네트워크 프로젝트 – Break With Your Phone

구성원:

장문창 (20140718, 컴공)

이지현 (20150246, 컴공)

목차

1. 개요

포켓몬 GO와 같이 AR, VR과 관련된 어플리케이션이 다수 출시되었다. 그런데 VR 어플리케이션의 경우 별도의 기기가 필요한 경우가 많다. 특히, 사용자의 손 움직임을 추적하기 위한 기기가 별도로 존재한다. 그래서 이번 프로젝트를 통해 손 움직임을 추적하는데 스마트폰을 사용할 수 있다는 점을 보이고자 한다. 이는 스마트폰의 센서 정보를 소켓 프로그래밍으로 전달하고, 이 정보를 이용하는 어플리케이션을 만들어 본다.

어플리케이션은 간단한 미니게임으로, 다가오는 장애물을 부숴서 점수를 획득하는 게임이다. 상황은 도로 위에 서서 다가오는 자동차와 비행기를 막대로 부수는 것이다. 이 때, 막대는 스마트폰의 센서 정보를 받아와서 실제로 스마트폰을 움직이는 방향대로 움직인다. 사용자는 게임을 진행하기 전에 게임에 스마트폰의 브라우저로 게임의 웹 서버로 접속한 후, 웹 소켓을 이용하여 센서 정보를 전송한다.

2. 개발 환경

사용한 언어: C#, HTML, Javascript

사용한 프레임워크: Unity 5.6.0f3

3. 목표 기능

- ✓ 웹 서버를 열어서 연결을 수행하기 위한 웹 페이지를 보여줄 수 있다.
- ✓ 플레이어는 열린 웹서버에 접속해서 프로그램과 웹 소켓을 통해 연결한다.
- ✓ 웹페이지에서 원점 조정을 버튼을 눌러 스마트폰의 현재 기울기를 인식하고 원점 조정을 한다.
- ✓ 원점 조정이 완료되면 게임 시작 버튼이 활성화된다.
- ✓ 게임 진행 중에는 배경음악이 재생된다.
- ✓ 시작 화면에는 게임 시작할 수 있는 Start 버튼과 프로그램을 종료하는 Exit 버튼이 있다.
- ✓ Start 버튼을 누르면 게임이 시작된다.
- ✓ Exit 버튼을 누르면 프로그램이 종료된다.
- ✓ 게임 시작 전 준비시간을 준다.
- ✓ 게임이 시작되면 자동차와 비행기 형태의 장애물들이 플레이어에게 접근한다.

- ✓ 플레이어는 스마트폰과 동일하게 움직이는 게임상의 막대를 이용해 장애물을 파괴한다.
- ✓ 파괴할 때마다 점수를 얻으며 하나라도 놓치게 되면 그대로 게임오버가 된다.
- ✓ 5점을 얻을 때마다 장애물들이 나오는 빈도와 움직이는 속도가 증가하게 된다.
- ✓ 게임 오버시 재시작 화면이 나타나고, 재시작 버튼을 눌러 메인으로 돌아갈 수 있다.
- ✓ 재시작 화면에서는 최종 점수를 보여준다.

4. 프로그램 구조 및 설계

4.1. Scene Structure

Main Game part에서는 크게 3가지의 Scene으로 구성된다. 첫 번째는 Start Scene, 두번째는 Play Scene, 마지막은 Restart Scene이다. 각각 게임을 플레이하면서 적절한 상황에서 각각의 Scene으로 넘어가서 게임에서 대처한다. Scene에 대한 흐름은 다음 과 같다.

