A Study on Motion-Based UI for Running Games with Kinect

Jimin Kim, Pyeong Oh, Hanho Lee, Sun-Jeong Kim
Interaction Design Graduate School, Hallym University
1 Hallymdaehak-gil, Chuncheon-si, Gangwon-do 200-702 Korea
{Rudengkim, vudhrh}@gmail.com, lee_hanho@naver.com, sunkim@hallym.ac.kr

ABSTRACT

This study examines the efficiency of human motion-based UI for video games with motion capture system, Kinect. We took an investigation to play with the Kinect sensor in the running game which was developed and designed using two kinds of UI. One UI consists of more intuitive and familiar motions such as turning and jumping. The other UI consists of arm motions like raising hands. As a result, UI with arm motions was easier for users to master and results in higher success rates to play than the other UI. Therefore we can conclude when a game is developed using Kinect and its UI is configured with motion recognition, the motion with the arms rather than the other parts of the body helps player better to enhance the play skills and immerse in the game.

KEYWORDS

NUI (Natural User Interface), Kinect, Video Games, Unity3D, Motion Recognition

1 INTRODUCTION

With the development of computers, the input devices for human computer interaction have been diversified into keyboard, mouse, touch pad, speech recognition, and so on. Nowadays many people have actively researched on NUI (Natural User Interface) especially using Kinect which is a motion sensor that came onto the market at low price by Microsoft on 2011. Kinect as gesture recognition sensor is showing potential as an interface of new generation replacing the mouse and keyboard. Some studies show the potential of using motion based interaction for learning. For example ‘Touching Notes’ present how gesture based interfaces can stimulate and motivate children into learning the basics of music notation [1]. For motivating students and enhancing effectiveness, ARCS model of motivation design is considered during developing the Kinect sensor-assisted game based learning system [2]. ARCS model consists of four major steps for learners to become and remain motivated in the learning process: Attention, Relevance, Confidence, and Satisfaction. In addition to learning system, Kinect is utilized in the area of rehabilitation. An interactive game-based rehabilitation tool for balance training of adults with neurological injury was developed [3]. Instead of WiiFit, Kinect provided markerless full-body tracking on a conventional PC. The Kinect-based rehabilitation game, “JewelMine” consists of a set of static balance training exercises which encourage the players to reach out of their base of support [4].

Most previous studies dealt with the accuracy of the recognition of an implementation of a gesture, but the efficiency and convenience of UI itself were not considered. Our research motivation is searching more natural and efficient motions to make it easy for users to play games with Kinect. In this study we focus on analyzing motion-based UI for running games with Kinect.

2 MOTION-BASED UI WITH KINECT

We implemented a running game using a game engine, Unity3D (ver. 4.6.0) and Kinect SDK (ver. 1.8). In common running games, a main character is usually running during play. To avoid obstacles, players choose to jump them or
turn left or right. For input of jumping and turning, we implemented two kinds of motion-based UIs and then examined which one is more natural and efficient for users to play.

Figure 1. Five obstacles are used in our running game.

2.1 Implementation of a Running Game

In our game, the road where a main character runs is randomly generated using prefabs. We display ten prefabs at each frame, and among them we place five obstacles in the scene. We prepare five types of obstacles like Figure 1.

Figure 2. Screen shots of our running game.
Figure 2 shows several screen shots of our running game. A main character keeps running and a user selects to jump or turn left or right whenever she meets an obstacle. If she fails to avoid an obstacle or gets out of the road, the game is over. The game score is the run distance.

To obtain human motion data from a Kinect motion sensor in Unity3D application, we used “Kinect Wrapper Package for Unity3D” [5] which is provided by Entertainment Technology Center (ETC) in Carnegie Mellon University. After importing this package we could use assets for motion tracking. To obtain the position data of joints we modified the function of `Update()` in the script ‘KinectPointController.cs’ which is attached to the prefab ‘KinectPointMan.’ Kinect tracks the skeleton and a tracked skeleton provides the information about the positions of twenty joints of the user’s body (Figure 3).

To recognize a gesture, we should compute a joint angle \( \theta \). We used an equation as follows:

\[
\theta = \arccos(\mathbf{u} \cdot \mathbf{v})
\]  

where an unit vector \( \mathbf{u} \) is \( (X_1-C_x, Y_1-C_y) / \| (X_1-C_x, Y_1-C_y) \| \), an unit vector \( \mathbf{v} \) is \( (X_2-C_x, Y_2-C_y) / \| (X_2-C_x, Y_2-C_y) \| \), and \( (C_x, C_y), (X_1, Y_1), \) and \( (X_2, Y_2) \) are the coordinates of joints’ positions (Figure 4).

As a matter of fact, the position of a joint is 3D coordinates. However we carried out the calculation in 2D for speed. In case of turning gestures, the coordinates of joints’ positions were projected onto XY plane and then a joint angle was computed in 2D. In case of a jumping gesture, the coordinates of joints’ positions were projected onto YZ plane and then a joint angle was computed in 2D too.

2.2 Design of Motion-Based UIs

For experiments we designed two kinds of motion-based UIs. One UI was designed with more intuitive and familiar motions like Figure 5. Figure 5 (a) shows a gesture to make a game character to turn right. Figure 5 (b) shows a gesture to make a game character to jump.
To turn, a user leans the upper body to left or right side. To recognize this motion, we got the positions of three joints: Head, Shoulder Center, and Hip Center. We decided whether to turn left or right by comparing the positions of x coordinates of Head and Shoulder Center. For example, if the value of x coordinates of Head is less than the value of x coordinates of Shoulder Center, this gesture is turning left. Otherwise, it is turning right.

To jump, a user bends the knee. To recognize this motion, we got the positions of three joints: Hip, Knee, and Ankle. After computing a joint angle, if this angle is within threshold range, we decided that a user makes a motion for jumping.

Because joint angles of initial standing posture are not equal to zero, we defined the range of threshold for each joint angle (Table 1).

<table>
<thead>
<tr>
<th>Types of Motions</th>
<th>Initial Value</th>
<th>Maximum Value</th>
<th>Minimum Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>UI with natural motions</td>
<td>Turning</td>
<td>-10~10</td>
<td>60</td>
</tr>
<tr>
<td>UI with arm motions</td>
<td>Turning</td>
<td>45~55</td>
<td>20</td>
</tr>
</tbody>
</table>

3 EXPERIMENTS and RESULTS

To compare two types of motion-based UIs, eleven university students took part in an experimental investigation. For each UI they played our running game ten times. We averaged top five game scores per person. Figure 7 shows its result. In the graph, vertical values (0~120) represent the average of game scores and horizontal values (1~11) represent the ID of person.

As a result, UI with arm motions is easier for users to master and results in higher success rates to play our running game than the other UI. We become aware of its reason as follows: people frequently use arms to express their judgment rather than other parts of the body.
and therefore arm motion becomes to be easier for users and trained more precisely than the motion of other parts of body.

4 CONCLUSIONS

In this study we designed two types of motion-based UI for a running game with Kinect. One UI consists of more natural and intuitive motions such as leaning the upper body to left or right side for turning left or right. The other UI consists of only arm motions like raising both hands for jumping. As the result of experiments, game players prefer arm motions because people usually use their arms to express their decision in daily life and it makes users easy to enhance the play skills, which helps them to immerse in the game. Therefore, for improving the mastery and immersion level of players, it is better to design motion-based UI with arm motions rather than the motion of other parts of body.

ACKNOWLEDGEMENTS

This research was supported by Hallym University Research Fund (HRF-201409-012).

REFERENCES


