Research on virtual reality interactive design and application monitor SDK tool design based on Unity 3D engine

Erkun ZHANG[1] and Derong DENG[1]

[1]WHY Engineering Consultant Co. Ltd.

December 19, 2022

Abstract

This paper studies the current application and monitoring situation of VR technology. Based on the Unity 3D game engine and VR device HTC Vive, the paper establishes a typical VR interaction scene and propose a VR application runtime monitoring scheme. The main work of this paper includes: (1) VR interactive scene where full-angle observation, movement of the person's perspective, object touch, object picking, object use and menu interaction are achieved; (2) VR monitoring SDK. The monitoring SDK collects the user device information, runtime performance information, exception information, and user behavior information for the VR application under the user environment, by mounting a monitoring script in the VR application scene; (3) Data analysis center, which is responsible for receiving and storing the data reported by the VR monitoring SDK and helps helps the developer to better understand the situation on operation of the VR application and analyze user behavior through statistics aggregation and algorithm analysis of the data.

Research on virtual reality interactive design and application monitor SDK tool design based on Unity 3D engine

Zhang Erkun, Deng Derong

(*Shenzhen Branch, WHY Engineering Consultant Co. Ltd., Shenzhen Guangdong 510000, China*)

Abstract: This paper studies the current application and monitoring situation of VR technology. Based on the Unity 3D game engine and VR device HTC Vive, the paper establishes a typical VR interaction scene and propose a VR application runtime monitoring scheme. The main work of this paper includes: (1) VR interactive scene. The VR interaction scene has general interactive behavior functions, including full-angle observation, movement of the person's perspective, object touch, object picking, object use and menu interaction; (2) VR monitoring SDK. The monitoring SDK collects the user device information, runtime performance information, exception information, and user behavior information for the VR application under the user environment, by mounting a monitoring script in the VR application scene, and then reports the data to the data analysis center; (3) Data analysis center. The data analysis center is responsible for receiving and storing the data reported by the VR monitoring SDK, and through statistics aggregation and algorithm analysis of the data, generating a monitoring report in the form of a visualized chart for developers, which helps the developer to better understand the situation on operation of the VR application and analyze user behavior; (4) Case study. A VR interaction scene is selected as a test case to introduce the process of VR application runtime monitoring scheme in detail as well as loading the SDK script, analyze the data monitoring report from the data analysis center, and verify the effectiveness of the scheme.

Key words: Virtual Machine; Application monitoring; Unity3D; HTC Vive

Introduction

Virtual Reality[1], also known as VR, is a technique to simulate a real world with comprehensive application of man-machine interact, multimedia, computer graphics etc.[2] VR technique is immersive, interactive and imaginative.[3] Immersive, indicates that virtual environment stimulate human in vision, hearing, smell, taste, touch; interactive means users have access to control and receive feedback; imaginative means users can interpret the future based on immersion and interaction. However, VR applications requires high performance on computer hardware, while actual hardware performance in user devices varies from each other. Stuck, decline in frame rate, or even crash may occur under certain circumstances, which hardly raise the awareness of developers if real-time monitoring on VR application is not realized, resulting in poor user experience and loss of potential customers. Man-machine interact in VR application is significantly different from traditional application, mainly in terms of way of observe and control:[4]

Way of observe. Traditional scene is realized via a display with fixed dimension. Users observe the scene via camera either in first person view (point of view, or POV) or third person view (God view). VR application on the contrary has no limitation from display. Users can observe the scene freely through vision. Although there's still the concept of camera in VR, the limitation of observation resulted from camera has significantly reduced.

Way of control. Keyboards, mouse and touch screen are typical means of input in traditional man-machine interact, which is not practical in VR man-machine interact since the vision is filled with virtual world, and users have no eye contact with external input devices. It is common way that users observe the virtual world from headset equipment, and control by joystick.

