(19) 대한민국특허청(KR)
(12) 공개특허공보(A)

(51) 。Int. Cl. [7]
G06T 15/50

(11) 공개번호 특2001-0113669
(43) 공개일자 2001년12월28일

(21) 출원번호 10-2001-7009655
(22) 출원일자 2001년07월31일
번역문 제출일자 2001년07월31일
(86) 국제출원번호 PCT/US2000/04148
(86) 국제출원출원일자 2000년02월17일
(87) 국제공개번호 WO 2000/49577
(87) 국제공개일자 2000년08월24일

(81) 지정국 국내특허 : 알바니아, 아르메니아, 오스트리아, 오스트레일리아, 아제르바이잔, 보스니아-헤르체고비나, 바베이도스, 불가리아, 브라질, 벨라루스, 캐나다, 스위스, 중국, 쿠바, 체코, 독일, 덴마크, 에스토니아, 스페인, 핀랜드, 영국, 그루지야, 헝가리, 이스라엘, 아이슬란드, 일본, 케냐, 키르기즈, 북한, 대한민국, 카자흐스탄, 세인트루시아, 스리랑카, 라이베리아, 레소토, 리투아니아, 룩셈부르크, 라트비아, 몰도바, 마다가스카르, 마케도니아, 몽고, 말라위, 멕시코, 노르웨이, 뉴질랜드, 슬로베니아, 슬로바키아, 타지키스탄, 투르크메니스탄, 터어키, 트리니다드토바고, 우크라이나, 우간다, 우즈베키스탄, 베트남, 폴란드, 포르투칼, 루마니아, 러시아, 수단, 스웨덴, 싱가포르, 아랍에미리트, 코스타리카, 도미니카연방, 모로코, 탄자니아, 남아프리카, 그레나다, 가나, 감비아, 크로아티아, 인도네시아, 인도, 시에라리온, 유고슬라비아, 짐바브웨,
AP ARIPO특허: 케냐, 레소토, 말라위, 수단, 스와질랜드, 우간다, 시에라리온, 가나, 감비아, 짐바브웨, 탄자니아,
EA 유라시아특허: 아르메니아, 아제르바이잔, 벨라루스, 키르기즈, 카자흐스탄, 몰도바, 러시아, 타지키스탄, 투르크메니스탄,
EP 유럽특허: 오스트리아, 벨기에, 스위스, 독일, 덴마크, 스페인, 프랑스, 영국, 그리스, 아일랜드, 이탈리아, 룩셈부르크, 모나코, 네덜란드, 포르투칼, 스웨덴, 핀랜드, 사이프러스,
OA OAPI특허: 부르키나파소, 베넹, 중앙아프리카, 콩고, 코트디브와르, 카메룬, 가봉, 기네, 말리, 모리타니, 니제르, 세네갈, 차드, 토고, 기네비쏘,

(30) 우선권주장 09/251,844 1999년02월17일 미국(US)
09/472,940 1999년12월27일 미국(US)

(71) 출원인 선 마이크로시스템즈, 인코포레이티드
케네쓰 올센
미합중국 캘리포니아 94303 팔로 알토 샌 안토니오 로드 901

(72) 발명자 디링,미카엘,에프.
미국,캘리포니아94024,로스알토스,쿠에스타드라이브657.
니글레,나다니엘,데이비드
미국,캘리포니아94588,플리잰톤,오크크리크코트7756.
넬슨,스콧,알.
미국,캘리포니아94588,플리잰톤클로브우드레인4429.

(74) 대리인 특허법인 원전

심사청구 : 없음

(54) 병렬적 샘플-픽셀 연산을 수행하도록 구성된 그래픽 시스템

요약

샘플버퍼(162)와 복수의 병렬의 샘플-픽셀 연산유닛(170)을 활용하도록 구성된 컴퓨터 그래픽 시스템으로서, 상기 샘플-픽셀 연산유닛은 샘플버퍼의 상이한 부분들을 병렬로 액세스하도록 구성된다. 본 그래픽 시스템은 프로세서(352), 샘플버퍼(162), 및 복수의 샘플-픽셀 연산유닛(170,360)을 포함할 수 있다. 상기 프로세서는 한 세트의 그래픽 데이터를 수신하고, 상기 그래픽 데이터에 기초하여 샘플들을 렌더링하도록 구성된다. 상기 샘플버퍼는 상기 샘플들을 저장하도록 구성된다. 상기 샘플-픽셀 연산유닛은 샘플버퍼로부터의 샘플들을 읽고 여과하여 출력픽셀을 생성한다. 상기 샘플-픽셀 연산유닛 각각은 이미지의 상이한 영역에 대응하여 픽셀을 생성하도록 구성된다. 상기 영역은 이미지의 수직적 또는 수평적 줄 또는 이미지의 직사각형 부분일 수 있다. 상기 영역들은 시각적 수차(aberration)를 방지하기 위해 오버랩(overlap)될 수 있다.

대표도

도 7

명세서

기술분야

본 발명은 일반적으로 컴퓨터 그래픽 분야에 관한 것으로서, 더욱 상세하게는 고성능 그래픽 시스템에 관한 것이다.

배경기술

컴퓨터 시스템은 컴퓨터 스크린 또는 디스플레이 장치에 시각적 출력을 하기 위하여 전형적으로 그래픽 시스템에 의존한다. 초기의 그래픽 시스템은 프로세서가 출력으로 만들어내는 것을 취하고 그것을 스크린에 나타내는 것만 책임졌다. 본질적으로, 그것들은 단순한 중계기 또는 인터페이스로서 작동했다. 그러나, 현대의 그래픽 시스템은 대단한 처리능력을 가진 그래픽 프로세서를 포함한다. 그것은 단순한 중계기라기보다는 보조프로세서로서 작동한다. 이러한 변화는 최근 디스플레이 장치로 보내어지는 데이터의 복잡성과 양의 증가에 기인한다. 예를 들어, 현대의 컴퓨터 디스플레이는 더 많은 픽셀과 더 훌륭한 색 깊이를 가지고 있으며 초기 모델보다 더 빠른 리프레쉬(refresh) 속도로 더욱 복잡한 화상을 표현할 수 있다. 마찬가지로, 표현되는 이미지는 더욱 복잡하며 앤티-앨리어스(anti-aliasing) 및 택스쳐 매핑(texture mapping)과 같은 진보된 기술을 포함할 수 있다.

그 결과, 그래픽 시스템에서의 상당한 처리능력이 없으면, CPU는 그래픽 계산을 수행하는데 엄청난 시간을 소비하게 될 것이다. 이는 컴퓨터 시스템으로부터 프로그램 실행과 관련한 다른 작업을 실행하는 데 필요한 처리능력을 빼앗아 전체적인 시스템 성능을 크게 감소시킨다. 그러나, 강력한 그래픽 시스템에 있어서는, CPU가 스크린 상에 박스를 그리도록 지시 받으면, CPU는 각 픽셀의 위치 및 컬러를 계산하지 않아도 된다. 그 대신에, 그 CPU는 " 이 좌표에 박스를 그려라" 고 비디오 카드에 요청을 할 수 있다. 그러면, 그래픽 시스템이 박스를 그려서, 프로세서가 자유로이 다른 작업을 수행할 수 있도록 한다.

일반적으로, 컴퓨터 내의 그래픽 시스템(단순히 그래픽 시스템으로도 호칭됨)은 성능 레벨을 증폭시키는 자체 프로세서를 포함하는 형태의 비디오 어댑터이다. 이러한 프로세서들은 그래픽 변환을 계산하기 위해 특수화되어 있으므로, 컴퓨터 시스템에서 사용되는 일반적인 목적의 CPU보다도 양호한 결과를 성취하는 경향이 있다. 더욱이, 그것들은 그래픽 시스템이 그래픽 계산을 다루는 동안 CPU가 자유로이 다른 명령들을 수행할 수 있도록 해준다. 그래픽 애플리케이션의 인기와 특히 멀티미디어 애플리케이션은 고성능 그래픽 시스템을 공통된 특성의 컴퓨터 시스템으로 만들어 왔다. 대부분의 컴퓨터 제조자는 이제 그들의 시스템에 고성능 그래픽 시스템을 포함시킨다.

그래픽 시스템은 제한된 기능만을 전형적으로 수행하기 때문에, 그것들은 주문대로 맞추어질 수 있고 따라서 컴퓨터의 일반적 목적의 중앙 프로세서보다 그래픽에 있어서 훨씬 효율적일 수 있다. 초기의 그래픽 시스템이 2차원(2D) 그래픽을 수행하는데 국한되었던 반면에, 그들의 기능성은 3차원(3D) 와이어-프레임 그래픽, 3D 입체를 지원할 정도로 증가하였고, 이제는 진보된 세이딩(shading), 포깅(fogging), 알파-혼합(alpha-blending), 그리고 거울같은 하이라이트(specular highlighting) 등의 특별 효과 및 텍스쳐(texture)를 가진 3차원(3D) 그래픽을 위한 지원을 포함한다.

3D 그래픽 시스템의 처리능력은 가파른 속도로 진보해 왔다. 몇 년 전에, 간단한 물체의 세이딩처리된 이미지는 초당 수 프레임으로만 렌더링(rendering)할 수 있었던 반면에, 오늘날의 시스템은 60Hz 또는 그 이상의 주파수에서 복잡한 물체를 렌더링하는 것을 지원한다. 이러한 속도의 증가에 있어서, 그리 멀지 않은 미래에, 그래픽 시스템은 문언적으로, 단일의 인간의 시각 시스템이 인식할 수 있는 것 보다 더욱 많은 픽셀을 만들 수 있을 것이다. 이러한 특별한 능력은 다중-뷰어 환경에서 이용될 수 있는 반면에, 보다 일반적인 기초적인 싱글-뷰어 환경에서 낭비될 수도 있다. 따라서, 인간의 해상도 조직의 다양한 성질과 조화할 능력(즉, 필요한 품질 또는 가장 인지 잘되는 품질을 가지는 능력)이 있는 그래픽 시스템이 요망된다.

픽셀의 수가 그래픽 시스템 성능을 결정하는 중요한 요소이지만, 그만큼 중요한 다른 요소로 이미지의 품질이 있다. 예를 들어, 이미지의 가장자리가 너무 날카롭거나 들쭉날쭉(" 앨리어스(aliased)하다" 라고도 한다)하면 높은 픽셀 밀도의 이미지라도 여전히 비현실적으로 보일 수 있다. 이러한 문제를 극복하는 잘 알려진 기술로는 앤티-앨리어싱(anti-aliasing)이 있다. 앤티-앨리어싱은 그래픽 요소의 경계를 따라 픽셀에 세이딩(shading)을 줌에 의해 물체의 가장자리를 부드럽게 하는 것을 포함한다. 보다 상세하게는, 앤티-앨리어싱은 고주파수 성분이 시각적 아티팩트(artifacts)를 교란하기 전에, 이미지로부터 고주파수 성분을 제거하는 것을 수반한다. 예를 들어, 앤티-앨리어싱은 특정 픽셀이 중간값을 가지도록 하여 (예컨대, 어두운 배경과 겹쳐진 밝은 물체의 그림자의 둘레에서) 이미지내의 높은 콘트라스트(contrast)의 가장자리(edge)를 부드럽거나 매끄럽게 할 수 있다.

컴퓨터 이미지의 현실성을 증가시키는데 사용되는 또 다른 시각효과로는 알파 혼합(alpha blending)이 있다. 알파 혼합은 물체의 투명도를 조절하는 기술로서, 물이나 유리 같은 투명한 표면의 현실적인 렌더링이 가능하도록 한다. 현실성을 더욱 증가시키기 위해 사용되는 또 다른 효과로는 포깅(fogging)이 있다. 포깅은 물체가 보는 사람으로부터 멀어져 가는 것처럼 흐릿하게 한다. 단순한 포깅(fogging)은 알파 혼합의 하나의 특별한 경우로서, 거리에 따라 알파도(degree of alpha)가 변화하여 물체가 관찰자로부터 멀어져 감에 따라 안개속으로 사라지게 보이도록 한다. 이 단순한 포깅은 " 뎁쓰 큐잉(depth cueing)" 또는 분위기 희석(atmospheric attenuation)으로도 불리우며, 즉 물체의 콘트라스트(contrast)를 낮추어 물체가 후퇴함에 따라 눈에 덜 띄게 보이도록 하는 것이다. 더욱 복잡한 유형의 포깅은, 투명 레벨과 관찰자로부터의 거리간의 더욱 복잡한 관계를 제공하기 위해, 간단한 선형 함수를 넘어선다. 본 기술분야의 소프트웨어 시스템의 현재 상태는, 밑에 깔려있는 안개에 향상된 현실성을 제공하기 위한 분위기의 모델을 활용함으로써

더욱 진보하고 있다.

위에서 열거된 기술이 컴퓨터 그래픽 이미지의 모습을 극적으로 향상시키는 반면에, 그것들은 또한 어떤 한계를 가지고 있다. 특히, 그것들은 자체적인 수차(abberation)를 나타낼 수 있고, 디스플레이 장치에서 표현되는 픽셀 밀도에 의해 전형적으로 제한된다.

그 결과, 렌더링되는 픽셀의 수 뿐 아니라 렌더링되는 이미지의 품질을 향상시키기 위하여, 향상된 성능 레벨을 활용할 수 있는 그래픽 시스템이 요망된다. 더욱이, 안티-앨리어싱(anti-aliasing)과 같이 그래픽 효과를 증진시키는 처리 능력을 향상시킬 수 있는 그래픽 시스템이 바람직하다.

종래 기술의 그래픽 시스템은 일반적으로 이 목적들에 미달되어 왔다. 종래 기술의 그래픽 시스템은 디스플레이상의 픽셀/비디오를 리프레쉬(refresh)하기 위한 종래의 프레임 버퍼를 사용한다. 프레임 버퍼는 디스플레이상에서 각각의 행과 열에 정확히 일치하는 행과 열의 픽셀을 저장한다. 종래 기술의 그래픽 시스템은 2D 및/또는 3D 이미지나 물체를 픽셀 형태로 프레임 버퍼로 렌더링(rendering)하고, 그리고 나서 디스플레이를 리프레쉬하기 위하여 스크린 리프레쉬 동안에 프레임 버퍼로부터 픽셀을 읽는다. 따라서, 프레임 버퍼는 디스플레이로 제공되는 출력 픽셀을 저장한다. 프레임 버퍼가 업데이트됨과 동시에 스크린을 리프레쉬함에 따라 발생될 수 있는 시각적 아티팩트(artifact)를 줄이기 위해, 대부분의 그래픽 시스템의 프레임 버퍼는 2중 버퍼로 되어 있다.

보다 현실적인 이미지를 얻기 위하여, 몇 개의 종래 기술의 그래픽 시스템은 픽셀당 두 개 이상의 샘플(sample)을 생성하는 것에 의해 진보되어 왔다. 여기서 이용되는 바와 같이, " 샘플(sample)" 이라는 용어는 물체 또는 이미지 상의 특정한 위치의 컬러, 색 깊이(z), 투명도 및 잠재적으로 다른 정보를 암시하는 계산된 컬러 정보(color information)를 말한다. 예를 들어, 하나의 샘플은 다음과 같은 성분값을 포함할 수 있다: 빨간색의 값, 녹색의 값, 파란색의 값, z 값, (예컨대, 샘플의 투명도를 나타내는) 알파값. 하나의 샘플은 또한, 예컨대, z-깊이 값, 흐릿함의 값, 강도의 값, 상대적 밝기 정보와 같은 다른 정보를 포함할 수도 있으며, 샘플이 컬러 정보라기 보다는 부분적 또는 완전한 제어 정보로 구성된 것이라는 지시(즉, " 샘플 제어 정보")를 포함할 수 있다. 픽셀보다 더 많은 샘플을 계산함에 의해(즉, 수퍼-샘플링(super-sampling)), 디스플레이 장치상에서 표현될 수 있는 것보다 더욱 상세한 이미지가 계산될 수 있다. 예를 들어, 그래픽 시스템은 디스플레이 장치로 출력될 수 있는 각 픽셀을 위해 4개의 샘플을 계산할 수 있다. 그 샘플들이 계산된 후에, 그것들은 프레임 버퍼에 저장되는 픽셀을 형성하기 위해 합성되거나 여과되고, 디스플레이 장치로 이송된다. 이러한 방법으로 형성된 픽셀을 이용하면 더욱 현실적인 최종 이미지를 만들 수 있는데, 이는 이미지 내에서 지나치게 갑작스런 변화라도 여과 프로세스(filtering process)에 의해 부드럽게 될 수 있기 때문이다.

이 종래 기술의 수퍼-샘플링 시스템들은 전형적으로 디스플레이 상의 픽셀 위치의 수보다도 훨씬 많은 수의 샘플을 생성한다. 이 종래 기술의 시스템들은 전형적으로 샘플을 계산하여 렌더 버퍼(render buffer)에 저장하는 렌더링 프로세서를 가지고 있다. 그러면, 필터링 하드웨어는 그 렌더 버퍼로부터 그 샘플들을 읽고, 그 샘플들을 여과하여 픽셀을 만들고, 그리고 나서 그 픽셀들을 전통적인 프레임 버퍼에 저장한다. 전통적인 프레임 버퍼는 전형적으로 2중-버퍼이며, 그중 한쪽이 필터링 하드웨어에 의해 업데이트되는 동안 다른 한쪽은 디스플레이 장치를 리프레쉬(refresh)하는데 이용된다. 일단 샘플이 여과되면, 그 결과인 픽셀은 디스플레이 장치를 리프레쉬하는데 사용되는 전통적인 프레임 버퍼에 저장된다. 그러나, 이 시스템들은 일반적으로 전통적인 프레임 버퍼에 의해 부과되는 한계와, 렌더 버퍼 및 여과로 인한 추가적인 잠복성(latency)에 의해 부과되는 한계를 겪어왔다. 그러므로, 전통적인 프레임 버퍼의 결점을 피하는 동시에 픽셀 수퍼-샘플링의 이점을 포함하는 진보된 그래픽 시스템이 요망된다.

" 가변 해상도 샘플 버퍼를 가지는 그래픽 시스템(Graphics System With a Variable Resolution Sample)" 이라는

명칭의 미국 특허 출원 번호 09/251844에서는, 디스플레이를 리프레쉬하기 위해 수퍼-샘플되는 샘플버퍼(super-sampled sample buffer) 및 샘플-픽셀 계산유닛(sample-pixel calculation unit)를 활용하는 컴퓨터 그래픽 시스템을 나타낸다. 그래픽 프로세서는 복수의 샘플을 생성하고 그것들을 샘플버퍼에 저장한다. 그래픽 프로세서는 바람직하게는 디스플레이 상의 픽셀 위치들의 적어도 하나의 일부분을 위해 두 개 이상의 샘플을 생성하고 저장한다. 따라서, 그 샘플버퍼는 디스플레이 상의 픽셀 위치의 수보다 훨씬 많을 수 있는 샘플을 저장하는 수퍼-샘플되는 샘플버퍼(super-sampled sample buffer)이다. 샘플-픽셀 연산유닛(sample-pixel calculation unit)은 수퍼-샘플되는 샘플버퍼(super-sampled sample buffer)로부터 샘플들을 읽거나 그 샘플들을 각각의 출력 픽셀들로 여과하거나 콘벌브(convolve)하도록 구성되는데, 여기서 출력 픽셀들은 디스플레이를 리프레쉬 하기 위해 제공된다. 샘플-픽셀 연산유닛은 하나 또는 그 이상의 샘플을 선별하고 그것들을 출력 픽셀을 생성하기 위해 여과한다. 샘플-픽셀 연산유닛은 샘플을 얻고, 디스플레이로 프레임 버퍼의 개재 없이 직접 제공되는 픽셀을 생성하도록 작동할 수 있다.

발명의 상세한 설명

위에서 나타난 문제점은 샘플버퍼와 복수의 병렬의 샘플-픽셀 연산유닛을 활용하도록 구성된 그래픽 시스템으로서, 그 샘플-픽셀 연산유닛이 샘플버퍼의 상이한 부분에 병렬로 액세스하도록 구성되어 있는 그래픽 시스템에 의해 적어도 부분적으로는 해결될 수 있을 것이다. 유리하게는, 이러한 구성은 (실시예에 따라서) 그래픽 시스템으로 하여금 픽셀을 저장하는 전통적인 프레임 버퍼 대신에 샘플버퍼를 사용하도록 할 수 있다. 샘플-픽셀 연산유닛은 병렬로 작동하도록 구성될 수 있기 때문에, 그래픽 시스템의 대기시간은 어떤 실시예에서는 줄여질 수 있을 것이다.

일례로, 그래픽 시스템은 하나 이상의 그래픽 프로세서, 샘플버퍼 및 복수의 샘플-픽셀 연산유닛을 포함할 수 있다. 그래픽 프로세서는 한 세트의 3차원 그래픽 데이터를 수신하고 그 그래픽 데이터에 기초하여 복수의 샘플을 렌더링할 수 있다. 샘플버퍼는 상기 샘플-픽셀 연산유닛를 위한 상기 복수의 샘플을 저장하도록 구성될 수 있으며(예컨대, 이중-버퍼로 된 구성에서), 상기 샘플-픽셀 연산유닛은 상기 샘플버퍼로부터 샘플을 수신하고 여과하여 출력픽셀을 생성하도록 구성될 수 있다. 출력픽셀은 디스플레이 장치 상의 이미지를 형성하기 위해 사용될 수 있다. 각 샘플-픽셀 연산유닛은 이미지의 상이한 영역에 대응하는 픽셀을 생성하도록 구성된다. 그 영역은 이미지의 수직 줄(즉, 열), 수평 줄(즉,행), 또는 이미지의 직사각형 모양의 부분일 수도 있다. 여기에서 사용되는 용어 " 수평 행" 과 " 수평 줄" 은 상호 교환하여 사용될 수 있으며, " 수직 열" 과 " 수직 줄" 도 상호 교환하여 사용될 수 있음에 주의한다. 각 영역은 시각적 수차(예컨대, 이미지에 나타난 금, 선, 틈)를 방지하기 위하 이미지의 다른 영역을 오버랩할 수도 있다. 앞에서 언급한 바와 같이, 각 샘플-픽셀 연산유닛은 그 자신의 영역 또는 여러 영역에 병렬로 작동하도록 유리하게 구성될 수 있다. 샘플-픽셀 연산유닛은 (i)어느 샘플들이 소정의 필터 포락선(filter envelope) 안에 있는가 판단하고, (ii)그 샘플들을 중량(weighting)으로 곱하고, (iii)그 결과값을 더하고, (iv)그 결과를 정규화하여 출력픽셀을 형성함으로써 그 샘플들을 처리하도록 구성된다. 중량값은 필터 포락선 안의 그 샘플의 위치에 따라 변화할 수 있다(예컨대, 가중인자는 샘플이 필터 포락선의 중심으로부터 멀리 이동될수록 감소할 것이다). 일부 실시예에서, 가중인자(weighting factor)는 정규화되거나 또는 미리-정규화될 수 있으며, 그 경우 출력이 이미 정규화되어 있을 것이므로 그 결과적인 출력픽셀은 정규화를 거치지 않을 것이다. 정규화된 가중인자(weighting factor)는, 보다 적은 갯수의 기여 샘플로 생성된 픽셀이 보다 많은 갯수의 기여 샘플로 생성된 픽셀을 압도하지 않음을 확실하게 하도록 조절된다. 정규화는, 가변적 갯수의 샘플이 각 출력 픽셀에 기여할 수 있는 그래픽 시스템의 실시예에서, 일반적으로 수행될 수 있을 것이다. 또한, 정규화는 가변적 샘플 패턴을 허용하는 시스템 및 필터의 중심의 피치(pitch)가 샘플 패턴에 대하여 광범위하게 변화하는 시스템에서 수행될 수 있을 것이다.

일부 실시예에서, 상기 그래픽 시스템은 사용되는 여역의 크기나 유형을 유동적으로 바꾸도록 구성될 수 있다(예컨대, 프레임 기준에서 사용되는 수직적 열의 폭을 바꿈). 그래픽 시스템의 어떤 실시예는 가변 해상도 또는 가변 밀도 프레임 버퍼를 지원할 수 있다. 이 구성들에서, 그래픽 시스템은 이미지의 어느 특별한 영역에서 더욱 고밀도로 샘플을 렌더링할 수 있다(예컨대, 이미지의 센터 또는 보는 사람의 주의가 가장 집중될 것 같은 이미지의 일부분). 유리하게는, 사

용되는 영역의 사이즈 및/또는 모습을 유동적으로 변화시킬 수 있는 능력은 그래픽 시스템으로 하여금 각 샘플-픽셀 연산유닛이 특정한 프레임을 위해 처리하는 샘플의 수를 균일하게 한다(또는 균일에 가깝게 한다).

샘플은 컬러 성분 및 알파(예컨대, 투명) 성분을 포함할 수 있고 " 빈(bins)" 안에 저장되어 샘플을 저장하고 샘플버퍼로부터 회수하는 과정을 단순화할 수 있다. 아래에 더욱 상세히 설명되는 바와 같이, 빈(bin)은 샘플버퍼를 더욱 작은 세트의 저장 장소로 조직화하고 나누는 수단이다. 더욱이, 어떤 실시예에서 3차원 그래픽 데이터는 압축된 형태로 (예컨대, 기하 압축) 수신될 수 있다. 이 실시예들에서 그래픽 프로세서는 샘플을 렌더링하기 전에 3차원 그래픽 데이터를 압축풀도록 구성될 수 있다. 여기서 사용되는 바와 같이, 용어 " 컬러 성분(color component)" 은 픽셀이나 샘플의 컬러를 결정하는데 사용되는 샘플마다(per-sample) 또는 픽셀마다(per-pixel)의 기준에서의 정보를 포함한다. 예를 들어, RGB 정보와 투명 정보가 컬러 성분이 될 수 있다.

한 세트의 3차원 그래픽 데이터를 렌더링하는 방법이 또한 고려될 수 있다. 일례에서 상기 방법은 (i)3차원 그래픽 데이터를 수신하고, (ii)상기 그래픽 데이터에 기초하여 하나 이상의 샘플을 생성하고, (iii)상기 샘플을 저장하고, (iv)저장된 샘플을 선별하고, (iv)상기 선별된 샘플을 병렬로 여과하여 출력픽셀을 생성함을 포함한다. 상기 저장 샘플은, 위에서 기술한 바와 같이, 복수의 영역에 따라 선별될 수 있다.

도면의 간단한 설명

본 발명의 다른 목적 및 특징과 이점은 물론 앞에서 설명한 목적은 다음과 같은 첨부도면과 함께 다음의 상세한 설명을 참조함으로써 보다 완전하게 이해할 수 있을 것이다.

도 1A는 그래픽 시스템의 일례를 포함하는 컴퓨터 시스템의 일례를 나타낸 도면;

도 1B는 그래픽 시스템을 포함하는 컴퓨터 시스템의 또 다른 실시예를 나타낸 도면;

도 1C는 가상 현실 워크 스테이션의 일부인 컴퓨터 시스템의 또 다른 실시예를 나타낸 도면;

도 2는 도 1A-C의 컴퓨터 시스템이 연결되어 있는 네트워크의 일례를 나타낸 도면;

도 3A는 가상 현실 워크 스테이션으로서의 도 1의 그래픽 시스템의 또 다른 실시예를 나타낸 다이어그램;

도 3B는 샘플버퍼를 가진 그래픽 시스템의 일례를 나타내는 보다 상세한 다이어그램;

도 4는 종래의 픽셀 계산을 나타낸 도면;

도 5A는 수퍼-샘플링의 일례를 나타낸 도면;

도 5B는 샘플의 무작위 분배를 나타낸 도면;

도 6은 가변 해상도 수퍼-샘플되는 샘플버퍼(variable resolution super-sampled sample buffer)의 일례를 가지는 그래픽 시스템의 일례를 상세히 나타낸 도면;

도 7은 가변 해상도 수퍼-샘플되는 샘플버퍼(variable resolution super-sampled sample buffer) 및 이중-버퍼로 된 샘플위치 메모리의 일례를 가지는 그래픽 시스템의 또 다른 일례를 상세히 나타낸 도면;

도 8은 3개의 상이한 샘플위치도해의 실시예를 상세히 나타낸 도면;

도 9는 샘플위치도해의 일례를 상세히 나타낸 도면;

도 10은 샘플위치도해의 또 다른 일례를 상세히 나타낸 도면;

도 11A는 수직 스크린 줄(열)을 사용하여 샘플을 픽셀로 병렬로 전환하도록 구성된 그래픽 시스템의 일례를 상세히 나타낸 도면;

도 11B는 수직 스크린 줄(열)을 사용하여 샘플을 픽셀로 병렬로 전환하도록 구성된 그래픽 시스템의 다른 일례를 상세히 나타낸 도면;

도 12는 수평 스크린 줄(행)을 사용하여 샘플을 픽셀로 병렬로 전환하도록 구성된 그래픽 시스템의 다른 일례를 상세히 나타낸 도면;

도 13은 직사각형 영역을 사용하여 샘플을 픽셀로 병렬로 전환하도록 구성된 그래픽 시스템의 다른 일례를 상세히 나타낸 도면;

도 14는 샘플을 샘플버퍼로부터 읽는 방법을 상세히 나타낸 도면;

도 15는 경계조건을 다루는 방법의 일례를 상세히 나타낸 도면;

도 16는 경계조건을 다루는 방법의 또 다른 일례를 상세히 나타낸 도면;

도 17은 샘플을 수퍼-샘플되는 샘플버퍼로 드로어(draw)하는 방법의 일례를 나타낸 순서도;

도 18은 삼각형 꼭지점들을 코딩(coding)하는 방법의 일례를 나타낸 도면;

도 19는 샘플로부터 픽셀을 연산하는 방법의 일례를 나타낸 도면;

도 20은 예제적 샘플 세트에 대하여, 샘플을 픽셀로 연산하는 일례를 상세히 나타낸 도면;

도 21은 샘플의 밀도를 변화시키는 방법의 일례를 나타낸 도면;

도 22는 샘플의 밀도를 변화하는 방법의 또 다른 일례를 나타낸 도면;

도 23은 샘플의 밀도를 변화하는 한 방법의 또 다른 일례를 나타낸 도면;

도 24A-B는 샘플의 밀도를 변화하기 위해 눈-추적을 활용하는 방법의 일례를 상세히 나타낸 도면;

도 25A-B는 샘플의 밀도를 변화하기 위해 눈-추적을 활용하는 방법의 일례를 상세히 나타낸 도면이다.

본 발명이 다양하게 수정되고 대체적 형태로 되는 것이 가능하겠지만, 그 특정된 실시예가 도면의 예에 의해 나타나며 이하에서 상세히 설명될 것이다. 그러나, 그에 대한 도면 및 상세한 설명은 본 발명을 설명된 특정된 형태에만 국한시키는 것이 아니며, 그와는 반대로, 첨부된 특허청구범위에 의해 정의된 본 발명의 기술적 사상 및 범위 내에 있는 모든 수정, 등가, 대체를 포함한다.

실시예

컴퓨터 시스템 - - 도 1

이제 도 1A을 참조하면, 3차원(3-D) 그래픽 시스템을 포함하는 컴퓨터 시스템(80)의 일례가 나타나 있다. 3-D 그래픽 시스템은 특히 컴퓨터 시스템, 네트워크 PC, 인터넷 장치, HDTV 시스템과 인터랙티브 텔레비젼 시스템을 포함한 텔레비전, 퍼스널 디지털 어시스턴트(PDAs), 그리고 2D 및/또는 3D 그래픽을 나타내는 다른 장치 등을 포함한 다양한 시스템 중 어느 하나에 포함될 수 있다.

나타낸 바와 같이, 컴퓨터 시스템(80)은 시스템 유닛(82)과, 시스템 유닛(82)에 결합된 비디오 모니터 또는 디스플레이 장치(84)를 포함한다. 디스플레이 장치(84)는 (예컨대, CRT, LCD, 가스-플라즈마 디스플레이, 디지털 거울 디스플레이(DMD) 또는 반사 실리콘 LCD 등과 같은) 다양한 유형의 디스플레이 모니터나 장치 중 어느 것이라도 될 수 있다. 키보드(86) 및/또는 마우스(88), 또는 다른 입력 장치(예컨대, 트랙볼, 디지타이저, 타블렛, 6-도의 자유 입력장치, 머리 추적기(head tracker), 눈 추적기(eye tracker), 데이터 장갑(data glove), 보디 센서(body sensor) 등)를 포함한 다양한 입력 장치가 컴퓨터 시스템에 연결될 수 있다. 디스플레이 장치(84)에 3-D 그래픽 물체를 표현하기 위해 컴퓨터 시스템(80)에 의해 어플리케이션 소프트웨어가 실행될 수 있다.

컴퓨터 시스템(80)은 또한, 눈-추적 센서(92) 및/또는 3D-안경(90)을 포함할 수 있다. 3D안경(90)은 능동적(예컨대, LCD 셔터형) 또는 수동적(예컨대, 폴라라이즈거나, 적-녹색 등의)일 수 있고, 사용자가 디스플레이 장치(84)상에서 더욱 3차원적인 이미지를 볼 수 있도록 해 줄 수 있다. 안경(90)과 함께, 각 눈은 다소 상이한 이미지를 받으며, 보는 사람의 마음은 이를 " 진정한" 3차원적 모습이라고 해석하도록 한다. 센서(92)는 디스플레이 장치(84)상의 이미지의 어느 부분을 보는 사람이 바라보고 있는지(즉, 보는 사람의 시야가 중심을 두고 있는지) 판단하도록 구성될 수 있다. 센서(92)로부터 제공받는 정보는 다음에 설명되는 바와 같이 수많은 상이한 방법으로 이용될 수 있다.

가상 현실 컴퓨터 시스템 -- 도 1B

도 1B는 컴퓨터 시스템(70)의 또다른 일례를 나타낸다. 이 실시예에서, 시스템은 머리에 실장되는 디스플레이 장치(72), 머리-추적 센서(74), 및 데이터 장갑(76)을 포함한다. 머리에 실장되는 디스플레이(72)는 시스템 유닛(82)에 광섬유 연결(94) 또는 전기적 전도성 연결, 적외선 연결, 또는 무선(예컨대, 라디오 주파수) 연결 중 하나 이상을 경유하여 결합될 수 있다.

