한국컴퓨터정보학회 하계학술대회 논문집 제26권 제2호 (2018. 7)

레이싱 게임에서 빠른 도시 맵 생성을 위한 지형 패턴 합성 기법

김종현O
O강남대학교 소프트웨어응용학부
e-mail: jonghyunkim@kangnam.ac.krO

Terrain Pattern Synthesis Method for Quick City Map Generation in Racing Game

Jong-Hyun KimO
ODept. of Software Application, Kangnam University

● 요 약 ●

본 논문에서는 레이싱 게임에서 도시 맵을 빠르게 생성 및 합성할 수 있는 알고리즘을 제안한다. 레이싱 분류의 게임은 고정된 영역이 아닌 넓은 영역을 이동하는 특징이 있지만 이를 위한 맵을 디자인하고 개발하는 것은 많은 시간을 요구한다. 이 문제는 큰 지형 맵을 전처리 과정에서 생성함으로써 완화 시킬 수 있지만 고정적인 맵은 게임을 지루하게 만들며 그렇다고 다양한 맵을 모두 전처리 과정에서 처리하는 것은 비효율적이다. 이 같은 문제를 해결하기 위해 우리는 전처리 과정에서 지형 패턴을 만들어 실시간에 맵을 생성하고 다양한 맵을 랜덤하게 합성 할 수 있는 기법을 제안한다. 또한, View-dependent기법을 제안하는 프레임워크를 통합시켜 불필요한 렌더링 계산을 줄임으로써 효율성을 증대시킨다. 본 연구는 Unity 게임 엔진에서 개발했으며 레이싱 게임뿐만 아니라 다양한 콘텐츠에서 활용할 수 있다.

키워드: 지형 패턴 합성(Terrain pattern synthesis), 도시 맵 생성(City map generation), 레이싱 게임(Racing game)

I. Introduction

최근에 인터랙션의 중요성이 커짐에 따라 사용자의 움직임을 고려한 콘텐츠 개발이 증강/가상현실, 모바일 앱, 게임 등에서 활발하게 이루어지고 있다[1,2]. 대부분이 이동의 자유도를 높이고자 넓은 지형을 원하지만 고정적인 맵으로 인해 콘텐츠를 지루하게 만든다. 전처리 과정에서 일시적으로 맵을 다양하게 만들 수 있지만 리소스나 계산시간 측면에서 비효율적이다. 지형이나 텍스쳐를 만들기 위한 다양한 합성기법들이 존재하지만 게임에 적용하기에는 계산비용이 매우 클뿐 아니라[3], 이 접근법은 도로와 지형 내 존재하는 빌딩까지 합성하기에는 충분하지 않다. 제안하는 방법은 자동차가 넓은 지형을 움직일 수 있도록 자동으로 교차로와 도로뿐만 아니라 주변 빌딩까지 끊임없이 생성/합성하여 고해상도 도시 맵을 생성한다. 이를 수행하기 위한 본 연구의 장점은 아래와 같다.

- 시작 시 랜덤으로 일정 크기의 도로가 생성
- 사용자가 움직이는 방향으로 도로를 생성
- 모든 도로 방향과 건물들이 랜덤으로 배치
- 모든 길이 막히지 않고 연결될 수 있도록 합성
- 시점과 다른 방향의 오브젝트 렌더링을 비활성화

II. The Proposed Scheme

제안하는 방법은 도시 맵 합성을 효율적으로 계산하기 위해 전처리 과정에서 도로, 교차로, 빌딩 등 4가지 종류의 프리팹(Prefab) 생성을 시작으로 해서(Fig. 1 참조), 아래와 같은 순서로 실행된다.

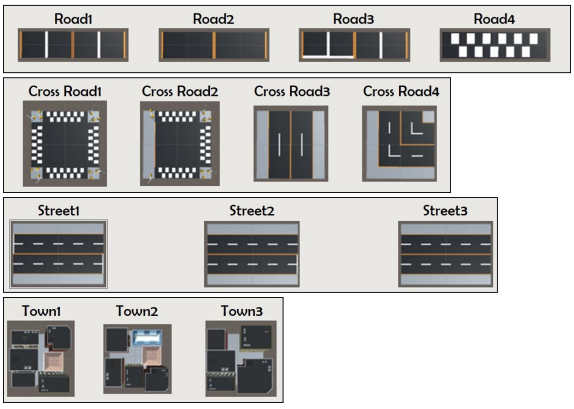

Fig. 1. Various prefabs in Unity.

