



- (51) 국제특허분류:
A61B 8/14 (2006.01)
- (21) 국제출원번호: PCT/KR2015/009039
- (22) 국제출원일: 2015년 8월 28일 (28.08.2015)
- (25) 출원언어: 한국어
- (26) 공개언어: 한국어
- (30) 우선권정보:
10-2014-0114032 2014년 8월 29일 (29.08.2014) KR
- (71) 출원인: 서강대학교 산학협력단 (SOGANG UNIVERSITY RESEARCH FOUNDATION) [KR/KR]; 121-742 서울시 마포구 백범로 35, Seoul (KR).
- (72) 발명자: 송태경 (SONG, Tai-Kyong); 03010 서울시 종로구 평창문화로 156, 101 동 703 호, Seoul (KR). 공우규 (KONG, ukyu); 01388 서울시 도봉구 해동로 242-11, 103 동 304 호, Seoul (KR).
- (74) 대리인: 특허법인 충현 (CHUNG HYUN PATENT & LAW FIRM); 137-130 서울시 서초구 바우피로 225 한마음빌딩 4층, Seoul (KR).
- (81) 지정국 (별도의 표시가 없는 한, 가능한 모든 종류의 국내 권리의 보호를 위하여): AE, AG, AL, AM, AO,

AT, AU, AZ, BA, BB, BG, BH, BN, BR, BW, BY, BZ, CA, CH, CL, CN, CO, CR, CU, CZ, DE, DK, DM, DO, DZ, EC, EE, EG, ES, FI, GB, GD, GE, GH, GM, GT, HN, HR, HU, ID, IL, IN, IR, IS, JP, KE, KG, KN, KP, KZ, LA, LC, LK, LR, LS, LU, LY, MA, MD, ME, MG, MK, MN, MW, MX, MY, MZ, NA, NG, NI, NO, NZ, OM, PA, PE, PG, PH, PL, PT, QA, RO, RS, RU, RW, SA, SC, SD, SE, SG, SK, SL, SM, ST, SV, SY, TH, TJ, TM, TN, TR, TT, TZ, UA, UG, US, UZ, VC, VN, ZA, ZM, ZW.

- (84) 지정국 (별도의 표시가 없는 한, 가능한 모든 종류의 역내 권리의 보호를 위하여): ARIPO (BW, GH, GM, KE, LR, LS, MW, MZ, NA, RW, SD, SL, ST, SZ, TZ, UG, ZM, ZW), 유라시아 (AM, AZ, BY, KG, KZ, RU, TJ, TM), 유럽 (AL, AT, BE, BG, CH, CY, CZ, DE, DK, EE, ES, FI, FR, GB, GR, HR, HU, IE, IS, IT, LT, LU, LV, MC, MK, MT, NL, NO, PL, PT, RO, RS, SE, SI, SK, SM, TR), OAPI (BF, BJ, CF, CG, CI, CM, GA, GN, GQ, GW, KM, ML, MR, NE, SN, TD, TG).

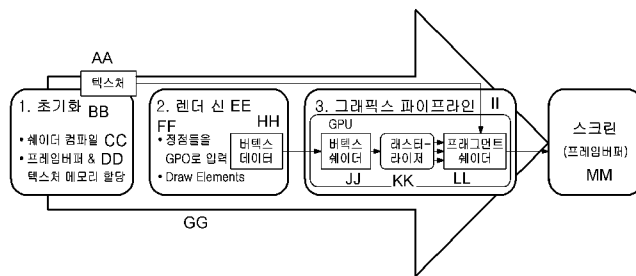
규칙 4.17에 의한 선언서:

- 신규성을 해치지 아니하는 개시 또는 신규성 상실의 예외에 관한 선언 (규칙 4.17(v))

[다음 쪽 계속]

(54) Title: METHOD FOR HIGH-SPEED PARALLEL PROCESSING FOR ULTRASONIC SIGNAL BY USING SMART DEVICE

(54) 발명의 명칭 : 스마트 기기를 이용한 초음파 신호의 고속 병렬 처리 방법



AA ... Texture
BB ... 1. Initialization
CC ... Shader compilation
DD ... Frame buffer and texture memory allocation
EE ... 2. Scene rendering
FF ... Input peaks into GPO
GG ... Draw elements
HH ... Vertex data
II ... 3. Graphics pipeline
JJ ... Vertex shader
KK ... Rasterizer
LL ... Fragment shader
MM ... Screen (Frame buffer)

(57) Abstract: The present invention relates to a method for high-speed parallel processing for an ultrasonic signal, the method used for generation of an ultrasonic image by a smart device, which is provided with a mobile graphic processing unit (GPU), by receiving an input of an ultrasonic signal. The method comprises the steps of: receiving an input of an ultrasonic signal beam-formed by means of a first rendering cycle, removing a DC component from the ultrasonic signal, and then separating an in-phase component and a quadrature component from the ultrasonic signal, from which the DC component has been removed, and separately outputting same; a smart device performing quadrature demodulation and envelope detection processing for the ultrasonic signal, having the in-phase component and the quadrature component, by means of a second rendering cycle; and the smart device performing scan conversion for the ultrasonic signal, which has been obtained as the result of the second rendering cycle, by means of a fifth rendering cycle, wherein the rendering cycles are formed as a graphics pipeline structure comprising a vertex shader procedure, a rasterizer procedure, and a fragment shader procedure. A method for high-speed parallel processing for an ultrasonic signal by using a smart device, according to the present invention, enables high-speed parallel processing for an ultrasonic signal by means of a mobile GPU inside a smart

device even in a mobile-based environment instead of a PC-based environment, thereby enabling the providing of an image having a frame rate that is useful for medical diagnosis.

(57) 요약서:

[다음 쪽 계속]

**공개:**

— 국제조사보고서와 함께 (조약 제 21 조(3))

본 발명은 모바일 GPU(Graphic Processing Unit)를 구비하는 스마트 기기가 초음파 신호를 입력 받아 초음파 영상을 생성하는데 이용되는 초음파 신호의 고속 병렬 처리 방법에 관한 것으로, 제 1 렌더 사이클(Render Cycle)을 통해 빔포밍된 초음파 신호를 입력받아, 상기 초음파 신호로부터 DC 성분을 제거한 후, DC 성분이 제거된 초음파 신호로부터 in-phase 성분과 quadrature 성분을 나누어 각각 출력하는 단계; 상기 스마트 기기가 제 2 렌더 사이클을 통해 상기 in-phase 성분과 quadrature 성분을 갖는 초음파 신호에 대하여 quadrature 복조 및 포락선 검파 처리를 수행하는 단계; 상기 스마트 기기가 제 5 렌더 사이클을 통해 상기 제 2 렌더 사이클의 처리 결과로서 획득된 초음파 신호에 대하여 스캔 변환을 수행하는 단계;를 포함하되, 상기 렌더 사이클들은, 버텍스 셰이더(vertex shader) 단계, 래스터라이저(Rasterizer) 단계 및 프래그먼트 셰이더(fragment shader) 단계를 포함하는 그래픽스 파이프라인(Graphics pipeline) 구조로 이루어진다. 이러한 구성에 의해, 본 발명의 스마트 기기를 이용한 초음파 신호의 고속 병렬 처리 방법은 PC 기반의 환경이 아닌 모바일 기반의 환경에서도 스마트 기기 내 모바일 GPU를 통해 초음파 신호에 대한 고속의 병렬처리를 수행함으로써, 의료 진단에 유용한 프레임율을 갖는 영상을 제공할 수 있는 효과가 있다.

명세서

발명의 명칭: 스마트 기기를 이용한 초음파 신호의 고속 병렬 처리 방법

기술분야

- [1] 본 발명은 스마트 기기를 이용한 초음파 신호의 고속 병렬 처리 방법에 관한 것으로, 특히 휴대 가능한 스마트 기기의 모바일 GPU를 통해 대상체로부터 반사된 초음파 신호를 고속으로 병렬처리함으로써, 시간과 장소에 구애받지 않고, 신속하고, 효율적으로 초음파 신호를 처리하여 영상화할 수 있는 스마트 기기를 이용한 초음파 신호의 고속 병렬 처리 방법에 관한 것이다.

배경기술

- [2] 최근 들어, 의료 분야에 IT 기술을 융합하여 구현되는 진단 의료 영상 분야의 기술이 급격히 발전하고 있는 추세이다. 그 중에서도 주로 사용되는 의료 초음파는 인체의 근육, 힘줄, 내부 장기, 장기의 크기, 구조 및 병리학적 손상 여부를 판단하기 위해 실시간으로 단층 영상을 가시화하는데 사용된다.
- [3] 이처럼 의료 초음파를 이용하여 진단 의료 영상을 구현하기 위해서는 대상물에 프로브를 대고 초음파를 발생시켜 상기 대상물로부터 반사된 초음파를 수신하여 영상을 구성한다. 즉, 초음파를 발생시키면 매우 짧은 시간 안에 음파가 매질 속을 지나가고, 음향 임피던스가 다른 두 매질 사이를 지날 때에는 반사파가 발생한다. 이에 발생된 반사파를 측정해 반사음이 되돌아 올 때까지의 시간을 통해 거리를 역산함으로써 진단 의료 영상을 생성하게 된다.
- [4] 이처럼 진단 의료 영상이 의료 분야에서 널리 사용됨에 따라, 남녀노소를 불문하고 많은 사용자에게 보급된 스마트 기기를 통해 시간과 장소에 상관없이 이동 중에도 초음파 신호를 이용한 진단 의료 영상을 생성하여 확인하고자 하는 요구가 점차 높아지고 있다. 하지만 이러한 요구에도 불구하고, 휴대성을 목적으로 사용되는 스마트 기기에서는 내부 CPU의 프로세싱 파워로 초음파 신호를 처리하는 경우, 실제 진단에 유용한 프레임율의 영상을 제공하기 어려운 문제점이 발생했다. 특히, 스마트 기기 내부에 GPU를 구비하더라도 PC를 기반으로 하는 환경에 비하여, CUDA, OpenCL 등과 같이 그래픽 처리 장치(GPU)에서 수행하는 병렬 처리 알고리즘을 스마트 기기의 환경에서는 PC 기반 환경과 같이 동일하게 처리하기 어려운 문제점이 발생했다.
- [5] <선행기술문헌> KR 10-2012-0059740 (적어도 하나의 GPU를 구비하는 초음파 시스템, 삼성메디슨 주식회사) 2012.06.11.

발명의 상세한 설명

기술적 과제

- [6] 상기와 같은 종래 기술의 문제점을 해결하기 위해, 본 발명은 스마트기기가 모바일 GPU의 그래픽스 파이프라인 구조를 이용하여 초음파 신호를 처리할 때,

상기 그래픽스 파이프라인 구조 내 프래그먼트 셰이더 단계에서 할당된 연산 처리 일부를 할당된 연산량이 상대적으로 적은 버텍스 셰이더 단계에서 수행하도록 함으로써, 그래픽 처리 장치의 연산을 효율적으로 수행할 수 있는 스마트 기기를 이용한 초음파 신호의 고속 병렬 처리 방법을 제공하고자 한다.

과제 해결 수단

- [7] 상기 기술적 과제를 해결하기 위하여, 본 발명의 일 실시예에 따른 모바일 GPU(Graphic Processing Unit)를 구비하는 스마트 기기가 초음파 신호를 입력받아 초음파 영상을 생성하는데 이용되는 초음파 신호의 고속 병렬 처리 방법은, 상기 스마트 기기가 제1 렌더 사이클(Render Cycle)을 통해 빔포밍된 초음파 신호를 입력받아, 상기 초음파 신호로부터 DC 성분을 제거한 후, DC 성분이 제거된 초음파 신호로부터 in-phase 성분과 quadrature 성분을 나누어 각각 출력하는 단계; 상기 스마트 기기가 제2 렌더 사이클을 통해 상기 in-phase 성분과 quadrature 성분을 갖는 초음파 신호에 대하여 quadrature 복조 및 포락선 검파 처리를 수행하는 단계; 및 상기 스마트 기기가 제5 렌더 사이클을 통해 상기 제2 렌더 사이클의 처리 결과로서 획득된 초음파 신호에 대하여 스캔 변환을 수행하는 단계;를 포함하되, 상기 렌더 사이클들은, 버텍스 셰이더(vertex shader) 단계, 래스터라이저(Rasterizer) 단계 및 프래그먼트 셰이더(fragment shader) 단계를 포함하는 그래픽스 파이프라인(Graphics pipeline) 구조로 이루어진다.
- [8] 일 실시예에 따른 초음파 신호의 고속 병렬 처리 방법에서, 상기 버텍스 셰이더 단계는, 상기 모바일 GPU가 복수 개의 정점(vertex)을 입력받고, 입력받은 정점을 이용하여 공간영역을 할당한 후, 할당한 공간영역에 대한 공간좌표를 연산하여 연산결과를 varying 변수 형태로 생성할 수 있다.
- [9] 일 실시예에 따른 초음파 신호의 고속 병렬 처리 방법에서, 상기 래스터라이저 단계는, 상기 모바일 GPU가 상기 버텍스 셰이더 단계에서 출력된 varying 변수와 대응하는 스크린 내 좌표값을 검색하고, 검색한 좌표값을 varying 변수 형태로 생성할 수 있다.
- [10] 일 실시예에 따른 초음파 신호의 고속 병렬 처리 방법에서, 상기 버텍스 셰이더 단계는, 상기 모바일 GPU가 상기 래스터라이저 단계에서 생성된 스크린 내 좌표값의 주변에 위치하는 좌표값을 연산하고, 연산결과를 varying 변수 형태로 생성하는 과정을 더 포함할 수 있다.
- [11] 일 실시예에 따른 초음파 신호의 고속 병렬 처리 방법에서, 상기 프래그먼트 셰이더 단계는, 상기 모바일 GPU가 상기 래스터라이저 단계에서 생성된 스크린 내 좌표값에 대한 컬러를 연산하여 연산결과를 생성할 수 있다.
- [12] 일 실시예에 따른 초음파 신호의 고속 병렬 처리 방법은, 상기 모바일 GPU가 상기 프래그먼트 셰이더 단계에서 생성된 스크린 내 좌표값에 대한 컬러 연산결과를 프레임버퍼에 저장하는 단계를 더 포함할 수 있다.
- [13] 일 실시예에 따른 초음파 신호의 고속 병렬 처리 방법에서, 상기 varying 변수는

상기 모바일 GPU의 사양에 따라 8개 내지 16개로 출력될 수 있다.

