

# REPORT

---



과목명 | 컴퓨터그래픽스  
학년 | 3학년  
학과 | 소프트웨어학부  
학번 | 20143068  
이름 | 서준교  
제출일 | 2018.06.23

# 목차

<b>1. 프로젝트 개요</b>	.....
- 프로젝트 요약	.....
- 팀 구성 및 역할 분담	.....
- 프로젝트 파일 구성	.....
<b>2. 구현된 기능</b>	.....
- Navigation	.....
- Scene Data 의 Save/Load	.....
- 충돌 감지	.....
- 게임적 요소	.....

# 1. 프로젝트 개요

## - 프로젝트 요약

이번 프로젝트에서는 실시간 렌더링을 통한 Navigation과 충돌 감지, 그리고 기타 게임적 요소를 추가한 Shader 기반의 모던 OpenGL을 이용한 3차원 가상 우주 박물관을 구현하였다. 각 Object는 Phong reflection model을 통해 렌더링되었으며, 특정 Object에는 Texture Mapping을 적용하였다.

## - 팀 구성 및 역할 분담

- \* 서준교(20143068) : Object 선정 및 배치, 보고서 작성
- \* 김철현(20143048) : 충돌 감지 구현
- \* 이진주(20163146) : Save/Load 기능 구현
- \* 공병민(20143029) : 게임적 요소 구현

역할 분담은 크게는 위와 같이 하였으며, 기능을 구현하면서 부족하거나 막혔던 부분에 대해서는 서로 의견을 공유하면서 해결하였다.

## - 프로젝트 파일 구성

- \* data 폴더 : 가상 박물관에 배치할 Object 파일이 들어있다.
- \* shader 폴더 : Vertex Shader와 Fragment Shader가 처리할 정보가 들어있다.
- \* scene 폴더 : Object들의 position, scale값, 회전값 등의 정보가 들어있는 텍스트 파일이 들어있다.
- \* Camera.h, Camera.cpp : 가상 박물관의 Navigation 기능을 구현한 소스파일
- \* Object.h, Object.cpp : Object 파일을 업로드하기 위해 구현된 소스파일
- \* Shader.h, Shader.cpp : Vertex Shader와 Fragment Shader가 한 프로그램상에서 data를 공유할 수 있도록 하는 소스파일
- \* SOIL.h : Texture mapping에 필요한 image를 관리하는 헤더파일
- \* mat.hpp, vec.hpp, transform.hpp : Homogeneous Coordinating에 필요한 vector, matrix 정보와 알고리즘이 포함된 소스파일
- \* main.cpp : 가상 박물관 내에 구현된 충돌, 게임 알고리즘이 구현되어 있는 소스파일