In addition, thanks to the immersion of VR application, a more vivid observation and more diverse interaction can be established. As a result, analyzing user manner acts as a significant role in excavating user demand as well as improving user experience. Furthermore, flaws in VR application can be discovered and corrected by evaluating user manner, improving application quality, strengthening user stickiness and growing overall profit.[5]

Monitoring and testing on VR application receive fewer awareness compared with VR technique itself for the time being. Major performance monitoring platform in China (i.e. WeTest, Umeng, Toushibao) mainly focus on mobile game application monitoring based on Unity 3D engine in terms of application performance, user in-app purchase etc., while lack of monitoring on VR applications.[6][7] For VR application monitoring, Ghostline、Retinad VR and FishBowl VR are among the most representative teams for the time being.[8] Ghostline can monitor user interaction, body condition and fluctuation in emotion, making it feasible to discover user manner under VR environment thanks to its quantitative analysis. However, monitoring the application performance and crash issue via Ghostline is not practicable. RetinadVR, on the contrary, collects application data including location, operating environment, application performance, VR device information and so on in forms of a Unity 3D plugin, while lack of awareness of user manner. FishBowlVR applies a real-man approach. A technician is assigned to a developer under actual demand raised by developer, and a comprehensive test report can be generated based on real-man test on application.

To sum up, VR application has already possessed great variety and applied in gaming, medical, tourism etc. thanks to its immersive, interactive and imaginative. It is also promising that VR can cooperate with BIM technology, which is being more and more popular in building service industry. Design of interaction as well as choose of VR device and development tool for VR application are different from each other currently. In addition, it is difficult for developers to establish real-time monitoring on VR application performance in user hardware environment as well as user behavior in interaction due to lack of monitoring tool for VR applications.

The targets of this paper are: a) build up an interactive VR scene based on Unity 3D engine and HTC Vive device, b) design a real-time monitoring SDK tool for VR application, and simultaneously collects data in terms of hardware configuration, performance, crash and user behavior, c) design a data analysis center for

processing data from monitoring SDK as well as generating monitoring report with graphs. At the end, a VR scene will be selected to verify the effectiveness of the SDK tool via carrying out real test.

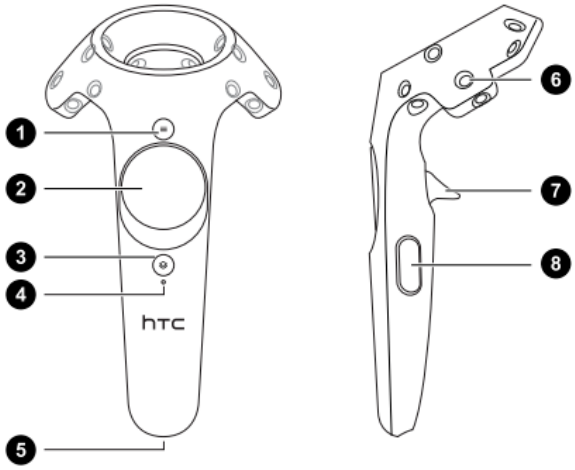
Building up a VR interactive scene

Considered that VR technology is not mature yet for the time being, desktop-based VR can generate a much more vivid world compared with portable device despite its poor portability. VR application has already possessed great variety and applied in gaming, medical, tourism even building construction thanks to its immersive, interactive and imaginative. However, design of interaction as well as choose of VR device and development tool for VR application are different from each other currently, making it difficult to realize real-time monitoring on VR application performance. Desktop-based VR based on Unity 3D engine and HTC Vive device are applied in this paper for VR interactive scene design and real-time monitor via monitoring SDK.

2.1 Introduction on Virtual Reality Toolkit

Virtual Reality Toolkit, or VRTK, provides the scripts of following features:

1. Event associated with device input and interaction. As an input, HTC Vive provides Touchpad panel, Grip button, Trigger button, Application Menu button and System button, shown as Fig.1. Each event is activated upon user input via pressing the button, where developers can define the content in detail. In addition, man-machine interact including touch, grab and use of object are all linked to an event respectively which is independent from events triggered from input. Such independence can provide developer with more flexibility in development via combining different button events and interaction events.
2. World pointer. World pointer is a visible laser beam launched from joystick designed for aiming at the target in VR application scenes. It is usually used as a method of selecting the target position in user displacement, or confirming the object for interaction in design of VR interactions.
3. User displacement. In VR scenes, user realize displacement either in speed mode or destination mode, where scripts are applied respectively for developers to execute, and extend if necessary.
4. Interact with object via joystick. Interaction can be realized once the scripts allowing for interact are associated with object in VR scene and joystick simultaneously. VRTK provides developers with simple interaction including touch, grab and use an object via joystick.



