(19) 대한민국특허청(KR)
(12) 공개특허공보(A)

(11) 공개번호 10-2016-0087127
(43) 공개일자 2016년07월21일

(51) 국제특허분류(Int. Cl.)
G06T 1/20 (2006.01)
(52) CPC특허분류
G06T 1/20 (2013.01)
(21) 출원번호 10-2015-0005655
(22) 출원일자 2015년01월13일
심사청구일자 2015년01월13일

(71) 출원인
한발대학교 산학협력단
대전광역시 유성구 동서대로 125 (덕명동)
(72) 발명자
최해철
대전광역시 유성구 반석동로 33 반석마을5단지아파트 508동 802호
이강운
대전광역시 동구 우암로311번길 29 (가양동)
백아람
대전광역시 유성구 지족로 240 (지족동, 노은해랑숲마을5단지아파트) 504동 1501호
(74) 대리인
특허법인 플러스

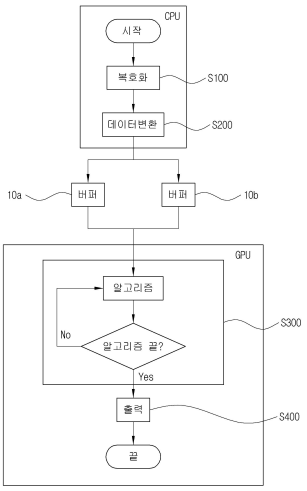
전체 청구항 수 : 총 6 항

(54) 발명의 명칭 **임베디드 CPU-GPU 분산처리를 이용한 영상처리 고속화 방법**

(57) 요 약

임베디드 CPU-GPU 분산처리를 이용한 영상처리 고속화 방법에 관한 것으로서, 더욱 상세하게, 임베드디 및 모바일 환경에서 영상처리를 함에 있어 CPU와 GPU사이에 복수의 버퍼를 구비하고 복수의 버퍼에 번갈아가며 데이터를 저장 및 읽기를 통해 손실 및 중복되는 프레임이 없게 하며, CPU-GPU의 분산처리 및 GPU의 다중알고리즘 구조를 통해 영상처리의 효율성을 높이는 임베디드 CPU-GPU 분산처리를 이용한 영상처리 고속화방법에 관한 것이다.

대 표 도 - 도1



이 발명을 지원한 국가연구개발사업
과제고유번호 1711005241
부처명 교육부
연구관리전문기관 한국연구재단
연구사업명 지역기초연구활성화
연구과제명 3D 영상 산업 인력 양성을 위한 다시점 비디오 부호화 및 자유시점 비디오 재현 기술 연구
기 여 율 1/1
주관기관 한밭대학교
연구기간 2012.05.01 ~ 2015.04.30

명 세 서

청구범위

청구항 1

임베디드 및 모바일 환경에서 CPU-GPU 병렬처리를 기반으로 영상데이터를 처리하는 방법으로,

카메라로부터 획득하거나, 메모리에 저장된 영상데이터를 복호화 하는 복호화단계(S100);

상기 복호화단계(S100)에서 복호화된 영상데이터를 GPU에서 사용할 텍스쳐 타입으로 변환하고, 변환된 텍스쳐 타입 데이터를 CPU-GPU사이에 위치한 복수의 버퍼(10)에 번갈아가며 순차적으로 저장하는 데이터변환단계(S200);

상기 데이터변환단계(S200)에서 상기 버퍼(10)에 저장된 데이터를 번갈아가며 순차적으로 불러와 미리 결정된 알고리즘을 적용하는 알고리즘단계(S300); 및

상기 알고리즘단계(S300)를 거친 데이터를 출력하는 출력단계(S400);

를 포함하며,

상기 복호화 단계(S100) 및 상기 데이터변환단계(S200)는 CPU에 의해 수행되고,

상기 알고리즘단계(S300) 및 상기 출력단계(S400)는 GPU에 의해 수행되는 것을 특징으로 하는 임베디드 CPU-GPU 분산처리를 이용한 영상처리 고속화 방법.

청구항 2

제 1항에 있어서,

상기 데이터변환단계(S200)에서 사용가능한 상기 버퍼(10)는 상기 알고리즘단계(S300)에서 사용중인 것 외의 상기 버퍼(10)인 것을 특징으로 하는 임베디드 CPU-GPU 분산처리를 이용한 영상처리 고속화 방법.