- [14] 일 실시예에 따른 초음파 신호의 고속 병렬 처리 방법은, 상기 모바일 GPU가 이전 렌더 사이클에서 생성된 이미지를 프레임버퍼에 저장하면, 저장된 이미지를 RTT(Render To Texture)기술을 이용하여 상기 모바일 GPU의 메모리에 해당하는 텍스처(Texture)로 전달하고, 전달한 이미지를 다음 렌더 사이클의 그래픽스 파이프라인 중 프래그먼트 셰이더 단계로 전달할 수 있다.
- [15] 일 실시예에 따른 초음파 신호의 고속 병렬 처리 방법은, 상기 모바일 GPU가 고정 함수 파이프라인(Fixed Function Pipeline) 구조를 이용하여 초음파 신호를 병렬 처리할 수 있다.
- [16] 일 실시예에 따른 초음파 신호의 고속 병렬 처리 방법은, OpenGL ES 3.0 환경에서 구현될 수 있다.
- [17] 일 실시예에 따른 초음파 신호의 고속 병렬 처리 방법은, 상기 모바일 GPU가 상기 프래그먼트 셰이더 단계에서 수행되도록 할당된 연산과정의 일부를 상기 버텍스 셰이더 단계에서 미리 수행하도록 제어할 수 있다.
- [18] 일 실시예에 따른 초음파 신호의 고속 병렬 처리 방법은, 상기 제2 렌더 사이클에서, 상기 스마트 기기가 상기 quadrature 복조 처리된 초음파 신호에 대해 데시메이션(decimation) 처리를 더 수행할 수 있다.
- [19] 일 실시예에 따른 초음파 신호의 고속 병렬 처리 방법은, 상기 제2 렌더 사이클에서, 상기 스마트 기기가 상기 포락선 검파 처리된 초음파 신호에 대해 로그 압축 처리를 더 수행할 수 있다.
- [20] 일 실시예에 따른 초음파 신호의 고속 병렬 처리 방법은, 상기 제2 렌더 사이클에서, 상기 스마트 기기가 상기 로그 압축 처리된 초음파 신호에 대해 영상의 전체 이득(gain)을 조절하는 이득 제어 처리를 더 수행할 수 있다.
- [21] 일 실시예에 따른 초음파 신호의 고속 병렬 처리 방법은, 상기 제2 렌더 사이클의 수행 이후, 상기 스마트 기기가 제3 렌더 사이클을 통해 블랙홀로 간주하는 문턱치값을 입력받아, 상기 문턱치값과 상기 제2 렌더 사이클로부터 입력받은 초음파 신호의 크기를 상호 비교하여 상기 초음파 신호에 대한 블랙홀을 제거하는 단계;를 더 포함할 수 있다.
- [22] 일 실시예에 따른 초음파 신호의 고속 병렬 처리 방법은, 상기 제2 렌더 사이클의 수행 이후, 상기 스마트 기기가 제4 렌더 사이클을 통해 상기 제2 렌더 사이클로부터 입력받은 초음파 신호에 대한 에지 강화를 수행하는 단계;를 더 포함할 수 있다.
- [23] 한편, 이하에서는 상기 기재된 초음파 신호의 고속 병렬 처리 방법을 컴퓨터에서 실행시키기 위한 프로그램을 기록한 컴퓨터로 읽을 수 있는 기록매체를 제공한다.

발명의 효과

- [24] 본 발명의 스마트 기기를 이용한 초음파 신호의 고속 병렬 처리 방법은 PC

기반의 환경이 아닌 모바일 기반의 환경에서도 스마트 기기 내 모바일 GPU를 통해 초음파 신호에 대한 고속의 병렬처리를 수행함으로써, 의료 진단에 유용한 프레임율을 갖는 영상을 제공할 수 있는 효과가 있다.

- [25] 또한 본 발명의 스마트 기기를 이용한 초음파 신호의 고속 병렬 처리 방법은 그래픽스 파이프라인 구조를 이용하여 초음파 신호를 병렬 처리 시, 상기 그래픽스 파이프라인의 구조 중 프래그먼트 셰이더 단계에서 할당된 연산량 일부를 버텍스 셰이더 단계가 수행하도록 함으로써, 초음파 신호의 병렬 처리를 위해 수행되어야 하는 연산 처리를 나누어 처리하도록 하여 보다 신속하게 초음파 처리를 수행할 수 있는 효과가 있다.

도면의 간단한 설명

- [26] 도 1은 하나의 렌더 사이클 처리 과정을 나타낸 도면이다.
- [27] 도 2는 렌더 사이클의 그래픽스 파이프라인 구조 내 연산처리 과정을 나타낸 도면이다.
- [28] 도 3은 두 개의 렌더 사이클 처리 과정을 나타낸 도면이다.
- [29] 도 4는 버텍스 셰이더 단계와 프래그먼트 셰이더 단계의 블록도이다.
- [30] 도 5는 축 방향으로 4096 샘플이 넘는 크기의 데이터를 텍스처에 업로드하는 과정을 나타낸 도면이다.
- [31] 도 6은 B-모드 이미징의 신호 경로를 나타낸 도면이다.
- [32] 도 7a 및 도 7b는 B-모드 이미징 처리를 위해 본 발명을 적용한 렌더 사이클 처리과정을 나타낸 도면이다.
- [33] 도 8은 그래픽스 파이프라인 내 9-탭 필터링 과정을 나타낸 도면이다.
- [34] 도 9는 첫 번째 렌더 사이클과 두 번째 렌더 사이클 간의 초음파 신호처리과정의 일 예를 나타낸 도면이다.
- [35] 도 10은 본 발명의 적용하여 획득한 B-모드 이미징의 동작 화면이다.

발명의 실시를 위한 최선의 형태

- [36] 본 발명의 실시를 위한 최선의 형태에 따라, 모바일 GPU(Graphic Processing Unit)를 구비하는 스마트 기기가 초음파 신호를 입력 받아 초음파 영상을 생성하는데 이용되는 초음파 신호의 고속 병렬 처리 방법은, 상기 스마트 기기가 제1 렌더 사이클(Render Cycle)을 통해 빔포밍된 초음파 신호를 입력받아, 상기 초음파 신호로부터 DC 성분을 제거한 후, DC 성분이 제거된 초음파 신호로부터 in-phase 성분과 quadrature 성분을 나누어 각각 출력하고, 상기 스마트 기기가 제2 렌더 사이클을 통해 상기 in-phase 성분과 quadrature 성분을 갖는 초음파 신호에 대하여 quadrature 복조 및 포락선 검파 처리를 수행하고, 상기 스마트 기기가 제5 렌더 사이클을 통해 상기 제2 렌더 사이클의 처리 결과로서 획득된 초음파 신호에 대하여 스캔 변환을 수행하며, 상기 렌더 사이클들은 버텍스 셰이더(vertex shader) 단계, 래스터라이저(Rasterizer) 단계 및 프래그먼트 셰이더(fragment shader) 단계를 포함하는 그래픽스 파이프라인(Graphics pipeline)

구조로 이루어진다.

발명의 실시를 위한 형태

- [37] 이하, 본 발명을 바람직한 실시 예와 첨부한 도면을 참고로 하여 본 발명이 속하는 기술 분야에서 통상의 지식을 가진 자가 용이하게 실시할 수 있도록 상세히 설명한다. 그러나 본 발명은 여러 가지 상이한 형태로 구현될 수 있으며, 여기에서 설명하는 실시 예에 한정되는 것은 아니다.
- [38] 우선, 본 발명이 적용되는 렌더 사이클 구조에 대하여 도 1을 통해 간략히 살펴보도록 한다.
- [39] 본 발명의 구현되는 OpenGL ES 환경에서 하나의 이미지를 생성해 내는 단위를 렌더 사이클(render cycle)이라 한다.
- [40] 도 1은 하나의 렌더 사이클 처리 과정을 나타낸 도면이다.
- [41] 도 1에 도시된 바와 같이, 이러한 렌더 사이클은 빔포밍된 데이터에 대하여 초기화(initialization)과정, 렌더 씬(render scene)과정, 그래픽스 파이프라인(graphics pipeline)과정을 수행하여 초음파 이미지를 생성한다.
- [42] 먼저, 초기화 과정에서는 그래픽스 파이프라인 구조 내 사용자가 프로그래밍할 수 있는 버텍스 셰이더(vertex shader) 단계와 프래그먼트 셰이더(fragment shader) 단계에 대하여 컴파일을 수행하고, 입력될 데이터를 저장할 그래픽 처리 장치(Graphic Processing Unit) 상의 메모리인 텍스처(texture)의 속성과 크기를 설정하며, 렌더 사이클의 출력 결과 이미지를 저장하는 프레임버퍼(framebuffer)의 속성과 크기를 설정한다.
- [43] 이어지는 렌더 씬 과정에서는 이미지가 존재하는 공간 영역을 할당하는데 사용될 정점들(vertices)을 이어지는 그래픽스 파이프라인 구조에 입력하면서 glDrawElements와 같은 명령어를 통해 그래픽스 파이프라인의 처리과정이 시작되도록 명령한다.
- [44] 이러한 그래픽스 파이프라인 처리과정은 실질적으로 그래픽 처리 장치(GPU)를 이용하여 연산의 대부분을 수행한다.
- [45] 이하, 도 2를 참조하여 실질적인 연산과정이 이루어지는 그래픽스 파이프라인 구조에 대하여 자세히 살펴보도록 한다.
- [46] 도 2는 렌더 사이클의 그래픽스 파이프라인 구조 내 연산처리 과정을 나타낸 도면으로서, 모바일 GPU(Graphic Processing Unit)를 구비하는 스마트 기기 상에서 구현될 수 있다.
- [47] 도 2에 도시된 바와 같이, OpenGL ES 3.0 환경의 그래픽스 파이프라인은 버텍스 셰이더(vertex shader)(210) 단계, 래스터라이저(rasterizer)(220) 단계, 프래그먼트 셰이더(fragment shader)(230) 단계와 텍스처(texture)(240) 단계로 이루어진다.
- [48] 버텍스 셰이더(210) 단계에서는 렌더 씬 과정을 통해 입력 받은 정점들(vertices)의 공간좌표를 계산한다. 이어서, 래스터라이저(220) 단계에서는

스크린 상에 존재하는 픽셀 중 상기 버텍스 셰이더(210) 단계에서 할당된 공간 안에 존재하는 좌표를 계산해 출력한다. 이후, 프래그먼트 셰이더(230) 단계에서는 상기 래스터라이저(220) 단계에서 출력된 픽셀들의 좌표를 받아 해당 픽셀들의 컬러를 계산한다. 이때, 상기 프래그먼트 셰이더(230) 단계에서 픽셀들의 컬러를 계산하는데 있어, GPU 메모리인 텍스처에 이전에 올려 두었던 이미지를 불러와 사용할 수 있다.

