2014 Graduate Research Conference
Department of Mechanical Engineering, Iowa State University
March 7, 2014

RAPID DEVELOPMENT OF NATURAL USER INTERACTION USING KINECT SENSORS AND VRPN

Timothy B. Morgan[1]*, Diana Jarrell[2], Theodore J. Heindel[3], Judy M. Vance[4]

Department of Mechanical Engineering, Iowa State University, Ames, IA, 50011, USA
[1]tbmorgan@iastate.edu, [2]djarrell@iastate.edu, [3]theindel@iastate.edu, [4]jmvance@iastate.edu

Research Program: Virtual Reality

Introduction

The difficulty of integrating input devices is not new. In the past, as new input devices became available, custom software was needed to implement the features of the new devices into existing applications. Furthermore, different applications using the same device often use the device in different ways. For example, different programs require different voice recognition vocabularies. In a worst-case scenario, these differences are hard-coded into the software, preventing non-programmers from changing the interaction.

One advance that has helped reduce device implementation problems is the virtual reality peripheral network (VRPN) [1]. VRPN provides a consistent interface for the most common types of inputs in virtual reality. Due to the advantages of VRPN, programmers have implemented VRPN servers for many low cost input devices [2,3].

The Kinect is another low cost input device that supports non-contact natural user interaction (NUI) in virtual environments. The Flexible Action and Articulated Skeleton Toolkit (FAAST) by Suma et al. provides a VRPN server for the Kinect [4]. While this is a significant step towards providing easy to use interaction, it does not provide access to one of the major abilities of the Kinect, namely the microphone array for voice recognition. Furthermore, FAAST does not support the data integration of multiple Kinects. This paper describes the development of a software tool to support rapid development of NUI using the Kinect through VRPN.

Implementation

The server was implemented using the official Microsoft Kinect for Windows SDK version 1.8 and the Microsoft C# .NET. To implement VRPN in the server, VanderKnyff's VrpnNet library was used [5]. However, a custom version was compiled to fix a bug that was discovered while implementing the server (this build is available at https://github.com/vancegroup).

Due to the limited programming experience of many virtual reality users, the Kinect via VRPN tool was designed to operate from a GUI (Fig. 1). Additionally, voice recognition was implemented using the Microsoft Speech Platform SDK v11. This tool provides a programmatic interface to define words and grammar, and returns recognized events with probabilities that a word has been detected. In addition to the voice recognition itself, beam forming and audio source estimation were implemented to enhance the usefulness of the voice recognition.

Multiple Kinect Skeleton Data

Another key feature is native support for the use of multiple Kinect sensors. In order to integrate skeleton data from multiple sensors, all the skeleton data from all the Kinects must first be transformed into a common coordinate system. The choice of coordinate systems is arbitrary, so we chose the direction of the y-axis to be directly opposing gravity, leaving the yaw and translation of the coordinate system user adjustable. This approach allows the pitch and roll of the Kinect to be automatically calibrated. This is achieved by using the Kinect's internal accelerometer to determine the direction of gravity and rotate the Kinect coordinate system such that the gravity vector is aligned with the negative y-axis [2].

Once all the skeletons are transformed into a common coordinate system, the positions of each skeleton (the center of mass of the skeleton, not each individual point) are compared with all the skeletons from the other Kinects. Once skeletons from the different Kinects are determined to be from the same person, the joint positions are determined. This process is repeated for all skeletons, from all Kinects, until a merged list, which represents each person in the measured area as a unique skeleton, is created.

VRPN Latency

There has also been some concern over the latency that the VRPN interface might introduce into the end application. To test latency, a simple VRPN client was written that records the time at which each VRPN message is received. For the testing of voice recognition latency, a user speaks a given word, and then clicks an onscreen button to record the time of the end of the word.

To create a realistic vocabulary for testing, a list of 100 words was generated by selecting 100 random words from the text of Mark Twain's Adventures of Huckleberry Finn, as obtained from Project Gutenberg [6]. All 100 words were required to meet the following conditions: 1) four letters in length or longer, 2) not a proper noun 3) no two words on the list could be homophones, 4) not a contraction.

*Presenting author: Timothy B. Morgan

As shown in Fig. 2, the total latency of voice commands averaged 912.4 ms, of which 912.0 ms was the latency of the voice recognition engine itself. The voice command measurement has an error of ±126 ms and the VRPN measurement has an error of ±0.5 ms. The latency of the voice engine itself does not appear to be dependent on the size of vocabulary for the range tested. The average VRPN latency is increased from 0.4 ms to 0.8 ms from one to one hundred words, largely due to the increase in the size of the settings list that must be parsed prior to transmitting the command. However, given that in all cases, the latency is less than 0.1% the latency of the voice recognition engine itself, the latency due to the VRPN implementation is considered insignificant.

