A Development of Interactive Game “Ting Ting”

Using Real and Virtual Objects

Si-Jung Kim, Moon-Sung Jang, Hong-Sic Kim
R&D Center
Korea Visuals Corporation
107-7 Samsung-dong, Gangnam-gu, Seoul, Korea
hisijung@korea.com

Tae-Yong Kuc
Intelligent Control & Dynamic Simulation Lab.
Sungkyunkwan University
Chunchun-dong, Suwon, Kyungki-do, Korea
tykuc@yurim.skku.ac.kr

Abstract - This paper presents an interactive virtual reality game using real and virtual objects with general beam projector and computer vision. We show how the computer recognizes the objects and makes the virtual objects move after detecting the real objects in the game space created by a beam projector. To achieve the goals, we developed two methods: Find Objects Distance (FOD) method and Find Reflection Angle (FRA) method. After detecting the real and virtual objects, the methods predict the next moving direction of virtual objects. This interactive virtual reality (VR) game is named as “Ting Ting” which contains various fire-shaped virtual objects and bamboo-shaped real objects. “Ting Ting” game’s core technology can be used in various application area as an interactive communication engine. These include but not limited to, for example, an education game in schools, or an entertainment application in museums and/or science halls.

Keywords: interaction, virtual reality, game, collision, vision, HCI, projector

1 Introduction

Over the decades, various kinds of devices such as computer vision, projector and sensor were developed by scientists and according to the development of each device, other researchers and engineers are developing new interface to play a game or study on the Internet through the e-learning system. More recently, digital electronic devices controlled by a computer have been developed in the interactive application area such as interactive game and bilateral e-learning system. For instance, the aggregation of a projector, a digital camcorder, a digital camera and sound devices with a computer provides us a new level of interactivity which is natural and human friendly. In developing interactive systems, “Videodesk” and “Digitaldesk” are used not only for a desk but also for the integrated computer system that adopts digital electronic devices such as touch screen and camera with a computer. In another research of natural interactive interface, Kenji and Jun Rekimoto worked for the content of computer with 3D hand interaction and full body interaction [2][3][13]. As for the input interface, a “Laser Shot system” was developed by ourselves. “The Laser Shot System” is a non-touchable multiple input device which was developed by using general laser pointers. This input device was used as remote interaction device for indication and selection of target object in a virtual aquarium. In the virtual aquarium, this input device was useful in selecting the menus, expanding the information and controlling the program from a long distance replacing the keyboard and mouse of computer. The laser pointer as an input device enables a user to obtain information from a long distance. In this research, the input device has been developed and used to solve the real problem and make more contents easily operate [5][8][9][12][14]. But one of important problems in implementing the user interface is how to track objects involved in the interaction. This means that one needs to solve the object recognition task in a limited time interval. The time limitation enters into conflict with the detailis and quality of the recognition.

This paper presents how Find Objects Distances (FOD) between real and virtual object and Find Reflection Angle (FRA) of the virtual objects can be found. Our algorithms calculate the distance between the virtual and real objects and determine a new path for each virtual objects. The interactive game “Ting Ting” was implemented to operate on a large-sized floor by using the FOD and the FRA with a general beam projector and vision.
2 Problem description and elimination

To interact with real and virtual objects in the real environment, several problems must first be tackled. These include the display problem, to display the scene fully with avoiding obstacles; fast real objects recognition problem without image distortion from the computer vision system; realistic motion generation problem like acting in real world. All of these problems have input and output condition and require internal function to perform exactly. The considered problems are listed into five categories from 2.1 to 2.5. However we deal only with three problems under 2.3, 2.4 and 2.5 in this paper.

2.1 Game Space Configuration

In interactive contents, the setup aspect becomes a challenge as there is a need to determine the game space in which the real and the virtual object are displayed and meet each other through an interaction device such as a computer vision system. The game space may consist of various elements as shown in figure 1: front type, front rear type, bottom type and bottom rear type. Many researchers have researched how to display the scene fully under the circumstances of perturbing obstacles. This problem has been researched by Rick Kjeldsen who used mirror or mechanical frames named “Everywhere Displays Projector” [4][7][10]. In this paper, we adopt figure 1, (c) as the game space and we premise that there are no obstacles which prevents the displaying in front of the screen. Also in this configuration, the projector is set up vertically as a displayer and along with a camera that is also set vertically.

2.2 Space Recognition Problem

After the setup of the physical environment, one more set up problem remains. We must obtain the whole scene to find the real and the virtual objects in the game space. It is difficult for a computer vision to capture the whole scene without any distortion as the projector and camera are not set up in an exact parallel along with the screen. To settle the display exactly on the game space inputted by the computer vision, first we select four points before starting the program. We set the four regions of interest areas as shown in figure 2.: top left, top right, bottom left, and bottom right. Using this step, the computer can know the interest area of the game space exactly and calculate the object’s movement and process when occurred some events [9].
2.3 Real and Virtual Objects Detection

In an interactive related game, real or virtual objects move according to the moving rule. For instance, a virtual environment, like a walkthrough, creates a computer-generated world filled with virtual objects. These objects should not pass through each object, and things should move as expected when pushed, pulled or grasped. As well known, these actions are the output of collision between two objects as shown below in figure 3. In this point we concentrate more on the real time response than on accurate detection. Collision detection has played a fundamental and important role in computer animation, physically based modeling, game, and simulation. In these applications, interactions between moving objects are modeled by dynamic constraints and contact analysis [8]. The objects' motions are constrained by various interactions, including collisions. In this paper, to achieve this we detected both the real and the virtual objects’ position from the computer vision, and calculated each object’s length while comparing both the real and the virtual object’s color.

