(19) 대한민국특허청(KR)
(12) 등록특허공보(B1)

(45) 공고일자 2008년02월28일
(11) 등록번호 10-0808087
(24) 등록일자 2008년02월21일

(51) Int. Cl.
G06T 15/00 (2006.01) *H04N 7/24* (2006.01)
(21) 출원번호 10-2007-0095571
(22) 출원일자 2007년09월19일
심사청구일자 2007년09월19일
(56) 선행기술조사문헌
KR1020000067384 A

(73) 특허권자
인하대학교 산학협력단
인천 남구 용현동 253 인하대학교
(72) 발명자
신병석
경기 안양시 동안구 갈산동 샘마을한양아파트 114동 402
계희원
서울 동작구 노량진2동 244-35
최이규
인천 남동구 간석1동 337-7번지
(74) 대리인
김건우

전체 청구항 수 : 총 2 항 심사관 : 장기정

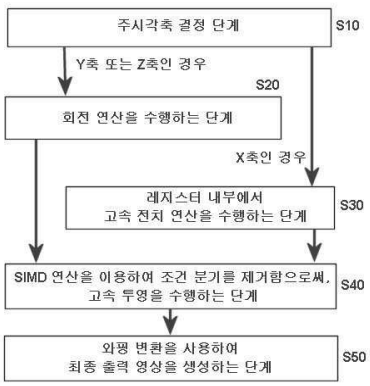
(54) 스트리밍 최대 휘소 투영 볼륨 렌더링 방법

(57) 요 약

본 발명은 스트리밍 최대 휘소 투영(Maximum Intensity Projection; MIP) 볼륨 렌더링 방법에 관한 것으로서, 본 발명에 스트리밍 MIP 볼륨 렌더링 방법은, (1) 가시화하고자 하는 3차원 볼륨 데이터의 관찰 시점을 변환 행렬(transformation matrix)을 이용하여 분류함으로써, 주시각축을 결정하는 제1 단계; (2) 상기 제1 단계에서 주시각축이 y축 또는 z축이라고 결정되는 경우, 스트리밍(streaming) 메모리 참조 조건을 만족시키는 회전 각도로 회전 연산을 수행하는 제2 단계; (3) 상기 제1 단계에서 주시각축이 x축이라고 결정되는 경우, 스트리밍 메모리 참조 조건을 만족하도록 레지스터 내부에서 고속 전치 연산을 수행하는 제3 단계; (4) 단일 명령 복수 데이터(Single Instruction Multiple Data; SIMD) 연산을 이용하여 조건 분기(conditional branch)를 제거함으로써, 상기 제2 단계 또는 상기 제3 단계에서의 연산 결과에 대하여 고속 투영을 수행하는 제4 단계; 및 (5) 상기 제4 단계에서 고속 투영된 영상에 대하여 왜곡을 바로 잡아 사용자가 관찰할 수 있는 최종 출력 영상을 생성하는 제5 단계를 포함하는 것을 그 구성상의 특징으로 한다.

본 발명의 스트리밍 MIP 볼륨 렌더링 방법에 따르면, 순차적인 메모리 접근으로 캐시 적중율이 높고 입출력 데이터 간의 1:1 연산을 보장하여 SIMD 연산을 사용할 수 있으므로, 빠른 이선형 보간과 조건 분기 제거가 가능하며, 그 결과 고속화된 렌더링이 가능하게 된다.

대표도 - 도6



특허청구의 범위

청구항 1

(1) 가시화하고자 하는 3차원 볼륨 데이터의 관찰 시점을 변환 행렬(transformation matrix)을 이용하여 분류함으로써, 주시각축을 결정하는 제1 단계;

(2) 상기 제1 단계에서 주시각축이 y축 또는 z축이라고 결정되는 경우, 스트리밍(streaming) 메모리 참조 조건을 만족시키는 회전 각도로 회전 연산을 수행하는 제2 단계;

(3) 상기 제1 단계에서 주시각축이 x축이라고 결정되는 경우, 스트리밍 메모리 참조 조건을 만족하도록 레지스터 내부에서 고속 전치 연산을 수행하는 제3 단계;

(4) 단일 명령 복수 데이터(Single Instruction Multiple Data; SIMD) 연산을 이용하여 조건 분기(conditional branch)를 제거함으로써, 상기 제2 단계 또는 상기 제3 단계에서의 연산 결과에 대하여 고속 투영을 수행하는 제4 단계; 및

(5) 상기 제4 단계에서 고속 투영된 영상에 대하여 왜곡을 바로 잡아 사용자가 관찰할 수 있는 최종 출력 영상을 생성하는 제5 단계

를 포함하는 스트리밍 최대 휘소 투영(Maximum Intensity Projection; MIP) 볼륨 렌더링 방법(Volume Rendering Method).

청구항 2

제1항에 있어서,

상기 제3 단계에서, 데이터를 4×4 크기의 타일 형태로 나눈 후, 상기 고속 전치 연산을 렌더링 연산 내부에서 수행하는 스트리밍 최대 휘소 투영 볼륨 렌더링 방법.

명 세 서

발명의 상세한 설명

기 술 분 야

<1> 본 발명은 볼륨 렌더링 방법에 관한 것으로, 특히 스트리밍 최대 휘소 투영(Maximum Intensity Projection; MIP) 볼륨 렌더링 방법에 관한 것이다.

