TickTockRay:
Smartwatch-Based 3D Pointing for Smartphone-Based Virtual Reality

Daniel Kharlamov
California State University Monterey Bay
`dakharlamov@csumb.edu`

Brandon Woodard
California State University Monterey Bay
`bwoodard@csumb.edu`

Liudmila Tahai
IDEA Laboratory, University of Waterloo
`ltahai@uwaterloo.edu`

Pietroszek, Krzysztof
California State University Monterey Bay
`kpietroszek@csumb.edu`

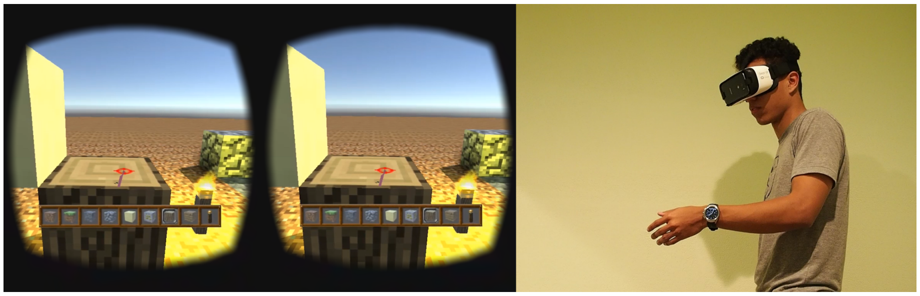

Figure 1: *TickTockRay enables freehand pointing in mobile VR using an off-the-shelf smartwatch.*

Abstract

TickTockRay is a smartwatch-based raycasting technique designed for smartphone-based head mounted displays. It demonstrates that smartwatch-based raycasting can be reliably implemented on an off-the-shelf smartphone and may provide a feasible alternative for specialized input devices. We release TickTockRay to the research community as an open-source plugin for Unity along with an example application, a Minecraft VR game clone, that shows the utility of the technique for placement and destruction of Minecraft blocks.

Keywords: freehand pointing, 3D pointing, smartwatch, smartphone, myo, game input, virtual reality, immersive systems

Concepts: •**Human-centered computing** → **Pointing;**

1 Introduction

Smartphone-based head-mounted displays (HMDs) democratize access to viewing virtual reality environments, yet provide very limited interaction capabilities. Our work demonstrates that it is feasible to use off-the-shelf smartphones' built-in sensors (accelerometer, magnetometer, and gyroscope), and bluetooth or wifi connectivity, to implement a 3D pointing for mobile virtuality. TickTockRay, an implementation of fixed-origin raycasting technique that utilizes a smartwatch as an input device addresses the need of an alternative input technique. We believe that a smartwatch-based raycasting is a good alternative to a head-rotation-controlled cursor or a specialized input device because both smartphones and smartwatches are devices that are carried by a user at all times. As an example application for TickTockRay, we implement simplified version of a Minecraft VR game[1] that supports a selection of block types from a toolbar, as well as placement, and destruction of Minecraft blocks.

2 Motivation

3D selection is a pre-requisite task for many 3D interactions. In smartphone-based immersive environments, 3D pointing capabilities are commonly implemented through a centrally-positioned reticle selection tool that follows the user's head rotation. Target acquisition is achieved either using dwell time (Google Cardboard) or with a click of a button on the side of the headset (Samsung Gear VR). Such implementation of 3D pointing leads to a number of issues. Using dwell time requires keeping the user's head in a static position for a prolonged amount of time, increasing the difficulty of acquiring small targets and causing fatigue of the neck muscles. Moreover, confirming selection with dwell time may result in a Midas touch effect, where centrally-positioned targets are acquired unintentionally. Confirming selection with a click of a button placed on the side of the headset may cause Heisenberg effect [Bowman et al. 2001], that is a miss-selection resulting from a sudden movement of the cursor out of the target at the time of a click due to the force applied to the headset by the user's finger. Alternative to head-rotation-based selection, a specialized input device can be used. However, a commonly used low-cost bluetooth-connected keyboard, touchpad, or game controller does not provide enough degrees of freedom for effective 3D pointing [Zhai 1995]. Voice input may be used, as in a classic "Put-that-there" study [Bolt 1980], yet it is not a reliable input channel in a loud environment. Spatial input devices, such as Leap Motion, Nintendo WiiMote, or Thalamic Myo could also be used with smartphone-based HMDs, however these devices are not typically carried by the user. Re-

---

VRST '16, November 02-04, 2016, Garching bei München, Germany
ISBN: 978-1-4503-4491-3/16/11
DOI: http://dx.doi.org/10.1145/2993369.2996311

[1] Code available at https://github.com/gameresearchlab/MinecraftVR

cent work proposes using a spare smartphone for 3D pointing [Pietroszek et al. 2014], an alternative that is valid only for a user who has access to a second smartphone. Researchers have also explored the use of wearables, including smartwatches, to enable 2D pointing [Kim et al. 2012; Katsuragawa et al. 2016]. In this work we significantly improve the demo version of TickTockRay [Kharlamov et al. 2016] by supporting dragging, and eliminating the Heisenberg effect. The work demonstrates that off-the-shelf smartwatch can be seamlessly used as a pointing device for 3D environments presented on a smartphone-based HMD.

