Journal of Korea Multimedia Society Vol. 20, No. 11, November 2017(pp. 1803-1810)
https://doi.org/10.9717/kmms.2017.20.11.1803

FPGA 기반의 터치스크린 다중입력처리를 위한 고속 렌더링 구현

윤준한†, 김진헌††

An Implementation of High Speed Rendering to Process Touch Screen Multiple Inputs based on FPGA

Junhan Yoon†, Jin Heon Kim††

ABSTRACT

A large amount of processing time is required if the process of detecting the touch position on the touch screen and displaying it on the display panel is performed only by software. In this paper, we propose a method to output information touched on the screen using H/W method in order to improve the response speed delay. In the FPGA module designed for the HDMI signal output to the display module, the touch information is input to the serial data signal including touch coordinate information, point size, and color information. Then the module render the image using HDMI signal input to the module and the touch information. This method has a pipeline structure so it has effect of reducing the delay time that occurs in outputting the touch information compared with the conventional software processing method.

Key words: Touch Screen, HDMI, Delay, H/W, FPGA

1. 서 론

터치스크린에서 정보를 입력받는 방식은 크게 간접 터치 방식과 직접 터치 방식으로 나누어진다. 간접 터치 방식은 대표적인 예로서 디스플레이에 리모컨과 같은 컨트롤러 기기를 이용하여 커서 또는 포인터의 위치를 조절하도록 간접적으로 구현하는 방식이다. 반면 직접 터치 방식의 경우 디스플레이 화면 전면에 터치 모듈을 부착하고, 사용자가 직접 손으로 터치 모듈을 터치하여 정보를 입력하는 방식을 의미한다. 휴먼 인터페이스 기기가 대두되면서, 사용자가 직접 화면을 터치하여 정보를 전달 또는 습득하는 것이 간접 터치 방식보다 직감적이기 때문에[1] 직접 터치 방식을 선호하는 추세이고, 태블릿 PC나 스마트 폰과 같은 각종 모바일 디바이스에서 직접 터치 방식의 터치스크린은 이미 친숙한 HID(Human Input Device)로 자리매김하고 있다[2].

이러한 직접 터치 방식에도 여러 가지 해결해야할 문제들이 있는데, 대표적으로 손가락의 움직임 정보를 입력받아 디스플레이하기까지 지연시간(Latency)이 존재한다는 점이다. 특히 정보 입력 단계와 출력 단계가 만들어내는 지연 시간이 간접 터치 방식에 비해 매우 두드러진다. 초고속 카메라를 이용해 실험한 결과, iOS, Android, Windows 등의 OS가 탑재된 상업제품들의 경우 50~200ms의 지연시간이 존재함을 확인하였다[4]. 이전 연구에 따르면 사용자가 손

※ Corresponding Author : Jin Heon Kim, Address: (02713) Seogyeong-ro 124, Seongbuk-gu, Seoul, Korea, TEL : +82-2-940-7747, E-mail : jinheon@skuniv.ac.kr
Receipt date : Sep. 4, 2017, Revision date : Oct. 30, 2017
Approval date : Nov. 1, 2017

† Dept. of Electronic Computer Engineering., Graduate School, Seo-Kyeong University
(E-mail : junhan9111@naver.com)
†† Dept. of Electronic Computer Engineering., Graduate School, Seo-Kyeong University

가락을 이용해 터치스크린을 컨트롤할 경우 가장 많이 하는 행위인 탭핑(Tapping)은 20ms, 드래깅(Dragging)은 2.38ms 이하의 좌표 정보 처리 시간이 소요될 때 시간지연(Time lag)을 느끼지 못한다[11]. 따라서 현재 사용되는 터치 정보 처리 방식으로는 사용자가 터치스크린을 사용하면서 느끼는 불편함을 완전히 해소하기 어렵다. 사용자가 입력받은 터치 좌표 정보를 드래깅할 경우, 지연시간으로 인해 실제 손가락이 터치하는 위치(actual position)보다 이전의 터치 좌표가 디스플레이 된다.