컴퓨터 네트워크 -- 도 2

이제 도 2를 참조하면, 적어도 하나의 서버 컴퓨터(502)와 하나 이상의 클라이언트 컴퓨터(506A-N)를 포함하는 컴퓨터 네트워크(500)가 나타나 있다. (도 4에 나타낸 실시예에서는, 클라이언트 컴퓨터(506A-B)가 묘사되었다). 하나 이상의 클라이언트 시스템이 각각 위에서 설명한 바와 같은 하나 이상의 그래픽 시스템(112)을 가지는 컴퓨터 시스템(80)과 유사하게 구성될 수 있다. 서버(502)와 클라이언트(들)(506)는 근거리 통신망(LAN), 광대역 통신망(WAN), 또는 인터넷 연결 등과 같은 다양한 연결(504)을 통하여 이어질 수 있다. 일례에서, 서버(502)는 (압축되어 있을 수도 있는) 3-D 기하 데이터를 저장하고 클라이언트(506)에 전달할 수 있다. 클라이언트(506)는 압축된 3-D 기하 데이터를 수신하고, (필요하면) 압축을 풀고 기하 데이터를 렌더링한다. 그러면 렌더링된 이미지는 클라이언트의 디스플레이 장치상에 표현된다. 클라이언트는 기하 데이터를 렌더링하고 수퍼-샘플되는 샘플버퍼와 위에서 설명된 실시간 필터 기술을 사용하여 이미지를 표현한다. 또 다른 실시예에서, 압축된 3-D 기하 데이터는 클라이언트 컴퓨터(506) 사이에서 전달될 수 있다.

컴퓨터 시스템 블록도 -- 도 3A

도 3A은 컴퓨터 시스템(80)을 나타내는 간략화된 블록도를 보이고 있다. 본 발명의 이해를 위해 필요하지 않는 컴퓨터 시스템(80)의 구성요소는 편의상 나타내지 않는다. 컴퓨터 시스템(80)은 호스트 중앙처리장치(CPU;102)와 시스템 버스(104)에 결합된3-D 그래픽 시스템(112)을 포함한다. 시스템 메모리(106)가 또한 시스템 버스(104)에 연결될 수 있다.

호스트 CPU(102)는 다양한 프로세서 기술 중 어느것에 의해서도 실현될 수 있다. 예를 들어, 호스트 CPU(102)는 하나 이상의 일반 목적의 마이크로프로세서, 병렬 프로세서, 벡터 프로세서, 디지털 시그널 프로세서 등의 하나 이상 또는 그 조합을 포함하여 구성될 수 있다. 시스템 메모리(106)는 다른 유형의 메모리 기술을 나타내는 하나 이상의 메모리 서브시스템을 포함할 수 있다. 예를 들어, 시스템 메모리(106)는 롬(ROM), (정적램인 " SRAM" , 동기 동적램인 " SDRAM" , 램버스 동적램인 " RDRAM" 램 등의) 램(RAM)과 대량 저장 장치를 포함할 수 있다.

시스템 버스(104)는 (호스트 프로세서와 메모리 서브시스템 사이의 통신을 위해) 하나 이상의 통신 버스 또는 호스트 컴퓨터 버스를 포함할 수 있다. 더욱이, 다양한 주변장치와 주변버스가 시스템 버스(104)에 연결될 수 있다.

그래픽 시스템(112)은 본 발명의 원칙에 따라 구성될 수 있고, 크로스바 스위치 또는 다른 유형의 버스 연결로직에 의해 시스템 버스(104)에 결합될 수 있다. 그래픽 시스템(112)은 프로젝션 장치(PD_1 - PD_L)와 디스플레이 장치(84) 각각을 대응하는 비디오 신호를 가지고 구동한다.

3-D 그래픽 시스템(112)은 시스템 버스(104)에 더하여 하나 이상의 다양한 유형의 버스에 결합될 수 있다. 더욱이, 3D 그래픽 시스템(112)은 통신포트에 결합될 수 있으며, 그에 의해, 인터넷 또는 근거리 통신망(LAN)과 같은 외부소스로부터 그래픽 데이터를 직접 수신할 수 있다.

호스트 CPU(102)는 시스템 버스(104)를 통해 프로그램 된 입/출력(I/O)에 따라서 그래픽 시스템(112)으로/부터 정보를 전송할/전송받을 수 있을 것이다. 그와 달리, 그래픽 시스템(112)은 직접 메모리 액세스(DMA) 또는 인텔리전트 버스-마스터링에 의해 시스템 메모리(106)에 액스스 할 수 있다.

OpenGL(Silicon Graphics,Inc.사의 등록상표) 또는 Java3D(Sun Microsystems,Inc.사의 등록상표)와 같은 애플리케이션 프로그래핑 인터페이스(API)에 적합한 그래픽 애플리케이션 프로그램은, 호스트 CPU(102)상에서 실행될 수 있으며 프로젝션 장치PD_1 - PD_L로 출력될 다각형과 같은 기하 원형(geometric primitive)을 정의하는 명령과 데이터를 생성한다. 그 후, 호스트 CPU(102)는 시스템 버스(104)를 통해 그래픽 데이터를 그래픽 시스템(112)으로 전송할 수 있다. 또 다른 실시예에서, 그래픽 시스템(112)은 DMA 액세스 싸이클을 사용하여 시스템 메모리(106)로부터 기하 데이터 배열을 읽을 수 있다. 또 다른 실시예에서, 그래픽 시스템(112)은 인텔사에 의해 공표된 AGP와 같은 직접포트를 통하여 시스템 메모리(106)에 결합될 수 있다.

그래픽 시스템(112)은 호스트 CPU(102) 및/또는 다른 시스템 메모리(106), 다른 메모리, 또는 (예컨대, 인터넷 등의) 네트워크와 같은 외부 소스로부터 또는 (예컨대, 텔레비전과 같은) 방송매체 또는 다양한 소스 중 어느 곳에서부터라도 그래픽 데이터를 수신받을 수 있다.

다음에 설명되는 바대로, 그래픽 시스템(112)은 보다 효율적인 마이크로코드(microcode) 제어를 할 수 있도록 구성될 수 있으며, 이는 호스트 CPU(102)에 의해 생성된 다각형에 대응하는 후속의 컬러값을 취급하는 성능을 향상시킨다. 그래픽 시스템(112)이 컴퓨터 시스템(80)의 일부로 묘사되는 한편, 그래픽 시스템(112)은 또한 독립 장치(예컨대, 자체 설치된 디스플레이)로서 구성될 수도 있다. 그래픽 시스템(112)은 단일 칩 장치로서 또는 시스템상의 칩이나 멀티-칩 모듈의 일부로서 구성될 수도 있다.

그래픽 시스템(112)은 네트워크 PC, 인터넷 장치, (HDTV 시스템 또는 인터랙티브 텔레비젼 시스템을 포함하는) 텔레비젼, 개인용 디지털 어시스턴트(PDA), 또는 2D 및/또는 3D 그래픽을 표시하는 다른 장치를 포함하는 다양한 시스템 중 어느것에도 포함될 수 있다.

이하에서 더 설명하는 바와 같이, 컴퓨터 시스템(80)의 3-D 그래픽 시스템은, 프로젝션 장치(PD_1 - PD_L) 또는 디스플레이 장치(84)에 표현되는 이미지의 품질과 현실성을 향상시키기 위하여, 프로그램될 수 있는 샘플-픽셀 연산유닛과 함께 수퍼-샘플되는 샘플버퍼를 포함한다. 각 샘플-픽셀 연산유닛은 샘플버퍼내의 샘플에 응답하여 픽셀을 생성하기 위한 다른 하드웨어 또는 필터(filter)나 콘벌브 파이프라인(convolve pipeline)을 포함할 수 있다. 각 샘플-픽셀 연산유닛은 샘플버퍼로부터 샘플을 얻고 프로젝션 장치(PD_1 - PD_L) 또는 디스플레이로 직접 제공되는 픽셀을 생성하도록 작동할 수 있다. 샘플-픽셀 연산유닛은 " 실-시간(real-time)" 또는 " 온-더-플라이(on-the-fly)" 방식으로 작동할 수 있다.

여기서 사용되는 바와 같이, 용어 " 여과하기(filter)" 와 " 콘벌브(convolve)" 는 서로 교환하여 사용될 수 있다. 여기서 사용되는 바와 같이, 용어 " 실-시간(real-time)" 은 프로젝션 장치(PD_1 - PD_L) 또는 디스플레이 장치의 리프레쉬 속도(refresh rate)에서 또는 그 근처에서 수행되는 기능을 말한다. " 온-더-플라이(on-the-fly)" 란 부드럽게 보이도록 하는 표현 동작(즉, 동작혼합)과 연속적으로 보이도록 하는 빛의 세기(즉, 명멸혼합)을 위해 요구되는 속도의 부근 또는 최소 속도에서 이미지를 생성하는 과정 또는 작동을 말한다. 이러한 개념들은 Russel L.De Valois and Karen K.De Valois 저(著) " Spatial Vision" , Oxford University Press, 1988에 더 자세히 설명되어 있다.

그래픽 시스템 - - 도 3B

도 3은 본 발명에 따른 그래픽 시스템(112)의 일례를 나타낸 블록도를 보이고 있다. 그래픽 시스템(112)은 하나 이상의 그래픽 프로세서(GPU;90), 하나 이상의 수퍼-샘플되는 샘플버퍼(162), 그리고 하나 이상의 샘플-픽셀 연산유닛(170-1 - 170-V)을 포함할 수 있다. 그래픽 시스템(112)은 또한 하나 이상의 디지털-아날로그 컨버터(DACs;178-1 - 178-L)를 포함할 수도 있다. 그래픽 프로세싱 유닛(90)은 특수화된 그래픽 프로세서나 연산유닛, 멀티미디어 프로세서, DSP, 또는 일반 목적의 프로세서라도 될 수 있다.

일례로, 그래픽 프로세서(90)는 하나 이상의 렌더링 유닛(150A-D)을 포함할 수 있다. 그래픽 프로세서(90)는 또한 하나 이상의 제어 유닛(140), 하나 이상의 데이터 메모리(152A-D), 그리고 하나 이상의 스케쥴 유닛(154)를 포함하고 있다. 샘플버퍼(162)는 또한 하나 이상의 샘플 메모리(160A-160N)를 포함할 수도 있다.

A. 제어 유닛(140)

제어 유닛(140)은 그래픽 시스템(112)과 컴퓨터 시스템(80) 사이의 데이터 전송을 제어함으로써 그래픽 시스템(112)과 컴퓨터 시스템(80) 사이의 인터페이스로서 동작한다. 두 개 이상의 렌더링 유닛(150A-D)을 포함하는 그래픽 시스템(112)의 실시예에서, 제어 유닛(140)은 또한 컴퓨터 시스템(80)으로부터 수신 받은 데이터 흐름을 개별적 렌더링 유닛(150A-D)으로 보내지는 대응하는 수의 병렬 흐름으로 나눌 수도 있다. 그래픽 데이터는 압축 형태로 컴퓨터 시스템(80)으로부터 수신 받을 수 있다. 그래픽 데이터 압축은 컴퓨터 시스템(80)과 그래픽 시스템(112)간에 요구되는 전송 대역폭을 유리하게 줄일 수 있다. 일례로, 제어 유닛(140)은 수신된 데이터 스트림을 분할하고 압축 형태로 렌더링 유닛(150A-D)으로 보내도록 구성될 수 있다.

그래픽 데이터는 하나 이상의 그래픽 원형(graphics primitives)을 포함할 수 있다. 여기서 이용되는 바와 같이, 그래픽 원형(graphics primitives)이라는 용어는 다각형(polygons), 파라미터적 표면(parametric surfaces), 스플라인(splines), NURBS(non-uniform rational B-splines;비-균일 적합 B-스플라인), 서브-디비젼 표면(sub-divisions surfaces), 프랙탈(fractals;차원 분열(次元分裂) 도형), 부피 원형(volume primitives), 입자 체계(particle systems)를 포함한다. 이 그래픽 원형은 James D.Foley 저 " Computer Graphics:Principles and Practice" et al., published by Addison Wesley Publishing Co.,Inc., 1996 의 교과서에 상세히 설명되어 있다.

여기에 나타낸 본 발명의 실시예와 예제는 간략화를 위해 다각형의 견지에서 기술되어 있다. 그러나, 그래픽 원형의 어떠한 유형도 이들 실시예들과 예제들의 다각형에 대신하거나 부가하여 사용될 수 있다.

B. 렌더링 유닛(Rendering Unit)

여기서 (드로어 유닛(draw unit)으로도 호칭되는) 렌더링 유닛(150A-D)은 그래픽 명령과 제어 유닛(140)으로부터의 데이터를 수신한 후 정확한 수행에 의존하는 많은 기능을 수행하도록 구성된다. 예를 들어, 렌더링 유닛(150A-D)은 그래픽 데이터 내에서 발생하는 다양한 그래픽 원형의 (수신된 그래픽 데이터가 압축된 경우에는) 압축풀기(decompression), 변환(transformation), 클리핑(clipping), 라이팅(lighting), 텍스쳐링(texturing), 심도 신호(depth cueing), 투명 처리(transparency processing), 셋-업(set-up), 보이는 물체 판단, 그리고 가상의 스크린 공간 렌더링(screen space rendering)을 수행하도록 구성될 수 있다.

수신되는 압축된 그래픽 데이터의 유형에 따라, 렌더링 유닛(150A-D)은 산술적 디코딩(arithmetic decoding), 런-길이 디코딩(run-length decoding), 허프만 디코딩(Huffman decoding), 그리고 (예컨대, LZ77, LZSS, LZ78, LZW와 같은) 사전 디코딩(dictionary decoding)을 수행하도록 구성될 수 있다. 또 다른 실시예에서, 렌더링 유닛(150A-D)는 기하 압축을 이용하여 압축된 그래픽 데이터를 디코드하도록 구성될 수 있다. 3D 그래픽 데이터의 기하 압축은 이미지 품질을 거의 그대로 유지하면서 데이터 크기를 크게 줄일 수 있다. 3D 기하도형을 압축하고 푸는 2가지 방법이:

미국 특허 번호 5,793,371, 출원번호 08/511,294, 1994년 8월 4일 출원, 발명의 명칭 "Method And Apparatus For Geometric Compression Of Three-Dimensional Graphics Data," 대리인 등록번호 5181-05900; 및

미국 특허 출원번호 09/095,777, 1998년 월 11일에 출원, 발명의 명칭 "Compression of Three-Dimensional Geometry Data Representing a Regularly Tiled Surface Portion of a Graphical Object," 대리인 등록번호 5181-06602에 설명되어 있다.

압축풀기를 지원하는 그래픽 시스템(112)의 실시예에서, 각 렌더링 유닛(150)로부터 수신되는 그래픽 데이터는 하나 이상의 그래픽 "원형(primitives)"으로 압축이 풀리고, 그 후 렌더될 수 있다. 원형(primitive)이라는 용어는 물체의 형상(예컨대, 꼭지점, 선, 삼각형, 2차원이나 3차원의 다각형, 다면체, 또는 3차원상의 자유형태의 표면)을 정의하는 물체의 구성요소를 말한다. 렌더링 유닛(150)은 어떠한 적절한 유형의 고성능 프로세서(예컨대, 특수화된 그래픽 프로세서 또는 연산유닛, 멀티미디어 프로세서, DSPs, 또는 일반 목적의 프로세서)라도 될 수 있다.

변환(transformation)이란 원형 또는 한 세트의 원형(primitive)을 포함하는 물체에 기하도형적 작용을 적용하는 것을 말한다. 예를 들어, 개개의 좌표 시스템에서 나타내어지는 물체는 적절한 순서의 전환, 회전, 및 크기변환 등을 사용하여 실세계 공간에서 임의의 위치, 방위 및 크기를 포함할 수 있다. 변환은 또한 반사(reflection), 비틀림(skewing), 또는 다른 밀접한 변환을 포함할 수 있다. 보다 일반적으로는, 변환은 비-선형적 작용을 포함할 수 있다.

라이팅(lighting)이란, 물체의 조도를 연산하는 것을 말한다. 라이팅 계산은 물체 또는 물체상에서 선택된 점(예컨대, 꼭지점)에 컬러 및/또는 밝기를 할당한다. 이용되는 세이딩 알고리즘(예컨대, 상수, 구란드(Gourand), 또는 퐁(Phong))에 따라서, 많은 다른 위치에서 라이팅 값이 구해질 수 있다. 예를 들어, 상수 세이딩(constant shading)이 이용되면(즉, 다각형의 빛이 부여된 표면이 상수 조도값을 할당받으면), 라이팅은 다각형 마다 한번 계산될 필요가 있다. 구란드(Gourand) 세이딩이 이용되면, 라이팅은 꼭지점 마다 한번 계산된다. 퐁(Phong) 세이딩은 라이팅을 픽셀 마다(per-pixel basis) 계산한다.

클리핑(clipping)이란, 실세계 공간에서의 3-D 시야 부피의 밖에 놓여지는 그래픽 원형(primitive) 또는 그래픽 원형의 일부분을 제거하는 것을 말한다. 3-D 시야 부피는 실세계 공간에 있는 가상의 관찰자에 보이는 실시계 공간의 일부분을 나타낼 수 있다. 예를 들어, 시야 부피는 실세계 공간에 있는 2-D 뷰 윈도우와 뷰 포인트(관찰 위치)에 의해 생성된 고체 원뿔형일 수 있다. 고체 원뿔은 뷰 포인트로부터 나와서 뷰 윈도우를 통과하는 모든 광선의 집합으로 상상될 수 있다. 뷰 포인트는 가상적 관찰자의 실세계 공간 배치를 나타낼 수 있다. 3-D 시야 부피의 밖에 놓여지는 원형 또는 원형의 일부분은 현재 보이지 않고 추가적인 처리로부터 제거될 수 있다. 3-D 시야 부피의 안에 놓여지는 그래픽 원형 또는 그래픽 원형의 일부분은 2-D 뷰 윈도우로 투영될 후보자이다.

클리핑과 투영 계산을 단순화 하기 위해서, 원형은 여기서 뷰포트 좌표 시스템으로 언급되는 2번째의 더욱 편리한 좌표 시스템으로 변형될 수 있다. 뷰포트 좌표에서, 시야 부피는 보다 클리핑에 편리할 수 있는 정준(正準)의 3-D 뷰포트로 사상된다. 용어 셋-업(set-up)이란, 이러한 그래픽 원형(primitive)을 뷰포트 좌표로 매핑(mapping)하는 것을 말한다.

상기 클리핑 계산에서 살아남은 그래픽 원형(primitives) 또는 원형의 일부는 가시 판단의 결과에 따라서, 2-D 뷰포트로 투영될 수 있다. 3-D에서 클리핑하는 대신에, 그래픽 원형은 (2-D 뷰포트를 포함하는) 2-D 뷰 평면에 투영될 수 있고, 그다음에 2-D 뷰포트에 대해 크리핑 될 수 있다.

가상 디스플레이 렌더링은 투영되는 그래픽 원형을 위한 샘플을 생성하도록 수행되는 연산을 말한다. 예를 들어, 3-D에서 삼각형의 꼭지점이 2-D 뷰포트 상에 투영될 수 있다. 투영된 삼각형은 샘플들을 포함할 수 있고, (예컨대, 적색, 녹색, 청색 및 z 값 등의) 값이 투영된 꼭지점들을 위해 이미 결정된 대응값에 기초하는 샘플에 할당될 수 있다. 예를 들어, 투영된 삼각형 안의 각 샘플을 위한 적색값은 꼭지점들의 알려진 적색값을 위해 보간(補間)될 수 있다. 투영된 삼각형을 위한 이 샘플값들은 샘플버퍼(162)에 저장될 수 있다. 실시예에 따라서, 샘플버퍼(16)는 또한 각 샘플을 위한 z 값을 저장한다. 이 z-값은 깊이-버퍼링을 포함하여, 많은 이유를 위해 샘플과 함께 저장된다. 연속적인 원형을 위한 샘플이 렌더됨에 따라, 샘플버퍼(162)내에 가상 이미지가 축적된다. 따라서, 2-D 뷰포트는 가상 이미지가 렌더링되는 가상 스크린이다. 그 가상 이미지를 포함하는 샘플값은 샘플버퍼(162)에 저장된다. 2-D 뷰포트안의 점(point)들은가상 스크린 좌표 X 및 Y의 견지에서 표현되며 가상 스크린 공간에 존재한다.

가상 이미지가 완전하면, 예컨대, 모든 그래픽 원형이 렌더링되었다면, 샘플-픽셀 연산유닛(170)은 가상 이미지를 포함하는 샘플에 액세스할 수 있고 그 샘플을 여과하여 픽셀값을 생성할 수 있다. 달리 말하면, 샘플-픽셀 연산유닛(170)은 가상 이미지의 공간적 콘볼루션을 콘볼루션 커널 f(X,Y)에 대해 수행하여 픽셀값을 생성한다. 예를 들어, 픽셀 P를 위한 적색값 R_p는 다음과 같은 관계를 기초로한 가상 스크린 공간에서의 장소(X_p, Y_p)에서 계산될 수 있다.

$$R_p = \frac{1}{E} \sum f(X_k - X_p, Y_k - Y_p) R(X_k, Y_k) \quad ,$$

여기서 상기 합계는 장소(X_p, Y_p)의 이웃에 있는 샘플(X_k, Y_k)에서 계산된다. 콘볼루션 커널 f(X,Y)은 원점 근처에서만 0값이 아니므로, 커널 f(X - X_p, Y - Y_p)은 장소(X_p, Y_p)의 근처에서만 0값이 아니다. E 값은 다음과 같은 관계에 따라 계산될 수 있는 정규화 값이다.

$$E = \sum f(X_k - X_p, Y_k - Y_p) \quad ,$$

여기서 상기 합계는 위와 같이 동일한 이웃에서 계산된다. 정규화 값의 합계 E는 적색 픽셀값 R_p를 위한 합계와 병행하여 수행될 수 있다. 장소(X_p, Y_p)는 픽셀 중심 또는 픽셀 원점으로 불리운다. 콘볼루션 커널 f(X,Y)이 원점(0,0)에 대해 대칭인 경우에 픽셀 중심이라는 용어가 사용된다.

픽셀값은 프로젝션 스크린 SCR상에 표현하기 위한 프로젝션 장치(PD_1 - PD_L)로 나타내어질 수 있다. 프로젝션 장치는 각각 집약된 이미지 IMG를 생성한다. 샘플-픽셀 연산유닛(170)은 또한 디스플레이 장치(84)에 표현하기 위한 픽셀값을 생성할 수 있다.

도 3에 나타낸 그래픽 시스템(112)의 실시예에 있어서, 렌더링 유니트(150A-D)는 픽셀값 대신에 샘플값을 계산한다. 이렇게 하면, 렌더링 유니트(150A-D)가 수퍼-샘플링, 즉 픽셀당 하나 이상의 샘플을 연상하도록 한다. 본 발명에서 수퍼-샘플링의 의미는 아래에서 더욱 자세히 논의된다. 수퍼-샘플링에 대해 보다 상세히는 다음의 책에 나타나 있다 :" Principles of Digital Image Synthesis" by Andrew Glassner, 1995, Morgan Kaufman Publishing (Volume 1) 및 " Renderman Companion:" by steve Upstill, 1990, Addison Wesley Publishing.

샘플버퍼(162)는 이중-버퍼로 되어 있어서 렌더링 유닛(150A-D)은 제1 가상 이미지를 위한 샘플을 샘플버퍼(162)의 제1 부분으로 써넣을 수 있으며, 이때 제2 가상 이미지는 동시에 샘플버퍼(162)의 제2 부분으로부터 샘플-픽셀 연산유닛(170)에 의해 읽힌다.

샘플버퍼(162)로 샘플로 렌더링되는 가상 이미지와 2-D 뷰포트는 집약된 이미지 IMG 또는 디스플레이 이미지 DIM로 물리적으로 표현되는 영역보다 큰 영역에 대응할 수 있다. 예를 들어, 2-D 뷰포트는 가시적인 서브윈도우를 포함할 수있다. 가시적인 서브윈도우는 집약된 이미지 IMG 및/또는 디스플레이 이미지 DIM에 대응할 수 있으며, 반면에 (가시적 서브윈도우 외부의) 2-D 뷰포트의 가장자리는 패닝(panning) 및 줌잉(zooming)과 같은 다양한 효과 허용할 수 있다. 달리 말하면, 상기 가시적 서브윈도우 내부에 놓여 있는 가상 이미지 부분만이 물리적으로 표시된다. 일례로, 가시적 서브윈도우는 2-D 뷰포트의 전체와 동일하다. 이 경우에, 가상 이미지의 전부가 물리적으로 표시된다.

렌더링 유닛(150A-D)가 예컨대, 하나 이상의 셋-업/압축풀기 유닛과 하나 이상의 라이팅(lighting) 유닛과 같은 수많은 작고 더욱 특수화된 기능의 유닛을 포함할 수도 있음에 유의한다.

C. 데이터 메모리(Data Memories)

각 렌더링 유닛(150A-D)은 지시(instruction) 및 데이터(data) 메모리(152A-D)와 결합될 수 있다. 일례로, 각 데이터 메모리(152A-D)는 렌더링 유닛(150A-D)을 위한 데이터와 명령들을 저장하도록 구성될 수 있다. 수행이 다양할 수 있는 반면, 일례로 각 데이터 메모리(152A-D)는 각 렌더링 유닛(150A-D)을 위해 총 16메가 바이트의 저장공간을 제공하는 2개의 8메가 바이트의 SDRAM을 포함할 수 있다. 또 다른 예로, RDRAM(램버스 DRAM)이 압축풀기 및 각 렌더링 유닛의 셋-업 동작을 지원하기 위해 이용될 수 있으며, 이때 SDRAM이 렌더링 유닛의 드로어 기능을 지원하는데 이용될 수 있다. 데이터 메모리(152A-D)는 " 텍스쳐 및 렌더 메모리(152A-D)" 로 불리워질 수도 있다.

D. 스케쥴 유닛(Schedule Unit)

스케쥴 유닛(154)은 렌더링 유닛(150A-D)과 샘플링 메모리(160A-N)사이에 결합될 수 있다. 스케쥴 유닛(154)은 완성된 샘플을 나열하고 그것들을 샘플 메모리(160A-N)에 저장하도록 구성된다. 보다 큰 구성에서, 다중 스케쥴 유닛(154)은 병렬로 이용될 수 있다는 점에 주목하라. 일례로, 스케쥴 유닛(154)은 크로스바 스위치(crossbar switch)로서 수행될 수 있다.

E. 샘플 메모리(Sample Memories)

수퍼-샘플되는 샘플버퍼(162)는 샘플 메모리(160A-160N)를 포함하며, 이것들은 렌더링 유니트에 의해 생성된 복수의 샘플을 저장하도록 구성된다. 여기서 사용되는 바와 같이, " 샘플버퍼" 란 샘플을 저장하는 하나 이상의 메모리를 말한다. 앞에서 언급했던 바와 같이, 하나 이상의 샘플이 출력 픽셀을 형성하기 위해 여과된다. 출력 픽셀값은 프로젝션 스크린 SCR상의 표시를 위해 프로젝션 장치(PD_1 - PD_L)로 제공될 수 있다. 축력 픽셀값은 또한 디스플레이 장치(84)로 제공될 수도 있다. 샘플버퍼(162)는 또한 픽셀 해상도에 따라서 수퍼-샘플링, 크리틱컬(critical) 샘플링, 또는 서브-샘플링을 지원하도록 구성될 수 있다. 달리 말하면, (샘플버퍼(162)안에 저장된) 가상 이미지내의 샘플(X_k, T_k) 사이의 평균 거리는 가상 스크린 공간내의 픽셀 중심 사이의 평균거리보다 작거나, 동등하거나, 또는 클 수 있다. 더욱이, 콘볼루션 커널 f(X, Y)이 0값이 아닌 기능적 값을 여러 픽셀중심을 포함하는 주변에 취할 수 있기 때문에, 단일한 샘플은 여러 출력픽셀값에 기여할 수 있다.

샘플 메모리(160A-160N)는 다양한 사이즈로 수많은 다른 유형의 메모리(예컨대, SDRAMs, SRAMs, RDRAMs, 3DRAMs, 또는 차세대 3DRAMs) 중 어느 것이라도 포함할 수 있다. 일례로, 각 스케쥴 유닛(154)은 4개의 샘플 메모리 뱅크와 결합되고, 각 뱅크는 4개의 3DRAM-64 메모리를 포함한다. 동시에, 3DRAM-64 메모리는 픽셀당 다중 샘플을 저장하는 116-비트 딥 수퍼-샘플되는 샘플버퍼를 형성할 수 있다. 예를 들어, 일례로, 각 샘플 메모리(160A-160N)는 픽셀마다 16개의 샘플을 저장할 수 있다.

3DRAM-64 메모리는 원칩의 단일 버퍼 Z를 가지고 완전한 내부 이중 버퍼링(internal double buffering)을 지원하도록 구성된 특수화된 메모리다. 이중 버퍼 부분은 2개의 RGBX 버퍼를 포함하며, 여기서 X는 다른 정보(예컨대, 알파)를 저장하는데 이용되는 4번째 채널이다. 3DRAM-64 메모리는 또한, 윈도우 ID 정보를 받아들이고 어느 버퍼의 내용이 출력될 것인가를 선택하는 내부 2-1 또는 3-1 멀티플렉서를 제어하는 검색 테이블을 가지고 있다. 3DRAM-64 메모리는 미츠비시 전기회사(Mitsubishi Electric Corporation)의 반도체 그룹에 의해 곧 이용가능하게 될 차세대 3DRAM 메모리이다. 일례로, 조합하여 사용되는 4개의 칩은 이중 버퍼가 달린 1280 X 1024 수퍼-샘플되는 샘플버퍼를 만드는데 충분하다. 3DRAM-64 메모리들이 내부적으로 이중-버퍼로 되어 있으므로, 이중-버퍼로 된 시스템에서 2개의 프레임 버퍼 각각을 위한 입력 핀(pins)은 (메모리 내부의 멀티플렉서를 사용하여) 시간에 의해 멀티플렉스된다. 출력 핀(pins)은 마찬가지로 시간에 의해 멀티플렉스 될 수 있다. 이렇게 하면 이중 버퍼의 이점을 제공하면서도 핀 카운트를 감소시킬 수 있다. 3DRAM-64 메모리는 더 나아가서 z 출력 핀을 가지지 않음으로써 핀 카운트를 감소시킨다. z 비교 및 메모리 버퍼 선택은 내부적으로 다루어지므로, 이는 (예컨대, 출력측에 선택 로직을 적게 사용하거나 또는 사용하지 않음으로써) 샘플버퍼(162)를 단순화할 수 있다. 3DRAM-64를 사용하면 또한 메모리 대역폭을 줄일 수 있는데, 왜냐하면 데이터를 읽어 내고, z 비교를 수행하고, 그리고 나서 데이터를 다시 써 넣는 종래기술의 공정 없이 정보가 메모리로 읽혀질 수 있기 때문이다. 그 대신에, 메모리가 상기에 설명된 단계를 내부적으로 수행하므로 데이터는 3DRAM-64로 간단히 써 넣어질 수 있다.

그러나, 그래픽 시스템(112)의 다른 실시예에서, 다른 메모리들(예컨대, SDRAMs, SRAMs, RDRAMs, 또는 현재 세대의 3DRAMs)이 샘플버퍼(162)를 형성하기 위해 사용될 수 있다.

그래픽 프로세서(90)는 특별한 샘플위치도해(예컨대, 정규의 그리드, 교란된 정규 그리드 등)에 따라 복수의 샘플위치를 생성하도록 구성될 수 있다. 다른 방법으로는, 샘플위치들(또는 샘플위치를 형성하기 위해 정규의 그리드 위치에 더해지는 오프셋)은 샘플위치 메모리(예컨대, RAM/ROM 테이블)로부터 읽혀질 수 있다. 렌더링될어야 할 다각형을 수신하면, 그래픽 프로세서(90)는 샘플위치에 기초하여 어느 샘플이 다각형 내에 해당하는지 결정한다. 그래픽 프로세서(90)는 다각형 내에 해당하는 샘플을 렌더링하고 렌더링된 샘플을 샘플 메모리(160A-N)에 저장한다. 여기서 사용된 바와 같이, 용어 " 렌더(render)" 와 " 드로어(draw)" 는 서로 교환하여 사용될 수 있으며, 샘플에 대한 컬러값을 계산하는 것을 말한다는 점에 주의하라. 심도값(depth values), 알파값(alpha values) 및 샘플에 대한 다른 값이 또한 렌더링 또는 드로어 프로세스에서 계산될 수 있다.

F. 샘플-픽셀 연산유닛(Sample-to-pixel Calculation Units)

샘플-픽셀 연산유닛(170-1 - 170-V ; 이하 샘플-픽셀 연산유닛(170)이라고 통합하여 호칭된다)은 샘플 메모리(160A-N)와 DACs(178-1 ~ 178-L) 사이에 결합될 수 있다. 샘플-픽셀 연산유닛(170)은 샘플 메모리(160A-N)로부터 선택된 샘플을 읽고 샘플에 대해 콘볼루션(즉, 필터 작용)을 수행하여 DACs(178-1 ~ 178-L)로 출력되는 출력 픽셀값을 생성하도록 구성된다. 샘플-픽셀 연산유닛(170)은 희망되는 출력의 유형에 따라서, 다른 시간에 다른 필터 기능을 수행할 수 있도록 프로그램될 수 있다. 일례로, 샘플-픽셀 연산유닛(170)은 5x5 수퍼-샘플 재건 대역-통과 필터를 수행하여, (샘플 메모리(160A-N)에 저장된) 수퍼-샘플되는 샘플버퍼의 데이터를 단일의 픽셀값으로 변환한다. 다른 실시예로서, 연산유닛(170A-D)은 출력 픽셀을 계산하기 위해 선택된 수의 샘플을 여과한다. 선택된 샘플은, 계산되는 픽셀의 중심에 따라 그 위치에 기초하여 샘플에 가중치를 주는 공간적 가중 인자(weighting factor)에 의해 곱해질 수 있다. 상기 여과 작용은 다양한 필터의 어떤 것이라도 사용할 수 있으며 한가지만으로 또는 조합하여 사용할 수 있다. 예컨대, 콘볼루션 작용에는 천막 필터(tent filter), 원형 및 타원형 필터, 미첼 필터(Mitchell filters), 대역 통과 필터, 싱크 함수 필터 등이 사용될 수 있다.

샘플-픽셀 연산유닛(170)은 또한 하나 이상의 다음의 특징으로 형성될 수도 있다: 모조 컬러 테이블을 사용하는 컬러 검색, 다이렉트 컬러, 인버스 감마 교정, 샘플의 픽셀로의 여과, 프로그래머블 감마 인코딩, 및 선택적인 컬러 공간 변환. 다른 특징의 샘플-픽셀 연산유닛(170)은 프로그래머블 비디오 타이밍 생성기, 프로그래머블 픽셀 클럭 합성기, 에지-혼합 함수, 핫스팟 교정함수, 컬러공간 및 크로스바 함수. 일단 샘플-픽셀 연산유닛이 타이밍과 각 픽셀의 색을 조절하면, 픽셀은 DACs(178-1 ~ 178-L)로 출력된다.

