제33회 KOI 공모부문 작품 설명서(양식)

접수번호
GG_H_028

KOI 공모부문

Guardians of Galaxy

참가분야	고등부	참가부문	개인

2016. 07.

지역명) 학 교 명	학 년	성 명
경기) 경기과학고등학교	2	김동규

□ 본 문

가. 개발동기

기존의 벽돌깨기 게임을 물리 원리를 기반으로 한 벽돌깨기 게임 프로그램으로 변형시켜 중력 및 가속도 등의 물리적 원리를 게임을 통해 체험해 볼 수 있도록 프로그램을 개발하였다.
기존의 게임에서의 양쪽의 벽을 제거하는 대신, 공의 움직임을 제한하는 요소로 중력을 추가하였고, 공과 보드의 속도와 가속도를 계산해 운동을 구현하고, 마찰을 적용하는 등 여러 가지 실제 물리 현상들을 체험할 수 있게 했다.

나. 프로그램소개

1. 프로그램 소개

벽돌 깨기에 중력을 적용한다면? 벽돌 깨기가 우주 스케일로 커진다면?

'Guardians of Galaxy'는 이러한 생각으로 시작되었다.

아래는 프로그램의 소개를 위해 게임 화면을 캡쳐한 것이다.

먼저 가운데에 지구가 있다. 이 지구 위를 cyan(청록)색 보드가 움직이게 된다. 처음에 보드에는 흰색 공이 올려져 있다. 지구 주변에는 붉은색 벽돌이 있고, 이 벽돌이 일정 시간마다 지구를 향해서 내려오게 된다. 당신은 이제 보드로 공을 잘 튕기면서 벽돌을 깨나가면 된다.

단, 공은 지구의 중력을 받는다

2. 프로그램 동작 설명

오른쪽 또는 왼쪽 방향키를 누르면, 위에서 언급한 것처럼 보드가 가속이 된다. 물론 보드의 최대 속력을 정해두어서 그 이상으로 빨라지는 일은 없다. 그리고 보드는 지구와 닿아 있어 마찰력을 받는다. 방향키를 누르지 않으면, 움직이고 있는 보드는 그 마찰로 인하여 느려지다가 다시 멈추게 된다.

Figure 1 공을 쏘아 올리기 전

Figure 2 공을 쏘아 올린 후

공은 처음 또는 공이 지구에 닿은 직후, 보드 위에 붙어서 움직이게 되며, space 키로 보드에서 지구 중심과 반대 방향으로 쏘아 올릴 수 있다. 만약 보드가 움직이고 있었다면, 공은 보드의 속도가 더해진 상태로 날아가게 된다. 공에는 지구의 중력이 작용하며, 이 때문에 공은 다시 지구 중심 방향으로 떨어지게 된다. 공이 지구에 닿게 되면 라이프가 하나 사라지고, 새로운 공이 보드에 올려지게 된다. 처음 공 개수(라이프)는 6개(5개+보드에 올려져 있는 공)이고, 공을 한 번 지구에 떨어뜨릴 때마다 남은 공이 하나씩 줄어든다. 당연히 공이 없으면 더 이상 게임이 불가능하므로 게임오버가 된다.

Figure 3 공 개수가 하나 줄어든 모습

Figure 4 남은 공이 하나도 없는 상황. 여기서 공을 또 지구에 떨어뜨리면 게임오버다

벽돌은 일정 주기로 지구 쪽을 향해 떨어진다. 연속적으로 가까워지는 것이 아니라 주기마다 어느 거리만큼 한 번에 떨어지는 것이다. 벽돌이 공과 부딪히면 그 벽돌은 사라진다. 벽돌이 하나라도 지구에 닿게 되면 그대로 게임이 오버다.