본 논문에서는 대표적인 지연 시간 요소인 터치 감지, 감지한 터치를 소프트웨어적으로 렌더링, 디스플레이의 세 가지 요소[11] 중 소프트웨어적인 렌더링 대신 하드웨어적인 렌더링을 사용하여 시간지연을 개선하는 방법을 제안한다. 제안한 하드웨어적인 렌더링은 FPGA(Field Programmable Gate Array)의 파이프라인(Piepeline) 및 병렬 처리 구조를 이용하고, 이를 HDMI(High-Definition Multimedia Interface) 신호를 통해 고속 출력하는 방식이다. 본 논문은 다음과 같이 구성되어 있다. 제 2장에서 기존의 관련연구들을 살펴보고, 제 3장에서 제안된 하드웨어 처리방식을 설명한다. 제 4장에서 구체적인 하드웨어 처리방식을 설명하고 제 5장에서 제안된 하드웨어 처리 방식과 기존 기법들을 실험을 통해 검증한다. 제 6장에서는 연구에 대한 결론을 맺는다.

2. 관련연구

일반적으로 지연시간에 대한 고찰 시, 우리가 어떤 움직임에 대해 인식하기 위해서는 모니터 상의 재생률이 최소 10Hz 이상이어야 한다는 Card, Mackinlay, Robertson[12]의 초기 이론이 인용된다. 이는 Moran, Newell[13]의 인식처리 및 이에 대한 선행연구인 시각적 인식에 대한 이론을 배경으로 한다. 후에 이를 바탕으로 100ms 이상의 지연시간은 사용자가 사용하기 힘들다는 100ms 규칙이 만들어지는 계기가 된다. Miller[14]는 라이트펜(light pen)을 이용하여 특정 모양을 의도적으로 느리게 그림으로써 사용자가 지연시간이 100ms인 상태에서 터치스크린을 이용할 만하다고 주장했으나 이를 정확한 수치나 지표로 제시하지 못함으로써 입증하지 못하였다. Anderson, Doherty, Ganapathy[15] 또한 웹 서핑(Web Surfing)과 같은 다양한 작업을 터치스크린 상에서 실행하여 580ms 이상의 지연시간 상에서는 사용자가 이용하기 불편하다고 했으나, 실험방법이 대표적으로 확대(zooming), 패닝(panning), 페이지 넘김(page-turning) 등으로 매우 간단하여 입증에 어려움이 있었다. 앞의 연구들이 지연시간에 대한 인식 수준을 명확히 증명해내기 어려웠던 반면, Ricardo Jota는 실험 참가자들을 모집하여 반응속도를 다양화한 터치스크린에 대해 각각 두 번의 터치 후, 반응속도 및 어떤 것이 더 빠른지에 대한 것을 기록하게 하는 실험을 통해 탭핑은 20ms 이하, 드래깅은 2.38 ms 이하일 때 시간지연을 느끼지 못한다는 결론을 내렸다.

