Adaptive Window View With Arbitrary Sensor Position

Vincentius Madya
Grab
vincentius.madya@grabtaxi.com

Suhendry Effendy
National University of Singapore
effendy@comp.nus.edu.sg

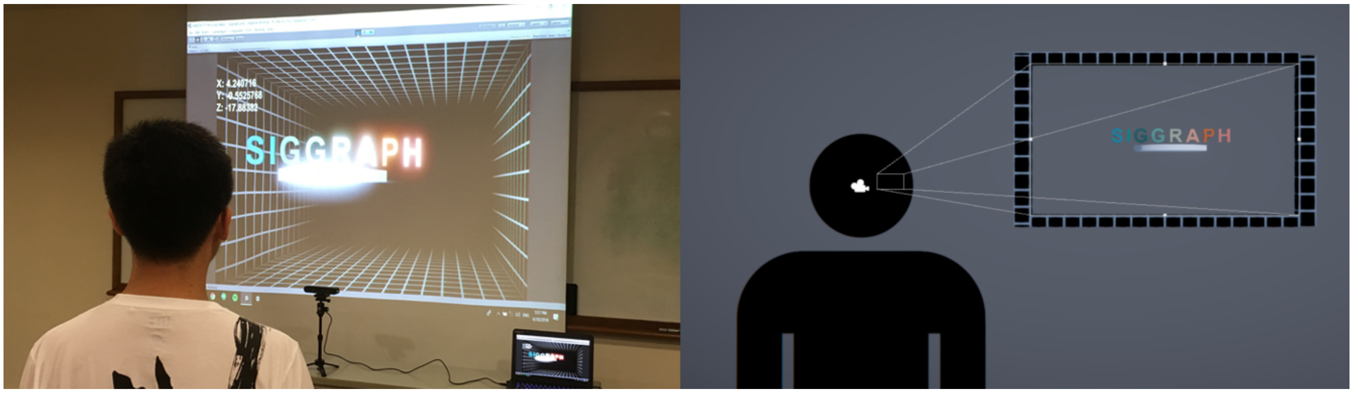

Figure 1: Adaptive window view simulator.

ABSTRACT

This paper demonstrates an adaptive window view based on observer's position with arbitrary sensor position. We show that it is possible to achieve a head-coupled display when the sensor position is not the same as the display position. We used Orbbec depth sensor to track the observer's position in the real-world and calculate the information to display the correct projection of the virtual space.

CCS CONCEPTS

• **Human-centered computing → Displays and imagers**; *Virtual reality*;

KEYWORDS

adaptive window view, head-coupled display, perspective, virtual reality

ACM Reference Format:
Vincentius Madya and Suhendry Effendy. 2018. Adaptive Window View With Arbitrary Sensor Position. In *Proceedings of Siggraph Asia (SIGGRAPH Asia'18).* ACM, New York, NY, USA, 2 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

1 INTRODUCTION

This paper demonstrate an adaptive window view simulator which display is determined by observer's position in the real-world (Figure 1). In contrast to most existing works[1–3] which require the sensor to be at the same place as the display, our system allows the sensor to be placed at any arbitrary position (not necessarily at the same place as the display).

The head-coupled perspective virtual space projected by the window view in this work is developed with Unity engine[1]. Head-coupled perspective is a technique to project the display based on user's head/eyes perspective[1].

There are several works on adaptive window view. Radikovic, et al. show that a system using head-coupled display and image-based rendering has a more positive effect on observer's mood[2]. Weikop, et al. build a similar system using Polhemus FasTrak electromagnetic tracking system[3]; however, electromagnetic tracking device is very sensitive to metalic objects and other electormagnetic fields in surrounding environment, thus it may cause problem.

2 IMPLEMENTATION

In order to build the system, we need a display window to the virtual world, a position sensor to keep track the observer's position in the real-world, and an engine to control the display based on the sensor. In this work, we simply use monitor as the display and Orbbec Astra 3D Camera as the position sensor; we demonstrate that such equipments are sufficient to achieve our goal. It is possible to substitute the 3D Camera with a conventional video camera, but doing so will require us to extract the observer's position manually from the retrieved images. The device we used already supplied us with the observer's position.

Most work on auto-adjusting display relies on the observer's position in the real-world as its input; however, the sensor usually is placed at the same position as (or on top of) the display[2, 3]. In our work, we take the problem further by allowing the sensor to be placed at arbitrary position but still able to capture the observer's position when the observer is able to see the display.