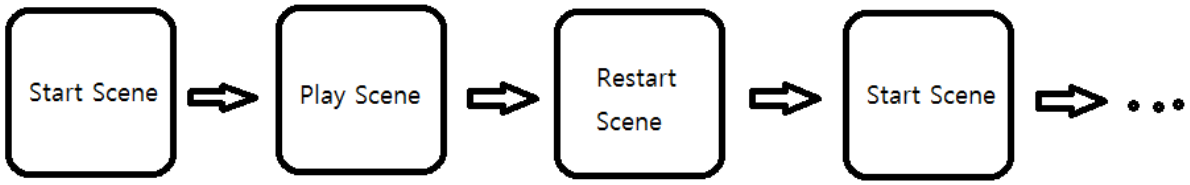


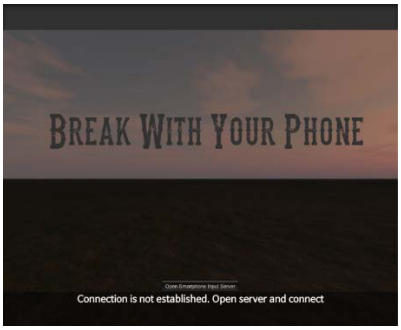
Figure 1

4.2. Start Scene

4.2.1. 동작

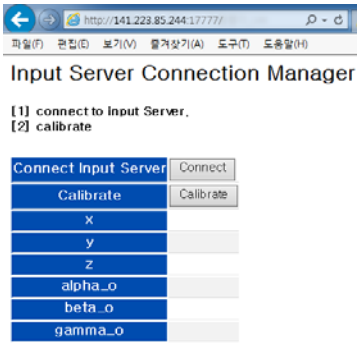
Start Scene은 게임하기 직전 볼 수 있는 단계이다. 게임의 제목과 간단한 배경을 볼 수 있다. 여기서 Web socket 통신을 연결 하여 스마트폰과 프로그램사이에 통신을 한다. 그리고 Start 버튼을 누르게 되면 Play Scene으로 이동하게 된다.

4.2.2. UI & 진행



프로그램 시작시 나타나는 화면이다. 가운데 있는 버튼을 눌러서 웹 서버를 열

수 있다. 웹 서버를 열면 아래의 설명에서 특정 주소로 접속하라는 안내가 나타난다.



웹 사이트를 접속하면 위와 같은 화면이 나타난다. Connect를 눌러 웹 소켓 통신을 시작한다. 소켓 통신이 활성화되면 아래와 같이 calibration 수행을 안내하는 그림이 나타난다.

Calibraiton을 진행하고 나면 아래와 같이 시작 버튼이 활성화 된다.

4.3. Play Scene

4.3.1. 동작

Play Scene에서 본격적인 게임을 할 수 있다. Start Scene으로부터 넘어온 직후 게임을 시작하기전에 약간 멈추어 Player가 게임을 준비할 수 있는 시간을 준다. Play

Scene은 물체가 다닐 수 있는 Road와 장애물인 Obstacle과 Obstacle을 생성하는 Obstacle Spawn과 유저가 직접 조정하는 몽둥이 Stick, 총 4개로 이루어져 있다. 해당 구조도는 다음과 같다.