사람들이 가지고 있는 터치스크린 지연시간의 지각 범위에 대한 연구와 동시에 직접 터치 방식의 지연시간을 줄이는 방법에 대해 연구들이 진행되어 왔다. 터치 센서에서 정보를 입력 받고 디스플레이하기 위해서는 입력받은 터치 정보와 현재 디스플레이 되는 화면을 결합하여 렌더링 하는 작업이 필요하다. 이를 실현하기 위해 기존의 소프트웨어적인 방식에서는 OS(Operating System) 상에서 사용자용 어플리케이션 프로그램을 통해 그래픽 컨트롤러를 제어하는데 이들에 대한 다양한 지연시간이 존재한다. 먼저 OS에서 터치스크린 정보처리 외 다른 작업이 가중될 경우, 프로세서는 작업의 우선순위를 판단하기 때문에 추가적인 지연시간을 가져올 수 있다. 또한 터치 정보에 대한 이벤트 감지 및 처리를 위해 쓰레드 프로그래밍을 사용하게 되는데, 일반적으로 한 번에 하나의 이벤트만 처리하는 구조를 갖고 있다. 예를 들어, 터치 정보에 대한 이벤트를 감지하였으나, 타 작업에 대한 쓰레드 처리가 끝나지 않았으므로 즉각적으로 터치 정보를 처리하지 못한다. 이런 플랫폼의 여러 가지 환경적 이유로, 이들 요소에 대한 직접적인 지연시간을 줄이는 방법 대신, 사용자의 시각적 인지에 기반한 지연시간을 줄이는 방법으로 시각적인 속임수를 쓰는 방법이 있다. 드래깅을 예로 들면 지연시간으로 인해 실제 터치한 좌표의 위치보다 이전인 좌표의 위치가 디스플레이 되지만, 다음 터치할 좌표를 미리 예측한다면 그 지연시간으로 인해 시각적으로는 현재 좌표를 디스플레이 하는 것처럼 보일 수 있다. 이에 대해 다음 터치 좌표를 예측하는 방법에 대해 칼만 필터(Kalman Filter)[16], 뉴럴 네

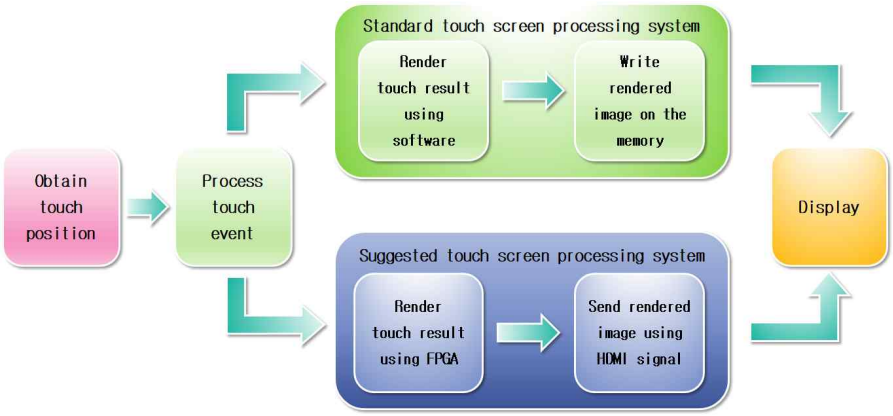

Fig. 1. Flowchart for comparison of touch screen processing system.

트워크(Neural Network)[17]와 같은 방법들을 이용한 연구들이 진행되고 있으며, 이 외에 터치스크린 모듈의 디지털 제어 회로의 응답속도를 빠르게 하는 방법이 있는데 물리적 터치에 해당하는 커패시턴스 값(Capacitance value)을 선형적인 방법이 아닌 적응형으로 탐색하여 응답 속도를 빠르게 한다[5].

3. 하드웨어 렌더링

Fig. 1에서 기존의 소프트웨어적인 방식은 터치스크린 모듈(Touch screen module)을 통해 터치 좌표 정보를 입력 받은 후 이를 이벤트 관리자(Event management)를 통해 응용프로그램(Application software)으로 전달한다. 좌표 정보를 입력받은 응용프로그램은 그래픽 프레임워크(Graphics frame-work)를 통해 터치 좌표 정보를 렌더링(Rendering) 후 영상 메모리에 렌더링 이미지를 쓴다. 마지막으로 디스플레이 모듈(Display module)를 통해 터치 좌표 정보가 입력된 렌더링 이미지를 출력한다.