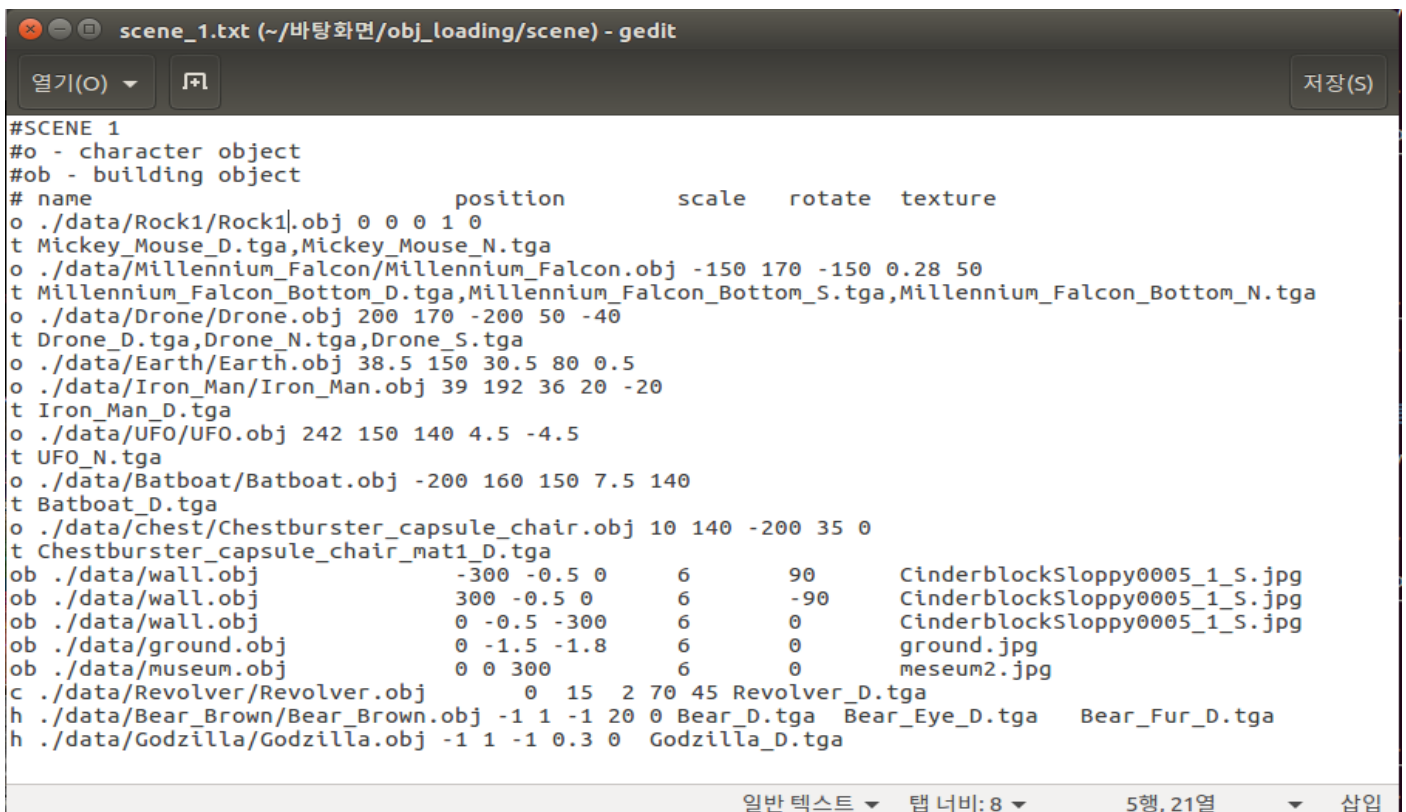
## 2. 구현된 기능

### - Navigation

가상 우주 박물관 내부를 키보드를 이용하여 이동할 수 있는 Navigation 기능을 구현하였다. 키 입력에 대해서 방향키는 special함수, 자판 입력(w, a, s, d)는 keyboard함수로 처리를 하였고, w, s는 OpenGL 3차원 좌표계 상에서 y축 방향, a, d는 방향 회전, 방향키는 x축, z축 방향으로 이동하도록 설계하였다.

### - Scene Data의 Save/Load

Object 배치의 편의성을 위해 가상 박물관에 존재하는 물체들의 Scene data를 파일 형식으로 저장하였다. Scene.txt 파일에는 각각의 Object에 대한 Position값, 회전 각도, Scale값, 파일명, 텍스처 파일의 정보가 들어있으며, 이 값들을 변경하고 Save한 다음 다시 프로그램을 실행시키면 변경된 정보가 반영되어 Object가 배치되도록 하였다. 또, 프로그램 상에서 Object를 움직인 다음 F버튼을 누르면 그 위치가 저장되고, 다시 실행 시 저장되었던 Object의 위치를 기반으로 Object가 배치된다. Q를 누르면 프로그램이 종료된다.

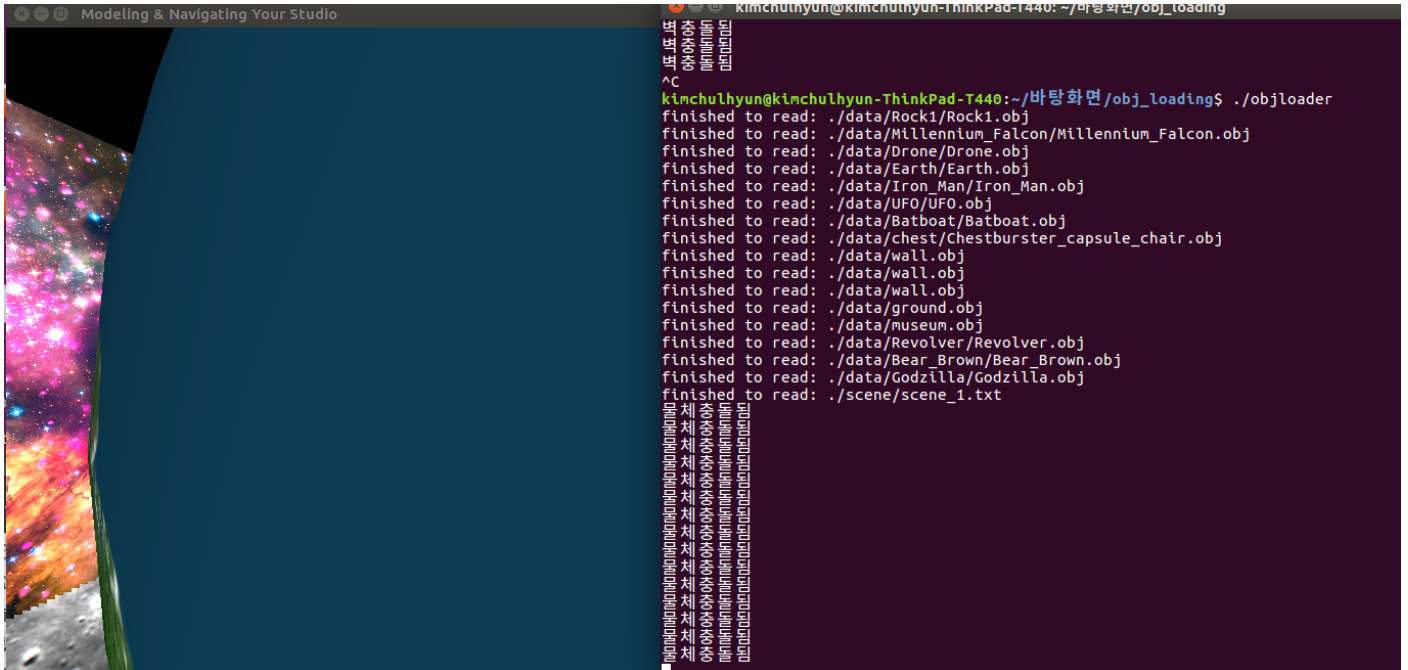


```
#SCENE 1
#o - character object
#ob - building object
# name                position                scale  rotate  texture
o ./data/Rock1/Rock1.obj 0 0 0 1 0
t Mickey_Mouse_D.tga,Mickey_Mouse_N.tga
o ./data/Millennium_Falcon/Millennium_Falcon.obj -150 170 -150 0.28 50
t Millennium_Falcon_Bottom_D.tga,Millennium_Falcon_Bottom_S.tga,Millennium_Falcon_Bottom_N.tga
o ./data/Drone/Drone.obj 200 170 -200 50 -40
t Drone_D.tga,Drone_N.tga,Drone_S.tga
o ./data/Earth/Earth.obj 38.5 150 30.5 80 0.5
o ./data/Iron_Man/Iron_Man.obj 39 192 36 20 -20
t Iron_Man_D.tga
o ./data/UFO/UFO.obj 242 150 140 4.5 -4.5
t UFO_N.tga
o ./data/Batboat/Batboat.obj -200 160 150 7.5 140
t Batboat_D.tga
o ./data/Chest/Chestburster_capsule_chair.obj 10 140 -200 35 0
t Chestburster_capsule_chair_mat1_D.tga
ob ./data/wall.obj -300 -0.5 0 6 90 CinderblockSloppy0005_1_S.jpg
ob ./data/wall.obj 300 -0.5 0 6 -90 CinderblockSloppy0005_1_S.jpg
ob ./data/wall.obj 0 -0.5 -300 6 0 CinderblockSloppy0005_1_S.jpg
ob ./data/ground.obj 0 -1.5 -1.8 6 0 ground.jpg
ob ./data/museum.obj 0 0 300 6 0 meseum2.jpg
c ./data/Revolver/Revolver.obj 0 15 2 70 45 Revolver_D.tga
h ./data/Bear_Brown/Bear_Brown.obj -1 1 -1 20 0 Bear_D.tga Bear_Eye_D.tga Bear_Fur_D.tga
h ./data/Godzilla/Godzilla.obj -1 1 -1 0.3 0 Godzilla_D.tga
```