청구항 3

제 1항에 있어서,

상기 알고리즘단계(S300)에서 상기 버퍼(10)로부터 받은 데이터에 적용하는 알고리즘은 프레임버퍼오브젝트(Frame Buffer Object)를 이용하여 하나 이상인 것을 특징으로 하는 임베디드 CPU-GPU 분산처리를 이용한 영상처리 고속화 방법.

청구항 4

제 1항에 있어서,

상기 복호화단계(S100) 및 상기 데이터변환단계(S200)는 모듈화 되어 병렬처리 되는 것을 특징으로 하는 임베디드 CPU-GPU 분산처리를 이용한 영상처리 고속화 방법.

청구항 5

제 1항 내지 제 4항 중 선택되는 어느 한 항에 기재된 임베디드 CPU-GPU 분산처리를 이용한 영상처리 고속화 방법을 구현하기 위한 컴퓨터 판독 가능한 기록매체.

청구항 6

제 1항 내지 제 4항 중 선택되는 어느 한 항에 기재된 임베디드 CPU-GPU 분산처리를 이용한 영상처리 고속화 방법을 구현하기 위한 컴퓨터 판독 가능한 기록매체에 저장된 프로그램.

발명의 설명

기 술 분 야

[0001] 본 발명은 임베디드 및 모바일에서 영상처리 고속화에 관한 것으로, 종래 보편적으로 활용되는 임베디드 및 모바일에서의 영상처리에 비해 CPU(중앙 처리 장치, Central Processing Unit)와 GPU(그래픽 연산 장치, Graphic Processing Unit)의 분산처리, CPU와 GPU사이에 위치한 복수의 버퍼 및 다중알고리즘 구조를 통한 영상처리 고속화 방법에 관한 것이다.

[0002] 반도체 기술의 발전과 공정 미세화로 CPU는 처음 등장한 이후 점점 성능이 높아져만 갔다. CPU의 성능을 높이는 방법은 CPU에 들어가는 트랜지스터의 숫자를 늘리는 것이었다. 그러나 이 방법은 CPU의 클럭이 4GHz에 이르면서 한계에 부딪히게 된다. CPU에 집적되는 트랜지스터의 숫자가 늘어나면서 그에 비례하여 CPU가 소모하는 전력이 늘어나고, 소모 전력이 늘어나면 발열도 그만큼 증가하는데 CPU코어의 재료인 반도체는 도체와 반대로 열을 받으면 저항이 줄어들기 때문에 누설전류가 생기는 동시에 과전류가 흘러 코어가 타버리는 상황이 온것이다. 이러한 한계를 극복하기 위해 1개의 CPU내에 동일한 성능으로 작동하는 프로세서를 여러개를 넣은 멀티코어 프로세서(Multi Core Processor)가 등장하게 되었다. 상기 멀티코어 프로세서는 CPU뿐만 아니라 GPU에도 적용되는 개념이다. 컴퓨터의 그래픽처리능력 가속을 위해 1980년대 처음 등장한 GPU또한 지속적으로 성능향상을 거듭해 현재 단순 계산능력으로는 CPU를 능가할 정도가 되었다. 이러한 GPU도 CPU처럼 성능향상의 벽에 막히게 되었고, CPU와 마찬가지로 여러 개의 독립적인 코어를 가진 GPU가 등장하게 되었다. 이런 멀티코어 프로세서로 만들어진 CPU및 GPU 활용하기 위한 여러 가지 방법이 제안되고 있다. 대표적인 예가 병렬처리와 GPGPU(범용목적 GPU, General Purpose Graphic Processing Unit)라는 개념이다. 병렬처리란 컴퓨터가 처리할 한 작업을 여러 개로 나눠 CPU 및 GPU에 존재하는 각각의 독립적인 프로세서에게 할당하여 동시에 처리하는 것이다. 이 방법은 오래전부터 고성능연산에 사용되던 방법으로, 흔히 말하던 슈퍼컴퓨터에 적용되던 개념이었지만 최근 컴퓨터 이용에서 발열과 전력 소모에 대한 관심이 높아지는 것과 더불어 멀티코어 프로세서의 등장과 함께 컴퓨터 구조의 강력한 패러다임으로 주목받게 되었다. GPGPU란 GPU가 본래목적인 그래픽처리가 아닌 CPU가 하는 작업을 대신하는 것이다. CPU는 컴퓨터의 여러 장치로부터 오는 신호를 처리하기 위하여 비순차 실행처럼 명령어 처리 자체를 하기 위한 장치가 많기 때문에 계산을 위한 장치는 많지 않다. 특히 메모리와 CPU의 속도차이를 개선시키기 위한 버퍼역할을 하는 캐시메모리(Cache memory)는 고성능 CPU면적의 절반 이상을 차지할 정도이기 때문에 실제 계산을 하는 장치는 개수가 GPU에 비해 상대적으로 적다. 이에 비해 GPU는 본래 목적이 계산량이 많은 영상처리를 전담하기 위해 장치 대부분이 단순계산능력을 높이기 위한 장치로 구성되어있다. 이런 차이로 CPU와 GPU의 계산능력의 차이가 일어나게 되었고 상기 GPGPU라는 개념이 등장하게 되었다. 이런 GPGPU는 영상처리와 같은 대용량 단순계산 위주의 작업에 유용하며 조건이나 분기가 일어나는 프로그래밍에서는 오히려 성능이 감소하는 특징을 보여준다.