1 - Application Menu
2 - TouchPad
3 - System
4 - Status Light
5 - USB Charging Adapter
6 - Tracking Sensor
7 - Trigger
8 - Grip

Fig.1 Layout of joystick button

2.2 VR interactive scene design

A typical interactive VR scene is established with funtions of 360vision, user displacement, object interaction and menu interaction, shown as Fig.2.

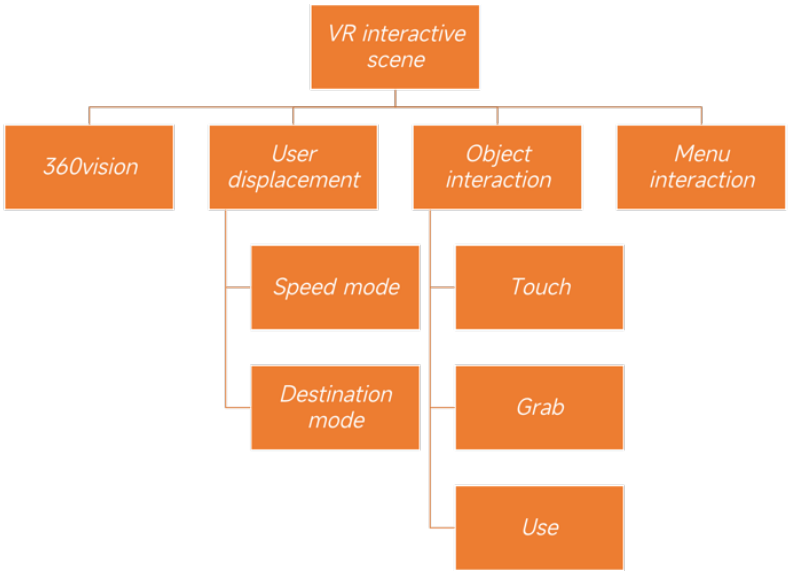


Fig.2 Interactive function of VR scene

360vision indicates that user can observe surrounding environment freely with control input via headsets. A real observe experience which has no limit from orientation can be established compared with traditional applications where vision is limited to camera.

User displacement indicates that user may change position in VR scene freely via joystick either in speed mode or destination mode. In speed mode, position coordinates change at a constant rate once the direction is determined, while in destination mode, coordinates of destination is selected via world pointer where Bézier curve is applied, and coordinate of current position is replaced by destination coordinate.