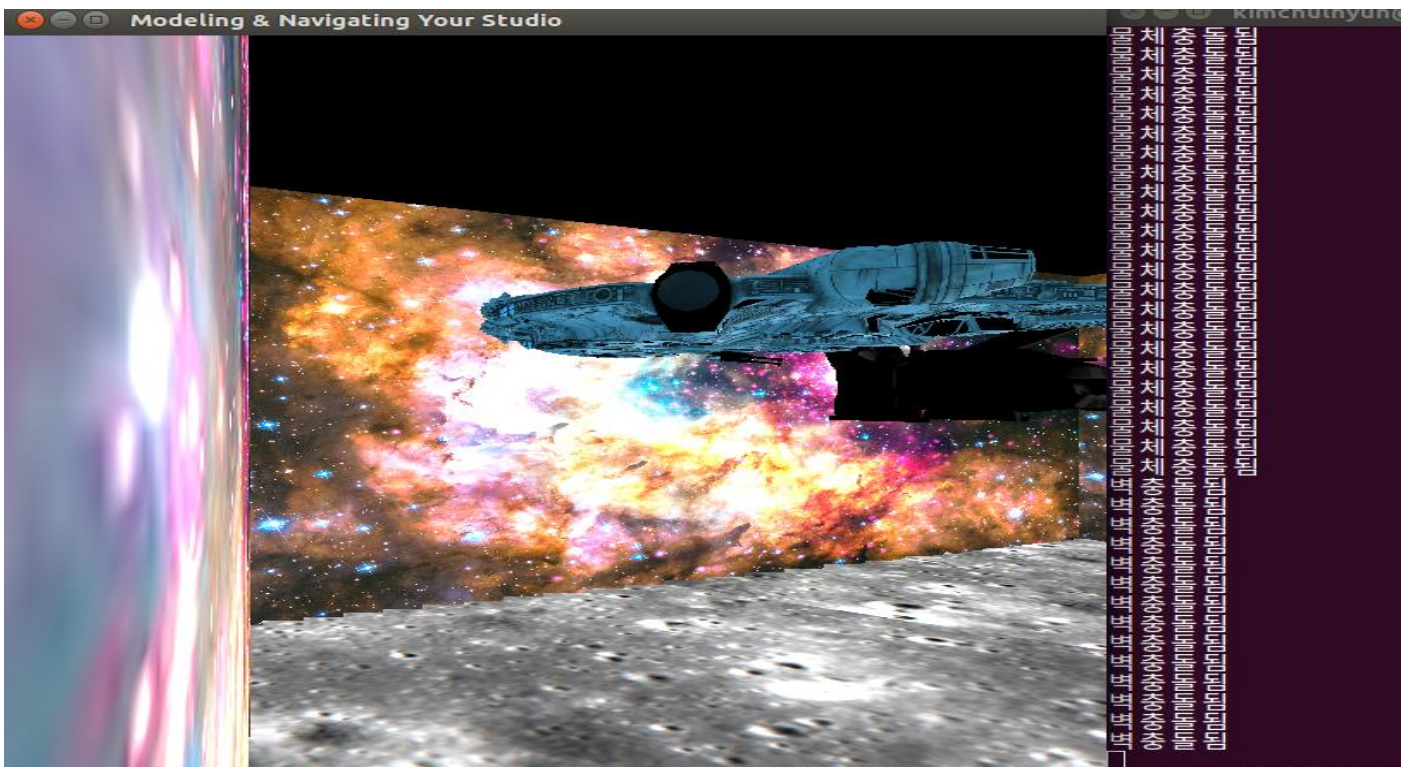
<object의 정보가 저장된 텍스트파일>

## - 충돌 감지

보다 사실적인 Navigation 구현을 위해서, 카메라 좌표계가 Object와 충돌하였을 때는 더 이상 앞으로 진행이 불가능하도록 해주는 충돌 감지 기능을 구현하였다. obj 파일 내부에 있는 vertex값을 입력받을 때 x축, y축, z축별로 최댓값, 최솟값을 구한 다음, 최댓값에서 최솟값을 뺀 값을 변의 길이로 하는 boundary box를 생성하여 해당 구역에 Camera 좌표가 진입을 하게 되면 더 이상 진행이 되지 않고, 충돌했음을 알리는 Message가 Terminal에 출력된다.



