CSED451 Computer Graphics

Assignment #3

3D Drawing

알파고알파고

20140908 컴퓨터공학과 박경원(pkw1012)

20140735 컴퓨터공학과 백우정(pcsewj14)

1. 프로그램 개요

이번 과제는 Assignment 1과 2 에서 구현했던 것을 3D로 새롭게 구현하는 것이었다. 주로 3D wire-frame과 3D camera control을 집중적으로 다루었다.

wire-frame이란 말 그대로 선으로 object의 골격을 나타낸 것이다. object 파일을 읽어 vertex간의 관계를 파악해야만 선으로 제대로된 형태를 구현할 수 있다.

camera control의 경우, shader에서 uniform으로 적용되는 matrix에 대해 이해하는 것이 중요하다.

1. 프로그래밍 환경

Visual Studio 2015를 이용하였다.

1. 프로그램 설계 및 구현
2. Type 정의

이번 assignment에서는 씬 그래프를 이용해야했기 때문에 한 cpp파일에 모든 것을 구현했던 assignment 2번과 같이 class를 세부적으로 나누어 type을 구현하였다.

구현한 class는 다음과 같다.

class GAME : 게임 전체를 관리

class MAP : 게임의 맵, 즉 들판, 도로, 강을 다룬다.

class ROAD : 도로 한 줄과, 그 위의 자동차를 다룬다.

class RIVER : 강 한 줄과, 그 위의 통나무를 다룬다.

class FIELD : 들판 한 줄과, 그 위의 나무를 다룬다.

class CAR : 자동차에 대한 함수를 구현했다.

class LOG : 통나무에 대한 함수를 구현했다.

class TREE : 나무에 대한 함수를 구현했다.

class PLAYER : 사용자가 조작하는 캐릭터를 관리한다.

1. 함수
2. vector<string> tokenize(string str, string delim)

class 내부 함수는 아니며, parsing을 위한 보조함수이다.

참고: https://github.com/IlhwanHwang/CSED451\_assn2/blob/master/assn2/utility.cpp

1. void load()

object 파일을 로드해야하는 class CAR와 class TREE에 존재한다.

참고: https://github.com/IlhwanHwang/CSED451\_assn2/blob/master/assn2/model.cpp

1. void init(\_)

class GAME, MAP 등 모든 class에 존재하며

object의 처음 좌표를 지정하고, 이를 shader program에 넘겨준다.

1. void buffer()

vertex array에 각 object에 대한 buffer를 만들어 준다.

1. void draw(GLint pro, GLint model)

object를 출력한다. 필요할 경우 model\_veiw matrix를 변형시켜 object를 움직인다.

1. void move()

animation이 들어가는 object의 경우, 이 함수를 사용하여 변화량을 바꾼다. 좌표값을 바꾸는 것이 아니라, △x, 혹은 △y 값을 바꾸었다. class CAR, class LOG에서 사용하였으며 이를 위 쪽 class에서 호출한다.

1. void reInit()

자동차나 통나무가 무한히 나오는 것처럼 보이게 하기 위해서 일정 지점에 도달하면 좌표를 큰 폭으로 수정해야 했다. 마치 원을 무한히 도는 것과 같은데, 그것이 이 함수에 구현되어있다. move()에서 호출된다.

아래는 class GAME에서만 구현된 함수이다.

1. bool finished()

class GAME에 구현되어 있다. 게임 종결 조건을 검사하는 함수이다. 캐릭터가 화면 밖으로 나갔을 때, 자동차와 충돌했을 때, 혹은 강에 빠졌을 때 true를 반환한다. specialkeyboard와 GAME의 move에서 호출된다.

1. bool treehere()

캐릭터가 이동하려는 곳에 나무가 있는지 검사한다. specialkeyboard에서 호출된다.

다음은 class GAME의 object가 선언되어있는 assn2.cpp의 함수들이다.

1. void specialkeyboard(int key, int x, int y)

방향키 입력을 받는 함수이다. 캐릭터 조종은 방향키를 이용하도록 구현하였다. 이동할 때마다 finished를 이용해 종결 조건인지, treehere을 이용해 나무가 있는지 검사한다.