[0003] 상술한 특징을 가지는 병렬처리 및 GPGPU를 사용하려면 해당 프로그램이 가능한 CPU 및 GPU뿐만 아니라 소프트웨어 적으로 변환시키는 레이어 혹은 라이브러리가 필요한데, 주로 nVIDIA사에서 만든 CUDA(Compute Unified Device Architecture)나 OpenCL 및 OpenGL이 사용된다.

[0004] 한편, 휴대용 전자기기의 발전과 함께 카메라를 갖춘 모바일 기기가 보편화되면서 임베디드 및 모바일에서 영상처리를 이용한 다양한 응용이 확산되고 있다. 그러나 모바일은 PC에 비해 전력공급이 제한적이고, PC와 모바일 내에 장착되는 CPU와 GPU의 크기차이로 인한 낮은 연산능력, 작은 메모리 및 발열과 같은 물리적 제약이 있었고, 임베디드 GPU는 CPU와 대역폭 차이, 프로그래밍 인터페이스 부족, 적은 코어와 쉐이더 유닛 수와 같은 구조적 제약이 많기 때문에 기존의 방법으로는 처리가 힘든 실정이다. 이를 해결하고자 PC에서 사용하는 GPGPU를 임베디드 및 모바일에서도 응용하고자 하는 연구가 계속되고 있으며, OpenGL ES 2.0과 같이 임베디드 및 모바일에서 사용 가능한 그래픽처리 어플리케이션이 개발되고 있다.

배 경 기 술

[0005] 상술한 바와 같이 임베디드 및 모바일에서 영상처리를 이용한 다양한 응용이 확산되고 있다. 상술한 병렬처리 및 GPGPU를 임베디드 및 모바일 환경에서 이용하려는 노력이 대표적이다. 선행문헌1(CPU and GPU Parallel Processing for Mobile Augmented Reality)에서는 모바일 환경에서 CPU와 GPU의 병렬처리를 통한 영상처리에 대해 개시하고 있다. 그러나 이런 기존의 병렬처리를 이용한 영상처리는 순차적으로 한 프레임씩 처리하는 구조보다 각 프로세서의 대기시간이 없어 총 수행시간이 감소하는 장점이 있지만, CPU와 GPU의 서로 다른 작업속도 차이로 인한 프레임 누락 및 작업하는 프레임이 중복된다는 문제점을 가지고 있다.

[0006] 또한 종래기술의 다른 단점으로는 단일알고리즘 처리구조를 들 수 있다. 보통의 영상처리는 여러 단계의 알고리즘으로 완성되지만 단일알고리즘 처리구조는 한 번의 알고리즘만 처리되어 화면에 출력하게 된다. 이러한 단일알고리즘 처리구조를 이용하여 여러 알고리즘을 처리하면 도 2와 같이 CPU와 GPU간의 데이터 전송 및 변환이 알고리즘 수만큼 발생하게 된다. PC에서는 임베디드 및 모바일 환경과 비교하여 대역폭이 크고, CPU와 GPU의 속도차이를 보완해줄 여러 가지 장치가 있지만 모바일 및 프로세서는 대역폭이 낮고, CPU와 GPU의 속도차이를 보완해줄 장치 또한 성능상의 제약이 있기 때문에 비효율적이라는 문제점이 있었다.

