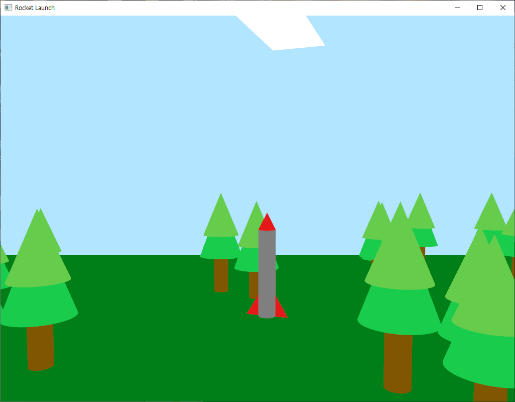
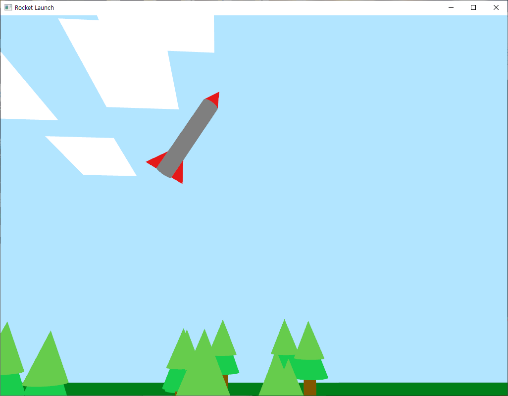
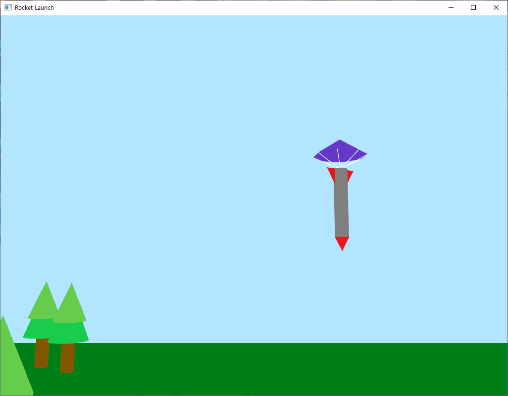
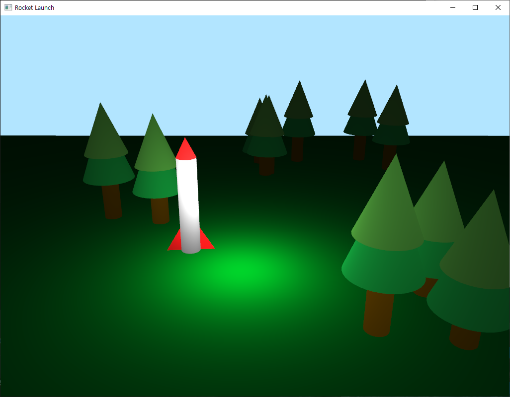
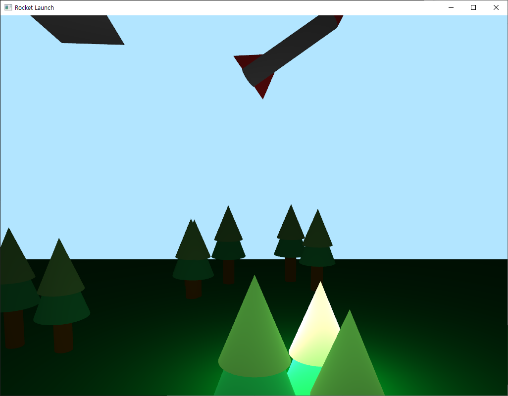
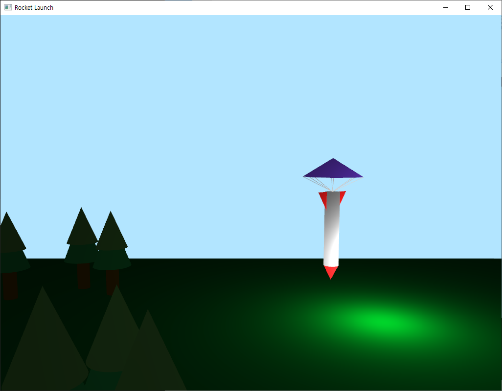
컴퓨터 그래픽스 과제 보고서