배 경 기 술

<2> 최대 휘소 투영(Maximum Intensity Projection; MIP) 볼륨 렌더링 방법은 볼륨 데이터에 포함된 높은 밀도값을 가진 구조물을 가시화 하는 방법으로서, 관찰자 영상의 각 화소에 대하여 기여하는 복셀들을 찾고, 그 중 가장 높은 밀도값을 선택하여 출력하는 방법이다. MIP 볼륨 렌더링 방법은, 직접 볼륨 렌더링 방법과는 달리, 관찰자와 가까운 구조물이 먼 구조물을 가리는 효과가 생기지 않는다. 따라서 MIP 결과 영상에는 깊이 정보가 드러나지 않으며, 관찰자는 여러 각도에서 데이터를 관찰함으로써 깊이 정보를 추정하게 되는데, 잦은 관찰 방향 변경에 신속히 대처하려면 MIP 볼륨 렌더링 방법의 렌더링 속도가 충분히 빨라야 한다.

<3> MIP 볼륨 렌더링 방법에는, 분류 및 음영 연산이 없으며, 누적 연산도 알파-블렌딩(alpha-blending)이 아닌 최대값의 선택 연산이다. 따라서 하나의 샘플에 대한 누적 연산은 직접 볼륨 렌더링에 비하여 단순하다. 그러나 MIP 볼륨 렌더링 방법은 직접 볼륨 렌더링 방법에서 사용할 수 있는 빠른 광선 종료나 빈공간 탐색과 같은 최적화 기법을 적용할 수 없기 때문에 고속 렌더링을 실현하기가 쉽지 않다.

<4> 도 1 내지 도 2에 도시된 코드를 통하여 MIP 볼륨 렌더링 방법의 주된 연산을 알아보기로 한다. 도 1은 객체 순서 기반의 MIP 볼륨 렌더링 방법에 대한 가상 코드를 나타내는 도면으로서, 도 1에 따르면, 각각의 샘플에 대하여 연관된 픽셀 좌표를 얻고, 픽셀 좌표에 해당하는 값과 현재까지 누적된 픽셀 값과 비교하여 큰 값으로 대치한다. 도 2는 영상 순서 기반의 MIP 볼륨 렌더링 방법에 대한 가상 코드를 나타내는 도면으로서, 도 2에 따르면, 영상을 구성하는 각각의 픽셀에 대하여 연관된 볼륨 샘플 좌표를 얻고, 각 샘플 좌표로부터 샘플 값을 얻어 최대값과 비교하여 누적한 뒤, 최종적인 최대값을 픽셀 값으로 삼는다. 도 1 및 도 2로부터, 객체 순서 기

반의 MIP 볼륨 렌더링 방법 및 영상 순서 기반의 MIP 볼륨 렌더링 방법 모두, 매 샘플에 대하여 보간(interpolation)과 크기 비교 계산이 필요한 것을 확인할 수 있다.

<5> MIP 볼륨 렌더링 방법은, 각 샘플 지점에서 보간을 통하여 얻어진 샘플 값을 영상에 누적된 값과 비교하여 최대값을 취하는 방식으로 진행된다. 따라서 앞서 도 1 및 도 2와 관련하여 설명한 바와 같이, 보간과 비교 연산이 가장 빈번하게 수행된다. MIP 볼륨 렌더링 방법에서는, 보간과 비교 연산이 많은 수의 샘플과 픽셀에 대하여 동일하게 수행되는 특징이 있다. 보간의 가중치가 같다면 동일한 식에 의하여 데이터만 변경하여 연산이 수행되며, 여러 샘플 값 중에 최대값을 선택하는 것도 마찬가지이다. 이러한 연산을 효율적으로 수행하기 위하여, 대부분의 중앙처리장치(CPU)는 단일 명령어 다중 데이터(Single Instruction Multiple Data; SIMD) 명령어 세트를 제공하고 있다.

<6> 도 3은 SIMD의 실행 모델을 나타내는 도면이다. 도 3에 도시된 바와 같이, SIMD 명령어에 대해서는, 여러 데이터에 대하여 같은 연산이 병렬적으로 수행되고 수행된 결과 또한 병렬적으로 저장된다. 즉, 복수의 데이터에 대하여 같은 연산을 동시에 실행할 수 있으므로, 연산 속도가 향상된다. 그러나 입력 데이터와 출력 데이터가 중앙처리장치의 레지스터(register) 크기와 일치해야 하므로, 데이터들이 연속적인 물리 공간에 위치하고 있는 경우에만 효율적인 연산이 가능하다. 물리적 메모리 공간에서 연속적으로 위치하지 않은 데이터의 경우, 메모리의 정보를 레지스터로 복사해야 하는 오버헤드가 발생하여 성능이 저하된다. 또한, SIMD 명령어 세트에서 지원하는 연산은 병렬적인 연산으로 수평적 연산은 존재하지 않거나 존재한다고 해도 속도가 느리다. SIMD 명령어 세트를 활용하려면 입출력 데이터의 메모리 참조 방식을 순차적으로 제한할 필요가 있다.

