ISSN 2616-5775 Vol. 2, Issue 2: 1-10, DOI: 10.25236/AJCIS.010032

Design and Implementation of Demonstration System for Underwater Submarine Cruise Based on Hololens

Heng Zhang*, Xiushan Zhang and Zhuo Chen

Department of Computer Science, Naval University of Engineering, Wuhan 430000, China
Corresponding author e-mail: 1979854043@qq.com

ABSTRACT. *In the field of combat simulation, how to build a realistic 3D underwater real-scene demonstration simulation system is always a difficult problem due to the particularity and complexity of the underwater environment. Take the submarine as a background, we propose a design idea to development an underwater demonstration system based on Hololens in this paper, and develop a submarine cruise demonstration system with good effect of 3D underwater real-scene display, which can realize the underwater roaming and fixed-point cruise of submarine. The system is developed by virtual reality engine Unity3D and head-mounted display device Hololens. At first, we use Unity3D to create a 3D underwater scene. Then, we make an interpolation to the fixed-points with position information, draw a trajectory of the planned navigation route, design underwater roaming and adjust the position of submarine in real-time. Finally, we release the system to Hololens for testing through LAN. The operator only needs to wear the helmet and use simple gestures to experience the realistic 3D underwater scene and see the submarine cruise in real-time. The test results show that our system can realize the high-level simulation of the underwater scene and real-time display of submarine situation, which can be used as a demonstration analysis platform for underwater submarine navigation missions.*

KEYWORDS: *Hololens, Unity3D, underwater, submarine cruise, demonstration system*

1. Introduction

In recent years, the status of underwater combat in the informationized sea battlefield has become increasingly prominent. The military value and strategic significance of the underwater battlefield space have been widely recognized. Various marine military powers have promoted and implemented strategies and plans related to underwater warfare, and developed a large number of underwater weapons and equipment[1]. Due to the complexity and particularity of underwater battlefield which is different from other battlefield space, there are many difficulties in the analysis of underwater battlefield situation and the presentation of underwater activities. It is difficult to describe the underwater activity situation accurately and intuitively by using the abstract, single and static description of the situation map, which makes the underwater situation simulation always one of the difficult problems in the field of combat simulation.

In this paper, we take the demonstration of submarine cruise as a background, propose a design idea to development an underwater demonstration system based on Hololens. We model the underwater scene in three dimensions, fit the trajectory and drive the submarine to cruise in real-time. Moreover, by using Hololens helmets, we can realize underwater submarine cruise demonstration by simply interacting with gaze, gestures, voices and so on, without the need for cable links or synchronized computer phones. The operation is more convenient and the interaction is more natural. The system can be used anytime, anywhere, without site restrictions. In a word, operators can experience the realistic 3D underwater scene and see the submarine cruise in real-time with our system conveniently.

2. System design

2.1 Function analysis

- Function of underwater scene display. The underwater environment is diverse and complex. In order to provide realistic 3D underwater scene, it is necessary to add sandstone and aquatic vegetation to the basic terrain, plankton, jellyfish and fish to the underwater space, and light, shadow and bubble effects to the whole scene, so as to make the whole display effect closer to underwater environment.
- Function of drawing submarine route in real-time. In order to demonstrate the submarine cruise, it is necessary to interpolate and draw a trajectory of the planned navigation route according to the position information of multiple fixed target points, drive the submarine to sail according to the planned route, and display the submarine posture information in real-time for auxiliary analysis.
- Function of interaction. The system needs to support underwater roaming and route selection. It can use three interactive modes provided by Hololens: gaze, gesture and voice to adjust the initial position of the submarine, rigger the start, pause and close of cruise demonstration.

2.2 Development process

We design the system development process as shown in Figure 1. It is mainly divided into three stages: resource preparation, project editing and release testing.

