Journal of Multimedia Information System VOL. 5, NO. 1, March 2018 (pp. 59-62): ISSN 2383-7632(Online)
http://dx.doi.org/10.9717/JMIS.2018.5.1.59

Brief Paper:
FPGA-based Hardware Prediction Rendering for Low-Latency Touch Platform

Seok Bin Song[1], Jin Heon Kim[2*]

Abstract: The delay between input action and visual interface feedback (“Latency”) in a touchscreen inking task reduces the user's performance. When the latency is less than 2.38ms, the user cannot perceive the latency in dragging task. This value is difficult to achieve on recent touchscreens and general purpose computers. So, methods of predicting touch points to reduce perceptible latency has been proposed. In general, touch points prediction is not perfect. When using point prediction, feedback of the predicted points is displayed on the screen, after a while, erased when the actual points are displayed. When this task is implemented by software, it causes additional latency to work to erase predicted points feedback. It therefore propose a platform for rendering point prediction feedback without additional latency by the FPGA. This platform transmits input points and HDMI signals rendering feedback of input points to the FPGA. The FPGA draws the feedback of points predicted based on the input points on the HDMI and displays the screen. Since hardware rendering changes the HDMI signal every frame, it does not require erasing work and rendering can be done within an early time regardless of the amount of rendering, so we will reduce the latency.

Key Words: Touch screen, Latency, Prediction, FPGA.

I. INTRODUCTION

Touchscreens are a type of interactive system. Touchscreens have direct touch devices and indirect touch devices. Many direct-touch devices such as smart phones, tablets, PCs, etc. are already familiar Human Input Devices (HID) [8], since the user feels the more intuitively direct-touch device than the indirect-touch device [2]. All interactive systems have a delay between input action and feedback, which is called "latency" [5]. delay time is an important issue of interactive systems and must be solved [1]. In the case of a dragging action which is one of the touch actions, the user responds most sensitively to the latency. When the user dragging with the direct-touch device, it is difficult to perceive the latency when the latency is 11ms or less. When the latency is 2.38ms or less, the latency is not perceive [3]. However, the recent direct-touchscreen device has a latency of 50ms to 200ms, so the latency appears visibly [4]. Latency in a direct-touch device has many sources, usually with three main components: 1) the physical sensor that captures touch points; 2) the software that processes touch events and render feedback; 3) the display itself [5]. Software feedback requires the most latency among the three components. Many methods have been studied for predicting touch points as a method for reducing the latency of software feedback [6].

We propose to change touch point prediction feedback rendering using conventional software rendering to hardware rendering using FPGA. In general, touch points prediction is not perfect. When using point prediction, feedback of the predicted points is displayed on the screen, after a while, erased when the actual points are displayed. When this task is implemented by software, it causes additional latency to work to erase predicted points feedback. When the FPGA modifies the HDMI signal directly, the rendering time is reduced and the predicted rendering is removed automatically when the next frame is displayed [7]. Because the OS uses the frame buffer on the memory to display the screen. However, the FPGA does not use memory to directly modify HDMI. Therefore, unless we modify the HDMI signal with the FPGA when outputting the next frame and displaying it on the OS, the previously

Manuscript received March 10, 2018; Accepted March 19, 2018. (ID No. JMIS-2018-0019)
Corresponding Author (*): Jin Heon Kim, Address : (02713) Seogyeong-ro 124, Seongbuk-gu, Seoul, Republic of Korea, TEL : +82-2-940-7747, jinheon@skuniv.ac.kr.
[1]Dept. of Computer Engineering, Seo-Kyeong University, Seoul, Republic of Korea, sukbin313@skuniv.ac.kr
[2]Dept. of Electronic Computer Engineering, Seo-Kyeong University, Seoul, Republic of Korea, jinheon@skuniv.ac.kr

rendered data disappears. Hence, this method works because hardware rendering does not cause additional latency when displaying feedback of predicted touch points.

II. RELATED WORK

There are many studies to numerically express the effect of touch latency. In these studies, it is possible to know the side-effects of touch latency on users and delay time with effective performance [3,5,10,12]. For each touch action such as tapping and dragging, the latency that the user perceives as reliable is different. Among them, in the case of dragging that responds most sensitively to the user, if it is 11ms or less, it shows high reliability and does not perceive latency when it is 2.38ms or less [5].

Touch coordinate prediction techniques are described in many papers and patents [9,10,11,22]. For example, Taylor series, Kalman filter, Curve fitting, Heuristic, linear short-term, and Quadratic are examples of typical points prediction algorithms [11,13,14]. All prediction algorithms have less accuracy as the prediction distance increases. Hence, the setting of the predicted distance should be carefully considered [14].