선행기술문헌

비특허문헌

[0007] (비특허문헌 0001) (선행문헌1) "CPU and GPU Parallel Processing for Mobile Augmented Reality", Image and Signal Processing(CISP), 6th International Congress, ppp. 133-137, Dec, 2013

발명의 내용

해결하려는 과제

[0008] 본 발명은 상기한 바와 같은 종래 기술을 개선시키고자 하는 것으로, 임베디드 및 모바일 환경에서 CPU와 GPU사이에 복수의 버퍼를 구비하고, GPU에서 다중알고리즘 처리구조를 구현하며 CPU와 GPU의 분산처리를 통해 영상처리의 효율성을 높이고자 하는 것이다.

과제의 해결 수단

[0009] 임베디드 및 모바일 환경에서 CPU-GPU 병렬처리를 기반으로 영상데이터를 처리하는 방법으로, 카메라로부터 획득하거나, 메모리에 저장된 영상데이터를 복호화 하는 복호화단계(S100), 상기 복호화단계(S100)에서 복호화된 영상데이터를 GPU에서 사용할 텍스쳐 타입으로 변환하고, 변환된 텍스쳐 타입 데이터를 CPU-GPU사이에 위치한 복수의 버퍼(10)에 번갈아가며 순차적으로 저장하는 데이터변환단계(S200), 상기 데이터변환단계(S200)에서 상기 버퍼(10)에 저장된 데이터를 번갈아가며 순차적으로 불러와 미리 결정된 알고리즘을 적용하는 알고리즘단계(S300) 및 상기 알고리즘단계(S300)를 거친 데이터를 출력하는 출력단계(S400)를 포함하며 상기 복호화 단계(S100) 및 상기 데이터변환단계(S200)는 CPU에 의해 수행되고 상기 알고리즘단계(S300) 및 상기 출력단계(S400)는 GPU에 의해 수행된다.

[0010] 상기 데이터변환단계(S200)에서 사용가능한 상기 버퍼(10)는 상기 알고리즘단계(S300)에서 사용중인 것 외의 상기 버퍼(10)이다.

[0011] 상기 알고리즘단계(S300)에서 상기 버퍼(10)로부터 받은 데이터에 적용하는 알고리즘은 프레임버퍼오브젝트(Frame Buffer Object)를 이용하여 하나 이상이다.

[0012] 상기 복호화단계(S100) 및 상기 데이터변환단계(S200)는 모듈화 되어 병렬처리 된다.

[0013] 본 발명의 상기 임베디드 CPU-GPU 분산처리를 이용한 영상처리 고속화 방법을 구현하기 위한 프로그램이 저장된 컴퓨터 판독 가능한 기록매체가 제공되는 것을 특징으로 한다.

[0014] 아울러, 임베디드 CPU-GPU 분산처리를 이용한 영상처리 고속화 방법을 구현하기 위해, 컴퓨터 판독 가능한 기록매체에 저장된 프로그램이 제공되는 것을 특징으로 한다.

발명의 효과

[0015] 본 발명에 따른 임베디드 CPU-GPU 분산처리를 이용한 영상처리 고속화 방법은 임베디드 및 모바일 환경에서 CPU와 GPU사이에 위치한 복수의 버퍼로부터 번갈아가며 데이터를 저장하고 읽음으로써 데이터가 손실되거나 작업이 중복되는 것을 없애는 효과가 있다.

[0016] 또한 임베디드 및 모바일환경에 설치된 GPU에서 다중알고리즘 처리구조와 CPU-GPU의 분산처리를 통해 영상처리의 효율성을 높이는 효과가 있다.

도면의 간단한 설명

[0017] 도 1은 본 발명의 일 실시예에 따른 순서도.

도 2는 종래 단일알고리즘 처리구조.

도 3은 본 발명의 일 실시예에 따른 다중알고리즘 처리구조.

발명을 실시하기 위한 구체적인 내용

[0018] 이하 첨부된 도면을 참조하면서 본 발명에 따른 바람직한 실시예를 상세히 설명하기로 한다. 이에 앞서, 본 명세서 및 청구범위에 사용된 용어나 단어는 통상적이거나 사전적인 의미로 한정해서 해석되어서는 아니 되며, 발명자는 그 자신의 발명을 가장 최선의 방법으로 설명하기 위해 용어의 개념을 적절하게 정의할 수 있다는 원칙에 입각하여, 본 발명의 기술적 사상에 부합하는 의미와 개념으로 해석되어야만 한다.

