Asian Journal of Social Sciences and Management Studies
ISSN(E) : 2313-7401
ISSN(P) : 2518-0096
Vol. 3, No. 4, 244-247, 2016
http://www.asianonlinejournals.com/index.php/AJSSMS

Natural User Input Management (NUIM) Through Computer Architecture and Operating System in Virtual Reality

M. Salman Akram[1]
Shahid Naseem[2]
Mushtaq Niazi[3]
Muhammad Waqas[4]
Tehami Mustaasam[5]
Ayesha Iqbal[6]

[1,3,4,5,6]National College of Business Administration & Economics, Lahore, Pakistan
[2]UCEST, Lahore Leads University, Lahore, Pakistan

(*Corresponding Author)*

Abstract

Virtual Reality (VR) takes natural user inputs (NUI), like gestures, motions, voices and produces visual overlays in digital form on top of reality that seen by different user at a time. Today's Operating Systems don't provide special support for VR application. As a result, today VR application are built as single hardware, single experiences, where the application itself performs as sensing and user input interpretation with machine hardware. In VR application environment new every inputs and outputs, an operating system that support VR applications needs to re-think every output and display abstraction exposed to exist applications. Disparate mouse, keyboard and other devices, which from explicit, which frequently has sensitive data mixed with user input. So, operating system with machine architecture must learn in VR applications to manage with the inherent noisiness of machine learning for recognizing for access to recognized objects like user face and skeleton in VR applications.

Keywords: Virtual reality, Gestures, Abstraction, Skeleton, Kinect.

Contents

Citation | M. Salman Akram; Shahid Naseem; Mushtaq Niazi; Muhammad Waqas; Tehami Mustaasam; Ayesha Iqbal (2016). Natural User Input Management (NUIM) Through Computer Architecture and Operating System in Virtual Reality. Asian Journal of Social Sciences and Management Studies, 3(4): 244-247.

DOI:	10.20448/journal.500/2016.3.4/500.4.244.247
ISSN(E) :	2313-7401
ISSN(P) :	2518-0096
Licensed:	This work is licensed under a Creative Commons Attribution 3.0 License
Author(s) Contribution:	All authors contributed to the conception and design of the study.
Funding:	This study received no specific financial support.
Competing Interests:	The authors declare that they have no conflict of interests.
Transparency:	The authors confirm that the manuscript is an honest, accurate, and transparent account of the study was reported; that no vital features of the study have been omitted; and that any discrepancies from the study as planned have been explained.
Ethical:	This study follows all ethical practices during writing.
History:	**Received:** 10 May 2016/ **Revised:** 2 July 2016/ **Accepted:** 11 July 2016/ **Published:** 15 July 2016
Publisher:	Asian Online Journal Publishing Group

1. Introduction

The last several years have seen the popularity of consumer virtual reality (VR) applications. VR application takes natural user interactions such as gestures, voice and motion tracking as input and overlays digital content on top of the real world. VR technology allows users to transfer almost any hardware / component of real world into the digital form. Nature of such technology can only static but also dynamic, e.g. whole body, arms or legs movement as well as other NUI. Another important step in virtual reality for machine is to recognize and identify an object. For purposes of speech, motion tracking, device must be able to sense the changes in real time; this is enabled by specific sensor dependent on used technology (Ondrej and Frantisek, 2014).

Existing current Operating Systems provide no specific and reliable support for VR applications. As an effect, today's VR applications are built as single support experiences on separate hardware and tools. In current existing Operating System where the applications itself perform sensing, depiction and user input interpretation (like motion, speech or gestures), aided by user-space libraries such as the field of view (FOV) software Microsoft Kinect Software Development Kit or Kinect SDK. Today, these applications can only run at a time, and they receive unrestricted, exclusive, access to read the NUI sensors and render augmentations.

As a matter of fact, this approach has two major challenges in Operating System. That is user privacy and lack of support for simultaneous applications. First, it is unwanted and undesirable to give any application complete access to speech and other sensor data. Consider Figure 1 which shows a voice frame captured from a Kinect for Windows SDK. Now a day, any application using the SDK has access to the raw voices and depth flow. On this case, that includes the different voices variation at a time.

