3D 애니메이션의 [illegible]

□ 소요환* / * 한남대학교 멀티미디어학부 교수

Ⅰ. 서 론

세계 애니메이션 산업은 지난 20여 년 동안 급격하게 성장하였으며 2005년도에는 약 770억 달러의 시장규모가 형성될 것으로 전망한다(출처: Digital Vector, 2003). 물론 이러한 예상 수치는 규모 산출에 제한적이고 다소 과감한 것일 수도 있지만, 지난 몇 십년 동안 애니메이션 기법이나 기술에서 나타난 발전은 애니메이션이 엔터테인먼트의 소스로서 가지는 중요성과 호소력을 확산시킨 것이 사실이다. 애니메이션은 영화, 방송, 인터넷, 게임 그리고 모바일에까지 광범위하게 확산되어 있다. 또한 인터넷이나 모바일이라는 새로운 배급 창구가 활성화되면서 애니메이션의 기능적 역할이, 보는 미디어에서 체험하는 미디어로, 일방향성 미디어에서 쌍방향성 미디어로 발전되고 있다. 이렇게 애니메이션의 개념이 확대되기 시작한 것은 아날로그식 2D 애니메이션, 즉 셀 애니메이션을 기반으로 하던 만화영화 형식의 애니메이션이 제작 시스템에 컴퓨터를 본격적으로 도입하면서 디지털로 전환된 시점부터이다. 이러한 컴퓨터 제작 시스템의 도입은 다양한 디지털 제작방식을 활성화시켰고, 엔지니어와 아티스트의 효율적인 결합으로 3D 애니메이션의 기술 발전을 조기에 이루어 놓았다.

최근 경쟁적으로 새로운 버전을 발표한 각 3D 소프트웨어 개발사들의 제품은 모델링, 렌더링, 애니메이션 등 관련된 핵심기술을 대부분 지원하면서 특정기술을 특화시켜 다른 제품과의 차별성을 확보하고 있다. 지난 2004년 8월 2일 Discreet는 SIGRAPH 2004를 통해 다양한 기능이 추가된 3DS Max 7을 발표했으며, Alias는 다양한 시각효과 plug-in 기능을 추가한 Maya 6.5를 출시하였다. 이러한 소프트웨어 개발사들의 치열한 경쟁과

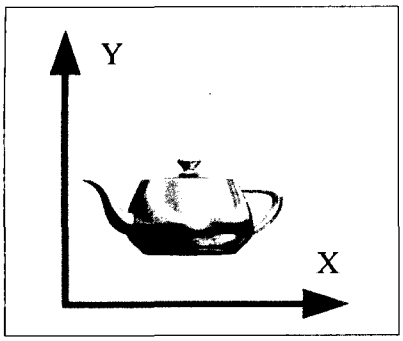
컴퓨터 하드웨어 및 소프트웨어 기술의 발전은 3D 애니메이션의 사실성을 더욱 가능하게 하였으며 사실적 표현 기술은 정교한 고해상도의 화상처리, 실사와 같은 영상처리, 입체 사운드의 청각효과 등과 더불어 공간구성, 조명, 재질, 맵핑 그리고 렌더링 등 시각적 표현요소들을 얼마나 효율적으로 활용하느냐에 달려 있다고 할 수 있다. 이러한 시각적 표현요소들은 디지털 기술의 발전으로 더욱 더 핵심적인 시각의 힘으로 활용된다. 최근의 3D 애니메이션 작품들의 경향을 보면 실사영상과 비교하기 힘들 정도의 획기적인 예술적 시각 기술을 보여 주고 있다. 또한 국내외 관련 기업이나 연구단체들은 비주얼 기술 분야에 대하여 집중적으로 많은 실험과 개발에 전념하고 있고 실제로 사실적 이미지를 표현하기위한 수많은 방법들이 개발되고 있다. 이러한 기술개발의 가장 큰 원인은 컴퓨터그래픽의 최종 목표가 사실적 표현의 완성이고 시각적 요소들은 사실적 표현을 위한 가장 중요한 기술요소 중에 하나이기 때문이다. 그리고 아마도 시각적 표현요소의 중요성은 2차원적인 평면 공간에서의 조형논리와 3차원적인 입체 공간에서의 조형논리가 다르게 해석되기 때문일 것이다.

본 고에서는 3차원 공간을 기반으로 한 소프트웨어의 사용에 중요한 역할을 담당하는 3D 애니메이션의 다양한 시각적 요소들을 분석하고 효율적인 3D 애니메이션 제작공정을 단계적으로 기술한다.