G. DACs

디지털-아날로그 컨버터(DACs)(178-1 ~ 178-L; 이하 통합하여 DACs(178)이라고 호칭한다)는 그래픽 시스템(112)의 최종 출력 단계로서 동작한다. DACs(178)는 연산 유닛(170)으로부터 수신된 디지털 픽셀 데이터를 아날로그 비디오 신호로 전환한다. 각각의 DACs(178-1 ~ 178-L)는 프로젝션 장치(PD_1 - PD_L)중 대응하는 하나에 결합될 수 있다. DAC(178-1)은 제1 스트림의 디지털 픽셀 데이터를 하나 이상의 연산 유닛(170)으로부터 수신하며, 제1 스트림을 제1 비디오 신호로 변환한다. 제1 비디오 신호는 프로젝션 장치(PD_1)로 제공된다. 마찬가지로, DACs(178-1 ~ 178-L)는 대응하는 디지털 픽셀 데이터의 스트림을 수신하며, 그 디지털 픽셀 데이터 스트림은 대응하는 아날로그 비디오 신호로 변환하여 프로젝션 장치(PD_1 - PD_L)중 대응하는 한가지에 제공된다

일례로, DACs(178)는 아날로그 비디오 신호 대신에 디지털 픽셀 데이터로 출력하기 위해 무시되거나 완전히 생략될 수도 있다는 점에 주목하라. 이것은 프로젝션 장치(PD_1 - PD_L)가 디지털 기술(예컨대, LCD-형 디스플레이 또는 디지털 마이크로-미러 디스플레이)에 기초하였을 때 유용하다.

수퍼-샘플링 - - 도 4-5

도 4는 수퍼-샘플되지 않은 예에 대해여 가상 스크린 공간의 일부를 나타낸다. 점(dot)들은 샘플 장소를 나타내며, 가상 스크린 공간 상에 겹쳐져 놓여있는 사각형 박스는 픽셀 경계를 정의한다. 각 픽셀의 중심에 한개의 샘플이 배치되고, 적색, 녹색, 청색, z 등이 그 샘플을 위해 계산된다. 예를 들어, 샘플(74)는 픽셀(70)의 중심에 할당되어 있다. 비록 렌더링 유닛(150)이 픽셀당 하나의 샘플만을 계산하지만, 샘플-픽셀 연산유닛(170)은 예컨대, 여러 픽셀에 걸치는 콘볼루션 필터를 사용함으로써 다중 샘플에 기초하여 여전히 출력픽셀값을 계산할 수 있다.

이제 도 5A를 보면, 수퍼-샘플링의 일례가 나타나 있다. 이 실시예에서, 2개의 샘플이 샘플마다 계산된다. 샘플들이 정규 그리드에 따라 분포되어 있다. 도면에 픽셀보다 더 많은 샘플이 있다고 해도, 예컨대, 각 픽셀의 중심에 가장 가까운 샘플을 제외한 모든 샘플을 버림으로써, 출력픽셀값은 픽셀 마다 하나의 샘플을 사용하여 계산될 수 있다. 그러나, 다중 샘플에 기초하여 픽셀값을 계산하는 경우에 많은 이점이 생긴다.

지원영역(support region;72)이 픽셀(70) 위에 겹쳐져 있으며, 픽셀(70)에 배치된 필터의 지원을 나타낸다. 필터 서포트는 필터가 0이 아닌 값을 취하여 걸쳐있는 장소의 세트(즉, 필터 커널)이다. 이 예에서, 지원영역(72)은 원형 디스크 모양이다. 픽셀(70)을 위한 출력 픽셀값(예컨대, 적색, 녹색, 청색 및 z 값)은 오직 샘플(74A 및 74B)에 의해서만 결정된다. 왜냐하면, 이것들은 지원영역(72) 내부에 해당하는 유일한 샘플이기 때문이다. 이 여과 작용은 표현되는 이미지의 가파른 에지(edge)를 부드럽게 함으로써(즉, 안티-앨리어싱(anti-aliasing)을 행함), 표현되는 이미지의 현실성을 유리하게 증진시킬 수 있다. 필터작용은 출력 픽셀(70)의 최종값을 형성하기 위해 단순히 샘플(74A-B)을 평균하거나, 또는 (픽셀(70)의 중심에서) 샘플(74B)의 기여를 증가시키고 샘플(74A)(즉, 픽셀(70)의 중심로부터 멀리 떨어져 있는 샘플)의 기여를 감소시킬 수 있다. 필터(72), 즉 지원영역(72)은, 연산되고 있는 각 출력픽셀을 위해 재배치되어, 지원영역(72)의 중심은 연산되고 있는 픽셀의 중심 위치와 일치한다. 다른 필터 및 필터 위치도해가 또한 가능하며 숙고될 수 있다.

이제 도 5B를 보면, 수퍼-샘플링의 또 다른 실시예가 나타나 있다. 이 실시예에서 샘플들은 무작위로 배치되어 있다. 다라서, 출력픽셀을 연산하기 위해 사용되는 샘플의 수는 픽셀마다 다를 수 있다.

수퍼-샘플되는 샘플버퍼와 실시간 샘플-픽셀 연산 - - 도 6-13

도 6은 그래픽 시스템(112)의 일례를 통하여, 데이터의 흐름을 위해 가능한 하나의 구성을 나타낸다. 도면에서 나타낸 바와 같이, 기하 데이터(350)는 그래픽 시스템(112)에 의해 수신되고 드로어 프로세스(352)을 행하기 위해 사용된다. 드로어 프로세스(352)는 하나 이상의 제어 유닛(140), 렌더링 유닛(150), 메모리(152)와 스케쥴 유닛(154)에 의해 수행된다. 기하 데이터(350)는 하나 이상의 다각형을 위한 데이터를 포함한다. 각 다각형은 복수의 꼭지점(예컨대, 삼각형의 경우에는 3개의 꼭지점)을 포함하는데, 그 중 어떤 것은 공유될 수 있다. x, y와 z 좌표와 같은 데이터, 컬러 데이터, 라이팅(lighting) 데이터와 텍스쳐 맵 정보가 각 꼭지점을 위해 포함될 수 있다.

꼭지점 데이터에 더하여, (렌더링 유닛(150A-D)에 의해 행해질 수 있는) 드로어 프로세스(352)는 또한 샘플위치 메모리(354)로부터 샘플 좌표를 수신한다. 샘플위치 정보는 가상 스크린 공간, 즉 2-D 뷰포트에서 샘플의 장소를 정의한다. 드로어 프로세스(352)는 현재 렌더링되고 있는 다각형 내에 해당하는 샘플을 선별하고, 그 다각형 안의 각 위치에 따른 이 각각의 샘플을 위한 한 세트의 값(예컨대, 적색, 녹색, 청색, z, 알파, 및/또는 필드 정보의 깊이)를 연상한다. 샘플에 대해, 삼각형 내부에 해당하는 샘플의 z값은 3개의 꼭지점들의 알려진 z값으로부터 보간(補間)될 수 있다. 계산되는 샘플값의 각 세트는 샘플버퍼(162)에 저장된다.

일례로, 샘플위치 메모리(354)는 렌더링 유닛(150A-D)의 내부에 구체화될 수 있다. 또 다른 예로, 샘플위치 메모리(354)는 메모리(152A-152D)의 일부로서 또는 분리된 메모리로서 인식될 수 있다.

샘플위치 메모리(354)는 샘플위치를 그들의 가상 스크린 좌표(X,Y)의 견지에서 저장할 수 있다. 그와 달리, 샘플위치 메모리(354)는 샘플을 위한 오프셋dX 및 dY만을 정규 그리드상의 위치에 대하여 저장하도록 구성될 수 있다. 오프셋만을 저장하는 것은 각 샘플을 위한 전체 좌표(X,Y)를 저장하는 것보다 적은 저장 공간을 사용할 수 있다. 샘플위치 메모리(354)내에 저장된 샘플위치 정보는 주어진 샘플위치 연산유닛(도시되는 않았음)에 의해 읽혀질 수 있으며 그래픽 프로세서(90)를 위한 예시적인 샘플위치를 계산하도록 처리될 수 있다. 샘플위치 오프셋에 대한 더 상세한 정보는 다

음에 포함되어 있다(도 9와 10의 설명을 보라).

또 다른 실시예로, 샘플위치 메모리(354)는 난수의 테이블을 저장하도록 구성될 수 있다. 샘플위치 메모리(354)는 또한 하나 이상의 다른 유형의 정규 그리드를 생성하기 위해 주어진 하드웨어를 포함할 수도 있다. 이 하드웨어는 프로그램될 수 있다. 저장된 난수는 하드웨어에 의해 생성된 정규 그리드 위치를 향해 오프셋으로서 더해질 것이다. 일례로, 샘플위치 메모리는 난수 테이블을 액세스하도록 또는 펼치도록 많은 상이한 방법으로 프로그램될 수 있고, 그래서 주어진 길이의 난수 테이블을 위해 더욱 명확한 무작위성을 제공할 수 있다. 따라서, 샘플위치 오프셋의 단순한 반복에 의해 야기되는 시작적 아티팩트를 생성하지 않으면서, 보다 적은 테이블이 사용될 수 있다.

샘플-픽셀 연산유닛(360)은 드로어 프로세스(352)에서와 동일한 샘플위치를 사용한다. 따라서, 일례로, 샘플위치 메모리(354)는 드러어 프로세스(352)를 위한 샘플위치를 계산하기 위해 일련의 무작위의 오프셋 을 생성할 수 있고, 그 후에 샘플-픽셀 연산 프로세스(360)을 위해 동일한 샘플위치를 계산하기 위한 동일한 순서의 무작위 오프셋을 재생성할 수 있다.

도면에서 나타낸 바와 같이, 샘플위치 메모리(354)는 정규 사각 그리드, 정규 육각 그리드, 교란된 정규 그리드, 또는 무작위 (확률적) 분배와 같은 수많은 다른 설계에 따라 생성된 샘플 오프셋을 저장하도록 구성될 수 있다. 그래픽 시스템(112)은 운영 체제, 장치 드라이버, 또는 어떤 유형의 샘플위치도해가 이용될 것인지를 지시하는 기하 데이터(350)로부터의 지시를 받을 수 있다. 따라서, 샘플위치 메모리(354)는 하나 이상의 다른 설계에 다른 위치 정보를 생성하도록 구성되거나 프로그램 될 수 있다. 다양한 샘플위치도해에 관한 더욱 상세한 정보는 아래에서 더 설명될 것이다(도 8의 설명을 보라).

일례로, 샘플위치 메모리(354)는, 결정된 샘플 포인트 또는 샘플 오프셋을 확률적으로 담는 RAM/ROM을 포함할 수 있다. 따라서, 가상 스크린 공간의 샘플의 밀도는 작은 스케일에서 관찰되면 균일하지 않을 수 있다. 가상 스크린 공간 내에서 다른 장소에 중심을 가진 동일한 여역의 2개의 빈(bin)은 다른 갯수의 샘플을 가질 수 있다. 여기서 사용되는 바와 같이, 용어 " 빈(bin)" 이란 가상의 스크린 공간내의 지역 또는 영역을 말한다.

빈의 배열은 가상 스크린 공간 위에 겹쳐질 수 있다. 즉, 2-D 뷰포트 및 샘플버퍼(162)내의 샘플의 저장은 빈(bin)의 견지에서 조직화될 수 있다. 샘플버퍼(162)는 빈에 대응하는 메모리 블록의 배열을 포함한다. 각 메모리 블록은 대응하는 빈 내에 해당하는 샘플들을 위한 샘플값(예컨대, 적색, 녹색, 청색, z, 알파 등)을 저장할 수 있다. 메모리 블록은 가상 스크린 공간내의 대응하는 빈 장소로부터 쉽게 계산될 수 있고, 그 반대로 그 계산으로부터 장소를 알 수 있는 어드레스를 가질 수 있다. 따라서, 빈의 사용은 샘플버퍼(162)내의 샘플값의 저장 및 액세스를 단순화 할 수 있다.

빈(bin)은 정규 배열, 예컨대 정사각형 배열, 직사각형 배열, 삼각형 배열, 6각형 배열 또는 불류칙적 배열 등에서 2-D 뷰포트를 붙일 수 있다. 사이즈 및 모양은 프로그램될 수 있다. 한 빈에 존재할 수 있는 샘플의 최대 갯수는 대응하는 메모리 블록에 할당된 저장공간에 의해 결정된다. 이 샘플의 최대 갯수는 여기서 빈 샘플 용량이라고 불리우거나 또는 간단히 빈 용량(bin capacity)라고 불리운다. 빈 용량은 다양한 값의 어느것이라도 취할 수 있다. 빈 용량 값은 프로그램될 수 있다. 여기서, 가상 스크린 공간 내의 빈에 대응하는 샘플버퍼(162)내의 메모리 블록은 여기서 메모리 빈(memory bin)이라 불리울 것이다.

한 빈 안의 각 샘플의 특정 위치는 RAM/ROM 테이블내의 샘플의 오프셋, 즉, 빈 위치(예컨대, 빈의 왼쪽 아래 구석 또

는 중심 등)에 대한 샘플의 오프셋을 검색함으로써 결정될 수 있다. 그러나, 수행에 따라서, 빈 용량의 모든 선택이 RAM/ROM 테이블내에 저장된 유일한 세트의 오프셋을 가지는 것은 아니다. 제1 빈 용량값을 위한 오프셋은 제2의 보다 큰 용량 값을 위해 저장된 한 서브셋의 오프셋을 액세스함에 의해 결정될 수 있다. 일례로, 각 빈 용량 값은 적어도 4개의 상이한 샘플위치도해를 지원한다. 상이한 샘플위치도해를 사용하면 반복되는 샘플위치에 기인한 최종적인 이미지 아티팩트를 줄일 수 있다.

일례로, 샘플위치 메모리(354)는 각 쌍이 x-오프셋과 y-오프셋을 포함하는 수 쌍의 8-비트 숫자를 저장할 수 있다. (예컨대, 타임 오프셋, z-오프셋 등과 같이 다른 오프셋도 또한 가능하다.) 빈 위치에 더해졌을 때, 각 쌍은 가상 스크린 공간의 특별한 위치, 즉 2-D 뷰포트를 정의한다. 리드 액세스 시간(read access time)을 향상시키기 위해, 메모리가 리드 싸이클(read cycle) 당 하나 이상의 샘플 장소를 출력할 수 있도록, 샘플위치 메모리(354)는 넓은/병렬의 방식으로 구축될 수 있다.

일단 샘플위치가 샘플위치 메모리(354)로부터 읽혀지면, 드로어 프로세스(Draw Process;352)는 현재 렌더되고 있는 다각형 내에 해당하는 샘플을 선택한다. 그러면, 드로어 프로세스(352)는 이 각각의 샘플을 위해 (알파 또는 다른 깊이의 필드 정보를 포함할 수 있는) z 및 컬러 정보를 계산하고 데이터를 샘플버퍼(162)에 저장한다. 일례로, 컬러와 같은 그밖의 샘플 구성요소를 이중 버퍼링하는 동안 샘플버퍼는 z값(및 알파값)을 싱글-버퍼만 할 수도 있다. 종래의 시스템과 달리, 그래픽 시스템(112)은 (비록 모든 샘플 구성요소가 이중-버퍼링되는 것은 아니더라도, 즉, 샘플은 이중-버퍼링되지 않는 구성요소를 가질 수도 있다) 모든 샘플을 이중-버퍼링할 수도 있다. 일례로, 샘플들은 샘플버퍼(162)의 빈(bin)내에 저장된다. 어떤 실시예에서는, 빈의 용량은, 프레임마다 변화할 수 있다. 더욱이 빈 용량은 샘플버퍼(162)로 렌더링되는 단일의 프레임의 빈마다 공간적으로 변화할 수도 있다. 예를 들어, 2-D 뷰포트의 가장자리에 있는 빈(bin)은 2-D 뷰포트의 중심에 대응하는 빈보다 더 적은 비 용량을 가질 수 있다. 관찰자들은 스크린 SCR 또는 디스플레이 이미지 DIM의 중심에 그들의 대부분의 주의를 기울이기 때문에, 2-D 뷰포트의 중심에서 향상된 이미지 품질을 제공하기 위해 더 많은 처리대역폭이 제공될 수 있다. 빈의 크기와 모습은 또한 지역마다 다를 수 있거나, 또는 프레임마다 다를 수 있다. 빈의 이용은 아래에서 더 자세히 설명될 것이다.

드로어 프로세스(352)와 병행 및 독립적으로, 여과 프로세스(360)는 (a)샘플위치 메모리(354)로부터 샘플위치를 읽고,(b)샘플버퍼(162)로부터 대응 샘플값을 읽고,(c)샘플값을 여과하고,(d)결과적인 출력 픽셀값을 하나 이상의 프로젝션 장치(PD_1 - PD_L) 및/또는 디스플레이 장치(84)로 출력하도록 구성될 수 있다. 샘플-픽셀 연산유닛(170)은 여과 프로세스(360)를 수행한다. 여과 프로세스(360)는 선택된 복수의 샘플, 즉 픽셀 센터의 주위에 해당하는 샘플을 위한 대응하는 데이터의 공간적 여과에 기초하여 출력 픽셀을 위한 적색, 녹색, 및 청색 값을 생성하도록 작동될 수 있다. 알파와 같은 다른 값도 또한 생성될 수 있다. 일례로,여과 프로세스(360)는 : (i)각 샘플로부터, 여과되는 출력 픽셀의 중심까지의 거리를 결정하고; (ii)각 샘플의 특성값(예컨대, 적색, 녹색, 청색, 알파)을 상기 샘플의 거리의 특정한 (프로그램 가능한) 함수인 필터 중량과 곱하고; (iii)특성 당 하나의 합계(예컨대, 적색의 합계, 녹색의 합계 등)로서, 가중된 특성값의 합계를 생성하고(iv)대응하는 픽셀 특성값을 생성하기 위해 합계를 정규화하도록 구성되어 있다. 여과 프로세스(360)는 아래에 보다 자세히 설명되어 있다(도 11, 12와 14에 관련한 설명을 보아라).

설명된 실시예에서, 필터 커널은 픽셀 중심으로부터의 거리에 대한 함수이며, 따라서 방사상으로 대칭적으로. 그러나, 다른 실시예에서, 필터 커널은 픽셀 중심으로부터의 X 및 Y 의 변위에 대한 더 일반적인 함수이다. 따라서, 필터의 지원, 즉 필터 커널이 0이 아닌 값을 취하는 2-D 부근은 원형의 디스크가 아닐 수도 있다. 상기 필터 커널의 지원 내부에 해당하는 어떤 샘플이라도 계산되고 있는 출력 픽셀에 영향을 줄 수 있다.

도 7을 보면, 그래픽 시스템(112)의 다른 실시예를 나타내는 도표를 나타내고 있다. 이 실시예에서, 2 이상의 샘플위치 메모리(354A 및 354B)가 활용된다. 따라서, 샘플위치 메모리(354A-B)는 반드시 이중-버퍼(double-buffer)로 되어 있어야 한다. 샘플위치가 프레임마다 동일하게 유지되면, 샘플위치는 싱글 버퍼로 될 수도 있다. 그러나, 샘플위치

가 프레임마다 변화할 수 있으면, 그래픽 시스템(112)은 샘플위치를 이중-버퍼링하도록 유리하게 구성될 수 있다. 샘플위치는 렌더링 측에서 이중 버퍼로 될 수 있고(즉, 메모리354A는 이중 버퍼로 될 수 있다), 그리고/또는 필터(filter) 측(예컨대, 메모리354B는 이중 버퍼로 될 수 있다)에서 이중 버퍼로 될 수 있다. 다른 조합도 또한 가능하다. 예를 들어, 메모리(354A)는 싱글-버퍼로 될 수 있는 반면, 메모리(354B)는 이중 버퍼로 될 수 있다. 이렇게 구성하면, 한쪽 메모리(354B)가 리프레쉬(refresh)를 위해 사용되는 동안, 메모리(354B)의 다른 측은 여과 프로세스(360)를 위해 사용될 수 있다. 이 구성에서, 각 프레임이 렌더됨에 따라 메모리(354A)로부터 이중-버퍼링 되는 메모리(354B)로 샘플위치(또는 오프셋)를 이동시킴으로써, 그래픽 시스템(112)은 프레임 당 샘플위치도해를 바꿀 수 있다. 따라서, 메모리(354A)에 저장되고 샘플값을 렌더하기 위해 드로어 프로세스(352)에 의해 사용되는 샘플위치는, 여과 프로세스(360)에 의해 사용을 위해 메모리(354B)로 복제될 수 있다. 일단 위치 정보가 메모리(354B)로 복제되면, 위치 메모리(354A)는 렌더될 제2 프레임을 위해 사용되는 새로운 샘플위치(또는 오프셋)을 실을 수 있다. 이러한 방법으로, 샘플위치 정보는 드로어 프로세스(352)로부터 여과 프로세스(360)까지 샘플값을 따른다.

그러나 또 다른 실시예에서는, 수퍼-샘플되는 샘플버퍼(162)내의 샘플들에 대하여 오프셋에 태그(tag)를 저장할 수 있다. 이 태그는 각 특정한 샘플과 관련하여 오프셋(측, 교란) dX 및 dY를 검색하는데 사용될 수 있다.

샘플위치도해

도 8은 많은 상이한 샘플위치도해를 나타낸다. 정규 그리드 위치도해(190)에서, 샘플들은 2-D 뷰포트상에 겹쳐져 있는 정규 그리드에 관하여 고정된 위치에 배치된다. 예를 들어, 샘플들은 정규 그리드에 의해 생성된 사각형드르이 중심에 위치될 수 있다. 보다 일반적으로는, 어떠한 2-D 뷰포트의 타일붙이기라도 정규의 위치도해를 생성할 수 있다. 예를 들어, 2-D 뷰포트는 삼각형으로 타일붙여질 수 있고, 따라서 샘플은 삼각형 타일의 중심(또는 꼬지점들)에 위치될 수 있다. 6각형의 타일링, 대수적 타일링, 및 펜로즈 타일링(Penrose tiling)과 같은 세미-정규 타일링도 또한 숙고된다.

교란된 정규 그리드 위치도해(192)에서, 샘플위치는 정규 그리그 또는 타일링 상에서의 고정된 위치의 세트로부터의 교란의 견지에서 정의된다. 일례로, 샘플은 그들의 대응하는 고정 그리드 위치로부터 무작위의 x 및 y 오프셋만큼 또는 (0도부터 360도에 이르는) 무작위의 각도와 (0부터 최대 반지름에 이르는) 무작위 반지름만큼 이동될 수 있다. 오프셋은 예컨데, 작은 수의 시드(seed)에 기초한 하드위에의의해, 저장된 오프셋의 테이블을 읽음에 의해, 의사-난수 함수를 사용함에 의해 많은 방법으로 생성될 수 있다. 다시 말해서, 교란된 정규 그리드 도해(192)는 어떠한 유형의 정규 그리드 또는 타일링에라도 기초할 수 있다. 샘플들은 그리드에 관련한 교란에 의해 생성된다(예컨대, 이 구성의 기하도형적 특성 때문에 6각형의 타일링이 특히 바람직하다).

확률적 샘플위치도해(194)는 샘플을 위치하는 제3의 잠재적 설계 유형을 나타낸다. 확률적 샘플위치는 샘플을 2-D 뷰포트에 걸쳐 무작위로 분배하는 것을 포함한다. 샘플의 무작위 위치는, 예컨대, 의사 난수(擬似亂數)를 생성하는 내부 클럭과 같은 난수발생기를 사용함으로써, 수많은 다른 방법에 의해 이루어질 수 있다. 난수 또는 위치는 또한 미리 계산되어 메모리에 저장될 수도 있다. 이 응용에 사용되는 바와 같이, 무작위 위치가 통계적 집단(예컨대, 뿌아송-디스크 분배(Poisson-disk distribution)로부터 선별될 수 있다는 점에 유의하라. 다른 유형의 무작위 및 의사-랜덤 위치가 " Principles of Digital Image Synthesis" , Andrew S. Glassner 저(著), Morgan Kaufman Publishers, 1995 출판의 논문의 제1권의 10장에 보다 상세히 나타나 있다.

이제 도 9를 보면, 교란된 정규 그리드 설계(192)의 일례가 나타나 있다. 이 실시예에서, 샘플은 정규 정사각형 그리드로부터 x- 및 y- 오프셋만큼 무작위로 오프셋을 가지고 있다. 확대된 영역이 나타내는 바와 같이, 샘플(198)은 대응하는 그리드 교차점(196)으로부터의 수평 변위를 특징짓는 x- 오프셋(134)을 가지고 있다. 무작위 x-오프셋(134)과 y-오프셋(136)은 특별한 범위 값에 제한될 수 있다. 예를 들어, X_{max} 가 그리드 사각형의 폭일때, x-오프셋은 0부터 X_{max} 의 범위에 제한될 수 있다. 마찬가지로, Y_{max} 그리드 사각형의 높이일때, y-오프셋은 0부터 Y_{max} 의 범위에 제한될 수 있다. 무작위 오프셋이 또한 그리드 교차점(196)에 관련하여 각도와 반지름으로 특정화 될 수 있다.

도 10은 교란된 정규 그리드 설계(192)의 또 다른 실시예를 나타낸다. 이 실시예에서, 샘플들은 " 빈(bins)" (138A-D)으로 그룹되어 진다. 이 실시예에서, 각 빈은 9개의 샘플을 포함, 즉, 9의 빈 용량을 가진다. 다른 실시예에서, 다른 빈 용량이 사용될 수도 있다(예컨대, 4개의 샘플, 16개의 샘플을 저장하는 빈). 각 샘플의 위치는 그것이 거주하는 빈의 원점에 관련한 오프셋으로서 결정된다. 빈의 원점은 빈의 왼쪽 아래 구석으로 선택될 수 있다(또는 빈 내부의 다른 편리한 어느 장소라도 가능하다) 예를 들어, 샘플(198)의 위치는 x-오프셋(124)과 y-오프셋(126)을 각각 빈(138D)의 원점(132D)의 x 및 y 좌표에 더함으로써 결정된다. 위에서 설명한 바와 같이, 이렇게 하면 일부 실시예에서 사용되는 샘플위치 메모리의 사이즈를 줄일 수 있다.

도 11 - 샘플을 픽셀로 변환하기

앞에서 논의된 바와 같이, 2-D 뷰포트는 한 배열의 공간적 빈으로 덮일 수 있다. 각 공간적 빈은 위치가 샘플위치 메모리(354)에 의해 결정되는 샘플로 채워있을 수 있다. 각 공간적 빈은 샘플버퍼(162)안의 메모리 빈에 대응한다. 메모리 빈은 대응하는 공간 빈 내에 존재하는 샘플을 위해 샘플값(적색, 녹색, 청색, z, 알파 등)을 저장한다. 샘플-픽셀 연산유닛(170; 콘벌브 유닛(170)이라고도 호칭된다)은 샘플버퍼(162)로부터 메모리 빈을 읽고 메모리 빈 안에 담긴 샘플값을 픽셀값으로 변환하도록 구성된다.

컬럼을 사용한 병렬적 샘플-픽셀 여과 -- 도 11A-11B

도 11A는 샘플버퍼(162)에 저장된 샘플을 픽셀로 신속히 변환하는 한가지 가능방법을 나타내고 있다. 2-D 뷰포트를 덮는 공간적 빈은 컬럼들(예컨대, 컬럼1-4)로 조직화될 수 있다. 각 컬럼은 공간적 빈의 2차원적 서브-배열을 포함한다. 컬럼들은 (예컨대, 하나 이상의 빈의 의해) 수평적으로 오버랩되도록 구성될 수 있다. 각 샘플-픽셀 연산유닛(170-1 ~ 170-4)은 컬럼들 중 하나에 대응하는 메모리 빈에 액세스하도록 구성될 수 있다. 예를 들어, 샘플-픽셀 연산유닛(170-1)은 컬럼 1의 공간적 빈에 대응하는 메모리 빈에 액세스 하도록 구성될 수 있다. 샘플버퍼(162)와 샘플-픽셀 연산유닛(170) 사이의 데이터 통로는 이 컬럼-방식 통신을 지원하기 위해 최적화될 수 있다.

오버랩의 양은 사용되는 필터 커널을 위한 필터 서포트의 수평 직경에 의존할 수 있다. 도 11A에서 나타난 예는 두 개의 빈의 오버랩 (정사각형(188)과 같은) 은 하나 이상의 샘플을 포함하는 단일의 빈을 나타낸다. 유리하게, 이렇게 구성하면 샘플-픽셀 연산유닛(170)은, 독립적으로 그리고 병렬로 작업할 수 있도록 할 수 있으며, 각각의 샘플-픽셀 연산유닛(170)은 그 컬럼을 받고 변환한다. 컬럼을 오버랩하면, 한 픽셀의 범위보다 더 큰 연산자에 대하여, 시각적 띠 또는 다른 아티팩트를 컬럼 경계에서 나타나지 않도록 방지할 수 있다.

더욱이, 도 11A의 실시예는 샘플버퍼(162)에 결합된 복수의 빈 캐쉬(bin cache;176)를 포함한다. 또한, 각 빈 캐쉬(176)는 대응하는 하나의 샘플-픽셀 연산유닛(170)과 결합한다. 일반적인 빈 캐쉬(176-I; I는 양의 정수값 중 아무 값을 취함) 컬럼I에 대응하는 메모리 빈의 집합을 저장하고 샘플-픽셀 연산유닛(170-I)을 위한 캐쉬로서 동작한다.

일반적인 빈 캐쉬(176-I)는 컬럼I를 위한 메모리 빈으로의 액세스를 용이하게 하는 샘플버퍼(162)에 대한 최적화된 결합을 가질 수 있다. 2개의 인접한 출력픽셀을 위한 샘플-픽셀 연산은 많은 동일한 빈을 포함할 수 있으며, 빈 캐쉬(176)는 샘플버퍼(162)에 대한 전체 액세스 대역폭을 증가시킬 수 있다. 샘플-픽셀 연산유닛(170)은 고성능 ALU(산술적 논리 유닛) 코어, 마이크로 프로세서 또는 DSP로부터의 기능적 유닛, 또는 하드웨어 곱셈기 및 덧셈기를 사용하는 주문설계 등을 포함하여, 많은 다른 방법으로 수행될 수 있다.

이제 도 11B을 보면, 병렬의 샘플-픽셀 연산을 수행하는 또 다른 방법을 보이고 있다. 이 실시예에서, 샘플버퍼(162)는 앞에 설명된 예와 같이 복수의 수직 컬럼 또는 줄로 나뉘어져 있다. 그러나, 이 실시예에서의 컬럼은 동일한 사이즈나 폭이 아니다. 예를 들어, 컬럼 1은 컬럼 4보다 상당히 적은 갯수의 샘플 빈을 포함한다. 이 실시예는 가변적 샘플 밀도를 지원하는 그래픽 시스템의 구성에 특히 유용하다. 앞에서 언급한 바와 같고 아래에 보다 상세하게 설명되는 바와 같이, 그래픽 시스템은 보다 높은 샘플 밀도에서 가장 유익한 최종 이미지의 영역, 즉 관찰자의 망막와에 대응하는 이미지 영역 또는 관찰자의 특별한 관심 영역에 대응하는 샘플버퍼(162)의 영역을 위한 더 많은 샘플(즉, 보다 높은 샘플 밀도)에 전념할 수 있다. 가변 샘플 밀도를 지원하는 이 시스템에서 , 컬럼의 폭을 변화시키는 능력은 그래픽 시스템이 각 샘플-픽셀 연산유닛에 의해 여과되는 샘플의 갯수를 유리하게 균일화하도록 허용한다. 예를 들어, 컬럼 1은 관찰자의 관점의 중심이 집중되는 표현되는 이미지의 부분에 대응할 수 있다. 따라서, 그래픽 시스템은 빈의 컬럼 1에 높은 밀도의 샘플을 제공할 수 있고, 그래픽 시스템은 빈의 컬럼 4에 낮은 밀도의 샘플을 제공할 수 있다. 따라서, 컬럼 1의 폭을 감소시키고 컬럼 4의 폭을 증가시킴으로써, 샘플-픽셀 연산유닛(170-1 및 170-4)은 각각 대략 동일한 개수의 샘플을 여과할 수 있다. 유리하게는, 샘플-픽셀 연산유닛들 사이에서 여과 부담의 균형을 도모하는 것은 그래픽 시스템이 샘플-픽셀 연산유닛을 보다 효율적인 방식으로 처리 자원을 사용할 수 있도록 해 준다.

어떤 실시예에서, 그래픽 시스템은 프레임 대 프레임 기준(또는 프레임의 일부분 기준)에서의 컬럼의 폭을 유동적으로 변화하도록 구성될 수 있다. 샘플 밀도를 유동적으로 변화하는 그래픽 시스템의 실시예에서(예컨대, 눈-추적, 망막와 점 추적, 주인공 추적), 샘플 밀도는 프레임 대 프레임 기준에서 변화할 수 있고, 따라서 프레임 대 프레임 기준에서 컬러 폭을 변화시키는 것은 샘플-픽셀 연산유닛(170)의 계산 자원이 보다 효율적인 방식으로 활용될 수 있도록 한다. 어떤 실시예에서, 컬럼 폭은 스캔 라인 기준상에서 또는 어떤 다른 시간-기준에서 변화할 수 있다. 시간에 따라 변화함에 더하여, 도면이 나타내는 바와 같이, 컬럼은 또한 시각적 아티팩트(예컨대, 갈라진 틈, 찢어짐, 또는 수직선)가 나타남을 방지하기 위해 (앞에서 설명된 실시예에서와 같이) 오버랩되도록 구성될 수 있다.

행(Rows)을 사용하는 병렬적 샘플-픽셀 여과 - - 도 12

이제 도 12를 보면, 그래픽 시스템의 또 다른 실시예가 나타나 있다. 이 실시예에서, 샘플버퍼(162)는 복수의 수평적 행 또는 줄로 나뉘어져 있다. 앞의 실시예에서와 같이, 행은 변화하는 샘플 밀도를 보충하기 위해 폭에 있어서 오버랩 및/또는 변화할 수 있다. 앞의 실시예에서와 같이, 각 행은 특별한 빈 캐쉬(176) 및 대응하는 샘플-픽셀 연산유닛(170)에 빈(및 샘플)을 제공할 수 있다.

행(Rows)을 사용하는 병렬적 샘플-픽셀 여과 - - 도 13

이제 도 13을 보면, 그래픽 시스템의 또 다른 실시예가 나타나 있다. 이 실시예에서, 샘플버퍼(162)는 복수의 직사각형 지역으로 나뉘어진다. 앞의 실시예에서와 같이, 직사각형 지역은 (예컨대, 프레임 대 프레임 또는 스캔라인 기준에서) 오버랩하거나, 상이한 사이즈를 가지거나, 그리고/또는 유동적으로 사이즈에 있어서 변화할 수 있거나 그렇지 않을 수도 있다. 각 지역은 특별한 빈 캐쉬(176) 및 대응하는 샘플-픽셀 연산유닛(170)에 빈(및 샘플을) 제공하도록 구성될 수 있다. 어떤 실시예에서, 각 직사각형 지역은 복수의 프로젝터(예컨대, LCD 프로젝터)의 하나에 의해 투영되는 이미지에 대응할 수 있다. 다른 실시예에서, 각 직사각형 지역은 단일의 디스플레이 장치에 표시되거나 투영되는 단일의 이미지의 특별한 부분에 대응할 수 있다. 앞의 실시예와 같이, 유익하게는, 샘플-픽셀 연산유닛(170)은 독립적으로, 병

렬로 작동하여서, 그래픽 시스템의 대기시간을 줄이도록 구성될 수 있다. 앞에서 언급한 바와 같이, 도 13에 나타내어진 직사각형 지역은 균일한 사이즈 및/또는 모양일 필요가 없다.