SIGGRAPH Asia'18, December 2018, Tokyo, Japan

ACM ISBN 978-x-xxxx-xxxx-x/YY/MM.
https://doi.org/10.1145/nnnnnnn.nnnnnnn

[1] https://unity3d.com/get-unity/download/

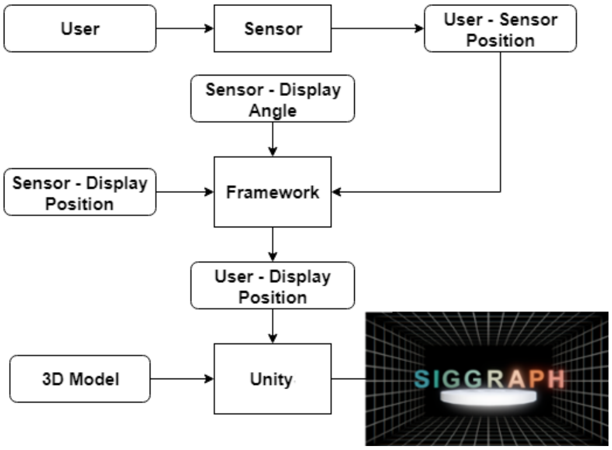

Figure 2: Workflow

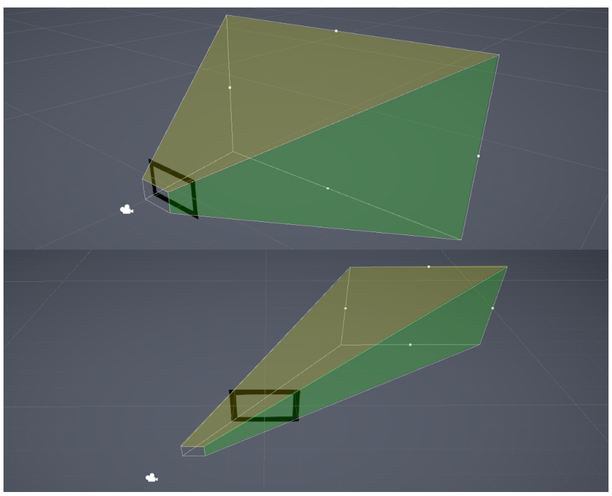

Figure 3: (Top) Frustrum's shape if seen at a close range. (Bottom) Frustum's shape if seen from the left side of the window.

2.1 User's Position

The observer's position in our work is obtained with a 3D Camera, which returns the human-point positions (e.g., head, body, arm, etc.) of the observer relative to the sensor. We only use the head position information. We then calculate the observer's position relative to the display based on the observer's position relative to the sensor combined with the position and angle of the sensor to the display (Figure 2).

2.2 Viewing Projection

To render the model (virtual space), we use custom frustum which change depends on the observer's position. We cannot use Unity3D's default perspective frustum because we have to cover exactly every edge of the display, i.e. top frustum to cover top display, right frustum to cover right display, bottom frustum to cover bottom display, and left frustum to cover left display (Figure 3).

(a) Left view **(b) Center view** **(c) Right view**

Figure 4: Sensor position in front of the display with no rotation.

(a) Left view **(b) Center view** **(c) Right view**

Figure 5: Sensor position to the right of the display with some rotation.

3 RESULTS

We evaluated our system by varying the sensor's location and rotation. Figure 4 shows the result when the sensor is located in the front of the display, and it has no rotation (the same angle with the display). Figure 5 shows the result when the sensor is located to the right of the display, and it is rotated (different angle with the display). To obtain a smoother projection transition, we performed averaging on the oberserver's head position for every 30 consecutive frames. The system works well and managed to obtain a head-coupled display despite of the sensor's location.

4 CONCLUSIONS

In this paper, we demonstrate an adaptive window view which simulate the virtual space based on obvserver's perspective. We show that it is possible for the sensor to be placed at arbitrary position (not necessarily on the same place as the display) to achieve our goal.

This system works well for a small room. However, as we only used single sensor for this work, there is a possibility that the observer's position is not captured by the sensor (e.g., due to sensor's range or obstacles). For future work, we plan to improve the system by deploying multiple sensors so that it can cover larger area, but cautions are needed, as there may be new challenges arise from this.

REFERENCES

[1] Kevin W Arthur, Kellogg S Booth, and Colin Ware. 1993. Evaluating 3d task performance for fish tank virtual worlds. *ACM Transactions on Information Systems (TOIS)* 11, 3 (1993), 239–265.

[2] Adrijan S Radikovic, John J Leggett, John Keyser, and Roger S Ulrich. 2005. Artificial window view of nature. In *CHI'05 Extended Abstracts on Human Factors in Computing Systems*. ACM, 1993–1996.

[3] Jonas M Weikop, Rasmus Stenholt, Rune E Laursen, and Erik Granum. 2005. Creating a Virtual Window using Image Based Rendering. In *Proceedings: 8th Annual International Workshop on Presence (PRESENCE 2005)*.