<7> (1) 빠른 이선형 보간(fast bilinear interpolation)

<8> 본 발명은, 고속 볼륨 렌더링 알고리즘인 쉬어-왑 분해 렌더링 알고리즘을 기반으로 하고 있으므로, 보간 방식으로서는, 선형 보간의 조합으로 얻을 수 있는 이선형 보간(bilinear interpolation)이 사용된다. 도 4는 본 발명에서 사용될 이선형 보간을 나타내는 도면이다. 도 4에 따르면, 이선형 보간은 선형 보간을 세 번 수행하여 얻을 수 있는데, 가로 방향의 선형 보간을 먼저 수행하여 P_{01}과 P_{23}을 먼저 구한 후 세로 방향의 선형 보간을 통하여 P를 얻을 수 있다. 마찬가지로, 세로 방향의 선형 보간을 먼저 수행하여 P_{02}와 P_{13}을 구한 후 가로 방향의 선형 보간을 통하여 P를 구할 수도 있다. 두 결과는 동일하므로 경우에 따라 효율적인 방법을 선택해야 한다. 쉬어-왑 분해 볼륨 렌더링과 같이 보간의 가중치가 각 픽셀에 대하여 일정한 경우, 복셀 스트림에 대하여 고정된 가중치를 곱하여 선형 보간을 가속화 할 수 있다.

<9> (2) 조건 분기 제거(eliminating conditional branches)

<10> 최대값을 결정하기 위한 비교는 조건 분기를 통한 대입이다. 여러 자료 중 최대값을 찾는 방법은 이미 자료가 정렬(sorting)되어 있지 않은 이상 자료의 개수만큼(O(n)) 반복된 비교 연산이 요구된다. 일반적으로 비교는 조건 분기를 통하여 이루어지는데, 분기의 결과를 얻을 때까지 중앙처리장치의 파이프라인이 멈추게 되므로 조건 분기는 오랜 시간이 걸리는 작업이라고 할 수 있다. 비록 대부분의 중앙처리장치는 분기 예측을 하고 있지만, 2-비트 예측과 같은 기계적 방법의 분기 예측을 사용하고 있으므로, 자료의 분포를 추정하기 어려운 렌더링 시의 분기 예측 적중률은 일반적인 프로세스 제어에서의 분기 예측 적중률에 비하여 낮다. 게다가 최근 중앙처리장치의 주파수가 높아지고 그에 따라 파이프라인이 깊어지고 있는 추세이므로, 조건 분기로 인하여 발생하는 오버헤드 문제는 더욱 심각하다. 1996년 등장한 MMX(MultiMedia eXtension) 명령어 세트는 일정 부분 이러한 문제를 해결할 수 있는 방법을 제공하였고, 이후 등장한 SSE(Streaming SIMD Extension) 명령어 세트는 두 데이터 스트림에 대하여 최대값 스트림을 반환하는 방법을 제공하여 쉽게 조건 분기를 제거할 수 있다. 도 5는 MMX 명령어 세트 및 SSE 명령어 세트를 이용하여 2개의 배열 7, 5, 3, 1과 2, 4, 6, 8을 비교하여 큰 값을 얻는 과정을 나타내는 도면이다. 도 5a에 도시된 바와 같이, MMX 명령어 세트를 사용할 경우에는, 큰 값에 대한 비트 마스크를 설정할 수 있으며, 이진 비트 연산인 AND와 OR를 사용하여 최대값을 얻을 수 있다. 이에 비하여, 도 5b에 도시된 바와 같이, SSE 명령어 세트에는 직접 최대값을 얻는 명령어가 존재한다. SSE 명령어 세트가 더 효율적이긴 하지만 어느 쪽이건 조건 분기 없이 큰 값을 얻을 수 있으므로 이를 이용하면 빠른 렌더링이 가능하다.

<11> 앞서 살펴본 바와 같이, MIP 볼륨 렌더링 방법의 주요 연산인 보간과 조건 분기는 SIMD 연산을 통하여 가속화 될 수 있는데, 이를 위해서는 SIMD 연산은 병렬적으로 동작하므로 입력과 출력이 모두 스트림화될 필요가 있다. 따라서 쉬어-왑 볼륨 렌더링에 적합하나, 쉬어-왑 볼륨 렌더링은 볼륨 데이터를 3벌 중복해서 가지지 않는 경우, 메모리 참조 연관성 문제가 발생한다. 즉, 주시각축이 z축인 경우를 제외하면 물리적 메모리를 순차적으

로 읽을 수 없다. 적당한 회전 변환을 한다고 가정하여도 복셀과 픽셀 간의 1:1이 성립하지 않으므로 SIMD 연산에는 적합하지 않다는 문제점이 있다.

발명의 내용

해결 하고자하는 과제

<12> 본 발명은 상기와 같은 문제점 및 필요성에 대한 인식에서 비롯된 것으로서, 빠른 이선형 보간과 조건 분기 제거가 가능한 스트리밍 최대 휘소 투영(MIP) 볼륨 렌더링 방법을 제공하는 것을 그 목적으로 한다.