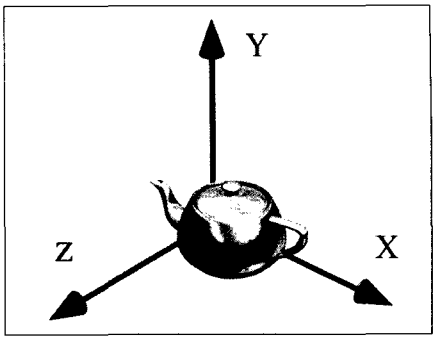
II. 가상공간에서의 표현요소

1. 3차원 공간 구성

2차원적인 평면에서의 공간구성은 일반적으로 2가지의 축으로 구성되어지는 공간을 시각적으로 인지한다. 다시 말하면 모든 실제의 객체들이 다른 시각적인 결과물로 보일 때는 가로(X축), 세로(Y축)의 2가지 축에 의존하는 구성 틀(Frame)로 만들어져 사람들의 눈을 통해 보이게 된다. 결국 3차원적인 공간구성도 2차원적인 구성 틀을 유지하며 보이게 된다는 것이다. 그래서 이미지를 통해 현실감 있는 가상공간을 만드는 작업은 재작 방식의 차이일 뿐 2차원의 평면 안에 3차원의 공간감이 느껴지게 한다는 점에서 공통점을 가지고 있다. 이러한 공통점에도 불구하고 3차원적인 공간구성은 구성의 틀에서 2차원적인 평면구성과 완전히 다른 세계라는 것은 틀림이 없다. 눈으로 보고 만지고 느낄 수 있는 물체는 입체로서 현실 속에서 일정한 공간을 점유하고 있으며 이 공간은 각각 가로, 세로, 깊이를 뜻하는 X, Y, Z 좌표를 통해 3차원으로 표시될 수 있다. 컴퓨터를 사용하여 이미지를 창조하는 사용자들에게 가장 난해한 문제점들 중의 하나는 바로 3차원 공간을 2차원의 모니터나 스크린 위에 실현시켜야 한다는 것이다. 예를 들어, 사진작가가 2차원 화면에 3차원의 공간을 만드는 작업을 한다면 이미 현실에 존재하는 공간 이미지를 조명과 카메라 렌즈, 노출 등을 이용하여 필름에 입력함으로써 얻어질 수 있다. 그러나 3차원 화면은 공간감을 전달할 수 있는 가상공간과 오브젝트를 사용자가 직접 창조해내야 한다는 차이가 있다. 사용자에 의해 가상의 3차원 디지털 공간이 어떻게 2차원의 화면 위에 창조되는지를 알기 위해서 3차원적인 일루전(illusion)이 어떻게 일어나는지 먼저 이해할 필요가 있다. 3D 가상공간은 좌표체계 위에 만들어진다. 〈그림 1〉과 같은 X, Y축의 2차원공간에 〈그림 2〉와 같이 Z축이 첨가된다. 3차원 공간 안에 오브젝트를 위치시킨다는 가정 하에서



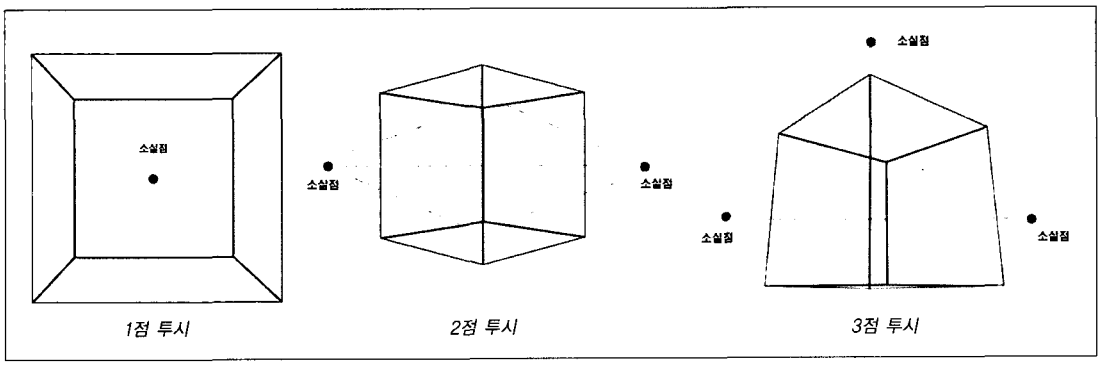
〈그림 1〉 X, Y 좌표의 2차원 공간구성



〈그림 2〉 X, Y, Z 좌표의 3차원 공간구성

이미지를 만드는 작업이다.

3차원 공간 안에 오브젝트를 위치시키려면 심도(Depth)에 대한 정보를 설정해 주어야 한다. 심도는 공간의 깊이를 만들어 주는 역할을 하는데 전통적으로 이와 같은 공간 입체감은 원근법에 의하여 만들어져 왔다. 일반적으로 2차원적인 화면구성에 3차원적인 공간구성을 인지시키기 위해서는 원근법을 기반으로 한 투시도법(혹은 원근법)의 원리를 이용한다. 15세기 르네상스 이탈리아에서 창안된 원근법(perspective)은 2차원적인 평면에서 3차원적인 공간의 깊이를 정확하게 묘사하기 위해 고안된 재현양식으로 입체적인 대상이 거리가 멀어짐에 따라 크기가 작아지는 현상과 평행선이 무한히 멀어지면서 마치 한점으로 수렴되는 듯이 나타나는 현상을 기하학적 비례의 원리에 입각하여 재현함으로써 시각 공간을 창조하는 방법이다. 원근법의 효과는 공간의 깊이를 구축함으로써 현실과 같은 시각적인 사실감을 창조하는데 있다. 원근법에서 평행선이 수렴되는 듯이 나타나는 가상의 한 점을 소실점(Vanishing point)이라고 하는데, 이 시각 공간의 중심에 해당하는 소실점과 보는 사람의 눈의 시점을 일치시키면 마치 보는 사람이 시각 공간을 실제로 자신의 시각으로 바라보는 착각을 일으키게 한다. 〈그림 3〉과 같이 투시도법은 소실점을 어디에 위치하느냐에 따라



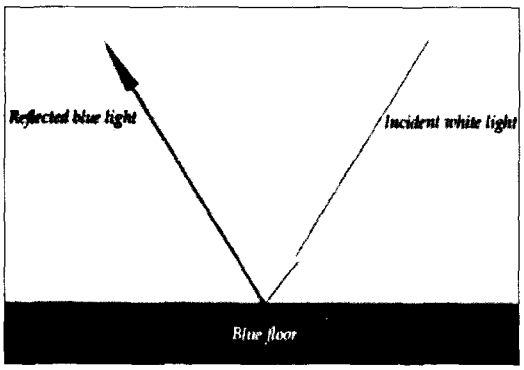
〈그림 3〉 3점 투시도법

공간구성의 결과가 다르게 느끼게 되고 소실점을 2점, 혹은 3점을 부여하는 공간을 구성하기도 한다.