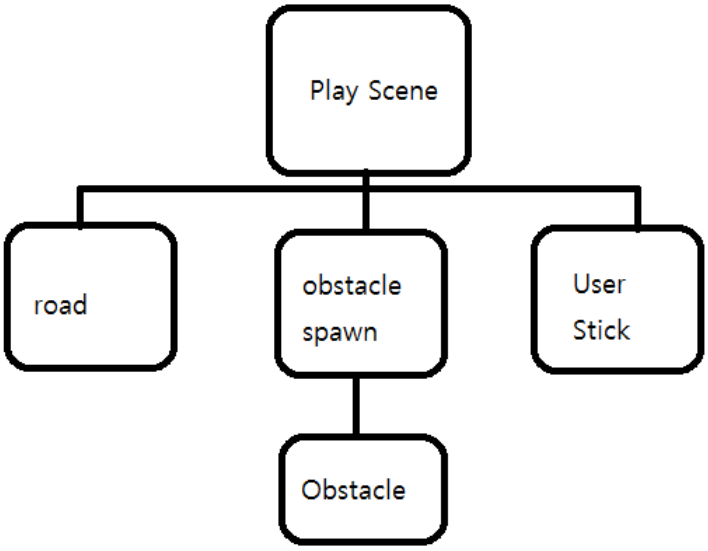


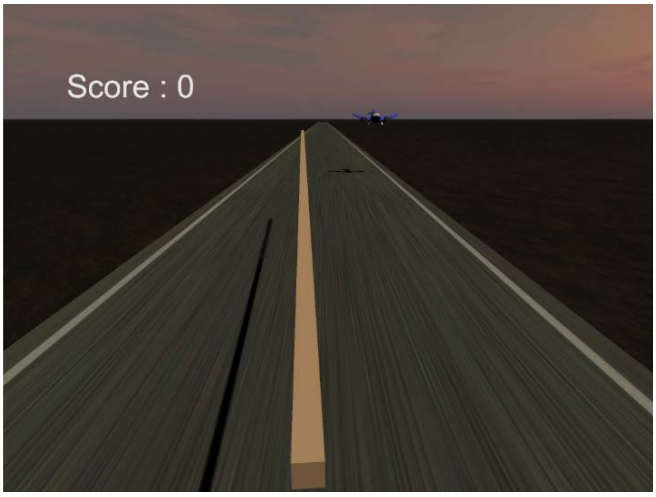
Figure 2

Road는 실제로 장애물이 내려올 수 있는 배경을 제공한다. 따라서 장애물이 내려올 수 있는 범위가 Road의 범위를 벗어나지 않는다.

Obstacle Spawn은 빠르게 내려오는 Obstacle을 Road 위의 x축 좌표 중 Random 한 값을, 고정된 y축 좌표, z축 좌표 생성한다. Obstacle은 생성된 경우 빠르게 유저로 있는 방향으로 다가온다. 유저의 Stick에 충돌 시 destroy되며, 유저가 치는 것을 실패해 화면 밖으로 사라졌을 때도 destroy 된다. 화면 바깥에서 사라지는 경우 유저가 장애물 격파에 실패 한 것이기 때문에 게임오버로 처리된다. 게임 오버 시 Restart Scene으로 이동한다.

Stick은 처음 초기위치에서 유저의 스마트폰으로부터 받아온 센서 정보를 통해 움직일 수 있다. Stick이 움직여 Obstacle과 부딪히게 되면 충돌 이벤트로 해당 Obstacle을 destroy하고 사용자의 점수를 올린다. 사용자가 5점을 얻을 때마다 생성되는 Obstacle의 수와 이동속도를 늘려서 난이도를 조정한다.

4.3.2. UI & 진행



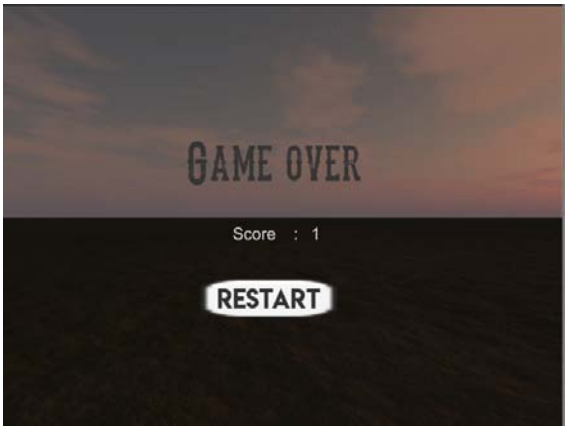
Player가 Obstacle을 격파에 성공할 시 올라가는 score는 좌측상단에 표시되어 있다. 그리고 플레이어가 조작하는 막대와 다가오는 장애물은 그림에 나타난 것처럼 나타난다.

4.4. Restart Scene

4.4.1. 동작

Restart Scene에 도달하게 되면 Play Scene에서 얻었던 점수를 표기한다. 또 화면에 Game over라는 글씨가 있으며 Restart라는 버튼이 존재한다. Player가 이 버튼을 누르게 되면 Start Scene으로 넘어간다.

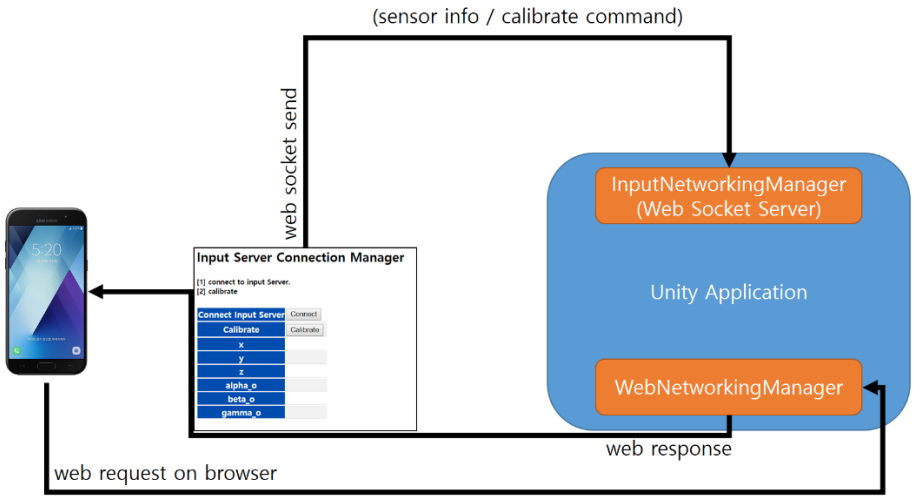
4.4.2. UI & 진행



게임이 끝나면 위와 같은 화면이 나타나며, restart를 눌러 start_scene으로 돌아갈 수 있다.