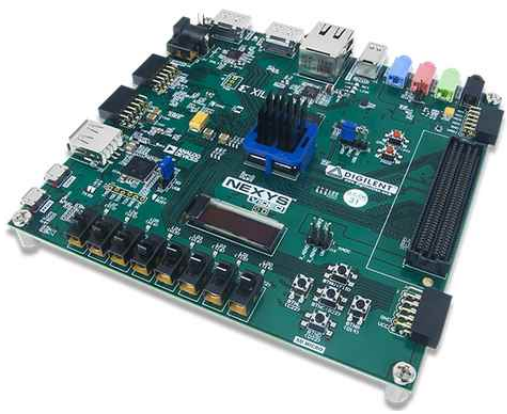

Fig. 2. Appearance of Nexys Video.

본 논문에서는 FPGA에서 입력된 터치 좌표 정보를 영상 메모리 없이 직접 처리하여 디스플레이하는 하드웨어적인 처리 방식을 제안한다. 터치스크린 모듈을 통해 터치 좌표 정보를 입력 받고, 터치 정보에 대한 색깔 및 크기를 사용자로부터 입력받아 좌표 정보와 함께 직렬 통신 인터페이스를 통해 FPGA로 전송한다. FPGA에서는 입력받은 그래픽 정보와 좌표 정보를 파이프라인 구조[18]를 통해 렌더링 후, HDMI 신호에 직접 관여하여 디스플레이 장치로 픽셀 데이터를 출력함으로써 처리 시간이 기존 방식에 비해 적게 소요되고, 하드웨어 병렬 처리 방식의 이점을 통해 다중 입력처리시 지연시간이 기존의 소프트웨어적인 방식에 비해 적게 소요된다. 또한 그래픽 프로세서와 같은 모듈을 이용할 필요가 없어 저가형 임베디드 시스템 구성에 용이하다.

4. 시스템 구성 및 제안 방식

제안한 하드웨어 방식의 처리기법은 Digilent 사의 Nexys Video[6]를 이용하여 구성되었다. Xilinx 사의 Xilinx Artix-7 FPGA XC7A200T-1SBG484C [7]이 탑재되어 있으며, 디스플레이의 해상도 사양은 1080p@60Hz로 설정하였다. 이 때 픽셀 클록(Pixel Clock)은 148.5MHz, 클록 당 소요 시간은 약 6.7ns 이다. 다음은 Nexys Video의 외관이다.

제안한 하드웨어 방식을 구현한 전체 시스템 구조

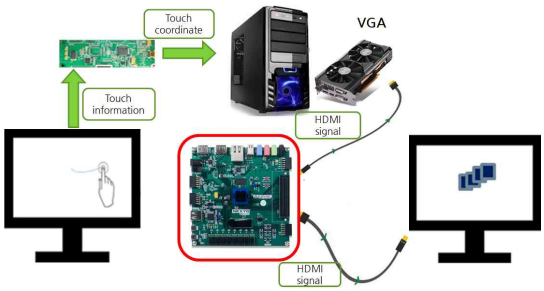

Fig. 3. System configuration diagram.

는 다음과 같다.

Fig. 3에서 왼편의 터치스크린에서 물리적 터치 정보를 입력 받으면 이를 터치 좌표로 환산하여 PC (Personal Computer) 또는 임베디드 보드(Embedded Board)로 보낸다. 이후 터치좌표와 그래픽 모듈에서 HDMI 신호를 통해 보내는 원본 영상을 FPGA에서 입력받아 렌더링 후 다시 HDMI 신호를 통해 디스플레이로 출력한다. 다음은 본 시스템의 블록 다이어그램이다.

Fig. 4에서 터치 입력이 들어왔을 때, PC 또는 임베디드 시스템에서 터치모듈(Touch Module)로부터 터치 정보(Touch inform)를 USB를 통해 입력받고 UART를 사용한 직렬통신 인터페이스를 이용해 FPGA로 전송한다. 다중 터치 입력의 경우 소프트웨어를 통해 각 터치에 대해 고유 ID(Identification)를 부여하여 차례대로 전송한다. 이 때 입력받은 위치 좌표를 현재 출력되는 원본 영상(Original image)의 픽셀 위치와 비교하여 일치하면 입력받은 그래픽 정보를 이용해 렌더링을 진행한다. 렌더링이 끝난 이미지를 HDMI 신호를 통해 디스플레이 모듈로 출력한다. FPGA의 내부구성은 다음과 같다.