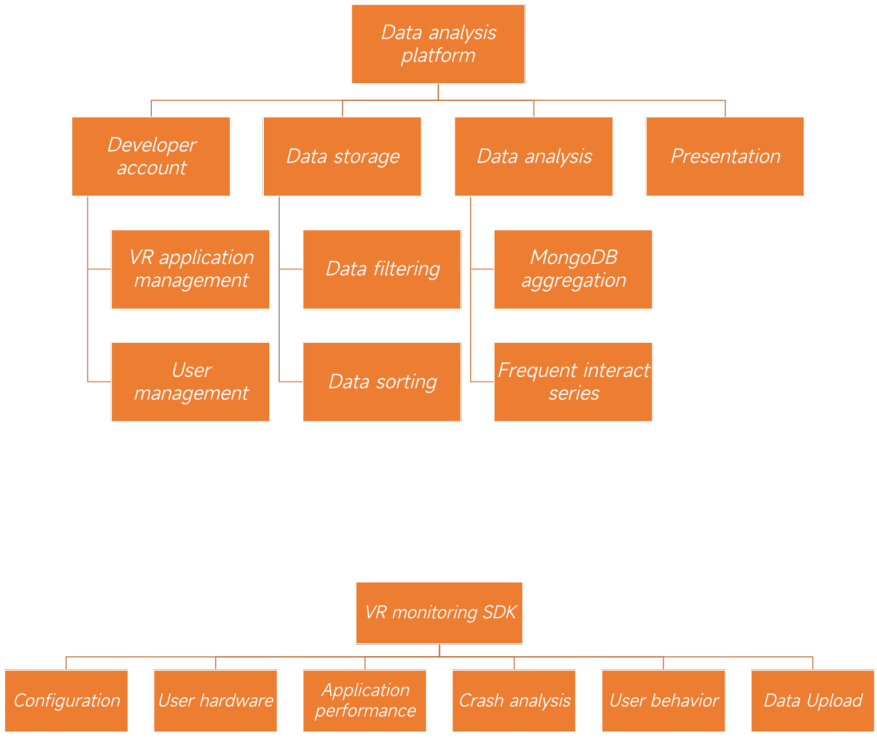
Object interaction supports interaction with object via joystick in forms of touch, grab and use. Once touched on an object in VR scene, reaction will be presented (i.e. effect of deformation) and feedback will be generated in forms of vibration on joystick which stimulates human sense of touch so as to simulate touching on an object in real world. Grab is established via creating a fixed joint associated with joystick and synchronize movement with joystick. Function of using an object is triggered by joystick input such as pressing a button, and the object may change in behavior which is determined by definition on object (i.e. light turning on or off).

Menu interaction indicates that a 2D menu can be called out upon input from joystick which enables interaction via world pointer laser beam. A 2D virtual keyboard is introduced in this VR project, users can 'press' a key by controlling the world pointer for typing text into text frame.

Considering the need of interaction, two joysticks is applied in this VR application which is handled by left hand and right hand respectively. Left joystick is responsible for user displacement, while right joystick is responsible for man-machine interaction including object interaction and menu interaction.

2.3 Establishment of a VR scene

The VR interactive scene in this paper is realized by HTC Vive with Unity 3D engine, SteamVR Plugin and VRTK, where C# programming language is applied to scripts. Models of objects are obtained mainly via Unity Asset Store. Major steps of building up a VR scene can be listed as follows:



1. VR environment configuration. Obtain UnityPackage which consists of SteamVR Plugin and VRTK from Unity Asset Store and import to Unity 3D project.
2. Define startup point. An apartment is introduced in this VR scene. Startup point is selected at living room so that user may start the exploration at the entrance of the apartment.
3. Configure presets. Preset [CameraRig] which consists of Camera(head), Controller(left) and Controller(right) represents user in VR interactive scene. Once imported, size can be adjusted to fit the size of objects. Camera(head), Controller(left) and Controller(right) is linked to headset, left joystick and right joystick respectively. Once [CameraRig] is set, full vision observation and interaction is established. User may start exploration in the VR scene once wearing up the HTC Vive headset and joysticks.

Design of real-time VR monitoring SDK

Due to the lack of monitoring tool currently, performance test on VR applications is often limited to pre-publish state via Profiler in Unity 3D engine, while application performance monitoring in operation in user environment is not practical. To make things worse, actual performance operating in user environment may much differ from performance test result carried out by developer before publish. Unprecedented crash may also occur, resulting in a more dissatisfied user experience. User interaction in VR scenes is more diverse and natural compared with traditional ones, thus it is of great value to monitor user behavior and further excavate demand from users, which developers are exactly looking for. A real-time VR monitoring SDK is built up for collecting and uploading data to analysis platform including user hardware, application performance, crash statistics and user behavior in interacting with VR applications.

