[ 게임 엔진 응용프로그래밍 ]

**VR 게임 이해 및 개발환경 구축**

인하대학교 미래인재개발원 문화콘텐츠프로그래밍

김예슬

**1. VR 게임 역 기획**

|  |  |
| --- | --- |
| Beat Saber logo.jpg | |
| 개발자 | 비트 게임 |
| 퍼블리셔 | 비트 게임 |
| 디자이너 | * Ján Ilavský * Vladimír Hrinčár * Peter Hrinčár |
| 아티스트 | Jan Ilavský |
| 작곡가 | Jaroslav Beck |
| 엔진 | Unity |
| 플랫폼 | 플레이스테이션 4, 윈도우, 오큘러스 퀘스트 |
| 발매일 | 2019. 05. 21 |
| 장르 | 리듬 |
| 모드 | 싱글 플레이어, 멀티 플레이어 |

두 손에 컨트롤러를 들고, 음악에 맞추어 날아오는 장애물을 타격하는 게임

컨트롤러는 화면 상에서 광선 검의 형태로 보여진다

**2. VR 게임 개발 프로그램 소스**

/// OSVR-Unity Connection

///

/// http://sensics.com/osvr

///

/// <copyright>

/// Copyright 2014 Sensics, Inc.

///

/// Licensed under the Apache License, Version 2.0 (the "License");

/// you may not use this file except in compliance with the License.

/// You may obtain a copy of the License at

///

/// http://www.apache.org/licenses/LICENSE-2.0

///

/// Unless required by applicable law or agreed to in writing, software

/// distributed under the License is distributed on an "AS IS" BASIS,

/// WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.

/// See the License for the specific language governing permissions and

/// limitations under the License.

/// </copyright>

using UnityEngine;

using System.Collections;

public class SampleEyeTracker3D : OSVR.Unity.RequiresEyeTracker3DInterface

{

// Update is called once per frame

void Update()

{

var state = this.Interface.GetState().Value;

if (state.BasePointValid)

{

this.transform.localPosition = state.BasePoint;

}

else

{

this.transform.localPosition = new Vector3(0f, 0f, 0f);

}

if (state.DirectionValid)

{

this.transform.localRotation = Quaternion.FromToRotation(Vector3.up, state.Direction);

}

else

{

this.transform.localRotation = Quaternion.identity;

}

}

}

using System.Collections;

using System.Collections.Generic;

using UnityEngine;

// This class manages arrays of gameobjects representing left and right hands

// Keypresses and GUI buttons control cycling through models

public class SampleHandsController : MonoBehaviour {

public TrackedObjectConfidenceManager leftHandObjectsManager;

public TrackedObjectConfidenceManager rightHandObjectsManager;

void Update () {

if(Input.GetKeyDown(KeyCode.RightArrow))

{

leftHandObjectsManager.SetCurrentIndex(leftHandObjectsManager.CurrentIndex + 1);

rightHandObjectsManager.SetCurrentIndex(rightHandObjectsManager.CurrentIndex + 1);

}

if (Input.GetKeyDown(KeyCode.LeftArrow))

{

leftHandObjectsManager.SetCurrentIndex(leftHandObjectsManager.CurrentIndex - 1);

rightHandObjectsManager.SetCurrentIndex(rightHandObjectsManager.CurrentIndex - 1);

}

}

void OnGUI()

{

if (GUILayout.Button("Next Hands (right arrow)"))

{

leftHandObjectsManager.SetCurrentIndex(leftHandObjectsManager.CurrentIndex + 1);

rightHandObjectsManager.SetCurrentIndex(rightHandObjectsManager.CurrentIndex + 1);

}

if (GUILayout.Button("Prev Hands (left arrow)"))

{

leftHandObjectsManager.SetCurrentIndex(leftHandObjectsManager.CurrentIndex - 1);

rightHandObjectsManager.SetCurrentIndex(rightHandObjectsManager.CurrentIndex - 1);

}

}

}