Fig. 5에서 HDMI 디코더를 통해 입력되는 원본 영상의 현재 픽셀을 카운트하여 위치 정보를 추출한다. 픽셀을 카운트하는 속도는 픽셀 클록(Pixel clock)에 동기화 하였으며, UART로부터 입력받은 터치 정보를 클록 스피드(Clock speed)가 200MHz인 내부의 마이크로 컨트롤러(Micro controller)를 통해 입력받아 위치 좌표(Touch position) 및 그래픽 정보(Touch color, thickness)를 비교 및 렌더링 모듈(Compare and Render module)의 입력 값으로 넣어준다. 이 때 다중 터치 입력의 경우 터치 입력을 받는 여러 개의 전용 레지스터를 만들어, 앞에서 부여된 고유 ID 전용 레지스터에 배속 후, 비교 및 렌더링 모듈로 전송한다. 이후 렌더링 된 이미지의 픽셀들을 HDMI 인코더를 통해 출력한다. 비교 및 렌더링 모듈의 동작 구조는 다음과 같다.

Fig. 6에서 입력받은 터치 위치 좌표(Touch position)와 이를 표현할 크기 정보(Pointer size)를 이용해 표현할 좌표의 픽셀 범위(Pointer pixel position)를 계산한다. 이를 현재 픽셀 위치(Present position)와 비교하여 나온 일치 신호(Sel)를 통해 해당 픽셀 위치의 픽셀 값을 원본 영상 픽셀 값(Original image color) 또는 터치 좌표 픽셀 값(Replace color)으로 결정하여 해당 픽셀 값(Selected rgb)으로 출력한다. 이 때 다중 터치 입력의 경우 여러 개의 좌표 픽셀 범위를 픽셀 비교기(Pixel compare module)의 조건으로 입력한다. 다음은 FPGA의 리소스(Resource) 활용도이다.

Fig. 7에서 (a)는 LUT(Look Up Table)와 FF(Flip Flop), (b)는 기타 리소스 사용량을 나타낸다. Nexys Video에서 지원하는 전체 리소스 양을 Available 이라 하고, Fig. 5와 같은 시스템 구성에 활용된 리소스

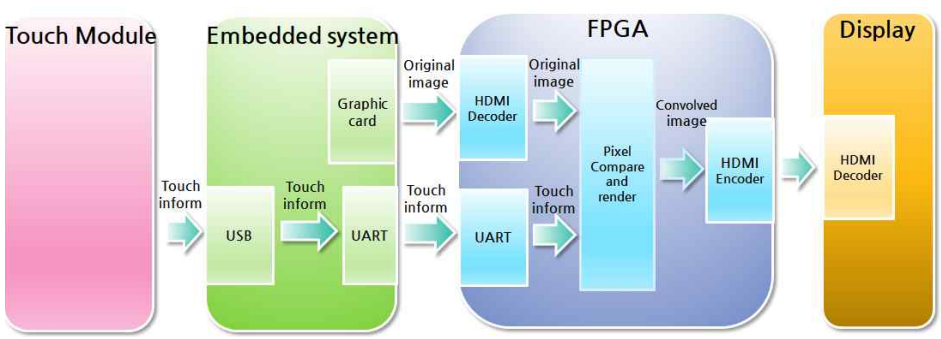

Fig. 4. System block diagram.

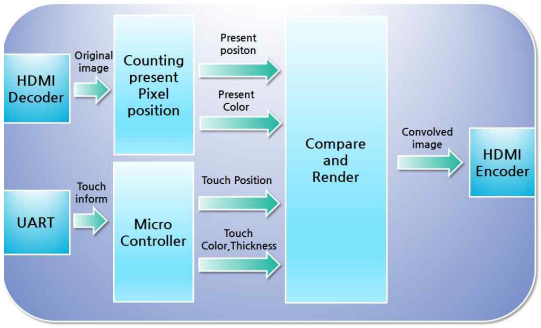

Fig. 5. FPGA block diagram.

