1, Ay1) (Ax2, Ay2)	B, C, D, E, F, I	
	1, 3, 7	(Bx1, By1) (Bx2, By2)	A, C, D, F	
	2, 4, 8	(Cx1, Cy1) (Cx2, Cy2)	A, B, D, F, G, I, J	

[표 3] Points 자료구조

ID	(x, y, z)
	(1.0, 2.2, 3.5)
	(2.1, 4.4, 7.2)
	(2.8, 4.8, 8.1)

4. 실험 및 성능평가

구현 환경은 다음과 같다. 지형 데이터로는 다양한 polygon 수를 갖는 TIN 데이터를 사용하였고, H/W 는 pentium II 400, RAM 128M 환경 하에서, Windows 2000, Visual C++ 6.0, OpenGL을 사용하여 구현하였다.

[표 4]는 본 논문에서의 지역 탐색 알고리즘을 기존의 Performer 라이브러리 및 QOTA[1]와 비교한 것이다. 표에서 볼 수 있듯이 제안하는 알고리즘의 탐색 시간은 polygon의 개수와는 직접적인 관련이 없음을 알 수 있으며, 이는 네비게이션시 sampling rate의 영향을 받는 것으로 추측된다. 따라서 polygon의 개수가 증가할수록 Performer나 QOTA 의 탐색 시간이 증가하는 반면 지역 탐색 방법에서는 커다란 차이가 없어 상대적으로 더 큰 성능향상을 기대할 수 있다. 저장 공간 측면에서는 기존의 방식에 비해 약 13%정도 저장 공간이 증가하는 결과를 보여주었다.

[표 4] Performer, QOTA 및 지역 탐색 알고리즘의 성능비교

	Performer	QOTA	지역 탐색 알고리즘		
	13,346	13,346	13,346	3,996	972
	1881	19.2	1.29	1.19	1.3

5. 결론 및 향후 연구 방향

지역 탐색 알고리즘의 특징은 기존의 방식에 비해 저장 공간을 많이 사용하는 반면, 탐색 과정에서 발생하는 비교 횟수를 크게 줄일 수 있어 실시간 네비게이션 시의 탐색 성능을 향상시킬 수 있다. 또한 가상환경의 지형 데이터의 크기에 상관없이 항상 일정한 탐색 시간을 보여주기 때문에 가상환경의 지형 데이터가 커짐에 따라 기존에 방식에 비해 상대적으로 더욱 뛰어난 성능을 보여준다.

그러나 기존의 지역 탐색 알고리즘에서는 sampling rate에 따른 polygon skipping 현상이 발생할 수 있다. 따라서 이 문제를 해결하기 위한 개선된 알고리즘이 요구된다. 또한, 앞서 언급했듯이 지역 탐색 알고리즘에서는 기존의 방식에 비해 저장 공간을 많이 사용하기 때문에, 탐색 시간에는 영향을 미치지 않으면서 저장 공간을 절약할 수 있는 방법에 대한 연구가 필요하다.

6. 참고 문헌

[1] John W. Barrus and Richard C. Waters, QOTA: A Fast, Multi -Purpose Algorithm For Terrain Following in Virtual Environments, Proc . of the 2nd Symposium on Virtual Reality Modeling Language, 1997

[2] Kwang-sung Shin, In-mook Lee, A terrain following simulation considering the dynamic of 2-wheel vehicle, 1998

[3] Franco P.Preparata, Michael Ian Shamos, COMPUTATIONAL GEOMETRY - AN INTRODUCTION, 1985, p179 ~ 218

[4] Diamond Park, Available at http://www.merl.com/projects/dp/