- [49] 이처럼 그래픽스 파이프라인 구조를 구성하는 단계 중 사용자가 프로그래밍이 가능한 부분은 버텍스 셰이더(210) 단계와 프래그먼트 셰이더(230) 단계이다. 이러한 상기 버텍스 셰이더 단계와 프래그먼트 셰이더 단계의 동작을 결정하는 셰이더의 프로그래밍을 통해 프로그래머가 의도하는 그래픽을 GPU에서 생성해 내도록 명령할 수 있다. 상기 그래픽스 파이프라인에서 생성한 결과는 프레임버퍼(Framebuffer)에 저장된다.
- [50] 이러한 일련의 과정이 하나의 이미지를 생성하기 위한 하나의 렌더 사이클의 과정이다. 또한, 하나의 렌더 사이클 과정을 통해 생성된 이미지를 입력으로 받아 다른 연산을 수행하도록 하는 방법을 RTT(Render to Texture)기술이라 한다.
- [51] 이하에서는, RTT 기술을 적용한 두 개의 렌더 사이클 처리과정에 대하여 살펴보도록 한다.
- [52] 도 3은 두 개의 렌더 사이클 처리 과정을 나타낸 도면이다.
- [53] 앞서 도 2를 통해 설명한 첫 번째 렌더 사이클의 결과물이 프레임버퍼에 저장되는데 이를 다시 텍스처로 렌더링(rendering)해서 이어지는 다음 렌더 사이클에서의 입력으로 사용한다.
- [54] 이하에서는 본 발명을 이용한 초음파 신호 처리 과정의 일 예를 자세히 살펴보도록 한다.
- [55] 먼저, 그래픽스 파이프라인에서 사용할 데이터가 스마트 기기의 메인 메모리에 프레임 단위의 데이터를 그래픽 처리 장치(GPU)에서 사용되기 위해, 스마트기기의 모바일 CPU가 상기 데이터를 GPU의 메모리인 텍스처로 업로드 한다. 렌더 사이클의 첫 번째 과정인 초기화 과정에서 그래픽 처리 장치(GPU)가 프레임버퍼와 텍스처 메모리를 할당하는데, 이때, 입력되는 데이터의 종류에 따라 텍스처 메모리의 속성을 다르게 정의할 수 있다. 예를 들어, 16비트 정수(integer) 형태를 갖는 빔포밍된 RF 데이터를 입력 받아 처리하는 경우에는 `glTexImage2D(GL_TEXTURE_2D, 0, GL_RED, sample_no, scanline_no, 0, GL_RED_INTEGER, GL_SHORT, inputData)`를 이용하여 함수로 입력될 데이터가 저장될 텍스처를 정의한다. 이때, 입력받은 데이터는 프레임 단위의 데이터이므로 2D 텍스처로 정의하기 위해 `glTexImage2D` 함수를 사용하고, 16 비트 정수 데이터이므로, 빨간 색에 해당되는 부분에 16비트 정수 데이터를 저장하겠다는 의미의 `GL_R16I`로 한다. 또한 `sample_no`에 프레임 데이터의 축 방향으로의 샘플 개수를 입력하고, `scanline_no`에 프레임 데이터의 주사선(scanline) 수를 입력하여 프레임 데이터의 크기 정보를 입력한다. 또한,

GL_RED는 앞서 GL_R16I에서 빨간색만을 사용한다는 의미이고, 뒤이어 오는 GL_RED_INTEGER는 데이터 종류가 정수인 것을 나타낸다. 마지막의 inputData는 입력된 데이터를 가리키는 포인터(pointer)이다.

- [56] 이어서, 데이터의 빔포밍 수행 시, 미세 딜레이(fine delay)를 적용한 위치의 데이터를 로드하거나, 스캔 변환을 수행할 때, 2중 선형 보간법(bilinear interpolation)의 수행이 필요하다. 본 발명이 적용되는 OpenGL ES 3.0 환경에서는 텍스처의 특징을 이용해 셰이더 프로그래밍에 2중 선형 연산을 수행하지 않고 텍스처의 정수 좌표가 아닌 곳의 데이터를 로드할 때 2중 선형 보간법을 수행한 결과를 로드할 수 있다. 렌더 사이클의 첫 번째 과정인 초기화 단계에서 프레임버퍼와 텍스처 메모리에 대한 크기 할당을 수행하는데, 여기서 텍스처의 속성을 지정해준다.
- [57] 특히, 하기와 같이 표현되는 glTexParameterf 함수를 이용하여 텍스처를 셰이더에서 좌표 액세스를 통해 로드할 때, 2중 선형 보간법을 수행해 로드할 것인지 또는 가장 가까운 값을 로드할 것인지를 여부를 결정할 수 있다.
- [58] glTexParameterf(GL_TEXTURE_2D, GL_TEXTURE_MAG_FILTER, GL_NEAREST);
- [59] glTexParameterf(GL_TEXTURE_2D, GL_TEXTURE_MIN_FILTER, GL_LINEAR);
- [60] 이때, 상기 GL_TEXTURE_MAG_FILTER는 텍스처의 본래 사이즈보다 확대를 해서 출력하는 경우를 나타내고, GL_NEAREST는 텍스처가 본래 사이즈보다 확대된 경우에는 본래 텍스처의 크기로 채워지지 않는 부분을 가장 가까운 값으로 채우겠다는 것을 나타낸다. 이 부분을 GL_LINEAR로 표현하면 2중 선형 보간을 수행한 결과로 화면을 채우게 된다. 또한 상기 GL_TEXTURE_MIN_FILTER는 텍스처의 본래 사이즈보다 축소를 해서 화면에 출력하는 경우를 나타내고, GL_NEAREST는 이때 본래 텍스처의 값중에 화면에 값을 채울 때 접근하는 좌표에 가장 가까운 텍스처의 값으로 화면을 채우는 것을 나타내고, GL_LINEAR는 2중 선형 보간을 통해 화면을 채우는 것을 나타낸다.
- [61] 상술한 과정을 통해 하나의 렌더 사이클 결과물은 프레임버퍼에 저장된다. 그리고 필요에 따라 프레임버퍼에 저장된 결과는 RTT(Render to Texture)기술을 이용해 이어지는 다음 렌더 사이클의 입력으로 사용될 수 있다. 렌더 사이클의 첫 번째 처리 과정인 초기화 단계에서 프레임버퍼와 텍스처 메모리 할당을 수행하는데, 그래픽 파이프라인의 출력이 저장될 프레임버퍼의 속성을 출력의 종류에 따라 속성을 다르게 정의한다. 예를 들어, Quadrature 복조를 수행해 In-phase 성분과 Quadrature 성분이 출력되는 경우에, 출력되는 In-phase 성분과 Quadrature 성분을 저장하기 위해 하기와 같이 프레임버퍼의 속성을 정의할 수 있다.
- [62] glTexImage2D(GL_TEXTURE_2D, 0, GL_RG16F, sample_no, scanline_no, 0, GL_RG, GL_FLOAT, 0);

- [63] 이때, 입력되는 데이터는 프레임 단위의 데이터이므로 2D 텍스처로 정의하기 위해, 모바일 GPU가 `glTexImage2D` 함수를 사용하고, 출력되는 In-phase 성분과 Quadrature 성분을 저장하는 과정에서 연산의 정확도를 위해 16bit float로 저장하며, 색상 중 빨간색과 녹색의 위치에 각각 In-phase 성분과 Quadrature 성분을 저장하기 위해 `GL_RG16F`로 선언한다. 또한, 모바일 GPU가 `sample_no`에 프레임 데이터의 축 방향으로의 샘플 개수를 입력하고, `scanline_no`에 프레임 데이터의 주사선 수를 입력해 프레임 데이터의 크기 정보를 입력한다. `GL_RG`는 앞서 `GL_RG16F`에서 In-phase 성분과 Quadrature 성분을 각각 픽셀의 빨간색과 녹색의 위치에 저장하기 위해 사용한다는 의미이고, 뒤이어 오는 `GL_FLOAT`는 데이터의 크기가 float 임을 나타낸다. `glTexImage2D` 함수의 마지막 위치에는 처음 입력되는 텍스처를 정의할 때는 입력되는 데이터의 포인터를 기재하나, RTT 기술을 이용하기 위한 프레임버퍼를 정의할 때는 해당 프레임버퍼를 비어둔다는 의미에서 0이 입력되며, 이때 입력된 데이터를 가리키는 포인터를 기재한다.
- [64] 렌더 사이클의 두 번째 처리 과정인 렌더 썬 단계에서는 모바일 GPU가 그래픽스 파이프라인에 진입하게 하는 명령어와 함께 그래픽스 파이프라인이 저장될 프레임버퍼를 지정하는 명령을 수행한다.
- [65] OpenGL ES 3.0의 그래픽스 파이프라인 중에서 버텍스 셰이더 단계에서 결정한 공간영역 내부에 존재하는 픽셀들의 좌표는 그래픽스 파이프라인 구조의 출력이 저장되는 프레임버퍼의 크기를 가지는 것으로 여겨 래스터라이저 단계에서 검색한다. 따라서, RTT 기술에서 렌더 타겟이 되는 텍스처인 프레임버퍼의 크기를 그래픽스 파이프라인에 입력되는 텍스처의 크기보다 작게 설정함으로써, 데시메이션 신호처리를 수행할 수 있다. 예를 들어, 축 방향으로 4096개의 샘플을 가지고 주사선 개수가 128개인 빔포밍된 데이터를 입력받아, 축 방향으로 ratio 2만큼 데시메이션을 수행하고자 하는 경우, 입력되는 빔포밍된 데이터가 저장될 텍스처를 폭이 4096, 높이가 128인 텍스처에 업로드하여 프래그먼트 셰이더에서 로드할 수 있도록 하고, 그래픽스 파이프라인의 출력이 저장되는 프레임버퍼의 크기는 폭이 2048, 높이가 128로 속성을 설정한다. 이에 따라, 그러면 래스터라이저 단계에서는 입력된 텍스처에서 축 방향으로 한 픽셀씩 건너뛴 화면 좌표를 검색하여 프래그먼트 셰이더 단계에 제공한다. 이러한 프래그먼트 셰이더 단계에서는 상기 래스터라이저 단계에서 검색된 좌표를 입력 받아 입력된 텍스처에서 데이터를 축 방향으로 한 픽셀씩 건너뛴 좌표의 값을 로드한다.
- [66] 다른 예를 살펴보면, 축 방향으로 2854개의 샘플을 가지고 주사선 개수가 192개인 빔포밍된 데이터를 입력받아, 축 방향으로 ratio 2만큼 데시메이션을 수행하고자 하는 경우, 입력되는 빔포밍된 데이터가 저장될 텍스처를 폭이 2,854, 높이가 192인 텍스처에 업로드하여 프래그먼트 셰이더 단계에서 로드할 수 있도록 하고, 그래픽스 파이프라인의 출력이 저장되는 프레임버퍼의 크기는

폭이 1,427, 높이가 192로 속성을 설정한다.

- [67] 특히, 모바일 GPU에는 두 개의 파이프라인 구조가 있는데, 첫 번째 파이프라인 구조는 고정 함수 파이프라인(fixed function pipeline)으로서, 버텍스 셰이더 단계와 프래그먼트 셰이더 단계에서 연산에 사용되는 물리적 프로세싱 유닛이 따로 할당되어 있다. 두 번째 파이프라인 구조는 통합 셰이더 모델(unified shader model)로 버텍스 셰이더 단계와 프래그먼트 셰이더 단계에서 연산에 사용되는 프로세싱 유닛이 하나로 통합되어 있어, 사용자가 셰이더 프로그래밍을 통해 작성한 동작에 따라 버텍스 셰이더 단계와 프래그먼트 셰이더 단계의 연산에 필요한 정도에 따라 프로세싱 유닛이 각각의 셰이더 단계의 연산에 배정된다. 고정 함수 파이프라인 구조를 갖는 모바일 GPU의 종류에는 ARM사의 Mali-400 series와 Qualcomm사의 Adreno 130 등이 있다. 또한, 통합 셰이더 모델 구조를 갖는 모바일 GPU의 종류에는 ARM사의 Mali-T600 series와 Qualcomm사의 Adreno 200 series, Adreno 300 series, Adreno 400 series 등이 있다. 이러한 모바일 GPU의 구조는 고정 함수 파이프라인 구조에서 통합 셰이더 모델 구조로 바뀌어가는 추세이다.
- [68] 본 발명에 따라 버텍스 셰이더 단계와 프래그먼트 셰이더 단계간 로드를 분담하기 위한 최적화 방법은 사용하는 스마트 기기의 모바일 GPU가 고정 함수 파이프라인 구조인 경우에 적용하여 효과를 볼 수 있다.
- [69] OpenGL ES 3.0 환경에서의 그래픽스 파이프라인을 이용한 초음파 신호처리는 사용자가 프로그래밍 가능한 버텍스 셰이더 단계와 프래그먼트 셰이더 단계간의 동작을 결정하는 셰이더 프로그래밍을 통해 구현된다. 특히, 입력되는 데이터와 출력되는 데이터의 형태 모두 직사각형 형태이므로, 버텍스 셰이더 단계에서는 정점(vertex) 4개만을 이용해 직사각형 영역을 할당하는 것 외에는 다른 연산을 하지 않으며, 대부분의 연산은 프래그먼트 셰이더 단계에서 수행하게 된다. 따라서, 통합 셰이더 모델처럼 셰이더의 프로그래밍에 따라 버텍스 셰이더 단계와 프래그먼트 셰이더 단계의 연산에 할당되는 프로세싱 유닛의 수가 유동적이지 않는 고정 함수 파이프라인 구조를 가지는 모바일 GPU를 사용하는 경우에는 셰이더 프로그래밍 상에서 프래그먼트 셰이더 단계에 할당된 연산 중 일부를 버텍스 셰이더 단계에서 수행하도록 함으로써, 사용되지 못하고 있는 버텍스 셰이더 단계의 연산에 할당되어 있는 프로세싱 유닛을 사용함으로써 초음파 신호의 연산 처리 속도를 높일 수 있다.
- [70] 이하, 도 4를 참조하여, 버텍스 셰이더 단계와 프래그먼트 셰이더의 연산과저을 자세히 살펴보도록 한다.
- [71] 도 4는 버텍스 셰이더 단계와 프래그먼트 셰이더 단계의 블록도이다.
- [72] 도 4에 도시된 바와 같이, 먼저 버텍스 셰이더 단계에서 수행한 연산 결과를 varying 변수로 출력하게 되고 출력된 값은 래스터라이저 단계를 통해 스크린 상에 존재하는 값들이 검색되어 프래그먼트 셰이더 단계로 varying 변수의 형태로 입력된다. 예를 들면, 버텍스 셰이더 단계에서 연산한 결과는 보통 공간

영역을 나타내는 좌표가 되는데, 이때 좌표값이 *varying* 변수로 출력되면, 해당 렌더 사이클의 스크린 즉, 렌더 타겟인 프레임버퍼의 크기에 맞는 화면상에서 버텍스 셰이더 단계에서 정의한 공간 영역 안에 들어가는 픽셀들의 좌표가 프래그먼트 셰이더 단계의 *varying* 변수로 입력된다.