Hardware rendering transmits rendering information such as touch coordinate information, color, size, and HDMI signals to the board equipped with the FPGA is mounted. The transmitted touch point and rendering data directly modifies the HDMI signal through the pipeline structure [15] and outputs it to the display device. The method of modifying the HDMI signal calculates the pixel range of the coordinates to be changed by using the information of the input touch point and size. Then, when it is within the changed pixel range as compared with the currently updated pixel position, the color of the pixel is changed to the color of the touch pen with the color of the original image. This method has a latency of less than 1ms with parallel processing using hardware. It is more effective for embedded systems with less delay time than software feedback rendering and without a graphics processing unit (GPU) [7].

III. SYSTEM CONFIGURATION

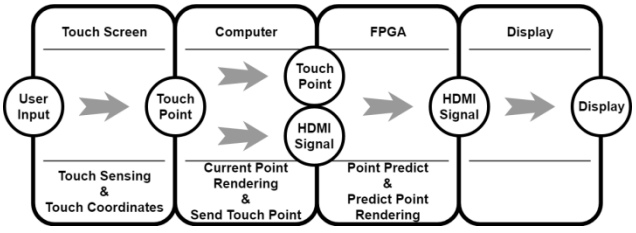

Fig. 1. System configuration diagram.

Figure 1 shows the process of displaying the user's touch input to the display. The conventional touch interactive system is as follows. 1) It senses the touch on the touchscreen, coordinates the touch points, and sends it to the computer. 2) The computer renders feedback of received points.

We added hardware predictive rendering to the existing software touch rendering system. In order to perform hardware prediction rendering, first, HDMI signals and rendering information such as touch point, color, size, etc. are transmitted to the board equipped with the FPGA. The feedback of the points predicted based on the received points directly modifies the HDMI signal.

The hardware of the platform is composed of AFO 65inch large touchscreen [19], Raspberry Pi 3 Model B Computer by Raspberry Pi Foundation [18], Digilent's Nexys Video Board [17] with Xilinx's Artix-7 xc7A200T-1SBG484C FPGA [16], and Samsung's 65inch LED Display [21].

IV. EXPERIMENT

We execute the touch rendering-software in an environment where Raspbian OS is installed in Rasspberry Pi 3 Model B. This software is a touch rendering software that uses the Qt graphics framework [20] and requires a 2ms latency when rendering a 10 x 10 pixel point. It is a very simple program to render touch point feedback and send points, rendering time is short. Commercial programs support a lot of work including inking work, so the rendering latency is much longer. The touchscreen requires a total 5ms latency in touch points coordinates and transmission [19]. So the total rendering time is 7ms. When prediction algorithms are implemented in software, an additional latency of 4ms occurs for each prediction point. In order to visually confirm the touch latency, we shot at 240 frames per second (FPS) using iPhone X's slow motion movie shooting function [23]. Touch points prediction uses a linear short-term algorithm that is simple to implement and has effective performance [14].

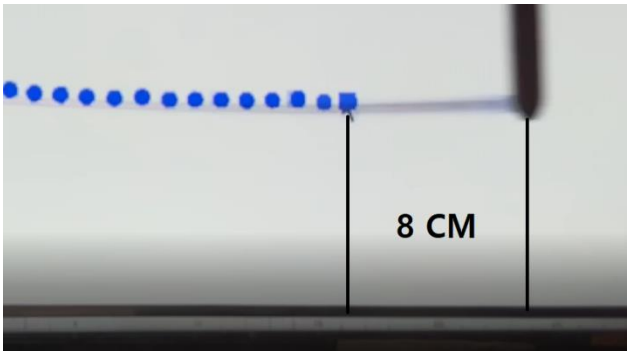

Fig. 2. Software rendering.

Figure 2 shows only software rendering without hardware prediction rendering. We can perceive the

difference between actual touch point and rendered feedback.

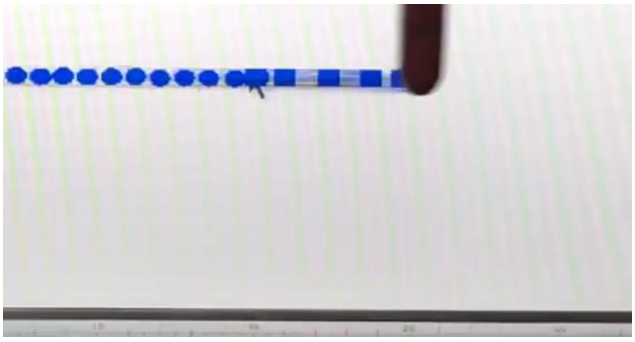

Fig. 3. Software rendering and hardware prediction rendering.