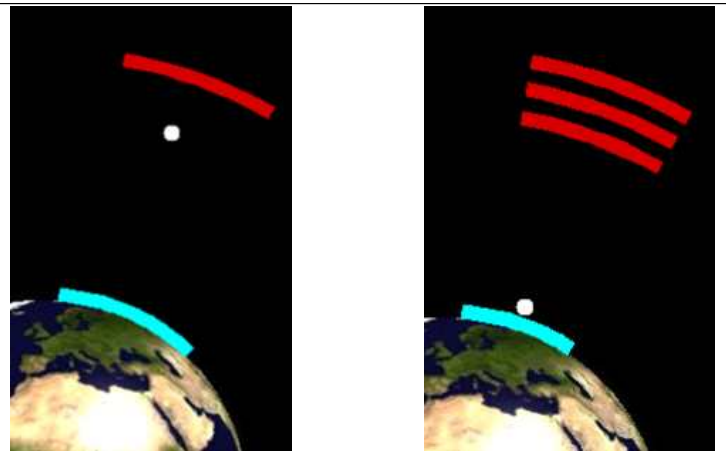


Figure 5 공을 쏘기 전(왼쪽)과 공을 쏘아 벽돌이 부서진 상황(오른쪽)

벽돌은 강도가 있는데, 예를 들어 강도가 3이라고 하면 공으로 3번 맞아야 부서진다. 강도가 큰 벽돌일수록 진한 색으로 나타냈다. 스테이지마다 일정한 범위 안에서 랜덤으로 강도가 정해진다.

Figure 6 강도가 서로 다른 벽돌들

현재 3개의 아이템이 있다. 공 추가, 보드 길이 연장, 보드 자석 기능 아이템이다. 아이템은 벽돌 하나를 깰 때마다 확률적으로 얻을 수 있다. 처음에는 아이템이 떨어지면 보드로 받아야 적용되게 하려고 했으나 몇몇 베타 테스터들의 요구로 그냥 벽돌이 깨지자마자 적용되도록 했다.

Figure 7 공이 추가됨

공은 벽돌 하나 당 하나씩만 추가될 수 있으며, 벽돌 위치에서 생겨난다. 최대 개수 제한은 없다.

Figure 8 보드 원래 길이

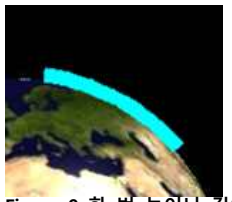


Figure 9 한 번 늘어난 길이

보드는 최대 3번까지 늘어날 수 있다. 한 번 늘어난 길이는 10초 간 유지된다.

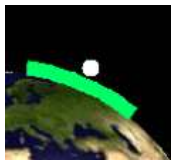


Figure 10 자석 효과가 생긴 보드

공이 보드에 다시 붙게 된다. 스페이스 바를 누르면 다시 발사할 수 있다. 10초 간 유지된다.

3. 화면 구성 설명

Figure 11 메인 화면

실행시키면 게임의 제목과 함께 그 아래에 START라는 버튼이 나온다. 당연히 START 버튼을 누르면 게임이 시작되며 위에서 보여준 게임 화면으로 플레이 하게 된다.

leftpanel은 점수와 남은 공의 개수를 보여준다.

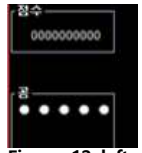


Figure 12 leftpanel

rightpanel은 스테이지, 행성, 행성의 질량, 행성의 반지름 정보를 보여준다.

Figure 13 rightpanel

클리어를 실패하면 다음과 같은 dialog가 뜬다.