변화하는 지역의 사이즈 또는 줄의 폭을 가지는 그래픽 시스템의 예에서, 오버랩의 양은 프레임 대 프레임 또는 서브-프레임 기준에서 유동적으로 변화할 수 있다. 샘플버퍼(162)가 나뉘어질 수 있는 다른 모양의 지역도 가능하며 숙고될 수 있다. 예를 들어, 일부 실시예에서 각 샘플-픽셀 연산유닛은 복수의 작은 지역 또는 줄로부터 빈(및 샘플)을 받을 수 있다.

일부 실시예에서, 샘플캐쉬(176)는 전체적인 수평적 스캔라인을 저장하기 위한 충분한 저장공간을 가질 수 있다. 이 이유때문에, 샘플버퍼를 지역으로 나누는 것은 유용할 것이다. 디스플레이 장치에 따라서, 지역들은 홀수만의 및 짝수만의 스캔라인의 일부분이 될 수 있다. 어떤 시스템, 예컨대 다중 디스플레이 장치를 가진 시스템에서, 각 지역은 단일의 디스플레이 장치 또는 디스플레이되고 있는 이미지의 사분면에 대응할 수 있다. 예를 들어, 4개의 프로젝터에 의해 형성된 이미지가 단일의, 큰 이미지를 형성하도록 함께 붙여진다고 가정하면, 각 샘플-픽셀 연산유닛은 특별한 프로젝터에 의해 표시되는 픽셀에 대응하는 샘플을 받을 수 잇을 것이다. 일부 실시예에서, 오랩되는 역의 지역은 두번 저장될 수 있고, 그에 의해 각 샘플-픽셀 연산유닛은 샘플버퍼의 특별한 영역으로의 배타적 액세스가 가능하다. 이것은 두개의 상이한 샘플-픽셀 연산유닛(또는 2개의 샘플캐쉬 제어기)이 동시에 동일한 세트의 메모리 장소에 액세스하는 것을 시도할 때 나타나는 시간 문제(timing problems)를 방지할 수 있다. 다른 실시예에서, 동일한 메모리 장소로의 하나 이상의 다중의 동시발생의 액세스를 가능하도록, 샘플버퍼는 다중-포트로 될 수 있다.

앞에서 언급한 바와 같이, 일부 실시예에서 샘플캐쉬는 샘플버퍼로부터 샘플을 읽도록 구성된다. 일부 실시예에서, 샘플은 샘플버퍼로부터 빈-대-빈 기준에서 읽혀질 수 있다. 샘플캐쉬 및/또는 샘플버퍼는, 여과되고 있는 (또는 여과되려고 하는) 하나 이상의 픽셀에 기여할 잠재성이 있는 모든 샘플이 대응 샘플-픽셀 연산유닛을 위해 사용될 수 있도록 구성되는 제어 로직(control logic)을 포함할 수 있다. 일부 수행에서, 샘플캐쉬는 (최대 필터 사이즈에 조화하도록) 5x5 빈과 같은 소정의 배열의 빈을 저장하기에 충분히 클 수 있다. 또다른 실시예에서, 5x5 빈 캐쉬 대신에, 샘플 캐쉬는 일련의 다중 적산기에 축적되면서 픽셀을 출력하도록 구성될 수 있다. 이 실시예에서, 샘플 및 그들의 중량의 숫자에 따라서, 상이한 계수가 각 픽셀을 위해 생성된다.

샘플버퍼로부터 샘플을 읽는 방법 -- 도 14

이제 도 14를 보면, 수퍼-샘플되는 샘플버퍼로부터의 샘플값을 읽는 방법의 일례를 보다 상세하게 나타내고 있다. 도면에서 나타내는 바와 같이, 콘볼루션 필터 커널(kernel;400)이 출력 픽셀을 생성하기 위해 컬럼 I을 (화살표 406의 방향으로) 가로질러 가며, 여기서 인덱스 I는 0부터 4까지의 범위에 있는 아무 값을 취할 수 있다. 샘플-픽셀 연산유닛(170-I)은 샘플-픽셀 필터 커널(400)을 수행할 수 있다. 빈 캐시(bin cache;176-I)는 컬럼 I에 대응하는 메모리 빈에 대한 신속한 액세스를 제공하기 위해 사용될 수 있다. 예를 들어, 샘플-픽셀 필터 커널(400)이 5 대 5 배열의 공간적 빈을 덮기 때문에, 빈 캐쉬(176-I)는 25 메모리 빈과 동일하거나 큰 용량을 가질 수 있다. 샘플을 픽셀로 변환하는 작용이 진행함에 따라, 메모리 빈은 수퍼-샘플되는 샘플버퍼(162)로부터 읽히고 빈 캐시(176-I)에 저장된다. 일례로, 더 이상 필요없는 빈(410), 예컨대 빈(410)은 새로운 빈에 의해 빈 캐시(176-I)에 덮어씌워진다. 각 출력픽셀이 생성됨에 따라, 샘플-픽셀 필터 커널(400)이 이동한다. 커널(400)은 컬럼 I 내에서 순차적 방식으로 화살표(406)로 지시된 방향으로 진행하는 것으로 구상화될 수 있다. 커널(400)이 컬럼 I의 오른쪽 경계(404) 컬럼의 끝에 도달할 때, 하나 이상의 행 아래의 빈으로 이동하며, 다시 왼쪽 컬럼 경계(402)로부터 수평적으로 진행하기 시작한다. 따라서, 샘플-픽셀 연산유닛은 주사선 방식으로 진행하며, 디스플레이를 위한 연속적인 컬럼의 출력 픽셀을 생성한다.

도 15는 출력픽셀값의 계산에 있어서 잠재적 경계조건을 나타낸다. 2-D 뷰포트(420)은 공간적 빈의 직사각형 배열로 덮힌 직사각형으로 나타나 있다. 모든 공간적 빈은 샘플버퍼(162)내의 메모리 빈에 대응하는 점을 상기하라. 메모리 빈은 샘플값 및/또는 대응하는 공간적 빈 안에 존재하는 샘플을 위한 샘플위치를 저장한다. 위에서 설명한 바와 같이, 샘플-픽셀 연산유닛(170)은 출력픽셀값(예컨대, 적색, 녹색, 청색 등)을 생성하기 위해 픽셀 중심의 주변에서 샘플을 여과한다. 픽셀 중심(PC_0)은, 필터 서포트(400)가 2-D 뷰포트 내에 전적으로 담겨있지는 않은 2-D 뷰포트(420)의 낮은 측 경계(Y=0)에 충분히 가깝다. 샘플-픽셀 연산유닛(170)은 다양한 방법에 의해 샘플위치 및/또는 필터 서포트(400)의 가장자리 부분(즉, 2-D 뷰포트(420)의 밖에 해당하는 부분)을 위한 샘플값을 생성할 수 있다.

일례로, 샘플-픽셀 연산유닛(170)은 필터 서포트(400)의 가장자리를 커버하기 위한 하나 이상의 더미 빈(dummy bin)을 생성할 수 있다. 더미 빈을 위한 샘플위치는 2-D 뷰포트 경계를 가로질러 공간 빈의 샘플위치를 반사함에 의해 생성된다. 예를 들어, 더미 빈(F,G,H,I,J)은 각각의 공간적 빈(A,B,C,D,E)에 대응하여 샘플위치를 경계선 Y=0을 가로질러 반사함에 의해 할당된 샘플위치일 수 있다. 소정의 컬러값은 더미 빈 내부의 더미 샘플과 연관될 수 있다. 예를 들어, RGB 컬러 벡터를 위한 값(0,0,0)은 각 더미 샘플에 할당될 수 있다. 픽셀 중심(PC_0)이 아래쪽으로 이동함에 따라(즉, 경계 Y=0를 향해 그것을 통과하여), 더미 샘플을 가진 추가적 더미 빈이 생성되어 (픽셀중심(PC_0)을 따라서 이동하는) 필터 서포트(400)를 커버할 수 있다. 필터 서포트(400) 내에 해당하는 더미 빈의 숫자는 증가하고, 필터 서포트(400)가 2-D 뷰포트(420)의 외부로 전적으로 이동하였을 때 최대에 도달한다. 따라서, 필터 서포트(400)에 기초하여 계산되는 필터값은, 픽셀중심(PC_0)이 경계를 가로지름에 따라, 소정의 배경색에 근접한다.

픽셀중심은 2-D 뷰포트(420)의 외부에 놓여 질 수도 있으며, 게다가, 필터 서포트의 일부가 2-D 뷰포트(420) 내에 놓여질 수 있도록 뷰포트 경계에 충분히 가까울 수 있다. 필터 서포트(401)는 하나의 그러한 픽셀중심에 대응한다. 샘플-픽셀 연산유닛(170)은 필터 서포트(401)의 외부 부분을 덮기 위해(즉, 2-D 뷰포트의 외부 부분) 더미 빈(Q,R,S,T,U,V)를 생성할 수 있다. 더미 빈(Q,R,S)은 공간적 빈(N,O,P) 및/또는 공간적 빈(K,L,M)의 샘플위치에 기초하여 할당된 샘플위치일 수 있다.

더미 빈을 위한 샘플위치는 또한 뷰포트 경계를 가로질러 공간적 빈에 대응하여 샘플위치를 전환함에 의하거나 또는, 정규의, 교란된 정규의 또는 확률적 샘플위치도해에 따라 온 더 플라이(on-the-fly)로 샘플위치를 생성함에 의하여 생성될 수 있다.

도 16은 픽셀값 계산을 수행하기 위한 방법의 다른 실시예를 나타낸다. 샘플-픽셀 연산유닛(170)은 2-D 뷰포트(420)의 가시적 서브윈도우(422)를 사용하여 픽셀값 계산을 수행할 수 있다. 가시적 서브윈도우는 가상 스크린 공간에서 아래 왼쪽 구석(X_1,Y_1) 및 위쪽 오른쪽 구석(X_2,Y_2)을 가진 직사각형으로 묘사된다. 어떤 실시예에서, 최종 픽셀값을 결정하기 위해 요구되는 연산의 갯수를 줄이기 위하여, 필터는 자동-정규화되거나 사전에-정규화될 수 있음에 유의하라.

샘플 픽셀을 수퍼-샘플되는 샘플버퍼로 렌더링하기 -- 도 17

도 17은 샘플 픽셀을 수퍼-샘플되는 샘플버퍼로 드로어(drawing) 또는 렌더링(rendering)하는 방법의 일례에 대한 흐름도이다. 도 12의 어떤 단계는 동시에 또는 다른 순서로 발생할 수 있다. 단계 200에서, 그래픽 시스템은 호스트 CPU(102)나 메인 메모리(106)로부터 직접 그래픽 명령과 그래픽 데이터를 수신한다. 단계 202에서, 그 명령과 데이터는 하나 이상의 렌더링 유닛(150A-D)으로 발송된다. 단계 204에서, 렌더링 유닛(150A-D)은 그래픽 데이터가 압축되었는지 판단한다. 만약 그래픽 데이터가 압축된 것이면, 렌더링 유닛(150A-D)은 데이터를 사용가능한 포맷, 단계 206에 나타난 바와 같이, 예컨대 삼각형으로 압축을 푼다. 다음에, 삼각형은 예컨대, 모델 공간에서부터 세계 공간으로 변환되고, 라이팅되고, 변환되는 등의 과정을 거친다(단계 208A).

만약 그래픽 시스템이 가변 해상도 수퍼 샘플링을 수행하면, 삼각형은 한 세트의 샘플 밀도 영역 경계(단계 208B)와

비교된다. 가변-해상도 수퍼-샘플링에서, 2-D 뷰포트의 다른 영역은 수많은 인자(예컨대, 눈이나 머리 추적에 의해 판단되는 관찰자의 프로젝션 스크린 SCR에 대한 관심의 중심)에 기초하여 할당된 다른 샘플 밀도일 수 있다. 샘플 밀도 영역은 아래에서 더욱 상세히 설명된다(아래의 가변 해상도 샘플버퍼라는 제목의 항목을 보라). 삼각형이 샘플-밀도 영역 경계(단계 210)를 가로지르면, 삼각형은 영역 경계를 따라 2개의 보다 작은 다각형으로 나뉘어 질 수 있다(단계 212). (삼각형을 일반적으로 쪼개면 삼각형과 사각형이 만들어지므로)다각형은 필요할 경우 삼각형으로 또다시 나뉘어질 수 있다. 일례로, 그래픽 시스템(112)은 원래의 삼각형을 두 번, 즉, 각 샘플밀도마다 한 번 렌더링하고 나서, 두개의 버전을 잘라서 두개의 각각의 샘플 밀도 영역으로 나누도록 구성될 수 있다.

단계 214에서, (예컨대, 정규 그리드, 교란된 정규 그리드, 또는 확률적) 샘플위치도해 중 하나가 샘플위치 메모리(354)로부터 선택된다. 샘플위치도해는 샘플위치 메모리(354)로 미리 프로그램되어 있는 것이 일반적이지만, 또한 " 온 더 플라이(on the fly)" 로 선택될 수도 있다. 단계216에서, 선택된 샘플위치도해와 공간적 빈의 사이즈 및 모습에 기초하여, 렌더링 유닛(150A-D)은 어느 빈이 삼각형의 영역 내에 배치된 샘플을 담을 것인가를 결정한다. 단계 218에서, 이들 빈(bins) 내의 샘플을 위한 오프셋(dX 및 dY)은 샘플위치 메모리(354)로부터 읽혀진다. 단계 220에서, 각 샘플의 위치는 오프셋(dX 및 dY)과 대응하는 빈 원점의 좌표를 이용하여 계산되며, 샘플이 삼각형 내부에 있는지를 판단하기 위해 삼각형의 꼭지점들과 비교된다. 단계 220은 아래에서 더욱 자세히 논의된다.

삼각형 내부에 있는 것으로 판단된 각 샘플에 대하여, 렌더링 유닛은 샘플의 컬러, 알파 및 다른 특성을 계산하는 것에 의해 샘플을 드로어(draw)한다. 이는 삼각형의 꼭지점들과 관련한 텍스쳐 맵 정보와 컬러에 기초한, 보간법(補間法;interplation)과 라이팅 연산을 포함할 수 있다. 일단 샘플이 렌더되면, 스케쥴 유닛(154)으로 전송되는데, 그러면 스케쥴 유닛(154)은 샘플을 샘플버퍼(162)에 저장한다(단계 224).

위에서 설명된 렌더링 방법의 실시예는 설명의 목적으로만 이용되는 것이며, 그에 제한하고자 하는 것이 아니다. 예를 들어, 몇가지 실시예에서, 도 13에서 직렬적으로 발생하는 것으로 나타난 단계들은 병렬적으로 수행될 수도 있다. 더욱이, 그래픽 시스템의 어떤 실시예에서, 몇 가지 단계들(예컨대, 기하 압축을 수행하지 않는 실시예에서 단계(204-206), 또는 가변 해상도 수퍼-샘플되는 샘플버퍼를 수행하지 않는 실시예에서 단계(201-212))은 축소되거나 삭제될 수도 있다.

어느 샘플이 렌더링되고 있는 다각형 내부에 존재하는가를 결정하기 -- 도18

어느 샘플이 렌더링되고 있는 다각형 내부에 존재하는가를 비교하는 것은 수많은 다른 방법으로 행해질 수 있다. 일례로, 우선, 삼각형을 정의하는 3개의 꼭지점간 델타(delta)가 결정된다. 예를 들어, 이 델타는 첫 번째 꼭지점부터 두 번째 꼭지점(v2-v1)=d12, 두 번째 꼭지점부터 세 번째 꼭지점(v3-v2)=d23, 그리고 3번째 꼭지점으로부터 다시 첫 번째 꼭지점(v1-v3)=d31의 순서로 취해진다. 이 델타들은 벡터를 형성하고, 각 벡터는 (예컨대, 델타 X 계수 및 델타 Y 계수의 두 개의 부호 비트를 사용함으로써) 좌표 평면의 4개의 사분면 중 하나에 속하는 것으로 분류될 수 있다. 벡터가 X-메이저 벡터인지 또는 Y-메이저 벡터인지를 결정하는 제3의 조건이 부가될 수 있다. 이것은 abs(delta_x) 값이 abs(delta_y)값보다 큰지 여부를 계산하는 것에 의해 결정될 수 있다. 이 3비트의 정보를 사용함으로써, 벡터는 각각 좌표 평면의 8개의 다른 영역 중 하나에 속하는 것으로 분류될 수 있다. 이 영역을 정의하기 위해 3비트가 사용되면, (2만큼 왼쪽으로 이동한) X-부호 비트, (1만큼 왼쪽으로 이동한) Y-부호 비트, 그리고 X-메이저 비트가 도 18에 나타낸 바와 같이 8개 영역을 생성하기 위해 사용될 수 있다.

다음으로, 삼각형의 내부 부분을 정의하기 위해 3개의 에지(edge) 방정식이 이용될 수 있다. 이 에지들은, rm=1에서, y=mx+b 또는 x=ry+c 형식의 어느 하나(또는 양자 모두)의 선으로 표현될 수 있다. 기울기를 표현하기 위해 필요한

수의 범위를 줄이기 위해, (기울기의 절대값이 0부터 1의 범위 내에 있도록) 에지 방정식을 위해 두 X-메이저 및 Y-메이저 방정식 형식이 사용될 수 있다. 따라서, 에지(또는 반-평면) 부등식은 2개의 대응하는 다음의 형식 중 하나로 표현될 수 있다:

X-메이저: y-m·x-b < 0, 포인트(x,y)가 에지의 아래에 있을 경우;

Y-메이저: x-r·y-c < 0, 포인트(x,y)가 에지의 왼쪽에 있을 경우;

(x,y)의 포인트가 에지에 의해 정의되는 선 아래에 있을 경우 X-메이저 부등식은 논리적인 참값(즉, 1과 동일한 부호 비트)을 만든다. (x,y)의 포인트가 에지에 의해 정의되는 선의 왼쪽에 있을 경우 Y-메이저 방정식은 논리적 참값을 만든다. 삼각형의 내부를 포함하는 측은 각각의 선형 부등식을 위해 열려져 있고, 여기서 억셉트 비트로 호칭되는 불린 변수(Boolean variable)에 의해 특정될 수 있다. 따라서, 다음과 같으면 샘플(x,y)은 에지의 내부측에 있다.

X-메이저: (y-m·x-b< 0)< xor> accept = true;

Y-메이저: (x-m·y-b< 0)< xor> accept = true;

주어진 에지를 위한 억셉트 비트는, (a)에지 델타 벡터가 존재하는 영역(0부터 7까지) 및 (b)시계방향의 횡단이 cw=1이고 반-시계방향의 횡단이 cw=0일때의 에지 횡단의 직감에 기초하여, 다음의 테이블에 따라서 계산될 수 있다. 기호 " !" 는 논리 보수(logical complement).

1:억셉트=!cw

0:억셉트=cw

4:억셉트=cw

5:억셉트=cw

7:억셉트=cw

6:억셉트=!cw

2:억셉트=!cw

3:억셉트=!cw

이 표현 규칙을 깨는 것도 또한 행해진다(예컨대, 좌표축이 양의 8분공간에 속하는 것으로 정의될 수 있다). 마찬가지로, X-메이저는 기울기 상에서 균형을 이루는 모든 포인트를 가지는 것으로 정의될 수 있다.

다른 실시예로서, 에지의 억셉트 측은 에지 부등식을 삼각형의 셋째 꼭지점(즉, 에지를 형성하는 2개의 꼭지점 중 어느 것도 아닌 꼭지점)에 적용하는 것으로써 결정될 수 있다. 이 방법은 큰 추가적 비용을 초래할 수 있으며, 위에서 설명된 기술에 의해서 회피될 수 있다.

삼각형의 " 지향성(faced-ness)" 을 결정하기 위하여(즉, 삼각형이 시계방향 또는 반-시계방향인지 결정하기 위하여), 삼각형의 두 에지의 델타-방향(delta-direction)이 검사되고 두 에지의 기울기가 비교될 수 있다. 예를 들어, 에지(12)가 1의 델타-방향을 가지고 두 번째 에지(에지(23))가 0, 4, 또는 5의 델타-방향을 가진다고 가정하면, 삼각형은 반시계방향이다. 그러나, 만약 에지(23)가 3, 2, 또는 6의 델타-방향을 가지면, 삼각형은 시계방향이다. 만약 에지(2

3)가 1의 델타-방향을 가지면(즉, 에지(12)와 같이), 두 에지의 기울기를 비교하면 균형이 깨진다(양자 모두 x-메이저이다). 에지(12)가 보다 큰 기울기를 가지면, 삼각형은 시계방향이다. 에지(23)가 (에지(12)와 정 반대인) 7의 델타-방향을 가지면, 삼각형이 시계방향인지 또는 반시계방향인지의 견지에서 기울기는 반대 결과와 비교된다.

적절한 지향성을 판단하는 모든 경우에서, 에지(12) 및 에지(23) 델타-방향(delta-direction)의 모든 조합에 동일한 분석이 철저하게 적용될 수 있다. 균형을 이루는 경우에서 기울기가 동일하면, 삼각형은 변질된다(즉, 내부 공간이 없게 됨). 그것은 명백하게 검사될 수 있고 가려낼 수 있으며, 또는, 수에 관해 적절히 주의를 기울이면, 아무 픽셀도 렌더링하지 않으므로 묵과될 수 있을 것이다. 삼각형이 뷰 평면(view plane)을 쪼개는 것은 한가지 특별한 경우이지만, 그것은 (예컨대, 전면과 후면 클리핑이 수행될 때) 파이프라인(pipeline)에서 먼저 감지될 수 있을 것이다.

대부분의 경우에서, 삼각형의 한쪽만이 렌더링된다는 점에 주목하라. 따라서, 상기 분색에 의해 결정된 삼각형의 지향성이 거절될 것이라면, 삼각형은 가려내어 질 수 있다(즉, 샘플의 생성없이 더 이상의 처리 없음을 가정할 수 있다). 이 지향성 결정은 이미 계산된 인자를 넘어서 (즉, 에지(12) 기울기와 에지(23) 기울기와의) 하나의 추가적인 비교를 이용만 하는 점에 주목하라. 많은 종래의 접근방법은 (이전의 셋-업 계산 단계를 통하여) 보다 복잡한 계산을 활용할 수 있게 한다.

샘플값으로부터 출력 픽셀값을 생성하기 -- 도 19

도 19은 출력 픽셀을 생성하기 위하여, 수퍼-샘플되는 샘플버퍼(162)에 저장되어 있는 샘플을 선택하고 여과하는 방법의 일례에 대한 흐름도이다. 단계 250에서, 한 스트림의 메모리 빈(bins)이 수퍼-샘플되는 샘플버퍼(162)로부터 읽힌다. 단계 252에서, 샘플-픽셀 연산유닛(170)이 콘볼루션 프로세서 동안 쉽게 억세스할 수 있게 하기 위해 하나 이상의 캐시(176)에 저장될 수 있다(단계 252). 단계 254에서, 메모리 빈(bins)은 현재 생성되고 있는, 출력 픽셀에 기여하는 샘플을 담고 있는지 판단하기 위해 검사받는다(254). 그러면, 출력 픽셀에 기여할 수 있는 빈(bin) 안에 있는 각 샘플은, 그 샘플이 실제로 공헌을 하는지 판단하기 위해 개별적으로 검사받는다(단계 256-258). 이러한 판단은, 그 샘플과 생성되는 출력 픽셀의 중심사이의 거리에 기초하여 할 수 있다.

일례로, 샘플-픽셀 연산유닛(170)은 이 거리(즉, 샘플위치에서의 필터의 범위 또는 포락선)를 계산한 후 그것을 사용하여 필터 범위에 따라 필터 가중치를 저장하고 있는 테이블에 색인을 달도록 구성될 수 있다(단계 260). 그러나, 또 다른 실시예로, (전형적으로 제곱근 함수를 포함하는) 픽셀의 중심로부터 샘플까지의 거리를 결정하기 위한 잠재적으로 고비용의 계산은, 필터 중량 테이블(table of filter weights)의 인덱스와 일치하는 거리를 사용함으로써 피할 수 있다. 다른 방법으로는, x 및 y의 함수는 거리 계산에 의존하는 함수 대신에 사용할 수 있다. 일례로, 이것은 거리의 부동소수점 포맷(예컨대, 4 또는 5 비트의 가수(假數)와 3 비트의 지수)을 활용함으로써 이루어 질 수 있으며, 그에 의해 값의 증가된 범위를 보상하면서도 많은 정확성이 유지될 수 있다. 일례로, 테이블은 ROM에서 실행될 수 있다. 그러나, RAM 테이블 또한 사용될 수 있다. 유리하게는, RAM 테이블을 사용하면, 몇가지 실시예에서, 그래픽 시스템이 프레임마다(per-frame basis) 필터 계수를 변화시킬 수 있다. 예를 들어, 필터 계수는 디스플레이의 알려진 결핍을 보상하거나 또는 사용자의 개인적 기호를 위해 변화될 수 있다. 일부 실시예에서, RAM 테이블을 사용하면 사용자가 상이한 필터를 선별할 수 있도록 한다(예컨대, 디스플레이 장치상의 샤프니스(sharpness) 제어 또는 윈도우 시스템의 제어판을 통하여). 수많은 상이한 필터가 상이한 디스플레이 형태에 따라 희망되는 레벨의 샤프니스를 생성하도록 수행될 수 있을 것이다. 예를 들어, 제어판은 LCD 디스플레이의 최적화된 하나의 세팅과 CRT 디스플레이를 위해 최적화된 또다른 세팅을 가질 수 있다. 그래픽 시스템은 또한, 프레임 내의 스크린 영역마다(screen area basis)의 필터 계수 또는 출력 픽셀 마다(per-output pixel basis)에서의 필터 계수를 변화시킬 수 있다. 또 다른 실시예로, 특수화된 하드웨어(예컨대, 곱셈기와 덧셈기)를 사용하여 각 샘플을 위한 희망하는 필터 중량을 실제로 계산할 수 있다. 콘볼루션 필터의

한계 밖의 샘플을 위한 필터 중량(filter weight)은 0의 필터 중량으로 단순히 곱해질 수 있으며(단계 62), 또는 그것들은 연산에서 완전히 제거될 수도 있다.

일단 샘플을 위한 필터 중량이 결정되면, 샘플은 필터 중량으로 곱해질 수 있다(단계 264). 그러면 가중된 샘플은 최종 출력 픽셀의 컬러값을 결정하기 위하여 현재 총계에 더해질 수 있다(266). 필터 중량은 또한 현재 총 픽셀 필터 중량에 더해질 수 있으며(단계 268), 이것은 여과된 픽셀을 정규화하는데 사용된다. 정규화는 콘볼루션 프로세스에 의해 만들어진 게인(gain)을 보상함에 의해 여과된 픽셀(예컨대, 다른 픽셀보다 더 많은 샘플을 가진 픽셀)이 너무 밝거나 또는 너무 어둡게 나타나는 것을 유익하게 방지한다. 모든 기여하는 샘플이 가중되고 더해진 후에, 전체 픽셀 필터 중량은 여과에 의하여 게인(gain)을 나누기 위해 사용될 수 있다(단계 270). 마지막으로, 정규화된 출력 픽셀은, 최종적으로 디스플레이 되기 전에, 다음의 프로세스(반드시 이 순서대로일 필요는 없음) 중의 하나 이상을 통하여 출력되며 그리고/또는 처리될 수 있다: 감마 수정(gamma correction), 의사 컬러테이블을 이용한 컬러 검색, 다이렉트 컬러, 인버스 감마 수정(inverse gamma correction), 프로그래머블 감마 인코딩(programmable gamma encoding), 컬러 공간 변환(color space conversion), 디지털-아날로그 변환(단계 274).

일부 실시예에서, 그래픽 시스템은 샘플과 함께 출력되는 마스크(mask)를 생성하기 위해 각 샘플의 알파 정보를 사용하도록 구성될 수 있다. 마스크는 실시간(real-time)의 부드러운 가장자리의 청색 스크린 효과를 수행하기 위해 사용된다. 예를 들어, 마스크는 렌더링된 이미지의 어느 부분이 (그리고 얼마나 많이) 마스크되어야 하는지 지시하는데 사용될 수 있다. 이 마스크는, (두 이미지의 겹치는 영역에 관하여 앤티-앨리어스(anti-aliased)된) 부드러운 청색 스크린 효과 또는 유령 효과(예컨대, 부분적으로 투명한 물체를 부드럽게 다른 물체,장면, 또는 비디오 스트림에 겹치게 함) 렌더링된 이미지를 다른 이미지(예컨대, 비디오 카메라로부터의 신호)와 혼합(blend)하기 위해 그래픽 시스템 또는 외부적 하드웨어에 의해 사용될 수 있을 것이다.

출력 픽셀 콘볼루션의 예 -- 도 20

도 20은 출력 픽셀 콘볼루션의 단순화된 예를 나타내고 있다. 도면에서 나타내는 바와 같이, 4개의 빈(bin;288A-D)은 출력 픽셀에 기여할 수 있는 샘플을 담고 있다. 이 예에서, 출력 픽셀의 중심, 즉, 각 출력 픽셀을 생성하기 위해 필터가 적용되는 중심 장소은 빈(bin)의 경계(288A-288D)에 위치한다. 각 빈(bin)은 16개의 샘플을 포함하고, 출력 픽셀을 생성하기 위해 4개의 bin(2 x 2)의 배열이 여과된다. 원형 필터가 사용된다고 가정하면, 픽셀 중심로부터의 각 샘플의 거리는 어느 필터 값이 그 샘플에 적용될 것인가를 결정한다. 예를 들어, 샘플(296)이 상대적으로 픽셀 중심에 가깝고, 그래서 8의 필터 값을 가지는 필터영역 내에 해당한다. 마찬가지로, 샘플(294)과 샘플(292)은 4와 2의 필터 값을 가지는 필터영역 내에 해당한다. 그러나 샘플(290)은 최대 필터 범위 밖에 있으며, 그래서 0의 필터값을 가진다. 그러므로 샘플(290)은 출력 픽셀의 값에 기여하지 않을 것이다. 이 유형의 필터는 픽셀 중심에 가장 가까이 위치하는 샘플이 가장 많이 기여하는 반면, 픽셀 중심에서 더 멀리 떨어져 있는 픽셀은 최종 출력 픽셀값에 보다 덜 기여할 것이다. 이 유형의 여과는 이미지의 갑작스런 변화(예컨대, 검은선으로부터 밝은 배경에 이르기까지)를 부드럽게 함에 의해 앤티-앨리어싱을 자동적으로 수행한다. 앤티-앨리어싱을 위한 다른 특히 유용한 유형의 필터로는 윈도우 싱크 필터(windowed sinc filter)가 있다. 유리하게는, 윈도우 싱크 필터는 혼합되거나 " 불분명" 하게 된 이미지의 일부를 다시 날카롭게 하는 음의 돌출부를 가지고 있다. 음의 돌출부는 계산되는 픽셀로부터 샘플을 빼도록 하는 필터 영역이다. 콘트라스트에서 음의 돌출부의 어느 한 쪽에 있는 샘플들이 계산되는 픽셀에 더해진다.

샘플(290-296)의 예제 값이 박스(300-308)에 나타나 있다. 이 예에서, 각 샘플은 그 샘플의 위치 데이터에 더하여 적색, 녹색, 청색, 알파값을 포함한다. 블록(310)은, 비정규화된 출력 픽셀에 대해, 각 픽셀 구성요소값을 연산하는 것을 나타낸다. 블록(310)이 나타내는 바와 같이, 잠재적으로 바람직하지 않은 게인(gain)이 최종 픽셀 값에 들어간다(

즉, 2000의 적색 구성요소값을 가지는 출력픽셀은 그 샘플의 적색 구성요소값의 어느 것 보다도 훨씬 높다), 앞서 언급한 바와 같이, 필터값이 정규화값(308)을 얻기 위해 더해질 수 있다. 정규화값(308)은 출력 픽셀로부터 불필요한 게인(gain)을 나누는데 이용된다. 블록(312)은 이 과정 및 최종 정규화된 예제 픽셀값을 나타내고 있다.

여기서 사용된 값들은 설명의 목적을 위해 선택된 것일 뿐이며, 그에 제한하고자 하는 의도가 아니라는 점에 주의해야 한다. 예를 들어, 필터는 각각 다른 필터값을 가지고 수많은 영역을 가질 수 있다. 일례로, 어떤 영역은 음의 필터값을 가질 수 있다. 활용되는 필터는, 픽셀 중심로부터의 샘플의 거리에 기초한 각 샘플을 위해 값이 구해지는 연속함수일 수 있다. 또한, 부동소수점값이 정확도의 증가를 위해 이용될 수도 있다는 점에 주목하라. 다양한 필터, 예컨대, 박스(box), 텐트(tent), 원기둥(cylinder), 원뿔(cone), 가우션(Gaussian), 카트멀-롬(Catmull-Rom), 미첼과 네드라발리 필터(Mitchell and Netravalli filter), 윈도우 싱크(windowed sinc) 등이 활용될 수 있다.

필터 중량치(filter weights)는 반드시 도면에서 나타난 바처럼 2의 제곱일 필요는 없다. 도면에 나타난 예는 설명을 위해 단순화된 것이다. (예컨대, 픽셀 또는 필터 중심으로부터의 샘플의 거리에 따라 인덱스된 수 많은 기입사항을 가진)필터 중량치의 테이블이 사용될 수 있다. 더욱이, (비록 빈 안의 일부 샘플이 0의 중량치를 가지고 있어서 최종 픽셀값에 아무런 기여를 하지 않더라도) 일부 실시예에서 각 빈의 각 샘플은 픽셀값을 형성하기 위해 더해질 수 있다.

풀-스크린 앤티-앨리어싱(Full-Screen Anti-aliasing)

현재 수많은 3D 그래픽 시스템이 라인(line)과 도트(dot)에 대해 실-시간(real time) 앤티-앨리어싱만을 제공하고 있다. 어떤 시스템은 다각형의 에지(edge)를 " 흐릿(fuzzed)" 하게 할 수 있지만, 이 기술은 전형적으로 모든 다각형이 깊이(depth)에 있어서 예비적으로 소트(sort)되어 있어야만 최적으로 동작한다. 이는 (다각형의 깊이를 예비-소트하지 않는) 대부분의 애플리케이션을 위한, 일반-목적의 3D 렌더링 하드웨어의 목적에 역행한다. 일례로, 그래픽 시스템(112)은 5x5-콘볼루션 필터로 여과된, 출력 픽셀 당 16개 샘플까지 확률적으로 샘플링함으로써, 풀-스크린 앤티-앨리어싱을 수행하도록 구성될 수 있다.