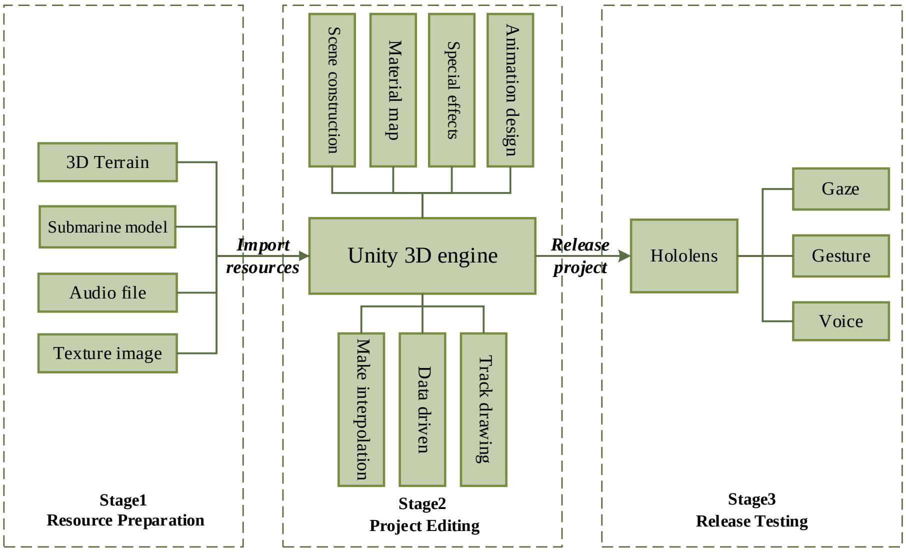


Figure. 1 Flow chat of system development

Stage1: Resource preparation. Using Unity3D's own terrain engine to create 3D underwater terrain. Using 3D modeling software 3DMaxs to create the model of submarine. Collecting texture images and audio resources related to bubbles, light-shadows, waterweeds, fish and so on.

Stage2: Project editing. Import the 3D underwater terrain, submarine model, audio and texture resources into Unity 3D. Using Unity 3D's own physics engine, animation controller and audio controller to design the underwater scene. Combining with C# scripts, the submarine motion is driven and the route is drawn.

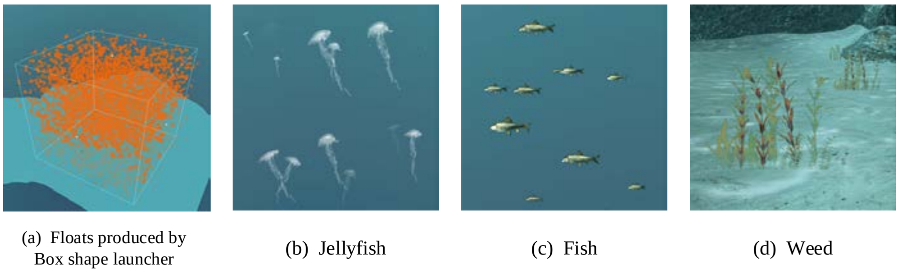
Stage3: Release testing. Import and install the system project built in Unity 3D into Hololens through the LAN. Let operators take Hololens helmet and test the submarine cruise demonstration system through gaze, gesture and voice interaction. Re-develop and improve the inadequacies.

3. Methonds

3.1 The Creation of underwater scene

In order to better reproduce the special effects of underwater environment in the scene, we need to adopt special design methods like special effects lens of movies [2]. Our system mainly uses Unity 3D's own particle system, lighting system and rendering components to realize the realistic underwater environment.

Particle System: The particle system in Unity 3D is a two-dimensional image rendered in three-dimensional space. It can simulate any associative three-dimensional effects, such as smoke, fire, water droplets, snow, explosion and various atmospheric effects. We just need to create the Particle System object in scene, and then set its parameter module to realize the effect we want. In the initial module, we can set parameters such as the duration of particle initialization, cycle mode, launch speed, size and so on. In the emission module, we can set specific time to generate the number of particles to simulate underwater floats, which needs to generate a large number of particles. In the shape module, we can define the shape of the particle launcher such as Sphere, Hemisphere, Cone, Box and so on, which can provide initial force along the normal or random direction of the shape surface, and control the position and direction of particle emission. As shown in Figure 2 (a), it is the distribution effect of underwater floats produced by the Box shape launcher. Texture Sheet Animation module and Render module can be used together to set animated texture for particles. The floating weeds, jellyfish and fish produced by the Particle System are shown in Figure 2.