### <물체에 충돌했을 때>



### <벽에 충돌했을 때>

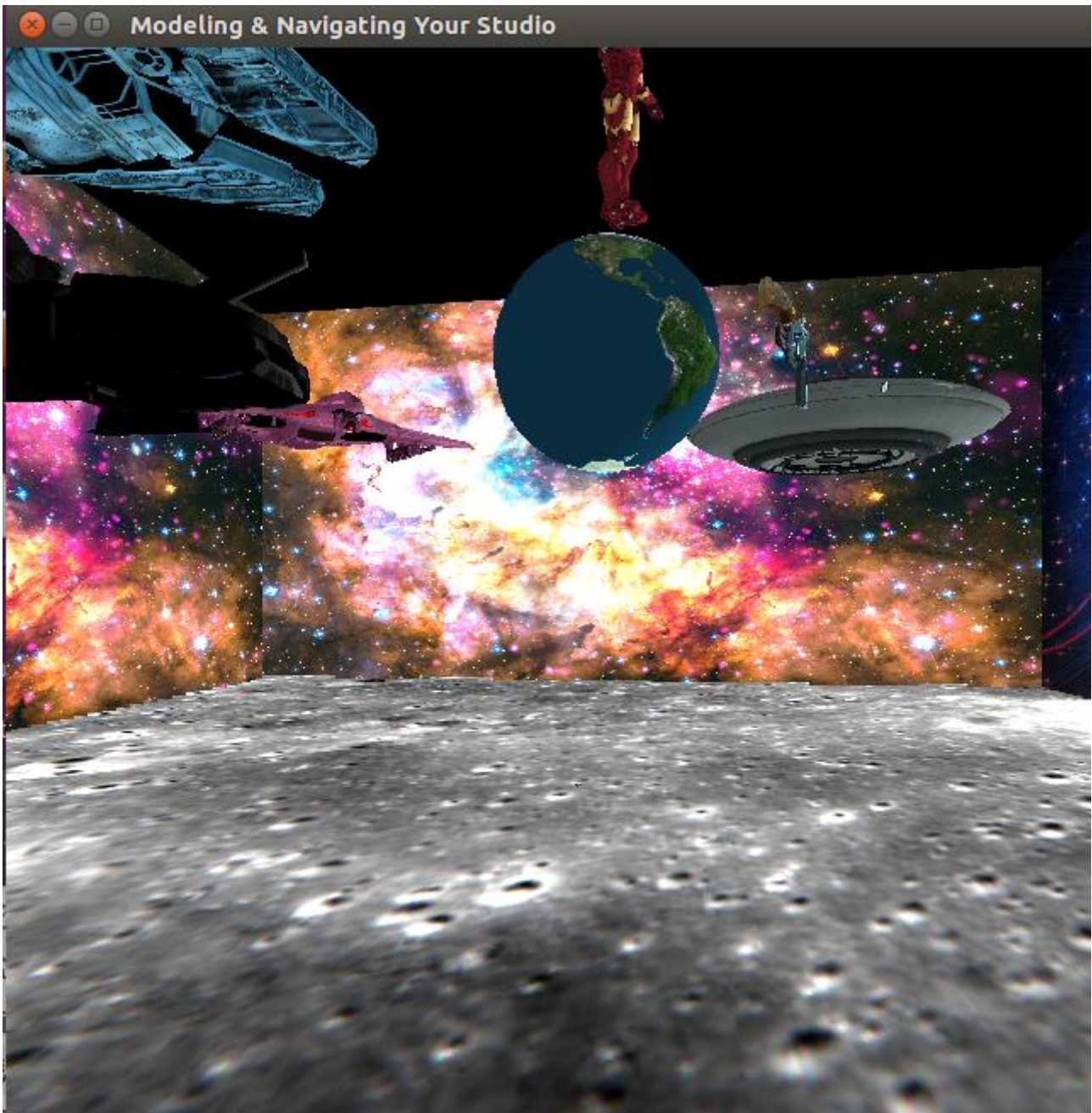


## - 게임적 요소

가상 우주 박물관에 몇몇 게임적 요소를 구현하였다.