- [73] 이어서, 프래그먼트 셰이더 단계에서는 하나의 *varying* 변수로 받은 좌표들로 텍스처에 업로드한 입력 데이터를 액세스해 연산에 사용한다.
- [74] 만약, 렌더 사이클의 처리 과정 내 필터링 연산이 포함되어, 프래그먼트 셰이더 단계에서 한 픽셀의 값을 계산하기 위해서 필터의 탭 수만큼의 텍스처 데이터를 액세스할 필요가 있다면, 프래그먼트 셰이더 단계에서는 *varying* 변수 형태로 입력받은 하나의 좌표에 덧셈연산하는 과정을 통해 텍스처의 이웃 좌표의 값을 계산하여 텍스처에서 이웃 좌표의 값을 로드해야 한다. 하지만 도 4에 도시된 셰이더의 구조를 살펴보면 *varying* 변수는 GPU의 사양에 따라 최소 8개에서 최대 16개까지 사용될 수 있다. 즉, 모바일 GPU가 버텍스 셰이더 단계에서 이웃 좌표에 해당될 값을 미리 연산을 하고, 그 연산 결과를 최대 16개의 *varying* 변수로 프래그먼트 셰이더 단계로 전달하면 프래그먼트 셰이더 단계에서는 이웃 좌표를 연산하기 위한 과정을 생략할 수 있다. 다만, 필터의 탭 수가 16개를 넘는 경우, 프래그먼트 셰이더 단계에서의 이웃 좌표의 연산은 여전히 필요할 수 있다. 하지만 그렇더라도 이전보다 적은 수의 좌표연산만이 프래그먼트 셰이더 단계에서 수행하게 된다.
- [75] 또한, 앞서 설명한 OpenGL ES 3.0 Texture의 특징을 이용한 2중 선형 보간법에서도 초음파 신호처리 과정에서 2중 선형 보간이 필요한 경우에 OpenGL ES 3.0 texture의 특징을 이용해 별다른 2중 선형 보간을 수행하기 위한 프로그래밍 없이 바로 2중 선형 보간이 가능한 것을 알 수 있다. 하지만 텍스처의 속성을 GL_LINEAR로 설정하여 2중 선형 보간을 수행하도록 한 경우와, GL_NEAREST로 설정하여 가까운 값을 가져오도록 한 경우에 연산속도가 차이가 난다. 따라서, 2중 선형 보간이 필요 없이 텍스처의 정수 좌표에서만 데이터를 로드하는 경우에는 텍스처의 속성을 GL_NEAREST로 설정하여 연산 속도를 높일 수 있다.
- [76] 이하에서는 축 방향으로 4096 sample이 넘는 크기를 갖는 데이터를 텍스처에 업로드하는 과정에 대하여 살펴보도록 한다.
- [77] 현재 상용화된 모바일 GPU는 텍스처의 크기를 폭, 높이를 각각 4096 픽셀로 제한하고 있다. 따라서 축 방향으로 4096 샘플이 넘는 크기를 갖는 초음파 신호 데이터는 지금까지 상술한 방법으로는 텍스처에 업로드하여 셰이더 단계에서 로드해 사용할 수 없다.
- [78] 이처럼, 높이의 크기가 4096로 제한되는 것은 픽셀 기준이기 때문에, 초기화 과정에서 텍스처를 GL_RGBA로 정의하면 한 픽셀은 빨간색, 녹색, 파란색, 알파의 자리에 각각 초음파 데이터를 하나씩 넣을 수 있어, 최대 4개의 정보 저장이 가능하다. 즉, 최대 축 방향으로 16,384(4096×4) 샘플의 데이터가 입력될

수 있다.

[79] 도 5는 축 방향으로 4096 샘플이 넘는 크기의 데이터를 텍스처에 업로드하는 과정을 나타낸 도면이다.

[80] 도 5에 도시된 바와 같이, 축 방향으로 7808 샘플, 측면 방향으로 128 샘플을 갖는 초음파 데이터를 텍스처로 업로드한다.

[81] 예를 들어, 주사선 0(sc0)의 축 방향으로의 데이터를 상단부터 순서대로 s0, s1, ... 라 하면, 텍스처에 저장되는 형태는 텍스처의 속성을 GL_RG16I로 지정해 빨간색의 자리에 16bit integer인 s0, s2, s4, ... s7806, s7808을 저장하고, 녹색의 자리에 s1, s3, s5, ... s7807이 저장되도록 한다. 나머지 주사선에 대해서도 동일하게 초음파 신호 데이터가 저장되도록 한다. 프래그먼트 셰이더 단계에서는 이와 같은 형태로 저장된 초음파 신호 데이터를 사용하기 위해서 래스터라이저 단계에서 전달되는 좌표의 픽셀 데이터를 로드한 후 빨간색, 녹색 자리 중 필요한 위치의 데이터를 골라 사용하게 된다.

[82] 이하에서는 본 발명에 따른 스마트 기기를 이용한 초음파 신호의 고속 병렬 처리 방법을 이용하여 B-모드 이미징을 처리하는 과정에 대하여 자세히 살펴보도록 한다.

[83] 먼저, 본 실시 예를 구현하는데 사용되는 대상 기기는 Google new Nexus7 (2013)이고, OS는 Android 4.4.2 KitKat이며, Qualcomm사의 Snapdragon S4 pro AP를 탑재하고 있으며, 모바일 GPU는 Adreno 320이 탑재되어 있다. 또한 본 실시 예의 개발환경은 Eclipse 4.2.1을 사용하며 Android NDK r9c 버전을 이용해 Android 어플리케이션 개발에 JAVA와 native language(C++)의 연동이 가능하도록 한다.

[84] 도 6은 B-모드 이미징의 신호 경로를 나타낸 도면이다.

[85] 도 6에서 사용되는 입력 데이터는 빔포밍된 데이터로 축 방향으로 2854 샘플, 주사선은 192개이며, 중심 주파수는 3.3 MHz, 샘플링율은 20 MHz로 얻은 데이터이다. 신호 경로는 일반적인 B-모드 이미징의 처리 과정을 나타낸다. 상기와 같은 특징을 갖는 입력 데이터로부터 DC 성분을 제거한 후, 디지털 시간 이득 보상(TGC: Time Gain Compensator)을 수행하고, 이어서 Quadrature 복조와 ratio 2로 데시메이션을 수행하며, 포락선 검파와 로그 압축을 수행한다. 이후, 영상의 전체적 이득(Gain)을 조절할 수 있는 이득 제어, 블랙홀 필터링, 에지 강화 및 디지털 스캔 변환(DSC: Digital Scan Conversion)을 연속하여 수행한다.

[86] 이러한 B-모드 이미징의 신호 경로를 OpenGL ES 3.0으로 구현하기 위한 렌더 사이클은 도 7a 및 도 7b와 같이 나타나 있다.

[87] 도 7a은 B-모드 이미징 처리를 위해 본 발명을 적용한 렌더 사이클 처리과정을 나타낸 도면이다.

[88] 먼저, 신호 경로는 하나의 신호 블록의 수행 결과가 다음 신호 블록에서 처리되기 위해서 이미 완료되어 있어야 하는 경우에 다른 렌더 사이클로 나누어진다. 예를 들어, 앞선 신호 블록을 수행 후 다음 신호 블록에서 여러 탭의

필터링을 수행하는 경우에는 GPU를 통해 구현되는 그래픽스 파이프라인에서 현재 연산하고자 하는 픽셀의 값을 구하기 위해서는 현재 픽셀의 좌표에 해당하는 텍스처값 이외에 주변의 값들이 함께 필요하게 된다. 따라서, 주변의 값도 함께 액세스하기 위해서는 이미 주변의 값들이 계산된 결과가 존재해야 하므로, 렌더 사이클을 분리하여 앞선 신호 블록의 수행 결과를 프레임 단위로 프레임버퍼에 저장한 후, RTT기술로 텍스처화 하여 다음 렌더 사이클에서 입력으로 사용한다.

- [89] 또한 하나의 렌더 사이클 내부의 연산과정은 동시에 수행되므로, 렌더 사이클의 개수를 최소화로 설정하는 것이 성능 향상에 유리하다. 이때, 렌더 사이클에서 수행되는 연산은 출력 결과물이 저장될 버퍼 상의 픽셀 수만큼 수행될 수 있다.
- [90] 상술한 기준을 통해 나누어지는 각각의 렌더 사이클 중에서, 제1 렌더 사이클을 자세히 살펴보도록 한다.
- [91] 제1 렌더 사이클에서는 빔포밍된 데이터로부터 DC 성분을 제거하고, Quadrature 복조과정의 일부를 수행하여 데이터를 출력(x_i, x_q)한다. 이때, 상기 DC 성분을 제거하는 DC 성분 제거 신호 블록(710)은 32-탭 힐버트 필터(Hilbert Filter)를 포함하고, 이러한 필터를 통해 고역 통과 필터링을 수행하여 빔포밍된 데이터로부터 DC 성분을 제거할 수 있다.
- [92] 도 8은 그래픽스 파이프라인 내 9-탭 필터링 과정을 나타낸 도면이다.
- [93] 도 8에 도시된 바와 같이, 프래그먼트 셰이더 단계에서는 프레임버퍼에 존재하는 픽셀의 수만큼 프래그먼트 셰이더 단계의 `main()` 함수를 반복적으로 수행한다. 상기 `main()` 함수가 한 번 수행할 때마다 프레임버퍼의 해당 픽셀 좌표의 컬러값이 결정된다. 따라서 필터링을 수행하는 경우에는 하나의 `main()` 안에서 현재 연산중인 프레임버퍼의 픽셀 좌표에 해당하는 텍스처(입력 이미지) 상의 이웃 좌표의 데이터가 필요하게 된다. 따라서, 도 8을 자세히 살펴보면 현재 연산중인 픽셀의 좌표가 (x, y) 라면 $p0-p8$ 의 값을 y 축의 값의 계산을 통해 텍스처 상의 이웃 데이터를 `texture()` 함수로 로드한다. 이후, 필터 계수와 로드한 $p0-p8$ 의 컨벌루션(convolution) 연산이 수행되고, 그 결과가 Pout으로 출력되어 프레임버퍼상에서 해당 좌표에 저장된다. 또한 다음 픽셀의 값을 계산하기 위해서는 프레임버퍼의 다음 좌표를 기준으로 `main()`을 반복하여 수행한다. 이와 같은 동일한 방법으로 `main()`을 프레임버퍼에 존재하는 픽셀 수만큼 반복적으로 수행해 필터링을 수행한 결과를 프레임 단위로 프레임버퍼에 저장한다. 같은 방법으로 B-모드 이미징의 예시에서는 16-탭 고역 통과 필터링을 수행한다.
- [94] 이후, Quadrature 복조 신호 블록(720)에서 `cos`과 `sin`을 곱해 in-phase 성분과 quadrature 성분을 출력하는 과정까지 제1 렌더 사이클에서 신호처리를 수행한다. 상기 제1 렌더 사이클의 출력은 in-phase 성분과 quadrature 성분 두 가지이기 때문에 제1 렌더 사이클의 출력이 저장될 프레임버퍼는 GL_RG16F로

하여 빨간색 자리에 in-phase 성분 저장하고, 녹색 자리에 quadrature 성분을 저장하도록 한다. 프레임버퍼의 크기는 입력과 동일하다. 상기 제1 렌더 사이클의 출력 데이터를 제2 렌더 사이클로 전달할 때는 연산의 정확도를 유지하기 위해 16bit float 형태로 프레임버퍼에 저장한다.