과제 해결수단

<13> 상기한 목적을 달성하기 위한 본 발명의 특징에 따른, 스트리밍 최대 휘소 투영(Maximum Intensity Projection; MIP) 볼륨 렌더링 방법은,

<14> (1) 가시화하고자 하는 3차원 볼륨 데이터의 관찰 시점을 변환 행렬(transformation matrix)을 이용하여 분류함으로써, 주시각축을 결정하는 제1 단계;

<15> (2) 상기 제1 단계에서 주시각축이 y축 또는 z축이라고 결정되는 경우, 스트리밍(streaming) 메모리 참조 조건을 만족시키는 회전 각도로 회전 연산을 수행하는 제2 단계;

<16> (3) 상기 제1 단계에서 주시각축이 x축이라고 결정되는 경우, 스트리밍 메모리 참조 조건을 만족하도록 레지스터 내부에서 고속 전치 연산을 수행하는 제3 단계;

<17> (4) 단일 명령 복수 데이터(Single Instruction Multiple Data; SIMD) 연산을 이용하여 조건 분기(conditional branch)를 제거함으로써, 상기 제2 단계 또는 상기 제3 단계에서의 연산 결과에 대하여 고속 투영을 수행하는 제4 단계; 및

<18> (5) 상기 제4 단계에서 고속 투영된 영상에 대하여 왜곡을 바로 잡아 사용자가 관찰할 수 있는 최종 출력 영상을 생성하는 제5 단계를 포함하는 것을 그 구성상의 특징으로 한다.

<19> 바람직하게는, 상기 제3 단계에서, 데이터를 4 × 4 크기의 타일 형태로 나눈 후, 상기 고속 전치 연산을 렌더링 연산 내부에서 수행한다.

효 과

<20> 본 발명에서 제안된 스트리밍 MIP 볼륨 렌더링 방법에 따르면, 순차적인 메모리 접근으로 캐시 적중율이 높고 입출력 데이터간의 1:1 연산을 보장하여 SIMD 연산을 사용할 수 있으므로, 빠른 이선형 보간과 조건 분기 제거가 가능하며, 그 결과 고속화된 렌더링이 가능하게 된다.

발명의 실시를 위한 구체적인 내용

<21> 이하에서는 첨부된 도면들을 참조하여, 본 발명에 따른 실시예에 대하여 상세하게 설명하기로 한다.

<22> 도 6은 본 발명의 일 실시예에 따른 스트리밍 MIP 볼륨 렌더링 방법을 나타내는 도면이다. 도 6에 도시된 바와 같이, 본 발명에 따른 스트리밍 MIP 볼륨 렌더링 방법은, 주시각축을 결정하는 단계(S10), 단계 S10에서 주시각축이 Y축 또는 Z축으로 결정되는 경우, 회전 연산을 수행하는 단계(S20), 단계 S10에서 주시각축이 X축으로 결정되는 경우, 레지스터 내부에서 고속 전치 연산을 수행하는 단계(S30), 단계 S20 또는 단계 S30에서의 연산 결과에 대하여 SIMD 연산을 이용하여 조건 분기를 제거함으로써, 고속 투영을 수행하는 단계(S40), 및 와핑 변환을 사용하여 최종 출력 영상을 생성하는 단계(S50)를 포함한다.

<23> 단계 S10에서는 주시각축을 결정하는데, 주시각축은 가시화하고자 하는 3차원 볼륨 데이터의 관찰 시점을 변환 행렬(transformation matrix)을 이용하여 분류함으로써 결정될 수 있다. 여기서, '주시각축'이란 시선 벡터와 가장 평행한 좌표축으로 정의되며, 시점을 변환 행렬을 이용하여 분류한다는 것은, 데이터를 관찰하는 방향인 '시선 벡터'가 물체 좌표계를 구성하는 x축, y축, z축 중 어떤 방향과 가장 평행한지(어느 축이 주시각축이 되는지) 판별하는 과정을 의미한다. 단계 S10에서 주시각축이 Y축 또는 Z축으로 결정되는 경우, 단계 S20에서는 스트리밍(streaming) 메모리 참조 조건을 만족시키는 회전 각도로 회전 연산을 수행하게 된다. 여기서, 스트리밍 메모리 참조 조건이란, 데이터가 물리적으로 인접해야 하며, 볼륨 스트림 데이터와 픽셀 스트림 데이터 간의 1:1 대응이 이루어지는 조건을 의미한다. 단계 S10에서 주시각축이 X축으로 결정되는 경우, 단계 S30에서는 스

트리밍 메모리 참조 조건을 만족하도록 레지스터 내부에서 고속 전치 연산을 수행하게 된다. 여기서, 스트리밍 메모리 참조 조건은 앞서 설명한 조건과 동일하다. 단계 S40에서는 상기 제2 단계 또는 상기 제3 단계에서의 연산 결과에 대하여, 단일 명령 복수 데이터(SIMD) 연산을 이용하여 조건 분기를 제거함으로써, 고속 투영을 수행하게 된다. 개인용 컴퓨터의 x86기반 SIMD 연산은 마스킹(masking) 연산을 포함하고 있기 때문에, SIMD 연산을 이용할 경우 조건 분기를 제거할 수 있다. 물론, 본 발명의 특징에 따르면, 앞서 언급한 바와 같이, SSE 명령어 세트의 두 데이터 스트림에 대하여 최대값 스트림을 반환하는 방법을 이용함으로써 쉽게 조건 분기를 제거하는 것도 가능하다. 마지막으로, 단계 S50에서는, 와핑 변환을 사용하여 최종 출력 영상을 생성하게 된다. 본 발명에서 사용하고 있는 '쉬어-왑 분해' 알고리즘은 '쉬어' 단계와 '왑(와핑)' 단계로 나뉘는데, 상기 '쉬어' 단계에서는 '주시각축'을 사용하여 가시화하기 때문에 영상이 왜곡되게 되며, '왑' 단계에서 '와핑 변환'을 통해 왜곡을 바로잡게 된다. 따라서, 상기 단계 S50은 상기 단계 S40에서 고속 투영된 영상에 대하여 왜곡을 바로 잡아 사용자가 관찰할 수 있는 최종 출력 영상을 생성하는 단계라고 할 수 있다. 특히, 단계 S30에서, 데이터를 4×4 크기의 타일 형태로 나눈 후, 이에 대하여 고속 전치 연산을 렌더링 연산 내부에서 수행할 수 있다.