(a) Floats produced by Box shape launcher

(b) Jellyfish

(c) Fish

(d) Weed

Figure. 2 Effects produced by Particle System

Lighting System: There are four basic light sources in Unity 3D, which are Directional Light, Spot Light, Point Light and Area Light. In the light setting panel, we can adjust the illumination mode and color intensity of the light source. We can also set the parameters of shadow control and halo effect to adjust the influence of light and shadow on objects, so as to better express the scene effect to be displayed.

Rendering settings: In Unity 3D, we can render the scene with settings through Windows ->Lighting ->Settings. In order to create a realistic underwater effect, we turn on the fog effect in the scene and set the color, density and other properties of fog.

3.2 Submarine cruise and route drawing

In order to realize the submarine sailing along the fixed target points and drawing the navigation route, there are three steps, as shown in Figure 3. Firstly, the fixed target points must be interpolated. Then, track the position information of subdivided points to drive the real-time movement of submarine. Finally, draw the route by using the LineRenderer component of Unity 3D.

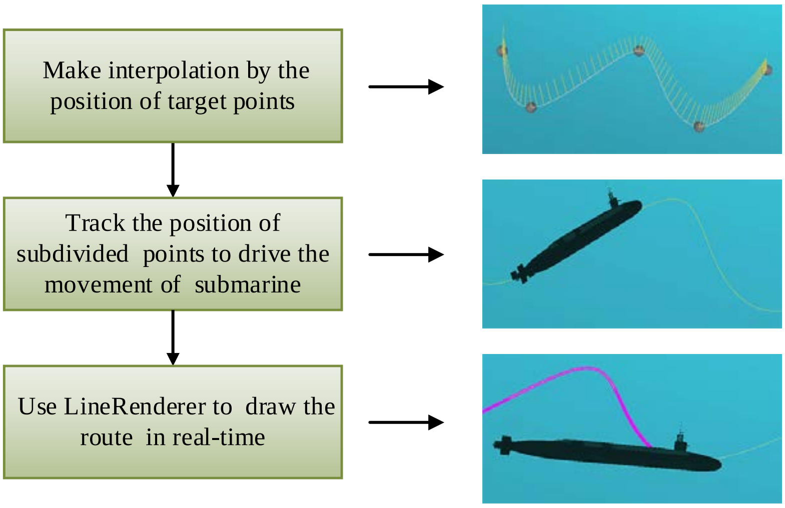

Make interpolation by the position of target points
Track the position of subdivided points to drive the movement of submarine
Use LineRenderer to draw the route in real-time

Figure. 3 Realization process of submarine cruise and route drawing

Make interpolation: Considering that when the submarine cruise is simulated, the straight line is stiffer and the curve is more natural, and the target points need to be passed smoothly. Therefore, our system uses Bezier curve to make interpolation, and combines C# script to realize the Cubic-Bezier.