The SDK runs as scripts assigned to Camera(head), Controller(left) and Controller(right) respectively, each

script runs at the same pattern as Unity 3D engine and collects data from device. The SDK can be divided into the following modules (as shown in Fig.3):

1. Configuration, which is set prior to the use of SDK. Developers shall register the VR application on test platform and obtain a unique appId which is designed to distinguish each application. Only when the appId is set in SDK can the monitor begin. Calling for public information including device serial number, scene number and scene name is also realized via configuration module.
2. User hardware, which is designed to collect hardware information in user environment via *Awake* method by calling *SystemInfo*function provided by Unity 3D. In *Awake* method, all elements in VR scene are called upon completion of setup. *SystemInfo*collects information of user device serial number, CPU, GPU and operating system so as to assist developers better modify VR applications to achieve better user experience.
3. Application performance, which represents the operation quality of VR applications in user environment. Frame rate as well as Mono stack memory are monitored real-time. Here, frame rate is defined as number of frames rendered by Unity 3D engine in one second[10]. The higher frame rate, the smoother graphic can be presented. Mono stack memory[11]can be expanded dynamically and automatically according to need of system. Such information can reveal whether there's flaw in application performance or not.
4. Crash statistics, which is designed to collect user environment information when crash occurs. Here, crash is defined as unprecedented error occurred in C# script assigned to VR applications. Such error may not lead to malfunction on Unity 3D engine itself, but is likely to result in application error (i.e. unable to interact with object).
5. User behavior, which collects information in interaction including scene information, joystick button information, object interaction, which represents user demand and better assist developers to modify VR applications.
6. Data upload, is responsible for generating JSON file based on collected data and upload to analysis platform via HTTP agreement.

Design of data analysis platform

4.1 MongoDB database

Data analysis platform is designed to receive, analyze and present data from monitoring SDK. MongoDB database is applied in storage of data considering that:

1. Data stored at platform grows with increase of applications registered at monitoring SDK as well as number of users. MongoDB can realize increase of capacity via dynamic mechanism, assuring high efficiency in database operation.
2. Data structure in MongoDB database is collection-oriented, which is made up of KVPs (Key-Value Pair) that saved as a BSON file, which is similar to JSON file. Data obtained from SDK in JSON file format can be directly stored to MongoDB without converting to Java object, reducing time cost.
3. MongoDB supports various types of data including array, which is involved in JSON file from SDK. In addition, format of fields in every line of data is not limited, making it much easier to expand the database since no rebuilt is required on existing data structure.

4.2 Evaluation of Frequent Interact Sequence

Excavation of Frequent Interact Sequence (FIS) is similar to that of Frequent Element Sequence (FES), where difference lies in that time variable is considered in FIS, while not in FES. Both Apriori and FP-Tree algorithms can be applied to excavation of FIS, where the former scans the database back and forth and generates numerous of results, while the latter only scans the database twice throughout the entire process, making it more efficient in operation. FP-Tree algorithm is applied in this study for evaluation of FIS

considering time variable in interaction. Only Maximum Frequent Interact Sequence (MFIS) is considered since any FIS can be regarded as part of MFIS. Major steps can be listed as follows:

1. Define frequency threshold *Minsup* , define MFISTree as empty.
2. Scan user interact record stored in database, and compare whether the frequency is greater than *Minsup* or not. Elements which frequency exceeds *Minsup* are set as header in order of frequency.
3. Scan user interact record in database for another time to generate MFISTree, which process is similar to FP-Tree. Major difference lies in that generation of MFISTree will not change the order of interact record according to order of header. In other words, time order of interact record is maintained in MFISTree.
4. Generate a InvTree according to order of elements in header so as to obtain the FIS which ends with the element.
5. Examine all FISs obtained from InvTree and only store MFISs via KMP algorithm.

4.3 Layout of data analysis platform

Data analysis platform consists of account management, data storage, data analysis and report modules, shown as Fig.4.

Account management module provides developers with developer account register as well as management functions. In addition, application management is also realized, where developers can register application to obtain a unique appId, which is a string generated by UUID and consists of 32 hexadecimal numbers. Only when appId is configured in VR monitoring SDK can data collection on application and upload be carried out.

Data storage module receives and stores data from SDK. Once received the JSON file from SDK, examination on data is carried out to remove illegal or abnormal data to ensure accuracy in following evaluations. Data passed the examination will be stored to MongoDB database.

