회화적 렌더링에서 움직임을 따라 회전하는 붓질 기법

Rotating Brush Strokes to Track Movement for Painterly Rendering

한정훈*, 기현우**, 김효원***, 오경수****
숭실대학교

요약 ˘ ˘ 회화는 2차원 평면 위에 색과 선을 사용하여 여러 가지 형상을 표현하는 조형예술이다. 본 논문에서는 이러한 회화의 평면적 특성에 입각하여 3차원 물체들로 구성된 장면을 화가가 캔버스 위에 붓을 놀려 채색한 것과 같은 회화 스타일로 렌더링하는 방법을 소개하고, 카메라의 시점과 시선이 변화하는 동적인 장면에서 여전히 평면적 특성을 유지하면서 붓질의 방향이 처음에 지정된 방향을 유지하도록 변화하는 방법을 제안한다. 회화의 정의에 따라 캔버스와 같은 2차원 평면 위에 붓으로 칠한듯한 효과를 연출하기 위해서 본 논문에서는 화면 공간 위에서 크기가 동일한 빌보드를 사용하여 붓질-붓을 한번 놀려 색을 칠하는 것-을 한다. 화면 전체에 붓질을 하여 장면을 렌더링 하여도 장면을 바라보는 카메라가 움직이면 처음의 붓질 방향을 유지하기 위하여 붓질의 방향 역시 변하여야 한다. 만일 붓질이 변하지 않고 동일한 방향을 유지한다면 마치 일정한 붓질 모양의 패턴이 있는 유리 뒤로 물체들이 움직이는 것 같은 시각적 오류가 있는 결과를 얻게 된다. 이것을 막기 위하여 본 논문에서는 장면 안에서 시점이나 시선의 방향이 바뀌는 애니메이션이 일어날 때 그에 맞춰 물체 위의 붓질이 함께 회전하는 방법을 제안한다. 붓질이 회전할 각도는 첫 프레임의 장면과 현재 프레임의 장면의 샘플 포인트들에 대하여 위치 차이를 비교하는, least-square solution을 사용하는 Horn의 2차원 유사성 검사를 수행하여 얻는다. 본 논문에서는 실험을 통해, 처음에 회화 스타일로 렌더링된 장면에서 실시간으로 카메라를 움직이며 붓질의 방향이 변화하는 모습을 관찰하였고, 처음에 지정된 방향을 유지하도록 회전하는 것을 확인하였다.

Abstract ˘ ˘ We introduce a method of rendering a scene lying 3D objects which is like that artist draw on a canvas by brushing. Painting is the art area presenting something created by color and line on 2D plane. We are brushing on billboards on screen space for the 2D brushing effect according to the definition of "Painting". Brushing orientation is haven to rotate for preventing the orientation in the first scene in the case that object or camera are moving. If the brushing isn't rotated, shower-door effect is watched on the scene as undesirable result. We present a brushing rotating method for keeping the orientation changing the direction of view and object rigid animation. The brushing direction is computed with Horn's 2D similarity transform by least-square solution. We watched the changing brushing to track the motion of object and view.

핵심어: rotating brushing, 2D style painting, Non Photo-realistic Rendering

본 논문은 2007년 숭실대학교 '서울시 산학연 협력사업(10581cooperateOrg93112)' 학술 연구비 지원에 의하여 연구되었음.
*주저자 : 숭실대학교 미디어학과 석사과정; e-mail: duckweed@ssu.ac.kr
**공동저자 : 숭실대학교 미디어학과 석사; e-mail: kih@ssu.ac.kr
***공동저자 : 숭실대학교 미디어학과 석사과정; e-mail: blueciel@ssu.ac.kr
****교신저자 : 숭실대학교 미디어학과 교수; e-mail: oks@ssu.ac.kr

1. 소개

렌더링은 장면의 사실성 획득을 위하여 실사의 재현을 목적으로 하는 실사 렌더링(Photorealistic Rendering)과 표현의 다양성 획득을 위한 기법의 개발을 목적으로 하는 비실사 렌더링(Non Photo-Realistic Rendering)으로 발전하여 왔다. 특히 비실사 렌더링 분야는 장면을 추상화하거나, 기존의 예술적 표현 방법과 수단을 컴퓨터로 모사하여 왔다. 만화 스타일로 렌더링하기 위해 단순하며 극단적인 명암 변화와 경계선을 추가하는 여러 가지 기법들, 유화와 수채화 등과 같은 실제의 회화를 모사하기 위하여 재료와 도구, 사람의 기술을 분석하고 재현해내는 기법들 등이 연구되어 왔다.