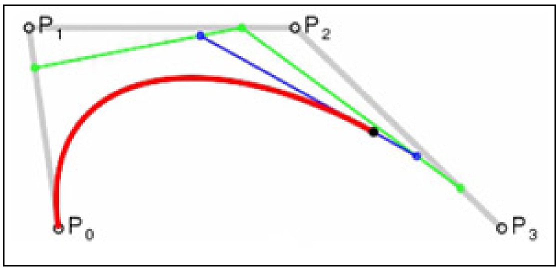


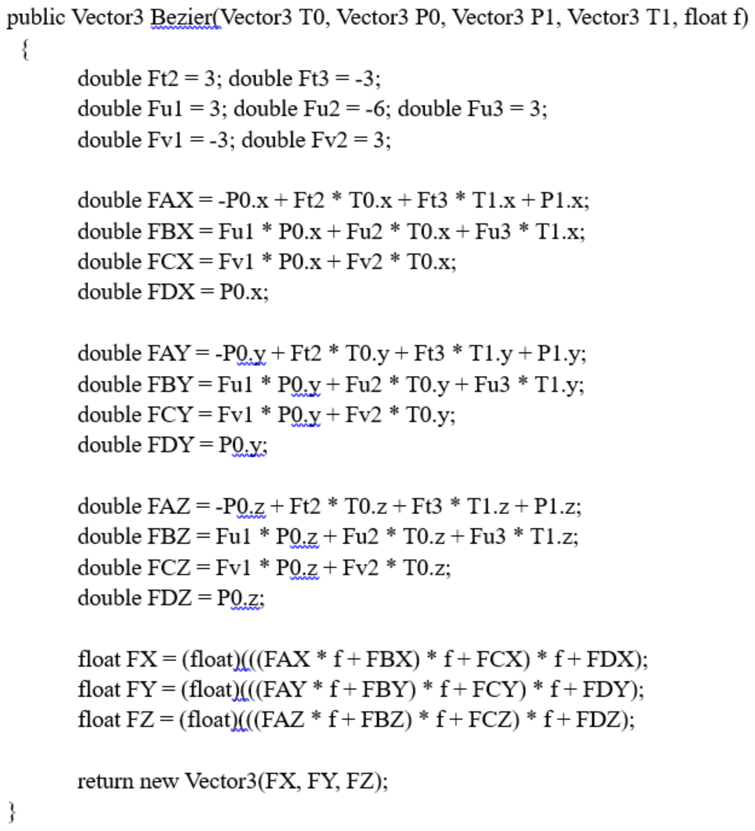
Figure. 4 Principle of Cubic-Bezier curve

As shown in Figure 4, The Cubic-Bezier curve is a cubic polynomial with four control points. The coordinates of four points P0, P1, P2 and P3 are known. The other points on the curve can be calculated by the mathematical expression as (1), and its matrix form can be written as (2).

$$\begin{aligned}\mathrm{Q(t)} &= \sum_{i}^{3} P_i B_{i,3}(t) = P_0 B_{0,3}(t) + P_1 B_{1,3}(t) + P_2 B_{2,3}(t) + P_3 B_{3,3}(t) \\ &= (1-t)^3 P_0 + 3t(1-t)^2 P_1 + 3t^2(1-t) P_2 + t^3 P_3 \quad ,t\in[0,1]\end{aligned} \tag{1}$$

$$\mathrm{Q(t)} = \begin{bmatrix} t^3 & t^2 & t & 1 \end{bmatrix} \begin{bmatrix} -1 & 3 & -3 & 1 \\ 3 & -6 & 3 & 0 \\ -3 & 3 & 0 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} P_0 \\ P_1 \\ P_2 \\ P_3 \end{bmatrix} \quad ,t\in[0,1] \tag{2}$$

When t=0 and t=1, we can find that Q(0) =P0, Q(1) = P3. This shows that Bezier curve passes through the starting point P0 and the end point P3, P1 and P2 are the control points to control the trend of the curve[3], which actually play an auxiliary role and do not need to be drawn out. According to the principle of Cubic-Bezier curve, the code of Cubic-Bezier Interpolation function can be compiled as (3), where T0, T1 are the starting point and end point, P1, P2 are the intermediate control points, and the return value is the coordinate Vector 3(FX,FY,FZ) of the calculated point.



```
public Vector3 Bezier(Vector3 T0, Vector3 P0, Vector3 P1, Vector3 T1, float f)
  {
        double Ft2 = 3; double Ft3 = -3;
        double Fu1 = 3; double Fu2 = -6; double Fu3 = 3;
        double Fv1 = -3; double Fv2 = 3;

        double FAX = -P0.x + Ft2 * T0.x + Ft3 * T1.x + P1.x;
        double FBX = Fu1 * P0.x + Fu2 * T0.x + Fu3 * T1.x;
        double FCX = Fv1 * P0.x + Fv2 * T0.x;
        double FDX = P0.x;

        double FAY = -P0.y + Ft2 * T0.y + Ft3 * T1.y + P1.y;
        double FBY = Fu1 * P0.y + Fu2 * T0.y + Fu3 * T1.y;
        double FCY = Fv1 * P0.y + Fv2 * T0.y;
        double FDY = P0.y;

        double FAZ = -P0.z + Ft2 * T0.z + Ft3 * T1.z + P1.z;
        double FBZ = Fu1 * P0.z + Fu2 * T0.z + Fu3 * T1.z;
        double FCZ = Fv1 * P0.z + Fv2 * T0.z;
        double FDZ = P0.z;

        float FX = (float)(((FAX * f + FBX) * f + FCX) * f + FDX);
        float FY = (float)(((FAY * f + FBY) * f + FCY) * f + FDY);
        float FZ = (float)(((FAZ * f + FBZ) * f + FCZ) * f + FDZ);

        return new Vector3(FX, FY, FZ);
}
```

