Multi-player VR Game built upon Wireless Sensor Network

HyunHo Jee
Game Research Center
College of Information and Communications
Korea University, Seoul, Korea
hhjee@korea.ac.kr

Doo-seop Eom
School of Electrical Engineering
Korea University, Seoul, Korea
eomds@korea.ac.kr

Taeyoung Kim
School of Electrical Engineering
Korea University, Seoul, Korea
taekim@korea.ac.kr

HunKi Park
Game Research Center
College of Information and Communications
Korea University, Seoul, Korea
hunki@korea.ac.kr

Hyoil Lee
Game Research Center
College of Information and Communications
Korea University, Seoul, Korea
onetop21@korea.ac.kr

JungHyun Han
Game Research Center
College of Information and Communications
Korea University, Seoul, Korea
jhan@korea.ac.kr

ABSTRACT

This paper presents a multi-player VR game platform, which uses a motion-tracking system with wireless sensor network (WSN). We use transmitters/receivers of ultrasound and RF signals, and acceleration sensors. Multiple users play the VR game with physical devices in the real world, and the poses of the corresponding virtual devices of the game world are computed and reflected based on the information such as positions, orientations, and velocities of the physical devices. The implementation shows a new type of natural interface for the next-generation VR games which can be well integrated with WSN.

Keywords

wireless sensor network, virtual reality game

1. INTRODUCTION

Wireless sensor network (WSN) provides ubiquitous sensing, computing and communication capabilities. Location-awareness and motion tracking are the key issues in WSN, and virtual reality (VR) applications can benefit from such capabilities. We have developed a WSN-based motion-tracking system and built a multi-player VR game on it by focusing on the scheduling algorithm to detect the identities and poses of the multiple physical devices.

2. WSN-BASED VR GAME PLATFORM

The WSN-based VR game space is presented in Figure 1. The proposed system is composed of *magic wands* controlled by the game players, a game-service PC, a *sink node*, a TV, and *beacons* located at the corners of the TV. We have designed and developed a wireless sensor node which has an *RF transceiver* (transmitter and receiver) and an *ultrasonic sensor* for the magic wands and beacons. In addition, the magic wand has a 3D *acceleration sensor* for measuring the acceleration along three axes. The sink node has only an *RF transceiver*, which is the same as that in the magic wand and beacon.

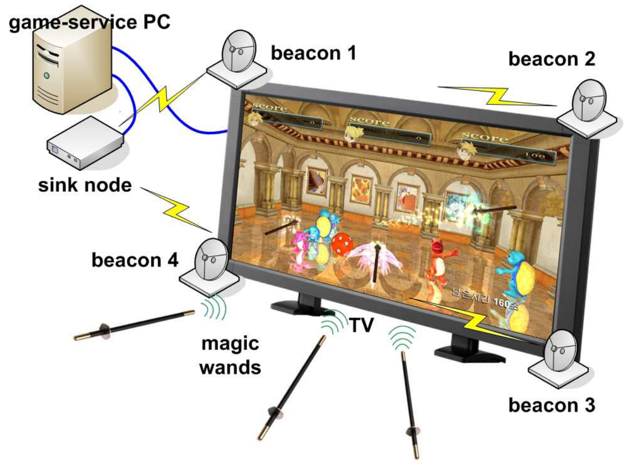


Figure 1: Game space

The sink node synchronizes and coordinates the magic wands and beacons. The algorithm of the program residing in the sink node is shown in the flow chart of Figure 2. Initially, it broadcasts RF signals to the magic wands. As soon as each magic wand receives the broadcast, it responds to the broadcast with an RF message in which its ID is contained. In this way, every active magic wand is identified. The sink node then sends the "start" RF message to all the active magic wands and the four beacons for *synchronization.*

The magic wands work in a *synchronized mode*, and (one at a time) send ultrasonic signals to the beacons. When a beacon receives an ultrasonic signal, it estimates the distance of the magic wand using the standard technique of *time-of-arrival.* It then transmits the estimated distance to the sink

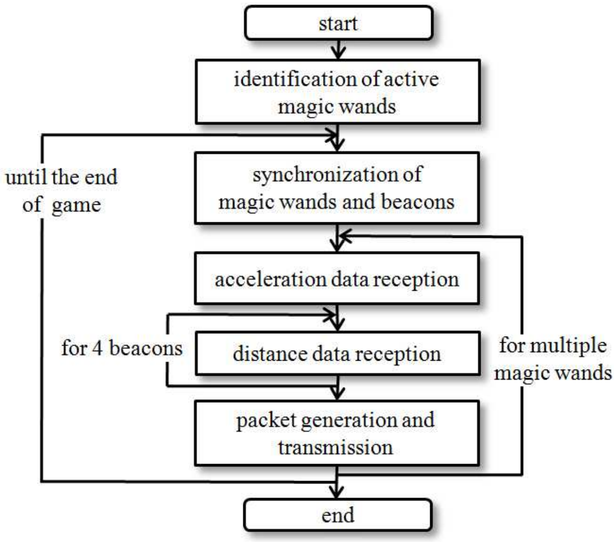


Figure 2: Flow chart of the sink node's program

0	1	2	3	4	5	6	7	8	9
START	wand ID	d_1		d_2		d_3		d_4	

10	11	12	13	14	15	16	17
a_x		a_y		a_z		button on/off	END

Figure 3: Packet of each magic wand

node in an RF message. The four beacons, in turn, send the RF signals at 2 ms intervals.