- [95] 즉, 상기 제1 렌더 사이클에서는 2854×192 픽셀 크기를 갖는 빔포밍된 데이터가 입력되어, $2854 \times 192 = 547,968$ 회의 연산을 거쳐 2854×192 픽셀의 출력 데이터 x_i, x_q 를 출력한다.
- [96] 이어서, 제2 렌더 사이클에서는 상기 제1 렌더 사이클에서 프레임버퍼에 저장한 in-phase 성분과 quadrature 성분을 RTT기술을 이용해 입력 텍스처로 받아 데시메이션 신호처리(730)를 수행한다. 데시메이션율이 2 이므로, 제2 렌더 사이클의 결과가 저장될 프레임버퍼의 폭을 2854의 절반인 1427로 설정한다. 또한, 저역 통과 필터링이 Quadrature 복조 과정(720)과 데시메이션 과정(730)에 중복적으로 필요하므로 각 과정에서의 저역 통과 필터링을 한 번에 수행할 있도록 필터를 설계한다. 따라서 입력받은 데이터에 대하여 저역 통과 필터링을 수행하고, 포락선 검파를 square-root를 이용해 수행한 후 로그 압축을 수행하고, 최종적으로 이득을 곱해 전체적인 영상의 이득을 조절할 수 있도록 한다. 상기 제2 렌더 사이클의 결과는 포락선 검파 과정(740)을 거치면서 성분이 하나로 합성되므로, 출력 데이터가 저장될 프레임버퍼는 GL_R16F로 빨간색 자리에 16 bit float로 저장된다.
- [97] 즉, 제2 렌더 사이클에서는 앞서 제1 렌더 사이클에서 출력된 2854×192 픽셀 크기를 갖는 데이터 (x_i, x_q)를 입력받아, 나머지 quadrature 복조 과정(720)을 수행한다. Quadrature 복조과정(720) 뒤와 데시메이션 과정(730) 앞에 들어가는 안티-앨리어싱(anti-aliasing) 필터(LPF)는 quadrature 복조와 데시메이션 사이에서 한번 수행한다. 이어서, 입력된 데이터에 대하여 square-root를 이용해 포락선 성분을 검출(740)하고, 로그 압축(750)을 통해 스케일을 맞춰준다. 마지막으로 전체적인 게인(760)을 곱해 전체 이득을 제어하여 1427×192 픽셀 크기를 갖는 로그 압축 데이터(Log compressed data)를 출력한다. 이때, 상기 로그 압축 데이터를 출력하기 위해, 총 $1427 \times 192 = 273,984$ 회의 연산이 수행된다.
- [98] 제3 렌더 사이클에서는 3×3 평균화 필터를 이용해 블랙홀 필터링(770)을 수행한다. 평균화를 수행하는 정도를 설정하기 위해, 사용자로부터 해당 픽셀의 값을 블랙홀로 간주하는 정도를 나타내는 문턱치값을 입력받아 문턱치값 3×3 의 윈도우의 중앙에 위치한 값이 상기 문턱치 값 이하일 때만 평균화한 결과를 출력하고, 그렇지 않은 경우에는 기존의 중앙에 위치한 값을 출력하도록 한다. 이때, 상기 3×3 평균화 필터의 경우에도 한 픽셀의 값을 연산하는데 있어, 이웃 좌표의 값이 필요하므로 앞선 렌더 사이클의 연산이 모두 끝난 뒤에 수행되어야 함에 따라 렌더 사이클을 분리한다.
- [99] 즉, 제3 렌더 사이클은 앞서 제2 렌더 사이클에서 출력된 1427×192 픽셀 크기를 갖는 로그 압축 데이터(Log compressed data)를 입력받아, 입력 받은 로그 압축

데이터에 대하여 3×3 평균화 필터링을 통해 블랙홀을 제거하고, 이에 따라, 1427×192 픽셀 크기를 갖는 블랙홀이 필터링된 데이터(Blackhole filtered data)를 출력한다. 이때, 상기 블랙홀이 필터링된 데이터를 출력하기 위해 수행되는 연산횟수는 총 $1427 \times 192 = 273,984$ 회이다.

[100] 이후, 제4 렌더 사이클에서는 3×3 소벨 필터(Sobel filter)를 이용해 데이터의 윤곽선을 검출하여 에지 강화(780)를 수행한다. 상기 소벨 필터링을 통해 에지 성분을 검출한 영상과 기존 영상을 더하여 에지 강화가 이루어지는데, 검출한 에지 성분의 영상을 더하는 정도를 나타내는 가중치 변수를 사용자로부터 입력받아 에지 강화의 강도를 설정한다. 특히, 3×3 소벨 필터의 경우도 한 픽셀의 값을 연산하는데 이웃 좌표의 값이 필요하므로, 앞선 렌더 사이클의 연산이 모두 끝난 뒤에 수행되어야 함에 따라, 렌더 사이클을 분리한다.

[101] 즉, 제4 렌더 사이클은 상기 제3 렌더 사이클로부터 출력된 1427×192 픽셀 크기를 갖는 블랙홀이 제거된 데이터(blackhole filtered data)를 입력받아, 3×3 소벨 필터링(Sobel filtering)을 통해 에지 성분을 검출하여 에지 강화를 수행함으로써, 1427×192 픽셀 크기를 갖는 에지가 강화된 데이터(Edge enhanced data)를 출력한다. 이때, 상기 에지가 강화된 데이터가 출력되기 위해 연산되는 횟수는 총 $1427 \times 192 = 273,984$ 회이다.

[102] 이어서, 제5 렌더 사이클에서는 스캔 변환(790)을 수행한다. 본 실시 예에서 사용하는 데이터는 컨벡스 프로브(convex probe)로부터 획득한 데이터이므로 컨벡스 스캔 변환을 수행한다. 이때, 상기 스캔 변환이 필터링의 경우처럼 이웃좌표의 값이 필요하지 않음에도 렌더 사이클을 분리한 것은 하나의 렌더 사이클에서의 연산 개수는 렌더 타겟의 픽셀 수에 비례하기 때문이다. 따라서, 스캔 변환의 출력은 폭이 830 픽셀, 높이가 640 픽셀이므로 한 픽셀의 연산에 필요한 연산량을 α 라 하면 총 $830 \times 640 \times \alpha (531200\alpha)$ 만큼의 연산이 필요하다.

[103] 즉, 제5 렌더 사이클은 앞서 제4 렌더 사이클로부터 출력된 1427×192 픽셀 크기를 갖는 에지가 강화된 데이터(Edge enhanced data)를 입력받아, 입력 받은 데이터에 대하여 디지털 스캔 변환을 통해 830×640 픽셀 크기를 갖는 최종 출력 이미지 (Digital scan conversed image)를 생성하여 출력한다. 이때, 연산수는 $830 \times 640 = 531,200$ 회가 된다.

[104] 반면 제4 렌더 사이클의 경우에는 $1427 \times 192 \times \alpha (273984\alpha)$ 만큼의 연산을 필요로 하게 되는데, 만약 렌더 사이클을 분리하지 않고, 한 렌더 사이클에서 같이 연산을 수행하는 경우, 한 렌더 사이클에서는 연산이 동시에 이루어지게 되므로, 필요 이상으로 소벨 필터링을 수행하게 되는 문제가 발생하므로, 이를 방지할 수 있다.

[105] 따라서, 필요하지 않은 연산수를 줄이기 위해 출력 프레임 버퍼의 픽셀 수를 고려하여 렌더 사이클을 분리하는 것이 중요하다.

[106] 도 7b는 렌더 사이클 처리과정을 나타낸 도 7a에서 초음파 신호를 처리하는데 요구되는 최소 연산만으로서 각각의 렌더 사이클을 재구성한 도면이다. 보다

구체적으로, 도 7a의 제2 렌더 사이클에서 테시메이션 과정(730)과 게인(760) 과정을 제거하였고, 또한 도 7a의 제3 렌더 사이클에 포함되는 블랙홀 필터링(770) 과정을 제거하였으며, 마지막으로 도 7a의 제4 렌더 사이클에 포함되는 에지 강화(780) 과정을 제거하였다. 따라서, 그 결과 전체 렌더 사이클은 3개로 감소하였으며, 도 7b와 같이 재구성되었다. 즉, 재구성된 제2 렌더 사이클의 Quadrature 복조(720) 과정과 포락선 검파(740) 과정은 '복조'의 관점에서 하나의 렌더 사이클 내에서 묶일 수 있다.

- [107] 물론 일련의 과정은 각각 별도의 렌더 사이클을 통해 구현될 수도 있으나, 성능 향상을 위해 가능한 렌더 사이클의 수를 최소화하는 것이 바람직하다. 앞서 소개한 도 3의 초기화 단계에서 master CPU로 수행해야 할 OpenGL ES API 수가 많아지고 렌더 사이클 사이에 데이터 이동이 있기 때문에 추가적인 시간이 소비된다. 따라서 렌더 사이클 수는 최소한으로 하는 것이 연산처리 속도 측면에서 유리하다. 그러나 필터링 연산이 포함되는 신호처리 블록의 경우 이전 신호처리 블록과 동일한 렌더 사이클에서 수행하면 연산이 중복적으로 수행된다. 연산 포인트에 대한 필터링을 수행할 때마다 필터링에 필요한 이웃 데이터에 대해서 이전 신호처리 블록의 연산을 수행해야 하기 때문이다. 이를 고려한 최적화된 렌더 사이클의 구성이 바로 도 7b에 해당한다. 한편, 도 7b에서 제거된 과정들은 구현상의 필요에 따라 각각 선택적으로 추가될 수 있다.
- [108] 도 10은 본 발명의 적용하여 획득한 Nexus7에서의 B-모드 이미징의 동작 화면이다.
- [109] 도 10에 도시된 바와 같이, 시간 이득 보상(TGC)을 8단계로 나누어 각 깊이별로 영상의 밝기를 사용자가 조절가능하게 하였으며, 전체적인 이득(overall gain) 또한 조절 가능하도록 하였다. 평균화 필터링의 정도를 설정하는 문턱치의 값과 에지 강화의 정도를 설정하는 가중치 값을 사용자가 조절할 수 있도록 사용자 인터페이스를 만든다.
- [110] 또한, 이러한 스마트 기기를 이용한 초음파 신호의 고속 병렬 처리 방법은 컴퓨터로 실행하기 위한 프로그램이 기록된 컴퓨터 판독가능 기록매체에 저장될 수 있다. 이때, 컴퓨터가 읽을 수 있는 기록매체는 컴퓨터 시스템에 의하여 읽혀질 수 있는 데이터가 저장되는 모든 종류의 기록 장치를 포함한다. 컴퓨터가 읽을 수 있는 기록 장치의 예로는 ROM, RAM, CD-ROM, DVD±ROM, DVD-RAM, 자기 테이프, 플로피 디스크, 하드 디스크(hard disk), 광데이터 저장장치 등이 있다. 또한, 컴퓨터가 읽을 수 있는 기록매체는 네트워크로 연결된 컴퓨터 장치에 분산되어 분산방식으로 컴퓨터가 읽을 수 있는 코드가 저장되고 실행될 수 있다.
- [111] 상기에서는 본 발명의 바람직한 실시 예에 대하여 설명하였지만, 본 발명은 이에 한정되는 것이 아니고 본 발명의 기술 사상 범위 내에서 여러 가지로 변형하여 실시하는 것이 가능하고 이 또한 첨부된 특허청구범위에 속하는 것은 당연하다.

산업상 이용가능성

- [112] 상술한 바와 같이, 본 발명을 통해 PC 환경에서처럼 스마트 기기에서도 초음파 신호처리를 GPU를 이용해 고속의 병렬형태로 수행할 수 있다. 이에 더하여 초음파 영상 시스템의 초소형화에 기여할 수 있다. 또한 남녀노소를 불문하고 널리 보급화된 스마트 기기에 단순히 어플리케이션 형태의 소프트웨어를 설치하고, 이를 통해 초음파 영상을 획득할 수 있다는 점에서 초음파 시스템의 비용 절감과 대중화에 기여할 수 있다. 초음파 영상에 대한 정보를 담고 있는 데이터를 원격으로 전달받아 스마트 기기에서의 신호처리를 통해 초음파 영상을 얻을 수 있다는 점에서 원격 진료의 보급화 또한 기대할 수 있다.
- [113] 본 발명의 스마트 기기를 이용한 초음파 신호의 고속 병렬 처리 방법은 PC 기반의 환경이 아닌 모바일 기반의 환경에서도 스마트 기기 내 모바일 GPU를 통해 초음파 신호에 대한 고속의 병렬처리를 수행함으로써, 의료 진단에 유용한 프레임율을 갖는 영상을 제공할 수 있는 효과가 있다.
- [114] 또한 본 발명의 스마트 기기를 이용한 초음파 신호의 고속 병렬 처리 방법은 그래픽스 파이프라인 구조를 이용하여 초음파 신호를 병렬 처리 시, 상기 그래픽스 파이프라인의 구조 중 프래그먼트 셰이더 단계에서 할당된 연산량 일부를 버텍스 셰이더 단계가 수행하도록 함으로써, 초음파 신호의 병렬 처리를 위해 수행되어야 하는 연산 처리를 나누어 처리하도록 하여 보다 신속하게 초음파 처리를 수행할 수 있는 효과가 있다.