본 논문에서는 방향을 가지고 있는 붓질을 물체 위에 배치시켜 회화 스타일로 렌더링된 장면 안에서 카메라가 실시간으로 움직일 때 붓질의 방향이 회전할 방향을 계산하는 방법을 제안한다.

회화는 2차원 평면 위에 색과 선을 사용하여 여러 가지 형상을 표현하는 조형예술이다[1]. 일단 붓질을 사용하여 3차원 장면을 2차원 회화의 형태로 표현을 하였더라도 장면을 바라보는 카메라의 시점과 시선의 방향이 바뀌는 동적인 장면에서 계속하여 평면적 특성을 유지하기 위해서는 붓질의 위치와 방향, 크기를 적절히 조절해야 한다.

붓질의 방향이 애니메이션에 상관없이 일정한 방향으로 고정되어 있다면 위치가 고정되어 있는 것과 유사하게 화면 위에 특정 패턴이 있는 것처럼 보일뿐더러, 처음에 화면에 투영에 물체의 모양에 따라 적절히 배치하여 물체와 붓질이 대응하도록 하였어도 바뀐 장면에서는 동일한 물체 위의 붓질이 처음과 달리 물체와 대응하지 않게 된다.[그림 1 우측하단]. 처음에 배치한 붓질의 방향, 즉 사용자의 의도가 유지되도록 하기 위해서는 물체의 움직임에 따라 물체 위의 붓질의 방향이 함께 변하고, 시점의 변화에 따라 화면 전체의 붓질의 방향이 변하여야 한다[그림 1 좌측하단]. 예를 들어 나무의 결에 따라 붓질을 배치했다면 나무가 옆으로 쓰러져도 붓질은 처음의 상태와 같이 나무의 결에 따른 방향을 가지고 있어야 하고, 가상의 카메라가 갸웃거리며 움직이면-시선 방향을 축으로 삼아 회전하면- 화면 전체의 붓질이 시선의 움직임에 따라 회전해야 원래의 붓질 방향이 유지된다.

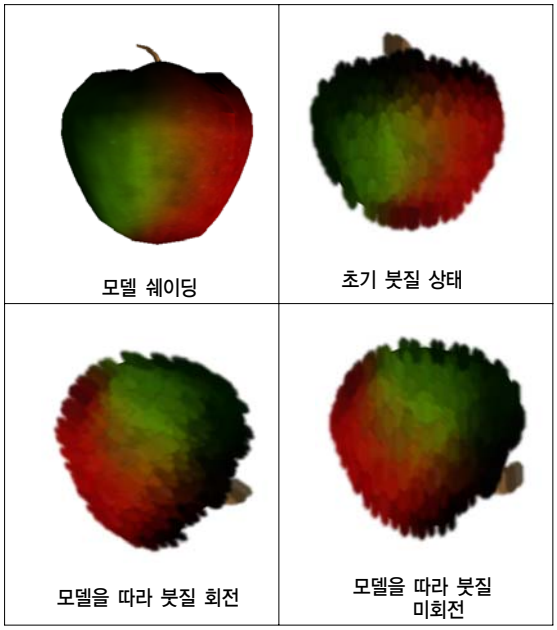


그림 1. 붓질의 방향

Meier[3]는 붓질의 방향을 결정하기 위하여 물체의 모양에 따른 참조 이미지를 생성하여 이에 따라 방향이 변하게 하였다. 하지만 사용자가 장면 안의 물체에 대하여 항상 물체의 기하학적 구조에 따라 붓질을 배치하기를 원하지는 않을 것이며 각각의 물체에 사용자가 원하는 붓질을 주기 위해서는 물체마다 참조 이미지를 만들어주어야 한다.