<24> 도 7은 스트리밍 MIP 볼륨 렌더링을 위한 좌표계의 변환을 나타내는 도면이다. 도 7a, 7b, 7c는 쉬어-투영 변환 후의 좌표계를 나타내며, 각각에 적당한 변환을 하여 각각 7d, 7e, 7f로 변환한다. 변환의 결과, 볼륨 x축이 영상의 i축과 평행하게 된다. SIMD 연산은 입력과 출력의 개수가 같도록 요구하므로(도 5 참조), MIP 볼륨 렌더링 연산에서 볼륨 x축과 영상의 i축의 길이도 같아야 한다. 따라서 7f의 경우 회전이 아닌 대칭 연산이 사용되고 있다.

<25> 주시각축이 z축이나 y축인 경우 앞서 설명한 바와 같이, 쉬어-왑 분해 과정에서 회전 변환을 추가하여 메모리 참조를 볼륨과 영상 모두에서 순차적으로 할 수 있도록 진행할 수 있다. 이에 비하여, 주시각축이 x축인 경우, 볼륨의 각 x-y평면에 대하여 전치 연산을 적용하여 메모리의 배치 순서를 바꾼 후 i축과 평행한 x축 방향으로 렌더링한다. 임의 각도의 회전 변환이 없으므로 홀이 발생하거나 투영 연산이 복잡해지지 않으며, MIP는 렌더링 순서에 독립적이므로 스캔 라인 순서를 고려하지 않아도 된다. 단, 주시각축이 x축인 경우 전치 연산을 수행하는 오버헤드가 발생하는 단점이 있다. 도 8, 도 9 및 도 10은 각각 주시각축이 z축, y축, x축인 경우의 본 발명의 일 실시예에 따른 가상 코드를 나타내는 도면이다. 도 8에 따르면, 주시각축이 z축인 경우, 각 스캔 라인에 대하여 볼륨 스트림을 얻고, 영상 스트림과 비교하여 최대값을 저장한다. 도 9에 따르면, 주시각축이 y축인 경우, 주시각축인 y를 가장 상위에서 반복하는 점에서만 차이가 있을 뿐, 도 8에 도시된 주시각축이 z축인 경우와 비슷한 형태가 된다. 도 10에 따르면, 주시각축이 x축인 경우, 각 슬라이스에 대하여 먼저 전치 변환을 적용한 후, 일반적인 알고리즘을 적용한다. 각 주시각축에 대한 렌더링 알고리즘에서, 영상과 볼륨 간의 좌표 변환은 매 스캔 라인마다 1회만 일어나므로, 좌표 변환에 대한 오버헤드는 무시할 수 있다. 따라서 알고리즘의 성능은 주로 보간(Interpolation) 연산과 최대값(Max) 연산에서 결정된다. 본 발명에서 제안된 방법은 순차적인 메모리 접근으로 캐시 적중율이 높으며, 입출력 데이터 간의 1:1 연산을 보장하여 SIMD 연산을 사용할 수 있으므로 고속화된 렌더링이 가능하게 된다.

<26> 본 발명에 사용할 수 있는 효율적인 캐시 메모리 적용 방법에 대하여 좀 더 살펴보기로 한다. 대용량 데이터를 렌더링할 경우 전치 연산은 속도 저하를 유발하므로, 본 발명에서는 전치 연산을 렌더링 연산 내부에 삽입하여 성능을 보다 향상시키는 방법을 제안한다. 이와 같은 방법은 루프 차단(loop blocking) 방법으로 알려져 있다. 일반적인 루프 차단 방법은 데이터를 캐시 메모리에 적재할 수 있는 작은 크기로 나누어서 연산을 수행하는 것인데 비하여, 본 발명에서 제안한 방법은 그보다 더 작은 4×4 크기의 타일 형태로 데이터를 나누어서 레지스터 단위에서 전치 연산을 수행한다. 도 11은 본 발명에서 제안한 변형된 루프 차단 방법의 소스 코드를 나타내는 도면이다. 도 11에 도시된 방법을 채택할 경우, SIMD 연산이 레지스터 내부에서 수행되므로 메모리를 참조하는데 필요한 오버헤드를 없앨 수 있다. 특히, 일반적인 CPU에는 8개 이상의 SIMD 레지스터가 존재한다는 점을 고려하면, 도 11의 알고리즘이 시간이 많이 필요한 메모리 참조 없이 CPU 내부에서 수행 가능하다는 점은 더욱 분명하다. 이를 이용한 수정된 알고리즘을 도 12에 제시하고 있다. 도 12는 주시각축이 x축인 경우의 수정된 알고리즘을 나타내는 도면으로서, 전치 연산이 렌더링 내부에 포함되어 있어, 객체와 영상 간의 좌표 변환은 각 스캔 라인마다 발생하므로 성능 저하를 최소화할 수 있다. 도 12에서 제안된 알고리즘은 메모리 참조 비용을 줄여 성능 향상에 기여한다.