[0019] 따라서 본 명세서에 기재된 실시예와 도면에 도시된 구성은 본 발명의 가장 바람직한 일 실시예에 불과할 뿐이고 본 발명의 기술적 사상을 모두 대변하는 것은 아니므로, 본 출원시점에 있어서 이들을 대체할 수 있는 다양한 균등물과 변형예들이 있을 수 있음을 이해하여야 한다.

[0020] 도 1은 본 발명의 일실시예에 따른 임베디드 CPU-GPU 분산처리를 이용한 영상처리 고속화 방법에 대해 간략하게 도시한 도면이다. 도 1을 참조로 하여 본 발명의 일실시예에 따른 부호화 모드 결정 방법에 대해서 상세히 설명한다.

[0021] 도 1에 도시된 바와 같이, 본 발명의 일실시예에 따른 임베디드 CPU-GPU 분산처리를 이용한 영상처리 고속화 방법은 임베디드 및 모바일 환경에서 CPU-GPU 병렬처리를 기반으로 영상데이터를 처리하는 방법으로, 복호화단계(S100), 데이터변환단계(S200), 알고리즘단계(S300) 및 출력단계(S400)를 포함한다. 상기 복호화단계(S100) 및 상기 데이터변환단계(S200)는 CPU에 의해서 수행되고, 상기 알고리즘단계(S300) 및 상기 출력단계(S400)는 GPU에 의해서 수행된다.

[0022] 상기 복호화단계(S100)는 카메라로부터 획득하거나 메모리에 저장된 영상데이터를 CPU에서 사용가능한 데이터로 복호화 한다. 카메라로부터 받아들이는 영상은 아날로그 영상신호로써 컴퓨터가 다룰 수 있는 디지털 신호로의 변환이 필요하다. 이 아날로그 방식의 영상신호를 단위 펄스의 유무로 구분되는 디지털 부호로 변환하는 작업을 부호화라고 하고 그 반대 작업을 복호화라고 한다. 이 부호화는 아날로그신호를 디지털 신호로의 변환 뿐만 아니라 방대한 양의 아날로그 신호를 압축해서 메모리나 하드디스크에 저장하는 것도 포함한다. 상기 복호화단계(S100)에서 복호화 하는 작업은 이 부호화되어 저장된 데이터에서 영상처리가 가능한 데이터로 변환하는 작업이다.

[0023] 상기 복호화단계(S100)에서 복호화된 데이터는 CPU에서 사용하는 데이터타입으로, GPU에서 사용하기 위해서는 GPU에서 사용하는 데이터타입인 텍스쳐로 바꿔줄 필요가 있다. 상기 이유로 상기 데이터변환단계(S200)는 상기 복호화단계(S100)에서 복호화된 영상데이터를 GPU에서 사용할 텍스쳐 타입으로 변환하고, 변환된 텍스쳐 타입 데이터를 CPU-GPU사이에 위치한 두 개의 버퍼(10)에 번갈아가며 순차적으로 저장한다.

[0024] 상기 복호화단계(S100) 및 상기 데이터변환단계(S200)는 모듈화 되어 병렬처리 된다. 상기 두 단계를 모듈화 및

병렬처리하는 것는 영상 데이터 타입 변환 시간을 줄이기 위함이다. 상기 모듈화 후 병렬처리는 CPU내 독립 코어가 여러 개인 멀티코어 CPU일 경우 가능하며, 이 때 병렬처리는 CPU와 GPU의 병렬처리가 아닌 CPU내 독립 코어별로 작업을 할당하여 병렬처리 하는 것을 의미한다.

[0025] 상기 알고리즘단계(S300)는 상기 두 개의 버퍼(10)에 저장된 데이터를 번갈아가며 순차적으로 불러와 미리 결정된 알고리즘을 적용하며, 상기 알고리즘은 프레임버퍼오브젝트(Frame Buffer Object)를 이용하여 하나 이상이다.