4.5. Web socket part

4.5.1. 전체적인 구조



네트워크 통신이 진행되는 전체적인 모습은 위 그림과 같다. 스마트폰에서 웹 브라우저를 통해 웹 페이지를 열고, 이후 웹 페이지를 client로 이용하여 web socket server와 통신하는 구조로 되어있다. 따라서 Unity Application에서는 Web Server와 Web Socket Server 두 가지 종류의 서버를 관리하게 된다.

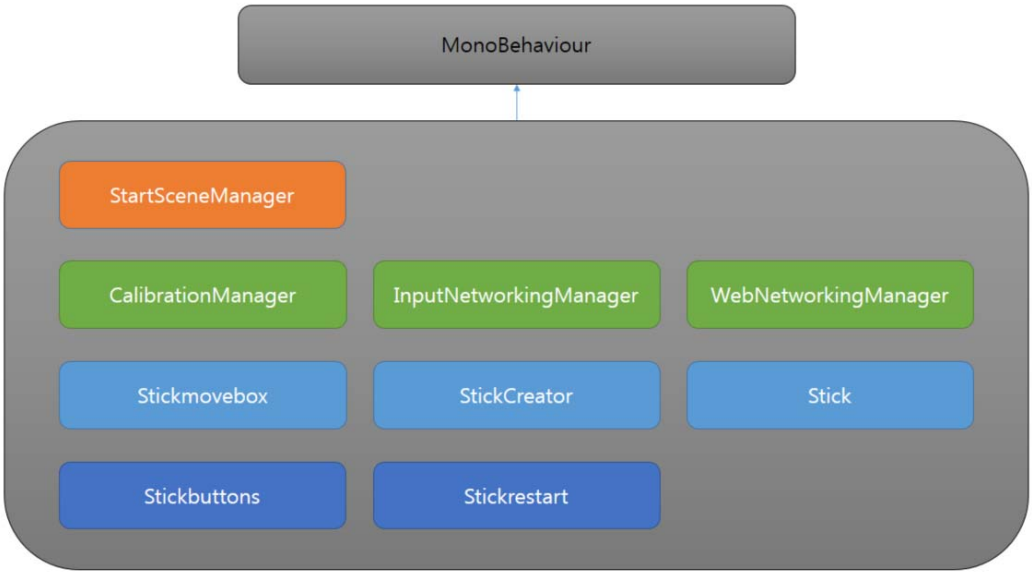
4.5.2. 동작

먼저, Unity Application에서 웹 서버를 연 후, 스마트폰에서 웹 브라우저를 통해 web request를 전송한다. 이어서 WebNetworkingManager에서 InputNetworkingManager와 웹 소켓 통신을 할 수 있는 웹 페이지를 response로 전송한다. 그 다음, 스마트폰에 나타난 웹 페이지에서 connect를 눌러 InputNetworkingManager와 Web Socket 연결을 수행한다. 그리고 calibrate 버튼을 눌러 calibrate가 수행되었음을 전송하고, 지속적으로 센서 정보를 InputNetworkingManager로 Web Socket을 통해 전송한다.

4.5.3. 관련 클래스

InputNetworkingManager와 WebNetworkingManager 클래스에서 각각 Web Socket, Web 서버를 관리한다. (Calibration 정보 저장을 위해 CalibrationManager를 별도로 두고 있지만, status만 저장할 뿐 네트워크 파트와는 독립되어 있다.)

4.6. Class Structure



게임 프로그램에 작성된 class의 전체구조도는 위와 같다. Unity 프레임워크를 이용한 이상 component 로 이용되는 클래스는 MonoBehaviour 클래스의 하위 클래스가 되므로 위와 같이 나타난다. (component로 이용되는 class 외에는 이용하지 않았다) 그리고 위 그림에서 클래스를 기능에 따라 4가지 색으로 분류하였다.

