논문 2008-45CI-3-8

Shader Space Navigator: 유사 쉐이더 검색 시스템

(Shader Space Navigator: A Similar Shader Retrieval System)

이 재 호*, 장 민 희**, 김 두 열**, 김 상 욱****, 김 민 호***, 최 진 성*

(Jae-Ho Lee, Min-Hee Jang, Du-Yeol Kim, Sang-Wook Kim, Min-Ho Kim, and Jin-Sung Choi)

요 약

본 논문에서는 그래픽 디자이너의 쉐이더 제작 과정을 효과적으로 지원할 수 있는 쉐이더 검색 시스템에 대하여 논의한다. 쉐이더를 제작하는 과정에는 몇 가지 난점이 존재한다. 첫 째, 쉐이더에 대한 전문적인 지식이 요구된다. 둘 째, 렌더링 시간이 매우 길다. 셋 째, 쉐이더 제작 과정 시 시행착오를 많이 겪는다. 본 논문에서는 이러한 문제들을 극복하기 위하여 유사 쉐이더 검색 시스템인 Shader Space Navigator를 제안한다. Shader Space Navigator는 쉐이더의 에트리뷰트들을 분석하여 다수의 쉐이더 데이터들이 저장되어 있는 쉐이더 데이터베이스 안에서 그래픽 디자이너가 원하는 쉐이더와 매우 유사한 쉐이더를 검색해 준다. 검색된 쉐이더에 몇 번의 간단한 수정만을 가함으로써 그래픽 디자이너가 원하는 최종 쉐이더를 얻을 수 있기 때문에 기존의 난점들을 해결할 수 있다. 본 논문에서는 쉐이더 데이터베이스 구축 방법과 Shader Space Navigator의 구현 이슈에 대하여 논의하고 쉐이더 검색 결과에 대한 예제를 보인다.

Abstract

In this paper, we first point out difficulties faced by CG artists in the shading process: (1) a lot of technical details on shaders required, (2) long rendering time, and (3) repeated trials-and-errors. To make them overcome such difficulties, we propose Shader Space Navigator, a system that efficiently searches for shaders similar to a given query shader from a shader database containing a large number of quality shaders. With Shader Space Navigator, CG artists find appropriate shaders from the database that are very close to the final result shader, and thus complete the shading process easily by slightly tuning some attributes of those shaders. Thus, the CG artists can create their final shaders in an intuitive and efficient way without a large number of time-consuming rendering processes. Also, we deal with implementation issues related to Shader Space Navigator and constructing an abundant shader database in detail.

Keywords : 쉐이더, 렌더링, 유사 쉐이더 검색, 쉐이더 데이터베이스

I. 서 론

렌더링은 컴퓨터 그래픽스 분야에서 매우 큰 비중을 차지하는 기술이다[1]. 렌더링이란, 위치, 조명과 같은 외부의 특성과 3차원 객체의 특성을 결합해서 나타나는 상황을 2차원 화상을 통하여 보여주는 기술을 의미한다[2]. 이러한 렌더링 기술은 영화, 애니메이션, 게임 등 다양한 분야에서 널리 활용되고 있다.

쉐이더란 형태를 제외한 색상, 질감, 무늬 등 객체를 표현하는 요소를 의미한다[3]. 형태가 정해진 객체의 표면에 쉐이더를 입힘으로써 실제와 같은 모습을 화상에

*정회원, **** 평생회원, 한국전자통신연구원
디지털콘텐츠연구본부
(Digital Content Research Division, ETRI)
** 비회원, 한양대학교 전자컴퓨터통신공학과
(Department of Electronics Computer Engineering, Hanyang University)
*** 정회원, 스튜디오 떠다니는 섬
(Studio Floation Island)
※ 본 연구는 정보통신부 및 정보통신연구진흥원의 IT 신성장동력핵심기술개발 사업의 일환으로 수행하였음(2006-S-045-01, 기능 확장형 초고속 렌더러). 또한, 본 연구는 지식경제부 및 정보통신연구진흥원의 대학 IT연구센터 지원사업의 부분적인 지원을 받았음(IITA-2008-C1090-0801-0040).
접수일자: 2008년4월26일, 수정완료일: 2008년5월6일

서 보여준다. 쉐이더는 렌더링 노드들의 네트워크로 표현되고 이를 쉐이딩 네트워크라 한다[3]. 각 렌더링 노드는 객체를 표현하기 위한 고유의 특징을 가지고 있으며, 그 종류는 매우 다양하다. 또한, 각 렌더링 노드는 수십개의 애트리뷰트로 구성된다. 그래픽 디자이너들은 다양한 렌더링 노드들을 유기적으로 조합하여 하나의 쉐이딩 네트워크를 완성하여 렌더링을 수행한다. 렌더링 시스템은 쉐이딩 네트워크 내에 있는 에트리뷰트의 특징을 이용하여 쉐이딩 네트워크에 대응되는 객체를 실감나게 2차원 공간상에 보여준다.

그림 1은 쉐이딩 네트워크의 예이다. 그림 1의 (a)와 같은 쉐이더를 표현하기 위하여 (b)와 같은 복잡한 쉐이딩 네트워크가 사용된다.

그래픽 디자이너가 기존의 렌더링 시스템 상에서 쉐이더를 생성할 때에는 다음과 같은 어려움이 있다. 첫째, 그래픽 디자이너가 원하는 쉐이더를 생성하기 위해서 매우 많은 사전 지식과 노력이 요구된다. 그래픽 디자이너가 하나의 쉐이더를 생성하기 위해서는 그림 1과 같이 다양한 렌더링 노드들을 활용하여 쉐이딩 네트워크를 구성해야 한다. 또한, 각 렌더링 노드들의 수십개의 애트리뷰트들을 일일이 조절해주어야 비로소 그래픽 디자이너가 원하는 모양의 쉐이더가 생성 가능하다. 즉, 하나의 쉐이더를 생성하기 위해서는 모든 렌더링 노드와 그 에트리뷰트들의 의미를 알고 있어야 하며, 렌더링 노드들의 조합과 에트리뷰트 조절을 일일이 수행해 봐야 된다는 것이다.

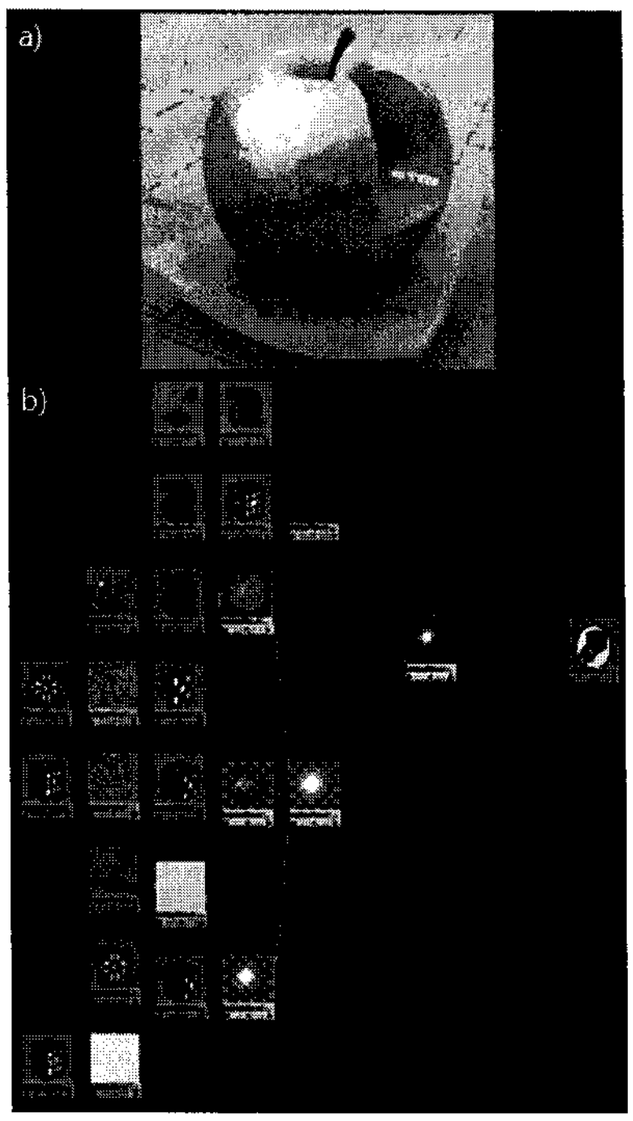


그림 1. 쉐이딩 네트워크의 예
Fig. 1. An example of shading network.