[0026] 도 2는 종래의 단일알고리즘 처리구조를 도시한 것이다. 도 2와 같은 단일알고리즘 처리구조를 사용하면 여러개의 알고리즘으로 이루어지는 영상처리시 CPU와 GPU사이의 데이터 전송이 알고리즘의 수만큼 일어나게 되고, CPU와 GPU사이에 데이터가 전송되기 때문에 각 프로세서에서 사용하는 데이터 타입으로 변환하는 작업 또한 필요하다. 이 데이터를 변환하는 시간과 데이터를 전송하는 시간은 알고리즘 수만큼 총 작업시간에 포함되기 때문에 비효율적이었다.

[0027] 도 3은 이를 극복하고자 프레임버퍼오브젝트를 이용한 다중알고리즘 처리구조를 도시한 것으로, CPU에서 데이터를 GPU로 데이터를 넘겨주면 GPU내에서 미리 정해진 하나 이상의 알고리즘을 적용하여 출력하는 구조이다. 구체적으로는 GPU의 프레임버퍼로 전송되는 영상을 동일한 크기와 영역을 만들어 텍스처메모리의 프레임버퍼오브젝트에 복사한다. 프레임버퍼로 전송된 영상은 오프-스크린 렌더링(Off-Screen Rendering)명령을 이용하여 화면에 출력하지 않고 소비되도록 한다. 프레임버퍼오브젝트에 저장된 데이터는 정해진 복수의 알고리즘을 통해 사용자가 원하는 작업을 하게 되고 프레임버퍼를 통해 출력하게 된다.

[0028] 상기 데이터변환단계(S200)와 상기 알고리즘단계(S300)에서 사용하는 상기 버퍼(10)란 데이터를 한 곳에서 다른 한 곳으로 전송하는 동안 일시적으로 그 데이터를 보관하는 메모리의 영역으로, 상기 버퍼(10)를 이용하는 것은 병렬처리시 CPU와 GPU의 작업속도차이로 인해 작업하는 프레임의 누락되는 것을 막아 영상의 정확도를 높이고, 중복되는 프레임작업을 막아 작업시간을 단축하기 위함이다. 종래의 순차처리구조에서는 CPU가 첫 번째 프레임의 데이터를 변환시키고 GPU에 데이터를 넘겨주면, GPU가 정해진 영상처리를 한 후 출력하고 다시 CPU가 두 번째 프레임의 데이터 타입을 변환시키는 식의 순차처리구조를 사용했다. 상기 순차처리구조는 손실되는 프레임이 없다는 장점이 있었지만, 한 프로세서가 작업하는 동안 다른 프로세서는 대기하기 때문에 총 작업시간이 증가하는 단점이 있다. 이를 극복하고자 CPU가 첫 번째 프레임의 작업을 끝내고 GPU로 데이터를 보내면 대기시간 없이 바로 두 번째 프레임의 작업을 시작하고, GPU는 CPU가 끝낸 첫 번째 프레임의 데이터로 작업을 시작하는 방식인 병렬처리구조를 사용하기 시작했다. 이는 각 프로세서의 대기시간이 없어 총 작업시간이 감속했지만, 서로 다른 작업 속도 차이로 인해 작업 대상이 되는 프레임이 누락되거나 중복되는 단점을 가지게 되었다. 그러나 상기 버퍼(10)를 CPU와 GPU사이에 두고 사용하게 되면 CPU와 GPU의 작업속도 차이를 보완함으로써 프레임 누락을 방지하여 영상처리의 정확성을 향상시키고, 프레임을 중복처리하지 않아 불필요한 GPU사용을 방지하는 효과가 있다. 상기 버퍼(10)가 두 개인 이유는 임베디드 및 모바일 환경에서의 부족한 CPU의 용량을 고려한 것이다. 상기 데이터변환단계(S200)에서 CPU는 각각의 상기 버퍼(10)에 번갈아가며 순차적으로 데이터를 저장하게 되고, 상기 알고리즘단계(S300)에서 GPU는 각각의 버퍼(10)에서 순차적으로 데이터를 호출하게 된다. 상기 CPU와 GPU가 번갈아가며 순차적으로 각각 두 개의 버퍼(10)에 저장하고 불러올 때, GPU가 사용중인 버퍼(10)는 CPU가 사용하지 못한다. 이는 GPU가 버퍼(10)로부터 데이터를 불러오는 중 CPU가 해당 버퍼(10)를 사용하게 되면 데이터를 삭제하고 새로운 데이터를 사용하면 데이터의 누락이 발생할 수 있기 때문이다.