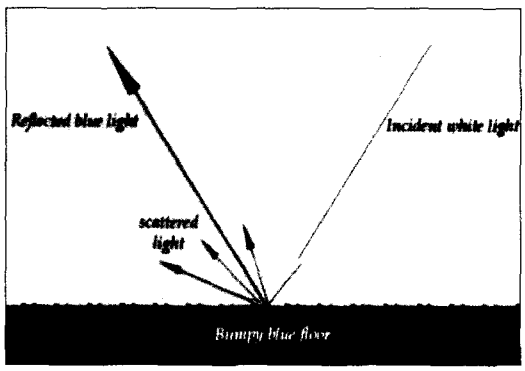
이와 같은 개념을 이용해 원근법적인 투사(perspective projection)로 그림을 그리면 같은 크기라 해도 멀리 떨어진 부분은 가까이 있는 부분보다 작게 그려진다. 원근법을 사용하는 이유는 이 방식이 사람의 눈이나 렌즈를 통해 화상을 받아들이는 방식과 같기 때문이다. 결국 컴퓨터에 기반을 두고 2차원의 화면에 3차원의 공간감을 만들기 위해서는 원근법적인 투사의 개념을 충실히 구현시켜 줄 수 있는 소프트웨어가 요구된다. 이와 같은 소프트웨어의 도움으로 기하학적인 공간과 모델이 완성되면 실사 영화와 마찬가지로 조명과 카메라를 설치하여 작업이 진행되어야 한다.

2. 조명(Light)의 구성

가상공간이 형성되어지면 공간에 위치하는 어떠한 조형 오브젝트가 만들어 질것이다. 이러한 모델링 이미지를 시각적으로 인지시키기 위해서는 빛 또는 조명(Light)의 존재가 절대적으로 필요하게 된다. 자연적인 빛 혹은 인공적인 조명은 공간의 이미지에 있어서 가장 중요한 시각적인 표현요소 중의 하나일 것이다. 빛은 오브젝트를 비추는 근본적인 역할 외에도 톤의 차이, 윤곽, 외형, 색상, 질감, 깊이 등을 결정한다. 그리고 균형, 조화, 대비를 제공하면서 화면 구성적 관계를 창조하기도 한다. 이러한 조명의 요소는 시각적인 디지털 기술의 발전으로 더욱더 핵심적인 시각의 힘으로 활용되며 3차원 공간을 기반으로 한 3D 컴퓨터그래픽의 작업에 중요한 역할을 담당한다. 일반적으로 사람의 눈이 무엇인가를 보고 인식할 수 있는 것은 빛의 광원이 무엇인가에 닿고 그곳에서 반사된 빛이 사람의 눈에 들어와 눈의 시신경이 그것을 인식하기 때문이라 생각하고 있다. 즉 무엇인가를 보기 위해서는 빛의 반사가 사람의 눈에 들어 올 필요가 있다는 것이다. 빛의 색은 빛 자체의 소스에 의존한다. 대부분의 빛은 삼원색(RGB)이 균형 있게 혼합되어 있기 때문에 무색으로 보인다. 무색의 빛은 존재하는 모든 가능한 색으로 구성된다. 빛이 어떠한 물체에 비춰지면 무색의 빛의 광선은 색이 변화된다. 만약 빛이 흰색 오브젝트를 비춘다면 광선은 본래의 색으로 반사되어 나타나고 오브젝트가 검정색이라면 빛의 원래 색이 무엇이건 간에 오브젝



〈그림 4〉이론적인 빛의 반사



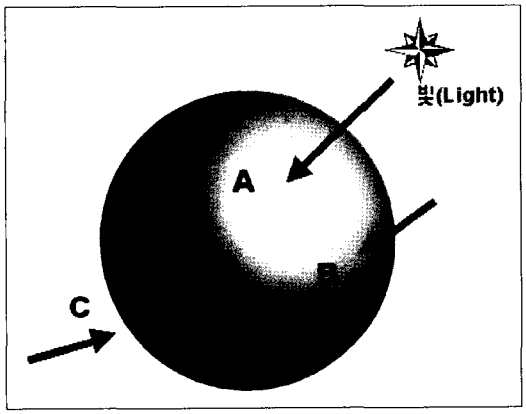
〈그림 5〉 실제상황의 빛의 난반사

트는 빛을 흡수하고 어떠한 반사도 하지 않는다. 그래서 기본적으로 사람의 눈이 완전히 검정색의 오브젝트를 볼 때, 어떤 빛도 사람의 눈에 들어가지 않기 때문에 오브젝트가 검정색으로 보인다. 그림은 파란색 바닥면에 반사되는 일반적인 빛이다. 〈그림 4〉에서 바닥은 파란색을 제외하고 일반적인 광선의 모든 색을 흡수하고 반사한다. 이러한 경우 바닥면의 조명되는 빛은 입사각과 반사각이 같아지는 법칙이 존재한다. 그러나 이것은 이론상의 논리이고 실제상황에서는 〈그림 5〉와 같이 난반사하여 빛은 반사각을 중심으로 원추 형상의 난반사가 일어난다.