Data analysis module carries out aggregation statistics on application data stored in MongoDB database, including user device, operation performance, crash report and user interaction. In addition, evaluation of frequent interact sequence (FIS) is also carried out. Since FIS represents how users interact with VR application, developers may realize upgrade on application according to user manner so as to improve user experience.

Data report module generates results from analysis on application data in an intuitive way. Developers can either examine result in Web page or download full report in various formats. Echarts[9] which is an open-source JavaScript-based toolbox developed by Baidu is selected as the tool to generate data report in this study considering of its outstanding compatibility which supports major browsers on desktop as well as mobile devices.

Case study

A sample VR application is selected to verify the effectiveness of VR monitoring SDK and data report.

5.1 Configuration of VR monitoring SDK

VR monitoring SDK as well as data analysis platform are web-based (www.i-test.com.cn) built on Aliyun. Major steps in configuration are as follows:

1. Download EasyMonitor.unitypackage from download page in SDK.
2. Once logged in the SDK with developer account, register the VR application and obtain appId.
3. Import EasyMonitor.unitypackage to Unity 3D project and set appId obtained from SDK in Config, shown as Fig.5.

4. Attach HeadsetMonitor script to Camera(head), attach ControllerMonitor script to both Controller(left) and Controller(right), while distinguish the left handle by selecting *isLeft* property on script.

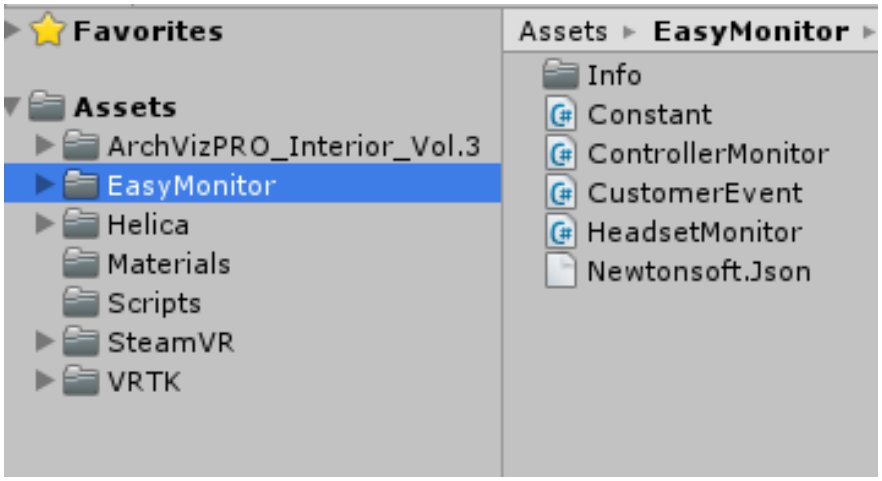


Fig.5 EasyMonitor import

VR monitoring SDK is built into VR application once above steps are all done, and the application may distribute to users after re-compelling. Real-time data collection and upload including user device, operation performance, crash report and user interaction can be realized upon operation in user environment. Data analysis platform receives data from SDK, carry out statistics analysis, and finally generate report for developers. In this study, the VR application is operated in a user environment which is exactly the minimum requirement from the application.

5.2 Results

Real-time monitoring is realized, and reports including use of handle object, 7-day usage, runtime frame rate, crash information are generated, shown as Fig. 6 – 11.

