Implementation of the Teaching Experiment system based on Unity3d engine

Guang Hong[1], Fu-ping Deng[2], Dong-yu Zhang[3] and Chao Fen[4]
[1,2,3,4] Department 5, Wuhan Mechanical Engineering Academy, Wuhan, 430075 China

Keywords: Teaching experiment system, Unity3d engine, inversion of control (IoC).

Abstract. Aiming at the shortage of the actual experimental equipment, virtual reality technology was adopted for setting up an experiment system. The framework is the strange IoC and the engine is Unity3D for the system. A step operation of the experiment is divided into five parts, interaction behavior, interaction part, response part, active change and passive change. To meet IoC, interaction behavior is set as finite state machine (FSM) and change is set as model. Then the flow chart of the IoC system is given to implement the virtual experiment. The key technology is model and FSM.Then modeling methods are given for response behavior model and result model. Moreover the flow of implementing FSM is given. Finally an experiment instance shows the system is useful and valuable.

Introduction

Experimental teaching has played an extremely important role in education. However the shortage of experimental equipment has become the weakest link in the teaching course. With the emergence of virtual reality technology, it bring about a new way for experiment teaching. Virtual reality technology has been applied to various kinds experiment course teaching for its good interactivity, operability and cheapness.

The investigation shows that the virtual experimental teaching software gets long-lasting development and plays a major part in different education stages. Virtual experiment has developed for a long time at abroad, the typical system include Learn Anytime Anywhere Physics [1] and Newton [2]. Domestic research is mainly concentrated in universities, such as China University of Science and Technology, Sichuan Normal University, Central China Normal University [3], etc.

To satisfy the simulation training function of simulation software and remote operation, a cross-platform development engine is suitable, which can be applied to network platform, desktop system and handheld devices. Unity3D engine can meet the above platform at the same time, so we eventually choose Unity3D engine.

In this paper we designed a virtual experiment system by Unity3d engine in order to improve the teaching quality of vocational education and cultivate students' learning ability, practical ability and innovative thinking ability.

The system frame structure

Strange is a super-lightweight and highly extensible Inversion-of-Control (IoC[4]) framework, written specifically for C# and Unity. It is shown in figure 1 and includes root(contextView),MVSC Context(Context binds dependencies together),Controller(Commands Inject Models and Services),View(Views and Mediators Talk),Services(Services communicate with the world outside the app),and Models(Models store data in simple value objects).Root instantiates Context, and ContexEvent start kicks off first Command. Mediators dispatch IEvents mapped to Commands, and then Commands dispatch back to Mediators, and Commands act on the APIs of Models and Services. Services may set data on Models. Then Models and Services may dispatch back to Mediators, or Commands may dispatch back on their behalf.

For a virtual experiment system, the most important course is operation steps. One step operation begins from an interaction behavior of the operator to changes both from interaction parts and response parts. Its definition is shown in figure 2.

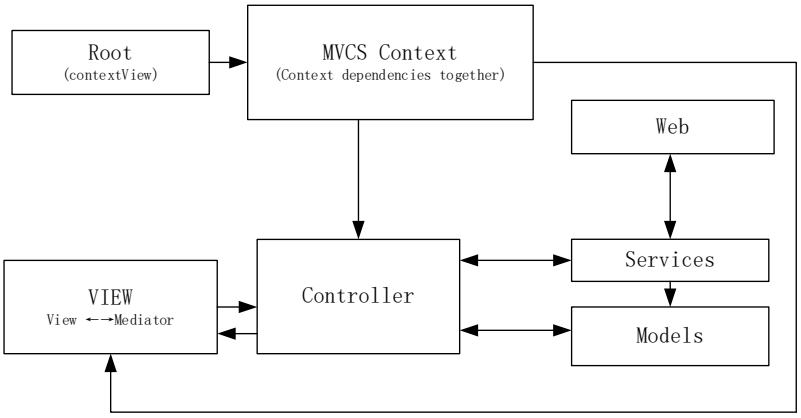

Figure 1.the framework of Strange IoC

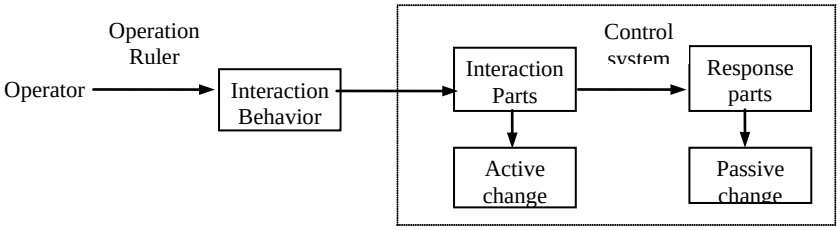

Figure 2 Difinition of one step opeation

To meet IoC, we set Interaction behavior as Finite state machine (FSM).FSM is a mathematic model describing finite state, transition behavior between all the states and all the actions of the states. FSM can be shown by a directional figure, it can be defined as[5]:

$$M=(S, \Sigma, \delta, S_0, S_F) \quad (1)$$

S——a set of the finite state

Σ——a set of all the events

δ——$S\times\Sigma\rightarrow S$, transition function, commonly it defined as $\delta(S_1,a)=S_2$,that means if event a happens, the state will transform S_1 into S_2.

S_0——S_0 is the subset of set S.it is a particular state, namely is initial state.

S_F——S_F is the subset of set S.it is also a particular state, namely is final state.

Every state is a controller.Active change and passive chagne is reusable models.For instance rotation parameter, falling parameter, color change parameter and so on are all models.

The final flow chart of the experiment system is as shown in figure 3.

The entry point to your app is a class called a ContextView, which is simply a MonoBehaviour that instantiates the MVCSContext.

The MVCSContext (technically, a subclass of MVCSContext) is where you set up all your bindings. The core of Strange is a very simple package for binding. This means, essentially, that we can bind (connect) one or more of anything to one or more of anything else. Tie an interface to a class that implements that interface. Or tie an event to a handler. Or tie two classes such that when one comes into being, the other one is automatically created.

The dispatcher is a communication bus, allowing you to send messages throughout your app. The dispatcher used in MVCSContext sends objects called TmEvents. Alternatively, you can follow the steps outlined above to re-wire the Context to use Signals.

Commands are classes triggered by IEvents or Signals. When a Command executes it carries out some part of the application logic.

Models store state.

Views are MonoBehaviours attached to GameObjects: the bits of the game your player actually sees and interacts with.

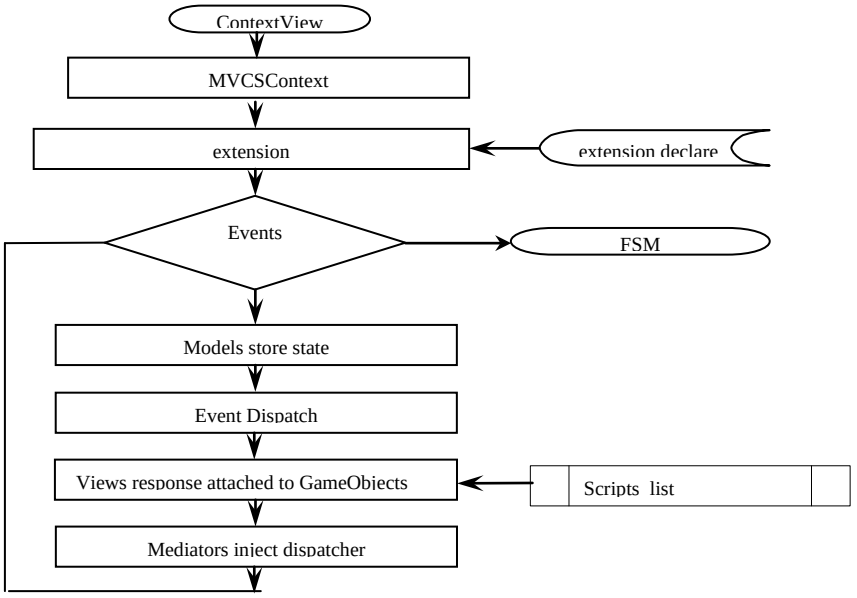

Figure 3 The flow chart of the experiment system

Mediators are also MonoBehaviours, but with the very specific function of insulating the View from the rest of the app.

The implementation of the virtual experiment system

Key technology is models and FSM for the experiment system.We built response behavior model and response result model.