본 논문에서는 방향에 대한 별도의 참조 이미지 없이 사용자가 처음의 장면에 대한 붓질을 결정하면 지속적으로 그 붓질의 방향을 유지하는 방법을 제안한다. 다시 말해 처음에 물체와 그 물체 위의 붓질을 대응시켜놓으면 카메라의 위치, 방향에 변화가 일어나도 처음의 물체와 붓질의 대응 상태를 가능한 유지하도록 붓질의 방향이 변화한다. 본 논문에서는 붓질의 방향을 결정하기 위하여 Simon이 사용한 것[2]과 같이, 붓질의 위치가 될 파티클에 대하여 '2차원 유사성 변환' 을 계산하는 Horn의 방법[4]을 사용한다. 구체적으로, 첫 프레임에서 파티클들의 화면 좌표계에서의 위치와 현재 프레임에서 화면 좌표계에서의 위치를 least-squares solution을 사용하여 비교한다. 이 방법을 사용하면 첫 프레임의 상태에서 현재 프레임의 상태로 변환할 때 오차를 최소로 하는 변환을 얻을 수 있다. 얻어진 변환을 이용하면 붓질을 적절하게 회전할 수 있다. [그림 1]과 같이 시선방향을 축으로 회전할 경우에는 거의 완벽하게 붓질이 물체의 움직임을 따라간다.

붓질의 방향을 결정하기 이전에 먼제 물체 위에 붓질을 배치시켜야 한다. 붓질의 위치를 결정하기 위하여 가장 많이 사용되는 방법은 붓질의 위치를 물체의 3차원 공간에서 결

정하는 것이 아니라 화면 공간(screen-space)에서 결정하는 방법이다. 구체적으로 이 방법은 붓질과 같은 채색을 위한 텍스쳐의 좌표나 채색을 위해 사용될 2차원 이미지의 위치를 모델이나 월드 좌표계에서 결정하는 것이 아니라 물체들이 투영된 화면 공간에서 결정하는 방법이다. 이와 달리 현 시점에서 붓질의 이미지로 텍스쳐를 잘 만들어 물체에 맵핑하는 방법도 있는데 이 방법의 경우 시점이 바뀌는 동적인 장면에서는 원근감 때문에 물체의 시각적 평면성이 깨지게 된다. [그림 2]와 같이 원근 투영에 의해 눈에서 멀어질수록 붓질은 작아지고 눈에서 가까울수록 커진다. [그림 3]은 화면 공간에 투영된 물체의 이미지 위에 붓질을 배치시킨 것으로 눈과의 거리와 상관없이 동일한 크기의 붓질을 사용하였다. 화면 위의 붓질의 크기를 일정하게 유지하는 쪽이 그렇지 않은 경우보다 평면성이 잘 지켜지는 것을 알 수 있다. 따라서 본 논문에서는 화면 위에 붓질의 위치를 정하고 그 크기를 항상 일정하도록 렌더링하는 방법을 사용한다.

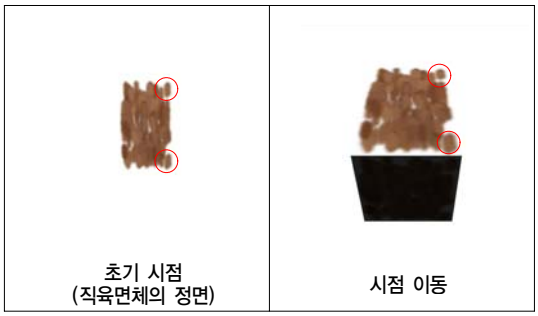


그림 2. 물체 위에 붓질 텍스쳐를 입혀 렌더링한 경우

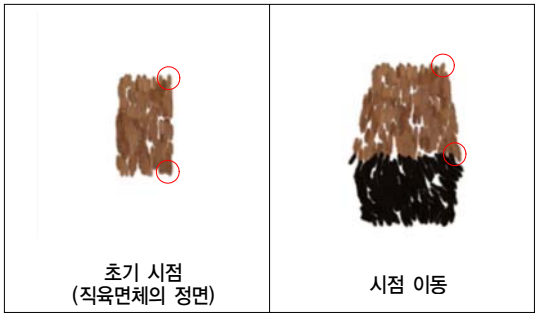


그림 3. 빌보드를 사용하여 물체 위에 붓질을 렌더링한 경우

화면 공간에서 붓질이 놓일 위치를 결정하는 방법도 다양한데, 시점이나 물체의 위치 변화를 고려했을 때 위치를 분산시키는 방법은 크게 매 프레임 화면 위의 임의의 위치에 배치하는 방법, 고정된 지점에 위치를 배치하는 방법으로 나눌 수 있다[2]. 매 프레임 임의로 배치시키는 방법은 프레임간 붓질의 불연속적 위치변화로 인한 깜박임이 발생하고, 붓질의 위치를 고정시키는 방법은 화면 위에 특정패턴이 보이는, 바라지 않는 효과를 발생시킨다.

