**기초 컴퓨터 그래픽스**

**HW3 README**

학번 20171687 이름 전민수

**1. [환경 명세]**

1) 본인 프로그램의 실제 구동 환경을 명시 할 것 (OS, CPU, GPU, Compiler 등등)

- window10 64bit, AMD Ryzen 5600X, rtx 3070, visual studio 2019 – win 32, debug

**2. [요구사항]**

   1. Modeling Transformation

1) 먼저 가상의 3차원 세상의 바닥과 좌표의 기준이 되는 세상 좌표계를 그려라 (최대 10점)

- 확인 방법:

실행을 시키면, 바닥이 보인다.

실행을 시킨 후, 2를 눌러 카메라를 변경하고, o를 눌러 좌표계를 가리고 있던 호랑이를 치우면 좌표계가 보인다.

2) 최대 5개까지의 서로 다른 정적인 물체를 서로 다른 모델링 변환을 사용하여 가상의 세상에 배치하라 (물체 당 5점 최대 25점)

a) 사용한 물체: 용

- 부여한 **서로 다른** 모델링 변환:

용을 일으켜 세우기위한 회전변환, 용을 원하는 곳에 배치하기 위한 회전변환

용을 키우기 위한 scaling

용의 위치를 바꾸기 위한 translation

- 확인 방법:

실행 후, 5번을 눌러 움직일 수 있는 카메라로 간 후, q를 눌러 줌을 최대로 당긴다. 중앙정도에 하늘에 떠있는 용을 확인할 수 있다.

b) 사용한 물체: 아이언맨

- 부여한 **서로 다른** 모델링 변환:

아이언멘의 위치를 바꾸기 위한 translation

아이언맨을 원하는 방향을 바라보게 하기 위한 rotate

아이언맨의 크기를 원하는 대로 맞추기 위한 scaling

- 확인 방법:

실행 후, 5번을 눌러 움직일 수 있는 카메라로 간 후, q를 눌러 줌을 최대로 당긴다. 우측하단을 보면 위치해 있는 아이언 맨을 볼 수 있다.

c) 사용한 물체: 탱크

- 부여한 **서로 다른** 모델링 변환:

탱크의 위치를 바꾸기 위한 translation

탱크를 원하는 방향을 바라보게 하기 위한 rotate

탱크의 크기를 원하는 대로 맞추기 위한 scaling

- 확인 방법:

실행 후, 5번을 눌러 움직일 수 있는 카메라로 간 후, q를 눌러 줌을 최대로 당긴다. 가운데 하단을 보면 위치해 있는 탱크를 볼 수 있다.

d) 사용한 물체: 옵티머스

- 부여한 **서로 다른** 모델링 변환:

옵티머스의 위치를 바꾸기 위한 translation

옵티머스를 원하는 방향을 바라보게 하기 위한 rotate

옵티머스의 크기를 원하는 대로 맞추기 위한 scaling

- 확인 방법: 실행 후 왼쪽 화살표를 두 번 누르면 보인다.

e) 사용한 물체: 소

- 부여한 **서로 다른**모델링 변환:

소의 위치를 바꾸기 위한 translation(소가 호랑이와 부딪혔을 때 호랑이가 움직이는 방향으로 소가 살짝 밀린다.)

소를 원하는 방향을 바라보게 하기 위한 rotate

소의 크기를 원하는 대로 맞추기 위한 scaling

- 확인 방법:

실행 후, 5번을 눌러 움직일 수 있는 카메라로 간 후, q를 눌러 줌을 최대로 당긴다. 왼쪽 상단을 보면 위치해 있는 소를 볼 수 있다.

호랑이는 랜덤한 경로를 따라 움직이기 때문에, 호랑이가 소와 만날때까지 기다리면, 소가 호랑이가 움직이는 방향으로 살짝 밀리는 모습을 관찰 할 수 있다.

3) 최대 4개까지의 서로 다른 동적인 물체를 가상의 세상에 배치하라 (물체 당 10점 최대 40점). 각 동적 물체는 이동 변환, 크기 변환, 그리고 회전 변환 등의 기본 기하 변환 중 최소한 두 개 이상을 사용하여 서로 다른 움직임을 표현해야 하며, 각 동적 물체는 키보드 또는 마우스 동작을 통하여 움직임과 멈춤을 조절할 수가 있었야 한다 (자신이 선택한 최대 네 개의 동적인 물체에 대해 이 기능이 구현이 안되어 있으면 물체 당 4점 감점). 요구 사항은 아니나 동적인 물체들 중 최소한 1개의 물체에 대해서는 뉴턴의 운동의 법칙과 같이 물리적으로 충실한 방법을 사용하여 움직임을 표현해볼 것.

a) 사용한 물체: 호랑이

- 부여한 **서로 다른** 모델링 변환:

호랑이는 랜덤한 경로를 움직인다.