가변-해상도 수퍼 샘플링 -- 도 21-25

이제 도 21을 보면, 샘플버퍼(162)를 나누기 위한 한가지 가능한 다이어그램이 나타나 있다. 이 실시예에서, 샘플버퍼(162)는 중심 영역(354), 중간 영역(352) 및 주변 영역(350)의 3가지 영역으로 나뉜다. 이 각각의 영역은 직사각형의 형성으로된 외부 경계를 가지고 있지만, 중간 및 주변 영역은 그 중심에 직사각형의 홀을 가지고 있다. 각 영역은 특별한 (프레임 당) 지속적인 특성, 예컨대, 지속적인 밀도의 샘플 밀도 및 픽셀 빈의 지속적인 사이즈로 구성될 수 있다. 일례로, 전체 밀도 범위는 256이 될 수 있다. 즉, 한 영역은 16 스크린 픽셀(4 x 4) 당 하나의 샘플과 1 스크린 픽셀을 위한 16 샘플 사이를 지원할 수 있다. 일례로, 샘플 밀도는, 선형적 또는 비선형적으로, 각 영역을 가로지른다. 다른 실시예에서, 디스플레이는 복수의 지속적인 크기의 영역(예컨대, 크기에 있어 4 x 4 픽셀 또는 크기에 있어 40 x 40 픽셀)으로 나뉠 수 있다는 점에 유의하라.

하나 이상의 영역 구석(예컨대, 중심 영역의 구석)을 포함하는 다각형을 위한 연산을 수행하기 위해서, 샘플버퍼는 복수의 서브영역으로 더 나뉘어질 수 있다. 도 21에서, 샘플버퍼(162)의 일례는, 나타낸 대로, 서브영역으로 나뉘어 졌다. 이러한 각각의 서브-영역은 직사각형이며, 그래픽 시스템(112)가 서브-영역을 가진 2D 어드레스로부터 샘플버퍼(162)내의 선형 어드레스로 전환할 수 있도록 한다. 따라서, 일부 실시예에서 각 서브-영역은 서브-영역 안의 픽셀을 위한 저장이 시작되는 곳을 지시하는 메모리 베이스 어드레스를 가진다. 각 서브-영역은 또한 그 폭과 관련한 " 보폭(stride)" 파라미터를 가진다.

수퍼-샘플되는 샘플버퍼의 또 다른 분할로는 원형의 모습이다. 지금 도22를 보면 그러한 한 예가 나타나 있다. 예컨대, 각 영역은 그와 관련하여 2개의 반경(즉, 360-368)을 가질 수 있으며, 이는 그 영역을 3개의 동심원-영역으로 나눈다. 원-영역은 모두 동일한 스크린 점, 즉 중심 센터-점에 중심을 두고 있을 수 있다. 그러나, 중심 센터-점은 항상 중심영역의 중앙에 배치될 필요는 없다는 점에 유의하라. 일부 실시예에서, 그것은 스크린 밖(즉, 디스플레이 장치의 가시 표시면 밖)에 배치될 수도 있다. 나타내어진 실시예는 7개의 독특한 원형-영역을 지원한다. 그 원의 일부는 2개의 상이한 영역을 가로질러 나뉘어질 수 있으며, 그에 의해 독특한 원형-영역을 5 이하로 줄일 수 있다.

원형 영역은 실제로 사용되는 지속적인 샘플 밀도의 영역의 윤곽을 그릴 수 있다. 예를 들어, 도면에 나타난 실시예에서, 중심 영역(354)은 스크린 픽셀 당 8 샘플의 샘플버퍼밀도를 할당 할 수 있지만, 가장 안쪽의 원(368)의 밖에서는, 그것은 픽셀 당 오직 4 샘플을 사용할 수 있으며, 그 다음 원(366)의 밖에서는 그것은 픽셀당 오직 2개의 샘플을 사용할 수 있다. 따라서, 이 실시예에서 링(ring)들이 반드시 실제의 메모리를 절약할 필요는 없지만, 그것들은 잠재적으로 (픽셀 콘볼루션 대역폭은 물론) 샘플버퍼의 안쪽 및 바깥쪽으로의 메모리 대역폭을 절약할 수 있다. 상이한 효과의 샘플 밀도를 지시함에 더하여, 링들은 또한 사용되는 상이한 샘플 위치도해를 지시하기 위해 사용될 수 있다. 앞에서 언급한 바와 같이, 이 샘플 위치도해는 RAM/ROM의 칩상에 또는 프로그램될 수 있는 메모리에 저장될 수 있다.

앞에서 논의한 바와 같이, 일부 실시예에서 수퍼-샘플되는 샘플버퍼(162)는 빈으로 더욱 나뉘어질 수 있다. 예를 들어, 빈(bin)은 단일의 샘플이나 한 배열의 샘플(예컨대, 2x2 또는 4x4 샘플들)을 저장할 수 있다. 일례로, 비록 다른 구성이 가능하고 숙고될 수 있더라도, 각 빈은 1에서 16개의 샘플 포인트를 저장할 수 있다. 각 영역은 특별한 빈 크기와 지속적인 메모리 샘플 밀도와 함께 구성될 수 있다. 보다 낮은 밀도 영역이 반드시 보다 큰 빈 크기를 가질 필요는 없다는 점에 유의하라. 일례로, 영역(또는 적어도 내부 영역)은 그 영역을 포함하는 빈 크기의 정확한 정수의 배수이다.

가변-해상도 수퍼-샘플링은 디스플레이 장치상에 표시되는 각 픽셀을 위한 다양한 수의 샘플을 연산하는 것을 포함한다. 이미지의 특정 지역(예컨대, 물체의 가장자리 근처)은 보다 많은 갯수의 샘플이 있을 때 이득을 얻을 수 있는 한편, 다른 지역(예컨대, 상수의 컬러 및 밝기를 가지는 부드러운 지역)은 기타의 샘플을 필요로 하지 않을 수 있다. 메모리와 대역폭을 절약하기 위해서는, 증가된 해상도로부터 이득을 얻을 수 있는 영역에서만 특별한 샘플이 사용될 수 있을 것이다. 예를 들어, 디스플레이의 일부가 상수의 청색으로 채색되면(예컨대, 배경으로), 특별한 샘플은 모두 단순히 (표시되고 있는 배경색과 동일한) 상수값을 가지므로 특별히 유용하지는 않을 것이다. 그와 대조적으로, 스크린 상의 제2 지역이 3D로 렌더링되는 복잡한 텍스쳐와 에지를 가지는 물체를 표시하고 있으면, 추가적인 샘플을 사용하는 것은 앨리어싱(aliasing)과 같은 특별한 아티팩트를 피하기 위해서 유용할 것이다. 이미지의 어느 지역이 보다 높은 샘플 밀도로부터 이득을 얻을 수 있을 것인가 결정하거나 예측하는 많은 상이한 방법이 사용될 수 있다. 예를 들어, 에지 분석(edge analysis)이 최종 이미지에 대해 실행될 수 있을 것이며, 그 정보는 어떻게 그 샘플 밀도가 분포되어야 하는지 예측하게 위해 사용된다. 또한, 소프트웨어를 응용하면 프레임의 어느 지역이 보다 높은 샘플 밀도로 할당된 것인지 지시할 수 있을 것이다.

수많은 상이한 방법이 가변-해상도 수퍼-샘플링을 수행하기 위해 이용될 수 있다. 이 방법들은 다음의 2가지 일반적인 분류에 해당하는 경향이 있다: (1) 드로어(draw) 또는 렌더링(rendering) 프로세스에 관련된 방법들, 및 (2) 콘볼루션(convolution) 프로세스에 관련된 방법들. 예를 들어, 다음의 방법중 어느 것을 사용함에 의해, 샘플들은 수퍼-샘플되는 샘플버퍼(162)로 렌더링 될 수 있다.

1) 균일한 샘플 밀도;

2) 영역 마다의 기준(per-region basis)에서의 가변적인 샘플 밀도(예컨대, 중간, 중심, 및 주변);

3) 스캔-라인 기준(scan-line basis)에서의(또는 적은 갯수의 스캔라인 기준에서의) 밀도를 변화함에 의해 가변적인 샘플 밀도.

스캔-라인 기준(scan-line basis)에서의 가변적 샘플 밀도는 밀도 검색 테이블을 이용함으로써 이루어질 수 있다. 예를 들어, 테이블은 특정한 스캔라인의 처음의 5개의 픽셀이 각각 3개의 샘플을 가지고 있다고 지정할 수 있으며, 다음의 4개의 픽셀이 각각 2개의 샘플을 가지고 있다고 있다는 등으로 지정할 수 있다.

콘볼루션 측에서는, 다음의 방법이 가능하다:

1) 균일한 콘볼루션 필터;

2) 연속적으로 가변적인 콘볼루션 필터;

3) 다중의 공간적 주파수에서 작동하는 콘볼루션 필터.

균일한 콘벌브 필터는, 예를 들어, 각 연산되는 픽셀을 위한 지속적인 범위(또는 선택된 샘플의 수)를 가지고 있다.. 대조적으로, 연속적으로 가변적인 콘볼루션 필터는 픽셀을 연산하는데 사용되는 샘플의 갯수를 점차적으로 변화할 수 있다. 그 기능은 관심의 중심에서의 최대에서부터 주변영역에서의 최소에 이르기까지 연속적으로 변화할 수 있다. (렌터링 측과 콘볼루션 측 모두에 대해) 이 방법의 다른 조합도 또한 가능하다. 예를 들어, 연속적으로 가변의 콘볼루션 필터가 샘플에 대해 사용될 수 있는 한편, 지속적인 샘플 밀도가 렌터링 측에 대해 사용될 수 있다.

이미지의 어느 지역에 픽셀 당 보다 많은 샘플이 할당할 것인가 결정하기 위한 상이한 방법이 또한 숙고된다. 일례로, 스크린 상의 이미지가 주 초점을 가지면(예컨대, 컴퓨터 게임의 마리오와 같은 캐릭터), 보다 많은 갯수의 샘플이 마리오 부근 지역을 위해 연산될 수 있을 것이며, 다른 지역(예컨대, 배경 근처 또는 스크린의 에지 부근)의 픽셀을 위해서는 보다 적은 갯수의 샘플이 연산될 수 있을 것이다.

또다른 실시예에서, 관찰자의 중심점은 눈/머리/손-추적에 의해 판단될 수 있다. 머리-추적 예에서, 관찰자가 바라보는 방향은 관찰자의 머리의 방위로부터 판단되거나 예측되며, 이는 다양한 메커니즘을 이용하여 측정될 수 있다. 예를 들어, 그래픽 시스템(112)에 방위 정보를 제공하기 위하여, 관찰자가 입은 (눈/머리 추적을 가진) 헬멧 또는 투구가 단독으로 또는 손-추적 메커니즘, 지방이 또는 눈-추적 센서가 사용될 수 있을 것이다. 다른 대체적 예제로는, 사용자의 이미에 부착된 적외선 반사 점을 사용한 머리-추적, 또는 머리-추적 센서 및/또는 눈-추적 센서를 가진 안경을 사용하는 것을 포함한다. 머리추적 및 손-추적을 사용하는 한가지 방법이 미국 특허번호 5,446,834(발명의 명칭" Method and Apparatus for High Resolution Virtual Reality Systems Using Head Tracked Display," 발명자 Michael Deering, 공개일 1995년 8월 29일)에 나타나 있다. 머리 추적을 위한 다른 방법이 또한 가능하며 숙고된다.(예컨대, 적외선 센서, 전자기 센서, 캐패서티 센서, 비디오 카메라, 음파 및 초음파 감지기, 옷에 부착된 센서, 비디오 추적 장치, 전도성 잉크, 압력 계기, 동력-피드백 감지기, 광섬유 센서, 공기 센서, 마그네틱 추적 장치, 기계적 스위치).

앞에서 언급한 바와 같이, 눈-추적은 머리-추적과 연계하여 사용될 때 특히 유익하다. 눈-추적에 대한 실시예에서, 관찰자가 보는 방향은 관찰자의 머리에 관련한 관찰자의 눈의 방위를 감지함에 의해 직접 측정될 수 있다. 디스플레이 장치에 관련한 관찰자의 머리의 위치 및 방위에 관련한 다른 정보와 조합되었을 때, 이 정보는 관찰자의 중심점(또는 2개의 눈-추적 센서가 사용될 경우의 중심점들)의 정확한 측정을 가능하게 할 것이다. 눈 추적을 위한 한 가지 가능한 방법이 미국 특허번호 5,638,176(발명의 명칭" 저렴한 간섭계의 눈 추적 시스템"). 눈 추적을 위한 다른 방법이 또한 가능하며 숙고될 수 있다(예컨대, 위에서 나열된 머리추적을 위한 방법).

어느 방법이 사용되는가에 관계없이, 관찰자의 중심점은 위치를 바꾸며, 샘플의 분포 또한 그러하다. 예를 들어, 만약 관찰자가 바라보는 것이 스크린 상의 위쪽 왼쪽 구석에 중심이 있다면, 스크린 상의 위쪽 왼쪽 구석에 대응하는 픽셀은 각각 8개 또는 16개의 샘플이 할당되는 반면에, 반대편 구선에 있는 픽셀은 픽셀당 오직 1개 또는 2개의 샘플만이 할당될 것이다. 일단 관찰자의 응시가 바뀌면, 픽셀마다의 샘플의 할당도 바뀐다. 관찰자의 응시가 스크린 상의 아래쪽의 오른쪽으로 이동하면, 스크린 상의 윗쪽의 왼쪽 구석의 픽셀은 오직 1개 또는 2개의 샘플만 할당될 것이다. 따라서 픽셀 당 샘플의 갯수는 관찰자의 중심점에 관련하여 스크린의 상이한 지역에 따라 능동적으로 변화할 수 있다. 일부 실시예에서, 다중 사용자는 각각 그래픽 시스템(112)으로의 입력을 제공하는 머리/눈/손 추적 메커니즘을 가질 수 있다. 이 실시예에서, 높고 낮은 새플 밀도의 대응하는 지역을 가지는, 스크린상의 2 이상의 중심점을 상상할 수 있다. 앞에서 언급한 바와 같이, 이 샘플 밀도들은 렌더링 프로세스만, 여과 프로세스만, 또는 양 프로세스 모두에 영향을 줄 수 있다.

이제 도 24A-B를 보면, 픽셀 마다의 샘플의 갯수를 할당하는 방법의 일례가 나타나 있다. 본 방법은 하나 이상의 중심점에 관련한 픽셀의 장소에 따라서 샘플의 갯수를 할당한다. 도 24A에서, 중심점(즉, 관찰자가 응시하는 초점; 362)을 결정하기 위해 눈-추적 또는 머리-추적 장치(360)가 사용된다. 이것은 관찰자의 눈(도면에서 364로 나타난다)이 직면하는 방향을 결정하기 위한 추적 장치(360)를 사용함으로써 결정된다. 도면이 나타내는 바와 같이, 이 실시예에서, 픽셀은 (중심점(362)의 주위에 중심을 두고 있는) 중심 영역(354), 중간 영역(352), 및 주변 영역(350)으로 나뉜다.

3개의 샘플 픽셀이 도면에 지시되어 있다. 샘플 픽셀(374)는 중심 영역(314)의 내부에 배치되어 있다. 중심 영역(314)이 8개의 샘플을 가지는 것으로 가정하고, 각 픽셀이 4개의 빈에 접하면, 최대 32개의 샘플이 각 픽셀에 기여할 것이다. 샘플 픽셀(372)은 중간 영역(352) 내부에 배치되어 있다. 중간 영역(352)가 4개의 샘플을 가지는 빈으로 구성되고, 각 픽셀을 위한 콘볼루션 반경이 4개의 빈에 접하면, 최대 16개의 샘플이 각 픽셀에 기여할 것이다. 샘플 픽셀(370)은 주변 영역(350)에 배치되어 잇다. 주변 영역(370)이 각 하나의 샘플을 가지는 빈으로 구성되고, 각 픽셀을 위한 콘볼루션 반경이 1개의 빈을 접하면, 주변 영역(350) 내의 픽셀에 대해서는 하나의 샘플 당 하나의 픽셀의 상관관계가 있다. 이러한 값들은 단순히 예시일 뿐이고, 다른 숫자의 영역, 빈 마다의 샘플, 콘볼루션 반경이 사용될 수 있다.

이제 도 24B를 보면, 동일한 예제가 나타나 있지만, 중심점(362)가 다르다. 도면이 나타내는 바와 같이, 추적 장치(360)가 중심점(362)의 위치의 변화를 감지하면, 그것은 그래픽 시스템으로 입력을 제공하며, 이것은 중심 영역(354)와 중간 영역(352)의 위치를 조절한다. 일부 실시예에서, 그 영역들의 일부(예컨대, 중간영역(352))는 디스플레이 장치(84)의 가장자리를 넘어설 수 있다. 이 예에서, 픽셀(370)은 지금 중심점(354)의 안에 있는 반면, 픽셀(372) 및 픽셀(374)는 주변 영역 안에 있다. 샘플의 구성을 도 24A의 예처럼 가정하면, 최대 32 샘플이 픽셀(370)에 기여할 수 있는 반면, 오직 하나의 샘플이 픽셀(372)및 픽셀(374)에 기여할 것이다. 유리하게는, 이 구성은 중심점 부근의 영역(즉, 관찰자가 바라보는 초점)을 위해서 더욱 많은 샘플을 할당할 수 있다. 이것은 디스플레이 장치(84)상의 모든 픽셀을 위한 많은 수의 샘플을 연선할 필요없이, 더욱 현실적인 이미지를 관찰자에게 제공할 수 있다.

이제 도 25A-B를 보면, 가변 해상도 수퍼-샘플되는 샘플버퍼로 구성된 컴퓨터 시스템의 또 다른 실시예가 나타나 있다. 이 실시예에서, 관찰자의 관심의 중심은 주인공(362)의 위치에 의해 결정된다. 주인공(362)이 스크린 주위를 움직임에 따라, 중간 및 중심 영역은 주인공(362)의 주위에 중심을 두게 된다. 일부 실시예에서 주인공은 (예컨대, 키보드 입력 또는 마우스에 의해 이동되는) 단순한 커서일 수도 있다.

또 다른 실시예에서, 보다 높은 샘플 밀도를 가진 영역이 디스플레이 장치(84)의 스크린의 중앙 주위에서 중심을 둘 수 있다. 유리하게는, 이렇게 하는 것은 (대부분의 시간동안 관찰자의 관심의 초점이 되는) 스크린의 중심에서 더욱 날카로운 이미지를 제공하면서도, 소프트웨어 및 하드웨어의 제어요구를 줄일 수 있다.

비록 위의 실시예들이 상당히 자세히 설명되었지만, 다른 변형도 가능하다. 위의 설명이 완전히 이해되면, 본 기술분야에서 능숙한 자로서는 수많은 변경과 수정을 할 수 있음이 명백할 것이다. 다음의 특허청구의 범위는 그러한 모든 변경과 수정을 포함하는 것으로 해석되어야 한다. 여기서 사용된 표제는 조직화의 목적일 뿐이며, 여기서 제공되는 발명의 상세한 설명이나 첨부된 특허청구의 범위를 제한하고자 하는 것이 아님에 주의하라.

산업상 이용 가능성

본 기술분야에서의 숙련자에 의해 본 명세서 및 첨부도면이 재고(再考)되어 이해되면, 여기에 설명된 본 시스템 및 방법은 그래픽 시스템과 서브시스템, 컴퓨터, 계산 장치, 셋-톱 박스, 게임 콘솔, 피디에이(PDA;personal digital assistants), 디지털 텔레비전, 비디오 프로세서, 그래픽 프로세서, 멀티미디어 시스템 및 프로세서, 가상 현실 시스템 및 그밖의 렌더링 시스템 및/또는 디스플레이 그래픽 데이터 등을 포함하며 이에 한정되지 않는 수많은 다른 분야에 적용될 수 있다.

(57) 청구의 범위

청구항 1.

한 세트의 그래픽 데이터를 수신하고 상기 그래픽 데이터에 기초하여 복수의 샘플을 렌더링하는 하나 이상의 프로세서(352)와;

상기 복수의 샘플을 저장하도록 구성된 샘플버퍼(162)와;

복수의 샘플-픽셀 연산유닛(360)으로서, 상기 샘플-픽셀 연산유닛은 출력픽셀을 생성하기 위해 상기 샘플버퍼로부터 샘플을 수신하고 여과하도록 구성되고, 상기 출력픽셀은 디스플레이 장치상의 이미지를 형성하는데 사용될 수 있고, 상기 복수의 샘플-픽셀 연산유닛의 각각은 이미지의 복수의 영역의 상이한 하나에 대응하여 픽셀을 생성하도록 구성되어 있는 샘플-픽셀 연산유닛(360)을 포함하는 그래픽 시스템.

청구항 2.

한 세트의 그래픽 데이터를 렌더링하는 방법으로서,

상기 그래픽 데이터를 수신하는 단계와;

상기 그래픽 터이터에 기초하여 하나 이상의 샘플을 생성하는 단계와;

상기 샘플을 저장하는 단계와;

상기 샘플을 복수의 영역으로 나누는 단계와;

상기 복수의 영역으로부터 저장샘플을 선택하는 단계와;

디스플레이 장치상의 이미지를 형성하기 위해 사용될 수 있는 복수의 출력픽셀을 형성하기 위하여 상기 선택된 샘플을 병렬로 여과하는 단계를 포함하는, 한 세트의 그래픽 데이터를 렌더링하는 방법.

청구항 3.

제2항에 있어서,

상기 각각의 샘플은 컬러 성분을 포함하고,

상기 여과하는 단계는 어떤 샘플이 소정의 필터 포락선 내부에 있는지 결정하는 단계와; 상기 소정의 필터 포락선 내부의 상기 샘플에 하나 이상의 가중인자를 곱하는 단계로서, 상기 가중인자는 상기 필터 포락선의 중심에 관련한 상기 샘플의 위치에 따라 변화하는 단계와; 출력픽셀을 생성하기 위해 상기 가중된 샘플을 합산하는 단계를 포함하는, 한 세트의 그래픽 데이터를 렌더링하는 방법.

청구항 4.

디스플레이 장치(84)와;

한 세트의 그래픽 데이터를 수신하는 수단과;

상기 세트의 그래픽 데이터에 기초하여 복수의 샘플을 렌더링하도록 구성된 하나 이상의 프로세서(352)와;

상기 렌더링된 샘플을 저장하도록 구성된 샘플버퍼(162)와;

출력픽셀을 생성하기 위해 저장샘플을 여과하도록 구성된 복수의 샘플-픽셀 연산유닛(170)으로서, 상기 출력픽셀은 디스플레이 장치상의 이미지를 형성하기 위해 사용될 수 있으며, 상기 복수의 샘플-픽셀 연산유닛의 각각은 상기 이미지의 복수의 상이한 영역중 하나에 대응하는 픽셀을 생성하도록 구성된 컴퓨터 시스템.

청구항 5.

제1항 내지 제4항 중 어느 한 항에 있어서,

상기 샘플 각각은 컬러 성분을 포함하고;

상기 샘플-픽셀 연산유닛은 어느 샘플이 소정의 필터 포락선내에 있는가 결정하고, 상기 소정의 필터 포락선 내의 샘플에 하나 이상의 가중인자를 곱하고, 이때 상기 가중인자는 상기 필터 포락선의 중심에 관련하여 상기 샘플의 위치에 따라 변화하고, 하나 이상의 출력픽셀을 형성하기 위해 가중된 샘플을 합산하는 시스템 또는 방법.

청구항 6.

제1항 내지 제5항 중 어느 한 항에 있어서,

상기 각 영역은 상기 이미지의 상이한 수직의 또는 수평의 줄에 대응하는 시스템 또는 방법.

청구항 7.

제1항 내지 제5항 중 어느 한 항에 있어서,

상기 각 영역은 하나 이상의 홀수 또는 짝수의 스캔라인에 대응하는 이미지의 부분들을 포함하는 시스템 또는 방법.

청구항 8.

제1항 내지 제5항 중 어느 한 항에 있어서,

상기 각 영역은 상기 이미지의 상이한 4분면을 포함하는 시스템 또는 방법.

청구항 9.

제1항 내지 제8항 중 어느 한 항에 있어서,

상기 복수의 영역은 오버랩(overlap)하는 시스템 또는 방법.

청구항 10.

제1항 내지 제9항 중 어느 한 항에 있어서,

상기 디스플레이 장치는 복수의 개인적인 디스플레이 장치를 포함하며,

상기 각 영역은 상기 복수의 개인적인 디스플레이 장치 중 단일의 하나에 대응하는 시스템 또는 방법.

청구항 11.

제1항 내지 제9항 중 어느 한 항에 있어서,

상기 디스플레이 장치는 복수의 개인적 디스플레이 장치를 포함하며,

상기 각 영역은 상기 복수의 개인적 디스플레이 장치 중 상이한 하나에 대응하는 시스템 또는 방법.

청구항 12.

제1항 내지 제11항 중 어느 한 항에 있어서,

상기 영역은 프레임-대-프레임 기준의 차원에서 변화하는 시스템 또는 방법.

청구항 13.

제1항 내지 제11항 중 어느 한 항에 있어서,

상기 이미지의 상기 영역은 샘플-픽셀 연산유닛의 각각에 의해 여과되는 샘플의 수의 균형을 잡기 위하여 프레임-대-프레임 기준의 차원에서 변화하는 시스템 또는 방법.

청구항 14.

제1항 내지 제5항, 제9항 내지 제13항 중 어느 한 항에 있어서,

상기 각 영역은 상기 이미지의 상이한 직사각형 부분인 시스템 또는 방법.

청구항 15.

제1항 내지 제14항 중 어느 한 항에 있어서,

상기 각 샘플은 컬러 성분을 포함하고,

상기 샘플-픽셀 연산유닛은 어느 샘플이 소정의 필터 포락선 안에 있는가 결정하고; 상기 소정의 필터 포락선 안의 샘플에, 상기 필터 포락선의 중심에 관련하여 상기 샘플의 위치에 따라 변화하는 하나 이상의 가중인자를 곱하도록 구성된 시스템 또는 방법.

청구항 16.

제1항 내지 제15항 중 어느 한 항에 있어서,

상기 각 샘플은 z-성분을 포함하는 시스템 또는 방법.

청구항 17.

제1항 내지 제16항 중 어느 한 항에 있어서,

상기 샘플버퍼에 저장된 상기 샘플은 이중으로 버퍼되는 시스템 또는 방법.

청구항 18.

제1항 내지 제17항 중 어느 한 항에 있어서,

상기 영역은 시간에 따라 사이즈가 변화하는 시스템 또는 방법.

청구항 19.

제1항 내지 제18항 중 어느 한 항에 있어서,

상기 각 영역은, 샘플-픽셀 연산유닛에 의해 여과되는 샘플의 수를 균형잡기 위해, 프레임-대-프레임 기준에서 사이즈가 변화하는 시스템 또는 방법.

청구항 20.

제1항 내지 제19항 중 어느 한 항에 있어서,

상기 각 영역은, 각 영역의 샘플의 수를 균일하게 하기 위해, 프레임-대-프레임 기준에서 사이즈가 변화하는 시스템 또는 방법.

청구항 21.

제1항 내지 제20항 중 어느 한 항에 있어서,

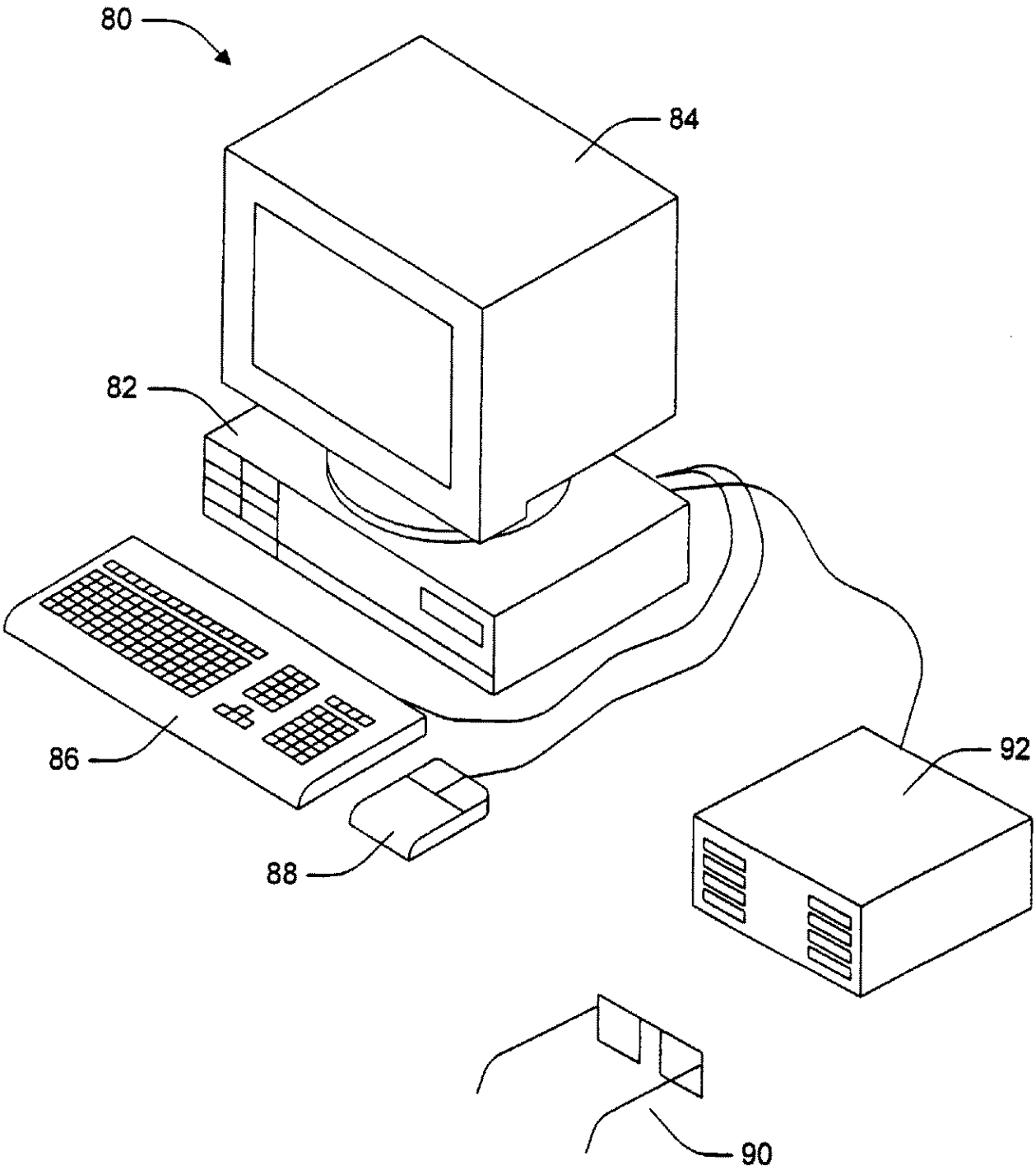
상기 샘플버퍼에 저장되는 상기 샘플은 빈(bin)에 저장되는 시스템 또는 방법.

청구항 22.

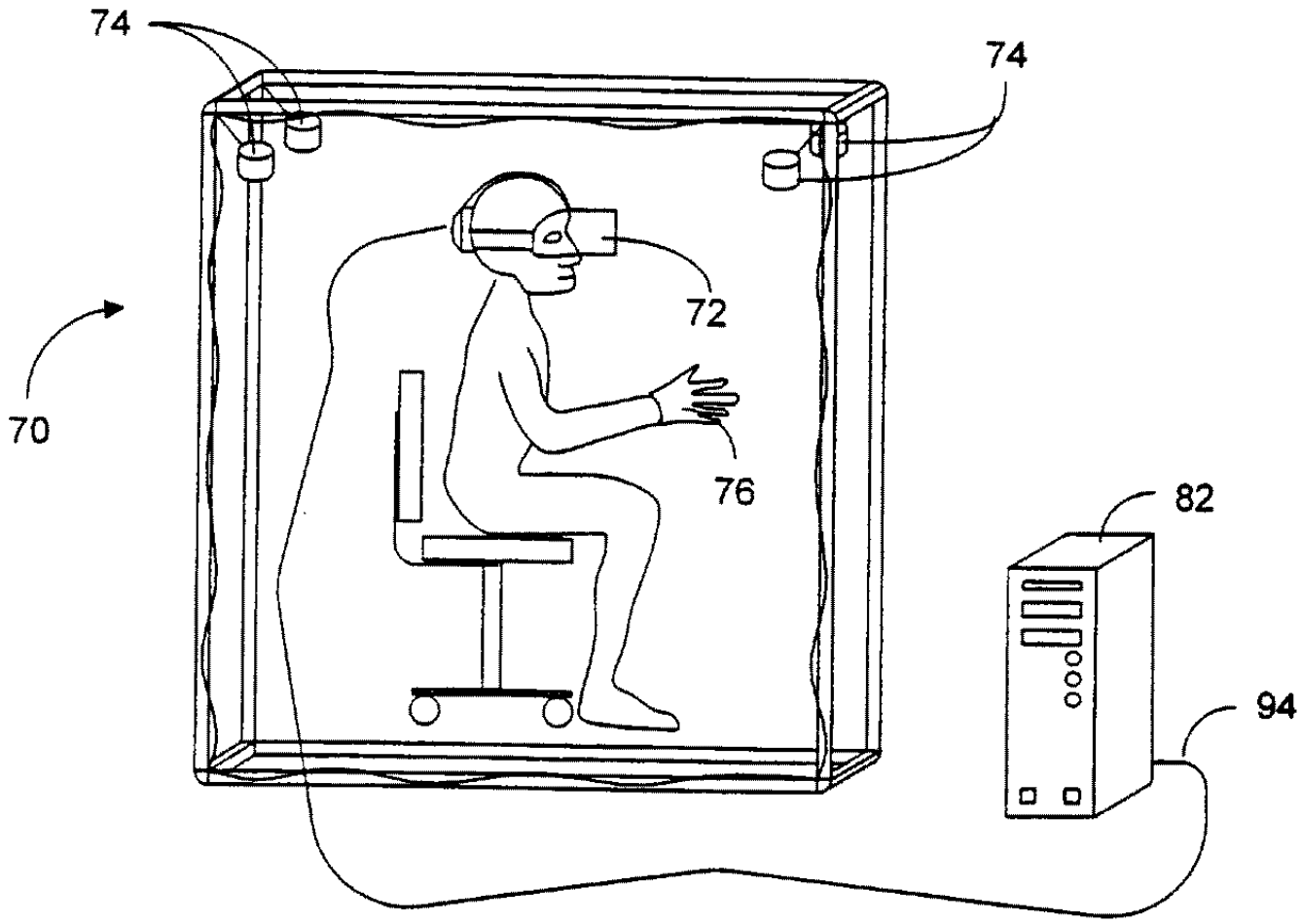
제1항 내지 제21항 중 어느 한 항에 있어서,

각 픽셀을 위해 여과되는 상기 샘플의 숫자는 영역에 따라 변화하는 시스템 또는 방법.