Meier는 이 두 가지 문제점-깜박임과 패턴-을 해결하기 위하여 '파티클(Particle)'을 사용한다. 물체의 표면에 파티클을 생성하고, 생성된 파티클들을 화면 공간으로 옮겨와 그 위치를 붓질의 위치로 사용한다. 이 방법을 사용하면 매 프레임 붓질의 위치가 변화하면서도 이전 프레임에서의 위치와 현재 프레임에서의 위치가 연속성을 유지한다. 다만 본 논문에서는 속도의 개선을 위해 시선 벡터와 폴리곤의 법선 벡터를 이용하여 폴리곤이 현재 시점에서 눈에 보이는 면적을 계산하여 폴리곤의 안에서 렌더링될 파티클의 수를 조절하기 때문에 약간의 깜박임이 발생한다.

본 논문에서는 파티클을 렌더링하기 위하여 빌보드(Billboard)를 사용하는데, 미리 만들어둔 붓질 이미지를 빌보드에 맵핑하여 파티클을 붓질로써 렌더링한다. 즉, 빌보드의 크기가 붓질의 크기가 된다. 앞서 말한 것과 같이 물체의 2차원적 평면성을 유지하기 위하여 시점과의 거리에 상관없이 화면 공간에서 모든 빌보드의 크기를 동일하게 만든다.

2. 붓질 렌더링하기

본 논문의 알고리즘에 대한 렌더링 파이프라인을 [그림 4, 5]에 나타내었다. 본 논문에서는 움직임 없이 정지된 영상인 첫 번째 프레임을 붓질을 사용하여 렌더링하는 방법과 이후 장면에 변화가 일어났을 때 붓질의 방향을 결정하는 방법으로 나누어 설명한다.

2.1 첫 장면 렌더링 하기

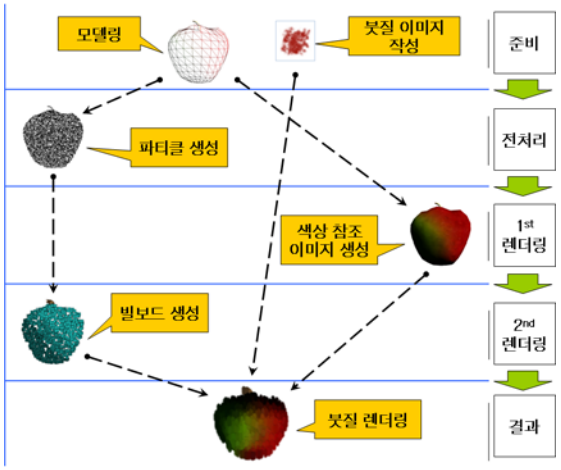


그림 4. 첫 장면 렌더링 하기
(점선 화살표의 시작은 입력, 끝은 출력)

카메라에 애니메이션이 일어나기 이전에 3차원 물체 위에 붓질을 위치시켜 첫 장면을 렌더링 해야 한다. 그 과정을 [그림 4]에 나타내었다.

먼저 3차원 물체로 이루어진 장면과 붓질로써 사용할 하나의 붓질 이미지를 만든다. 그리고 전처리 과정으로 물체의 표면에 파티클들을 생성한다. 파티클은 물체를 이루는 각각

의 폴리곤 내부에 지정한 개수만큼 랜덤한 위치에 만든다. 파티클을 모두 생성하고 나면 붓질의 색상을 결정하기 위하여 물체를 렌더링하여 색상 참조 이미지를 만든다.

이후 파티클을 렌더링하기 위하여 빌보드를 만든다. 빌보드는 원래 시선의 방향과 항상 수직인 성질을 가지는데 여기에 더해 본 논문에서는 시점과의 거리에 상관없이 화면 공간에서 동일한 크기를 가지도록 만든다. 이런 성질로 인하여 빌보드는 3차원 물체를 2차원 이미지처럼 나타내기에 적합하다. 빌보드를 만들기 위해 물체의 모델 공간에 있는 파티클의 위치를 투영 공간으로 변환한 뒤 그 위치를 중심으로 사각형 빌보드를 만든다.

본 논문에서는 현재 각각의 빌보드에 대한 방향을 설정하는 시스템을 가지고 있지 않기 때문에 모든 빌보드의 초기 방향을 동일하게 설정하고 있다.

빌보드가 만들어졌으면 첫 번째 렌더링 패스에서 만든 색상 참조 이미지로부터 색상 정보를 얻어와 빌보드 위에 입혀지는 붓질 이미지와 알파 블렌딩하여 붓질을 완성한다. 이 방법은 Meier의 방법을 기반으로 실시간 시스템을 만들어낸 Daniel의 방법[5]과 동일하다.