<27> 다음으로, 본 발명에서 제안한 방법의 성능을 실험을 통하여 분석한 결과를 살펴보기로 한다. 실험은 2.4 GHz 펜티엄 4 프로세서와 1GB의 주메모리를 가진 PC에서 수행되었다. 프로세서의 1차 캐시의 크기는 8 KB이고, 2차 캐시의 크기는 512 KB이며, 캐시 블록의 크기는 64 바이트이다. 소프트웨어 환경은 Windows XP를 기반으로

Visual C++ 컴파일러와 DirectX 9.0을 사용하였다. 표 1은 본 실험에서 사용된 데이터의 종류와 크기를 나타내고 있는데, 각 데이터 집합은 CT 영상으로 각 복셀 당 2 바이트를 차지한다.

표 1

<28>

데이터 집합	크기	MB
다리(Leg)	256× 256× 110	13.75
두뇌(Brain)	512× 512× 185	92.5
대동맥(Aorta)	512× 512× 310	155
몸(Body)	512× 512× 512	256

<29> 본 발명에서 제안된 방법의 성능 향상 효과를 보이기 위하여 표 2를 제시한다. 표 2에서는, 본 발명에서 제안한 방법의 성능을, 광선 추적법(ray casting), 쉬어-왑(shear-warp) 렌더링 방법의 성능과 비교하여 제시하고 있다. 표 2의 결과에서 단위는 초(s)이고, 관찰 방향을 바꿔가며 평균적인 렌더링 시간을 측정한 것이며, 광선 추적법은 고화질 렌더링이기 때문에 시간을 단축하기 위해서 화면의 가로 세로 해상도를 각각 반으로 줄인 결과이다. 표 2로부터, 본 발명에서 제안한 스트리밍 렌더링은 기존의 광선 추적법이나 쉬어-왑 렌더링에 비하여 현저히 향상된 결과를 보이며, 특히 본 발명에서 제안한 방법들을 함께 사용하는 경우 더욱 큰 속도 향상을 얻을 수 있음을 확인할 수 있다.

표 2

<30>

방법	다리	두뇌	대동맥	몸
광선 추적법	0.578	1.262	2.559	4.583
쉬어-왑	0.306	2.013	3.745	4.548
본 발명에서 제안한 스트리밍 방법만 사용	0.115	0.327	0.523	0.742
본 발명에서 제안한 방법들 함께 사용	0.095	0.205	0.303	0.455

<31> SIMD 연산을 사용한 분기 제거의 효과를 알아보기 위하여 다음 표 3을 제시한다. 표 3은 몸(Body) 데이터 집합을 이용하여 y축을 기준으로 관찰한 결과이다. 표 3으로부터, 본 발명에서 제안한 방법이 두드러진 성능 향상을 보이고 있음을 확인할 수 있다.

표 3

<32>

방법	시간
분기 제거 없음	0.650
본 발명에서 제안된 방법 (분기 제거 있음)	0.443

<33> 다음으로, 캐시 메모리를 효율적으로 사용하는 전치 연산 방법의 실험 결과가 표 4에 제시되어 있다. 표 4는 주시각축이 x축인 경우 몸(Body) 데이터 집합을 이용한 실험 결과를 나타내고 있다. 기존 쉬어-왑 렌더링 방법은 매우 느리기 때문에, 캐시 효율을 향상시키는 Bricked 볼륨 레이아웃(Bricked volume layout) 기법을 사용하여도, 본 발명에서 제안한 방법 중 스트리밍 방법만을 사용한 경우보다 느리다. 더욱이 스트리밍 방법을 캐시 효율 방법과 함께 사용할 경우, 더욱 높은 성능을 나타낸다.

표 4

<34>

방법	주시각축 = x축 (s)
본 발명에서 제안한 스트리밍 방법	1.047
스트리밍 방법 + 캐시 효율 방법	0.477
쉬어-왑 렌더링 방법	8.406

쉬어-왑 방법 + Bricked 볼륨 레이아웃(4^3)	2.959
쉬어-왑 방법 + Bricked 볼륨 레이아웃(8^3)	1.594
쉬어-왑 방법 + Bricked 볼륨 레이아웃(16^3)	1.374
쉬어-왑 방법 + Bricked 볼륨 레이아웃(32^3)	2.628

<35> 마지막으로, 도 13은 본 발명에서 제안한 방법에 의하여 얻어진 각 데이터 집합에 대한 렌더링 결과 영상을 나타내는 도면이다. 도 13으로부터, 본 발명에서 제안한 방법에 의해 얻어진 렌더링 결과 영상이 기존의 쉬어-왑 렌더링 결과 영상과 동등한 화질의 영상을 출력함을 확인할 수 있다.

<36> 이상 설명한 본 발명은 본 발명이 속한 기술분야에서 통상의 지식을 가진 자에 의하여 다양한 변형이나 응용이 가능하며, 본 발명에 따른 기술적 사상의 범위는 아래의 특허청구범위에 의하여 정해져야 할 것이다.