Figure 3 shows a combination of software rendering and hardware prediction rendering. The predicted points are five. When rendering this work in software, additional memory is required and the latency is increased by 20ms. However, regardless of the number of points, hardware rendering has latency of 0.74ms and does not require additional memory [7]. In Figure 2, the distance between the feedback and the actual touch point is noticeable. However, in Figure 3, it appears that there is almost no distance between the actual touch point and the feedback. This is because the touch points were predicted and rendered.

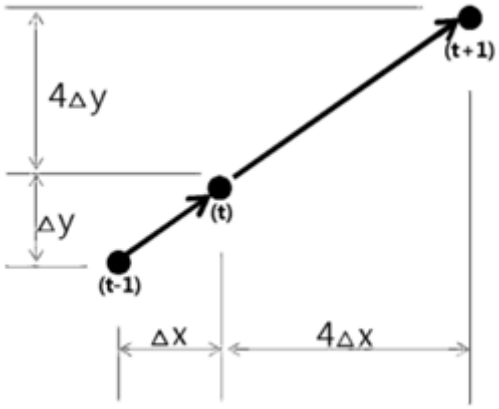

Fig. 4. Touch point predict algorithm.

Figure. 4 is a diagram representing a method of predicting touch point. In the two-dimensional plane, let the previous touch point be t-1 and the current touch point be t. It is possible to know the amount of change in the X point and the amount of change in the Y point via the previous touch point and the current touch point. This change amount can be applied to the current point to calculate the predicted point t + 1. Through this experiment, we can see that predictive rendering of touchpoints is visually effective.

Table. 1. Latency differences between hardware prediction line rendering and software prediction line rendering.

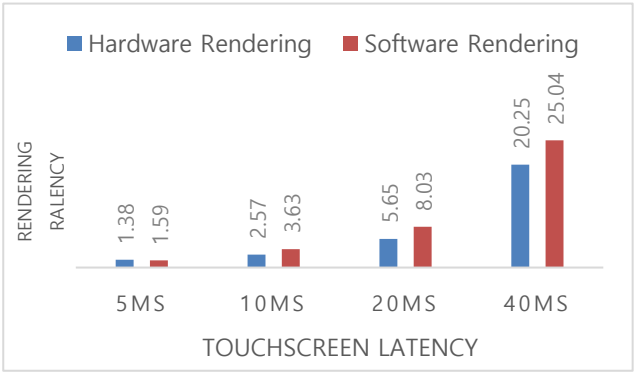

Table 1 shows the total latency difference between software prediction line rendering and hardware prediction line rendering due to touchscreen latency. The latency of each experiment is including latency of software touch line rendering. The vertical axis is the sum of rendering times excluding the touchscreen latency. This experiment generates virtual touch events that drag 1200 pixels per second. After 200 experiments, we measured the average of the latency required each time a touch event was generated. Hardware prediction line rendering is not yet implemented. However, since hardware rendering always requires the same latency regardless of the amount of rendering, we can assume that the hardware predicted line rendering latency is 1ms. In Table 1, we can see that the longer latency of the touchscreen, the greater the latency difference between hardware predictive rendering and software predicted rendering when performing predictive rendering. As the latency of the touchscreen increases, the distance between points also increases, so the prediction distance also increases. As the prediction distance increases, the amount of rendering increases, so hardware prediction is less latency than software prediction.

V. CONCLUSION

Comparing Figure 2 and Figure 3, we can see that the difference between the two pictures is clear and prediction rendering is effective. Table 1 shows that hardware prediction rendering requires less latency than software prediction rendering. Hence, Hardware predictive rendering using FPGA is effective in reducing latency. Hardware rendering always requires the same latency regardless of graphics processing speed. Hence, It is better to apply it to an embedded system with low graphics processing speed.

We plan to study more precise touch points prediction algorithm, multi-touch prediction, hardware

implementation of prediction algorithm, and hardware line rendering.

Acknowledgement

This work was supported by the Technology Innovation Program (10062411, Developed large-size touch interactive with the fastest response speed in the world) funded By the Ministry of Trade, Industry & Energy (MOTIE, Korea)

REFERENCES

[1] Meehan, M., Razzaque, S., Whitton, M. C., and Brooks, F. P., "Effect of Latency on Presence in Stressful Virtual Environments," *IEEE VR*, pp. 138-141, March 2003.