3.2 카메라의 움직임을 따라 붓질 회전하기

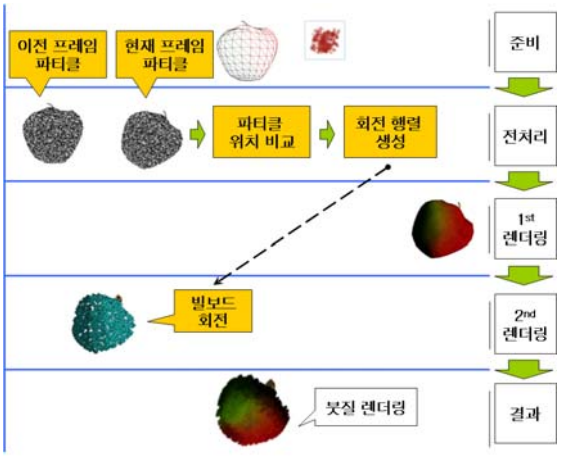


그림 5. 움직임을 따라 붓질 회전하기
(점선 화살표의 시작은 입력, 끝은 출력)

첫 장면이 렌더링된 후부터는 매 프레임 이전 프레임의 파티클들의 투영 공간 좌표와 현재 프레임의 좌표를 비교한다. [그림 5]은 이 과정을 보여준다.

좌표를 비교하기 위해서는 비교할 대상인 샘플 포인트들을 정해야 한다. 빌보드의 중심 위치의 파티클과 그 외 다른 파티클과 모델 공간에서 거리를 계산하여 일정거리 이내의 파티클만을 샘플 포인트인 인접 파티클로 설정한다. 모델이 하나의 단일한 강체가 아니라 강체들의 집합체일 경우 각각의 강체별로 붓질이 다른 방향을 가져야하기 때문에 거리를 이용하여 인접 파티클을 결정한다. 또한 물체가 하나의 강체로 이루어졌다해도 모델의 파티클 전체를 비교할 경우 보이는 면의 파티클들과 보이지 않는 면의 파티클들의 투영 공간에서 움직임의 방향이 반대가 되기 때문에 보이지 않는 면의 파티클들의 움직임을 비교대상에 넣어서는 안된다. 본 논문의 방법 역시 모델 좌표에서 인접 파티클을 결정하였기 때문에 보이지 않는 면의 파티클을 참조할 위험이 있으나 인접 파티클을 결정하는 기준 거리를 작게 잡아 에러를 줄였다.

각 빌보드가 될 파티클마다 인접 파티클들이 정해지고 나면 각 인접 파티클들의 투영 좌표를 이용하여 비교를 시작한다. 이 비교를 통해 이전 프레임의 인접 파티클들이 현재 프레임의 파티클들의 위치로 가장 가깝게 움직일 수 있는 변환 z를 찾는다. 이 변환 z는 least-square 방법[4]을 사용하여 식(1)와 같이 나타낼 수 있다. 다시 말해 이전 프레임의 위치와 현재 프레임의 위치의 차이 E를 최소로 만드는 변환이 z이다.

$$E = \sum_i |z p_i - c_i|^2 \tag{1}$$

식(1)에서 i는 인접 파티클 인덱스, p는 이전 프레임에서 빌보드로 렌더링 될 파티클의 인접 파티클 좌표, c는 현재 프레임에서 빌보드로 렌더링 될 파티클의 인접 파티클 좌표이고 이 때 p와 c의 좌표는 빌보드가 될 파티클과의 상대적인 좌표를 의미한다.

z값은 식(2)를 통하여 계산할 수 있다. z는 2차원 벡터로써 표현이 된다. 이 z와 벡터 (1,0)와의 각도가 빌보드의 회전각이 된다. 변환 z는 방향 이외에 크기 변환 정보도 가지는데 본 논문에서는 물체를 평면적으로 표현하기 위하여 빌보드의 크기를 일정하게 만들고 있기 때문에 사용하지 않는다.

$$z = \left(\sum_i p_i \cdot c_i, \sum_i p_i \times c_i \right) / \sum_i |p_i|^2 \tag{2}$$

z값에 의한 변환은 근사값을 구하는 것으로 매프레임 계산할 경우 오차가 누적될 수 있다. 본 논문에서는 p의 값으로 이전 프레임에서의 파티클의 좌표를 사용하지 않고 화면이 렌더링되는 첫 프레임에서의 파티클들의 좌표를 사용한다. 이 방법은 오차가 누적되지 않도록 할뿐더러 첫 프레임에서 화면에 렌더링 되는 붓질에 대하여 어떠한 변환(위치이동, 회전) 일어나더라도 첫 화면과 동일한 시점, 시선으로 돌아오면 붓질 역시 처음의 방향으로 돌아오는 장점이 있다.