[0029] 상기 출력단계(S400)는 상기 알고리즘단계(S300)에서 처리된 데이터를 화면에 출력한다.

[0030] 하기 표 1은 상기 제안하는 방법과 기존의 순차처리구조, 병렬처리구조의 작업결과를 비교한 것이다. 실험은 ARM v7기반의 듀얼코어 1.3Ghz CPU와 트리플코어로 구성된 SGX 543MP3 GPU 그리고 1G 메모리로 구성된 하드웨어를 가지고 안드로이드 환경의 모바일 기기로 수행했으며 프로그래밍 언어는 OpenGL ES 2.0을 사용했다. 실험방법은 500프레임 길이의 영상을 그레이스케일(Grayscale) 알고리즘과 소벨 엣지 검출(Sobel Edge detector) 알고리즘을 동시에 적용하고 그 작업시간을 비교했다.

표 1

[0031]

해상도	작업시간/작업 프레임수	순차처리구조	병렬처리구조	본발명
480P	시간(초)	15.0	8.2	10.0
	프레임	500	411	500
720P	시간(초)	24.8	14.9	19.5
	프레임	500	374	500
1080P	시간(초)	47.1	35.5	33.7
	프레임	500	542	500

[0032] 상기 표 1을 보면 순차처리구조는 작업시간이 다른 작업방식에 비해 느리지만 손실되는 프레임이 없는 것을 알 수 있다. 그에 비해 병렬처리구조는 작업시간이 순차처리구조에 비해 빠르지만 해상도가 480P 및 720P에서는 프레임이 500에 미치지 못해 데이터가 손실되고, 해상도가 1080P에서는 일부 프레임을 이중으로 처리해 불필요한 작업시간이 늘어난다. 상기 본 발명에서 제안하는 방법은 순차처리와 병렬처리의 장장점을 합해놓은 방법으로 해상도가 480P와 720P에서는 병렬처리구조 보다는 느리지만 데이터손실이 없고, 해상도가 1080P에서는 작업속도가 오히려 병렬처리구조보다 빠르다. 상기 실험의 결과를 토대로 판단해보면 본 발명에서 제안하는 방법은 작업량이 많아질수록 상기 순차처리구조나 병렬처리구조보다 빠르며 데이터의 손실 또한 최소화 할 수 있다는 것을 알 수 있다.

[0033] 상술한 임베디드 CPU-GPU 분산처리를 이용한 영상처리 고속화 방법은 이를 구현하기 위한 명령어들의 프로그램이 유형적으로 구현됨으로써, 컴퓨터를 통해 판독될 수 있는 기록매체에 포함되어 제공될 수도 있음을 당업자들이 쉽게 이해할 수 있을 것이다. 다시 말해, 다양한 컴퓨터 수단을 통하여 수행될 수 있는 프로그램 명령 형태로 구현되어, 컴퓨터 판독 가능한 기록매체에 기록될 수 있다. 상기 컴퓨터 판독 가능한 기록매체는 프로그램 명령, 데이터 파일, 데이터 구조 등을 단독으로 또는 조합하여 포함할 수 있다. 상기 컴퓨터 판독 가능한 기록매체에 기록되는 프로그램 명령은 본 발명을 위하여 특별히 설계되고 구성된 것들이거나 컴퓨터 소프트웨어 당업자에게 공지되어 사용 가능한 것일 수도 있다. 상기 컴퓨터 판독 가능한 기록매체의 예에는 하드 디스크, 플로피 디스크 및 자기테이프와 같은 자기 매체(magnetic media), CD-ROM, DVD와 같은 광기록매체(optical media), 플롭티컬 디스크(floptical disk)와 같은 자기-광 매체(magneto-optical media), 및 롬(ROM), 램(RAM), 플래시 메모리, USB 메모리 등과 같은 프로그램 명령을 저장하고 수행하도록 특별히 구성된 하드웨어 장치가 포함된다. 상기 컴퓨터 판독 가능한 기록매체는 프로그램 명령, 데이터 구조 등을 지정하는 신호를 전송하는 반송파를 포함하는 광 또는 금속선, 도파관 등의 전송 매체일 수도 있다. 프로그램 명령의 예에는 컴파일러에 의해 만들어지는 것과 같은 기계어 코드뿐만 아니라 인터프리터 등을 사용해서 컴퓨터에 의해서 실행될 수 있는 고급 언어 코드를 포함한다. 상기 하드웨어 장치는 본 발명의 동작을 수행하기 위해 하나 이상의 소프트웨어 모듈로서 작동하도록 구성될 수 있으며, 그 역도 마찬가지이다.