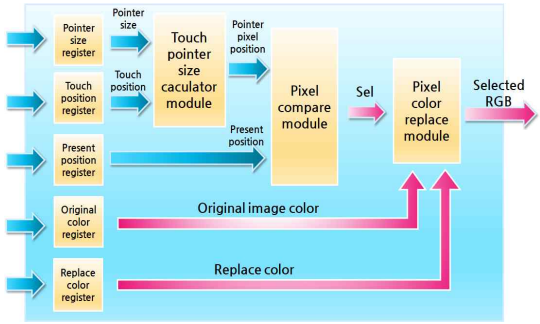

Fig. 6. Compare and Render block diagram.

양을 Utilization 이라 하였다. 이 때 Micro controller와 HDMI 인코더, 디코더와 같은 요소를 제외한 Fig. 6과 같은 로직에 사용된 리소스 양은 LUT 기준 2106개, FF 기준 2599개로 적은 수의 리소스 양으로 시스템 구성이 가능함을 확인할 수 있다.

5. 실험 및 분석

기존 소프트웨어 처리 방식과 제안된 하드웨어 처리 방식에 대해 다중 입력 터치 수에 따른 처리시간을 비교하였다. 전체 과정 중 터치 정보 및 그래픽 옵션 정보를 받고 디스플레이 하기 전까지 처리하는 시간으로 제한하였으며, 이전 과정인 터치 정보 입력 및 전달 시간과 이후 과정의 디스플레이 시간은 동일한 것으로 한다. 다음은 실험 환경이다.

소프트웨어 방식은 대표적인 그래픽 라이브러리인 OpenGL[9], OpenCV[8], SDL[10]을 이용하였으며, 소프트웨어 방식과 하드웨어 방식 모두 어플리케이션 프로그램을 이용하여 임의의 터치 좌표를 제공한다. 렌더링에 걸리는 시간은 클록을 이용한 타이머 함수를 이용하였고, 매 실험마다 처리 시간이 조금씩 다르므로 각각 100회의 실험에 대한 평균 시간을 측정하였다. 결과는 다음과 같다.

Table 2에서 소프트웨어 방식의 OpenCV, SDL, OpenGL은 동시에 터치한 좌표의 개수가 많을수록 처리 시간이 늘어난 방면, 하드웨어 방식의 FPGA의 렌더링 시간은 파이프라인 및 병렬 처리 구조로 인해 입력받은 터치 좌표 개수에 관계없이 빠른 렌더링 속도를 보여주고 있다. 다음은 Table 2을 그래프로 나타낸 결과이다.

Table 1. Experimental environment

Processor	Video	Serial interface	Resolution
Intel Core i7-2006k 3.4GHz (8 CPUs)	Nvidia Geforce GT 440 Bus width : 128bit Memory : 512MB	UART Speed :115200bps	1080p@60Hz

Table 2. Measurement of processing time according to rendering method

Multi touch count	Standard processing system[ms]			Suggested processing system
	OpenCV[ms]	SDL[ms]	OpenGL[ms]	FPGA[ms]
1	9.523	3.098	1.551	0.74
2	19.534	8.164	2.989	0.74
3	28.154	11.989	4.551	0.74
4	38.722	15.557	6.784	0.74
5	47.651	20.254	6.788	0.74
6	56.124	23.28	8.852	0.74
7	69.198	23.544	9.689	0.74