하지만 첫 프레임의 화면에 보이지 않는 파티클들, 즉 시선 벡터와 물체 표면의 법선벡터와 내적값이 0과 -1 사이의 값을 가지는 폴리곤 위의 파티클들은 이후 프레임에서 화면에 보였을 때 직관적으로 이해하기 어려운 방향을 가지게 되는데 이것은 첫 화면에서 보이지 않는 면에 있는, 파티클과 인접 파티클들과의 상대적인 위치관계가 눈에 보이는 위치로 이동할 때 z값 계산을 통한 2차원 회전만으로 표현할 수 없는 방식으로 변하기 때문이다.

본 논문에서는 이 비교를 통하여 구해진 회전각을 이용하여 만들어진 회전행렬을 파티클을 정점 셰이더로 4회 전달하여 생성되는 빌보드에 곱하여 시선축을 기준으로 회전시킨다.

4. 결과

본 논문에서는 CPU – Intel Pentium4 3GHz, VGA – Nvidia 7600GS를 사용하여 실험하였다.

본 논문에서는 파티클의 위치에 붓질을 렌더링하기 위하여 사용하는 빌보드를 눈과 물체의 거리와 독립적으로 크기를 일정하게 하고 있다. [그림 6]는 물체와 시점과의 거리를 변경하며 붓질의 크기를 촬영한 결과이다. 촬영을 위하여 파티클의 개수는 시점에 따라 각각 임의로 조절하였다.

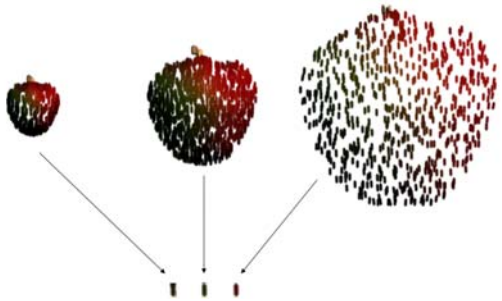


그림 6. 시점에 따른 빌보드의 크기

[표 1]은 여러 가지 물체를 사용하여 본 논문의 방법으로 렌더링한 결과이다. [표 2]는 [표 1]의 물체들을 렌더링 할 때 본 논문의 방법의 각 부분에 대한 속도를 측정한 결과이다. 본 논문의 구현은 크게 전처리 부분과 렌더링 부분으로 나눌 수 있다.

본 논문에서는 전처리 과정에서 모든 파티클에 대하여 각각 인접한 파티클 리스트를 작성하였다. 리스트에는 최대 50개까지의 인접하다고 판정된 파티클이 저장된다. 인접의 여부는 모델 좌표계에서의 파티클간 거리를 계산하여 판단한다.

렌더링 부분에서 매프레임 화면에 렌더링되는 파티클들에 대하여 각각의 파티클이 가지고 있는 인접 파티클들의 위치를 비교하여 빌보드가 회전해야할 회전각을 얻었다.

표 1. 여러 가지 모델과 렌더링 결과

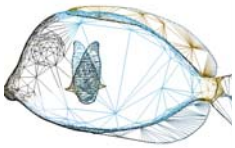
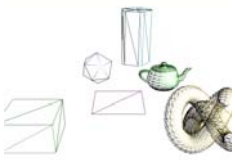
모델 종류	모델 정보	렌더링 결과
사과		
물고기		
간단한 물건들		

표 2. 속도 성능 결과

모델	사과	물고기	간단한 물건들
총 폴리곤 수 (number)	550	8,618	3,978
총 파티클 수 (number)	45,658	90,213	355,467
전처리 단계 (sec)	0.19845	1.34773	4.61882
화면에 보이는 파티클 수 (number)	327	3562	2137
렌더링 단계 (sec)	0.00165	0.00556	0.00722
전체 속도 (fps)	30.75	12.60	4.73

[표 3]은 시선축을 기준으로 눈을 회전하였을 때 붓질이 처음의 붓질상태를 지속적으로 유지하는 결과를 보여준다. [표 3]에서 붓질이 초기의 방향을 계속 유지하는지 확인하기 위하여, 최초의 렌더링 결과를 이미지 편집도구로 회전시킨 결과와 실제 본 논문의 알고리즘을 사용하여 눈을 회전한 결과를 비교하였다.