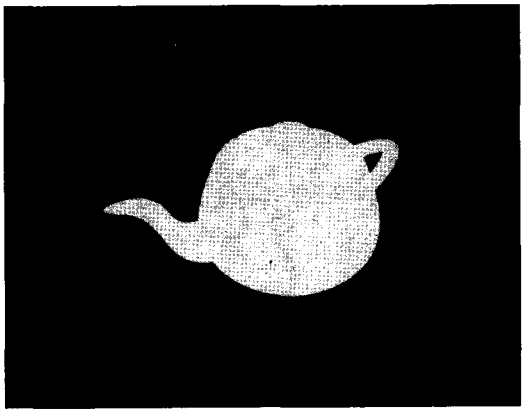
또한 입체적인 오브젝트와 같은 경우 밝은 부분과 어두운 부분이 공존하게 된다. 밝기의 차이는 그 부분의 면이 빛에 대하여 어떠한 각도로 향하고 있는가에 의해 생긴다. 〈그림 6〉의 A 부분은 빛에 가까운 쪽을 향하고 B 부분은 빛에서 옆을 향하고 있고 C 부분은 빛이 닿지 않은 위치가 된다. 그 위치가 밝기의 차이가 되고 있다. 디지털 가상공간 안에 빛을 가지고 갈 수는 없지만 빛이라는 매개변수들을 설정하여 숫자로 표시하면 태양 빛이나 인공조명과 같은 광원을 만들 수 있다. 광원이 비춰지는 빛의 매개변수를 변화시키면 강하고 밝은 광원이나 어둡고 약한 광원이 되고 면의 밝기는 광원을 향하는 입체 조형물에 각도에 의해 계산된다.

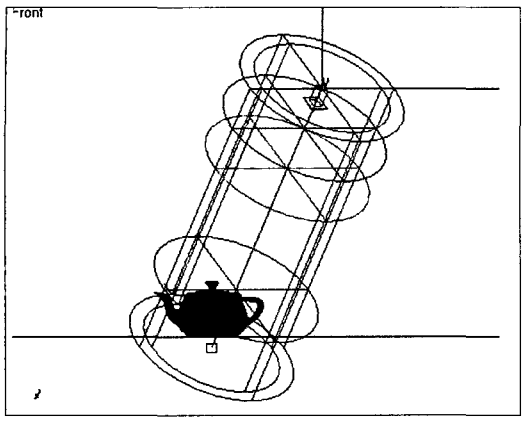
또한 실제의 광원에는 빛을 내는 방식에 따라 몇 가지의 광원 종류가 있다. 3차원 디지털 가상공간에서는 실제 환경에서의 빛에 대한 기본 개념을 거의 흡사하게 시뮬레이션한다. 3D 공간에서 만약 장면에 빛을 사용 안한다면 화면은 아무것도 보이지 않는 검정색으로 보일 것이다. 그렇기 때문에 가상공간에서는 필수적으로 인공적인 조명을 설치하여야 한다. 3D 공간에서는 장면에 어떻게 조명을 설치하는가에 따라 효과가 결정된다. 3D 가상공간에서의 기본적인 조명구조는 주변조명(Ambient light), 직사조명(Directional light), 점조명(Point light/Omni light), 스폿조명(Spot lights) 그리고 태양의 효과를 시뮬레이션하는 태양조명(Sunlight) 등이 있다.

주변조명의 경우 모든 공간에 동일하게 빛을 비추는 균일성을 가지고 있으나 방향성(Targeting)이

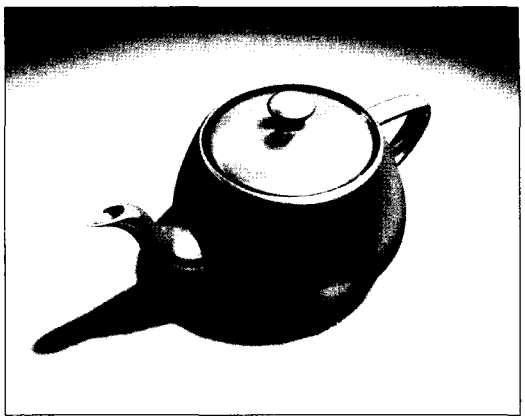


〈그림 6〉 입체적인 오브젝트의 명암구분


〈그림 7〉 주변조명만을 적용한 렌더링 결과



〈그림 8〉 직사조명의 광원 형태

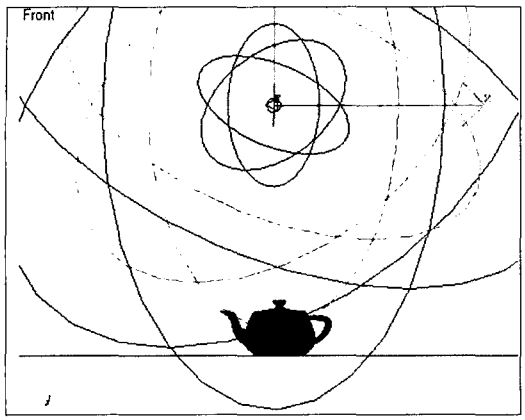


〈그림 9〉 직사조명만을 적용한 렌더링 결과

없다는 특징이 있다. 주변조명은 오브젝트의 위치를 중심으로 360° 모든 방향에서 오브젝트의 표면을 향해 균일한 강도로 빛을 비춰지는 조명으로 오브젝트의 그림자를 감소시키는 효과를 발생한다. 따라서 어떤 공간을 주변조명으로 비출 경우 〈그림 7〉과 같이 렌더링된 오브젝트들은 단순하고 평면적인 실루엣을 갖고 있는 것처럼 보이게 된다. 주변조명은 대표적인 간접조명이지만 자연적인 상황에서는 이와 같은 빛을 쉽게 찾아보기는 어렵고 주변조명에 비추어진 오브젝트는 입체감을 상실하기 쉬우며 탈색된 느낌을 준다.