3 TickTockRay Design

TickTockRay[2] implements fixed-origin raycasting using off-the-shelf smartwatch hardware. In the classic version of raycasting [Liang and Green 1994] the ray originates in the tracking device. In contrast, our implementation relies upon the ray originating from a fixed point, located in the camera. Fixed-origin raycasting is implemented instead of moving-origin raycasting because the smartwatch position in 3D space is hard to track reliably using only the smartwatch's built-in sensors. The C/D ratio, a coefficient that maps the physical movement of the pointing device to the resulting cursor movement in a system, is set to 1:1, with exact correspondence between the ray and the smartwatch's rotation. Such C/D ratio enables a user to select targets in the entire virtual reality control space. To manipulate the rotation of the ray, our implementation uses Android SDK's RotationVector, which is a result of sensor fusion of gyroscope, accelerometer and magnetometer sensors. In our experiments with two smartwatch models, LG G and LG Urbane 2, the RotationVector provided the rotation data at 50Hz with no noticeable jitter, minimal latency and no drift.

3.1 Selection Confirmation

When the ray intersects a target, a selection confirmation must be performed to complete the acquisition of the target. This could be done by touching the smartwatch's touchscreen, but such interactions felt awkward to perform, and was fatiguing for our pilot users. It also caused the Heisenberg effect. Another possibility was to confirm selection via a motion gesture. We pilot-tested gestures such as *grabbing*, *fingersnapping*, and *poking*, but found that we were unable to detect these gestures in real time with no latency and high accuracy using the built-in sensors. Moreover, even when the gesture was pronounced well enough to be correctly recognized, it often resulted in a strong Heisenberg effect. To confirm the selection, we decided to use a quick forearm twist by a minimum of 40°. We mapped the counterclockwise twist to the left-mouse-button-down event, and the clockwise twist to the left-mouse-button-up event. When the clockwise twist is immediately followed by a return to the original position, the system registered a left-button click, a command that confirms the target acquisition. To recognize the right click, we reverse the mapping: the clockwise twist is mapped to the right-mouse-button-down event, and the counterclockwise twist to the right-mouse-button-up event. Finally, to address the Heisenberg effect, we introduce a new state: a *potential click*. When the wrist is twisted by 20° we temporarily freeze the cursor, expecting that the rotation may be the start of a click, so that the continued rotation does not move the cursor out of the target. The value of 20° was determined experimentally as it did not produce any false positives in our pilot study. This technique eliminated the Heisenberg effect for our target sizes, the Minecraft blocks. Further investigation is required to verify whether the technique is effective for smaller targets. Finally, the detection of the mouse-down event is confirmed with auditory feedback, a "tick", while the mouse-up is confirmed with a "tock" sound. Thus, the audio feedback for a click sounds as "tick-tock".

3.2 Fatigue

In order to reduce the gorilla arm effect [Hincapié-Ramos and Guo 2014], users are encouraged to interact by moving the forearm, keeping the elbow close to the body, and, if needed, reposition their entire body, instead of making wide movements with their shoulder joint. We hypothesize that this approach will result in lower fatigue than using the whole arm. We will verify the hypothesis experimentally in future work.

4 Limitations and Future Work

Our work demonstrates the feasibility of using a modern smartwatch to implement a 3D pointing technique for smartphone-based virtual reality environments. However, in developing this prototype we have also identified its limitations, and subsequently, future work to address those limitations. Most importantly, we left a formal evaluation of TickTockRay's performance to prospective work. Also, while TickTockRay is currently implemented on a smartwatch, it is important to note that 9-axis inertial measurement units (IMUs) are now integrated into any number of inexpensive and wearable devices, such as those manufactured by Fitbit or Thalamic Labs. Future implementations may seek to run on a user's device of choice.

5 Conclusion

TickTockRay demonstrates that wearable devices, such as a smartwatch, are now capable of providing robust, effective support for 3D pointing and selection in virtual environments. For ubiquitous 3D interaction users do not need to own and carry specialized pointing devices: off-the-shelf smartwatches are already able to support these interactions.

References

BOLT, R. 1980. "Put-that-there". In *Proc. of SIGGRAPH '80*, 262–270.

BOWMAN, D., ET AL. 2001. Using pinch gloves (tm) for both natural and abstract interaction techniques in virtual environments. In *Proc. of HCI International*, no. 0106, 629–633.

HINCAPIÉ-RAMOS, J., AND GUO, X. 2014. Consumed Endurance: A metric to quantify arm fatigue of mid-air interactions. In *Proc. of CHI'14*, 1063–1072.

KATSURAGAWA, K., ET AL. 2016. Watchpoint: Freehand pointing with a smartwatch in a ubiquitous display environment. In *Proc. of AVI'16*, ACM, 128–135.

KHARLAMOV, D., ET AL. 2016. Ticktockray demo: Smartwatch raycasting for mobile hmds. In *Proc. of SUI'16.*

KIM, D., ET AL. 2012. Digits: Freehand 3D Interactions Anywhere Using a Wrist-worn Gloveless Sensor. In *Proc. of UIST'12*.

LIANG, J., AND GREEN, M. 1994. JDCAD: A highly interactive 3 D modeling system. *Computers & Graphics 18*, 449–506.

PIETROSZEK, K., ET AL. 2014. Smartcasting: A Discount 3D Interaction Technique for Public Displays. In *Proc. OzCHI '14*.

ZHAI, S. 1995. Human Performance in Six Degree of Freedom Input Control. *Industrial Engineering*, 179.

---

[2] Code at https://github.com/gameresearchlab/TickTockRayPlugin