표 3. 회전 결과들 간의 차이 비교

회전 각도	Image tool rotation	Our method rotation
90		
180		

[그림 7-1,2]은 눈을 화면의 상하좌우로 움직였을 때의 결과이다. 시점과 시선이 위치 이동을 할 때는 샘플 포인트 간의 상대적인 위치 변화가 적을 것이다. 따라서 붓질 역시 처음의 방향을 계속 유지한다.

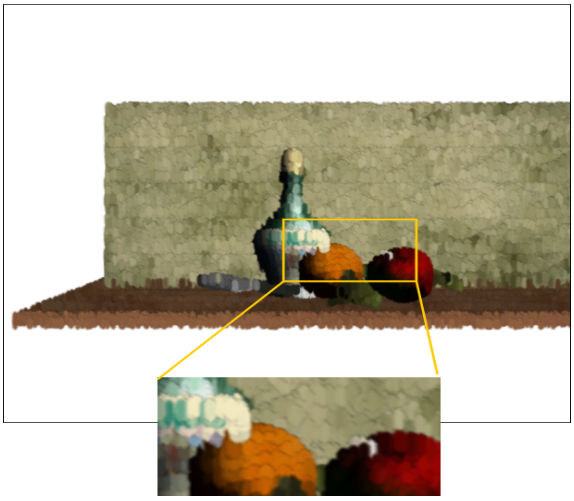


그림 7-1. 초기 붓질 상태

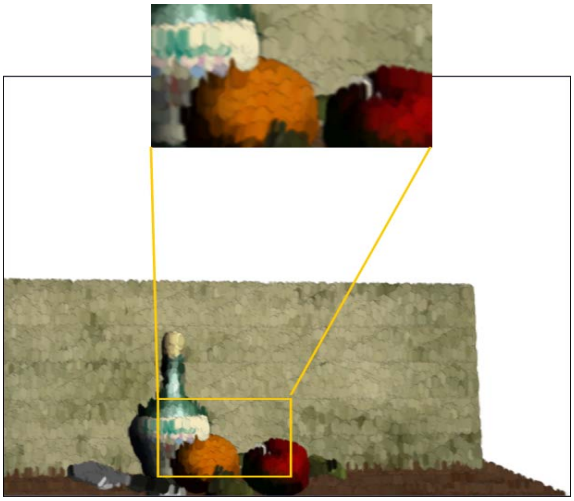


그림 7-2. [그림 7-1]에서 시점과 시선을 위쪽으로 이동한 결과

[그림 8]은 카메라를 임의로 동적으로 움직이며 촬영한 프레임 시퀀스이다.

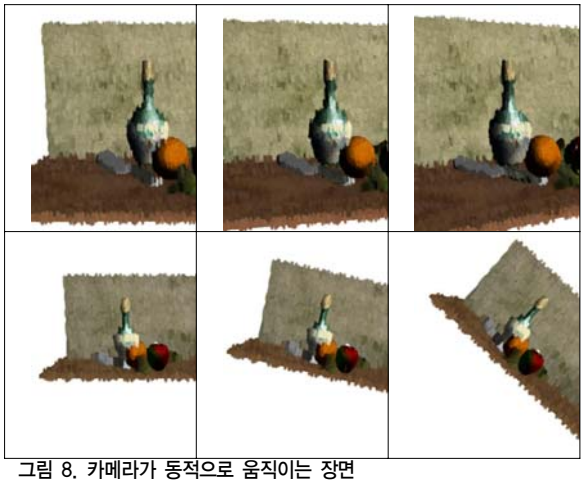
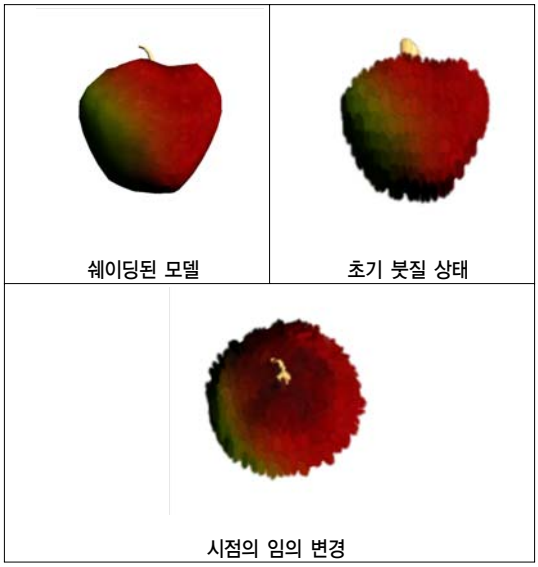