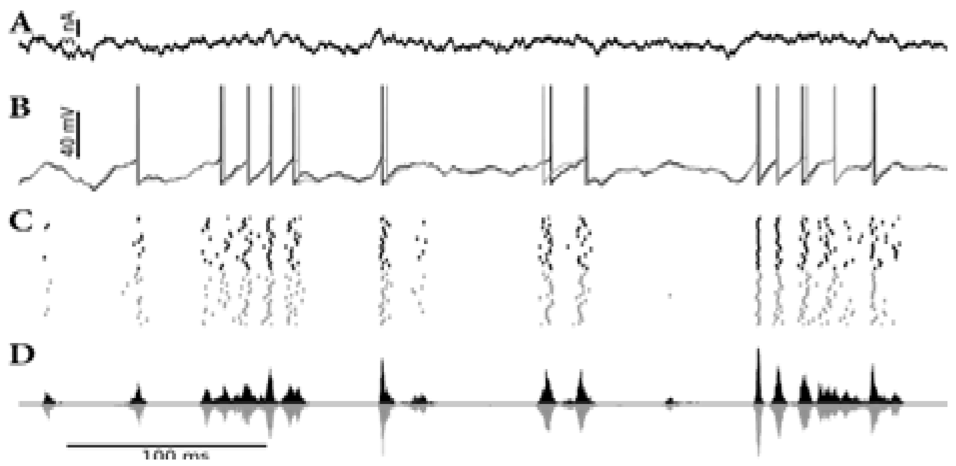

Figure-1. voice frame captured variation at a time

Therefore, we must recognize how the Operating System interprets and delivers natural user input to application using machine hardware that is preserving user privacy. In Operating System, we required a finer-grained and additional usable permission system to allow slightest privilege granting in the Virtual Reality environment. Let's take an example, instead of granting any application to access the camera, as currently happening in smartphone now a day, Operating System needs to control the access to user's face, eyes, skeleton and other objects etc. Second, instead of persist running VR applications at once, we disagree that it is desirable and forceful to let different multiple VR applications from different vendors simultaneously read sensor all generated inputs and render by virtual overlays in VR. Evidence is rising that such a multi-application of VR platform can be highly attractive.

2. Related Work

The characteristics a virtual reality system must have in order to exploit the perceptual and spatial skills of users (Cagliari, 2006). Different input device could serve as an effective Virtual Reality platform and provide environment of sufficient immersion and presence, e.g. through multimodal interaction. There are limited possibilities for VR applications manage NUI in different visualizations and processing power either device has small visual field or not (Jane and Jaehoon, 2006). In Mobile Systems may need to identify object recognition to the cloud or to specialized hardware for NUI, which raises heterogeneity issues similar to those tackled by the Helios satellite kernel architecture (Kyungmin and Jason, 2015).

3. Propose Methodology

Overall, VR applications require new mechanism and abstraction from the Operating System and Hardware Architecture for multiplexing sensor inputs and recognized input gestures across multiple applications. The prime requirement of a VR Operating System is that it should be real-time, as we know that the concept of Virtual Reality is based on direct input from a user body, so it must be able to respond quickly to any movement in order to accurately simulate reality. The requirements of a given OS are dependent on the use of Virtual Reality system and hardware equipment. Certain VR systems do not require high performance and have fewer features while other needs essential real-time performance (Kyungmin *et al.*, 2015).

When we perceive that integration into the Operating System machine hardware / architecture, it does not mean that functionality must necessarily be placed in the OS kernel. Instead, we mean that shared, trusted functionality that is isolated from every application should be provided (Figure 2).