직사조명은 태양광의 형태와 가장 유사한 조명이며 인위적인 광원 없이 일정한 강도와 방향을 가지고 있는 특징이 있다. 태양은 지구로부터 멀리 떨어져 있기 때문에 광원은 〈그림 8〉과 같이 마치 평행한 광원의 형태를 가지고 지구를 비추는 모습으로 보이며 광선의 강도 또한 일정하게 유지되도록 시뮬레이션한다. 이러한 직사조명은 주변조명과 다르게 〈그림 9〉와 같이 오브젝트의 그림자를 만들 수 있기 때문에 입체감을 구현할 수 있다. 정교하게 설계된 복수의 직사조명 세트를 이용해 다양한 그림자를 만들 수 있기 때문에 가상의 공간에서 입체감을 만들어 내는 데 매우 중요한 역할을 한다. 대부분의 3D 소프트웨어에서는 직사조명의 광원 형태를 기반으로 다양한 조도계산법을 사용한다.

점조명은 3차원의 가상공간 안에서 광원의 위치가 설정되어 있는 조명으로 특정한 광원의 위치로부터 〈그림 10〉과 같이 360° 방향으로 빛을 내보내게 된다는 점에서 직사조명과 구분된다. 직사조명이나 주변조명과 달리 3차원 공간의 일정한 부분을 비출 때 주로 사용하게 되는데 사실적인 효과를 만들어 내는데 특히 유용한 방법이다. 강도에 있어서도 점조명은 빛의 강도가 빛이 퍼져나가는 길이에 따라 점차 줄어드는 특징을 보인다. 3D 소프트웨어에서는 이와 같이 빛의 강도가 거리에 반비례하는 현상을 감소(Falloff)현상이라고 한다. 태양광원을 설정한 경우가 아니라면 감소기능을 적절히 이용할 필요가 있다. 점조명에 의해 발생하는 그림자는 직사조명에 의해 만들어지는 그림자와 같이 평행한 특징을 보이지 않는다. 〈그림 11〉과 같이 광원의 위치를 중심으로 방

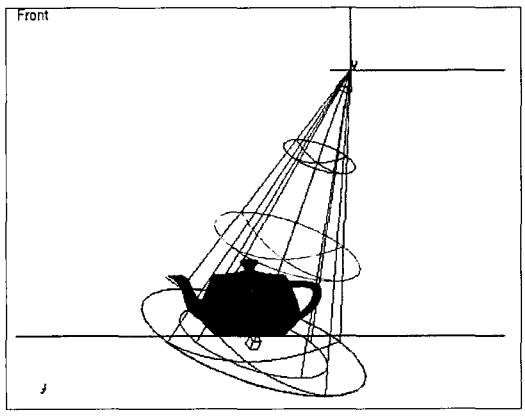


〈그림 10〉 점조명의 광원 형태

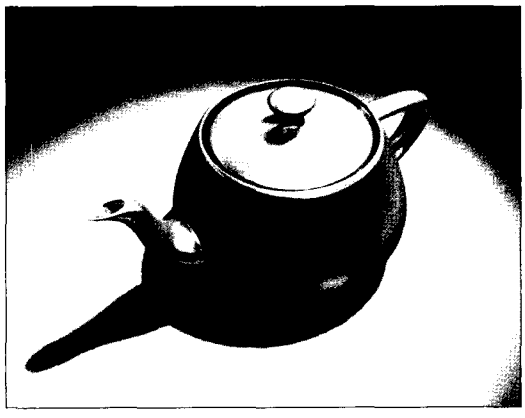
〈그림 11〉 점조명만을 적용한 렌더링 결과

사형의 그림자가 발생하는 것이 빛의 성격에 부합되기 때문이다.

스폿조명은 점조명과 유사한 특징을 지니고 있으나 〈그림 12〉와 같이 방향성이 있다는 점에서 차이가 있다. 즉 점조명이 360° 방향으로 빛을 내보낸다면 스폿조명은 그 중 일정 방향에 한하여 빛을 발산하는 조명이라고 할 수 있다. 점조명과 마찬가지로 스폿조명을 설정할 때도 프로그램 상에 감소기능의 범위를 지정해 주어야 한다. 대부분의 3D 소프트웨어에서는 스폿조명의 외곽선 형태를 지정할 수 있는데 경계선의 부드러움에 대한 수치, 즉 연성지수(Softness value)를 0으로 설정하면 조명이 비추는 범위와 그렇지 않은 범위가 선을 경계로 분명히 나누어지게 된다. 〈그림 13〉과 같이 연성지수를 높게 설정하면 조명된 밝은 부분에서 조명을 받지 않는 어두운 부분까지 완만한 명도 변화를 만들어 낼 수 있다.