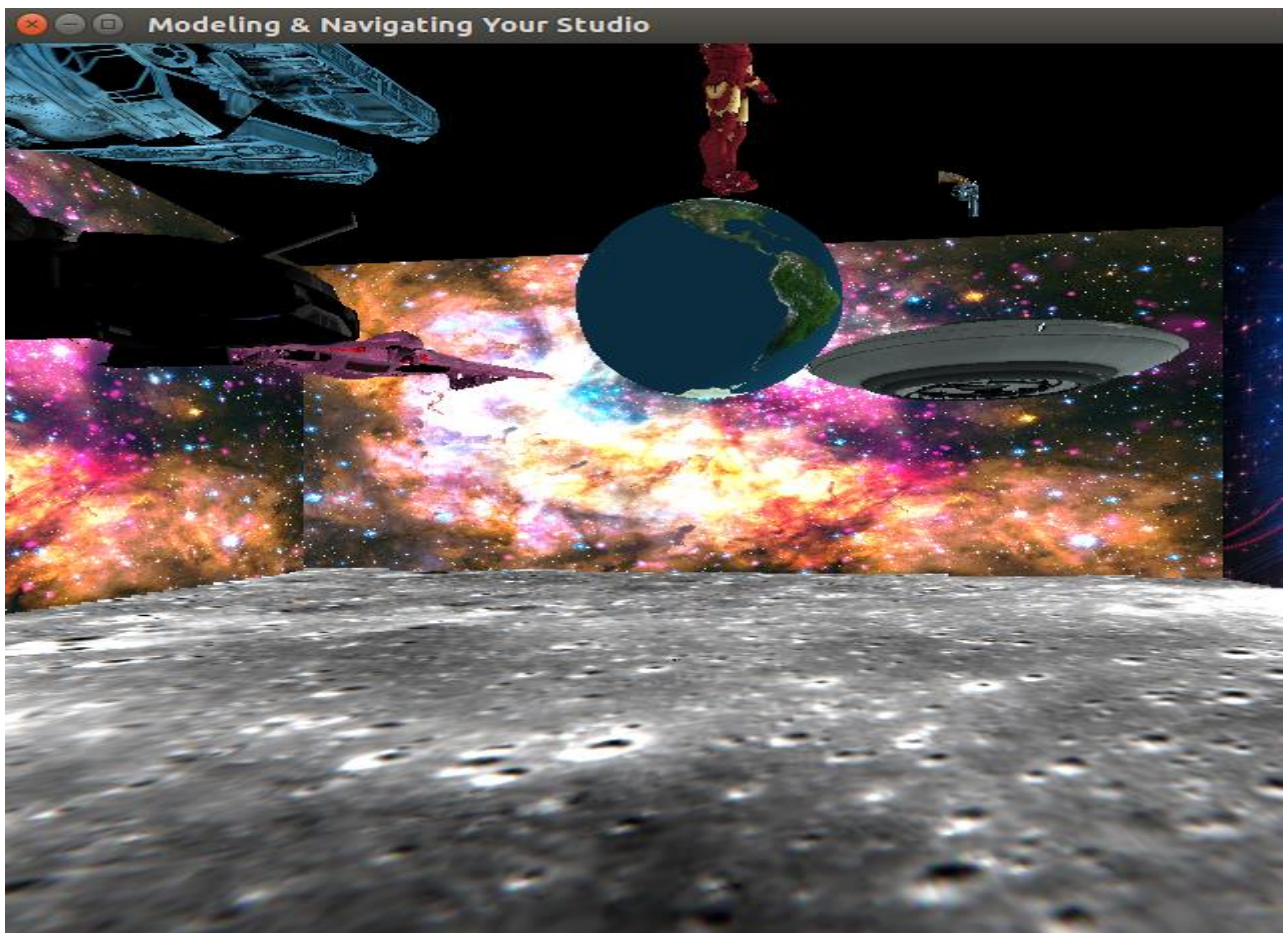
1) Object를 선택하여 이동, 삭제

m을 누르면 화면상에 리볼버가 등장한다. 그리고 c를 누르면 리볼버가 각각의 object를 가리키게 되고, m을 한번 더 누르게 되면 해당 object를 키보드로 이동시킬 수 있다. m을 누르지 않고 z를 누르면 해당 object가 화면상에서 사라지게 되고, v를 누르면 다시 나타난다.