StartSceneManager는 네트워크 연결 여부, 원점 조정 여부에 따라 시작 화면의 버튼을 활성/비활성 시키는 역할을 수행하며, 배경음악이 scene이 전환 되어도 유지되도록 관리한다.

CalibrationManager는 calibration이 필요한 경우 설명을 보여주고, calibration 진행 정보를 저장한다.

InputNetworkingManager는 WebSocket 통신을 수행하여 센서 정보를 받아오고 저장한다.

WebNetworkingManager는 웹 요청을 받아서 response를 전달해주는 역할을 한다. 또한, 웹서버 활성 버튼과 설명 등 관련 UI를 관리한다.

Stickmovebox는 장애물의 기본 속도를 저장하고 있고, 장애물을 움직인다. 플레이어가 장애물을 부수지 못한 경우 restart_scene으로 전환시키는 역할을 한다.

StickCreator는 장애물을 생성시키고 점수에 따라 장애물의 속도와 생성 빈도를 조정한다. 또한, Score를 화면에 나타낸다.

Stick은 센서 정보를 바탕으로 막대를 움직이는 역할을 하며, 장애물과 부딪힌 경우 장애물을 부수고 점수를 증가시킨다.

Stickbuttons는 시작 화면과 재시작 화면에서 사용되는 버튼을 눌렀을 때 scene 전환을 수행한다.

Stickrestart는 restart scene에서 최종 스코어를 표시한다.

5. 프로그램 작성 결과

목표 기능을 구현하였으며, 정상 동작함을 확인하였다. 자세한 내용은 프로그램 구조 및 설계 중 UI & 진행 부분에 서술되어있다.

프로그램 작동시 주의할 점은, 프로그램이 네트워크 통신을 사용하기 때문에 방화벽 설정이 필요할 수 있다는 점이다. 방화벽에서 해당 프로그램을 예외 처리하고, 17777 및 17778 포트에 대한 통신을 허용하면 된다. 또한, 프로그램에서 서버를 열기 때문에 NAT를 이용하는 경우 (공유기를 통해 사설 아이피를 이용하고 있는 경우) 서버를 열 수 없어서 프로그램이 정상 동작하지 않는다.

6. 프로그램 빌드

이 프로젝트는 Unity 5.6.0f3으로 작성 되었기 때문에 빌드를 하기 위해서는 Unity를 설치해야 한다. Unity를 이용해 프로젝트를 연 후 빌드를 수행한 후, 빌드 된 위치에 Assets 폴더를 만들고 프로젝트 폴더의 Assets 내부에 있는 responseBody.html을 복사해 넣으면 프로그램이 정상적으로 동작한다. (responseBody.html은 File I/O를 이용하기 때문에 빌드와 별도로 복사가 필요하다)

7. 고찰

스마트폰의 자이로센서 정보를 바탕으로 이와 동일하게 방향을 가지는 막대를 표현하는 것이 구현하기 쉽지 않았다. 그래서 필요한 회전축 중 두 가지만 사용하여 구현하였으며, 이에 따라 진행 중에 스마트폰을 반대로 향하거나 뒤집는 경우 제대로 방향이 표현되지 않을 수 있다. 조금 더 센서 정보를 제대로 활용하는 계산법이 필요하다. 그 외에는 실제 움직임처럼 잘 표현된다.

또한, 가속도 센서를 이용하여 실제로 스마트폰의 움직임을 트래킹하고자 했으나 가속도 센서를 받아올 때 웹 소켓 통신을 이용하기도 하고, 센서 자체의 가속도가 완전히 정확한 것이 아니기 때문에 가속도를 이용하여 움직임에 반영하는 경우 오차가 지속적으로 발생한다. 이에 따라 막대는 의도한 대로 움직이지 않았다. 실제로 VR 기기에서도 가속도 센서만을 이용해서는 손에 쥐는 기기의 센서만을 이용하여 위치를 트래킹할 수 없기 때문에, 별도의 장치를 이용한다. 따라서 스마트폰의 움직임 트래킹은 포기하고, 가속도가 발생하는 방향으로 약간의 움직임이 일어난 후 다시 제자리로 돌아오게 하였다.