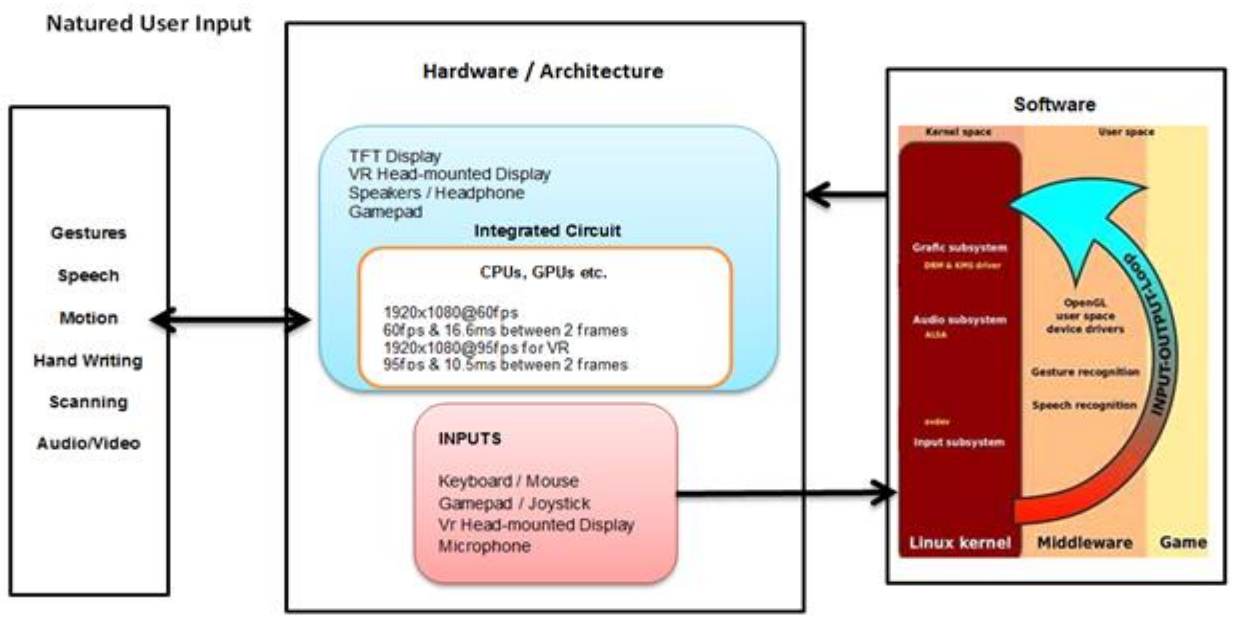

Figure-2. Integration into Machine Hardware Architecture

Another option for VR application protecting those inputs with a fine-grained access control policy enforced centrally by the OS, and providing presentation abstraction for applications. Fine-grained programs have different protection NUI and may interact often in the course of a computation. Several Operating Systems use hardware-based protection to prevent progress from inadvertently and / or maliciously modifying one another. Each process has an address space that defines a set of memory segments and the process's access rights to those segments. A process can only access memory in its own address space. In addition, the operating system has a security model that associates processes with their access right to system resources for NUI. Using address space of suitable granularity and process access rights to controlled resources, an operating system can control a process's operations as desired (Mishra, 2016).