Each magic wand also sends an RF message containing the acceleration data to the sink node, the moment after sending the ultrasound signal to the beacons. Figure 3 shows the data for a magic wand, collected by the sink node. The 18-byte packet contains the distance data from the four beacons (d_1 through d_4), and the acceleration data for three axes (a_x, a_y and a_z) which are generated by the acceleration sensor of the magic wand.

The packet for a magic wand is sent to the game-service PC, and the sink node then repeats the process for collecting and sending the 18-byte packet for the next magic wand. When all the active magic wands are processed, the sink node again synchronizes the magic wands and beacons, and begins the next iteration.

The game-service PC uses the distance data in the packet and the *trilateration* method for computing the position (x,y,z) of a magic wand. In the current implementation, the initial position (x_0,y_0,z_0) of the magic wand is located in the center of the game space. The trilateration method uses only three distance values, and therefore the largest one among the four distance values is rejected.

The orientation of the magic wand is directly gained from the acceleration data for three axes. Additionally, the angular velocities are roughly estimated by analyzing the orientation changes for generating special effects in the game.

3. PERFORMANCE ANALYSIS

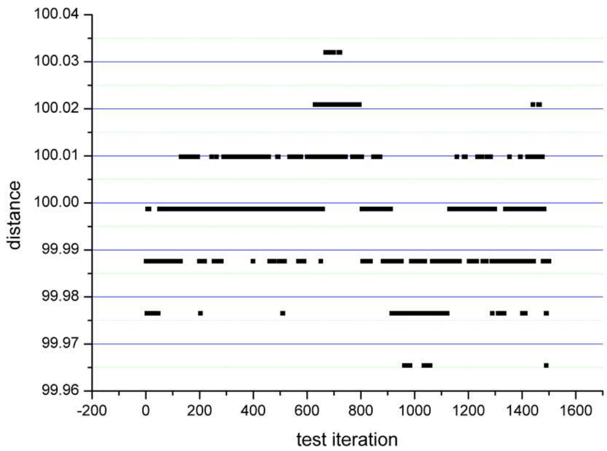
The distance between the magic wand and the beacon is estimated using the arrival time of the ultrasonic signal. We estimated the distance between a beacon and a magic wand at a fixed position 1,500 times for the purpose of performance analysis. Figure 4 shows the experimental result when the magic wand and beacon are 100cm apart. The statistics prove that the proposed WSN can provide precise location information and can be well integrated into location-sensing VR games.

Distance (cm)	Frequency (1,500 times in total)
99.9655	11
99.9765	189
99.9875	318
99.9985	587
100.0095	274
100.0205	111

(a) estimated distances

mean $\bar{X}$	$= 99.99722$ cm
variance S^2	$= 0.000160388$ cm
standard deviation S	$= 0.01266$ cm
standard error SE	$= 0.000326994$ cm

(b) statistical analysis



(c) distances in graph

Figure 4: Test Result

4. IMPLEMENTATION

The size of the beacon node is 8.2 × 7.9 × 6.6 cm^3 and 8.9 × 12 × 5.1 cm^3, and the sink node is of a similar size. They are light enough to be easily placed in an indoor environment. The sensor node uses a 16-bit timer, and the motion data of the magic wand are computed at 50Hz. In case two players participate in the VR game, the system samples the ID and pose of each magic wand at 25Hz, which is sufficient to control the virtual wand in the game world. We have developed a VR game named "CatchMon," the physical space of which is shown in Figure 5. The system tracks the motion of the magic wand and determines the virtual wand's motion in the game world.

Figure 6 shows the screenshot of the game play. The game program is coded in C++ and run on a 1.4 GHz Pentium M CPU with a 512 MB RAM. DirectX 9.0c is used for rendering

Figure 5: VR game "CatchMon"

Figure 6: Screenshot of "CatchMon"

and AGEIA PhysX 2.7.2 is used for the physics simulation of collision detection and resolution between monsters and other objects such as boxes.

5. CONCLUSION

This paper presents a wireless sensor network and a VR game prototype based on it. The system can accurately detect the movements of multiple users differently from the previous works[1][2][3]. The implementation results prove that the WSN can be used for many VR applications that require precisely tracking the movements of multiple users or objects. The current implementation computes the motion data at 50Hz. and more efforts are being made to accept more than two players in the system while preserving the real-time performance.

6. ACKNOWLEDGMENTS

This research was supported by the MIC (Ministry of Information and Communication), Korea, under the ITRC (Information Technology Research Center) support program supervised by the IITA (Institute of Information Technology Advancement) (IITA-2007-(C1090-0701-0046))

7. REFERENCES

[1] H. Balakrishnan, R. Baliga, D. Curtis, M. Goraczko, A. Miu, B. Priyantha, A. Smith, K. Steele, S. Teller, and K. Wang. Lessons from developing and deploying the cricket indoor location system, November 2003.

[2] D. Eom, J. Jang, T. Kim, and J. Han. A vr game platform built upon wireless sensor network. In *Proc. International Symposium on Visual Computing*, pages 1062–1071, Lake Tahoe, USA, June 2006.

[3] N. B. Priyantha, A. Chakraborty, and H. Balakrishnan. The cricket location-support system. In *6th MOBICOM*, Boston, MA, August 2000.