![Figure 3. Collision Detection](image)

2.4 Finding Objects Distance (FOD)

Calculating the distance between the real and the virtual object is for the usage of deciding the collision stage or approaching stage of the real and the virtual objects. If there is a real object in the direction of the virtual object, we obtain the position of the two objects and compare it. To calculate the distance and compare between the real object and virtual object we use the Find Object Distance (FOD) method, further described in section 3.

2.5 Finding Reflection Angle (FRA)

When we play recent games, we feel that most of the objects played behave like objects in a real world in terms of movement and rotation as most games are developed under physical rules. Like in a game, we follow the physical rules when the virtual object is reflected by the real object. More specifically, we adapt the law of reflection as shown in figure 4 based on the laws of reflection in physics. In the law of reflection, if we know the incoming angle, we will know the outgoing angle as they are the same.

![Figure 4. The Law of Reflection](image)

The angle of reflection $\theta_r$ of a ray or beam is the angle measured from the reflected ray to the normal surface. From the law of reflection, $\theta_i = \theta_r$, where $\theta_i$ is the angle of incidence. $\theta_r$ is the measure between the ray and the normal of the surface that intersects the surface at the same point as the ray. Section 3 provides further description of the Finding Reflection Angle (FRA) [15].

3 Problem Solving

The following section consists of solving the problems mentioned on section 2. The specific problems are: finding two real and virtual objects in the game space, calculating the distance between the real and the virtual object and finding the reflection angle when the virtual object is reflected by the real object. To solve these problems, our approach is to use the simple and intrinsic method of geographical and geometry method rather than to adapt a complicated mathematical method.

3.1 Detection of Real Objects

As we know, each moving object has its own information such as color, motion vector and speed as shown in figure 5. Generally, there are many detection methods such as in facial recognition and fingerprint recognition. However these methods are highly complex so that using them makes it difficult to formulate fast real-time applications in interactive response games. A characteristic in detecting the real object is that the real object moves and has its own color. But we also know the virtual object’s position and moving angle. These are key factors to escape from the complexity of the problem we have nominated as the method “from the point of view of the virtual object”. This method compares some regular pixels according to the direction of the virtual object to redefine the color table (CT) of the real object. CT was saved before starting the program by computer vision. Through these steps we can detect the real object. And we can know where the real object is on the game space [1][6][11].

1193
3.2 Find Object Distance (FOD)

Find Object Distance method is needed in order to find out the distance between the real and the virtual objects as shown in figure 6. In figure 6, \( P_0 \) is a center point of the virtual object, \( P_c \) is the first collision point of the real object on the line drawn from point \( P_0 \) in given direction of the virtual object’s movement. Also we define the \( \beta \) as the moving direction of the virtual object and \( l_{\text{pixel}} \) as the length of each pixel of the game space. As we know in a bit-mapped image described by a matrix, conjunctions are described by ordered pair of \( x \) and \( y \) location. We described the real object and the virtual object’s position as \( P_0 \) and \( P_c \) on a bitmap plane individually in figure 6.

First of all find the square cell where the imaginary line from point \( P_0 \) in given direction \( \beta \) intersects the real object’s center point \( P_c \). To check whether this line has intersected with a square cell or not, it is sufficient to check the grid intersection points at \( P_1 \), \( P_2 \), \( P_3 \), \( P_4 \) and the real object \( P_c \). To do this, we need to check any grid intersection points that are encountered by the line. And then see if there is a real object color behind the grid or not. When there is a real object color behind either on a vertical or on a horizontal intersection, the checking process stops. The distance between both intersection points is compared and we choose the nearest distance. In brief, the method of finding intersections with horizontal grid lines can be described as in the following steps below.

**Step1:** Find \( Y_a \), the \( y \) size of the image pixel. If the line is turning up, \( Y_a \) will be negative.

**Step2:** Find \( X_a \) using the equation 1, where \( X_a \) is the length of \( x \) size of the pixel.

\[
X_a = \frac{1}{\tan(\beta_a) l_{\text{pixel}}} \tag{1}
\]

**Step3:** Check the grid at the intersection point. If there is a real object color behind the grid, stop and calculate the distance according the equation 1.

**Step4:** If there is no real object color, extend to the next intersection point. Set up the coordinates of the next intersection point to be \( X_{\text{new}} \) and \( Y_{\text{new}} \), where \( X_{\text{new}} \) is calculated by \( X_{\text{old}} \) plus \( X_a \) and \( Y_{\text{new}} \) is calculated by \( Y_{\text{old}} \) plus \( Y_a \).