[2] K. Kin, M. Agrawala, and T. DeRose, "Determining the Benefits of Directr-touch, bimanual, and Multifinger Input on a Multitouch Workstation." in *Proceeding of Graphics Interface*, pp. 119-127, May. 2009.

[3] R. Jota, A. Ng, P. Dietz, and D. Wig- dor, "How fast is fast enough?: User Perception of Latency & Latency Improvements in direct and Indirect Touch," in *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing System*, pp 1827–1836, April. 2015.

[4] Ng, A., Lepinski, J., Wigdor, D., Sanders, S., & Dietz, P., "Designing for low-latency direct-touch input." in *Proceedings of the 25th annual ACM symposium on User interface software and technology*, pp. 453-464, October. 2012.

[5] Jota, R., Ng, A., Dietz, P., & Wigdor, D., "How fast is fast enough? : a study of the effects of latency in direct-touch pointing tasks", in *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pp. 2291-2300, April. 2013.

[6] Takeshi Asano, Ehud Sharlin, Yoshifumi Kitamura, Kazuki Takashima, and Fumio Kishino. "Predictive interaction using the delphian desktop," in *Proceedings of the 18th Annual ACM Symposium on User Interface Soft- ware and Technology*, pp. 133–141, October. 2005.

[7] Jun Han Yoon and Jin Heon Kim, "An Implementation of High Speed Rendering to Process Touch Screen Multiple Inputs Based on FPGA," in *procedding of the Journal of Korea Multimedia Society 20(11)*, pp. 1803-1810, November 2017.

[8] J. S Park, J. M. Lim, and K. W. Kyeong, "Tangible Touch Interface Technology Trend," *Korean Information Processing Society Review*, Vol. 20, No 1, pp. 45-53, 2013

[9] Kim, B., and Lim, Y. "Mobile terminal and touch coordinate predicting method thereof," *WO Patent App*, Aug. 2014.

[10] LINCOLN, J. "Position lag reduction for computer drawing," *US Patent App*, Oct. 2013

[11] Mathieu Nancel, Daniel Vogel, Bruno De Araujo, Ricardo Jota, and Géry Casiez, "Next-Point Prediction Metrics for Perceived Spatial Errors," in *Proceedings of the 29th Annual Symposium on User Interface Software and Technology*, 2016.

[12] Kaaresoja, Topi Johannes, "Latency guidelines for touchscreen virtual button feedback," School of Computing Science College of Science and Engineering, University of Glasgow, PhD thesis, March 2015

[13] Joseph J. LaViola., "Double Exponential Smoothing: An Alternative to Kalman Filter-based Predictive Tracking," In *Proceedings of the Workshop on Virtual Environments 2003 (EGVE'03)*, pp. 199–206. 2003

[14] David Asselborn, and Jan Borchers, "A Predictive Approach to Compensate Latency of Tangibles on Capacitive Multi-Touch-Displays," Media Computing Group, Computer Science Department PWTH Aachen University, July 2016

[15] J.W. Park, J.Y. Ko, J.H. Park, M.H. Hong, Y.H. Lee and J.C. Shim, "A Wireless Temperature Control System based on FPGA," *Journal of Korea Multimedia Society*, Vol. 15, No. 7, pp. 920-930, 2012

[16] Xilinx, http://www.xilinx.com/products/silicon-devices/fpga/artix-7.html#documentation, (accessed Jul. 8. 2018).

[17] Digilent, https://store.digilentinc.com/nexys-video-artix-7-fpga-trainer-board-for-multimedia-applications/ (accessed March. 8. 2018)

[18] Raspberry Pi Foundation, https://www.raspberrypi.org/products/ (accessed March. 8. 2018)

[19] AFO, http://afoi.co.kr/business/module.php, (accessed March. 8. 2018)

[20] Qt, http://wiki.qt.io/Main (accessed March. 8. 2018)

[21]Samsung, https://www.samsung.com/us/business/products/displays/standalone/ed-series/ed-e-series-65-lh65edeplgc-go/?CID=AFL-hq-mul-0813-11000758 (accessed March. 8. 2018)

[22] Peter Tsoi and Jacob Xiao. "Advanced touch input on iOS," Technical report, Apple Inc., 2015.

[23] IPhone X, https://www.apple.com/kr/iphone-x/specs/ (accessed March. 8. 2018)