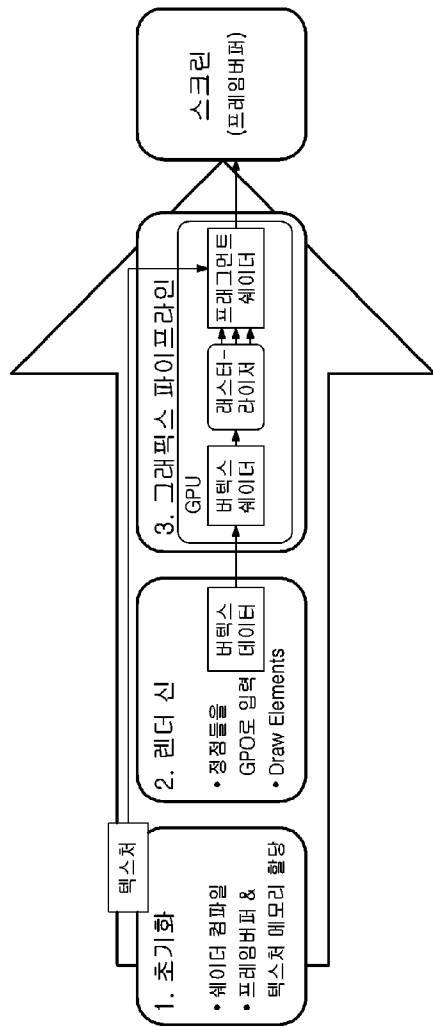
청구범위

- [청구항 1] 모바일 GPU(Graphic Processing Unit)를 구비하는 스마트 기기가 초음파 신호를 입력 받아 초음파 영상을 생성하는데 이용되는 초음파 신호의 고속 병렬 처리 방법에 있어서,
 상기 스마트 기기가 제1 렌더 사이클(Render Cycle)을 통해 빔포밍된 초음파 신호를 입력받아, 상기 초음파 신호로부터 DC 성분을 제거한 후, DC 성분이 제거된 초음파 신호로부터 in-phase 성분과 quadrature 성분을 나누어 각각 출력하는 단계;
 상기 스마트 기기가 제2 렌더 사이클을 통해 상기 in-phase 성분과 quadrature 성분을 갖는 초음파 신호에 대하여 quadrature 복조 및 포락선 검파 처리를 수행하는 단계; 및
 상기 스마트 기기가 제5 렌더 사이클을 통해 상기 제2 렌더 사이클의 처리 결과로서 획득된 초음파 신호에 대하여 스캔 변환을 수행하는 단계;
 를 포함하되,
 상기 렌더 사이클들은
 버텍스 셰이더(vertex shader) 단계, 래스터라이저(Rasterizer) 단계 및 프래그먼트 셰이더(fragment shader) 단계를 포함하는 그래픽스 파이프라인(Graphics pipeline) 구조로 이루어지는 것을 특징으로 하는 스마트 기기를 이용한 초음파 신호의 고속 병렬 처리 방법.
- [청구항 2] 제1항에 있어서,
 상기 버텍스 셰이더 단계는
 상기 모바일 GPU가 복수 개의 정점(vertex)을 입력받고, 입력받은 정점을 이용하여 공간영역을 할당한 후, 할당한 공간영역에 대한 공간좌표를 연산하여 연산결과를 varying 변수 형태로 생성하는 것을 특징으로 하는 스마트 기기를 이용한 초음파 신호의 고속 병렬 처리 방법.
- [청구항 3] 제2항에 있어서,
 상기 래스터라이저 단계는
 상기 모바일 GPU가 상기 버텍스 셰이더 단계에서 출력된 varying 변수와 대응하는 스크린 내 좌표값을 검색하고, 검색한 좌표값을 varying 변수 형태로 생성하는 것을 특징으로 하는 스마트 기기를 이용한 초음파 신호의 고속 병렬 처리 방법.
- [청구항 4] 제3항에 있어서,
 상기 버텍스 셰이더 단계는
 상기 모바일 GPU가 상기 래스터라이저 단계에서 생성된 스크린 내 좌표값의 주변에 위치하는 좌표값을 연산하고, 연산결과를 varying 변수 형태로 생성하는 과정을 더 포함하는 것을 특징으로 하는 스마트 기기를 이용한 초음파 신호의 고속 병렬 처리 방법.

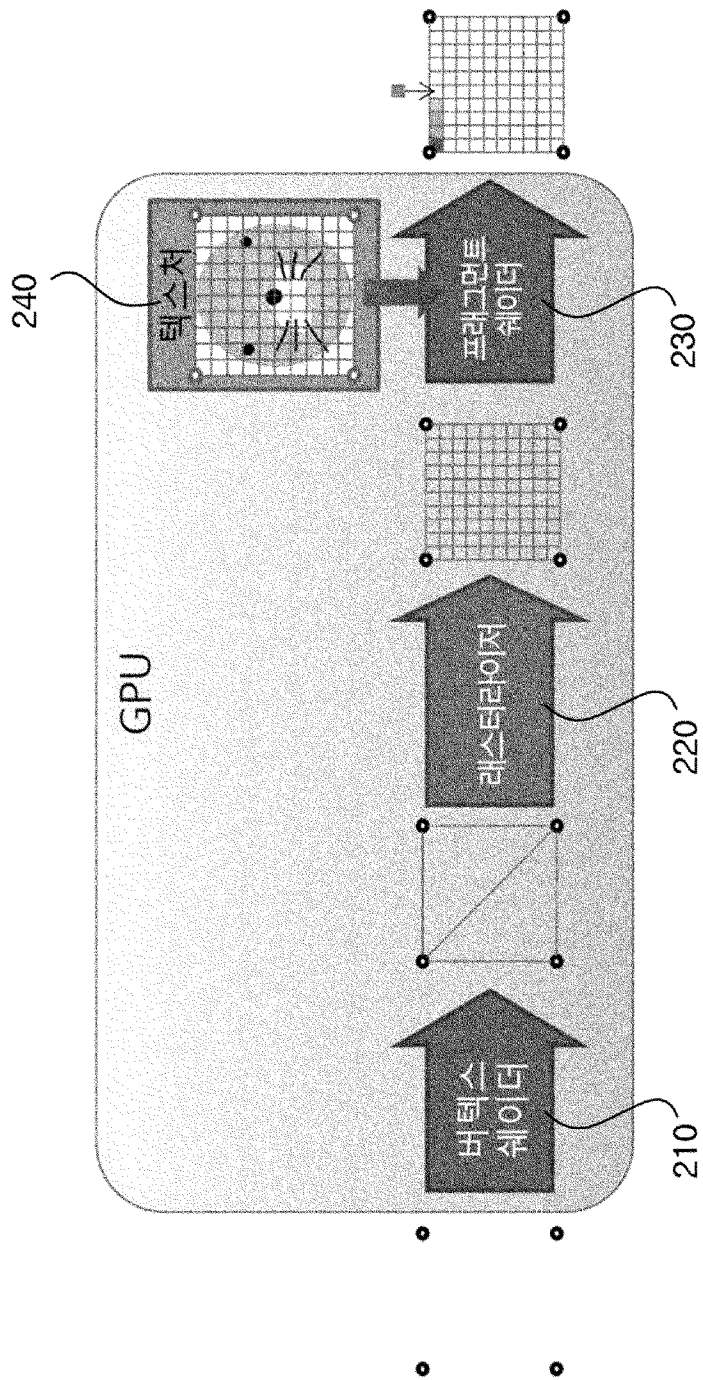
- [청구항 5] 제4항에 있어서,
상기 프래그먼트 셰이더 단계는
상기 모바일 GPU가 상기 래스터라이저 단계에서 생성된 스크린 내
좌표값에 대한 컬러를 연산하여 연산결과를 생성하는 것을 특징으로
하는 스마트 기기를 이용한 초음파 신호의 고속 병렬 처리 방법.
- [청구항 6] 제5항에 있어서,
상기 모바일 GPU가 상기 프래그먼트 셰이더 단계에서 생성된 스크린 내
좌표값에 대한 컬러 연산결과를 프레임버퍼에 저장하는 단계를 더
포함하는 것을 특징으로 하는 스마트 기기를 이용한 초음파 신호의 고속
병렬 처리 방법.
- [청구항 7] 제2항에 있어서,
상기 varying 변수는
상기 모바일 GPU의 사양에 따라 8개 내지 16개로 출력되는 것을
특징으로 하는 스마트 기기를 이용한 초음파 신호의 고속 병렬 처리 방법.
- [청구항 8] 제1항에 있어서,
상기 모바일 GPU가 이전 렌더 사이클에서 생성된 이미지를
프레임버퍼에 저장하면, 저장된 이미지를 RTT(Render To Texture)기술을
이용하여 상기 모바일 GPU의 메모리에 해당하는 텍스처(Texture)로
전달하고, 전달한 이미지를 다음 렌더 사이클의 그래픽스 파이프라인 중
프래그먼트 셰이더 단계로 전달하는 것을 특징으로 하는 스마트 기기를
이용한 초음파 신호의 고속 병렬 처리 방법.
- [청구항 9] 제1항에 있어서,
상기 모바일 GPU가 고정 함수 파이프라인(Fixed Function Pipeline) 구조를
이용하여 초음파 신호를 병렬 처리하는 것을 특징으로 하는 스마트
기기를 이용한 초음파 신호의 고속 병렬 처리 방법.
- [청구항 10] 제1항에 있어서,
OpenGL ES 3.0 환경에서 구현되는 것을 특징으로 하는 스마트 기기를
이용한 초음파 신호의 고속 병렬 처리 방법.
- [청구항 11] 제1항에 있어서,
상기 모바일 GPU가 상기 프래그먼트 셰이더 단계에서 수행되도록
할당된 연산과정의 일부를 상기 버텍스 셰이더 단계에서 미리
수행하도록 제어하는 것을 특징으로 하는 스마트 기기를 이용한 초음파
신호의 고속 병렬 처리 방법.
- [청구항 12] 제1항에 있어서,
상기 제2 렌더 사이클에서,
상기 스마트 기기가 상기 quadrature 복조 처리된 초음파 신호에 대해
데시메이션(decimation) 처리를 더 수행하는 것을 특징으로 하는 스마트
기기를 이용한 초음파 신호의 고속 병렬 처리 방법.

- [청구항 13] 제1항에 있어서,
상기 제2 렌더 사이클에서,
상기 스마트 기기가 상기 포락선 검파 처리된 초음파 신호에 대해 로그 압축 처리를 더 수행하는 것을 특징으로 하는 스마트 기기를 이용한 초음파 신호의 고속 병렬 처리 방법.
- [청구항 14] 제13항에 있어서,
상기 제2 렌더 사이클에서,
상기 스마트 기기가 상기 로그 압축 처리된 초음파 신호에 대해 영상의 전체 이득(gain)을 조절하는 이득 제어 처리를 더 수행하는 것을 특징으로 하는 스마트 기기를 이용한 초음파 신호의 고속 병렬 처리 방법.
- [청구항 15] 제1항에 있어서,
상기 제2 렌더 사이클의 수행 이후,
상기 스마트 기기가 제3 렌더 사이클을 통해 블랙홀로 간주하는 문턱치값을 입력받아, 상기 문턱치값과 상기 제2 렌더 사이클로부터 입력받은 초음파 신호의 크기를 상호 비교하여 상기 초음파 신호에 대한 블랙홀을 제거하는 단계;
를 더 포함하는 스마트 기기를 이용한 초음파 신호의 고속 병렬 처리 방법.
- [청구항 16] 제1항에 있어서,
상기 제2 렌더 사이클의 수행 이후,
상기 스마트 기기가 제4 렌더 사이클을 통해 상기 제2 렌더 사이클로부터 입력받은 초음파 신호에 대한 에지 강화를 수행하는 단계;
를 더 포함하는 스마트 기기를 이용한 초음파 신호의 고속 병렬 처리 방법.
- [청구항 17] 제1항에 따른 방법을 컴퓨터로 실행하기 위한 프로그램이 기록된 컴퓨터 판독가능 기록매체.