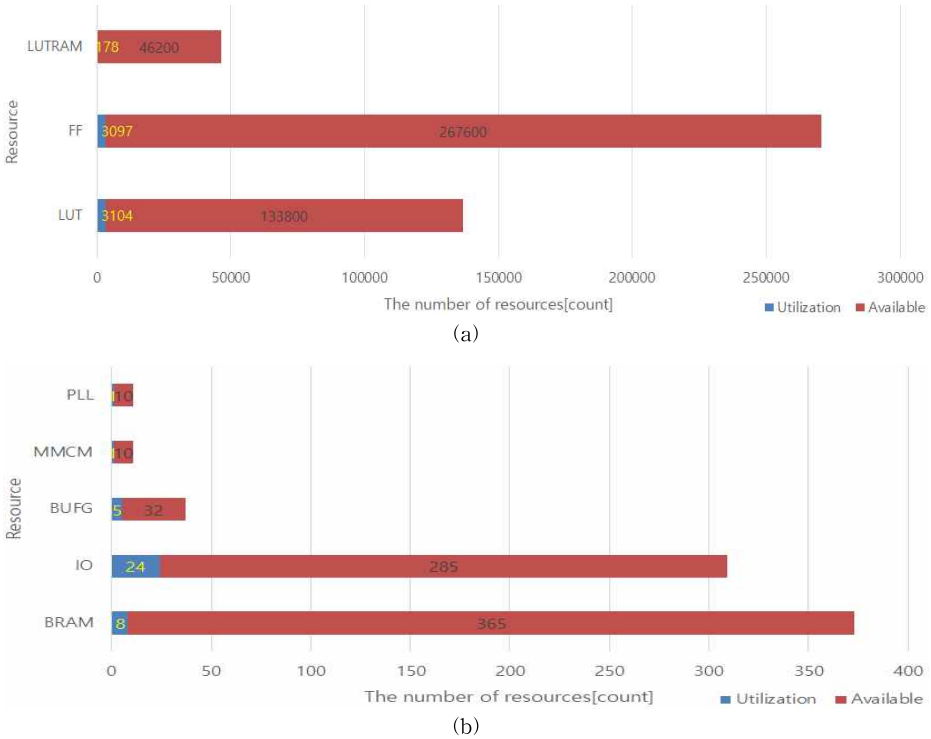

Fig. 7. FPGA resource utilization. (a) resource LUT and FF, (b) resource, Etc.

하드웨어 방식인 FPGA의 처리 속도를 보면, 입력 터치 정보가 Micro Controller 내부의 UART 로 보낼 때 걸리는 시간은 약 694.44μs, 다시 UART에서 렌더링 블록까지의 전송 시간은 평균 약 13.564μs, 내부 블록이 픽셀 클록에 동기화하여 동작하고 3~4의 클록이 소요되므로 단일 터치 처리 시 약 1ms 미만의 시간이 소요된다. 다중 입력 터치 처리의 경우 모듈 병렬 연산에 의해 처리 시간이 약 1ms 미만이 소요된다.

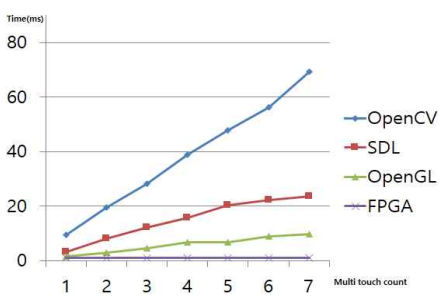

Fig. 8. Comparison of processing time according to rendering method.

6. 결 론

본 논문에서는 FPGA를 이용하여 터치스크린의 응답속도를 개선하는데 더하여 여러 개의 동시 입력 터치 좌표 처리까지 구현하였다. 영상 메모리를 이용하지 않고 하드웨어 방식으로 HDMI 인터페이스에 직접 관여하여 렌더링하기 때문에 동시에 입력하는 터치 정보가 많을수록 소프트웨어 처리 방식보다 효과적임을 확인하였다. 또한 구성 시스템의 리소스 양이 적어 효율적인 임베디드 시스템 구성이 용이할 것으로 보인다. 향후 다음 터치 좌표를 예측하여 미리 그려내는 기술 및 터치 좌표와 원본 영상의 자연스러운 렌더링 기술인 그라데이션(Gradation) 기능을 연구할 계획이다.