1. void phase1(int value)

void phase2(int value)

void phase3(int value)

캐릭터가 오른쪽으로 움직일 때, 걸어가는 애니메이션을 추가했는데, 걸어가는 동작을 만들어주는 함수들이다.

1. int main(int argc, char\*\* argv)

생성된 game object를 이용하여 게임을 실행시키는 함수이다.

다음은 class PLAYER에서 구현된 함수이다.

1. void quad(inti dx, int a, int b, int c, int d)

void colorcube(int index)

플레이어 캐릭터의 몸은 몸통, 머리, 팔, 다리 등의 계층 구조로 되어 있으며 이 각각의 구조를 상자 모양으로 만들었는데, 이 모양을 만들기 위한 함수이다.

1. void traverse(Node\* node)

캐릭터를 출력할 때, 트리 형태의 구조를 traverse하여 출력해야하며, 이 때 사용하는 함수이다.

1. void initNodes( )

계층 구조에서 몸통, 머리 등의 각각의 node들의 크기나 형태들을 초기화시켜주는 함수이다.

1. 추가 구현

해당 사항 없음

1. 프로그램 실행 방법

ctrl + F5를 누르면 된다. 이 때 헤더파일 Angel.h, vec.h, mat.h와 shader program을 불러와주는 InitShader.cpp가 반드시 있어야한다. 이 파일들은

<http://www.cs.unm.edu/~angel/BOOK/INTERACTIVE_COMPUTER_GRAPHICS/SIXTH_EDITION/CODE/include/> 에서 얻을 수 있었다.

또한 vertex shader와 fragment shader 역시 있어야 한다.

실행이 되면 방향키를 이용해 캐릭터를 움직을 수 있다.

‘왼쪽’, ‘오른쪽’의 방향키로는 캐릭터의 몸체를 돌리고,

‘위쪽’의 방향키로 캐릭터를 전진시킨다.

시점의 변화는 숫자키로 할 수 있다.

‘1’은 위에서 바라보는 시점이고, ‘2’는 캐릭터의 뒤쪽에서 바라보는 시점,

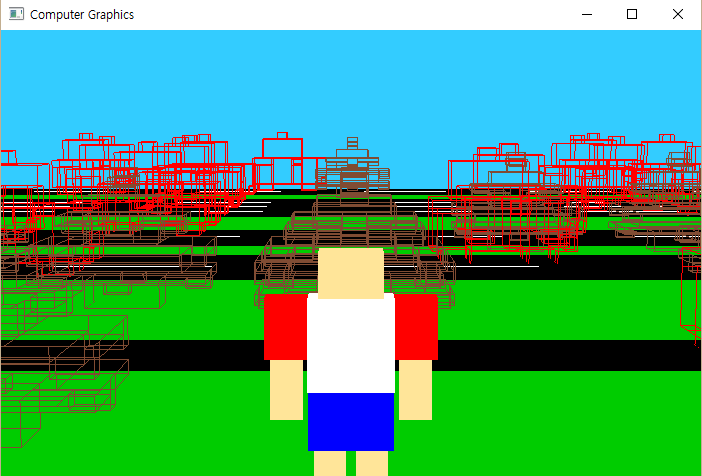
그리고 ‘3’은 캐릭터의 눈으로 바라보는 시점이다.

1. 예제

다음은 프로그램을 처음 실행시켰을 때 나오는 화면이다.