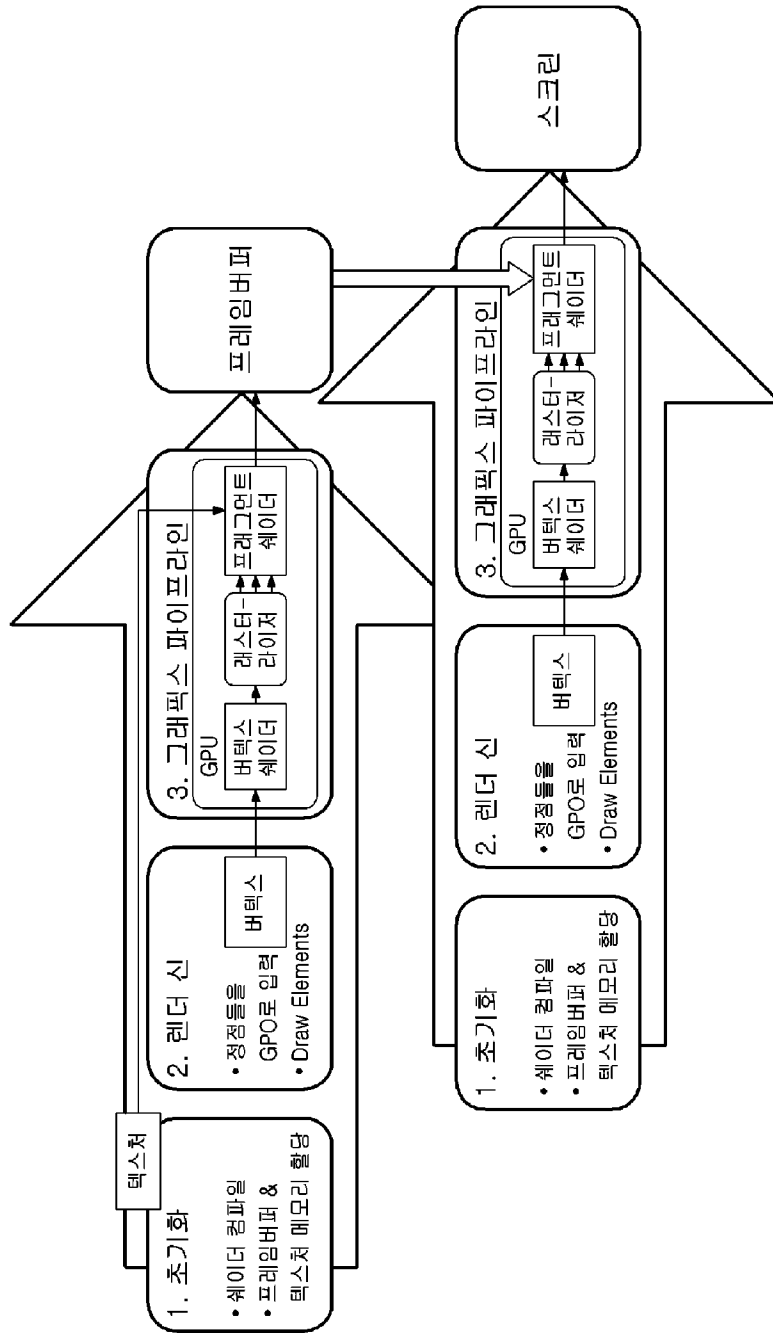
[도1]



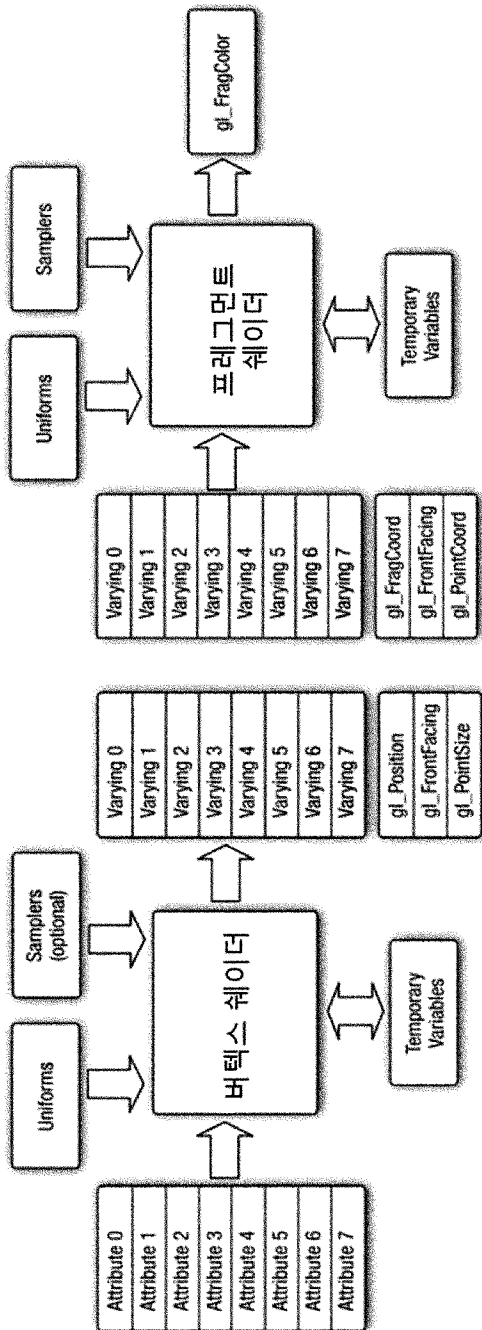
[도2]



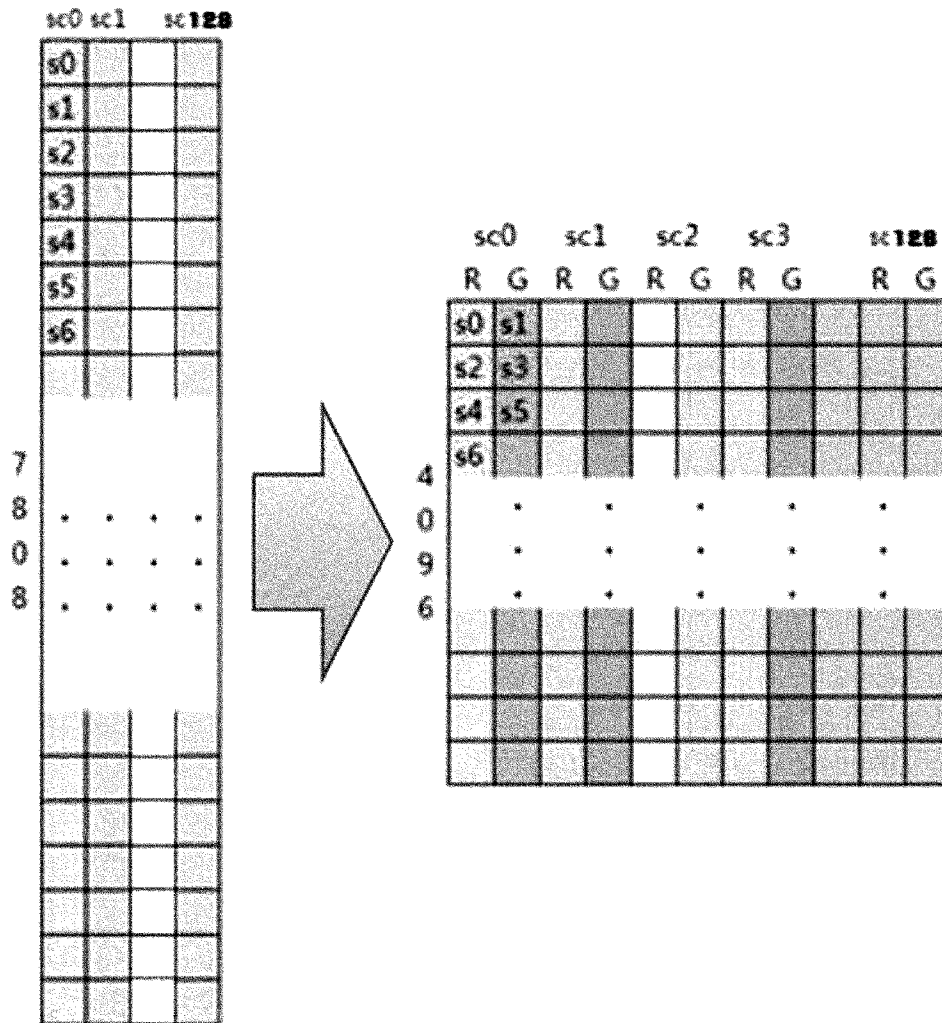
[도3]



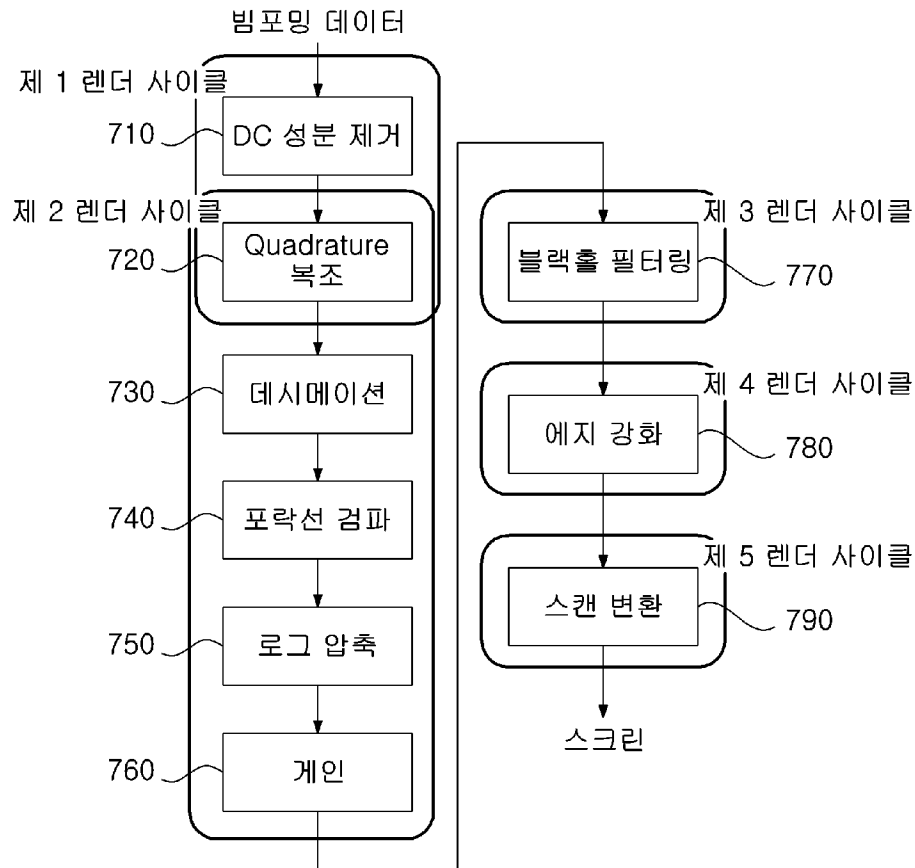
[도4]



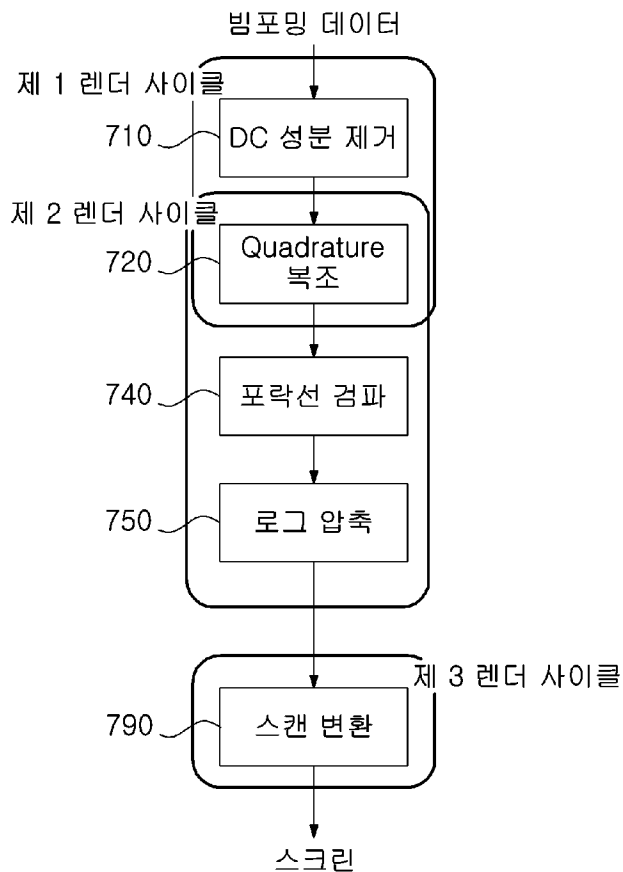
[도5]