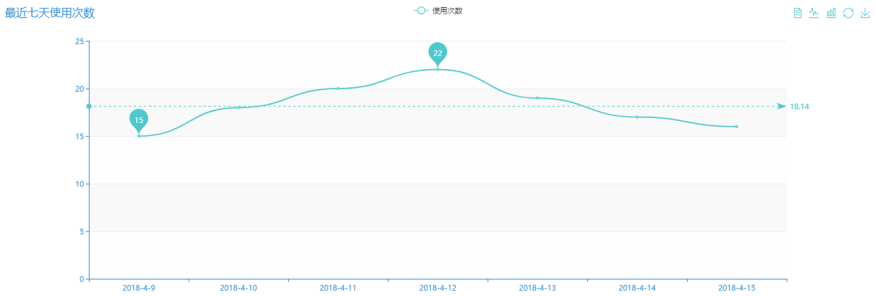
场景0 AVP_Interior_Vol.3 手柄按钮触发统计

手柄按钮名称	平均触发次数
左手柄TouchPad	15.56
左手柄Trigger	6.31
左手柄Grip	8.25
左手柄Application Menu	6.06
右手柄TouchPad	8.81
右手柄Trigger	23.19
右手柄Grip	16.5
右手柄Application Menu	5.44

Fig.6 Handle Object Statistics

崩溃异常名称	堆栈信息
IndexOutOfRangeException: Array index is out of range.	LightControl.StartUsing (UnityEngine.GameObject usingObject) (at Assets/Scripts/LightControl.cs:17) VRTK.VRTK_InteractUse.UseInteractedObject (UnityEngine.GameObject touchedObject) (at Assets/VRTK/Scripts/Interactions/VRTK_InteractUse.cs:313) VRTK.VRTK_InteractUse.AttemptUseObject () (at Assets/VRTK/Scripts/Interactions/VRTK_InteractUse.cs:367) VRTK.VRTK_InteractUse.DoStartUseObject (System.Object sender, ControllerInteractionEventArgs e) (at Assets/VRTK/Scripts/Interactions/VRTK_InteractUse.cs:378) VRTK.VRTK_ControllerEvents.OnTriggerPressed (ControllerInteractionEventArgs e) (at Assets/VRTK/Scripts/Interactions/VRTK_ControllerEvents.cs:498) VRTK.VRTK_ControllerEvents.Update () (at Assets/VRTK/Scripts/Interactions/VRTK_ControllerEvents.cs:1150)
DivideByZeroException: Division by zero	Gun.StartUsing (UnityEngine.GameObject usingObject) (at Assets/Scripts/Gun.cs:22) VRTK.VRTK_InteractUse.UseInteractedObject (UnityEngine.GameObject touchedObject) (at Assets/VRTK/Scripts/Interactions/VRTK_InteractUse.cs:313) VRTK.VRTK_InteractUse.AttemptUseObject () (at Assets/VRTK/Scripts/Interactions/VRTK_InteractUse.cs:367) VRTK.VRTK_InteractUse.DoStartUseObject (System.Object sender, ControllerInteractionEventArgs e) (at Assets/VRTK/Scripts/Interactions/VRTK_InteractUse.cs:378) VRTK.VRTK_ControllerEvents.OnTriggerPressed (ControllerInteractionEventArgs e) (at Assets/VRTK/Scripts/Interactions/VRTK_ControllerEvents.cs:498) VRTK.VRTK_ControllerEvents.Update () (at Assets/VRTK/Scripts/Interactions/VRTK_ControllerEvents.cs:1150)

场景名称	场景序号	场景路径	场景根对象数目	平均使用时长/秒
AVP_Interior_Vol.3	0	Assets/ArchVizPRO_Interior_Vol.3/3D SCENE/AVP_Interior_Vol.3.unity	17	126.66



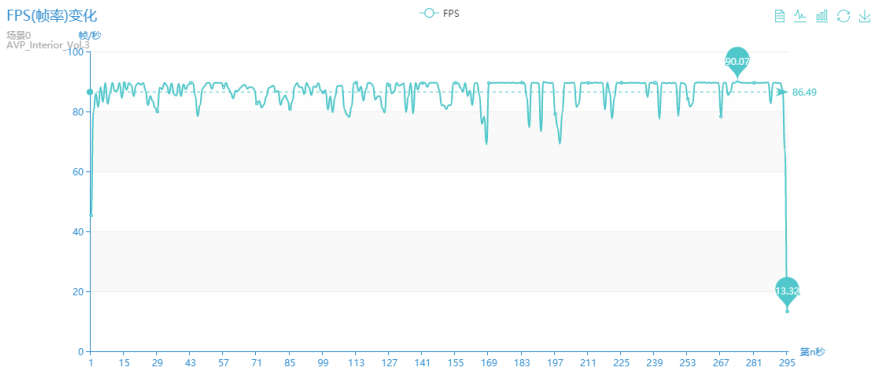
场景0 AVP_Interior_Vol.3 凝视物体统计

物体名称	平均凝视次数
Collision	11.75
Door	9.54
PM-40_Variant1	8.31
Fireplace	7.43
LightSaber	7.24
Outside_Round (1)	6.78
Book1	5.94
Sculpture	5.59
TableLow	4.41
LampFloor	4.06