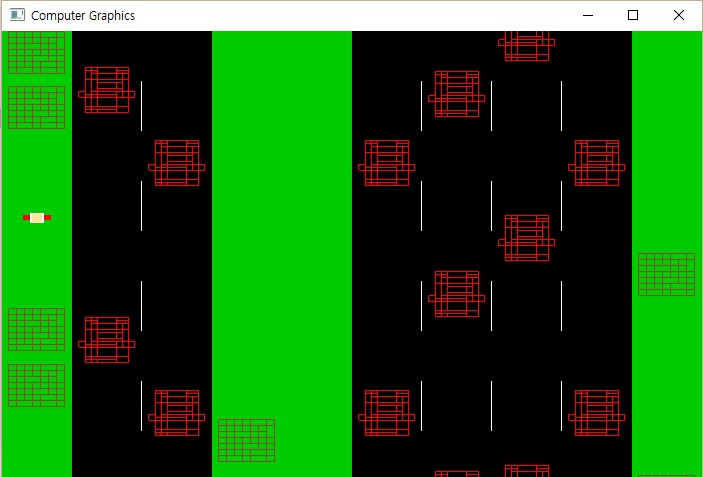
숫자 1~3을 통하여 시점을 변경시킬 수 있으며, 위의 화면은 시점 1로 위에서 내려다본 시점이다. 다음은 시점 2로 캐릭터의 뒤에서 바라보는 시점이다.



다음은 시점 3으로 캐릭터의 눈에서 바라보는 시점이다.



캐릭터는 방향키를 이동하여 움직일 수 있다. 좌우 방향키는 각각 시계반대방향, 시계방향으로 캐릭터를 회전시킬 때 쓰이며, 위 방향키를 통하여 캐릭터를 앞으로 전진시킬 수 있다. 다음은 캐릭터를 회전시켰을 때의 모습이다.



1. 토론

3D를 새로 한다는 점에서 익숙하지 않았지만 재밌는 과제였다. wire-frame이 무엇인지를 먼저 배울 수 있었고, 이를 구현하기 위해 vertex간의 관계를 잘 파악해야했다. object 파일을 분석하면서 face가 vertex간의 정보를 어떻게 저장하고 있는지 확인하고 그에 맞게 각 vertex들을 buffer에 넣어서 line으로 출력했다.

다만 조금 힘들었던 것은, 이 때까지 object하나 당 모든 vertex를 생산해서 buffer를 하나씩 썼는데, 파일 load를 하면서 매번 vertex를 만들기에는 양이 너무 많았다. 그래서 성질이 같은 object에서는 vertex와 buffer는 한 번씩만 생산하고, 좌표만 다양하게 설정해주는 방법으로 고쳤다. 예를 들어 class CAR는 road 한 줄에서의 car들을 관리하고 있었는데 이렇게 되면 road가 생성될 때 마다 class CAR에 있는 load가 불려지기 때문에 매우 비효율적이다. 때문에 class CAR를 road 한 줄이 아닌, map 전체의 car를 관리하는 class로 변경하였다.

또한 load를 하고 나니 프로그램의 속도가 매우 느려졌는데, 코드를 검토하면서 비효율적인 부분을 많이 찾을 수 있었다. matrix의 곱이 반복문 전체가 같게 적용이 되는데도, 반복문 안에 있는 것과 같이 코드를 고치면서 신경쓰지 못한 부분을 고쳐나갔다.

1. 결론

이번 assignment에서는 3D로 물체를 표현하면서 이때까지 수업시간에 배웠던 다양한 함수들을 많이 써볼수 있었다. object 파일에 삼각형으로 표현된 물체들을 관계에 알맞게 사각형으로 바꿔서 출력하고, 시점이동에 따라 캐릭터를 회전시켜보았다.

1. 개선 방향

이번 assignment에서 object를 받아올 때 파일을 미리 분석하여 vertex가 몇 개인지 분석하여 그것을 위한 배열을 미리 만들어놓고 시작했는데, 확장성을 고려한다면 파일 자체를 프로그램이 분석하여 vertex의 정보나 face의 정보를 받아오도록 하는 것이 더욱 나을 것 같다.

1. 참고문헌

<http://www.cs.unm.edu/~angel/BOOK/INTERACTIVE_COMPUTER_GRAPHICS/SIXTH_EDITION/CODE/>

https://github.com/IlhwanHwang/CSED451\_assn2/blob/master/assn2

Interactive Computer Graphics - A Top-Down Approach 6e By Edward Angel and Dave Shreiner (Pearson, 2012) BBS