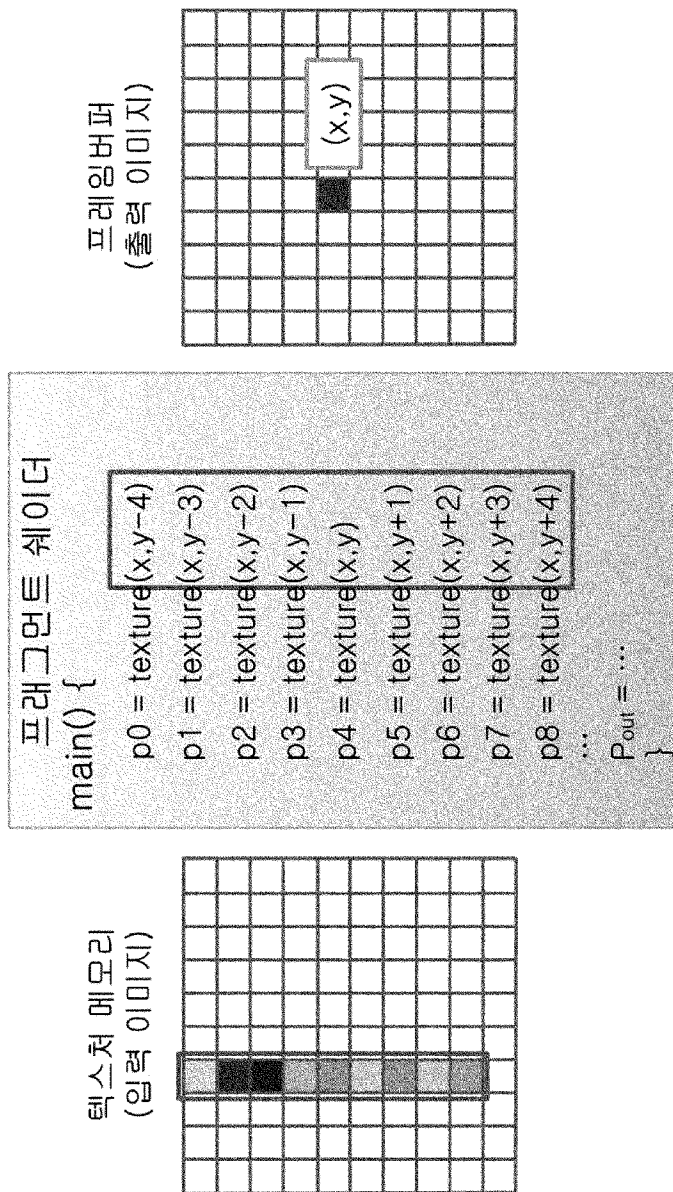
[도7a]



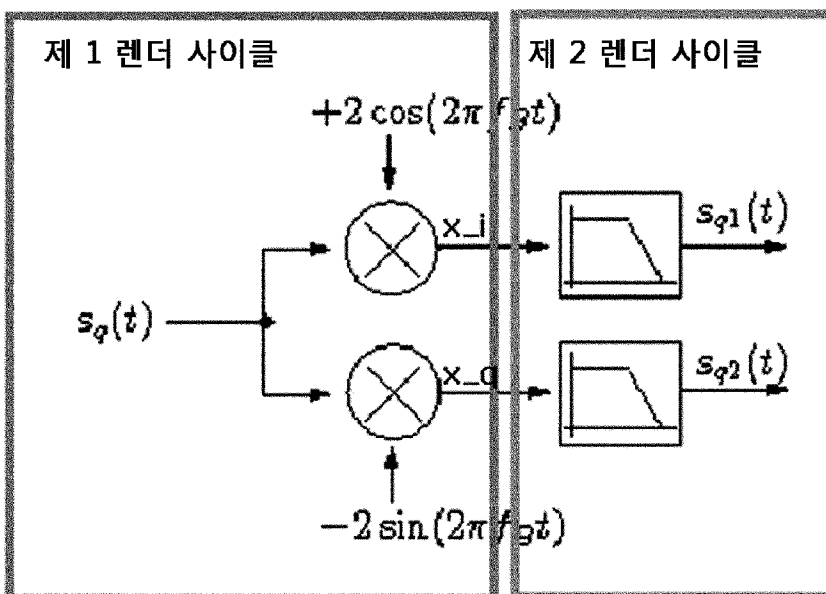
[도7b]



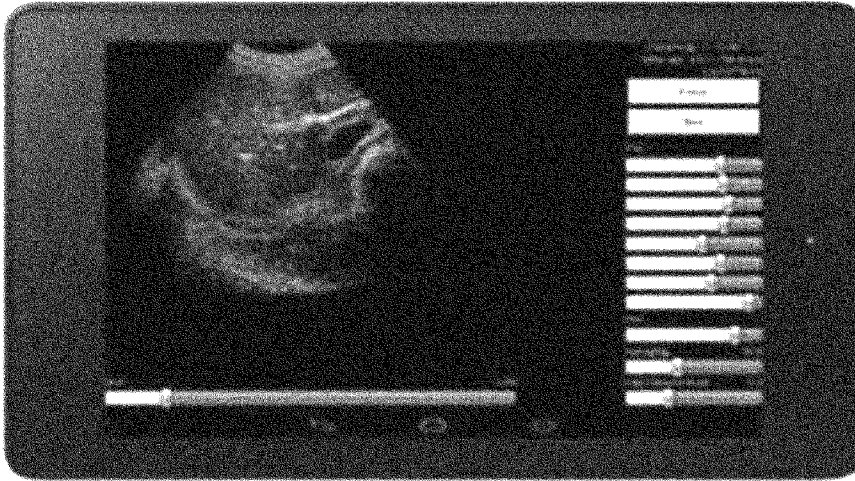
[도8]



[도9]



[도10]



INTERNATIONAL SEARCH REPORT

International application No.

PCT/KR2015/009039**A. CLASSIFICATION OF SUBJECT MATTER****A61B 8/14(2006.01)i**

According to International Patent Classification (IPC) or to both national classification and IPC

B. FIELDS SEARCHED

Minimum documentation searched (classification system followed by classification symbols)

A61B 8/14; G06F 19/00; G06T 15/80; G06F 17/00; G06T 3/40; A61B 8/00; H04B 1/40

Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched

Korean Utility models and applications for Utility models: IPC as above

Japanese Utility models and applications for Utility models: IPC as above

Electronic data base consulted during the international search (name of data base and, where practicable, search terms used)

eKOMPASS (KIPO internal) & Keywords: smart device, ultrasonic signal, GPU, graphics pipeline, vertex shader

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
Y	KR 10-2010-0006521 A (MEDISON CO., LTD.) 19 January 2010 See abstract, paragraph [0021], claim 1 and figure 1.	1,9,10,17
A		2-8,11-16
Y	KR 10-2013-0029149 A (SAMSUNG ELECTRONICS CO., LTD.) 22 March 2013 See abstract, paragraphs [0036]-[0061], [0092], claims 1-3 and figure 3.	1,9,10,17
Y	KR 10-2011-0108159 A (SAMSUNG ELECTRONICS CO., LTD.) 05 October 2011 See abstract, paragraph [0054].	10
A	KR 10-2009-0048243 A (MEDISON CO., LTD.) 13 May 2009 See abstract, claims 1-5 and figure 1.	1-17
A	KR 10-2013-0030663 A (SAMSUNG MEDISON CO., LTD.) 27 March 2013 See abstract, claim 1 and figure 1.	1-17



Further documents are listed in the continuation of Box C.



See patent family annex.

* Special categories of cited documents:

"A" document defining the general state of the art which is not considered to be of particular relevance

"E" earlier application or patent but published on or after the international filing date

"L" document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified)

"O" document referring to an oral disclosure, use, exhibition or other means

"P" document published prior to the international filing date but later than the priority date claimed

"T" later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention

"X" document of particular relevance; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone

"Y" document of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, such combination being obvious to a person skilled in the art

"&" document member of the same patent family

Date of the actual completion of the international search

08 DECEMBER 2015 (08.12.2015)

Date of mailing of the international search report

08 DECEMBER 2015 (08.12.2015)

Name and mailing address of the ISA/KR

Korean Intellectual Property Office
Government Complex-Daejeon, 189 Seonsa-ro, Daejeon 302-701,
Republic of Korea

Facsimile No. 82-42-472-7140

Authorized officer

Telephone No.

INTERNATIONAL SEARCH REPORT
Information on patent family members

International application No.

PCT/KR2015/009039

Patent document cited in search report	Publication date	Patent family member	Publication date
KR 10-2010-0006521 A	19/01/2010	EP 2143384 A1	13/01/2010
		JP 2010-017557 A	28/01/2010
		US 2010-0010344 A1	14/01/2010
KR 10-2013-0029149 A	22/03/2013	US 2013-0063440 A1	14/03/2013
		US 8952962 B2	10/02/2015
KR 10-2011-0108159 A	05/10/2011	US 2011-0234611 A1	29/09/2011
KR 10-2009-0048243 A	13/05/2009	EP 2058672 A1	13/05/2009
		JP 2009-119263 A	04/06/2009
		US 2009-0156934 A1	18/06/2009
KR 10-2013-0030663 A	27/03/2013	EP 2570081 A1	20/03/2013
		EP 2570081 B1	09/09/2015
		US 2013-0070993 A1	21/03/2013
		US 9002087 B2	07/04/2015

A. 발명이 속하는 기술분류(국제특허분류(IPC))

A61B 8/14(2006.01)i

B. 조사된 분야

조사된 최소문헌(국제특허분류를 기재)

A61B 8/14; G06F 19/00; G06T 15/80; G06F 17/00; G06T 3/40; A61B 8/00; H04B 1/40

조사된 기술분야에 속하는 최소문헌 이외의 문헌

한국등록실용신안공보 및 한국공개실용신안공보: 조사된 최소문헌란에 기재된 IPC

일본등록실용신안공보 및 일본공개실용신안공보: 조사된 최소문헌란에 기재된 IPC

국제조사에 이용된 전산 데이터베이스(데이터베이스의 명칭 및 검색어(해당하는 경우))

eKOMPASS(특허청 내부 검색시스템) & 키워드: 스마트기기, 초음파 신호, GPU, 그래픽스 파이프라인, 버텍스 셰이더

C. 관련 문헌

카테고리*	인용문헌명 및 관련 구절(해당하는 경우)의 기재	관련 청구항
Y	KR 10-2010-0006521 A (주식회사 메디슨) 2010.01.19 요약, 문단번호 [0021], 청구항 1 및 도면 1 참조.	1,9,10,17
A		2-8,11-16
Y	KR 10-2013-0029149 A (삼성전자주식회사) 2013.03.22 요약, 문단번호 [0036]-[0061],[0092], 청구항 1-3 및 도면 3 참조.	1,9,10,17
Y	KR 10-2011-0108159 A (삼성전자주식회사) 2011.10.05 요약, 문단번호 [0054] 참조.	10
A	KR 10-2009-0048243 A (주식회사 메디슨) 2009.05.13 요약, 청구항 1-5 및 도면 1 참조.	1-17
A	KR 10-2013-0030663 A (삼성메디슨 주식회사) 2013.03.27 요약, 청구항 1 및 도면 1 참조.	1-17

☐ 추가 문헌이 C(계속)에 기재되어 있습니다.☒ 대응특허에 관한 별지를 참조하십시오.

* 인용된 문헌의 특별 카테고리:

“A” 특별히 관련이 없는 것으로 보이는 일반적인 기술수준을 정의한 문헌

“T” 국제출원일 또는 우선일 후에 공개된 문헌으로, 출원과 상충하지 않으며 발명의 기초가 되는 원리나 이론을 이해하기 위해 인용된 문헌

“E” 국제출원일보다 빠른 출원일 또는 우선일을 가지나 국제출원일 이후에 공개된 선출원 또는 특허 문헌

“X” 특별한 관련이 있는 문헌. 해당 문헌 하나만으로 청구된 발명의 신규성 또는 진보성이 없는 것으로 본다.

“L” 우선권 주장에 의문을 제기하는 문헌 또는 다른 인용문헌의 공개일 또는 다른 특별한 이유(이유를 명시)를 밝히기 위하여 인용된 문헌

“Y” 특별한 관련이 있는 문헌. 해당 문헌이 하나 이상의 다른 문헌과 조합하는 경우로 그 조합이 당업자에게 자명한 경우 청구된 발명은 진보성이 없는 것으로 본다.

“O” 구두 개시, 사용, 전시 또는 기타 수단을 언급하고 있는 문헌

“&” 동일한 대응특허문헌에 속하는 문헌

“P” 우선일 이후에 공개되었으나 국제출원일 이전에 공개된 문헌

국제조사의 실제 완료일

2015년 12월 08일 (08.12.2015)

국제조사보고서 발송일

2015년 12월 08일 (08.12.2015)

ISA/KR의 명칭 및 우편주소

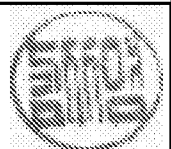
대한민국 특허청
(35208) 대전광역시 서구 청사로 189,
4동 (둔산동, 정부대전청사)

팩스 번호 +82-42-472-7140

심사관

김자영

전화번호 +82-42-481-8131



국제조사보고서에서 인용된 특허문헌	공개일	대응특허문헌	공개일
KR 10-2010-0006521 A	2010/01/19	EP 2143384 A1 JP 2010-017557 A US 2010-0010344 A1	2010/01/13 2010/01/28 2010/01/14
KR 10-2013-0029149 A	2013/03/22	US 2013-0063440 A1 US 8952962 B2	2013/03/14 2015/02/10
KR 10-2011-0108159 A	2011/10/05	US 2011-0234611 A1	2011/09/29
KR 10-2009-0048243 A	2009/05/13	EP 2058672 A1 JP 2009-119263 A US 2009-0156934 A1	2009/05/13 2009/06/04 2009/06/18
KR 10-2013-0030663 A	2013/03/27	EP 2570081 A1 EP 2570081 B1 US 2013-0070993 A1 US 9002087 B2	2013/03/20 2015/09/09 2013/03/21 2015/04/07