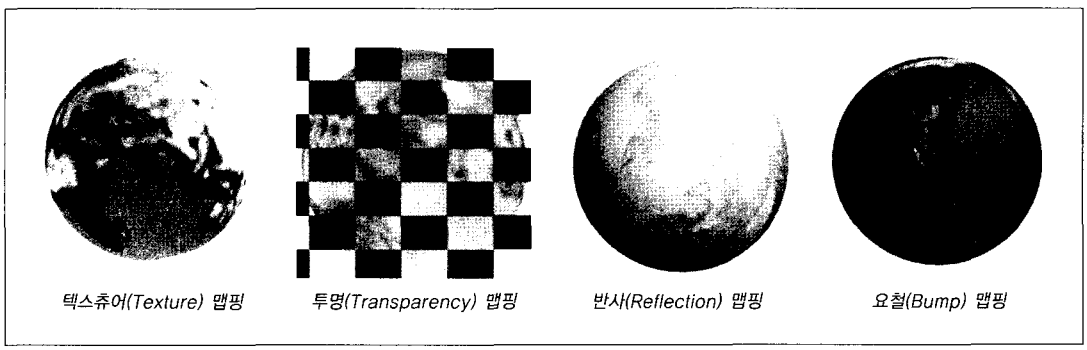
〈그림 12〉 스폿조명의 광원 형태



〈그림 13〉 스폿조명만을 적용한 렌더링 결과

3. 색상과 재질 구성

오브젝트를 위한 공간이 구성되고 빛을 첨가시킨다면 그 다음은 오브젝트에 색(Color)과 재질(Material)을 결정짓는 것이다. 오브젝트가 무슨 색으로 보이는가, 무슨 색으로 표현하면 적절한가, 그리고 어떠한 방법으로 오브젝트에 색을 표현을 할 수 있는가를 고민해야한다. 물론 재질의 색은 빛의 색과 오브젝트의 조형성과도 연계돼야 할 것이다. 오브젝트의 색을 인식하기 위해서는 오브젝트에 비쳐지는 빛의 색이 포함되어야 한다. 다행히 태양 빛이나 인공조명 등과 같은 광원의 대부분은 무색으로 발하기 때문에 오브젝트가 어떤 색을 많이 반사 또는 흡수하는가에 따라 오브젝트의 색을 인지할 수 있다. 이 개념을 가상공간에 응용하면 모델링 데이터의 색을 표현할 수 있다. 예를 들어, R=100, G=50, B=0으로 한 경우 적색 빛은 100% 반사하고 녹색 빛은 50% 만 반사하고 나머지 50%는 흡수하며 청색은 전부 흡수하게 된다. 모델링 데이터의 표면이 어떤 색의 빛을 많이 반사하여 사람의 눈에 보이는가 하는 정보를 3차원 가상공간에서는 텍스츄어(Texture)라고 한다. 물론 텍스츄어는 넓은 의미로 오브젝트 표면의 상태도 포함하고 있다. 텍스츄어 작업이 이루어지지 않은 오브젝트는 아무런 외적 느낌을 줄 수 없는 매끈한 상태이기 때문에 사실감을 전달하기에는 역부족이다. 형태를 갖춘 모델에 사실감을 불어넣기 위해서는 오브젝트의 표면 특징을 보다 정교하게 정의해 주어야 하는데 이를 텍스츄어 맵핑(Mapping)이라고 한다. 오브젝트의 특징에 따라서 모델의 표면에 나무재질과 같은 질감과 색상을 표현하거나 광택이 나는 금속성의 메탈(Metal) 느낌을 만들어 주어야 할 경우가 있다. 또한 모델이 공 모양일 경우 구형 맵핑, 평면 모델링일 경우 평면적 맵핑 스타일을 적용하여 입체모델에 텍스츄어를 입히게 된다. 맵핑 과정을 통해 입혀지는 텍스츄어는 수학적인 공식에 따라 만들어지는 일종의 패턴일 수도 있고 3D 프로그램 외부에서 가져온 그림일 수도 있으며 프랙털 기하학(Fractal geometry)을 이용해 만든 구름이나 나무, 풀잎과 같은 불규칙한 자연물을 표현하는 이미지일 수도 있다. 이렇게 폴리곤이나 스플라인 등으로 만들어진 기본 뼈대(Skeleton)에 표면을 입히는 것은 마치 부챗살 위에 종이나 천을 입혀 면을 만드는 것처럼 골격만 있는 구조에 외피를 입히는 과정이라고 할 수 있다. 최근 3D 소프트웨어는 다양한 방법의 맵핑을 선보이고 있지만 큰 범위 안에서 〈그림 14〉와 같



〈그림 14〉 맵핑 스타일 적용 예제

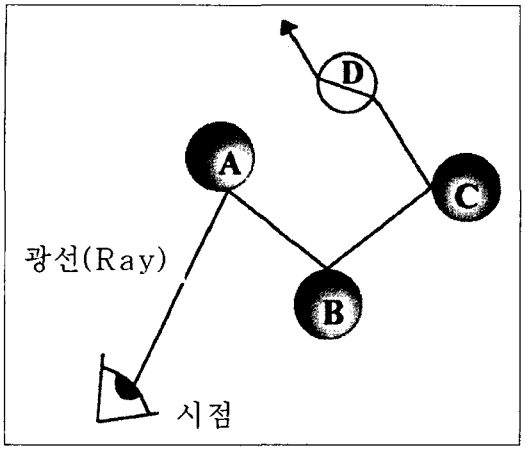
이 텍스츄어(Texture), 투명(Transparency), 반사(Reflection), 요철(Bump)과 같은 종류로 분류해 볼 수 있고 표현하고자 하는 물체의 질감에 따라 여러 종류의 맵을 조합하여 사용하기도 한다.

텍스츄어 맵핑은 비트맵으로 이루어진 오브젝트의 표면에 색과 패턴을 정의하는 것으로 이것은 마치 이미지가 오브젝트 위에 칠해진 듯한 효과를 보여준다. 투명 맵핑은 오브젝트의 표면에 불투명한 부분과 투명한 부분에 반짝이는 하이라이트 효과를 주면 시각적으로 유리와 같은 재질을 만들어 낼 수 있다. 반사 맵핑은 크롬이나 유리 혹은 반사하는 금속성의 표면을 만들어 내는데 유용한 방식으로 오브젝트 주위의 이미지까지 오브젝트 표면에 반사된 모습으로 나타낸다. 요철 맵핑은 표면에 입자감이 느껴지거나 굴곡이 있을 경우 또는 음각되었을 때 주로 사용하는 것으로 객체표면의 비트맵에 일관되게 적용되는 픽셀(Pixel)값을 바꾸어 줌으로써 올록볼록한 엠보싱과 같은 질감을 만들어 내는 방식이다. 이렇게 모델링과 맵핑 작업을 통해서 골격과 외적인 형상이 갖추어지면 완성된 오브젝트를 보다 사실적으로 표현하기 위하여 조명 작업을 진행해야 한다.