(3)

Route drawing: At first, we track the position information of the fitting points on the Bezier curve as the current time position of the submarine. Each time, the navigation vector and deflection angle are calculated according to the current and previous time coordinate data, and we can use Unity 3D’s own basic class transform to driven the position and rotation angle of the submarine[4]. Then, we add the LineRenderer component to the submarine object, set the number of drawing points and the position information of the corresponding point, as (4) shows.

```
lr = gameObject.GetComponent<LineRenderer>();
lr.SetVertexCount(points.Length);
while (index < points.Length)
 {
      lr.SetPosition(index, points[index].position);
      index++;
 }
```

(4)

3.3 The design of interaction

The design of interaction in our system mainly includes the interface of interaction and the human-computer interaction based on Hololens. The former is mainly used to improve the operability and display effect of the system, while the latter is mainly used to experience interaction and intelligent triggering.

The interactive interface of our system mainly includes the selection of cruise route, the start, pause, restart of the submarine cruise, and the display of the corresponding depth and time of the dive during the cruise. We use the combination of Cube and Text to make 3D buttons, add C# script to control the interaction when the button is being gazed at and to monitor the button trigger as we can see in Figure 5 (a). And use UIText to display the corresponding data information as a string, which is shown in Figure 5 (b).

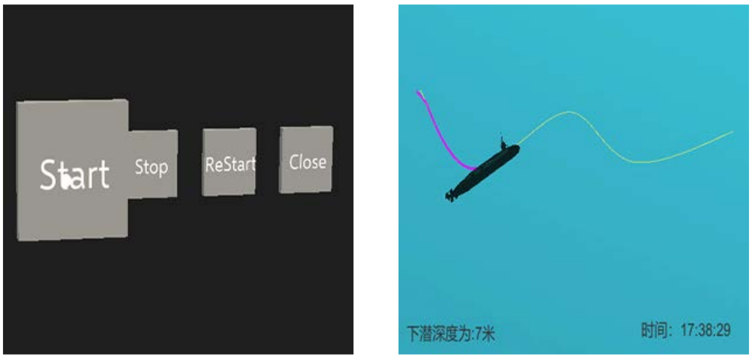
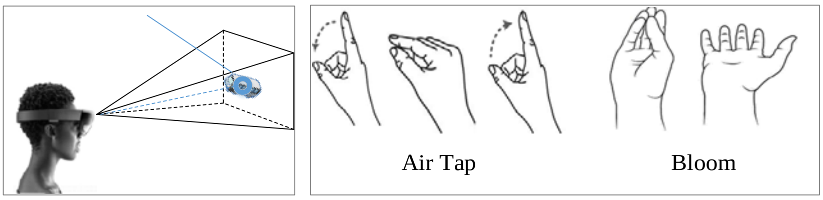


Figure. 5 Design of interactive interface

Our system is ultimately released to the Hololens device for users to experience, so the main human-computer interaction is based on Hololens' gaze and gestures. Gaze is the primary input method of Hololens. The form function is similar to the cursor of the desktop system. It depends on a forward ray between the eyes of the user's head. When the ray identifies the collided object, a circle cursor appears at the collision point, as shown in Figure 6 (a), which reminds the user that an object has been identified. The two basic gestures in Hololens are Air tap and Bloom gestures. As shown in Figure 6 (b), the Air tap is mainly used to achieve the effect of selected and activation, and it includes press and release two actions. The Bloom is used to call up menu [5].