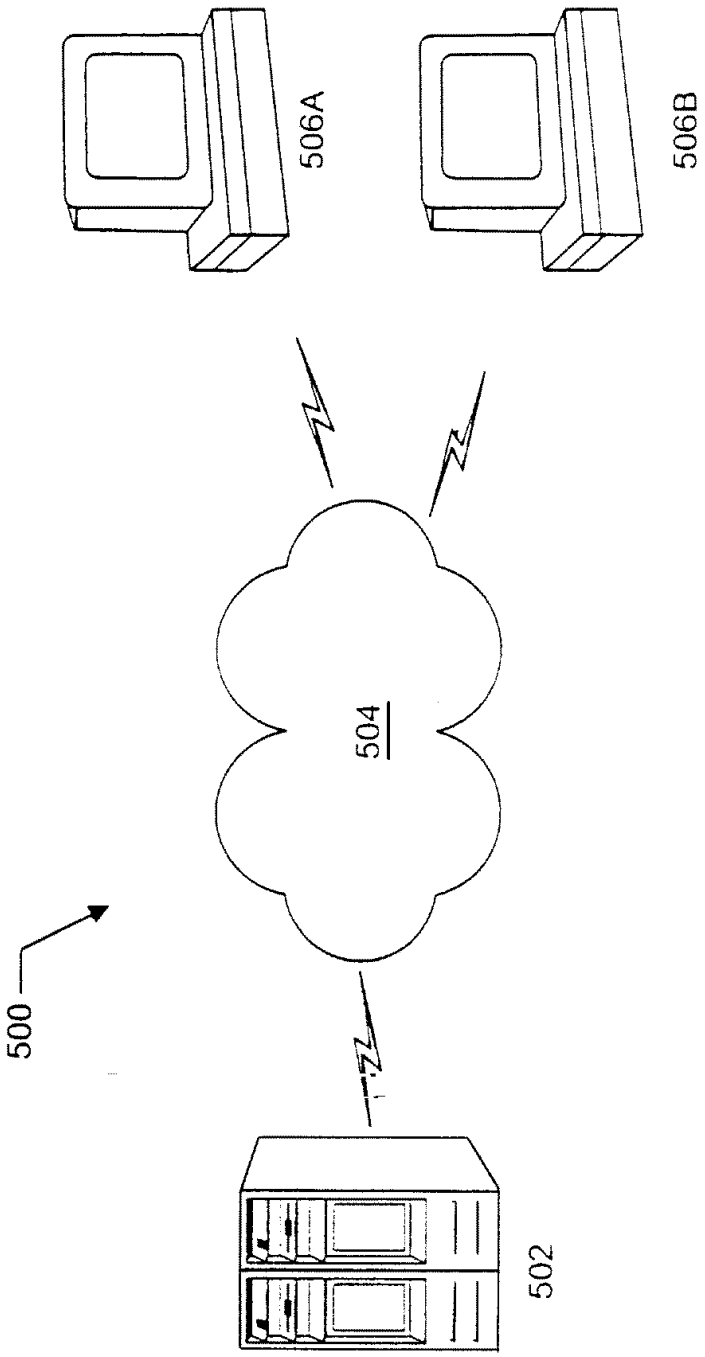
도면 1A



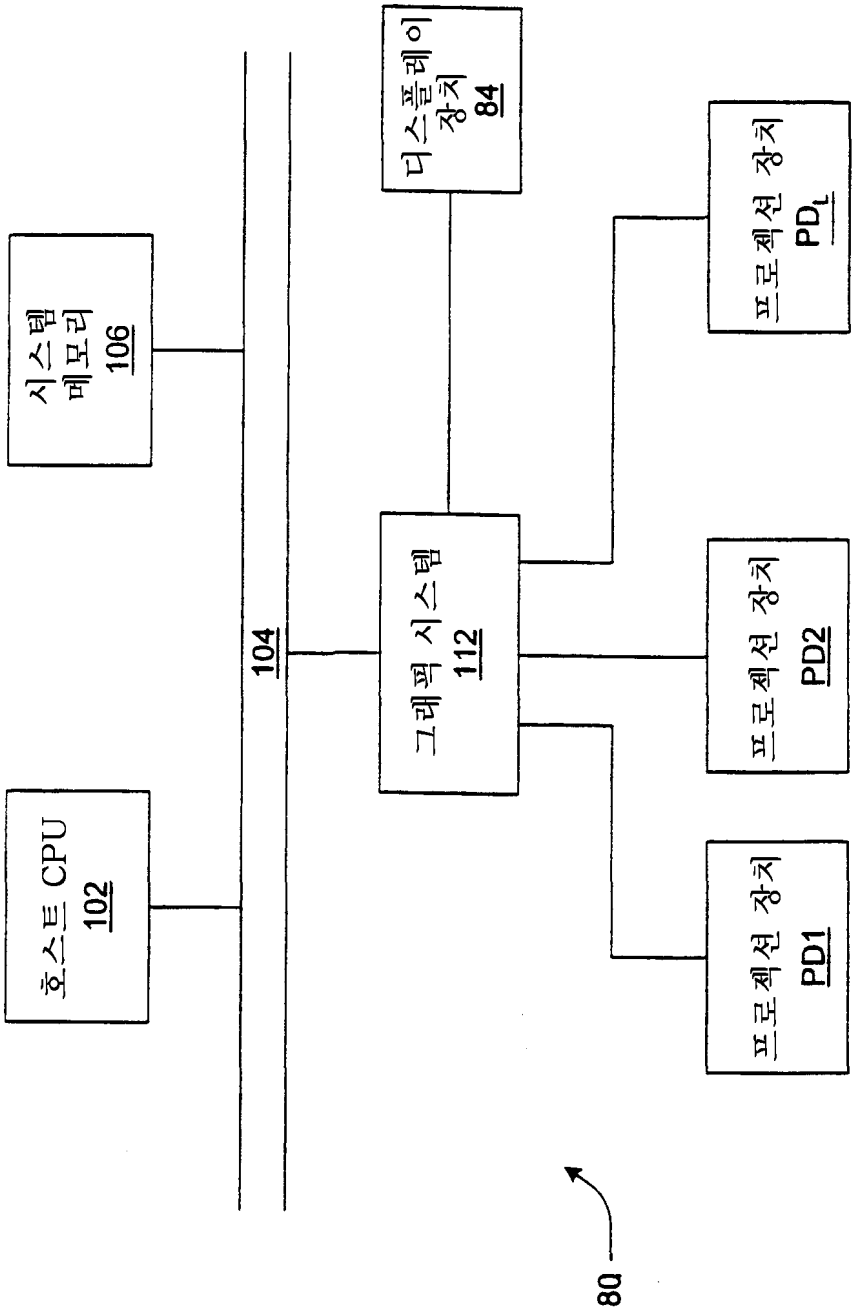
도면 1B



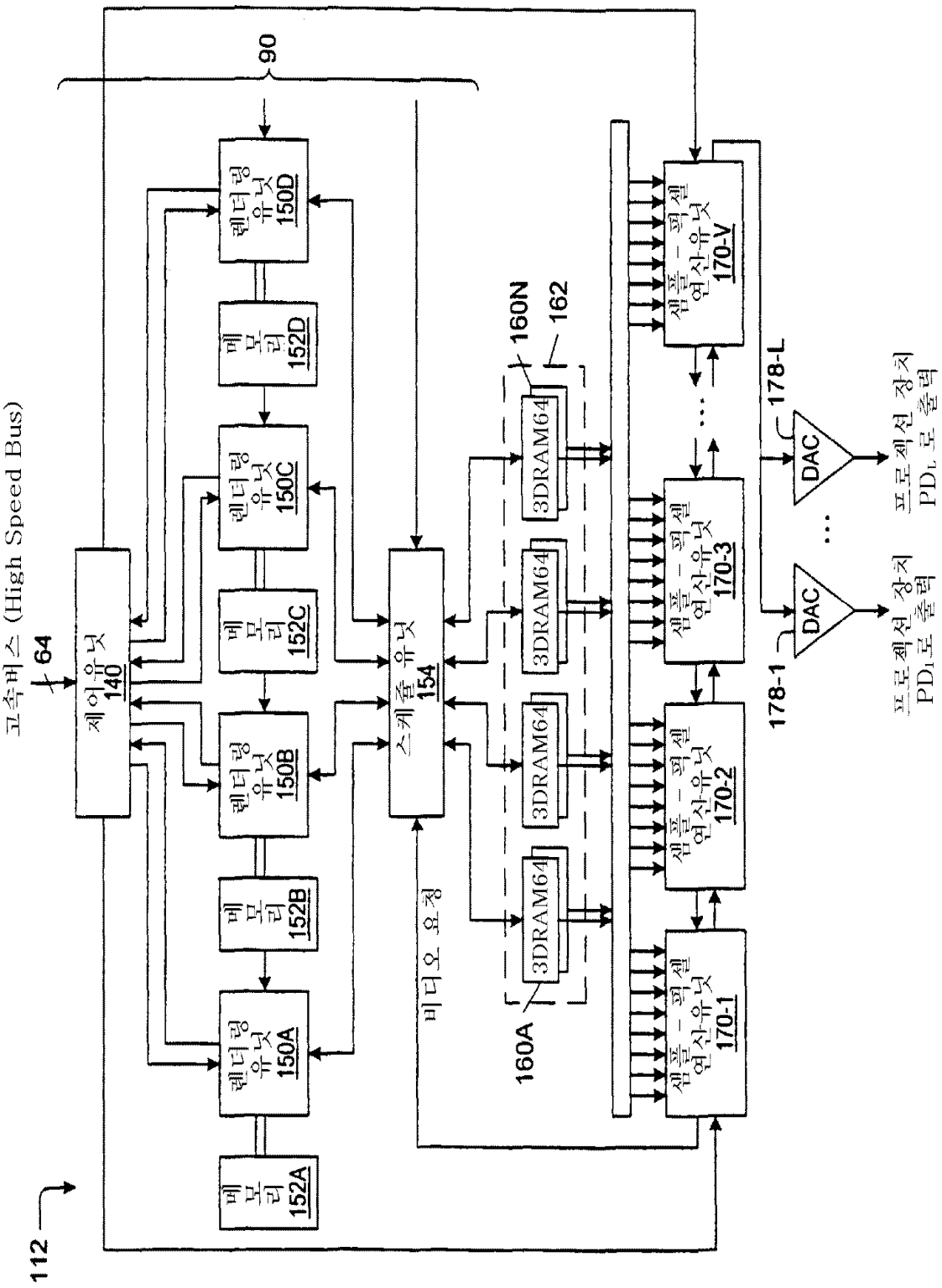
도면 2



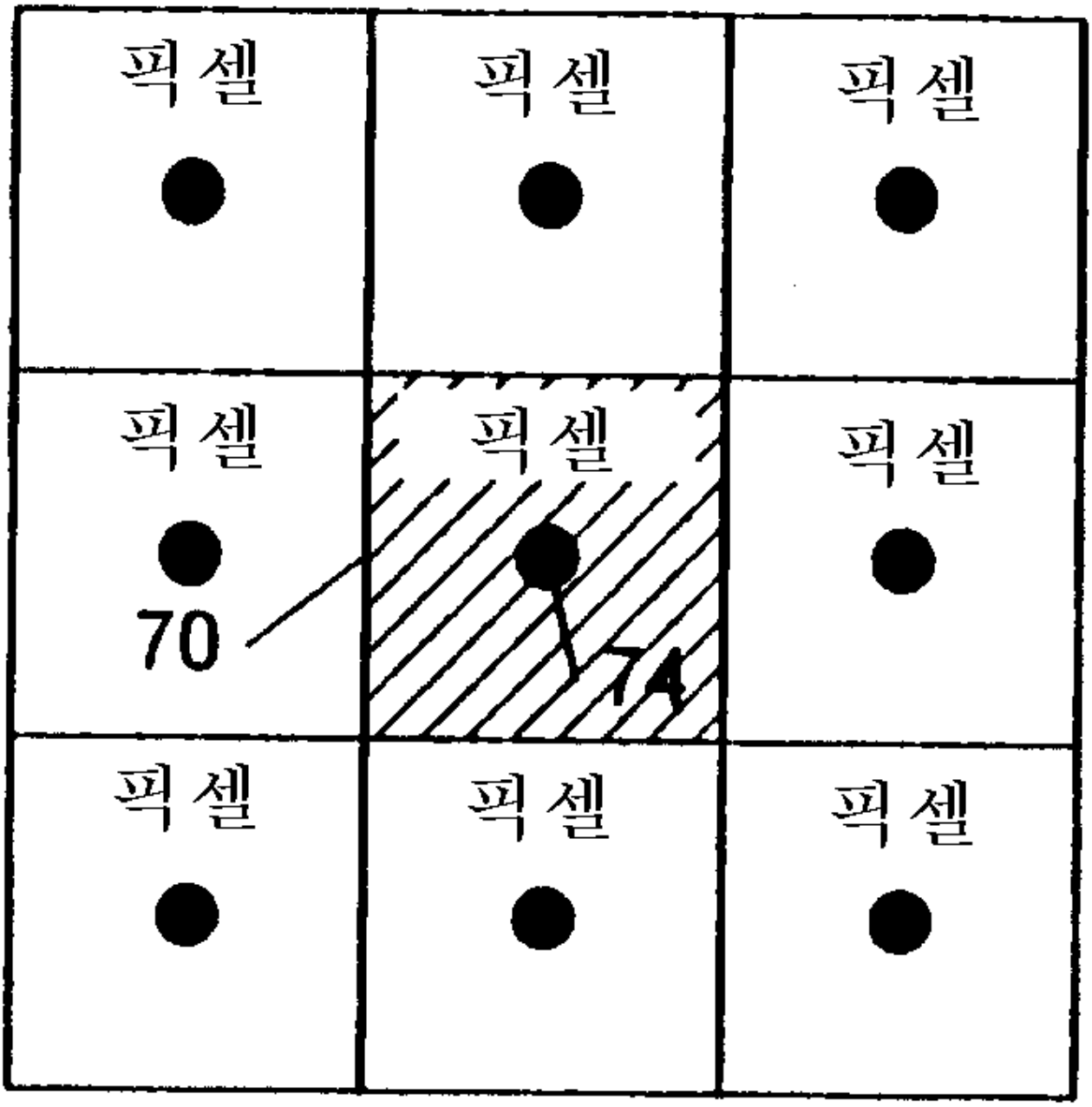
도면 3A


디스플레이 장치 84
시스템 메모리 106
104
그래픽 시스템 112
호스트 CPU 102
프로젝션 장치 PD$_L$
프로젝션 장치 PD2
프로젝션 장치 PD1
80

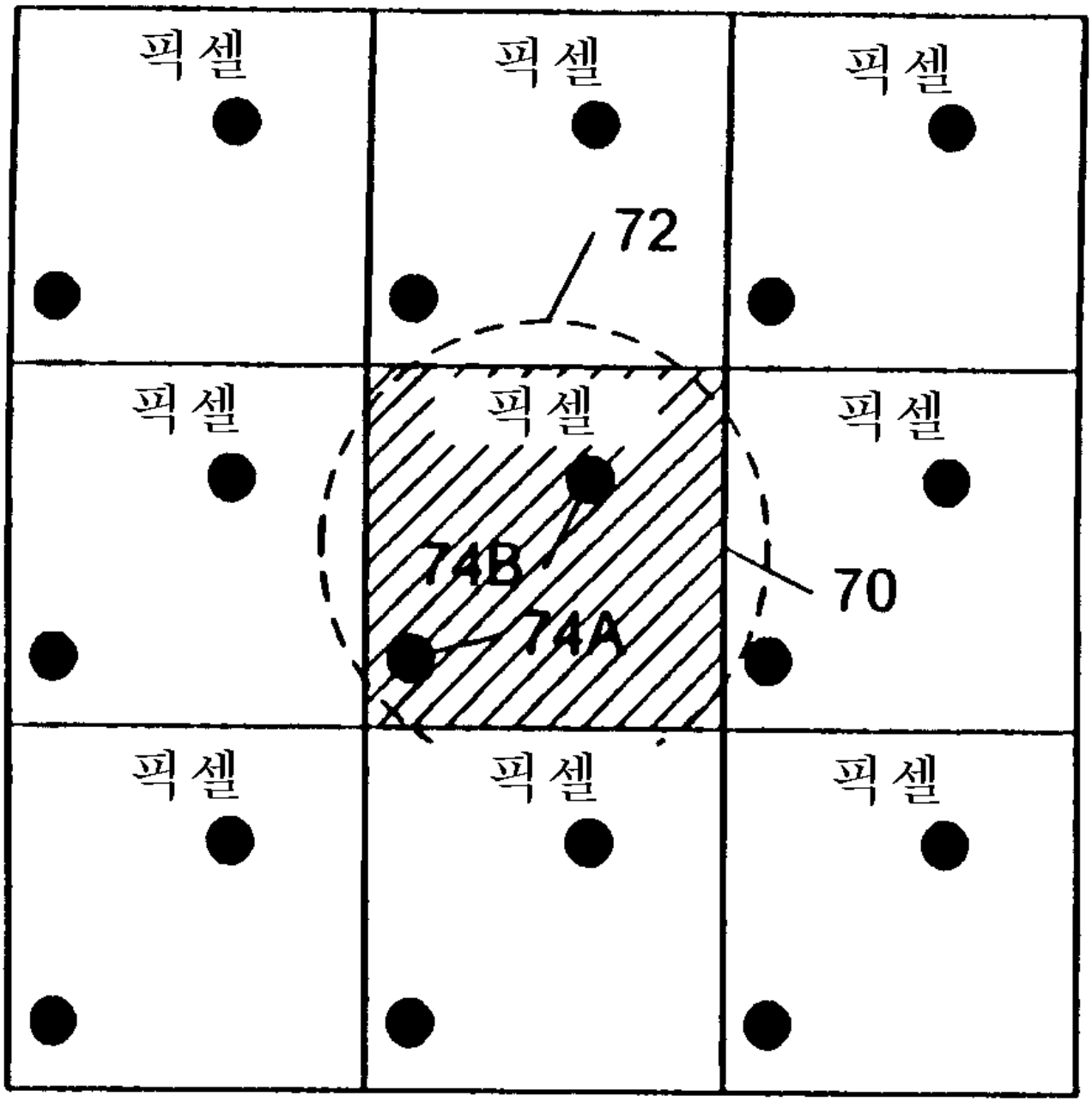
도면 3B


90
고속버스 (High Speed Bus)
64
112
제어유닛 140
렌더링 유닛 150A
렌더링 유닛 150B
렌더링 유닛 150C
렌더링 유닛 150D
메모리 152A
메모리 152B
메모리 152C
메모리 152D
스케줄 유닛 154
비디오 요청
160A
160N
162
3DRAM64
샘플-픽셀 연산유닛 170-1
샘플-픽셀 연산유닛 170-2
샘플-픽셀 연산유닛 170-3
샘플-픽셀 연산유닛 170-V
178-1
178-L
DAC
프로젝션 장치 PD1로 출력
프로젝션 장치 PDL로 출력

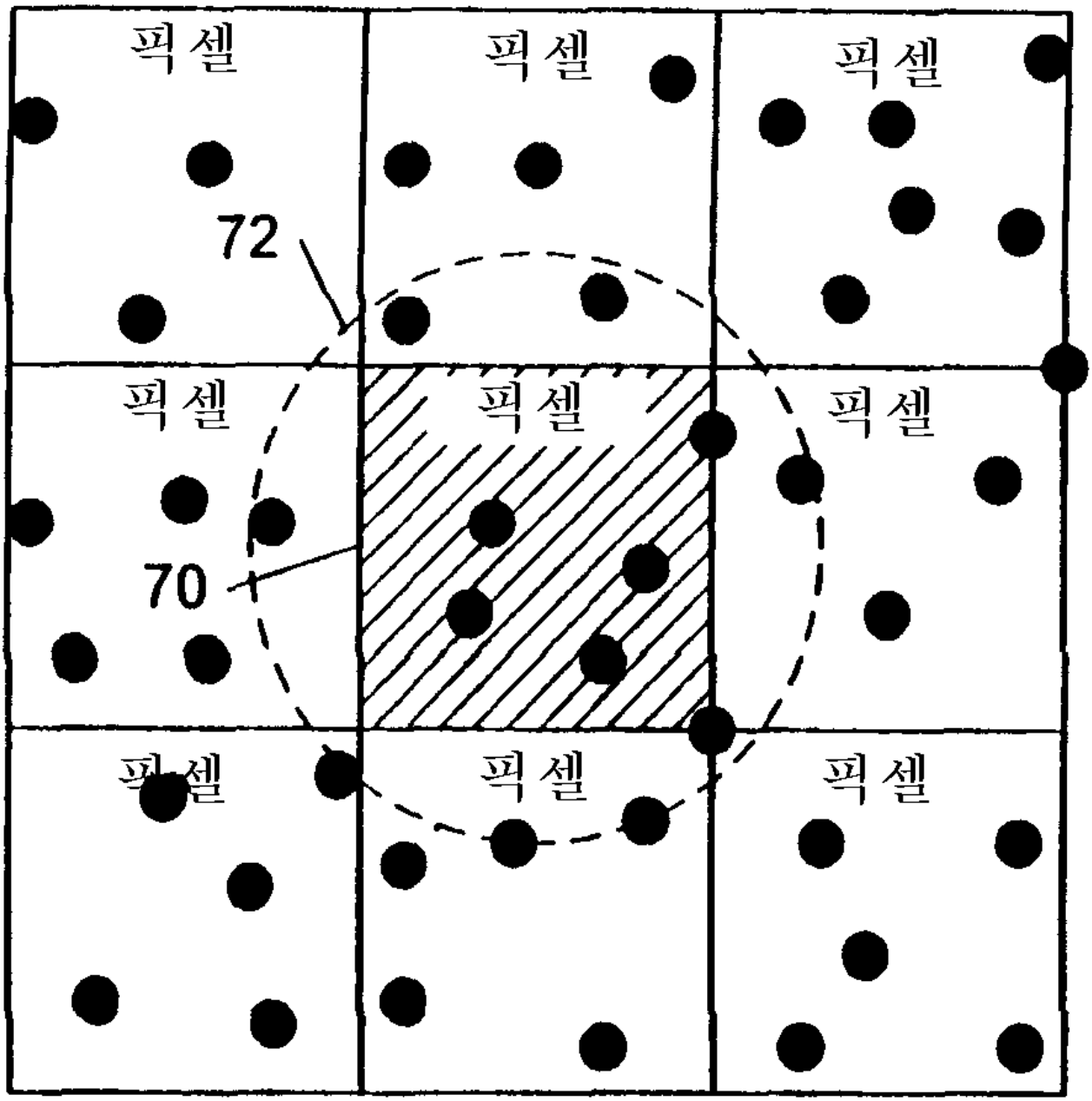
도면 4



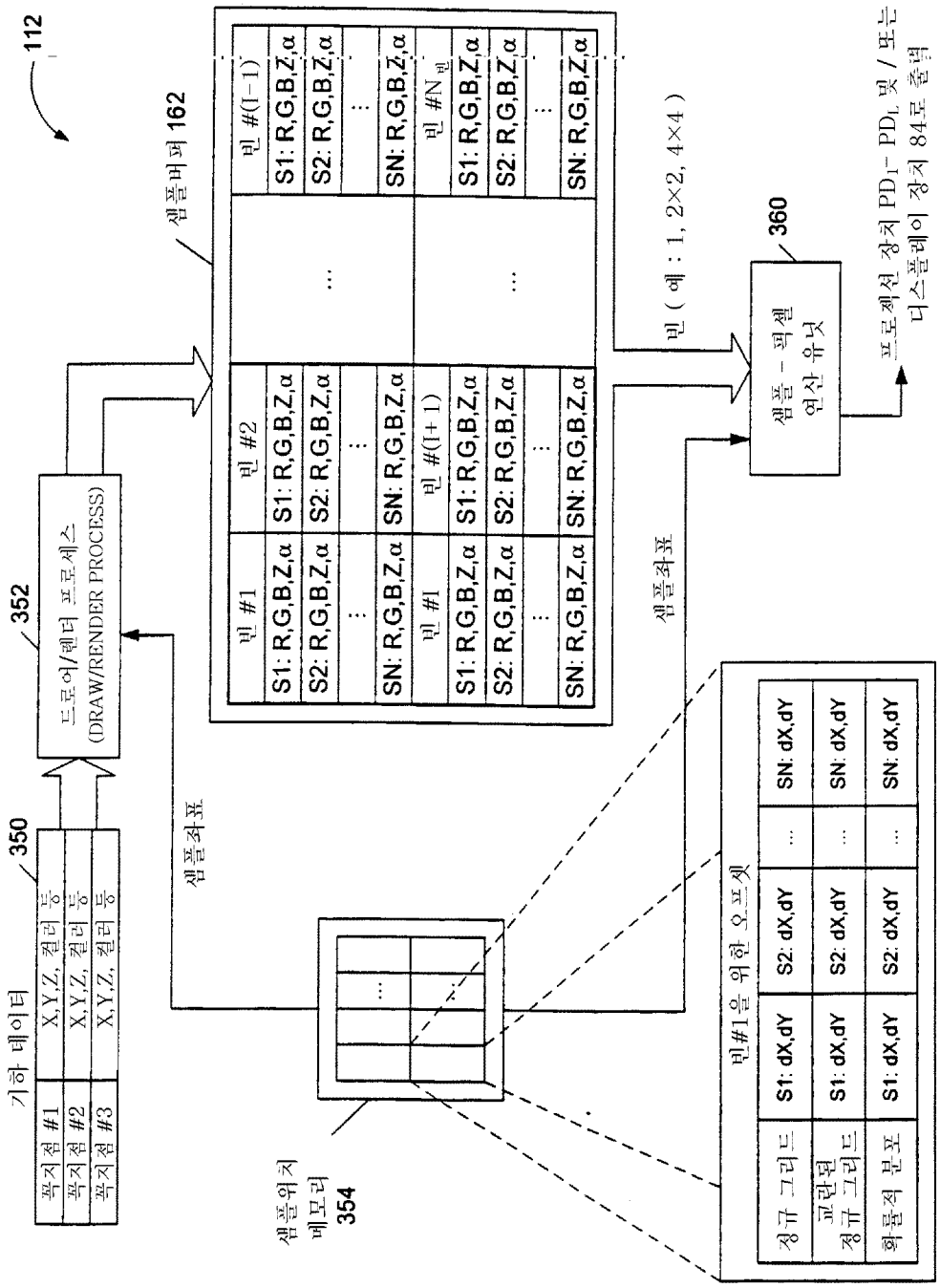
도면 5A



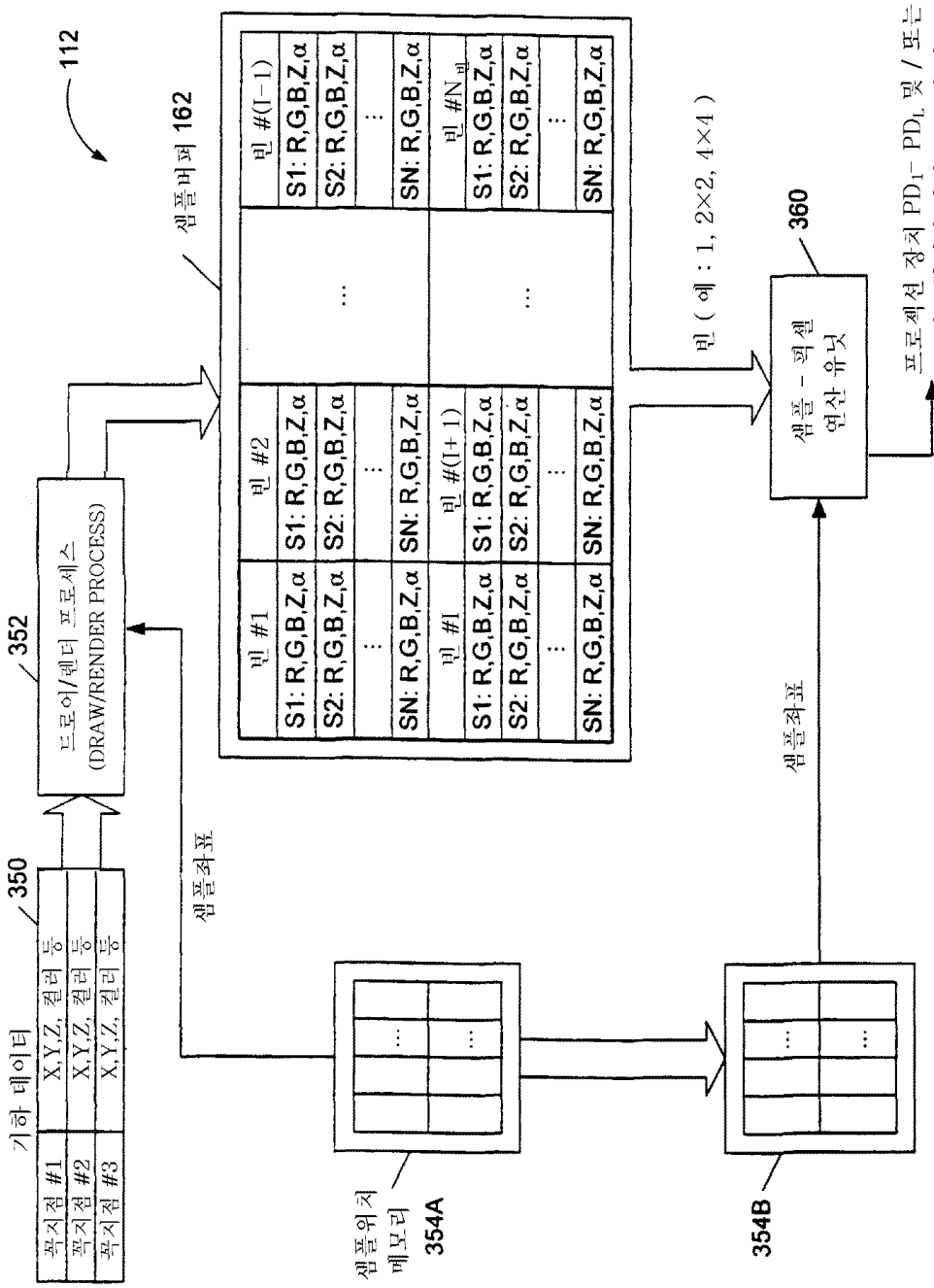
도면 5B



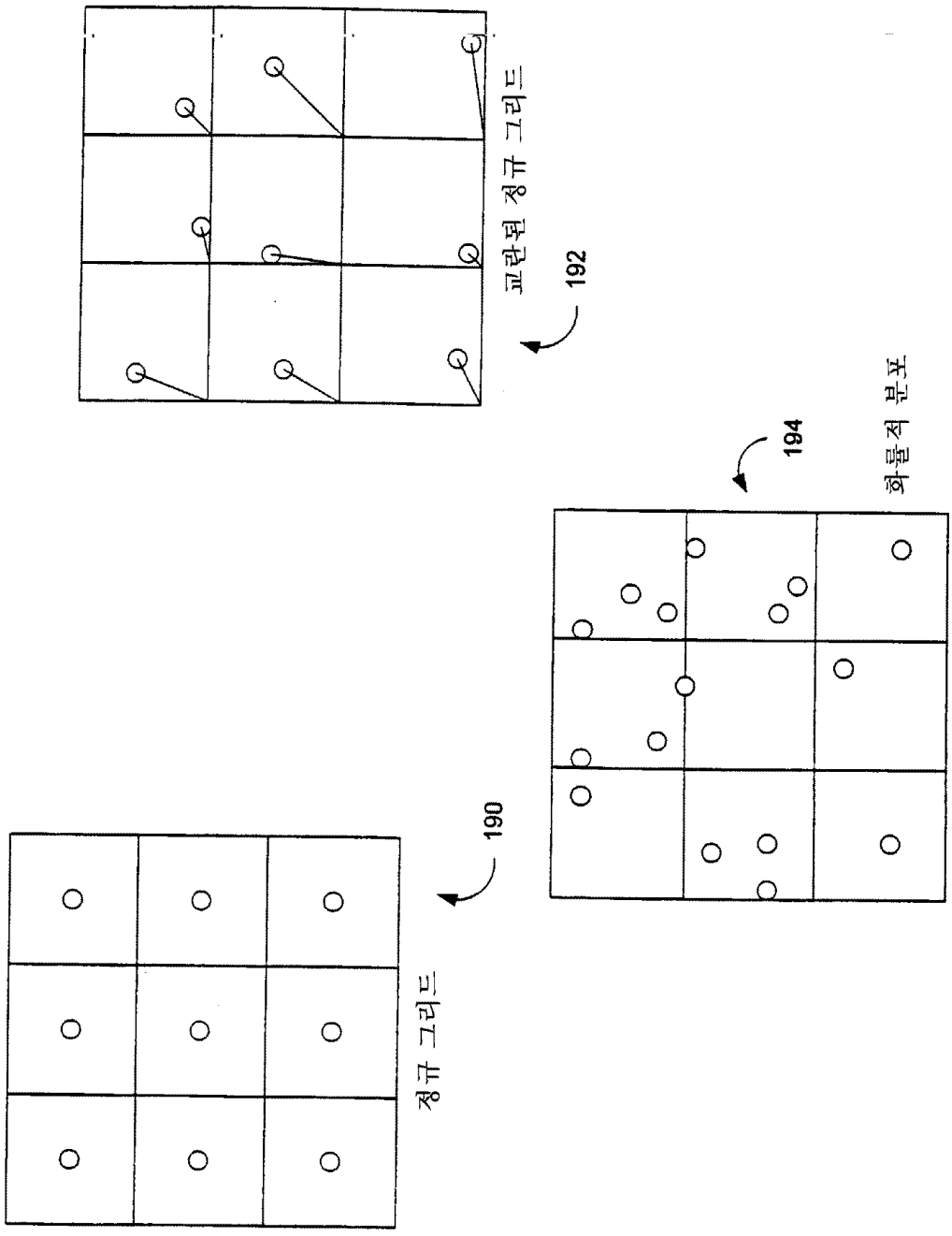
도면 6


112
기하 데이터
350
꼭지점 #1 X,Y,Z, 컬러 등
꼭지점 #2 X,Y,Z, 컬러 등
꼭지점 #3 X,Y,Z, 컬러 등
352
드로어/렌더 프로세스
(DRAW/RENDER PROCESS)
샘플버퍼 162
빈 #1
빈 #2
빈 #(I+1)
빈 #(I-1)
빈 #N빈
S1: R,G,B,Z,α
S2: R,G,B,Z,α
SN: R,G,B,Z,α
샘플좌표
샘플위치 메모리 354
빈#1을 위한 오프셋
정규 그리드
교란된 정규 그리드
확률적 분포
S1: dX,dY
S2: dX,dY
SN: dX,dY
빈 (예 : 1, 2×2, 4×4)
360
샘플 - 픽셀 연산 유닛
프로젝션 장치 PD_1- PD_L 및 / 또는 디스플레이 장치 84로 출력