두 번째, 렌더링은 일반적으로 매우 많은 처리 시간을 요구한다[4]. 그래픽 디자이너가 자신이 구성한 쉐이딩 네트워크에 대응되는 객체를 화면으로 확인하기 위해서는 렌더링이 요구된다. 그러나 렌더링은 매우 많은 처리 시간을 요구하는 연산이다. 쉐이딩 네트워크가 복잡할수록 렌더링 시간도 비례하여 증가하며, 수 시간까지 걸리는 경우도 존재한다.

세 번째, 그래픽 디자이너가 원하는 최종 쉐이더를 생성하기 위해서는 쉐이딩 네트워크의 변경과 렌더링 작업의 반복이라는 시행착오를 많이 겪어야 한다. 쉐이딩 네트워크에는 굉장히 많은 에트리뷰트가 존재하는데, 각 에트리뷰트의 값을 조절하였을 때에 최종 그림이 어떻게 나오는지 예측하기가 어렵기 때문이다. 따라서, 각 에트리뷰트마다 값을 변경하면서 그 결과를 렌더링을 통해서 확인해야 한다.

이러한 여러 어려움들로 인하여 결과적으로 하나의 쉐이더를 생성하기 위해서는 굉장히 많은 시간이 요구된다. 만약, 그래픽 디자이너가 만들어놓은 기존의 쉐이더 데이터들을 모아 쉐이더 데이터베이스를 구축할 수 있다면 그래픽 디자이너가 필요로 하는 쉐이더들을 쉽게 찾아낼 수 있다. 이는 위와 같은 어려움을 가지고 있는 그래픽 디자이너들에게 유용하게 사용될 수 있다. 우리는 이 점에 착안하여 유사 쉐이더 검색 시스템인 Shader Space Navigator를 개발하였다.

Shader Space Navigator는 그래픽 디자이너가 원하는 쉐이더와 유사한 쉐이더를 데이터베이스로부터 검색해주는 시스템으로써 쉐이더 생성 시 발생하는 시행착오의 횟수와 시간 낭비를 크게 감소시킬 수 있다. Shader Space Navigator는 그래픽 디자이너에 의하여 만들어진 쉐이더가 다수 저장되어 있는 쉐이더 데이터베이스가 있다는 가정에서 출발한다. 쉐이더 데이터베이스 안의 쉐이더 데이터들은 기본 카테고리로 분류되어 있다. 그래픽 디자이너는 카테고리를 선택한 후 그 카테고리 안의 다수의 쉐이더들 중 원하는 쉐이더를 선택한다. Shader Space Navigator는 그래픽 디자이너가 선택한 쉐이더와 유사한 쉐이더들을 데이터베이스로부터 검색하여 그래픽 디자이너에게 보여준다. 검색된 쉐이더들 중에서 그래픽 디자이너가 원하는 쉐이더를 다시 선택하면 Shader Space Navigator는 좀 더 유사한 쉐이더들을 검색한다. 이와 같은 과정을 반복함으로써 그래픽 디자이너가 원하는 쉐이더와 거의 유사한 쉐이

더를 검색할 수 있다. 마지막으로 선택된 쉐이더의 에트리뷰트를 약간만 수정함으로써 그래픽 디자이너가 원하는 최종 쉐이더를 손쉽게 얻을 수 있다. 저자들이 아는 한 이러한 유사 쉐이더 검색 시스템은 최초로 연구되는 것이다.

본 시스템을 활용하면 기존 렌더링 시스템의 난점을 극복할 수 있다. 몇 번의 검색만으로 원하는 쉐이더와 유사한 쉐이더를 검색할 수 있기 때문에 많은 쉐이딩 네트워크 생성을 위한 사전 지식과 노력이 불필요하다. 또한, 많은 처리 시간을 요구하는 렌더링 작업을 검색 작업으로 대신하기 때문에 렌더링 수행 시간이 크게 줄어 든다. 마지막으로, 화면으로 직접 이미지를 보면서 검색을 수행하기 때문에 기존 렌더링 시스템에서 여러 번 겪었던 시행착오의 횟수가 크게 줄어든다.

본 논문의 구성은 다음과 같다. 제 Ⅱ장에서는 관련 연구로서 기존의 렌더링 시스템과 렌더링 비용을 줄이기 위한 기존 기술들의 특성 및 장단점에 관하여 논의한다. 제 Ⅲ장에서는 본 시스템에서 제안하는 Shader Space Navigator의 기본 전략에 관하여 서술한다. 제 Ⅳ장에서는 다수의 쉐이더들을 확보하기 위한 쉐이더 데이터베이스의 구축 방법에 관해 논의한다. 제 Ⅴ장에서는 본 시스템의 구현 방법 및 인터페이스에 관하여 설명하고 다양한 질의에 대한 검색 결과를 보인다. 끝으로, 제 Ⅵ장에서는 논문을 요약하고 결론을 맺는다.

Ⅱ. 관련 연구

본 장에서는 렌더링의 품질을 높이기 위한 여러 가지 렌더러(Renderer)와 렌더링의 속도를 높이기 위한 여러 가지 기술들을 간략히 소개한다.

렌더링은 3D 애니메이션, 건축 디자인, 게임 제작 그리고 산업디자인 등 3차원 콘텐츠가 필요한 많은 산업에서 널리 사용되고 있다. 그래픽 디자이너가 쉐이더를 잘 만들었다고 하여도 어떠한 렌더러를 사용했느냐에 따라 화면에 보이는 결과가 달라질 수 있다. 실제와 같은 고품질의 렌더링 결과를 얻기 위해 real-time BRDF editing[5], real-time precodural shading[6], ray tracing[7], photon map[8], 그리고 radiosity[9]등, physical characteristic[10]의 렌더러(Renderer)들이 연구되어 왔다. 이 렌더러들은 고품질의 쉐이더를 생산하지만 렌더링의 속도는 고려하지 않았다는 단점을 가지고 있다.

고품질의 렌더링 결과에 못지않게 중요한 것이 렌더링의 속도이다. 아무리 실제와 같은 쉐이더를 생산한다고 하여도 렌더링 속도가 느리게 되면 실제 산업에서 사용하기 힘들다. 이에 따라 렌더링의 속도를 높여 3D 객체 생성 비용을 줄이기 위한 많은 연구들이 진행되어 왔다. 대표적으로 real-time re-lighting system[11], GPU rendering, hybrid rendering, hardware acceleration, 그리고 image-based rendering[12], Polygon Rendering[13] 같은 렌더링 가속 기법들이 있다.

위와 같은 렌더링 기법들은 뛰어난 품질의 렌더링 결과를 비교적 빠른 시간에 얻을 수 있다. 그러나 렌더링이라는 한계에서는 벗어날 수 없기 때문에 기존의 난점들을 모두 극복할 수는 없다. Shader Space Navigator는 유사 쉐이더 검색을 통해 렌더링의 횟수 자체를 줄이기 때문에 기존의 난점들을 쉽게 해결할 수 있다.

Ⅲ. 기본 전략

본 장에서는 Shader Space Navigator의 기본 전략에 대해서 설명한다. 기본 전략을 설명하기 위해 그래픽 디자이너에 의하여 만들어진 다수의 쉐이더가 저장되어 있는 쉐이더 데이터베이스가 구축되어 있다는 것을 가정한다. 본 논문에서는 실제로 대용량 쉐이더 데이터베이스를 구축하여 시스템에 활용하였다. 쉐이더 데이터베이스에 대한 내용은 4장에서 자세히 설명한다.