[0034] 본 발명은 상기한 실시예에 한정되지 아니하며, 적용범위가 다양함은 물론이고, 청구범위에서 청구하는 본 발명의 요지를 벗어남이 없이 다양한 변형 실시가 가능한 것은 물론이다.

부호의 설명

[0035] 10 : 버퍼

S100 : 복호화단계

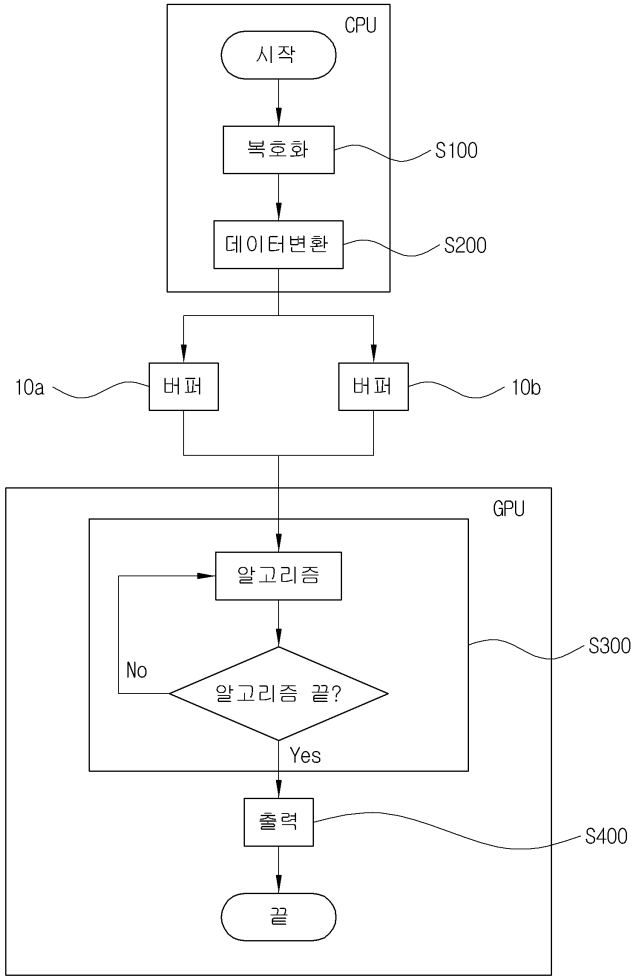
S200 : 데이터변환단계

S300 : 알고리즘단계

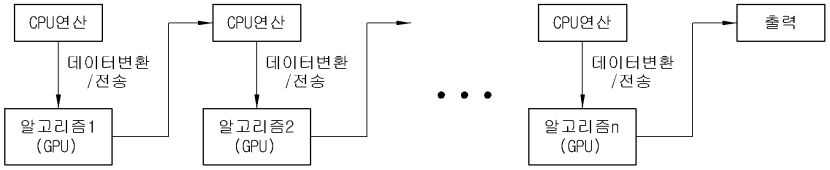
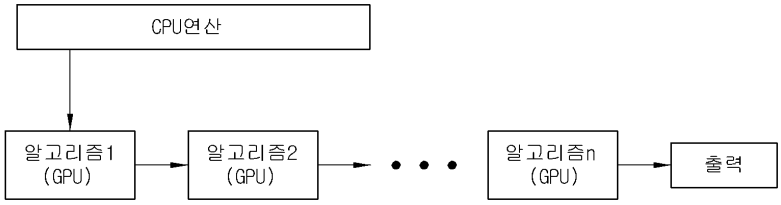
S400 : 출력단계

도면

도면1


CPU
시작
복호화
S100
데이터변환
S200
10a
버퍼
버퍼
10b
GPU
알고리즘
No
알고리즘 끝?
Yes
S300
출력
S400
끝

도면2


CPU연산
데이터변환
/전송
알고리즘1
(GPU)
CPU연산
데이터변환
/전송
알고리즘2
(GPU)
CPU연산
데이터변환
/전송
알고리즘n
(GPU)
출력

도면3