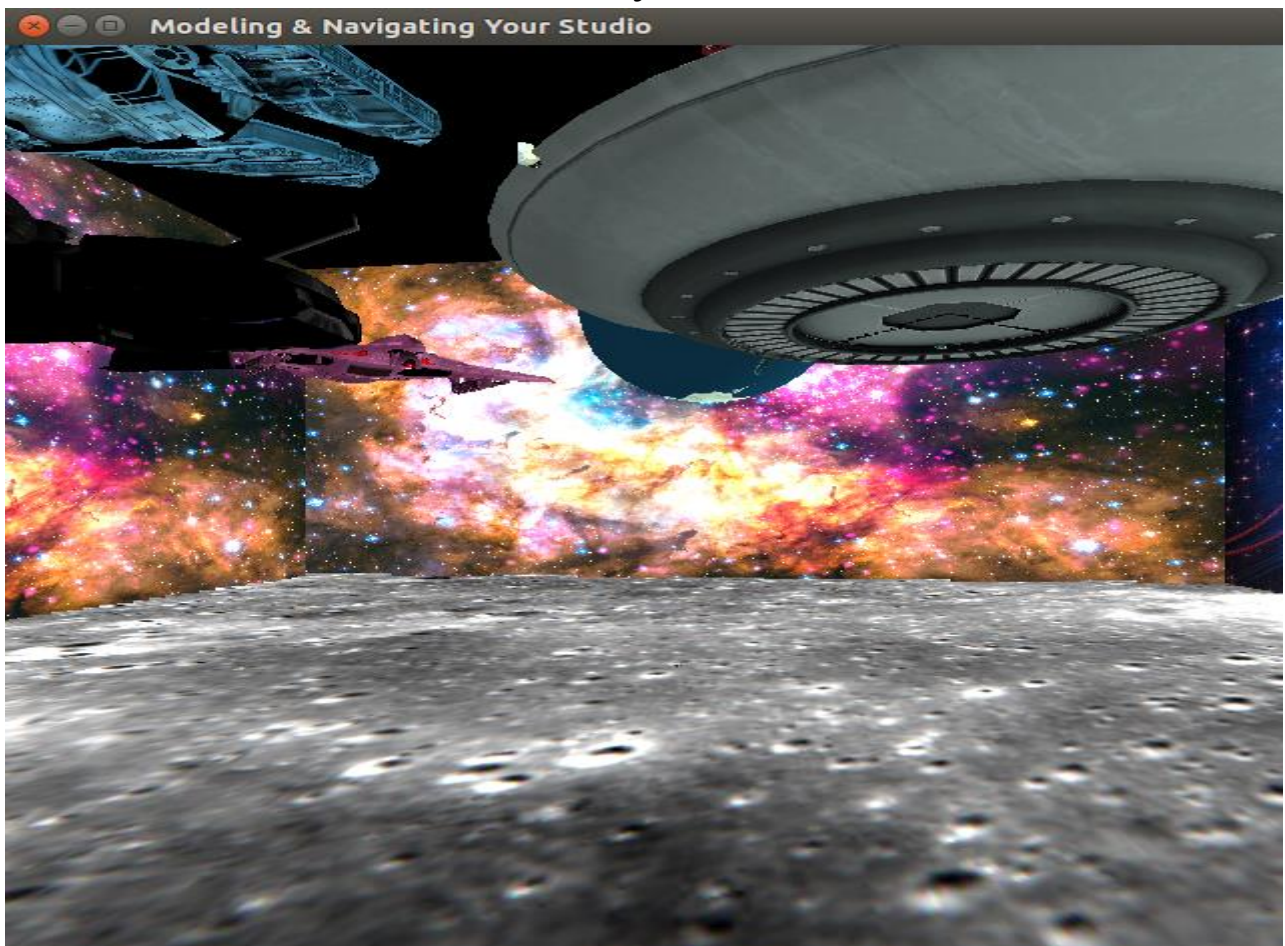


<m을 눌렀을 때 리볼버가 등장한 사진>



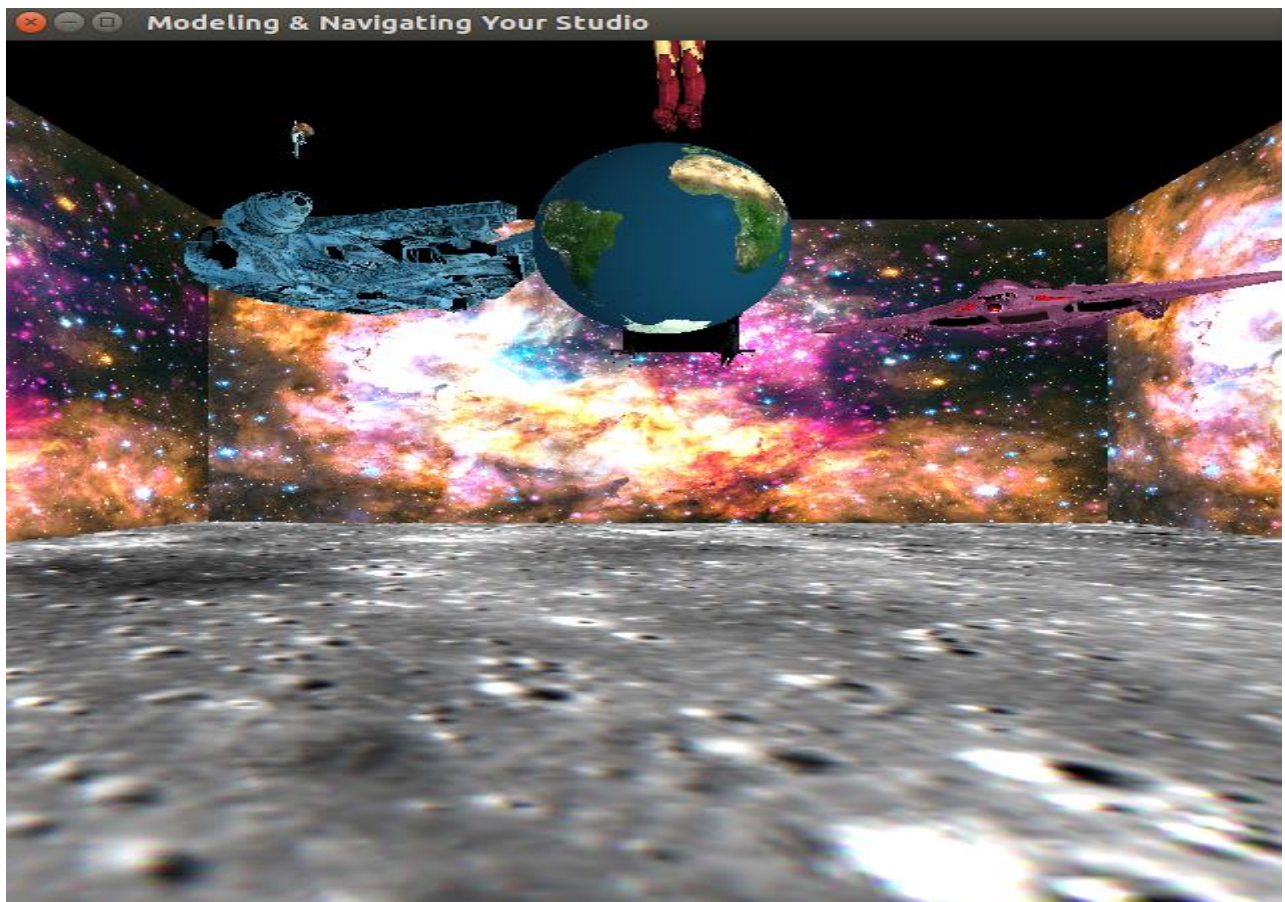


<c를 눌러 UFO Object를 가리킨 사진>

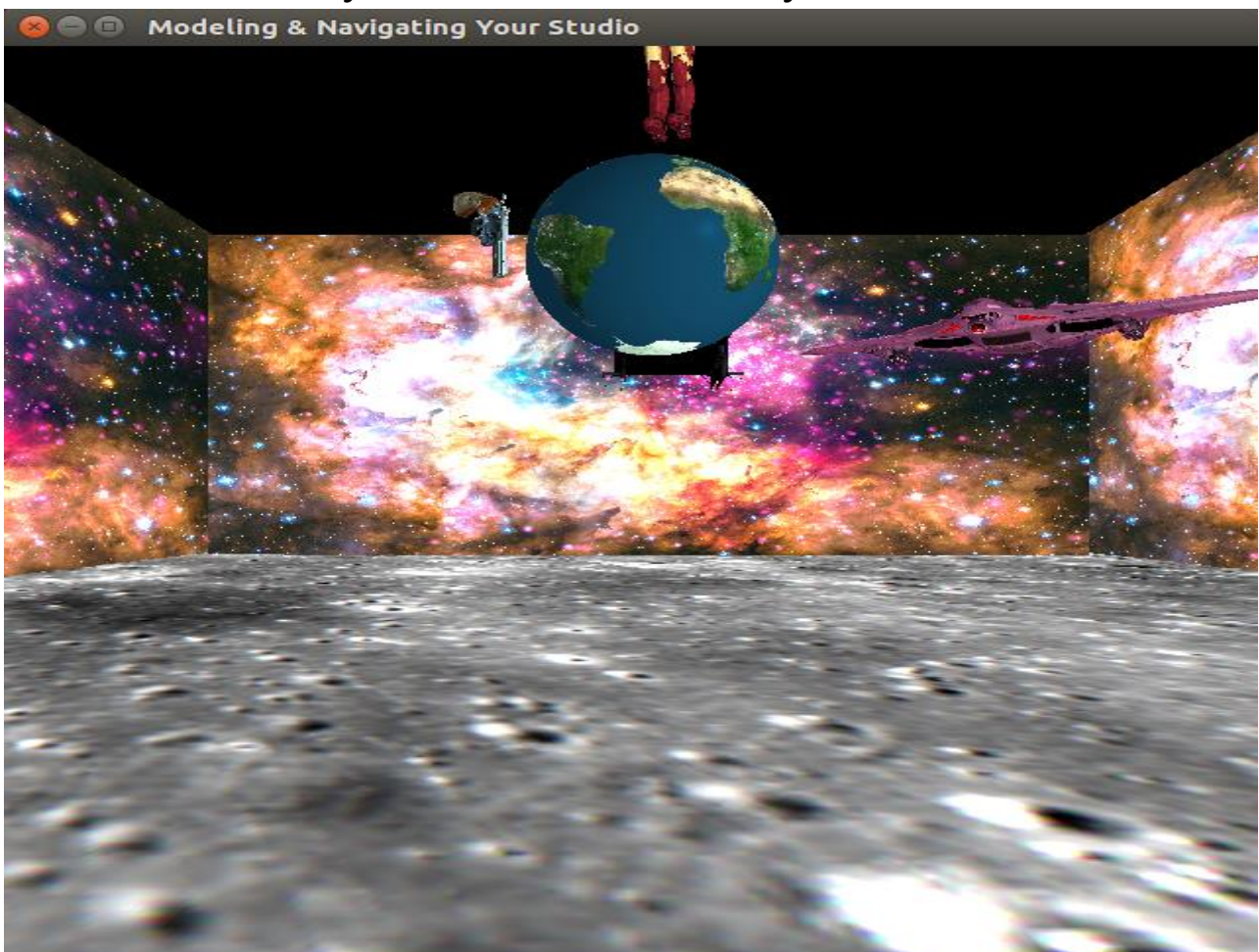


<선택된 UFO Object를 이동시킨 사진>