도로 랜덤 생성 : 도로를 만들기 위해 2가지 배열을 사용한다. 첫 번째는 같은 자리에 또 다시 오브젝트를 생성하는 일이 없도록 제한하기 위한 오브젝트 배열이다. 두 번째 배열은 생성된 오브젝트의

진행 방향을 저장할 배열이다. 도로를 생성할 때는 3개의 상태 변수를 사용하였으며(벽, 횡당 보도, 직진 도로), 우리는 교차로를 먼저 생성한 후 교차로 사이를 도로와 건물로 이어주었다.

사용자의 이동 방향으로 도로 생성 : 사용자의 이동 방향으로 도로를 생성하기 위해 자동차의 위치를 중심으로 만든 경계박스를 이용하여 트리거(Trigger) 상태가 아닌 경우 해당 도로를 삭제한다. 삭제 후 자동차의 이동 방향에 따라 도로 오브젝트를 합성한다.

사용자 시점 기반 렌더링 : View-dependent 렌더링 기법과 통합하기 위해 우리는 메인 카메라가 바라보는 방향에 포함되어 있지 않은 오브젝트 렌더링을 끄는 방법을 사용한다.

III. Conclusions

본 연구에서는 Unity를 이용하여 개발하였으며 자동차의 위치를 기준으로 도시 맵을 지역적으로 생성/합성하는 결과를 만들어 냈다. 결과의 우수성을 보여주기 위해 우리는 전처리 과정에서 맵을 만들어서 사용하는 기법과 결과를 비교하였다.

전처리 과정에서 큰 도시 맵을 만들어 놓고 레이싱 게임을 시작했으며(Fig. 2 참조), 렌더링 뷰에서 보듯이 도로와 빌딩이 자연스럽게 연결되었지만 큰 도시 맵을 매 프레임 처리해야 되기 때문에 게임 실행이 느려 실시간 처리가 힘들다(영상 참조).

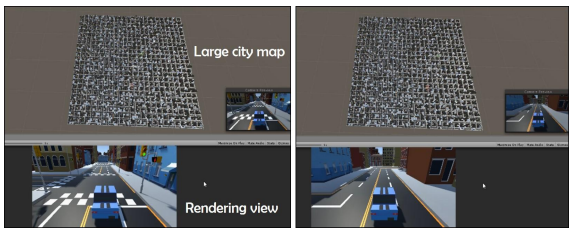

Fig. 2. Large city map calculated during preprocessing.

Fig. 3은 자동차 움직임에 따라 도로를 랜덤하게 생성하고 경계부분을 또 다른 연결 도로 자연스럽게 합성한 결과이다. Fig. 2와는 다르게 지역적으로 도시 맵을 생성했음에도 불구하고 끊임없이 연결된 도로 맵을 만들었으며, Fig. 2에 비해 게임 속도가 10배 이상 빨라졌다. Fig. 4는 View-dependent 렌더링 기술을 추가로 통합한 결과이며, 동일한 결과를 제작하는데 있어 Fig. 3보다 계산을 해야 하는 도시 맵 영역을 월등히 감소시켰다.

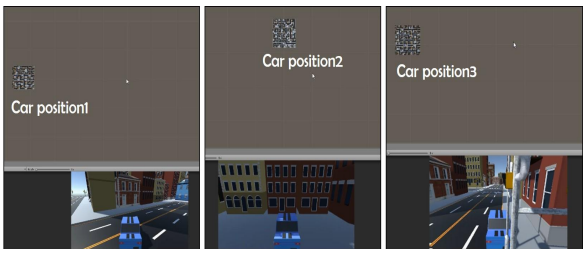

Fig. 3. City map generation with our method.

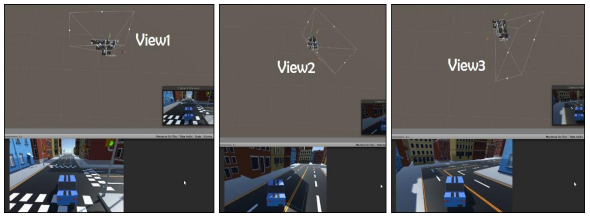

Fig. 4. City map generation with our method and view-dependent method.

REFERENCES

[1] Cakmak, Tuncay and Hager, Holger, “Cyberith Virtualizer: A Locomotion Device for Virtual Reality”, ACM SIGGRAPH, pp.6:1-6:1, 2014.

[2] Sutherland, Ivan E., “A Head-mounted Three Dimensional Display”, Proceedings of the December 9-11, 1968, Fall Joint Computer Conference, Part I, pp.757-764, 1968.

[3] Lefebvre, Sylvain and Hoppe, Hugues, “Parallel Controllable Texture Synthesis”, ACM SIGGRAPH, pp.777-786, 2005.