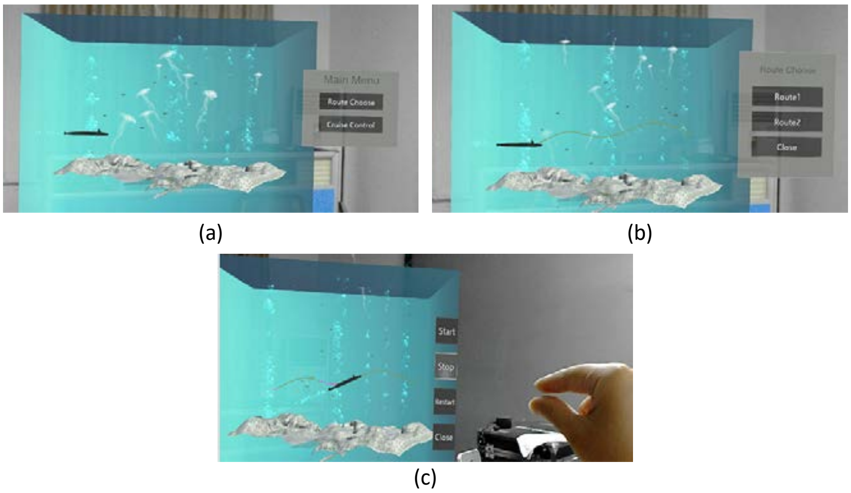
(a) Gaze

(b) Gesture

Figure. 6 Interaction based on Hololens

4. Experiment and simulation

In order to verify the effectiveness of our system, we build an experimental environment, select the basic points of the route in advance, and conduct actual tests on the functions of our system, such as route drawing, submarine cruise, interaction of trigger and so on. The operating platform of the system is Windows 10, we use Unity 2017.4.2 and Visual Studio 2017 software to develop the C# script., and release the system to the Hololens through WIFI. The operation of the system is shown in Figure 7. When the user wears the helmet and enters the system, there is a underwater sand table with interactive menu. Figure 7 (a) is the user’s perspective of the system. The main menu includes two selections: Route Choose and Cruise Control. When the route button is selected, there will pop up a new route choose menu, like Figure 7 (b), which shows the drawing effect of route1. When the cruise button is selected, there will pop up four control buttons, which can be used to control the start, stop, restart and close of the cruise, like Figure 7 (c), which shows the effect of controlling the stop of cruise with gestures.



(a) (b)

(c)

Figure. 7 User perspective of the system operation effect

5. Conclusion

In this paper, we preliminarily developed an underwater demonstration system of submarine cruise by using Hololens helmet and Unity 3D engine. Users only need to wear the helmet, no additional input equipment, and can view and control the whole cruise process from a global perspective with simple gesture trigger. The test results show that the system works well. The inadequacies are that the proportion of scene restoration is small and the submarine mission demonstration is single. The future work can make improvements to these, so that the system experience is better and the function is richer.

References

[1] Zhuo, L., Qingyu, W., Dayuan, G. (2016). Design and Implementation of the Situation Simulation System for Marine Underwater Battlefield. Command Control & Simulation. Vol.38 (4), 79-83.

[2] Boffety, M., Galland, F., Allais, A. (2012). Color image simulation for underwater optics. Applied Optics, 51 (23), 5633-5642.

[3] Jiayao, L., Ruizhi, S., Chunming, C., & Sicong, L. (2018) Roaming Path Generation Algorithm and Optimization Based On Bezier Curve. IFAC PapersOnLine, 51 (17).

[4] Bo, J., Jianqing, C., Wei, Z. (2018). Design and Implementation of 3D Real-time Display System of Sea Battle Field. Command Control & Simulation. Vol.40 (2), 106-109.

[5] Wolfer, J., Wolfer, J., & Wolfer, J. (2018). Approaching emerging technologies: exploring significant human-computer interaction in the budget-limited classroom.

[6] Zheng, N. (2018). The Research of Three-dimensional Underwater Combat Electronic Sand Table Based on VR Technology. Chinese Academy of Command and Control.