Response behavior model

Response behavior containment to response mode, response initial state, response final state, response speed. Response mode containment liquid drop to fall, the solid sink, the liquid surface transform etc. For example, the liquid drop from the height H_0 fall to the height H_e .Definite the bottom of the container as 0, the drop is The free fall and its acceleration is g, therefore the model of the position H of liquid drop is

$$H(t)=H_0-gt^2/2(0\le t\le t_e) \quad (2)$$

When the drop collides the surface,it stops fall.the span is t_e，

$$H_e=H_0-gt_e^2/2 \quad (3)$$

Then the liquid surface Rises to H_e. Suppose the Sectional area of the container is S and the volume of the drop is V_w， the count of the drop is n，

$$H_e=nV_w/S \quad (4)$$

Now te Solution is

$$t_e^2=2*H_0/g-2*nV_w/(g*S) \quad (5)$$

Response result model

Response result include response result mode, response result the variety beginning start appearance and response result variety be over appearance, response result variety speed, response result appearance condition. Response result mode containment liquid surface ascension and aqua change color, the solid fusing, aqua layering etc. With the aqua changes color for example, the beginning of aqua starts color $RGBA_0=(r_0,g_0,b_0,a_0)$ and ends the color as $RGBA_{end}=(r_{end},g_{end},b_{end},a_{end})$, change the speed as v, change countenance after producing condition containment to in a twinkling change countenance, flutter, water bath heating behind change countenance, flames heating behind change countenance etc.. Means with the vector B, then B=(vcon1,vcon2,vcon3,vcon4）, if need to satisfy a certain condition, that then takes to be worth to 1 and otherwise takes to be worth to 0, if need to satisfy a condition, each condition then takes to be

worth to 1| ns, condition is meant with the vector C at present, C= (con1,con2,con3,con4)^T, if a certain condition attains and then the value is 1.

The FSM Implementation

Firstly initializes the FSM parameter such as states set, events set, transition function, initial state and final state. Then creates a whole cycle, the cycle updates per frame and inquiry if any event in FSM happen. If no event happens, waits. If some event happens, updates the state. If the state is final state then suspend FSM otherwise does action of the state and return. The flow of implementing FSM is shown in fig 4.

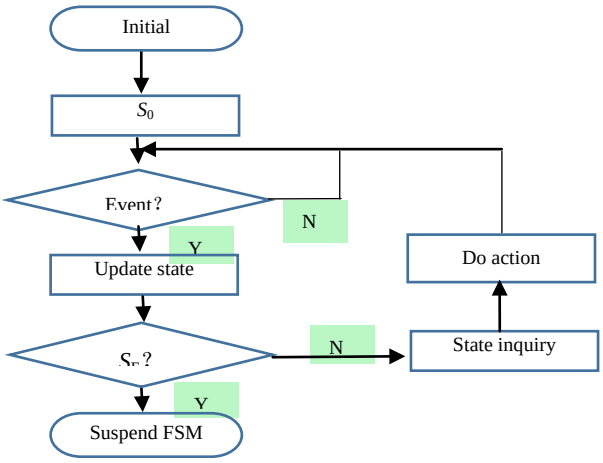

Figure 4 the flow of implementing FSM

Summary

One experiment result is shown in figure 5.

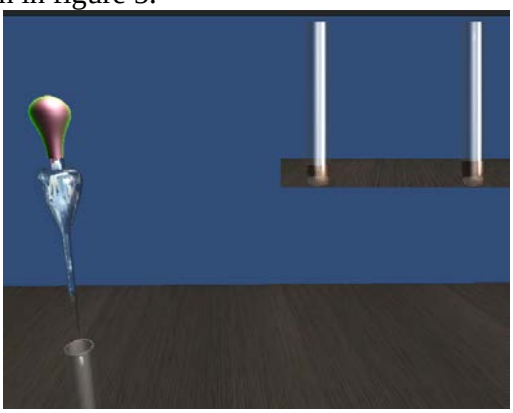

Figure 5 the experiment effect figure

The result shows the framework of the experiment is effective.We use the framework to design six different experiment,which proves that the development is high efficiency.

References

[1] Crinela Dumitrescu, Radu Lucian Olteanu,ect.Using virtual experiments in the teaching process[C],Procedia Social and Behavioral Sciences(2009)

[2] Information on http://www.newtonlab.com/English/newton

[3] Zhu Zhu.Design and Application Research of Virtual Experimental System Based on Uinty3D.Master's thesis of Central China Normal University(2012).

[4] Information on http://strangeioc.github.io/strangeioc/TheBigStrangeHowTo.html.

[5] Guang Hong, Weibing Bai, Xin Yu. A kind of crane loading and unloading simulation based on finite-state Machine. [C] International Conference on Mechatronics, Materials, Chemistry and Computer Engineering (2015).