When this calculation comes to an end the coordinate of the \( P_c \) point will be approximately the same as \((X_{\text{new}}, Y_{\text{new}})\). We can calculate the distance between them like absolute value of \( X_{\text{new}} - X_c / \cos(\beta) \). Finally, come back from the grid to the pixel map to get the distance in pixel’s size. To take the real size in pixels we must divide this calculated value on \( l_{\text{pixel}} \).

3.3 Find Angle Reflection (FRA)

After finding the distance of the two objects, the next potential problem is to find out the virtual objects’ reflection direction as shown in figure 7. In figure 7, line \( ED \) is the length of the real object, \( B \) is the virtual object’s position and \( \beta \) is its moving direction. The line \( BC \) is the imaginary moving path of virtual object \( B \) and line \( CR \) is the reflected path after collision with the real object. Also \( \alpha \) and \( d_a \) is the constant factor appeared by drawing the line \( BD \) to find the reflection angle \( \angle OCR \).
We must find the direction of the CR line which is the reflected direction of the BC line. We calculate the unknown direction of the CR line as defined by equation 2.

\[ \angle OCR = \beta + 2 \angle BCD \] (2)

To find out the angle \( \angle BCD \) in equation 2, we draw the additional line BD from point B to D on the line ED, where angle \( \alpha \) is given. Then we calculate the length of the new line BD using the same method of FOD described in the above section. Let the length of the line CB and BD be \( d \) and \( d_a \), respectively. Now we can determine \( \angle BCD \) defined by two given sides BC, BD and angle \( \alpha \) between them. We show the result in equation 3.

\[ \angle BCD = 90° - \alpha + \arctan \left( \frac{d_a}{d \cdot \sin(\alpha) - \tan(\alpha)} \right) \] (3)

The method of finding the reflection angle is briefly described below.

Step1: Draw the imaginary line BD to the line ED according to the optional angle \( \alpha \), where \( \alpha \) is a constant and given out temporally.

Step2: Find the length of line BD according the Find Object Distance (FOD) method.

Step3: Draw the perpendicular line from point C to line BD and calculate its length. Calculate the two angle caused by the perpendicular line using the parameters \( d \), \( d_a \), and \( \alpha \) and trigonometry.

4 Implementation

The interactive game "Ting Ting" which is played by the real and the virtual object was implemented under IBM compatible Pentium4 PC with computer vision and beam projector. We used Visual Studio dot Net of Microsoft with Direct Draw to calculate the motion information and draw the scene. Microrobot’s vision board (30 frame/sec) was used to acquire real and virtual objects from the game space. We adapted the 30 frames over processing vision board that cover the human motion to make real time processing into image processing. Last, we also used Sharp beam projector to represent virtual objects, generated by the PC.

4.1 System configuration

On the physical game space, we made the five meter side square environment according to the general hardware configuration in figure 1 (c). The game space consists of four main parts: the first one is the beam projector that displays the game scene; the second part is the computer vision in order to detect the real and virtual objects on the game space; the third is the computer part to control the overall device and to analyze the images, and the last part is the real object to play the interactive game. We showed the setted game space in figure 8.

4.2 System block

The interactive game "Ting Ting" needs mainly six functions: 1) capture module for capturing the real object, 2) real object finding module for calculating the real object’s position, 3) virtual object finding module for loading current virtual object’s status, 4) FOD for determining the real and the virtual object collision or not, 5) FRA for calculating the next moving direction, and 6) Renderer for traversing the scene graph. We showed the six main functions in figure 9.

![System Configuration](image1)

![Functional Block Diagram](image2)
4.3 Game configuration

"Ting Ting" game is about capturing the virtual object by the real objects. The virtual objects are drawn on the surface of the game space by the projector and the real objects are used by users at the real environment. Real objects types are that of two bamboo stick shaped images and virtual object consists of several one-type fire-shaped image as shown in figure 10.

(a) Implemented Real Object (b) Generated Virtual Object

Figure 10. Real and Virtual Object

5 Conclusion

The Find Object Distance (FOD) method and Find Reflection Angle (FRA) method are implemented completely. The interactive game “Ting Ting” was developed using two electronic devices, the beam projector and computer vision system with personal computer. We have tested this system using two virtual objects, which is fire-shaped and a real objects, which are two bamboo shaped. Under computer control, the virtual objects were moving around and changed their moving direction after collision with the real object. We showed this in figure 11. The interaction system based on the general beam projector and computer vision and controlled under by a personal computer combines the best simplicity: the natural interface and the power of computational resources. The resulted interactive interface is easy to use so that this can be used by anyone with minimal training. However, potential problems may appear as that of when we recognized the real objects in the dark level light circumstances.

In this paper, we presented the real and the virtual object communication interface including two related processing methods: the FOD and the FRA. The FOD and FRA modules can be useful to make a variety of interactive applications including entertainment and training related contents with fast result.

References