호랑이를 세우기 위한 rotation

호랑이를 원하는 각도로 돌리기 위한 rotation

호랑이를 원하는 위치에 두기 위한 translation

을 수행한다.

멈춰있을 때에는 호랑이의 발도 멈춰있는다.

호랑이의 방향은 100 타임마다 새로 정해진다.

- 확인 방법:

5번을 누른 후 q를 눌러 줌을 최대로 당긴 후에 p를 누른다.

b) 사용한 물체: 늑대

- 부여한 **서로 다른** 모델링 변환:

옵티머스의 주위를 반원 궤도로 돈 후에 옵티머스 위를 날아 올라 점프한다.

늑대를 원하는 각도로 돌리기 위한 rotation

늑대를 원하는 위치에 두기 위한 translation

늑대를 원하는 크기로 만들기 위한 scaling

을 수행한다.

늑대는 늑대 본인의 몸의 각도가 180도가 되기 전까지는 옵티머스의 반원 궤도를 돌다가 넘어가는 순간 몸의 각도를 270도로 고정하고 옵티머스를 기준으로 점프를 하는 운동을 한다.

늑대의 발은 멈춰있을 때에는 움직이지 않으며, 점프했을때에는 느리게 움직인다.

- 확인 방법:

5번을 누른 후 q를 눌러 줌을 최대로 당긴 후에 o를 누른다.

c) 사용한 물체: ben

- 부여한 **서로 다른** 모델링 변환:

ben은 넘어져 있는 상태로 시작한다. ben을 움직이라는 명령을 받았을 때 ben은 일어나서 원점을 기준으로 한바퀴를 돈다.

넘어져 있는 ben을 세우기 위한 회전변환

ben을 시작점에 위치시키기 위한 translation

ben을 원하는 크기로 만들기 위한 scaling

ben을 원운동을 하게 만들기 위한 rotation이 필요하다.

- 확인 방법:

5번을 누른 후 q를 눌러 줌을 최대로 당긴 후에 u를 누른다.

d) 사용한 물체: spider

- 부여한 **서로 다른** 모델링 변환:

이 거미는 중력의 영향을 받는것처럼 표현하였다. 움직이라는 명령을 받으면 뛰어오르며, 중력가속도를 주었다.

뛰어 오르기 위한 translation,

위치를 옮겨주기 위한 translation

크기를 맞추기 위한 scaling이 필요하다.

- 확인 방법:

5번을 누른 후 q를 눌러 줌을 최대로 당긴 후에 i를 누른다.

2. Viewing Transformation

1) 1번부터 4번까지의 카메라는 CCTV 카메라와 같이 주어진 위치에 고정하여 세상을 바라보는 카메라이다. 적절한 사용자 인터페이스 동작을 통하여 원하는 카메라에서 세상을 바라볼 수 있도록 하라

a) 1번 카메라 확인 방법:

숫자 1을 누른다.

b) 2번 카메라 확인 방법:

숫자 2를 누른다.

c) 3번 카메라 확인 방법:

숫자 3을 누른다.

d) 4번 카메라 확인 방법:

숫자 4를 누른다.

2) 5번 카메라는 동적인 카메라로서 사용자 인터페이스 동작을 통하여 다음과 같이 움직일 수 있 도록 하라

x,y,z 키와 w,a,s,d 키의 조합으로 카메라가 움직인다.

x는 u벡터 방향

y는 v벡터 방향

z는 n벡터 방향이다.

이 키를 통해 기준이 되는 축을 정한다

a) 5번 카메라 translation 확인 방법:

w와 s로 움직이는데, w가 +축방향, s가 -축방향으로 움직인다.

b) 5번 카메라 rotation 확인 방법:

a가 -축방향 기준으로 회전, d가 +축 기준으로 회전이다.

3) (추가) 1번 카메라에 대하여 고정된 위치를 중심으로 시선의 방향을 바꿀 수 있도록 하라. 어떠한 방식으로 구현할 지는 본인이 결정할 것.

a) 추가 구현 확인 방법:

방향키를 이용한다. 방향키를 이용해서 위 아래 좌 우를 살펴볼 수 있다.

3. Projection Transformation

1) 적절한 사용자 인터페이스 동작을 통하여 5번 카메라에 대하여 줌 인/줌 아웃 기능을 구현하라. 이때, 최대로 줌 인/줌 아웃할 수 있도록 적절히 범위를 설정하라.

- 확인 방법:

5번 카메라에서 q와 e 키를 활용하여 줌인 줌아웃을 할 수 있다.

q를 누르면 줌아웃이 되고 e키는 줌인이 된다.