그림 8. 카메라가 동적으로 움직이는 장면

시선을 시선축에 수직으로 이동하거나 시선축을 기준으로 회전하지 않고 그 외의 위치이동이나 회전을 한다면 이전 프레임에서 보이지 않은 면이 보일 수 있다. 이 경우 보이지 않던 면이 보이게 되면서 면이 뒤집어지는 변환이 일어나기 때문에 오직 화면 공간에 수직인 축에 대한 회전만이 가능한 빌보드로서는 회전을 줄을 수 없다. [그림 9]는 그 예를 보여준다.



[그림 9] 시점과 시선의 임의변경에 의한 결과

4. 결론

본 논문에서는 시간에 따라 물체의 위치나 시점과 시선의 변화가 있는 3차원 물체로 구성된 장면을 붓질을 사용하여 회화 스타일로 렌더링하는 방법을 제안하였다. 먼저 물체 표면에 파티클을 만들고 파티클들의 위치를 이미지 공간으로 가져와 붓질의 위치로써 사용하였다. 붓질을 렌더링한 후 장면 안에서 앞에서 말한 변화가 일어나면 첫 화면에서 파티클들의 위치와 현재 파티클들의 위치를 비교하여 붓질의 방향에 변화를 주었다.

이후 얻어진 결과를 통하여 본 논문에서는 시선축을 기준으로 시선이 회전하는 경우와 시선축에 수직인 방향으로 시점과 시선이 이동하는 경우 처음의 붓질 상태가 잘 유지되는 것을 확인하였다. 이외의 축으로 회전하거나 이동하는 경우에도 처음의 붓질 상태와의 차이를 가장 적게 만드는 회전을 보여주었다. 다만 첫 프레임에 보이지 않는 파티클이 보이게 되는 경우에 본 논문에서는 첫 화면에서의 빌보드의 방향만을 가지고 있기 때문에 이 부분의 빌보드의 방향에 대한 정보를 알 수 없다.

정확한 결과를 얻을 수 있었던 카메라의 회전과 위치이동은 3차원 물체에 물체가 2차원 물체처럼 보이도록 텍스쳐를 만들어 맵핑하는 방법에서도 동일하게 적용될 수 있다. 하지만 이 방법은 시점과 시선에 대하여 그 이외의 변환이 이루어졌을 때 2차원적 평면성을 상실한다. 본 논문의 경우 붓질을 동일한 크기의 빌보드를 사용하여 렌더링하기 때문에 붓질이 평면성을 유지한다.

장면 안에서 물체나 눈의 이동과 회전 이외에 물체의 크기가 변화하는 애니메이션도 있을 수 있다. 본 논문의 방법은 모델 공간에서 물체의 표면에 붓질이 위치할 샘플 포인트인 파티클들을 만들기 때문에 빌보드가 될 파티클과 그 인접 파티클들의 상대적인 관계가 변화하는 몰핑 애니메이션의 경우 회전을 적절하게 표현하지 못할 것이다.

참고문헌

[1] "회화 [繪畵, art of painting]", 브리태니커 백과, 예술 〉 미술 〉 회화, empas 백과사전 제공, http://100.empas.com/dicsearch/pentry.html?s=B&i=211065&v=45

[2] S. Breslav, K. Szerszen, L. Markosian, P. Barla, J. Thollot, "Dynamic 2D Patterns for Shading 3D Scenes" ACM Transaction on Graphics, Vol. 26, No. 3, Proceedings of SIGGRAPH '07, 2007

[3] M. Barbara J., "Painterly rendering for animation" ACM, In Proceedings of the 23rd Annual Conference on Computer Graphics and interactive Techniques SIGGRAPH '96. pp. 477-484, 1996.

[4] B. K. P. Horn, , "Closed-form solution of absolute orientation using unit quaternions", OSA, Vol. 4, Issue 4, Journal of the Optical Society of America A, pp. 629~ ,1987

[5] D. Sperl, "Realtime Painterly Rendering for Animation", www.incognitek.com