대용량 쉐이더 데이터베이스에서 그래픽 디자이너가 원하는 쉐이더를 찾는다는 것은 어려운 일이다. 왜냐하면 수많은 쉐이더들이 존재하는 데이터베이스에서 검색의 출발점을 찾기가 어렵기 때문이다. 이를 위해 Shader Space Navigator에서는 쉐이더 데이터베이스를 특성에 맞춰 각 카테고리로 분류하였다. 예를 들어, 유리, 금속, 종이, 플라스틱 같은 재질로 카테고리를 나눈 후 쉐이더 데이터들을 이 카테고리에 맞추어 특성대로 분류한 것이다. 이 외에도 본 시스템에서는 그래픽 디자이너가 좀 더 쉽게 원하는 쉐이더를 찾을 수 있도록 여러 가지 검색 인터페이스를 제공한다. 검색 인터페이스에 대한 내용은 Ⅴ장에서 자세히 설명한다.

미리 정의되어 있는 카테고리들 중에서 그래픽 디자이너가 원하는 카테고리를 선택하면 그 카테고리에 해당되는 쉐이더 데이터들이 그래픽 디자이너에게 보여진다. 그래픽 디자이너는 그 안의 쉐이더 데이터들 중 마음에 드는 쉐이더를 선택하여 Shader Space Navigator에게 선택된 쉐이더와 유사한 k개의 쉐이더를 검색해달라는 질의를 던진다. Shader Space Navigator는 이 쉐이더의 쉐이딩 네트워크에서 사용되는 렌더링 노드들과

그 에트리뷰트들의 값을 분석하여 그 값이 유사한 k개의 쉐이더들을 데이터베이스로부터 검색한다. 이렇게 검색된 k개의 쉐이더들 중 그래픽 디자이너가 최종 결과로 생각하는 쉐이더와 유사한 쉐이더를 다시 선택한다. 선택된 쉐이더를 또 다시 질의로 던지면, 이 쉐이더를 분석하여 그래픽 디자이너가 원하는 결과와 좀 더 유사한 쉐이더들을 검색할 수 있다. 위의 과정을 반복하여 그래픽 디자이너가 원하는 쉐이더와 점점 더 유사한 쉐이더를 검색해 나가기 때문에 최종적으로 그래픽 디자이너가 원하는 쉐이더와 거의 유사한 쉐이더를 검색할 수 있다.

그래픽 디자이너가 자신이 원하는 유사 쉐이더를 검색하였으면 이후부터 실제 렌더링 과정을 통해 최종 쉐이더를 생성해야 한다. 검색 결과로 나온 쉐이더는 그래픽 디자이너가 원하는 최종 쉐이더와 거의 유사하기 때문에 에트리뷰트를 조금만 수정하면 손쉽게 자신이 원하는 최종 쉐이더를 생성할 수 있다.

기존의 렌더링 시스템에서 쉐이더를 생성할 때에는 여러 가지 어려움들이 존재하는데 Shader Space Navigator를 활용하면 이를 쉽게 해결할 수 있다.

첫째, 하나의 쉐이더를 표현하기 위하여 매우 많은 사전 지식과 노력이 필요하다는 문제점을 해결할 수 있다. Shader Space Navigator는 그래픽 디자이너가 렌더링이 끝난 쉐이더 이미지만을 봄으로써 원하는 쉐이더와 유사한 것을 검색할 수 있다. 따라서, 쉐이딩 네트워크에 대한 사전지식이나 에트리뷰트 수정에 대한 고민 없이도 손쉽게 그래픽 디자이너가 원하는 쉐이더를 얻을 수 있다.

둘째, 렌더링의 많은 처리 시간 문제를 해결할 수 있다. 기존 시스템에서는 쉐이더 생성 시 중간 결과를 확인하기 위해 처리 시간이 긴 렌더링을 수행해야 했다. 그러나 Shader Space Navigator에서는 중간 결과를 짧은 검색 시간만을 들여 확인 가능하기 때문에 렌더링의 시간을 크게 단축시킬 수 있다. 즉, 긴 렌더링 수행 시간을 짧은 검색 시간으로 대체한 것이다.

셋째, 쉐이딩 네트워크의 변경과 렌더링 작업의 반복이라는 시행착오의 횟수를 크게 줄일 수 있다. 기존 시스템에서는 에트리뷰트 조절 시 최종 그림이 어떻게 나오는지 예측하기가 힘들다. Shader Space Navigator에서는 최종 이미지들을 눈으로 확인할 수 있기 때문에 단지 몇 번의 검색만으로 원하는 쉐이더를 쉽게 얻을 수 있다. 결과적으로 위와 같은 어려움들을 해결함으로써 그래픽 디자이너의 작업 시간을 크게 단축시킬 수 있는 것이다.

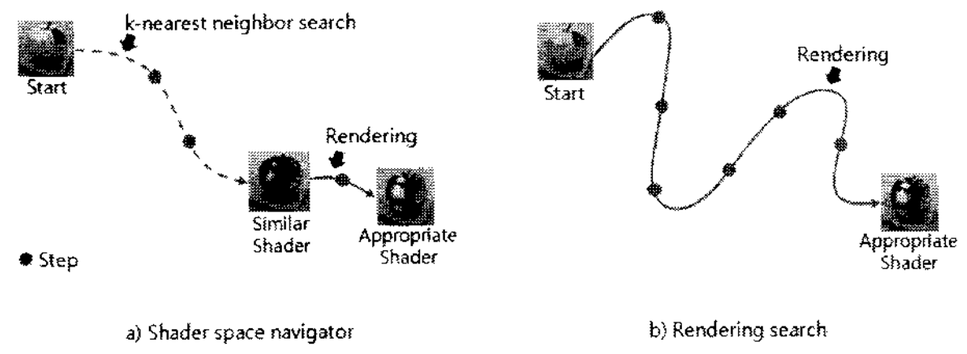


그림 2. 기존 렌더링 시스템과 Shader Space Navigator의 쉐이더 생성 과정
Fig. 2. Comparison of shading processes.

그림 2는 Shader Space Navigator를 사용한 렌더링 시스템과 기존 렌더링 시스템에서의 쉐이더 생성 과정을 그림으로 표현한 것이다.

그림 2(a)는 Shader Space Navigator를 사용한 쉐이더 생성 과정이고 그림 2(b)는 기존 렌더링 시스템의 쉐이더 생성 과정이다. 똑같은 쉐이더를 생성하기 위해 Shader Space Navigator는 단 3번의 과정(step)을 거쳤고 기존 렌더링 시스템은 6번의 과정을 거쳤다. 시행착오의 횟수가 크게 줄어든 것이다. 또한, 그 중에서도 Shader Space Navigator는 2번이 검색 과정이고 단 한 번의 렌더링 과정을 거쳤을 뿐이다. 반면, 기존 렌더링 시스템은 6번의 과정을 모두 렌더링에 사용하였다. 긴 렌더링 시간을 짧은 검색 시간으로 대체하여 렌더링 시간을 큰 폭으로 감소시킨 것이다. 이처럼, Shader Space Navigator를 활용하면 그래픽 디자이너의 작업 시간을 획기적으로 단축시킬 수 있다.

Ⅳ. 쉐이더 데이터베이스의 구축

Shader Space Navigator를 이용하여 원하는 쉐이더를 얻기 위해서는 그래픽 디자이너에 의하여 만들어진 다수의 쉐이더가 저장되어 있는 쉐이더 데이터베이스가 구축되어 있어야 한다. 본 장에서는 Shader Space Navigator의 기반인 쉐이더 데이터베이스의 구축 방법과 쉐이더 생성 시 기본 형태가 되는 샘플 신에 관하여 서술한다.