산업이용 가능성

<37> MIP 볼륨 렌더링 방법은 의료영상 처리에 널리 사용되고 있는 방법으로 상용 패키지로 판매되는 기능이다. 이 방법은 주로 조영된 혈관의 구조를 나타내어, 혈관의 협착 등을 진단하는데 사용된다. 본 발명에서 제안된 방법인 MIP 볼륨 렌더링의 고속화를 통하여, 기존의 방법에 비해 빠른 시간에 진단을 완료할 수 있다. 따라서 본 발명에서 제안된 방법의 구현은 기존 제품을 대치 가능하므로 직접 상품화가 용이할 것으로 기대된다.

도면의 간단한 설명

<38> 도 1은 객체 순서 기반의 MIP 볼륨 렌더링 방법에 대한 가상 코드를 나타내는 도면.

<39> 도 2는 영상 순서 기반의 MIP 볼륨 렌더링 방법에 대한 가상 코드를 나타내는 도면.

<40> 도 3은 단일 명령어 다중 데이터(SIMD)의 실행 모델을 나타내는 도면.

<41> 도 4는 본 발명에서 사용될 이선형 보간(bilinear interpolation)을 나타내는 도면.

<42> 도 5는 MMX 명령어 세트 및 SSE 명령어 세트를 이용하여 2개의 배열 7, 5, 3, 1과 2, 4, 6, 8을 비교하여 큰 값을 얻는 과정을 나타내는 도면.

<43> 도 6은 본 발명의 일 실시예에 따른 스트리밍 MIP 볼륨 렌더링 방법을 나타내는 도면.

<44> 도 7은 스트리밍 MIP 볼륨 렌더링을 위한 좌표계의 변환을 나타내는 도면.

<45> 도 8, 도 9 및 도 10은, 각각 주시각축이 z축, y축, x축인 경우의 본 발명의 일 실시예에 따른 가상 코드를 나타내는 도면.

<46> 도 11은 본 발명에서 제안한 변형된 루프 차단 방법의 소스 코드를 나타내는 도면.

<47> 도 12는 주시각축이 x축인 경우의 수정된 알고리즘을 나타내는 도면.

<48> 도 13은 본 발명에서 제안한 방법에 의하여 얻어진 각 데이터 집합에 대한 렌더링 결과 영상을 나타내는 도면.

<49> <도면 중 주요 부분에 대한 부호의 설명>

<50> S10: 주시각축을 결정하는 단계

<51> S20: 회전 연산을 수행하는 단계

<52> S30: 레지스터 내부에서 고속 전치 연산을 수행하는 단계

<53> S40: 고속 투영을 수행하는 단계

<54> S50: 와핑 변환을 사용하여 최종 출력 영상을 생성하는 단계

도면

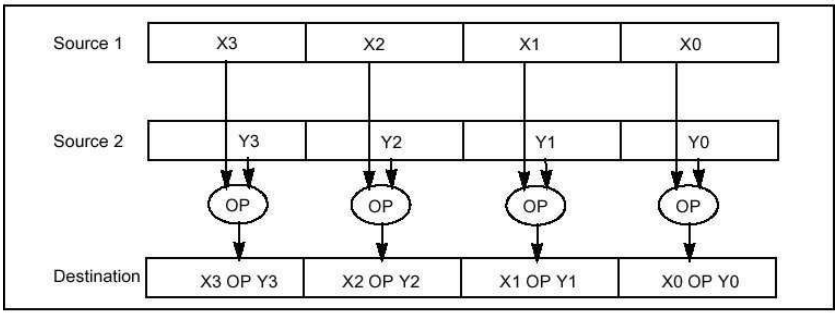
도면1

```
For each sample
    CurrentValue ← Get_Sample_Value()
    Position ← Get_pixel_position()
    MaxValue ← Max(Get_pixel_value(Position), CurrentValue)
    Set_pixel_value(MaxValue)
```

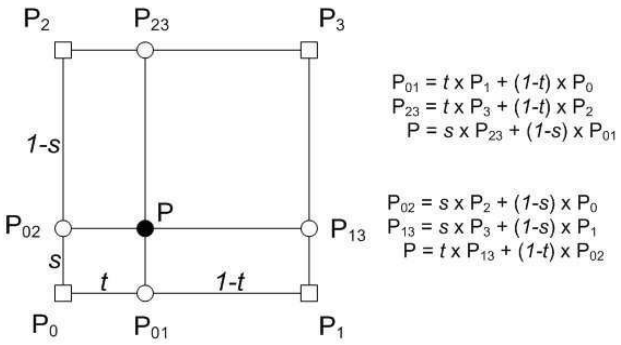
도면2

```
For each pixel
    MaxValue ← Initial value
    For each sample on pixel
        Position ← Get_sample_position()
        MaxValue ← Max(Get_sample_value(Position), MaxValue)
        Set_pixel_value(MaxValue)
```