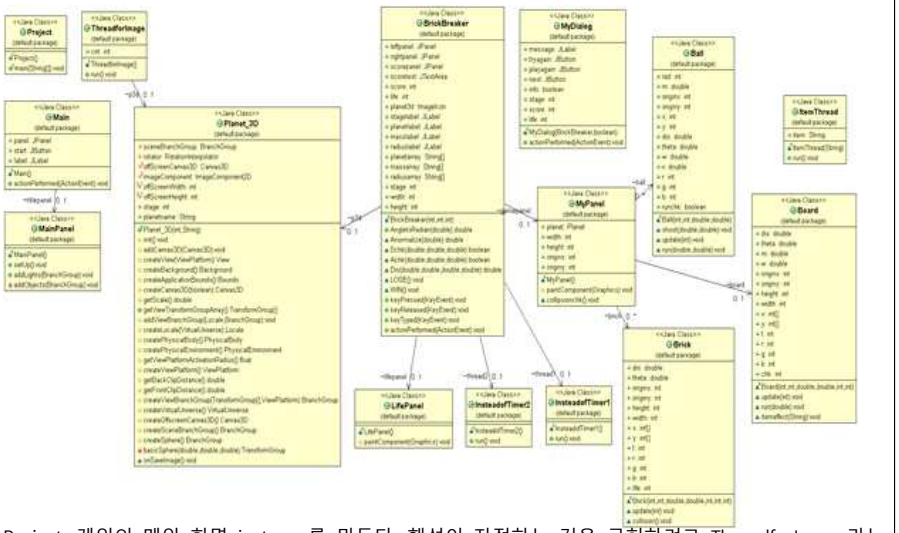
Figure 14 클리어 실패 시 dialog

클리어에 성공하면 다음과 같은 dialog가 뜬다.

Figure 15 클리어 성공 시 dialog

Figure 16 실행 직후 gamepanel(위)과 여러 이벤트가 일어난 후 gamepanel(아래)

4. 소스코드의 class diagram 및 각 class 설명



Project: 게임의 메인 화면 instance를 만든다. 행성이 자전하는 것을 구현하려고 ThreadforImage라는 클래스를 만들어 놓았으나 잘 되지 않아 일단 보류 중이다.

Main: JFrame을 상송한 게임의 메인 화면 class. Java3D로 그려진 게임 제목을 가진 MainPanel과 start 버튼을 가진 JPanel로 이루어진다.

MainPanel: JPanel을 상속한 class로, "Guardians of Galaxy"라는 글자를 Text 3D로 그려준다. Java3D 라이브러리를 추가해야 한다.

BrickBreaker: JFrame을 상속한 class로, 게임의 실행 창을 만드는 class이다.

InsteadofTimer1,2: BrickBreaker에서 쓰던 Timer1,2를 대체하기 위해 만든 Thread class.

LifePanel: JPanel을 상속한 class로, 라이프(남은 공)의 개수를 보여준다. Leftpanel에 포함된다.

MyPanel: JPanel을 상속한 class로, 게임이 진행되는 gamepanel이다. 공, 보드, 벽돌이 그려진다.

Ball: 공 정보를 담는 class.

Board: 보드 정보를 담는 class.

ItemThread: 보드의 연장이나 자석 효과의 적용시간이 끝난 후 효과를 제거해 주기 위한 Thread.

Brick: 벽돌 정보를 담는 class

Planet: 행성 정보를 담는 class.

MyDialog: JDialog를 상속한 class로, 게임의 클리어 실패/성공 여부를 알려주고, 현재 스테이지 다시 시도나 다음 스테이지로 넘어가게 해주는 버튼을 가지고 있다.

Planet_3D: JPanel을 상속한 class로, 지구 지도 이미지를 받아서 구에 texturing한 화면을 이미지로

저장해서 MyPanel의 배경에 사용할 수 있도록 한다. Java3D 라이브러리를 추가해야 한다. (원래는 MyPanel에 직접 그리려 했으나, java3d를 그리려면 Canvas를 사용해야 하는데, Canvas는 heavy weight component이므로 light weight component인 JPanel과 같이 사용할 수 없다. - java에 안 된다고 나와있다 함)

5. Method 설명

-void keyPressed(KeyEvent e): 키보드가 눌리는 event를 받아 처리하는 method. 오른쪽 또는 왼쪽 방향키를 누르면 보드와 아직 발사되지 않은 공에 속도를 더해준다. Space키를 누르면 아직 발사되지 않은 공을 발사한다.

-void collipsionchk(): 공과 벽돌, 공과 보드 사이의 충돌이 일어나는지 확인하고, 충돌 결과를 갱신시킨다. 공과 벽돌이 충돌하면 50점, 벽돌이 사라지면 점수에 100이 더해지고 일정 확률로 아이템 효과가 적용된다. 공은 진행방향이 바뀐다. 공과 보드가 충돌하면 공은 보드의 속도의 일부가 더해진 체로 운동하게 된다. 그리고 공이 지구에 닿으면 라이프를 하나 줄인다.