Ⅲ. 사실적 이미지의 표현법 분석

1. 수학적 알고리즘의 적용

디지털 가상공간에서 설정된 모델링, 빛, 텍스츄어 맵핑 등의 과정이 이루어지면 3차원의 모든 데이터를 연산 처리해 2차원적인 최종적인 화면으로 만들기 위해서는 렌더링(Rendering) 과정을 거친다. 렌더링이란 결국 이전까지 설정되었던 명령과 수치 값들을 모두 종합하여 계산하는 것이다. 렌더링 작업은 와이어 프레임에 텍스츄어를 입히고, 조명과 그림자를 만들어 가상공간 속에 완성된 입체적인 오브젝트를 2차원적인 이미지로 표현한다. 수학적인 알고리즘에 따라 렌더링 이론이나 방법은 Z 버퍼(Z Buffer), 스캔라인(Scanline), 광선추적법(Raytracing), 조도계산법(Radiosity) 등과 같은 여러 가지 형식으로 전개되며 동일한 장면도 사용자가 선택하는 렌더링 방법에 따라 완성된 이미지에 많은 차이가 나타난다. Z 버퍼 방식은 가상공간속의 Z축, 즉 깊이를 의미하며 시각의 위치에서 보이는 모델의 면만을 렌더링하고 보이지 않는 면들(그림자부분의 보이지 않는 면들을 포함)은 제거하여 렌더링하는 방식이다. 빠른 연산처리로 렌더링 속도가 빠르지만 상대적으로 리얼리즘이 결핍되어 단순한 이미지를 처리하는 경우 사용된다. 스캔라인 방식은 기본적으로 Z 버퍼 방식을 좀더 발전시킨 방법이라 볼 수 있다. 모델링을 한 3차원 공간을 개념적으로 얇게 슬라이스하고 카메라에서 어떻게 보이는가를 각각의 슬라이스를 조사하여 겹쳐 쌓음으로 화상을 만드는 방법이다. 슬라이스의 두께가 어떻게 결정되느냐에 따라 화질의 차이가 나고 일련의 수평선 계산을 통해 이미지를 표현하므로 입체조형 이미지의 미묘한 차이를 만들어 내지는 못하지만 이미지의 선명도가 중요시되는 애니메이션 분야에서는 많이 쓰이는 방식이다. 광선추적법 방식은 광원에서 출발하여 보는 이의 분에 도달하는 광선의 실질적인 동작을 계산하는 방식이다. 광선은 모델에 반사된 후 카메라에 의해 포착되는데 실질적으로 반사광 모두가 카메라에 도달하는 것은 아니다. 따라서 〈그림 15〉와 같이 광선추적법 방식은 시점으로부터 광원까지 반대방향으로 광선을



〈그림 15〉 광선추적법(Raytracing)의 기본 알고리즘

역추적하여 계산의 효율성을 높이는 방법을 이용하고 있다.

이 방식은 반사와 굴절, 투명 효과를 만들어 내는 데 뛰어나서 물과 같이 굴절현상이 심한 모델을 렌더링할 때 유용하다. 조도계산법 방식은 렌더링 방식 중 사진모사적 특성이 가장 강한 결과물을 만들어 내는 방식이다. 모델의 표면의 기하학적인 특성에 따라 반사, 굴절되는 빛을 계산하고 렌더링할 경우 다른 방식에 비해 그림자가 보다 부드럽고 어두운 범위 내에서의 이미지 생산이 보다 정교하게 이루어질 수 있기 때문에 소품질의 디지털 이미지를 생산하는데 매우 중요한 방식으로 평가받고 있다.

2. 알고리즘의 발달에 의한 사실적 표현

디지털 3D 공간에서 입체적인 조형 이미지를 사실적인 이미지로 표현하기위한 수많은 공학적인 기술과 방법들이 개발된다. 그만큼 3D 공간에서 사실적인 표현의 중요성은 최근 몇 년 동안 가장 해결해야할 문제점 중에 하나이다. 컴퓨터로 만든 이미지의 사실감을 증가하기 위해 다양한 효과를 시뮬레이트하는데 이 효과들은 거울 같은 반사(reflection)와 굴절(refraction), 확산 반사(diffuse inter reflection), 스펙트럼(spectrum) 효과 등 다양하다. 이 효과들은 대부분 오브젝트의 표면에 빛이 비추어지면 상호작용하고 시뮬레이션을 매우 간단하게 해결한다. 대부분의 3D 소프트웨어들은 저마다의 새로운 조명셋업들(예를 들어, 3DS Max의 경우 Sunlight, Skylight, Daylight, IES 등 사실적인 태양조명에 대한 추가 기능이 계속적으로 나타난다.)을 선보이고 있고 구체적인 특징들을 가지고 있다. 특히, 최근에는 Vray와 Mental Ray 등과 같은 수준 높은 렌더러의 등장으로 사실적인 이미지 표현을 더욱 쉽게 처리할 수 있도록 도와준다. 그러나 현재의 모든 수준 높은 조명방법의 대부분은 광선추적(Ray tracing)[1]과 조도계산(Radiosity)[2] 알고리즘의 기본에서 출발한다고 보면 될 것이다. 사실적인 이미지를 계산하는 이 2가지 방법들의 공학적인 개념은 매우 복잡하지만 3D 소프트웨어의 시각적인 적용범위에서의 큰 차이점은 광선을 계산하여 표현하는 방법이다. 광선추적법은 빛의 줄기가 각각의 광원으로부터 빛이 나는 객체와 카메라의 그림자까지 추적해가는 3D 렌더링 방법이고 쉐이딩(Shading)과 조도(illumination)는 대단히 실감나는 화면을 제공하며 실제의 굴절, 반사 및 투명