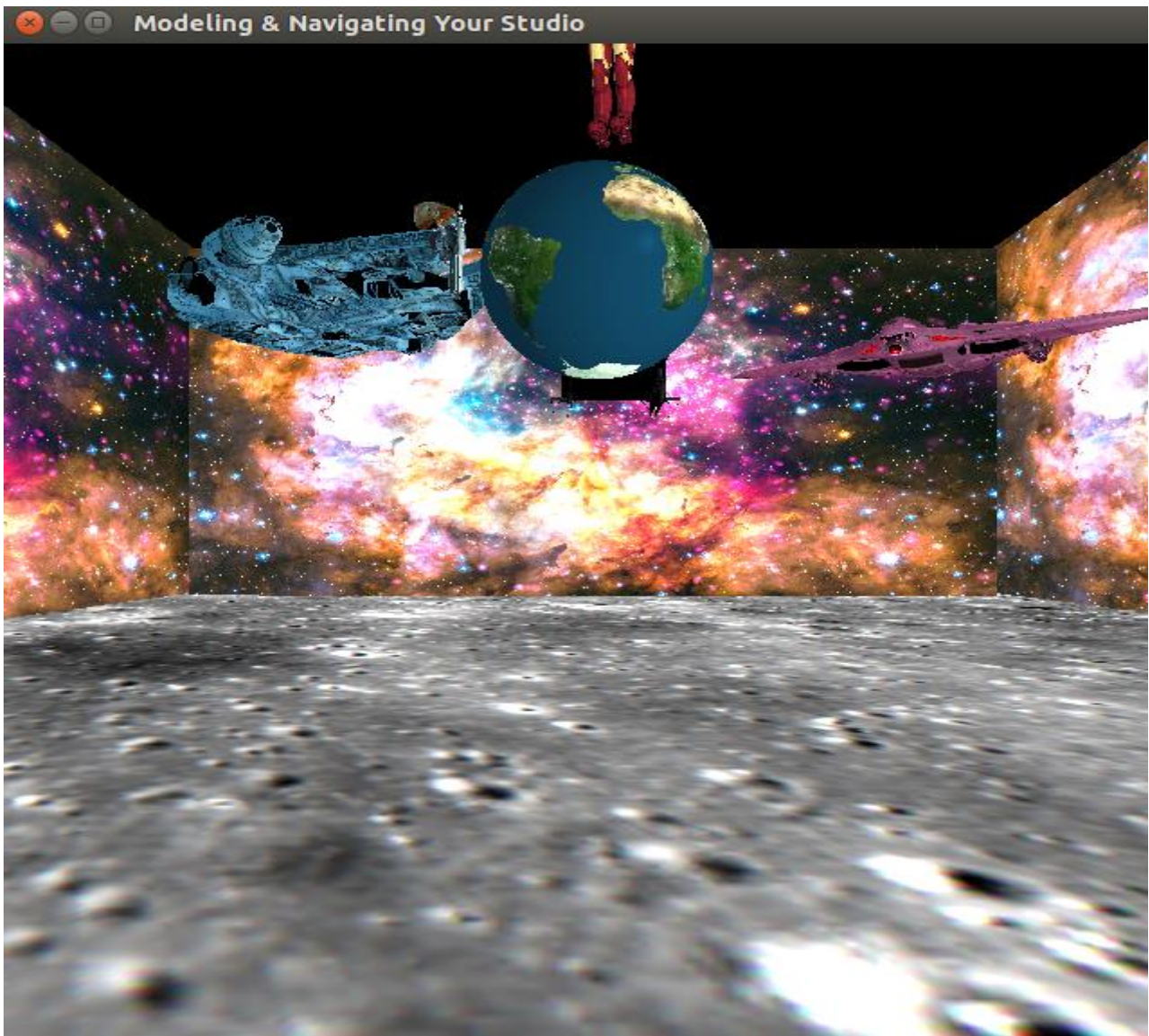


<Object 삭제를 위해 비행선 object를 선택>



<z버튼을 눌러 비행선 object를 삭제함>

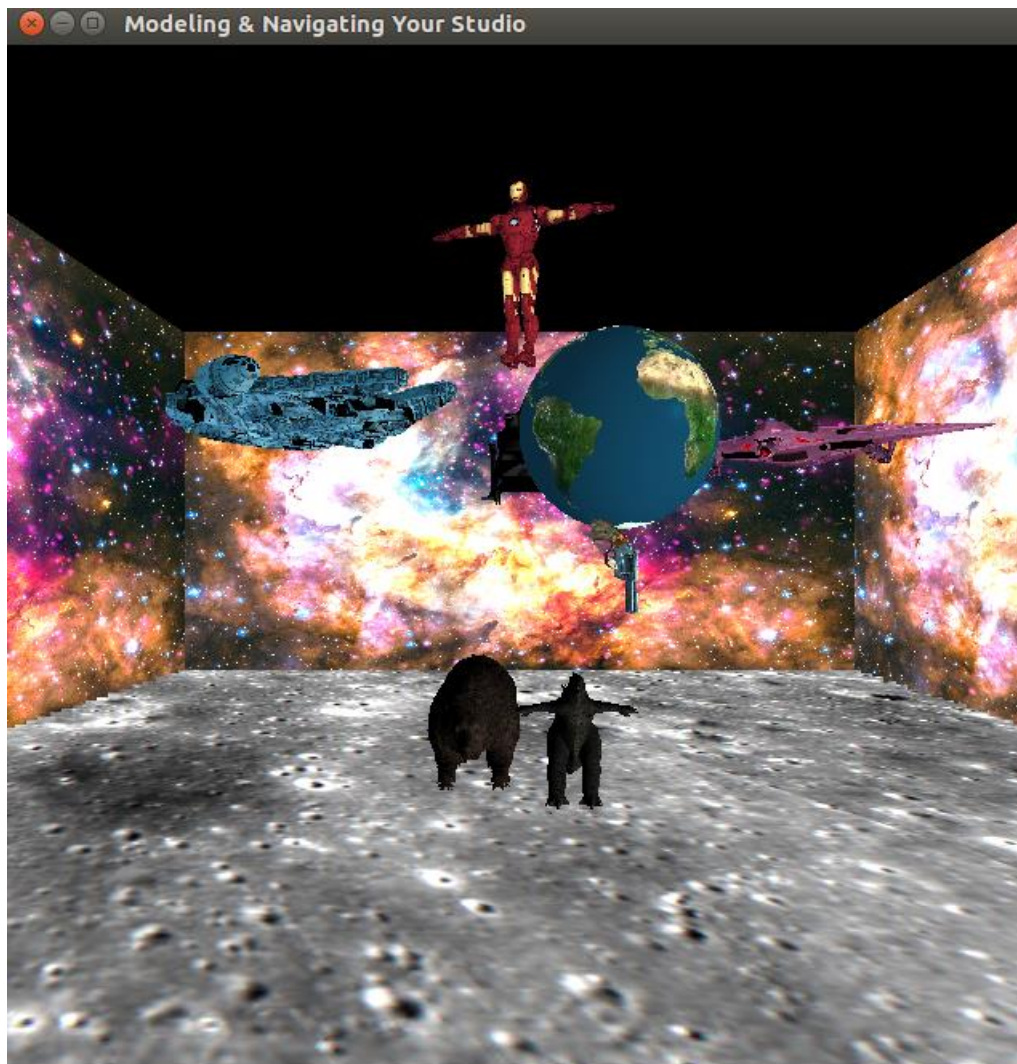




<v버튼을 눌러 삭제되었던 비행선 object를 복원시킴>

## 2) Game Mode (고릴라와 고질라 잡기)

m을 누르면 화면상에 리볼버가 등장하고 여기서 l를 누르게 되면 고질라와 곰이 등장한다. 리볼버를 이동시켜서 리볼버 위치의 x, z좌표와 곰, 고질라의 x, z좌표가 겹치게 되면 해당되는 곰 또는 고질라가 화면상에서 사라지고, 고질라와 곰을 모두 잡은 다음 l를 다시 누르게 되면 이동속도가 더 빨라진 곰과 고질라가 등장하게 된다. (난이도 UP)



<m을 누른 뒤 l를 눌러 게임모드에 진입한 모습 (리볼버로 고질라와 곰을 모두 잡으면 다음 level로 진입할 수 있다.)>