-void paintComponent(Graphics g): gamepanel을 그리는 method이다. Planet_3D class에 의해 만들어진 지구 이미지를 배경으로 하고, 벽돌과 보드는 구부러진 직사각형(부채꼴의 일부)로 그리고, 공은 원으로 그린다.

-void run(): Ball class(Ball.run(double,double))와 Board class(Board.run(double)) 모두 있는 method이다. 공과 보드에 가속도를 주고, 공과 보드의 위치를 갱신해준다.

-void update(int): Ball, Board, Brick class 모두 있는 method 다. run()과 비슷하게 위치나 속도만 갱신해주지만 편의를 위해 만든 것으로 약간 차이는 있다.

-void LOSE(): 게임 패배 시 호출되는 method이다. 타이머 1,2를 모두 정지시키고 패배를 알리는 MyDialog instance를 만든다.

-void WIN(): 게임 승리 시 호출되는 method이다. 타이머 1,2를 모두 정지시키고 승리를 알리는 MyDialog instance를 만든다. LOSE()와 WIN()은 하나로 합쳐도 될 듯하다.

-boolean Achk(double,double,double): 각도 세 개를 받아, 가운데 각도가 왼쪽 각도와 오른쪽 각도 사이에 있는지를 확인해 결과를 return하는 method이다.

-double Dis(double,double,double,double): 복소 좌표계에서의 두 위치를 받아 제 2 cosine법칙을 이용하여 두 위치 사이의 거리를 계산해 return하는 method이다.

-actionPerformed: 메인 화면과 MyDialog에서 버튼을 누를 때 이벤트를 실행하도록 하는 method이다. 원래는 BrickBreaker에도 Timer 때문에 있었으나, Timer가 Thread로 대체되면서 사라졌다.

Planet_3D, MainPanel의 method는 너무 복잡하므로 생략.

6. 수학 과학적 원리

-공, 보드, 벽돌의 위치를 표현하는데 복소 좌표계를 사용했다.

-복소 좌표에서 r 체크와 theta 체크, 거리 체크를 이용하여 공과 보드, 공과 벽돌의 충돌을 직접 구

현했다.

-공은 지구의 중력을 받아 지구의 중심 방향으로 낙하한다.

-보드에 작용하는 지구의 중력과 평형을 이루는 수직항력이 존재해서, 보드는 그로 인한 마찰력을 받는다.

-보드와 공이 r 방향과 theta 방향으로의 속력을 가지게 하고, 키보드를 통한 입력을 통해 가속도를 주는 방식으로 공과 보드의 운동을 표현했다.

-보드와 공이 충돌할 때, 보드와 공 사이의 마찰이 있음을 고려하여 공에 보드의 속도의 일부를 더해서 운동하도록 했다.

-공기에 의한 마찰을 실제 계산해 적용하진 않았지만, 그 효과를 고려하여 보드와 공이 어느 정도 이상으로는 빨리 움직일 수 없도록 종단속도를 정해놓았다.

-(사실 rightpanel을 채우기 위한 용도였긴 하지만) rightpanel에서 현재 스테이지의 행성 이름, 질량, 그리고 반지름 값을 보여준다.

7. 차별화되는 점

고전 게임인 벽돌 깨기를 변형시킨 것이지만, 사실 기존 게임과는 공으로 벽돌을 깬다는 점이나

플레이 방법 정도를 제외하고는 거의 모든 것이 다르다고 볼 수 있다. 가장 차별화 되는 점은, 기

존의 게임에서의 갇힌 구간이 아닌, 시작과 끝이 없는 원에서 공과 보드가 움직이고, 벽돌이 배치된다는 점일 것이다. 양쪽 벽이 없기 때문에 벽에 공이 튕기는 경우는 아예 존재하질 않는다. 그리고 지구를 배경으로 하기 때문에 공에 중력 현상이 적용된다. 사실상 배경이 지구인 것보다는 공의 움직임을 제한할 벽이 없다는 점에서, 그 역할을 대체할 방법으로 지구의 중력을 사용한 것이다. 그 외에도 기존의 벽돌 깨기에서 물리 법칙을 무시하는 모든 운동들에 다시 물리 현상을 적용시켜 구현했다. 그리고 벽돌이 조금씩 가까워진다는 점도 별로 중요해 보이진 않지만 다른 점은 다른 점일 것이다.

