HW1 보고서

컴퓨터공학부
2013-11413 우주

1. 실행 방법

Python 2.7 버전을 사용하였고 virtualenv로 격리하여 필요 패키지를 설치하였다. Python 2.7은 Ubuntu 16.04 에 기본 설치되어 있다. 실행은 아래 순서대로 진행하면 된다. 터미널에서 커맨드라인으로 실행해야 하는 부분은 앞에 $ 를 써 놓았다.

a. (pip가 설치되어 있지 않은 경우) $ sudo apt-get install python-pip
b. (virtualenv가 설치되어 있지 않은 경우) $ pip install virtualenv
c. 제출한 숙제 디렉토리로 이동. requirements.txt 가 존재하는 폴더.
d. (최초 실행시에만) $ virtualenv .venv
e. $ source .venv/bin/activate
f. (최초 실행시에만) $ pip install -r requirements.txt
g. $ python HW1.py

2. 조작법

총 12가지 키보드 인풋을 처리하였다. 방향키 이외의 키들은 영어 키보드일 때 작동한다.

a. 키보드 Up arrow : 카메라 위치를 뒤로 옮긴다.
b. 키보드 Down arrow : 카메라 위치를 앞으로 옮긴다.
c. 키보드 Right arrow : World Coordinate 의 z축의 양의 방향으로 회전한다.
d. 키보드 Left arrow : World Coordinate 의 z축의 음의 방향으로 회전한다.

e. 키보드 w 키 : 차를 z축의 양의 방향으로 회전한다. (Local Coordinate 회전)
f. 키보드 s 키 : 차를 z축의 음의 방향으로 회전한다. (Local Coordinate 회전)
g. 키보드 d 키 : 차를 앞으로 이동. (차의 앞부분이 보고 있는 방향으로)
h. 키보드 a 키 : 차를 뒤로 이동. (차의 뒷부분이 보고 있는 방향으로)

i. 키보드 i 키 : 사다리를 뒷쪽으로 이동.
j. 키보드 k 키 : 사다리를 앞쪽으로 이동.
k. 키보드 l 키 : 사다리의 길이 증가.
l. 키보드 j 키 : 사다리의 길이 감소.

3. 구현 내용

a. Create a hierarchical model using matrix stacks : **OK**

b. The model should consists of three-dimensional primitives such as polygons, boxes, cylinders, spheres and quadrics. : **OK. 2차원 primitives 로 3차원 도형 구현**

c. The model should have a hierarchy of at least three levels : **OK. 최대 4단계까지 사용.**

d. Animate the model to show the hierarchical structure : **OK. 바퀴 회전. 사다리 회전 및 늘리기**

e. Make it aesthetically pleasing or technically illustrative

4. 구현 방법

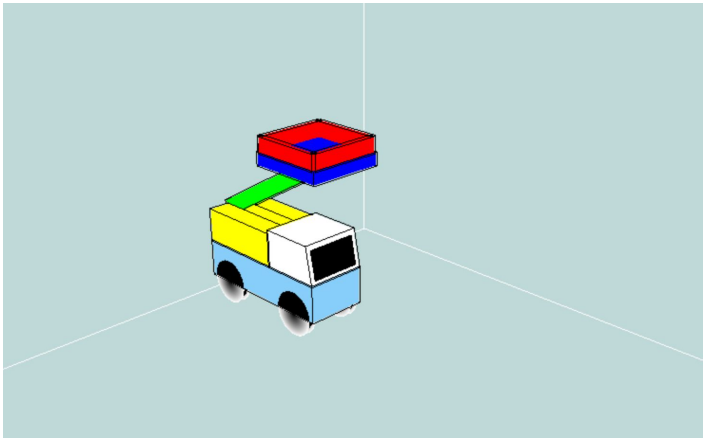

그림 1. 구현한 사다리 차

과제를 진행하기 전, 먼저 축을 결정하고 진행하였다. 위의 스크린샷에서 위로 향하고 있는 축이 Z축, 오른쪽 밑으로 향하고 있는 축이 Y축, 왼쪽 밑으로 향하고 있는 축이 X축이다. gluCylinder나 glutSolidCube 등 3차원 도형을 그리는 함수들은 사용하지 않고 GL_QUADS 등등 2차원 primitives 를 활용하여 3차원 도형을 만들었다. 사다리 차 구현은 앞 쪽 위의 흰 색 부분을 시작점으로 하였다. 그리는 순서는 앞의 흰 부분 => (PUSH) => 뒷 쪽 노란 부분 및 위의 사다리 부분 => (POP) => 차의 아래 부분 및 바퀴 순으로 그렸다. 각 부분을 그리면서 작업하고 고려한 내용은 다음과 같다.

흰 색 부분은 앞 쪽이 약간 경사진 모양이어서 glutSolidCube 같은 육면체를 그리는 함수를 쓰지 않고 GL_QUADS로 그렸고, 육면체의 모서리에는 GL_LINES 로 선을 넣어서 표시해주었다. 이후 그리는 육면체들은 모두 같은 식으로 그렸다.

그 다음으로 Y축으로만 translate 하여 뒷 부분으로 이동하여 노란 부분 및 사다리를 그려주었다. 이 때 초록색 사다리 부분은 특정 각도만큼 rotate 한 다음 Y축 방향으로 육면체를 그린 것이고 키보드 인풋으로 해당 회전 각도를 조절할 수 있게 하였고, 육면체(사다리)의 길이도 키보드 인풋으로 조절할 수 있게 하였다. 이렇게 해서 사다리가 움직이는 것을 애니메이트 할 수 있었다. 위의 짐 싣는 부분을 그린 부분은 사다리를 그릴 때 회전한 각도를 a라고 하면 다시 -a만큼 회전하여서 사다리의 각도 및 길이와는 상관 없이 아래 X-Y 평면과 항상 평행할 수 있게 하였다. 이 부분까지 그리는데 앞의 흰 부분 => 뒤의 노란 부분 => 사다리 => 위의 짐 싣는 부분 등을 그리면서 총 4단계의 스택을 활용하였다.

마지막으로 차의 아래 부분과 바퀴를 그렸다. 하늘색 큐브는 앞의 흰 부분을 그린 것과 동일한 방법으로 육면체를 그렸다. 바퀴를 그릴 때 원래 gluCylinder 함수를 사용하여 원기둥을 그렸는데 위와 아래의 원 부분이 채워지지 않아서 직접 바퀴를 그렸다. GL_TRIANGLE_FAN 을 활용하여 1도씩 움직이면서 삼각형을 그려서 위, 아래 원을 만든 다음 GL_QUAD_STRIP 으로 두 원 사이를 채워주어 바퀴(원기둥)을 그렸다. 차가 움직일 때 바퀴도 회전하는 것을 보여주기 위해서 반은 흰 색, 반은 검정색으로 채웠다. 이 부분까지 그리는데 앞의 흰 부분 => 아래 하늘색 부분 => 바퀴까지 하여 총 3단계의 스택을 활용하였다.