1. 샘플 신 모델링

서론에서 명시한 바와 같이 쉐이더란 형태를 제외한 색상, 질감, 무늬 등 객체를 표현하는 요소를 의미한다. 따라서 쉐이더의 특성을 올바로 보여주기 위해서는 기본 형태인 샘플 신(sample scene)이 정립되어 있어야 한다. 쉐이더에 대한 렌더링의 결과가 모두 이 샘플 신

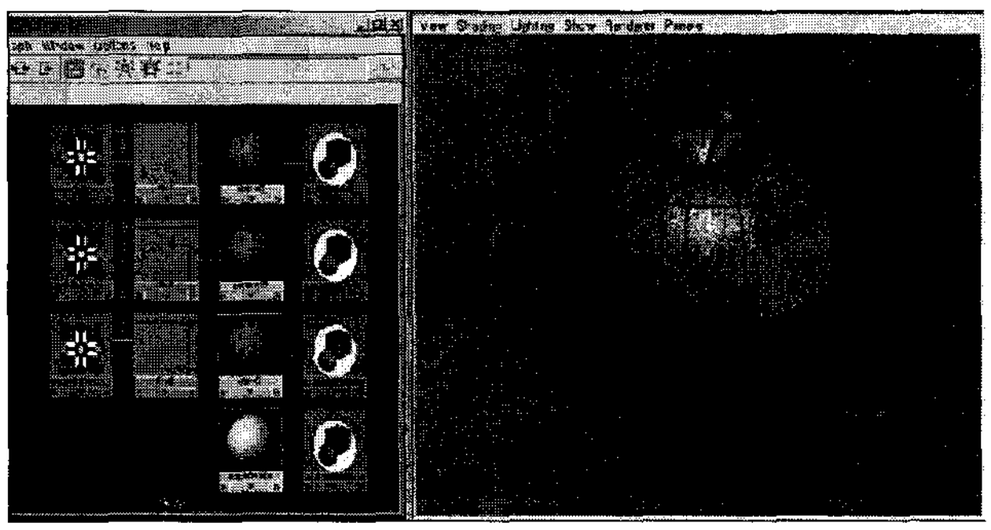


그림 3. Shader Space Navigator의 샘플 신
Fig. 3. A sample scene of Shader Space Navigator.

을 기반으로 보여 지기 때문에 올바른 샘플 신의 정립이 필요하다.

샘플 신이 쉐이더의 특성을 효과적으로 보여주기 위해서는 두 가지 요소가 갖추어 져야 한다. 첫째, 샘플 신은 단순성(simplicity)을 갖추고 있어야 한다. 렌더링 된 쉐이더의 이미지는 매우 작기 때문에 너무 복잡한 형태의 샘플 신은 쉐이더의 특성을 파악하는데 방해가 될수 있다. 따라서 샘플 신은 되도록 단순한 형태를 유지해야 한다. 둘째, 샘플 신은 기능성(functionality)를 갖추고 있어야 한다. 렌더링 된 이미지가 색의 차이, 질감, 굴곡진 정도 등 쉐이더의 다양한 특성을 확인할 수 있도록 해야 한다. 즉, 샘플 신은 최대한 단순하되 렌더링 된 이미지가 쉐이더 샘플로서의 기능을 모두 발휘하게 해야 한다는 것이다.

위의 조건들을 만족시키기 위하여 Shader Space Navigator에서는 사과 모양의 샘플 신을 사용하였다. 그림 3은 Shader Space Navigator에서 사용한 샘플 신을 나타낸다.

그림에서 보는 바와 같이 샘플 신은 단순성을 만족할 수 있도록 구 모양의 형태를 가지고 있다. 또한, 사과라는 샘플 신에 잘린 단면과 접시를 추가함으로써 기능성을 만족하도록 하였다. 잘린 단면을 이용하여 모서리나 각진 곳, 또는 다른 각도에서 쉐이더의 표현 등을 확인할 수 있고, 접시를 이용해서는 쉐이더의 발광이나 광택 정도 등을 확인할 수 있다. 참고로 본 샘플 신은 여러 스튜디오의 그래픽 디자이너들이 직접 테스트 하여 검증을 받은 것이다.

2. 쉐이더 데이터베이스 구축 방법

Shader Space Navigator의 검색이 만족스러운 결과를 가져오기 위해서는 우선 많은 양의 샘플 쉐이더가 확보되어야 한다. 'PIXAR', 'WETA', 그리고 'ILM' 같은 큰 규모의 스튜디오에서 독자적인 쉐이더 데이터베이스를 구축하고 있으나, 일반에 공개하지 않아 양질의 쉐이더 데이터들을 확보하기는 쉽지 않다. 따라서 Shader Space Navigator에서는 양질의 많은 샘플 쉐이더들을 자체적으로 생성하여 쉐이더 데이터베이스를 구축하기로 한다. 우선, 쉐이더 데이터베이스 구축 과정에 관하여 간단히 설명하기로 한다.

1. 쉐이더 카테고리를 정의한다.
2. 그래픽 디자이너를 통하여 각 카테고리마다 시드 쉐이더들을 생성한다.
3. 각 시드 쉐이더의 에트리뷰트 값들을 바꾸면서 샘플 쉐이더들을 생성한다.
4. 그래픽 디자이너를 통하여 생성된 쉐이더들을 분석하여 양질의 쉐이더가 생성될 수 있는 에트리뷰트 값의 범위를 분석한다.
5. 분석 결과를 토대로 에트리뷰트 값을 조절하여 샘플 쉐이더들을 다시 생성한다.
6. 4번과 5번 과정을 반복하여 양질의 샘플 쉐이더들을 생성할 수 있는 에트리뷰트 값의 범위를 결정한다.
7. 최종 결정된 에트리뷰트 값의 범위 안에서 필요한 만큼의 양질의 샘플 쉐이더들을 생성한다.

쉐이더 데이터베이스 구축 과정에 대해 좀 더 자세히 설명하기로 한다. 일단, 그래픽 디자이너를 통하여 쉐이더 데이터를 특성에 따라 종류별로 나눌 수 있는 17개의 카테고리를 설정한다. 그 카테고리는 다음과 같다. architecture, coating, emitter, fabric, glass, grass, ground, liquid, metal, paper, pattern, plastic, skin, stone, wall, wood, extra. 또, 각각의 카테고리는 하위 카테고리로 세분화된다. 예를 들어 metal 카테고리는 gold, silver, iron 같은 하위 카테고리로 세분화된다. 다음으로 분류된 카테고리를 바탕으로 각 카테고리의 특성에 맞는 시드 쉐이더를 생성한다. 그림 4는 17개 각 카테고리에 생성된 시드 쉐이더들을 나열한 것이다. 시드 쉐이더는 전문 그래픽 디자이너의 도움을 받아 카테고리를 대표할 수 있는 쉐이더들로 생성하였다.

시드 쉐이더를 생성한 후, 이를 바탕으로 각 카테고리에 맞는 샘플 쉐이더들을 생성해야 한다. 일단, 쉐이더 생성기를 구현하여 시드 쉐이더의 에트리뷰트 값을 무작위(random)로 바꾸어 첫 번째 샘플 쉐이더들을 생성한다. 이 샘플 쉐이더들은 각 카테고리 성격에 맞는