1) 광선추적법(Ray tracing)은 1968년 Appel에 의해 제안된 레이 캐스팅(Ray Casting)에 기반을 두고 있다. Appel이 제안한 개념은 인간이 물체를 볼 수 있는 것은 물체에서 반사된 빛을 감지함으로써 물체를 인식하지만, 물체에 반사된 빛이 관찰자의 눈으로까지 도달하는 경로를 추적하는 것은 불가능하므로 관찰자로부터 물체까지, 그리고 물체에서 광원까지의 경로를 역으로 추적하는 것이다. 초기에는 관찰자의 위치에서 보이는 부분만을 결정하는 선/면 제거를 위한 알고리즘으로 사용되었기 때문에 단순 광선추적법이라고도 하였다.

2) 조도계산법(Radiosity)은 광선추적 알고리즘의 몇 가지 결점을 보완하기 위해 광선추적과는 좀 다른 기술을 연구하기 시작했다. 1980년대 중반, 컴퓨터그래픽 연구원들은 빛의 파장을 사뮬레이트하는데 대해서 이 기술의 응용을 연구하기 시작했다. 이 기술이 컴퓨터그래픽 기술을 발전시킴에 따라 광선추적과 근본적으로 다른 Radiosity(조도계산)란 이론으로 확립되었다. 화면에 나타난 각각의 pixel을 위한 색을 결정하기보다 조도계산은 그 환경에서의 각각의 부분들에 대한 밝기를 계산한다. 이것은 오브젝트 표면의 요소인 Face를 나누는 형식으로 빛에 의한 표면의 밝기 값을 계산한다. 이렇게 계산된 각각의 표면들은 그 값을 지니고 있으므로 카메라를 이동시켜도 재차 조도계산을 할 필요가 없게 된다. 이러한 이론을 바탕으로 한 조도계산방법은 이후 공식화되었으며, 더욱 정확한 이론으로 발전해 왔다. 하지만, 정확한 결과를 얻어내기 위해서는 새로운 이론이었음에도 불구하고 계산에 있어서 상당한 시간이 요구되었다. 1988년, 조도계산 알고리즘은 이론적으로 개선

함을 렌더링하는데 많이 사용된다. 조도계산법은 각각의 광원으로부터 공간에 있는 모든 오브젝트의 표면에 분산되며 계산이 이뤄지는 동안 표면은 다른 물체에 의해 광원으로부터 가려지는 그림자 영역까지 빛의 소스가 확산 전달되게 시뮬레이트한다. 또한 장면 내부에 있는 각각의 광원으로부터 그 에너지를 분산시킨 뒤에 모든 표면을 검사하고 어떤 표면이 어느 만큼의 빛 에너지 반사율을 내포하고 있는지 결정함으로써 조도계산이 계속 진행되는 특징을 보인다.

Ⅳ. 맺음말

3D 공간에서 제작된 장면의 대부분의 특징은 실제 상황 속에서 느끼는 자연현상을 거의 흡사하게 시뮬레이션하는 것이다. 기본적으로 수준 높은 사실적인 장면을 연출하기위한 방법은 현실세계에서 빛과 그림자가 오브젝트에 어떻게 비쳐지는지를 심도있게 이해하고 이러한 기반 안에서 다양한 작업을 실험하는 것일 것이다. 컴퓨터 모델링은 사용자에게 절반의 결과만을 제공한다. 결국 나머지는 사용자의 몫일 것이다. 시각적 표현요소들을 구성하는 것은 무엇이 올바른 구성방법이고 무엇이 잘못된 구성방법이라고 정확하게 말할 수 없을 만큼 복잡하고 일정치 않은 형태를 보여준다. 아마도 이러한 것이 시각적 표현요소의 매력이라고 할 수 있을 것이다. 본 고는 3D 애니메이션의 광범위한 부분 중에서 사실적인 장면 연출을 위해 기본적으로 시각적 표현요소들을 이해하고 최근에 대두되는 광선추적법과 조도계산법의 이해를 통해 3D 애니메이션의 제작공정을 체계적으로 확립하고자 기술하였다.

참고문헌

John Kahrs, 〈Pixel Cinematography〉, 1996, Siggraph'96 Course#30
Atsushi Shiokawa, 〈컴퓨터그래픽스〉 2001, 성안당
김의준, 영화이론총서 제33집〈디지털 영상학 개론〉 1999, 집문당
최이정, 〈영상제작론〉 2004 , 커뮤니케이션북스
토마스 오헤니언, 〈디지털 필름 메이킹〉 1999, 책과길
피터와드, 〈영화 TV의 화면구성〉 2000, 책과길

필자소개

소 요 환

- *1992년 : 홍익대학교 회화과(학사)*
- *1995년 : 홍익대학교 서양화전공(석사)*
- *1998년 : 미국 New York Institute of Technology 필름/애니메이션전공(석사)*
- *2001년~2003년 : (주)비엔씨모바일 연구개발이사*
- *2003년~현재 : 한남대학교 멀티미디어학부 교수*
- *주관심분야 : 3D 애니메이션, 가상현실*