Discussion

The critical measure of success for the Kinect via VRPN software is whether or not it allows novice users to develop user interactions faster than with existing methods. In one example, a pre-existing virtual assembly software was used, which had been built prior to the decision to use the Kinect as an input device. Through the use of the Kinect via VRPN software, the developer was able to build Kinect-based user interaction (both voice and skeleton tracking) for the virtual assembly software in 20 minutes, without changing any code in the virtual assembly software.

Another example where the Kinect via VRPN software has found success is in virtual reality education. For a class project on the subject of virtual environments, a student with limited programming background was working to develop software to help young children learn English. In this case, the student was able to implement voice recognition in the project using the Kinect via VRPN implementation after being provided only a single, simple example.

Conclusion

This development effort has resulted in a simple tool for implementing natural user interaction in virtual reality using multiple Kinects and voice recognition. The latency added by using VRPN to transmit information has been shown to be minimal compared to the underlying latency of the Kinect sensor. All future improvements will be made publicly available at the project website (https://github.com/vancegroup).

Acknowledgments

The authors would like to thank Patrick Carlson and Ryan Pavlik for their many fruitful discussions about the implementation of this project and their work recompiling VRJuggler to support the VRPN text device.

References

[1] Taylor R. M. I., Hudson T. C., Seeger A., Weber H., Juliano J., and Helser A. T., 2001, "VRPN: A Device-Independent, Network-Transparent VR Peripheral System," VRST '01: Proceedings of the ACM Symposium on Virtual Reality Software and Technology, Alberta, Canada, pp. 55–61.

[2] Pavlik R. A., and Vance J. M., 2010, "A Modular Implementation of Wii Remote Head Tracking for Virtual Reality," ASME 2010 World Conference on Innovative Virtual Reality, ASME, Ames, Iowa, USA, pp. 351–359.

[3] Pavlik R. A., 2013, "rpavlik/razer-hydra-hid-protocol," GitHub [Online]. Available: https://github.com/rpavlik/razer-hydra-hid-protocol. [Accessed: 01-Sep-2013].

[4] Suma E. A., Lange B., Rizzo A. "Skip," Krum D. M., and Bolas M., 2011, "FAAST: The Flexible Action and Articulated Skeleton Toolkit," 2011 IEEE Virtual Reality Conference, IEEE, Singapore, pp. 247–248.

[5] VanderKnyff C., 2008, "VrpnNet 1.1.1" [Online]. Available: http://wwwx.cs.unc.edu/~chrisv/vrpnnet. [Accessed: 15-Mar-2012].

[6] Twain M., 1885, Adventures of Huckleberry Finn, Project Gutenberg.

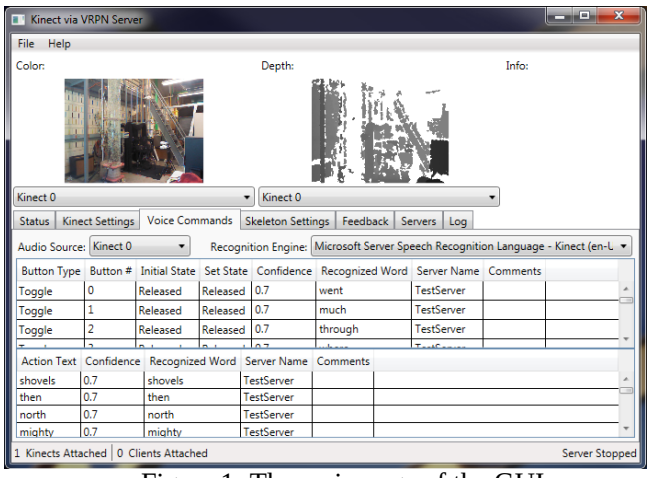

Figure 1: The main page of the GUI.

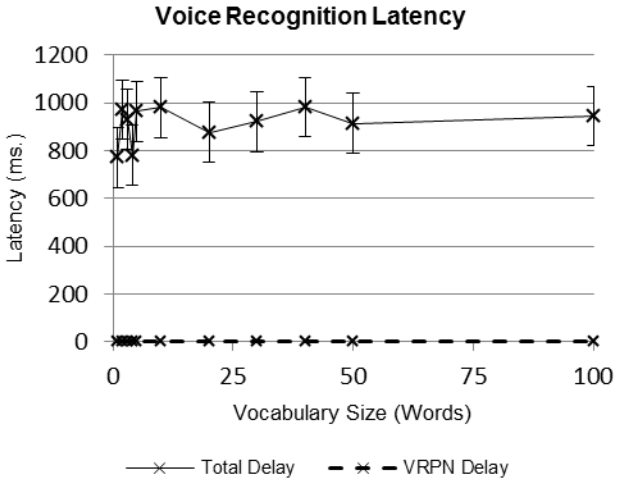

Figure 2: Latency of the voice recognition by vocabulary size.