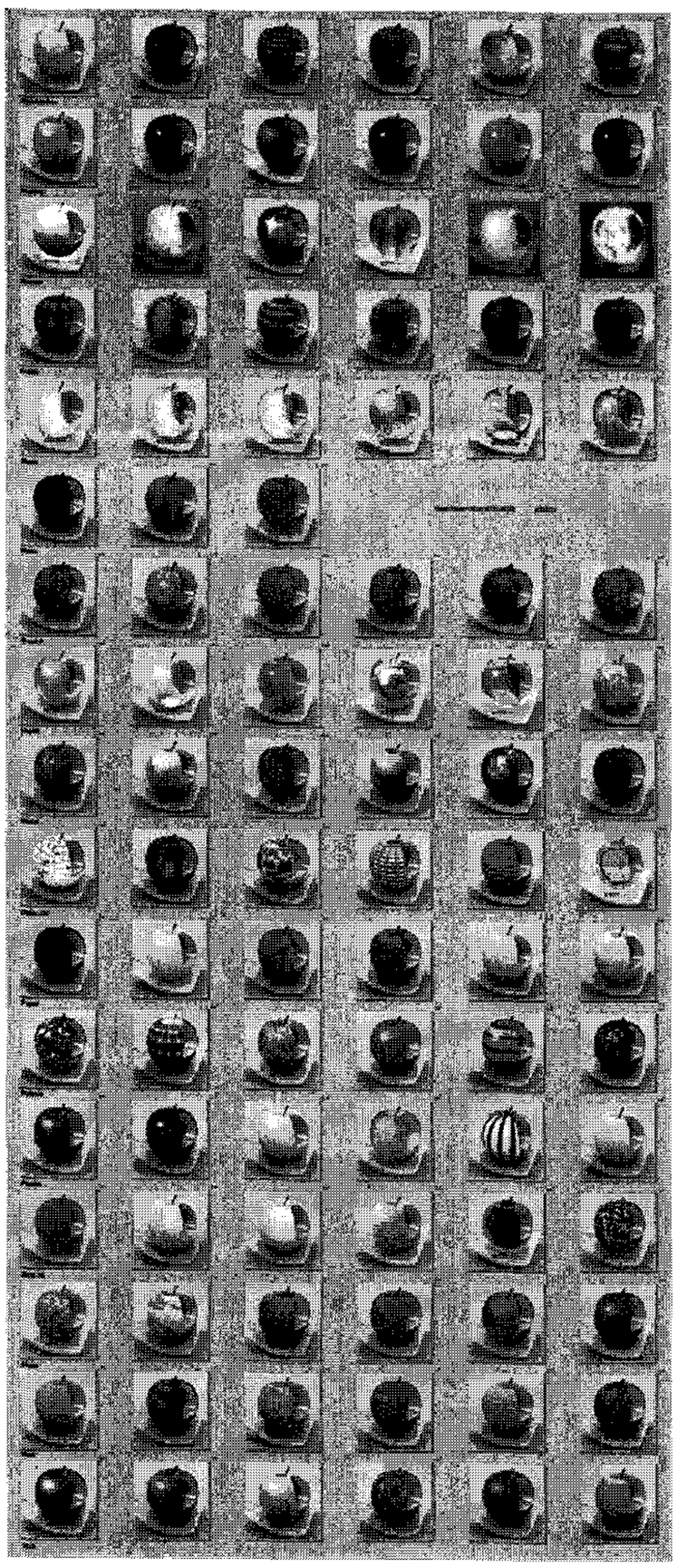


그림 4. 17개 카테고리에 생성된 시드 쉐이더
Fig. 4. Seed shaders in 17 categories.

샘플 쉐이더의 에트리뷰트 값의 범위를 분석하기 위해 사용된다. 그래픽 디자이너가 이 첫 번째 샘플 쉐이더들의 렌더링 된 이미지를 확인하여 카테고리의 성격에 맞게 생성된 쉐이더들을 고른다. 그 후, 그 쉐이더들의 에트리뷰트를 분석하여 각 카테고리 성격에 맞는 쉐이더가 생성될 수 있는 에트리뷰트 범위 값을 조절한다. 이렇게 조절된 에트리뷰트 범위 값을 반영하여 다시 샘플 쉐이더들을 생성한 후 이 데이터들을 또 다시 분석하게 된다.

그림 5는 이러한 샘플 쉐이더 생성 과정을 그림으로 나타낸 것이다. 이러한 피드백 과정을 거치게 됨으로써 각 카테고리의 특성을 반영한 의미 있는 애트리뷰트 범위 값이 설정된다.

최종적으로 설정된 에트리뷰트 범위 값 안에서 샘플 쉐이더들을 생성함으로써 양질의 데이터들로 구성된 쉐

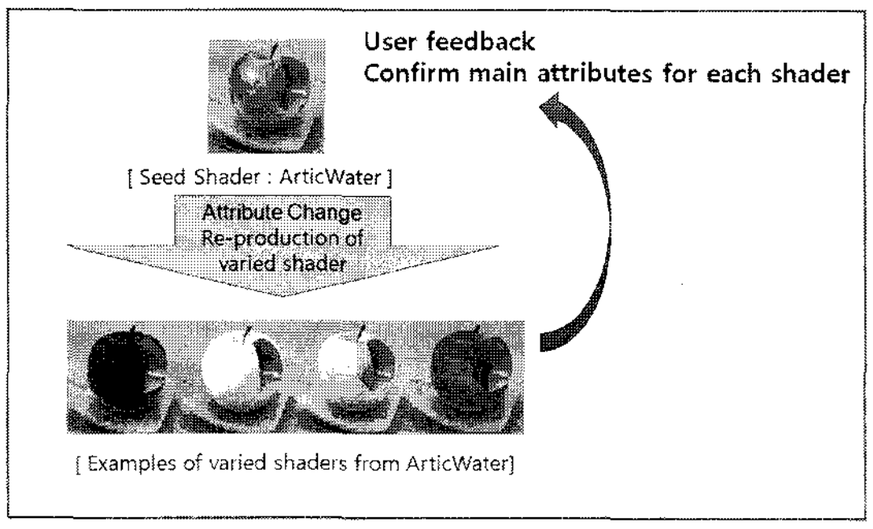


그림 5. 샘플 쉐이더 생성 과정
Fig. 5. Generating synthetic shaders.

이더 데이터베이스를 구축할 수 있다. Shader Space Navigator에서는 위와 같은 쉐이더 생성 과정을 거치면서 각 시드 당 1000개의 데이터를 생성하여 약 100,000개 정도의 샘플 쉐이더들로 이루어진 쉐이더 데이터베이스를 구축하였다. Shader Space Navigator는 다른 그래픽 디자이너가 만든 새로운 쉐이더들을 데이터베이스에 추가할 수 있게 함으로써 쉐이더 데이터베이스가 쉽게 확장될 수 있도록 하였다.

V. 구현 방법

1. 인지감 특성 추출

Shader Space Navigator는 그래픽 디자이너가 원하는 쉐이더를 효과적으로 검색할 수 있도록 하는 시스템이다. 본 시스템에서의 검색은 일반적으로 k-최근접 이웃 검색(k-nearest neighbor search)을 통해 이루어진다. k-최근접 이웃 검색이란, 그래픽 디자이너가 한 쉐이더를 선택하여 질의로 던지면 그와 가장 유사한 k개의 쉐이더들을 유사한 순서대로 검색해주는 질의를 의미한다[14]. 이러한 검색 방법이 수행되기 위해서는 두 쉐이더 간의 유사성을 측정할 수 있는 척도가 요구된다. 두 쉐이더간의 유사성은 쉐이딩 네트워크 구조와 각 렌더링 노드의 애트리뷰트들을 기반으로 측정되어야 한다. 이 경우, 단순히 두 쉐이더의 모든 애트리뷰트에 대해 유클리드 거리를 구할 수도 있지만, 다음과 같은 문제가 발생하게 된다.

비교하고자 하는 쉐이더들의 쉐이딩 네트워크 구조가 다른 경우, 사용되는 애트리뷰트의 수와 그 특성이 각각 다른 가변차원이기 때문에 유클리드 거리 자체를 적용할 수 없다는 문제가 발생한다. 즉 쉐이더마다 사용되는 애트리뷰트가 다르기 때문에 유사도 측정 자체가 불가능하다. 만약 비교하고자 하는 쉐이더들의 쉐이딩 네트워크 구조가 같다면 유클리드 거리로 유사도를

측정할 수 있다. 그러나 하나의 쉐이더는 적게는 수십 차원 많게는 수백 차원으로 이루어진 고차원의 데이터이기 때문에 차원의 저주[14~15]라는 문제가 발생하게 된다. 차원의 저주란(curse of dimensionality), 검색의 대상이 되는 객체의 애트리뷰트 수가 증가함에 따라 두 객체 간 거리에 대한 계산의 정확도 및 성능이 지수함수적으로 감소하는 현상을 말한다. 즉, 쉐이더의 애트리뷰트 수가 증가할수록 두 쉐이더 간의 유사도를 측정하기가 어렵다는 것이다. 따라서 두 쉐이더간 유사도를 측정하기 위해서는 가변차원 및 고차원에 대한 문제들이 해결되어야만 한다.

본 논문에서는 인지감 특성 추출방법을 제안하여 위의 문제들을 해결하고자 한다. 인지감 특성 추출이란, 쉐이더의 쉐이딩 네트워크 구조와 네트워크에 속하는 각 렌더링 노드의 애트리뷰트들의 특성을 분석하여 고정된 저차원인 인지감 특성으로 추출해내는 방법이다. 먼저, 인지감 특성 추출을 위해서 인지감이라는 4가지 감각을 기준으로 각 애트리뷰트들을 특성에 맞게 분류한다. 인지감이란 그래픽 디자이너가 쉐이더를 실제 사물로 인식하기 위해 느끼는 감각으로 본 논문에서는 색감, 무늬감, 재질감의 세 가지로 분류한다.