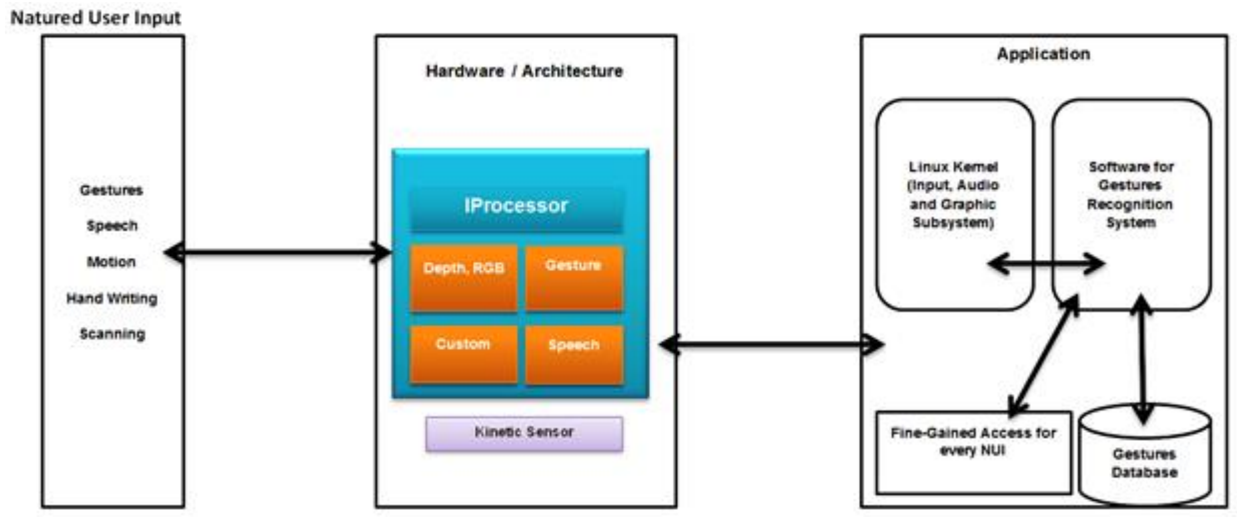

Figure-3. Choose right processor type in application and control the sensor function with fine-grained access

The Microsoft Kinect Software Development Kit (SDK) is used as a hardware abstraction layer to establish the connection to the Kinect. Microsoft anticipates for long-term support and often provides new features enhancement. The idea and functionality behind this model is not build upon the Software Development Kit of Microsoft for Kinect. It is negotiable with other SDKs which are compatible to the Kinect, like the SDK of OpenNI.

IProcessor represent (Figure 3) with the different sensor inside the Kinect for Microsoft device it is possible to recover Depth, Speech, RGB and Skeleton data. This information can be sophisticated with the IProcessor edge implementing different components. The IProcessor rectangle in Figure 3 contains calculation specific parts, that part can be exchanged. It is actually contains abstract parts and interfaces that can be used for implementing single combining different kinds of processors freely for handling more than one sensor and combining data in the calculation process. For example Graphics, Depth, Speech and Skeleton sensor can be combined in a multi-processor to cut an exist person out from the images background and it is also called "Green-Screen". Such processor can also switched on and off separately in order to avoid errors or save resources when no person is in front of the sensors (Banerjee *et al.*, 2007).

Once the application developers have different approaches for security mechanism should focus on NUI, what layers the mechanism should be adopted for operating system and the complexity of each mechanism. This mechanism needs to be created and integrated in way to have a proper relationship with other part of the operating system.

4. Conclusion

Overall, VR applications require new approach, mechanism and abstractions from the operating system for multiplexing sensor inputs and recognized NUI in different applications. In order to protect those inputs the best option is fire-gained across control policy imposed centrally by the OS. The OS need to handle the Physics between different applications which can be considered a new kind of cross-application interaction. The result has shown that fully support of operating system in virtual reality for NUI is case of release pressure on application. Now we need to create more application that should be reliable compatible while interacting with Operating Systems and computer architecture.

References

Banerjee, P.P., J. Cristian and S. Rizzi, 2007. Virtual reality simulations. Anesthesiology Clinics, 25(2): 337-348.

Cagliari, L., 2006. Virtual reality: Past, present and future, in virtual environments in clinical psychology and neuroscience. Amsterdam, Netherlands: Ios Press.

Jane, H. and J. Jaehoon, 2006. Hand-hled virtual reality: A feasibility study. Available from www.cs.wpi.edu/~gogo/hive/papers/Hwang_etal_VRST2006.pdf.

Kyungmin, L. and F. Jason, 2015. The case for operating system management of user attention. Available from www.eecs.umich.edu/~kyminlee/papers/hotmobile15.pdf.

Kyungmin, L., F. Jason, T.J. Giuli, N. Brian and P. Christopher, 2015. The case for operating system management of user attention. AMC: verifying user interface properties for vehicular applications. ACM: 1-6.

Mishra, D., 2016. Virtual reality. IJLTEMAS, 5(1): 2278-2540.

Ondrej, K. and J. Frantisek, 2014. Approach to hand tracking and gesture recognition based on depth-sensing Cameras and EMG monitoring. Acta Informatica Pragensia, 3(1): 104-112.