REFERENCE

[1] K. Kin, M. Agrawala, and T. DeRose, “Determining the Benefits of Direct-touch, Bimanual, and Multifinger Input on a Multitouch Workstation,” *Proceeding of Graphics Inter-*

face, pp. 119-124, 2009.

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

[3] P.A. Albinsson, "High Precision Touch Screen Interaction," *Proceeding of Special Interest Group on Computer Human Interaction Conference on Human Factors in Computing Systems,* pp. 105-112, 2003.

[4] A. Ng, J.L. Lepinski, D. Wigdor, S. Sanders, and P. Dietz, "Designing for Low-latency Direct-touch Input," *Proceeding of Association for Computing Machinery Symposium on, User Interface Software and Technology,* pp. 453-464, 2012.

[5] S.B. Park and J.H. Heo, "Implementation and Design of Control Circuit for Touch Screen with Faster Response Time," *The Journal of The Institute of Internet, Broadcasting and Communication,* Vol. 14, No. 1, pp. 155-159, 2014.

[6] Digilen, http://store.digilentin c.com/nexys-video-artix-7-fpga-trainer-board-for-multimedia-applications (accessed July, 16, 2017).

[7] Xilinx, http://www.xilinx.com/products/silicon-devices/fpga/artix-7.html#documentation (accessed Jul., 14, 2017).

[8] Wikipedia, http://en.wikipedia.org/wiki/Open CV (accessed Jul., 8, 2017).

[9] Wikipedia, http://en.wikipedia.org/wiki/Open GL (accessed Jul., 1, 2017).

[10] Simple Directmedia Layer, http://www.libsdl.org/ (accessed July, 3, 2017).

[11] R. Jota, A. Ng, P. Dietz, and D. Wigdor, "How Fast is Fast Enough?," *Proceeding of Special Interest Group on Computer Human Interaction Conference on Human Factors in Computing Systems,* pp. 2291-2300, 2013.

[12] S.K. Card, J.D. Mackinlay, and G.G. Robertson, "The Information Visualizer: An Information Workspace," *Proceedings of the Special Interest Group on Computer Human Interaction Conference on Human Factors in Computing Systems,* pp. 181-188, 1991.

[13] S.K. Card, T.P. Moran, and A. Newell, *The Psychology of Human-Computer Interaction*, L. Erlbaum Associates Incorporated Hillsdale, New Jersey, United States America, 1983.

[14] R.B. Miller, "Response Time in Man-computer Conversational Transactions," *Proceeding of Fall Joint Computer Conference,* pp. 267-277, 1968.

[15] G. Anderson, R. Doherty, and S. Ganapathy, "User Perception of Touch Screen Latency. Design," *Proceeding of International Conference of Design, User Experience, and Usability, Theory, Methods, Tools, and Practice,* pp. 195-202, 2011.

[16] Wikipedia, https://en.wikipedia.org/wiki/Kalman_filter (accessed Jul., 1, 2017).

[17] Wikipedia, https://en.wikipedia.org/wiki/Artificial_neural_network (accessed July, 1, 2017).

[18] 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.

윤 준 한

2017년 현재 서경대학교 컴퓨터공학과 석사과정
관심분야 : 디지털영상처리, 컴퓨터 비전, 영상신호처리(ISP)

김 진 헌

1982년 고려대학교 공과대학 전기공학과 공학사
1984년 고려대학교 대학원 전기공학과 공학석사
1983년 동양정밀공업(OPC) 중앙연구소 연구원
1986년 삼성종합기술원 선임연구원
1989년 ZyMOS 한국지사 FAE
1990년 고려대학교 대학원 전기공학과 공학박사
1995년 현재 서경대학교 컴퓨터공학과 부교수
관심분야 : 디지털영상처리, 영상신호처리(ISP), 영상/비디오시스템