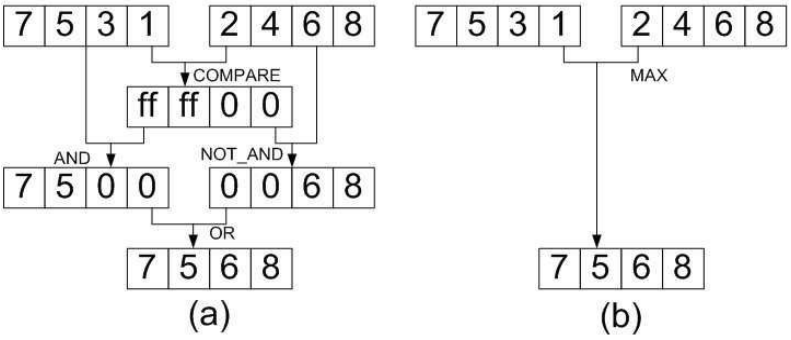
도면3



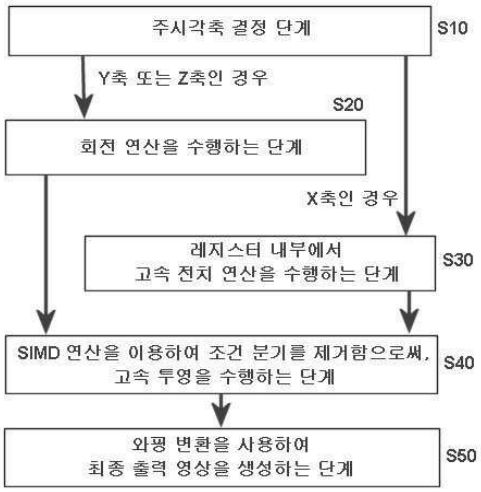
도면4



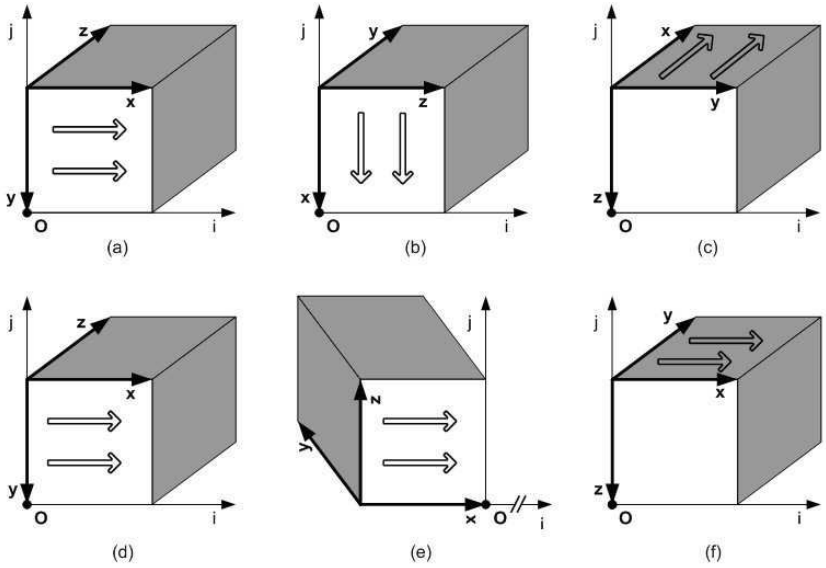
도면5


7 5 3 1
2 4 6 8
COMPARE
ff ff 0 0
AND
NOT_AND
7 5 0 0
0 0 6 8
OR
7 5 6 8
(a)
7 5 3 1
2 4 6 8
MAX
7 5 6 8
(b)

도면6


주시각축 결정 단계
S10
Y축 또는 Z축인 경우
S20
회전 연산을 수행하는 단계
X축인 경우
레지스터 내부에서
고속 전치 연산을 수행하는 단계
S30
SIMD 연산을 이용하여 조건 분기를 제거함으로써,
고속 투영을 수행하는 단계
S40
와핑 변환을 사용하여
최종 출력 영상을 생성하는 단계
S50

도면7



도면8

```
For each z
  For each y
    volume_stream ← Get slice buffer(y,z)
    sample_stream ← Interpolation(volume_stream)
    image_stream ← Get image buffer(y,z)
    image_stream ← Max(image_stream, sample_stream)
```

도면9

```
For each y
  For each z
    volume_stream ← Get slice buffer(y,z)
    sample_stream ← Interpolation(volume_stream)
    image_stream ← Get image buffer(y,z)
    image_stream ← Max(image_stream,sample_stream)
```

도면10

```
For each z
  t_stream ← Transpose slice(z)  // transposed stream
  For each y
    volume_stream ← Get slice buffer(t_stream , y)
    sample_stream ← Interpolation(volume_stream)
    image_stream ← Get image buffer(y,z)
    image_stream ← Max(image_stream, sample_stream)
```

도면11

```
Transpose(row0,row1,row2,row3)
   (
    __m64 tmp3, tmp2, tmp1, tmp0;
    tmp0     = _mm_unpackhi_pi16(row0,row1);
    tmp2     = _mm_unpacklo_pi16(row0,row1);
    tmp1     = _mm_unpackhi_pi16(row2,row3);
    tmp3     = _mm_unpacklo_pi16(row2,row3);
    (row0)   = _mm_unpacklo_pi32(tmp2,tmp3);
    (row1)   = _mm_unpackhi_pi32(tmp2,tmp3);
    (row2)   = _mm_unpacklo_pi32(tmp0,tmp1);
    (row3)   = _mm_unpackhi_pi32(tmp0,tmp1);
    )
```

도면12

```
For each z
   For each y mod 4 = 0
      volume_stream[y..y+3] ← Get slice buffer(y..y+3, z)
      For each x mod 4 = 0
         Transpose(volume_stream[y..y+3][x..x+3])
         Sample_stream ← Interpolation(volume_stream)
      image_stream ← Get image buffer(y,z)
  image_stream ← Max(image_stream, Sample_stream
```

도면13