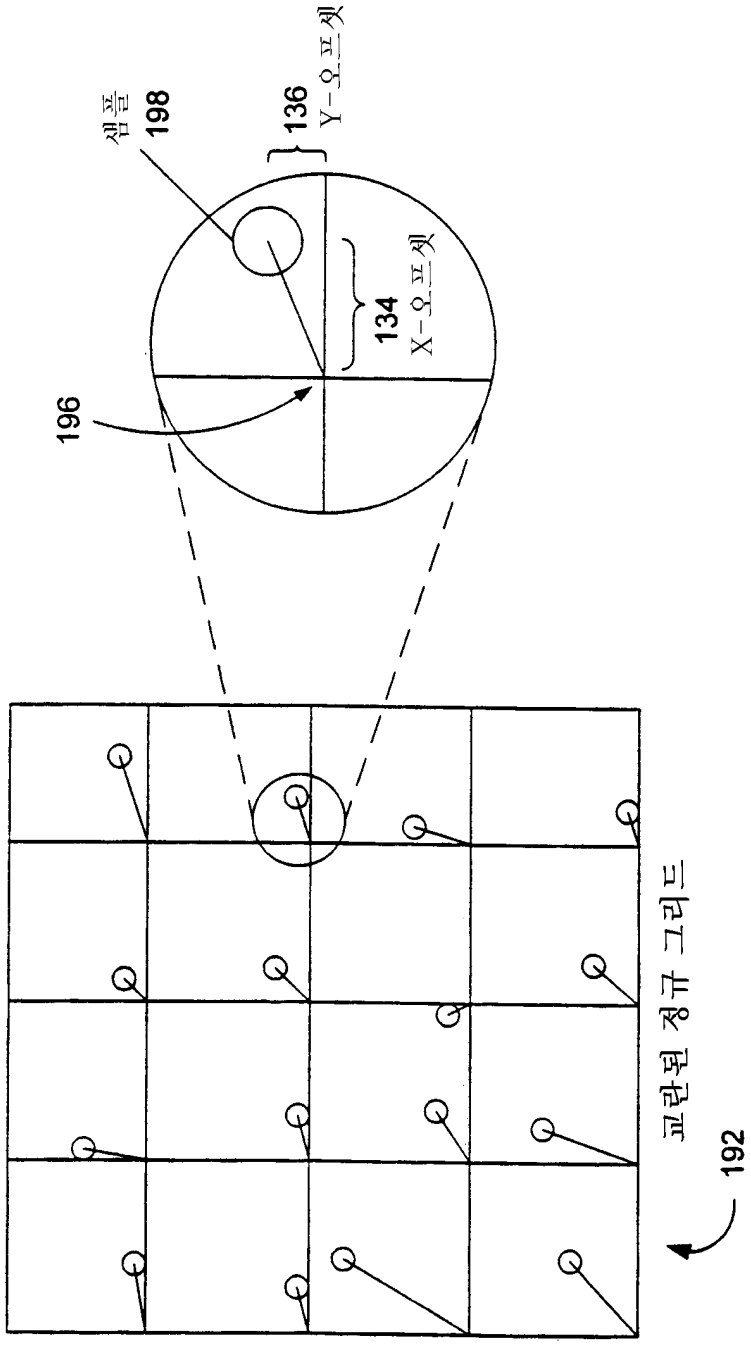
도면 7


112
샘플버퍼 162
빈 #1
빈 #2
빈 #(I-1)
빈 #I
빈 #(I+1)
빈 #N빈
S1: R,G,B,Z,α
S2: R,G,B,Z,α
SN: R,G,B,Z,α
352
드로잉/렌더 프로세스
(DRAW/RENDER PROCESS)
350
기하 데이터
꼭지점 #1
꼭지점 #2
꼭지점 #3
X,Y,Z, 컬러 등
샘플좌표
샘플위치 메모리
354A
354B
빈 (예 : 1, 2×2, 4×4)
360
샘플 - 픽셀 연산 유닛
프로젝션 장치 PD1- PDL 및 / 또는 디스플레이 장치 84로 출력

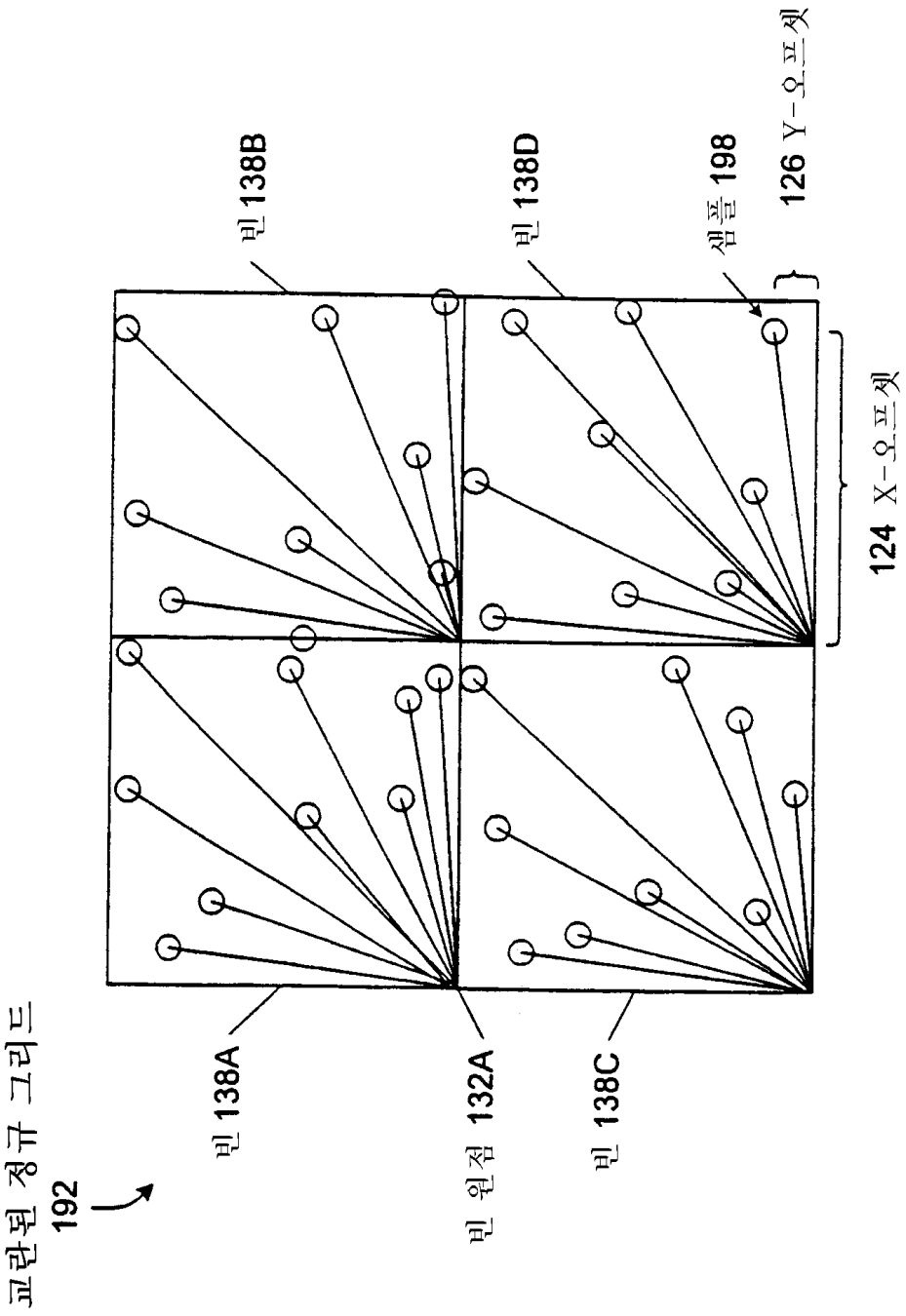
도면 8



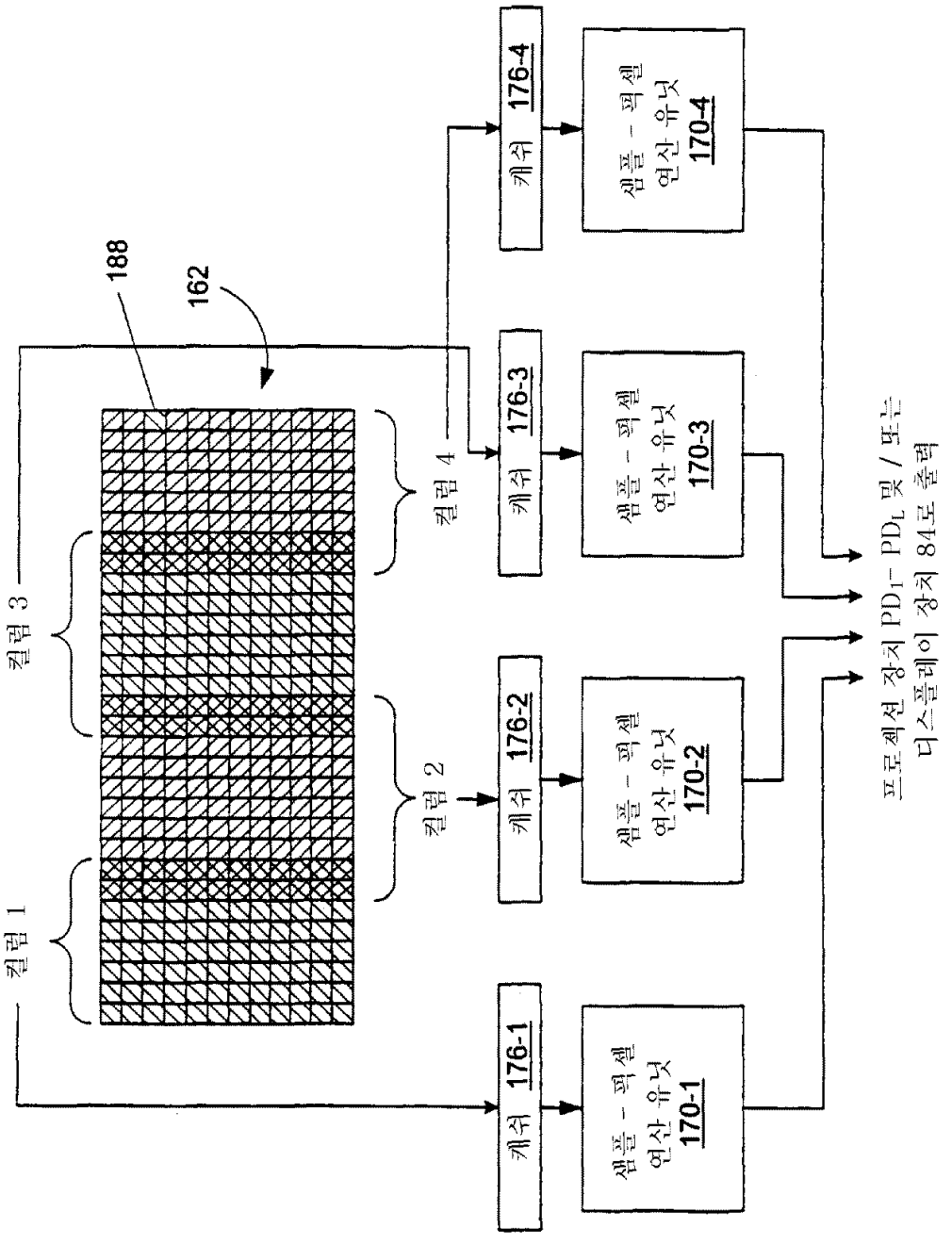
도면 9



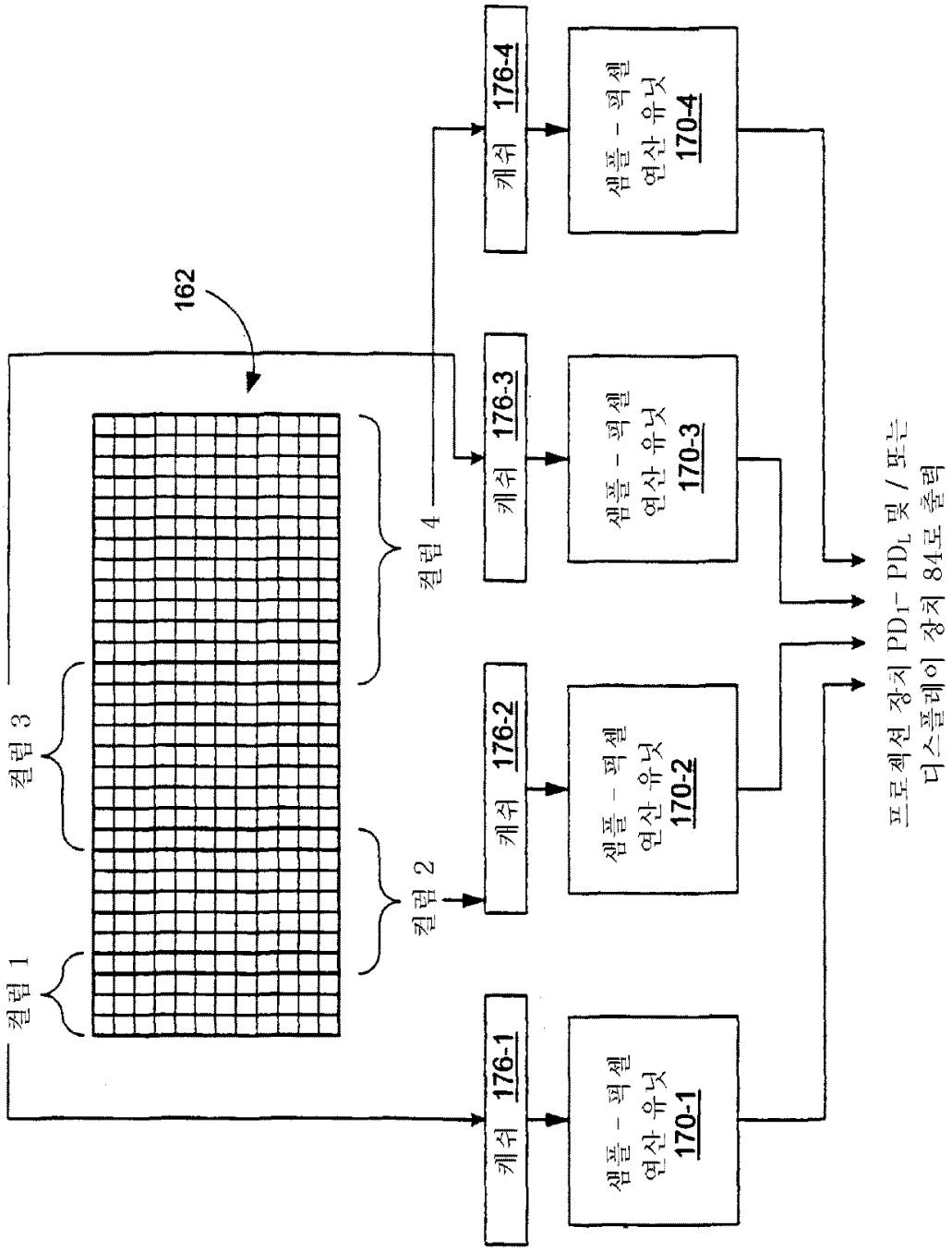
도면 10


교란된 정규 그리드
192
빈 138A
빈 원점 132A
빈 138C
빈 138B
빈 138D
샘플 198
126 Y-오프셋
124 X-오프셋

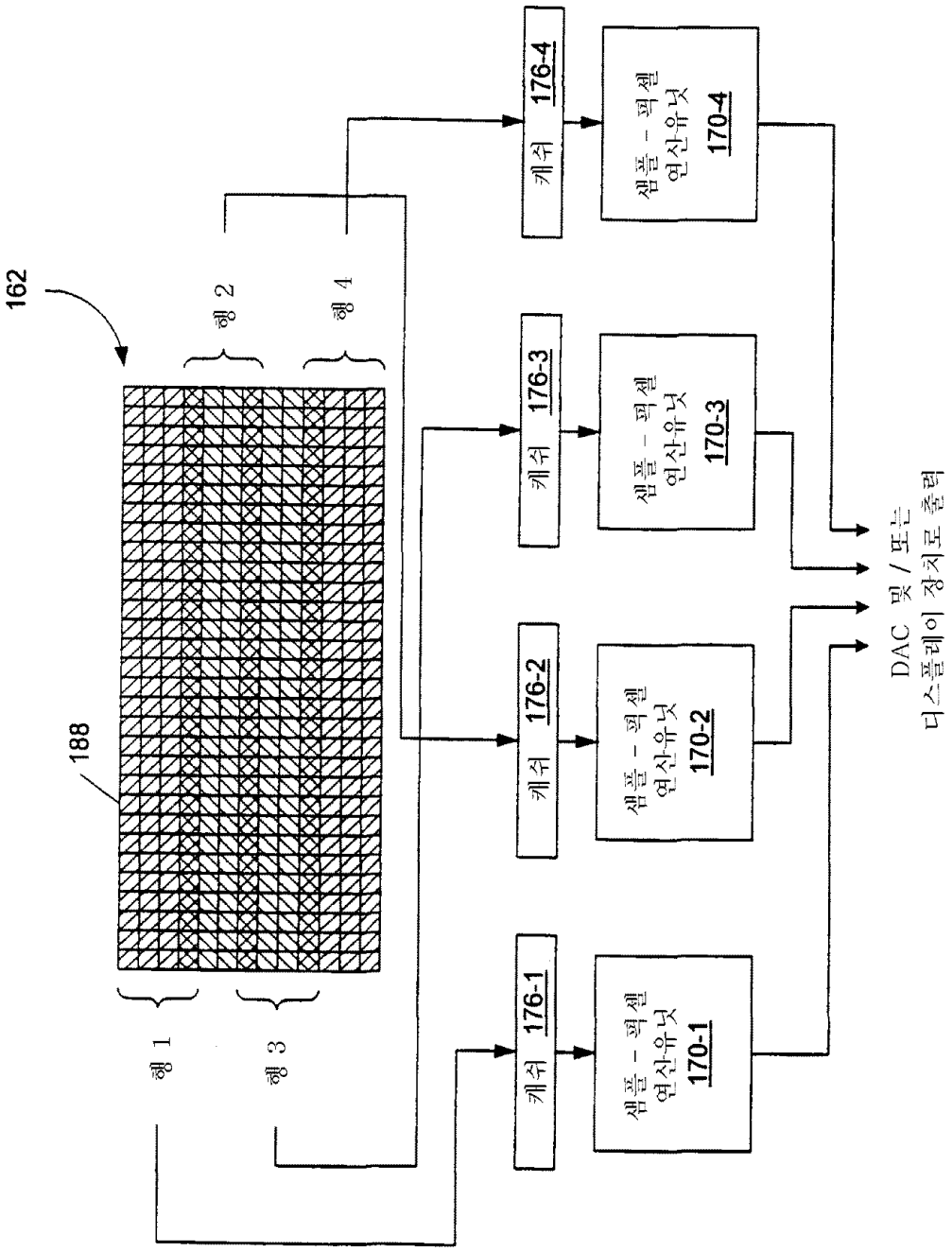
도면 11A


188
162
컬럼 1
컬럼 3
컬럼 2
컬럼 4
캐쉬 176-1
캐쉬 176-2
캐쉬 176-3
캐쉬 176-4
샘플 - 픽셀 연산 유닛 170-1
샘플 - 픽셀 연산 유닛 170-2
샘플 - 픽셀 연산 유닛 170-3
샘플 - 픽셀 연산 유닛 170-4
프로젝션 장치 PD_1- PD_L 및 / 또는 디스플레이 장치 84로 출력

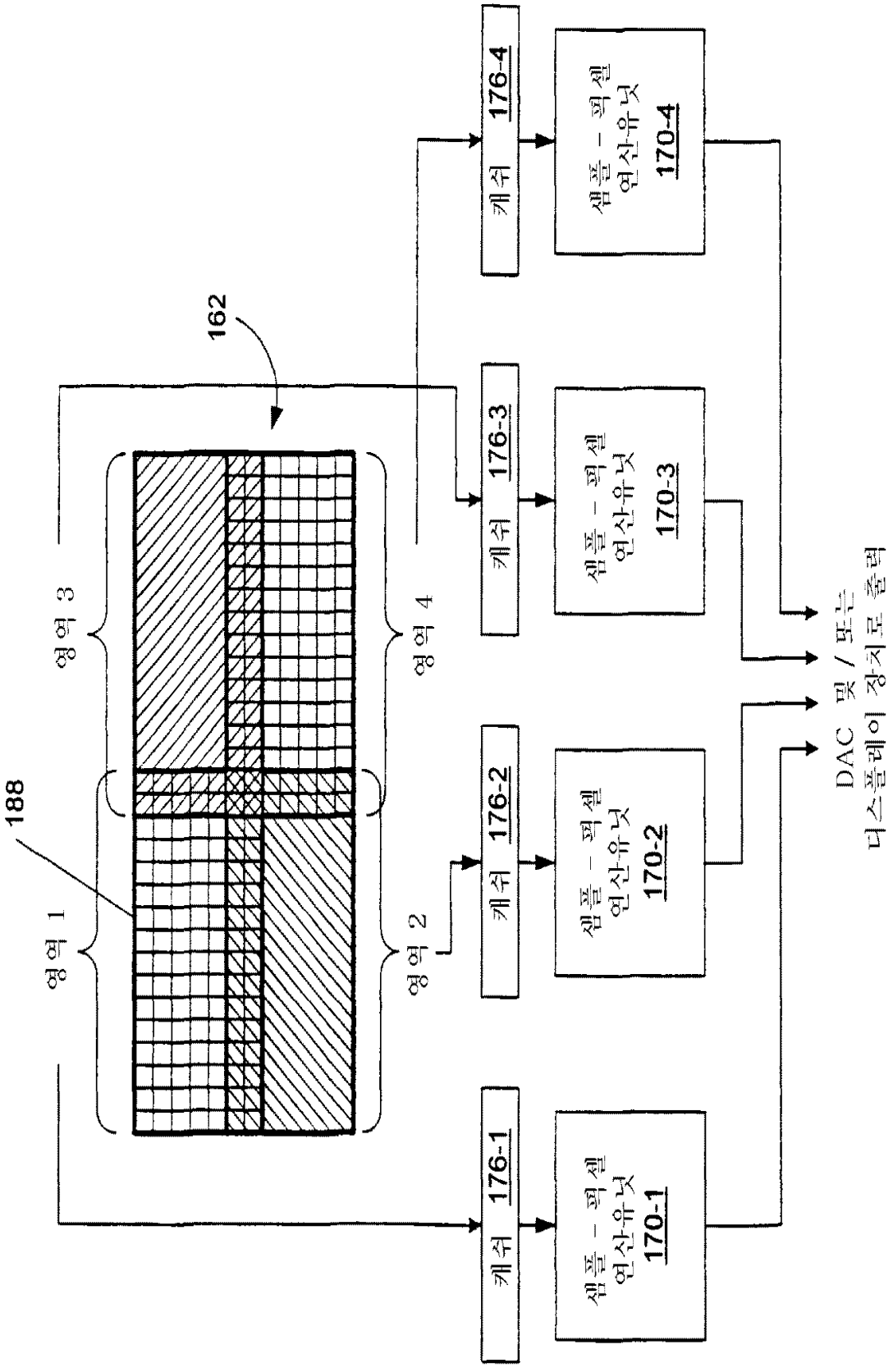
도면 11B


162
컬럼 1
컬럼 3
컬럼 2
컬럼 4
캐쉬 176-1
캐쉬 176-2
캐쉬 176-3
캐쉬 176-4
샘플 - 픽셀 연산 유닛 170-1
샘플 - 픽셀 연산 유닛 170-2
샘플 - 픽셀 연산 유닛 170-3
샘플 - 픽셀 연산 유닛 170-4
프로젝션 장치 PD_T - PD_L 및 / 또는 디스플레이 장치 84로 출력

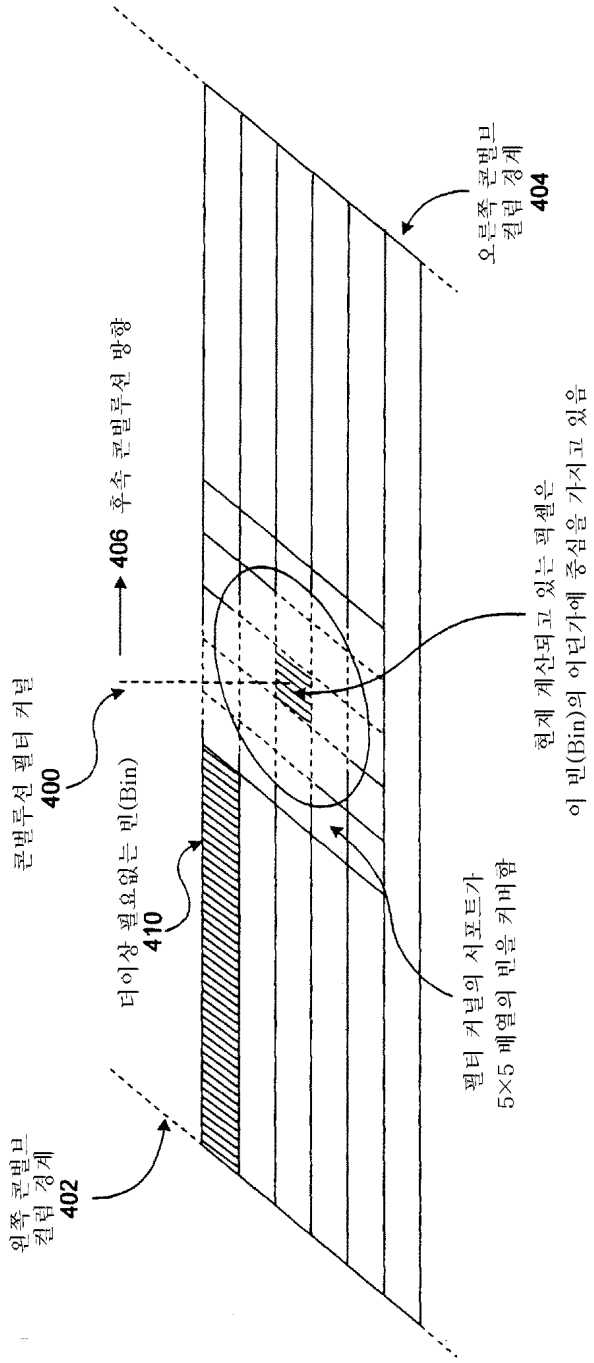
도면 12


162
188
행 1
행 2
행 3
행 4
캐쉬 176-1
캐쉬 176-2
캐쉬 176-3
캐쉬 176-4
샘플-픽셀 연산유닛 170-1
샘플-픽셀 연산유닛 170-2
샘플-픽셀 연산유닛 170-3
샘플-픽셀 연산유닛 170-4
DAC 및/또는 디스플레이 장치로 출력

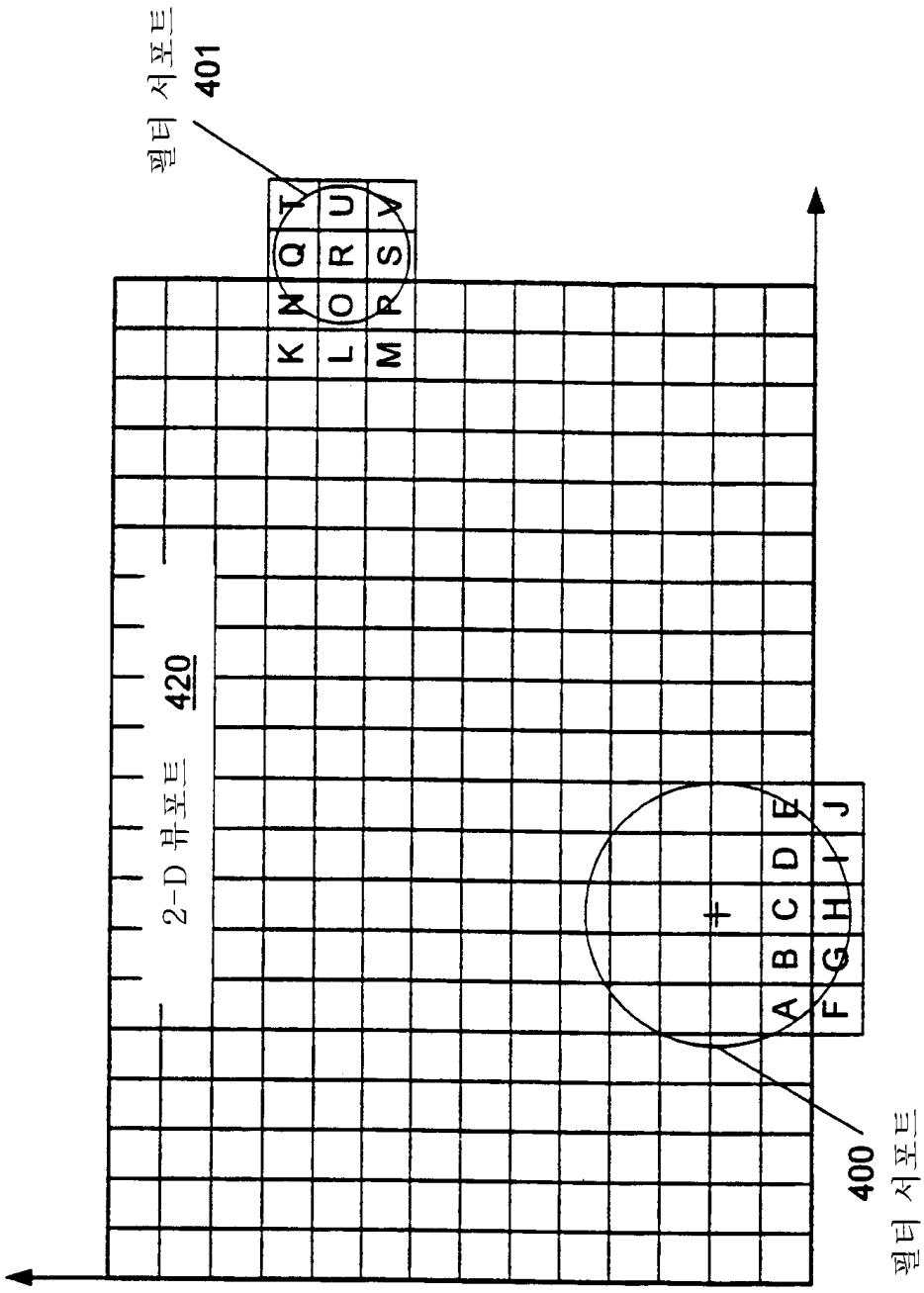
도면 13


162
188
영역 1
영역 2
영역 3
영역 4
캐쉬 176-1
캐쉬 176-2
캐쉬 176-3
캐쉬 176-4
샘플-픽셀 연산유닛 170-1
샘플-픽셀 연산유닛 170-2
샘플-픽셀 연산유닛 170-3
샘플-픽셀 연산유닛 170-4
DAC 및/또는 디스플레이 장치로 출력

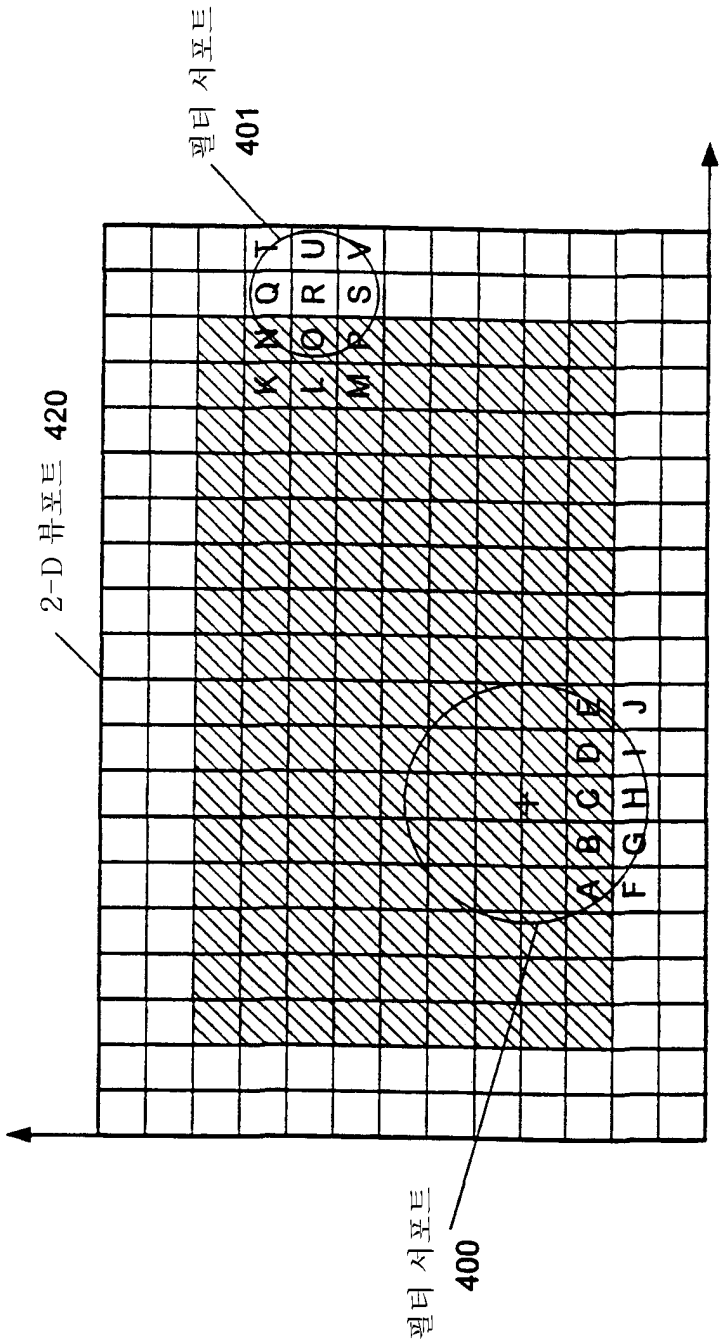
도면 14


왼쪽 콘벌브
컬럼 경계
402
콘벌루션 필터 커널
400
더이상 필요없는 빈(Bin)
410
406 후속 콘벌루션 방향
오른쪽 콘벌브
컬럼 경계
404
필터 커널의 서포트가
5×5 배열의 빈을 커버함
현재 계산되고 있는 픽셀은
이 빈(Bin)의 어딘가에 중심을 가지고 있음