색감이란 색을 통해 그래픽 디자이너가 느끼는 감각으로 본 논문에서는 하나의 쉐이더를 구성하고 있는 모든 색깔정보들을 의미한다. 예를 들어 어떤 쉐이더가 빨간색을 띈다면 이 빨간색에 영향을 끼치는 모든 애트리뷰트들이 이러한 색감에 포함된다. 무늬감이란 쉐이더의 표면에 나타난 점, 선, 면의 배열들에 의해 나타나는 감각을 의미한다. 예를 들어 쉐이더의 표면에 나타난 체크무늬의 크기나 줄무늬가 얼마나 빈번한가를 나타내는 애트리뷰트들이 이러한 무늬감에 포함된다. 재질감이란 쉐이더가 어떠한 소재로 이루어져 있는지를 표현하는 인지감을 의미한다. 예를 들어, 쉐이더가 유리처럼 투명한 소재인지, 거울처럼 빛을 잘 반사하는 소재인지를 나타내는 애트리뷰트들이 이러한 무늬감에 포함된다.

각 인지감마다 분류된 애트리뷰트들간의 관계와 그 특성을 반영하여 6차원 이하의 인지감 특성 집합으로 추출할 수 있는 식을 도출하였다. 이 식을 통하여 애트리뷰트들을 6차원 이하의 저차원이면서 고정된 인지감 특성 집합으로 추출해내는 것이 가능하기 때문에 앞서 논의했던 고차원 및 가변차원의 문제를 해결할 수 있다. 이러한 인지감 특성 추출 과정은 각 애트리뷰트들의 특성과 관계까지 고려하기 때문에 정확한 검색 결과를 가져온다. 또한, 인지감 특성 추출을 통해 각 인지감별 유사도측정이 가능해짐으로써 다양한 질의 구현이 가능하다.

2. 시스템 인터페이스

그림 6은 Shader Space Navigator의 인터페이스에 대한 그림이다. 시스템의 좌측 아래 카테고리 패널에는 쉐이더 데이터들을 분류한 카테고리들의 리스트가 있다. 가운데 Search Result 패널에는 카테고리를 선택하였을 때에 해당 카테고리에 속하는 쉐이더들의 렌더링된 이미지가 보인다. 좌측 위의 Material Details 패널엔 Search Result에서 선택한 쉐이더의 상세한 이미지가 보인다. 우측의 Control Panel에는 Shader Space Navigator가 제공하는 여러 가지 검색 인터페이스들이 구현되어 있다.

Shader Space Navigator는 k-최근접 이웃 질의를 기본 검색 인터페이스로 제공한다. 본 시스템에서 k-최근접 이웃 질의는 색감, 무늬감, 재질감의 각 인지감별로 검색이 가능하도록 되어 있다. 그래픽 디자이너는 질의를 수행하기 전에 Control panel에서 Number of Result 부분에 검색하고자 하는 쉐이더의 개수를 입력받는다. 그리고 Control panel의 Search Method 부분을 이용하여 세 인지감중 하나를 선택함으로써 해당 인지감에 대한 k-최근접 이웃 질의를 수행한다. Search Method 부분의 마지막에 있는 In this result는 체크 박스로서 현재 검색결과 내에서 다른 검색을 수행하고 싶을때에 사용한다.

Control 패널에서 Color assign 부분은 색기반 k-최근접 이웃 질의를 제공한다. 색기반 k-최근접 이웃 질의는 그래픽 디자이너가 구체적으로 원하는 쉐이더의 모양을 정하지 않았을 때에 가장 눈에 띄는 요소인 색

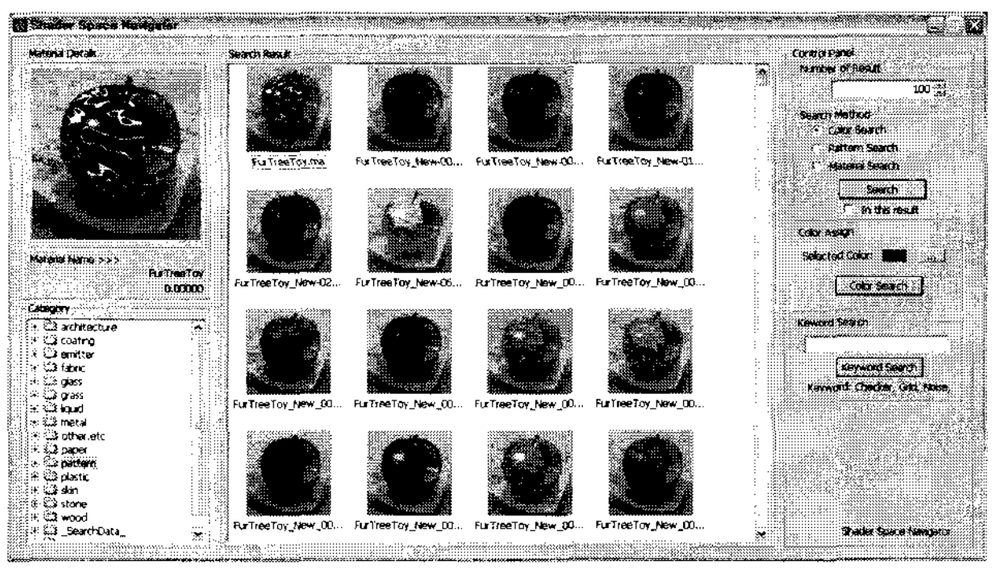


그림 6. Shader Space Navigator의 인터페이스
Fig. 6. Overview of Shader Space Navigator.

만을 이용하여 검색을 시작할 수 있도록 한다. 즉, 그래픽 디자이너가 질의로 던질 쉐이더를 찾지 못했을 때 쉐이더 대신 특정 색을 지정해줌으로써 검색을 가능하게 해주는 인터페이스다.

마지막으로, Control 패널에서 Keyword Search 부분은 키워드 기반 검색을 수행한다. 각 쉐이더가 갖는 특징에 따라 미리 금속, 옷감 등의 키워드를 부여하고, 이러한 키워드를 이용하여 쉐이더를 검색할 수 있는 키워드 검색 인터페이스를 제공한다.

3. 예제

이 절에서는 그래픽 디자이너가 Shader Space Navigator를 통해 원하는 쉐이더를 단계별로 찾아나가는 과정을 예를 들어 설명하기로 한다. 그래픽 디자이너가 파란색 계열의 색감에 체크무늬를 갖고 있으며 금속느낌의 재질감을 갖는 쉐이더를 찾고 싶어 한다고 가정하자. 일단 색기반 k-최근접 이웃질의를 사용하여 수많은 쉐이더 데이터들 중에서 파란색의 쉐이더들을 찾는다. 그림 7은 색기반 k-최근접 이웃 질의를 통해 100개의 파란색 계열 색감을 갖는 쉐이더들을 검색해낸 결과 화면이다.

그래픽 디자이너는 이 검색결과 내에서 마음에 드는 색감의 쉐이더를 선택하여 질의로 던짐으로써, 원하는 바에 더욱 근접한 색감의 쉐이더를 검색할 수 있다.

그림 8은 질의 쉐이더의 색감을 바탕으로 k-최근접 이웃 질의를 수행하여 그래픽 디자이너가 원하는 최종 쉐이더에 더 근접한 100개의 쉐이더들을 검색한 그림이다.

이 검색 결과 내에서 체크무늬를 갖는 쉐이더에 대해 무늬감 검색을 수행해보도록 하겠다. 색감을 파란색 계열로 고정시킨 상태에서 무늬감 검색을 수행하기 위해 현재 검색결과 내에서 검색을 수행하도록 In this result를 체크한다. 그리고 검색 결과 내에서 체크 무늬의 쉐이더를 골라 검색을 수행한다.

그림 9는 이와 같은 과정을 통해 파란색 계열 색감에 Check 종류의 무늬감을 갖는 쉐이더를 검색해 낸 결과 화면이다.