12171698 정현준

* 스크린샷 및 구현내용

* 원뿔, 원통 기반으로 로켓을 모델링했으며, 주변 조형물로 구름과 나무를 설치했다.
* 로켓은 실제 물리법칙이 적용된 것은 아니지만 상승할 때는 느려지고, 하강할 때는 빨라진다.
* 특정키로 충전 후 일정량이 충전되면 발사할 수 있고, 낙하산을 펼 수 있다.
* 조명이 없는 바이너리와 조명이 있는 바이너리를 둘 다 구현했다.
* 조명이 로켓의 바닥면을 따라가게 구현했다.
* 조작방법
* 방향키와 마우스로 카메라를 이동 및 회전시킬 수 있다.
* Z키로 로켓을 충전한다. Z키를 누르고 있는 동안 콘솔 창에 충전량이 표시된다. 충전량이 16이상일 때부터 발사할 수 있으며, 32가 넘어간 상태에서 발사할 경우 낙하산을 펼 수 있다.
* L키로 로켓을 발사한다.
* P키로 낙하산을 편다.
* R키로 카메라 시점을 초기화할 수 있다.
* 주요 코드에 대한 설명 & 소스코드 내에서의 위치

모든 모델 정보는 model.h에 포함되어 있다.

* gen\_rectangle, gen\_cylinder, gen\_cone, gen\_umbrella 함수는 각각 평면, 원통, 원뿔, 우산 모양을 가지는 정점 버퍼를 만들어 주는 함수다.
* cal\_normal, cal\_normalc 함수는 위 함수들로 생성한 정점 버퍼로부터 법선 벡터를 만들어주는 함수다.
* dup3 함수는 정점 컬러 버퍼를 생성해주는 함수다.
* 176줄부터 256줄까지 모델링에 필요한 모든 정점 정보들이 작성되어 있다.
* 258줄의 graphic\_buffer 클래스와 graphic\_object 클래스는 오브젝트들을 관리하고 그리는데 사용할 수 있는 메서드들이 구현되어 있다.

main.cpp에는 렌더링에 사용되는 모델 오브젝트와 렌더링 루프가 구현되어 있다.

* 91에서 123줄은 모델 오브젝트를 생성하는 부분이다.
* 157부터 230줄은 렌더링을 수행하는 부분이다.
* 157~158, 163, 171, 180, 221 줄은 조명처리를 위해 Transition Matrix를 쉐이더에 등록하는 부분이다.
* 193부터 218줄은 로켓의 위치를 계산하는 부분이다.

로켓의 움직임을 처리하는 부분은 렌더링 루프 내 193~218줄에 위치한다.

* 먼저 isLaunched 변수를 확인해 controls.cpp에서 발사 준비가 되었는지 확인한다.
* rotate는 현재 로켓의 각도를 나타내며 이 값이 90도가 넘으면 isSafeToUnfold를 통해 낙하산을 펼 수 있음을 알린다. 낙하산은 충전량이 32이상일때만 펴진다.
* 높이는 power \* sin(rotate)로 계산되고, 이동량은 rotate \* (power / 32 + 16)으로 계산된다.
* 낙하산이 펴진 상태라면 각도가 180가 될 때 까지 회전시키고, 하강 속도를 늦춘다.

controls.cpp에는 카메라 및 로켓 조작에 필요한 키 처리 부분이 구현되어 있다.

* 88줄에서 121줄 까지가 Z키, L키, P키, R키에 대한 동작이 구현되어 있는 부분이다.
* 로켓이 발사되려면 일정량 이상이 충전되어야 하는데, 이 충전량을 저장하고 있는 변수가 power 변수이다. 이 변수는 main.cpp에 선언되어 있어서 렌더링 루프에서 이 변수를 이용해 로켓의 발사 세기를 조절할 수 있다.

조명처리 부분은 StandardShading.vertexshader 및 .fragmentshader에 구현되어 있다.

* Vertexshader에선 월드 좌표계에서의 정점의 좌표인 Position\_worldspace를 fragmentshader로 넘겨준다.
* Fragmentshader는 광원의 위치와 정점의 좌표를 통해 거리를 계산하고, 컬러값을 계산한다.