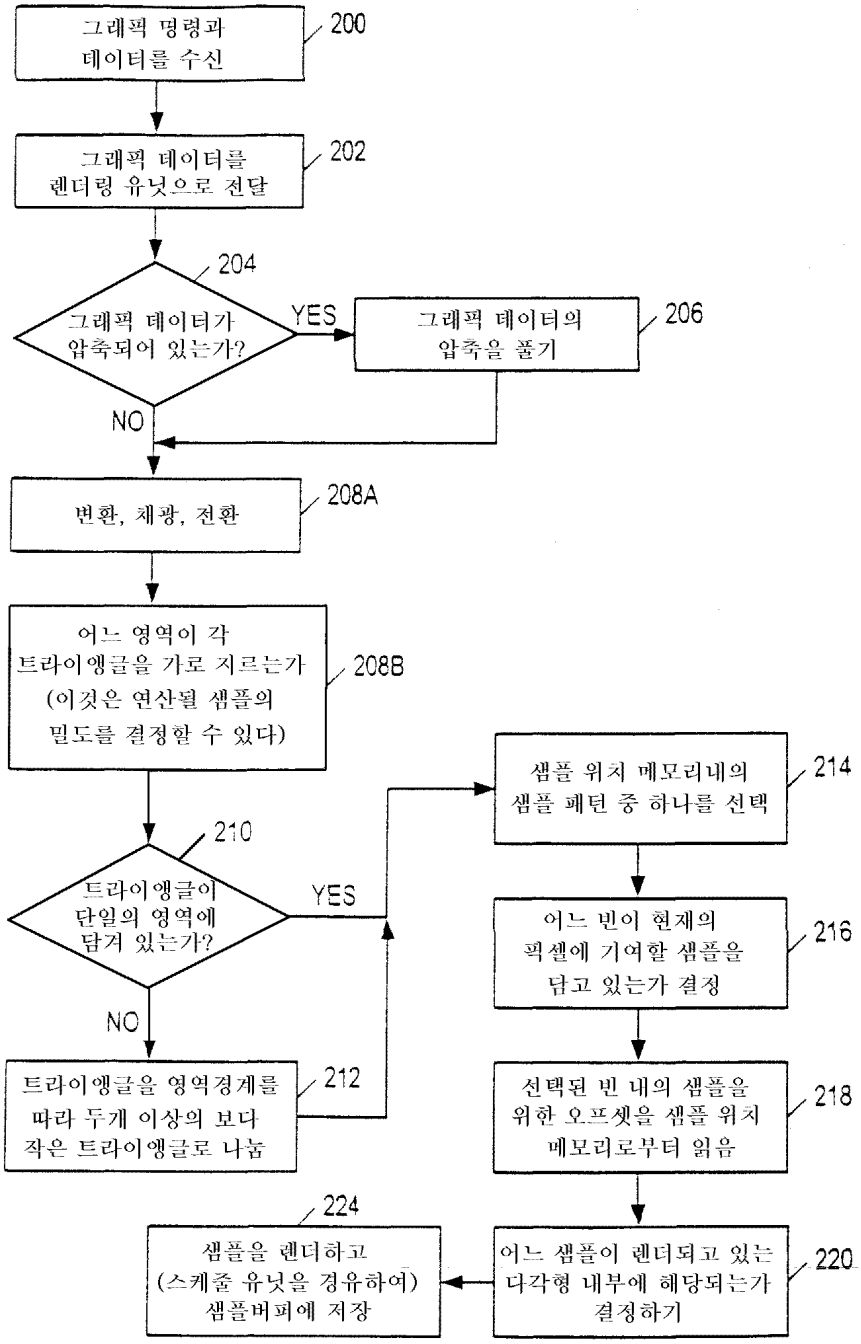
도면 15



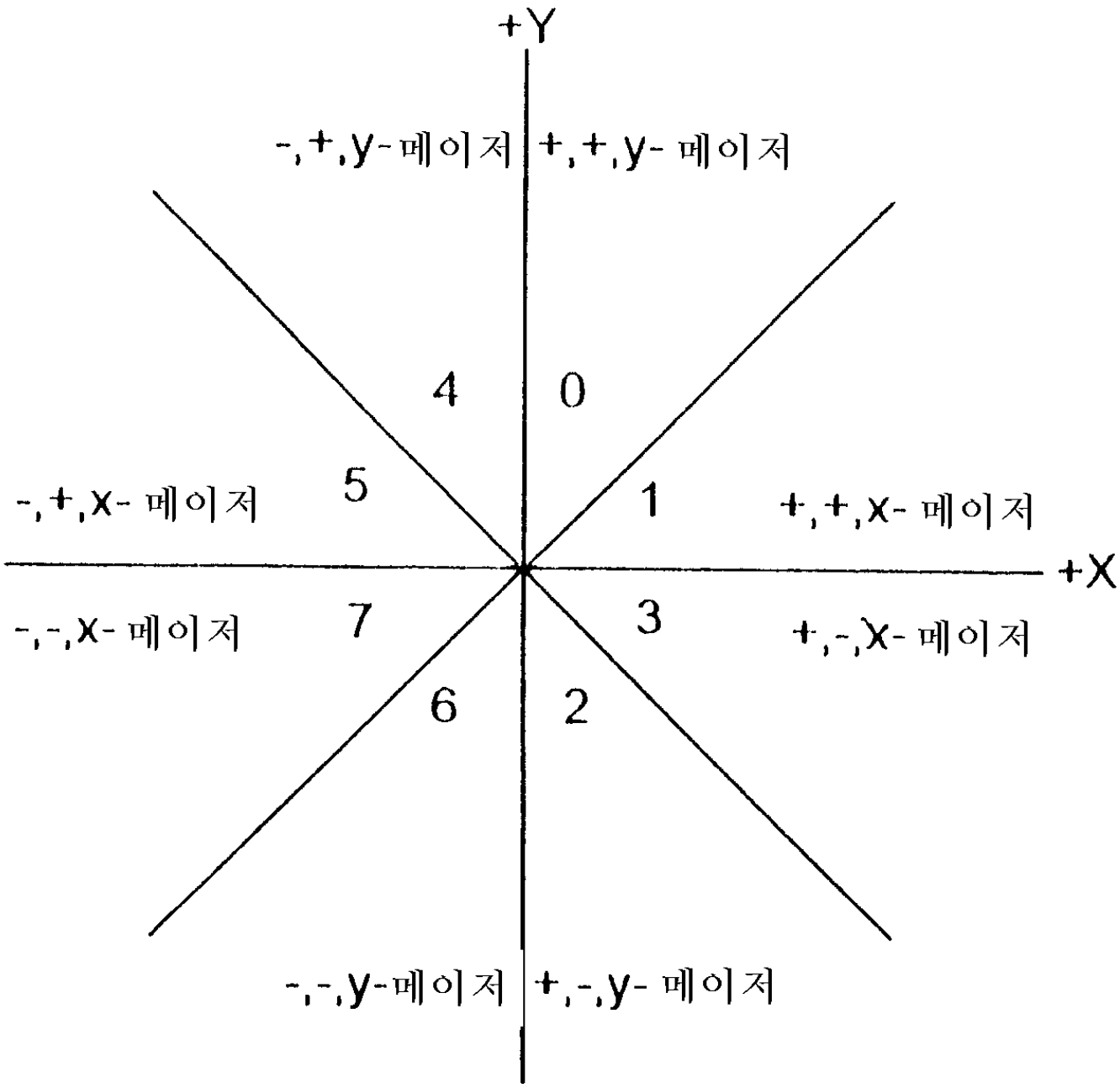
도면 16



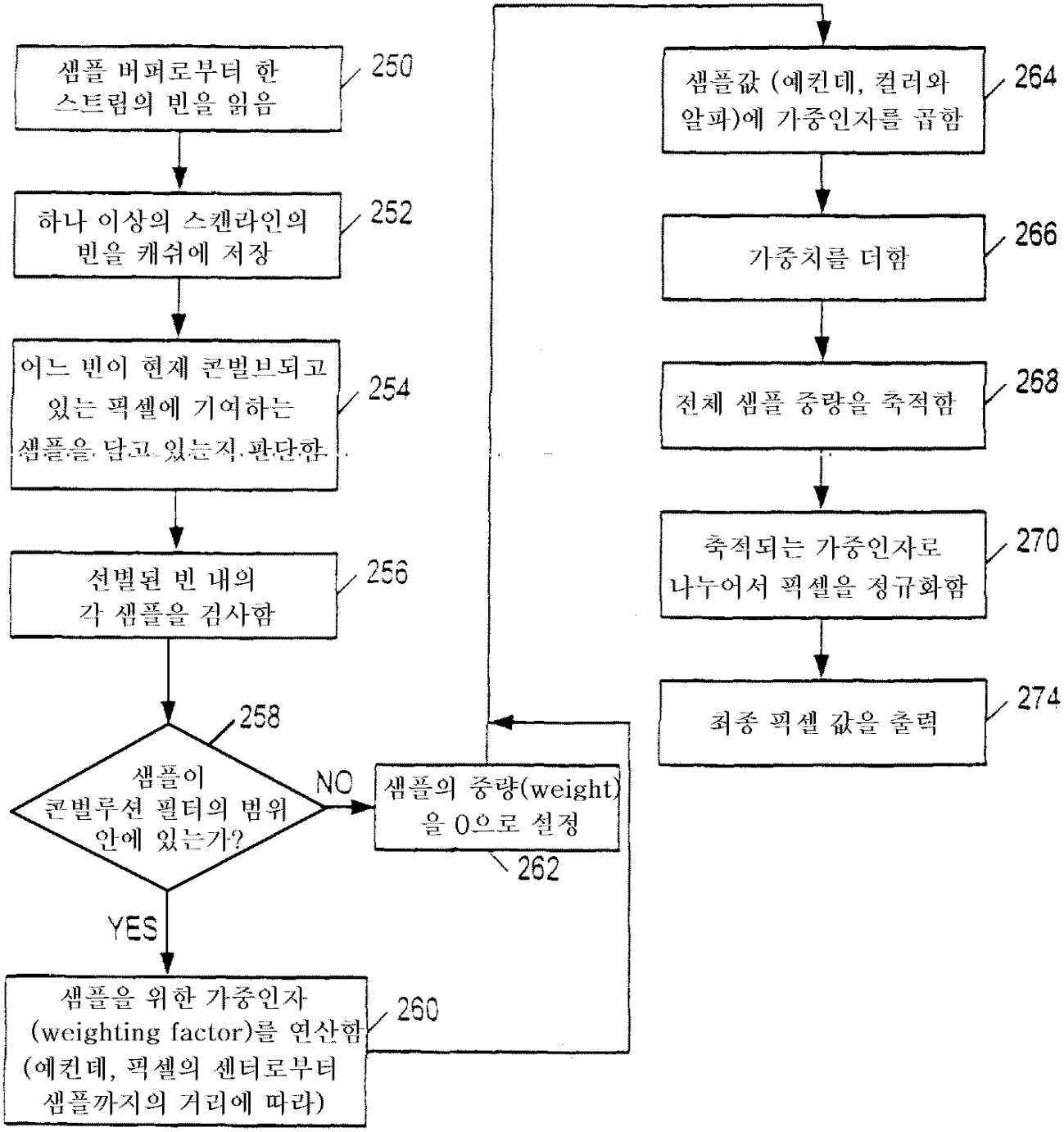
도면 17


그래픽 명령과 데이터를 수신
200
그래픽 데이터를 렌더링 유닛으로 전달
202
204
그래픽 데이터가 압축되어 있는가?
YES
그래픽 데이터의 압축을 풀기
206
NO
변환, 채광, 전환
208A
어느 영역이 각 트라이앵글을 가로 지르는가 (이것은 연산될 샘플의 밀도를 결정할 수 있다)
208B
210
트라이앵글이 단일의 영역에 담겨 있는가?
YES
NO
트라이앵글을 영역경계를 따라 두개 이상의 보다 작은 트라이앵글로 나눔
212
샘플 위치 메모리내의 샘플 패턴 중 하나를 선택
214
어느 빈이 현재의 픽셀에 기여할 샘플을 담고 있는가 결정
216
선택된 빈 내의 샘플을 위한 오프셋을 샘플 위치 메모리로부터 읽음
218
어느 샘플이 렌더되고 있는 다각형 내부에 해당되는가 결정하기
220
224
샘플을 렌더하고 (스케줄 유닛을 경유하여) 샘플버퍼에 저장

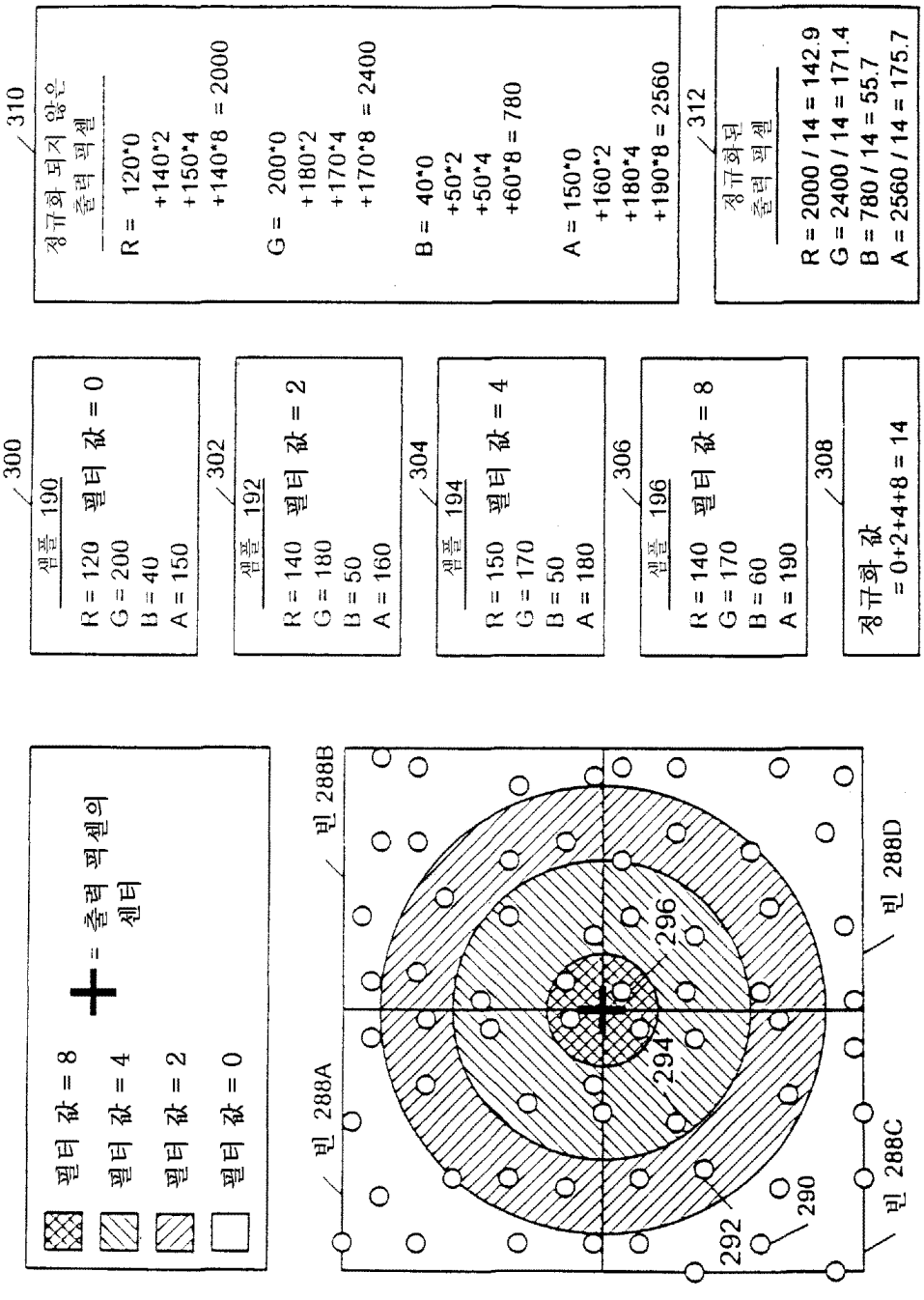
도면 18



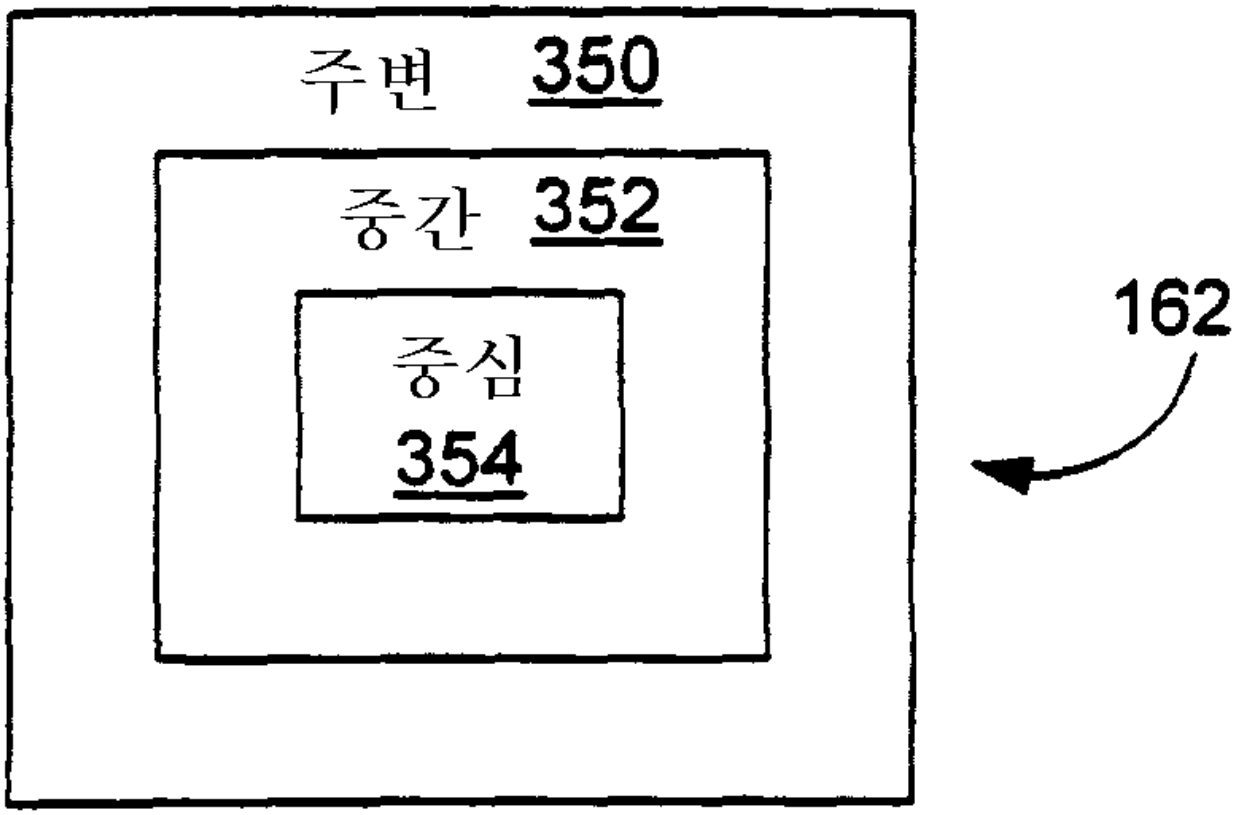
도면 19


샘플 버퍼로부터 한 스트림의 빈을 읽음
250
하나 이상의 스캔라인의 빈을 캐쉬에 저장
252
어느 빈이 현재 콘벌브되고 있는 픽셀에 기여하는 샘플을 담고 있는지 판단함
254
선별된 빈 내의 각 샘플을 검사함
256
샘플이 콘벌루션 필터의 범위 안에 있는가?
258
NO
YES
샘플의 중량(weight)을 0으로 설정
262
샘플을 위한 가중인자(weighting factor)를 연산함 (예컨데, 픽셀의 센터로부터 샘플까지의 거리에 따라)
260
샘플값 (예컨데, 컬러와 알파)에 가중인자를 곱함
264
가중치를 더함
266
전체 샘플 중량을 축적함
268
축적되는 가중인자로 나누어서 픽셀을 정규화함
270
최종 픽셀 값을 출력
274

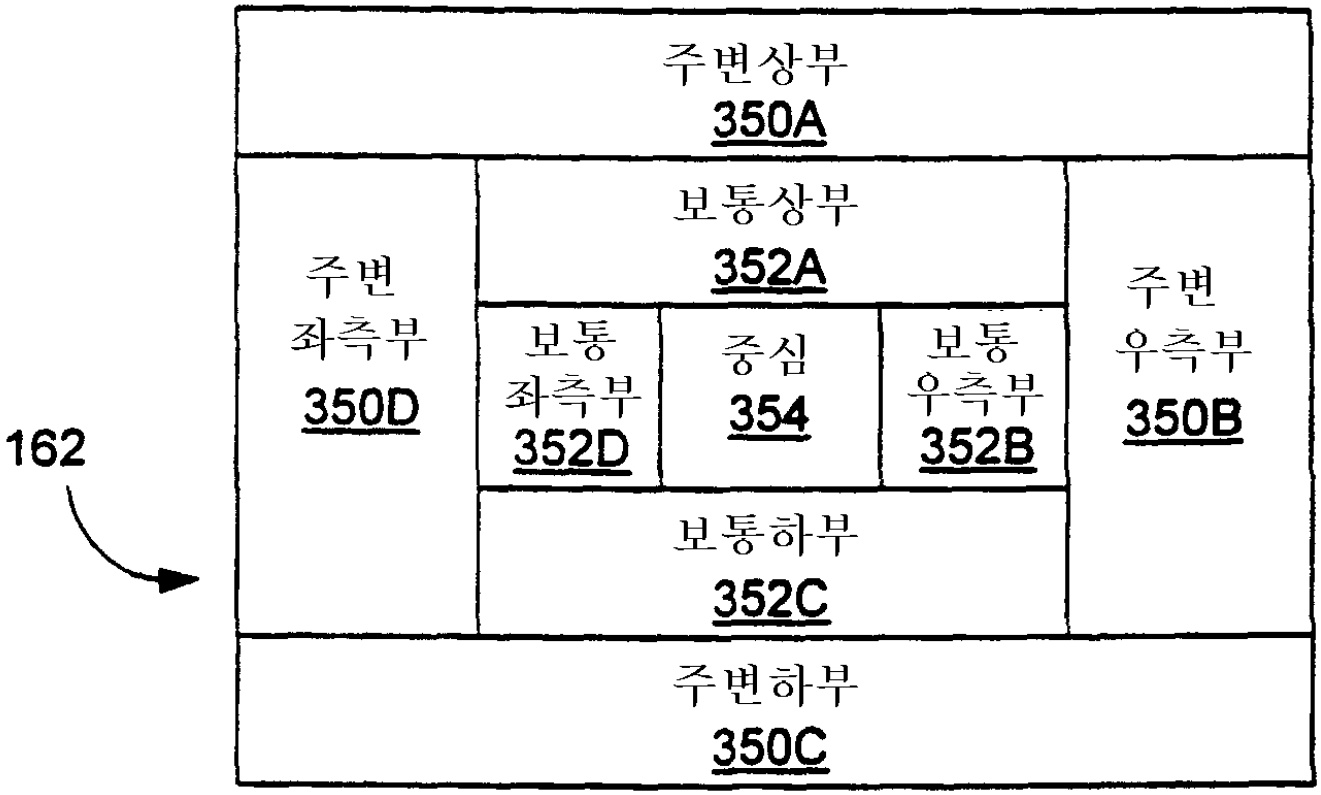
도면 20


310
정규화 되지 않은 출력 픽셀
R = 120*0 +140*2 +150*4 +140*8 = 2000
G = 200*0 +180*2 +170*4 +170*8 = 2400
B = 40*0 +50*2 +50*4 +60*8 = 780
A = 150*0 +160*2 +180*4 +190*8 = 2560
312
정규화된 출력 픽셀
R = 2000 / 14 = 142.9
G = 2400 / 14 = 171.4
B = 780 / 14 = 55.7
A = 2560 / 14 = 175.7
300
샘플 190
R = 120 필터 값 = 0
G = 200
B = 40
A = 150
302
샘플 192
R = 140 필터 값 = 2
G = 180
B = 50
A = 160
304
샘플 194
R = 150 필터 값 = 4
G = 170
B = 50
A = 180
306
샘플 196
R = 140 필터 값 = 8
G = 170
B = 60
A = 190
308
정규화 값
= 0+2+4+8 = 14
필터 값 = 8
필터 값 = 4
필터 값 = 2
필터 값 = 0
= 출력 픽셀의 센터
빈 288A
빈 288B
빈 288C
빈 288D
290
292
294
296

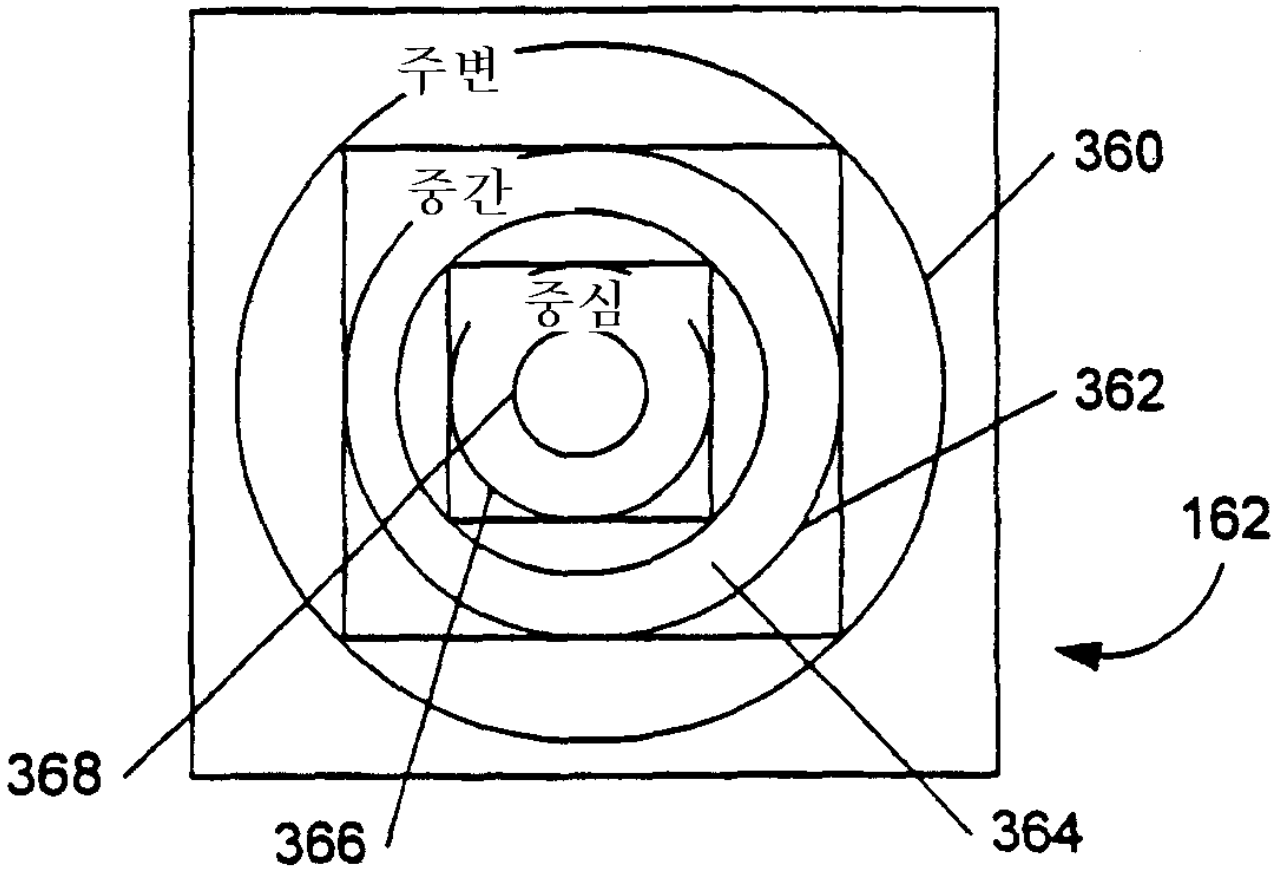
도면 21


주변 350
중간 352
중심 354
162

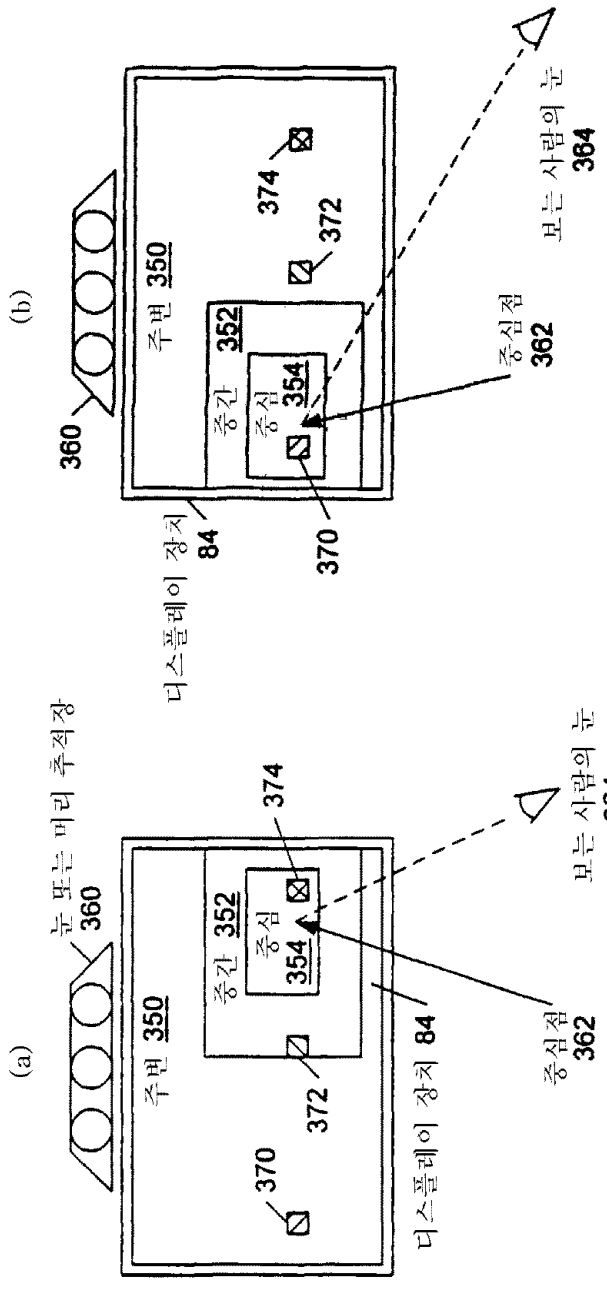
도면 22


주변상부 350A
보통상부 352A
주변 좌측부 350D
보통 좌측부 352D
중심 354
보통 우측부 352B
주변 우측부 350B
보통하부 352C
주변하부 350C
162

도면 23



도면 24



☒ 중심영역 = 빈마다 8개 샘플
코벌루션 반경은 4개의 빈과 접함.
전체 = 32개 샘플이 기여할 것임.

☒ 주변영역 = 빈마다 1개 샘플
콘벌루션 반경은 1개의 빈과 접함.
전체 = 1개 샘플이 기여할 것임.

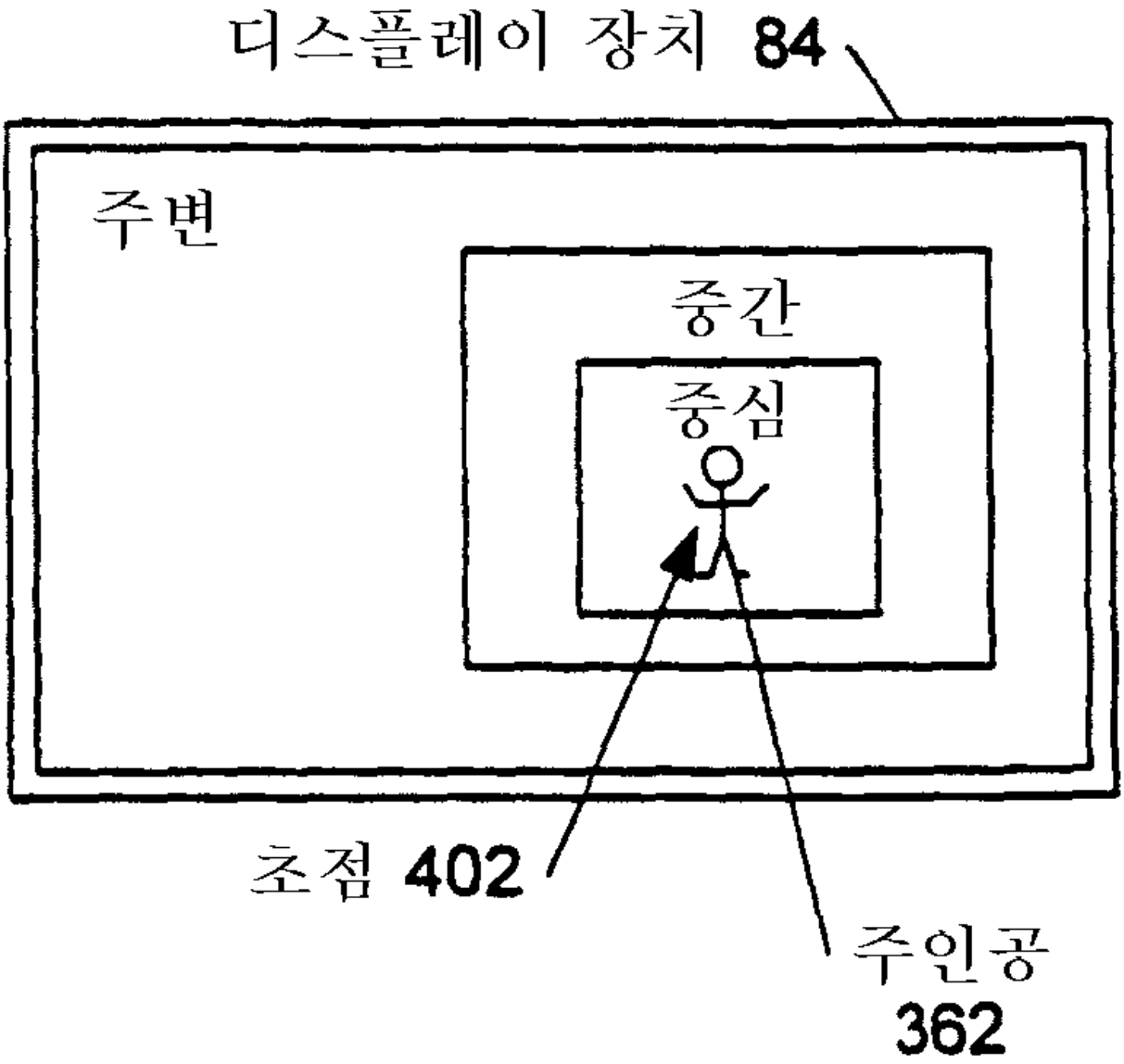
☒ 중간영역 = 빈마다 4개 샘플
코벌루션 반경은 4개의 빈과 접함.
전체 = 16개 샘플이 기여할 것임.

☒ 주변영역 = 빈마다 1개 샘플
콘벌루션 반경은 1개의 빈과 접함.
전체 = 32개 샘플이 기여할 것임.

☒ 주변영역 = 빈마다 1개 샘플
코벌루션 반경은 1개의 빈과 접함.
전체 = 1개 샘플이 기여할 것임.

☒ 중심영역 = 빈마다 8개 샘플
콘벌루션 반경은 4개의 빈과 접함.
전체 = 32개 샘플이 기여할 것임.

도면 25A



도면 25B