Fig.7 Sneezing object statistics

请输入频繁度阈值 确定

出现次数	频繁交互序列	场景序号	场景名称
56	touch PM-40_Variant1->grab PM-40_Variant1	0	AVP_Interior_Vol.3
51	touch Picture->grab Picture->touch Picture	0	AVP_Interior_Vol.3
47	use PM-40_Variant1->use PM-40_Variant1->use PM-40_Variant1	0	AVP_Interior_Vol.3
33	touch Book1->grab Book1->touch Book1	0	AVP_Interior_Vol.3
33	touch Book2->touch Book2	0	AVP_Interior_Vol.3



Conclusion

A typical VR interactive scene and a real-time VR application monitoring SDK are established. Test result shows that the VR scene can provide user with high quality experience, and the monitoring SDK can realize real-time monitor on VR applications, making it practical for developers to obtain application performance in operation in user environment as well as understand user behavior in interaction. However, limitation of this study still exists especially in following aspects: a) The study is based on Unity 3D engine and HTC Vive device. Different combination of engine and device (i.e. Unreal engine combined with Oculus Rift device) may obtain better application performance or better compatibility. The monitoring SDK in this study can be more functional by expanding to various combinations of device and engine; b) Only user hardware information, user behavior, crash statistics and performance data are supported and collected in this monitoring SDK. Type of data supported by this SDK can be further expanded to cope with demand from developers (i.e. network delay, memory usage in user environment etc.).

References:

1. Slater M, Sanchez-Vives M V. Enhancing Our Lives with Immersive Virtual Reality[J]. 2016, 3.
2. Yang J. Research on application and development of virtual reality technology[J]. Information & Communications, 2015(01): 138.
3. Zhou Z, Zhou Y, Xiao J. Survey on augmented virtual environment and augmented reality [J]. Scientia Sinica (Informationis), 2015, 45(02): 157-180.
4. Fengjun ZHANG, Guozhong DAI, Xiaolan PENG. A survey on human-computer interaction in virtual reality[J]. SCIENTIA SINICA Informationis ,2016, 46(12): 1711-1736.
5. Bailey J S, Burch M R. Research methods in applied behavior analysis[M]. Routledge, 2017.
6. Wen H. A Virtual Reality System of Chemical Defense Training based on HTC Vive device[D]. South China University of Technology, 2017.
7. Fan Z. Research on Indoor Positioning Technology of Large-scene Immersive Virtual Reality Game[D]. Chongqing University of Posts and Telecommunications, 2016.
8. VR net: A VR application analysis platform dedicated to developers (in Chinese) [EB/OL]. https://www.hiavr.com/news/product/6332.html.
9. Song J, Liu Y, Lin L, Li S, Xu F. Study on Technique of Dynamic Statistical Chart Drawing Based on eCharts[J]. Computer Knowledge and Technology, 2017, 12: 091.
10. Ding Q, Cai S, Chen Y, Yao Z. Brief discussion on simple performance improvement on Unity3D games (in Chinese) [J]. Scientific and Technological Innovation, 2015(31): 173.
11. Project M. Mono project applications, Oxygene, C Sharp, Moonlight, Banshee, CopyBot, Unity, Linden Scripting Language, Pinta, Acando, MonoDevelop, F-Spot, GNOME Do, Beagle, IFolder, Tomboy, IKVM.NET, Libopenmetaverse, Gtk Sharp, Muine, XSP, Easyfind[J]. Mono Project, 2011.