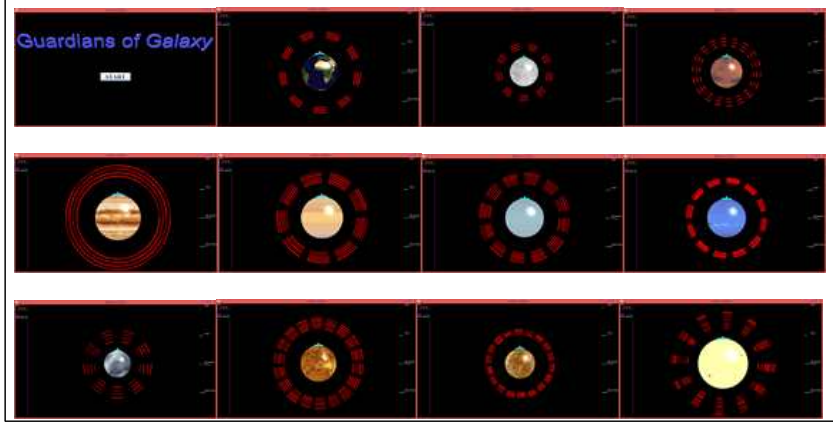
8. 추가

-물리 현상 – 충돌, 가속, 중력, 마찰력 등을 직접 구현했다.

-Planet_3D에서 Canvas3D에 Universe를 이용하여 지구를 그렸다. createSphere()로 구를 만든 다음, basicSphere()로 미리 받아놓은 지구 지도 이미지를 구에 texturing한 다음, 마지막으로 onSaveImage()로 지구가 그려진 Canvas3D를 이미지로 저장하는 기나긴 과정을 통해 gamepanel의 배경으로 넣을 이미지를 만들었다. 사실 미리 만들어 두어도 되지만, 나중에 행성을 추가할 때 계속 따로 만들지 않으려고 미리 class로 만들어 두었다.

-Java3D 라이브러리로 게임 제목도 3D로 그렸다. Text 3D는 Textured Sphere를 그리는 것보다 훨씬 간단하다.

9. 각 스테이지 화면



다. 프로그램사용법

-오른쪽 방향키: 누르면 보드가 지구 위를 시계방향으로 돌도록 가속된다.
-왼쪽 방향키: 누르면 보드가 지구 위를 반시계방향으로 돌도록 가속된다.
-Space 키: 공이 보드 위에 올려져 있을 때, 공을 위로 쏘아 올린다.
기존의 벽돌 깨기와 동일하게 이 세 가지 키만을 필요로 한다.

라. 프로그램설치방법

1.Guardians of Galaxy ZIP 파일을 받아 압축을 푼다.
2.그 안의 파일들을 모두 한 폴더 안에 넣는다.
3.JAR 파일을 열면 게임이 바로 실행된다. 이때 JRE(Java Runtime Environment)가 설치되어 있어야 한다. Java 홈페이지에서 다운받을 수 있다.

□ 기 타

가. 작품개발시 참고한 프로그램 및 문헌

책 - Java 3D Programming
Texturing을 위해 참고한 코드 -
http://stackoverflow.com/questions/22517973/light-and- textures-in- java3d
Java3D example - http://www.java3d.org/samples.html

나. 작품심사시 참고사항

☐ 개발일지

가. 개발일지
첨부

☐ 공공정보 활용일지

가. 개발시 참고한 공공정보
http://planetpixelemporium.com/earth.html에서 게임의 스테이지 배경을 만드는데 사용한 태양계 행성, 태양, 명왕성의 지도를 받아 사용했다.

☐ 작품소스 및 실행파일 : 대용량 e-mail로 제출