마지막으로, 검색 결과내에서 원하는 재질감을 갖는

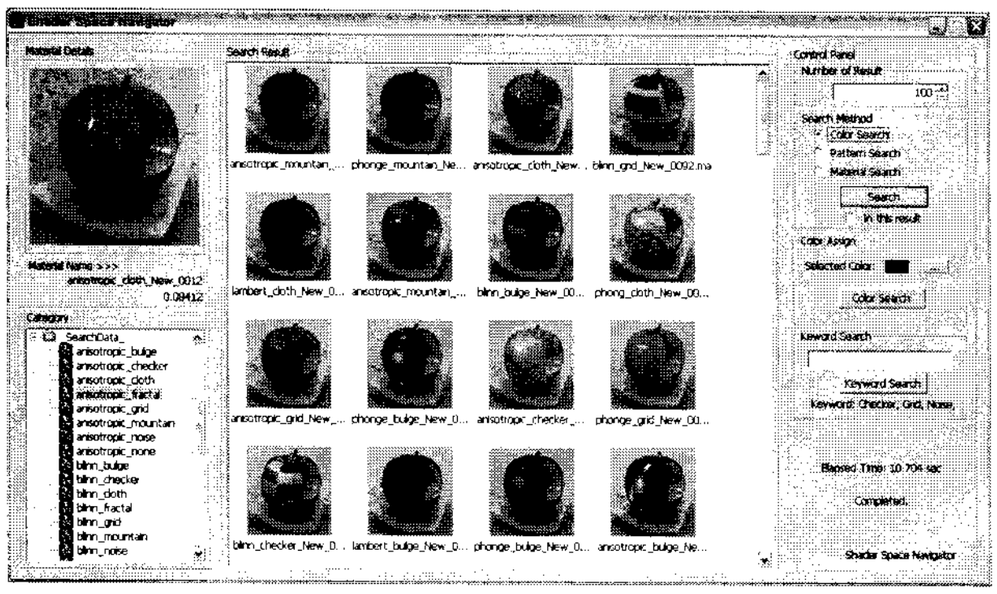


그림 7. 색상기반 k-최근접 이웃 질의의 예
Fig 7. Results of k-nearest neighbor search based on color assign.

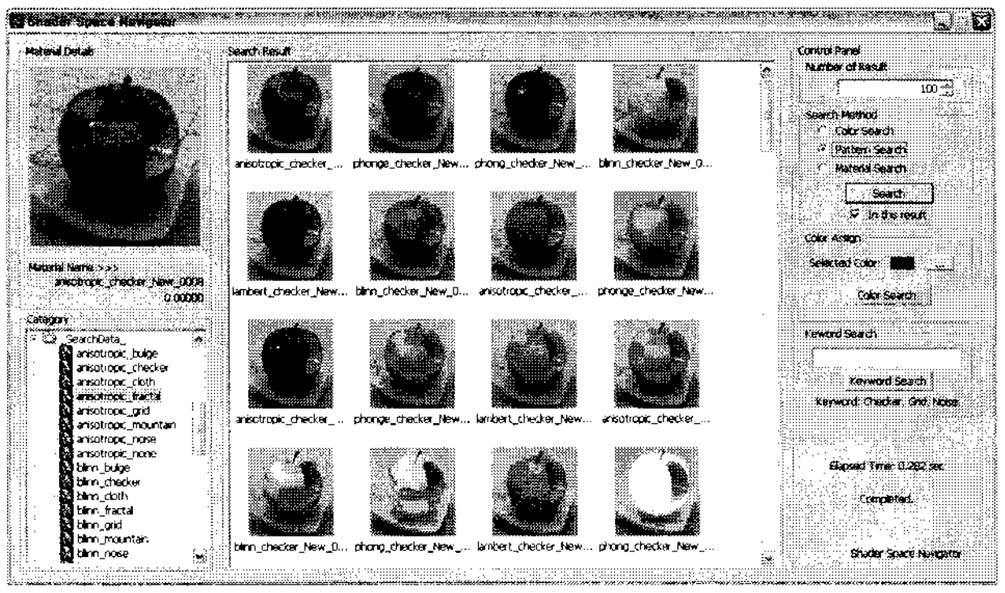


그림 9. k-최근접 질의의 무늬감 검색
Fig. 9. Results of k-nearest neighbor search based on texture perception.

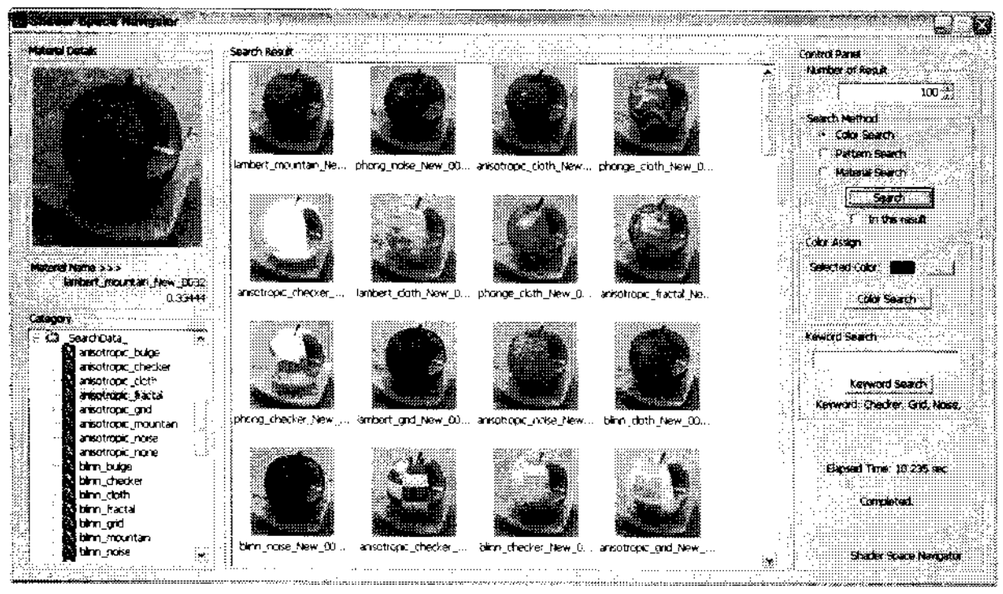


그림 8. k-최근접 이웃 질의의 색감 검색
Fig. 8. Results of k-nearest neighbor search based on color perception.

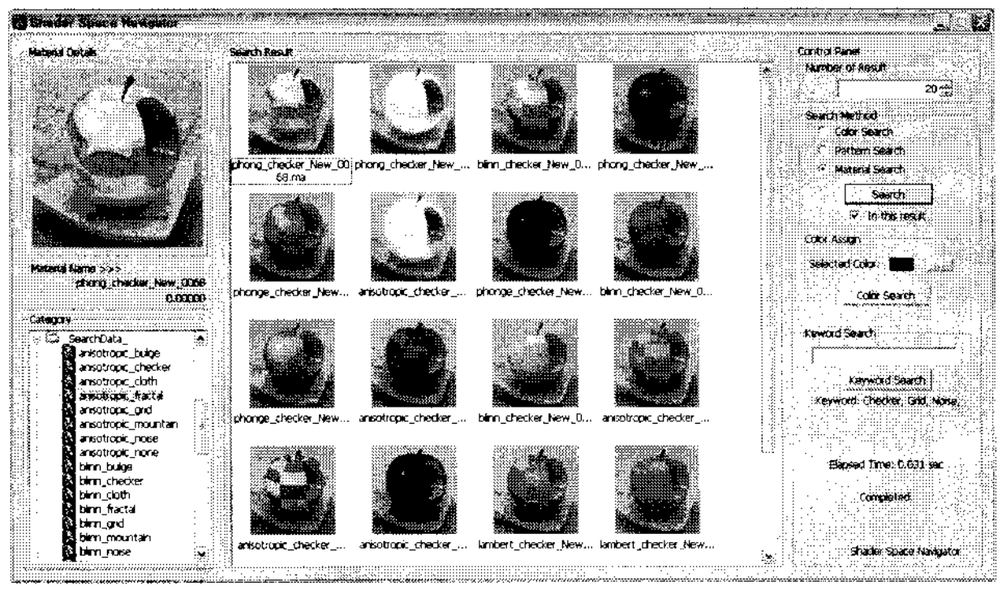


그림 10. k-최근접 이웃 질의의 재질감 검색
Fig. 10. Results of k-nearest neighbor search based on material perception.

쉐이더를 선택하여 재질감 검색을 수행한다. 무늬감과 마찬가지로 In this result를 체크하여 현재 검색 결과내에서 검색을 수행하도록 한다. 검색 결과 내에서 금속처럼 광택이 나고 반사율이 좋은 쉐이더를 골라 검색을 수행한다. 그림 10은 이와 같은 과정을 통해 파란색 계열에 Check 종류의 무늬감을 갖고 있으며 금속 느낌의 재질감을 갖는 쉐이더를 최종적으로 검색해 낸 결과 화면이다.

VI. 결 론

본 논문에서는 그래픽 디자이너가 쉐이더를 제작할 때 발생하는 몇 가지 문제점을 지적하였다. 쉐이더 생성 시에는 다음과 같은 문제점들이 존재한다. 첫 째, 쉐이더에 대한 전문적인 지식이 요구된다. 둘 째, 렌더링 시간이 매우 길다. 셋 째, 쉐이더 제작 과정 시 시행착오를 많이 겪는다. 이러한 문제들을 해결하기 위하여 본 논문에서는 쉐이더 검색 시스템인 Shader Space Navigator를 제안하였다. Shader Space Navigator는 쉐이더의 에트리뷰트들을 분석하여 다수의 쉐이더 데이터들이 저장되어 있는 쉐이더 데이터베이스 안에서 그래픽 디자이너가 원하는 쉐이더와 매우 유사한 쉐이더를 검색해 준다. Shader Space Navigator를 사용하면 검색된 쉐이더 데이터에 간단한 몇 번의 수정만을 가함으로써 그래픽 디자이너가 원하는 최종 쉐이더를 쉽게 얻을 수 있다. Shader Space Navigator가 좋은 검색 결과를 얻기 위해서는 대용량 쉐이더 데이터베이스가 요구된다. 본 논문에서는 양질의 쉐이더 데이터로 구성되어 있는 쉐이더 데이터베이스 구축 방법에 관하여 논의하였다. 또한, 쉐이더의 에트리뷰트들을 효과적으로 비교하여 유사한 쉐이더를 검색할 수 있는 방법에 관하여 논의하였다. 마지막으로 Shader Space Navigator의 실제 검색 예제를 제시함으로써 본 시스템의 유용성을 보였다.

참 고 문 헌

[1] J. Birn, "*Digital Lighting and Rendering*," New Riders Press, 2006.

[2] M. Pharr, and G. Humphereys, "*Physically Based Rendering: from Theory to Implementation*," Elsevier Press, 2004.

[3] S. Upstill, "*The RenderMan Companion: A Programmer's Guide to Realistic Computer Graphics*," Addison-Wesley Professional, 1990.

[4] J. Kajiya, "*The Rendering Equation*," In Proceedings of the 13th annual conference on Computer graphics and interactive techniques, ACM SIGGRAPH, pp. 143-150, 1986.

[5] B. Artzi, A. Overbeck, and R.Ramamoorthi, "*Real-time BRDF Editing in Complex Lighting*," ACM Transactions on Graphics, Vol.25, pp. 945-954, 2006.

[6] K. Proudfoot, W. Mark, and P. Hanrahan, "*Real-time Procedural Shading System for Programmable Graphics*," In Proceedings of the 28th annual conference on Computer graphics and interactive techniques, ACM SIGGRAPH, pp. 159 - 170, 2001.

[7] P. Shirley, and K. Morley, "*Realistic Ray Tracing*," A.K. Peters Press, 2001.

[8] P. Christensen, "*Faster Photon Map Global Illumination*," ACM Journal of Graphics Tools, Vol.4, pp. 1-10, 1999.

[9] C. Goral, K.E. Torrance, and B. Battaile, "*Modeling the Interaction of Light between Diffuse Surfaces*," In Proceedings of the 13th annual conference on Computer graphics and interactive techniques, ACM SIGGRAPH, Vol.18, pp. 213-222, 1984.

[10] 이명영, 이철희, 하영호, "*물리적 특성 모델링에 기반한 라이팅 환경의 렌더링 기법*," 대한전자공학회, 전자공학회논문지-SP, 제 43권 6호, pp. 46~56, 2006.

[11] F. Pellacini, K. Vidim, and J. Warren, "*LPICS: A Hybrid Hardware-accerated Relighing Engine for Computer Cinematography*," ACM Transactions on Graphics, Vol.24, pp. 464 - 470, 2005.

[12] Nimeroff, J. Dorsey, and H. Rushmeier, "*Implementation and Analysis of an Image-based Global Illumination Framework for Animated Environments*," IEEE Transaction on Visualization and Computer Graphics, Vol.2, pp. 283-298, 1999.

[13] 배성옥, 송계근, 경종민, "*고속 다각형 렌더링을 위한 새로운 하드웨어 구조*," 대한전자공학회, 대한전자공학회 학술대회 논문집 제 13권 2호, pp. 651-654, 1990.

[14] Beyer, Jonathan Goldstein, Raghu Ramakrishnan, and Uri Shaft, "When Is Nearest Neighbor Meaningful?," In proceedings of International Conference on Database Theory(ICDT), pp. 217-235, 1999.

[15] Roger Weber, Hans-Jörg Schek, and Stephen Blott, "A Quantitative Analysis and Performance

Study for Similarity-Search Methods in High-Dimensional Spaces," In Proceedings of International Conference on Very Large Databases(VLDB), pp. 194-205, 1998.

저 자 소 개

이 재 호(정회원)

1999년 한양대학교 전자공학과 학사 졸업.
2001년 한양대학교 전자공학과 석사 졸업.
2005년 한양대학교 전자공학과 박사 졸업.
2005년~현재 한국전자통신연구원 재직 (선임연구원).

<주관심분야: 컴퓨터그래픽스, 인공지능, 컴퓨터 비젼>

김 두 열(비회원)

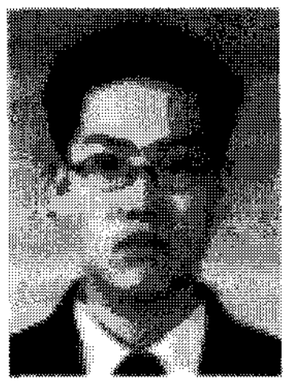
2007년 한양대학교 전자컴퓨터통신공학과 학사 졸업.
2007년~현재 한양대학교 전자컴퓨터통신공학과 석사 재학.

<주관심분야 : 데이터베이스 시스템, 데이터 마이닝, 멀티미디어 정보 검색, 사회 연결망 분석>

김 민 호(정회원)

2001년 극장판 애니메이션 <마리 이야기> 3D Supervisor
2004년 CGI R&D팀 <Nthma> 대표, Supervisor
2006년 극장판 애니메이션 <천년여우 여우비> 3D Supervisor
2008년 현재 디지털 스튜디오 <떠다니는 섬> 대표 겸 실장

<주관심분야 : 컴퓨터 그래픽스, 3D 모델링, 렌더링>



장 민 희(비회원)

2003년 홍익대학교 신소재공학과 학사 졸업.
2006년 한양대학교 전자컴퓨터통신공학과 석사 졸업.
2006년~현재 한양대학교 전자컴퓨터통신공학과 박사 재학.

<주관심 분야: 데이터베이스 시스템, 데이터 마이닝, 멀티미디어 정보 검색, 공간 데이터베이스/GIS, 이동객체 데이터베이스, 사회 연결망 분석>

김 상 욱(평생회원)

1989년 서울대학교 컴퓨터공학과 학사 졸업.
1991년 한국과학기술원 전산학과 석사 졸업.
1994년 한국과학기술원 전산학과 박사 졸업.
1995년~2000년 강원대학교 컴퓨터정보통신공학과 부교수.
2003년~현재 한양대학교 정보통신학부 교수.

<주관심분야 : 데이터베이스 시스템, 데이터 마이닝, 멀티미디어 정보 검색, 공간 데이터베이스/GIS, 주기억장치 데이터베이스, 이동 객체 데이터베이스/텔레매틱스, 사회 연결망 분석, 웹 데이터 분석>

최 진 성(정회원)

1989년 경북대학교 전자공학과 학사 졸업
1994년 경북대학교 전자공학과 석사 졸업
2001년~현재 한국전자통신연구원 재직(팀장)

<주관심분야 : 컴퓨터그래픽스, 가상현실, 과학적 가시화>