

절차적 패턴의 자동 생성을 위한 인터랙티브 디자인 시스템

(Interactive Design System for Automatic Generation of Procedural Patterns)

강재구[†] 황용호^{**} 홍현기^{***}

(Jae-Koo Kang) (Yong-Ho Hwang) (Hyun-Ki Hong)

요약 절차적(procedural) 텍스처는 다양한 해상도로 텍스처 공간을 표현할 수 있으며, 매우 적은 저장 공간을 사용하는 장점을 가지고 있다. 그러나 절차적 표현을 위한 암시적(implicit) 방법을 이해하기 어렵고 프로그래밍에 익숙하지 않은 일반 사용자가 다루기 힘들다. 본 논문에서는 사용자의 직관적인 디자인 작업을 통해 절차적 패턴을 자동으로 생성하는 인터랙티브 시스템이 제안된다. 전체 디자인 과정은 다중 레이어(layer)를 기반으로 하며 데이터, 연산 및 출력 노드 등으로 구성된 그래프 구조로 표현된다. 제안된 시스템은 일반 사용자에게 유연한 인터페이스를 제공하여 프로그래밍 작업 없이 절차적 텍스처 셰이더를 자동으로 생성하도록 한다. 시뮬레이션 결과로부터 제안된 방법이 다양한 패턴을 효과적으로 제작할 수 있음을 확인하였다.

키워드 : 절차적 텍스처, 인터랙티브 디자인 시스템, 암시적 방법, 그래프 구조

Abstract Procedural texture has many advantages that its representation is extremely compact and unfixed in resolution. However, it can be difficult for an end-user having no programming skill to build and debug. This paper presents a new interactive design system for procedural texture generation. The user's interactive design process based on multi-layer operations is represented with a graph structure, which consists of the data node, the operation and the output. Since our system generates automatically a shading language, a user who is not acquainted with implicit concepts can make procedural texturing by using an intuitive and flexible interface. Simulation results showed that the proposed method can make a variety of texture patterns efficiently.

Key words : procedural texture, interactive design system, implicit method, graph structure

1. 서론

컴퓨터 그래픽스에서 물체를 사실적으로 표현하고자 하는 연구는 매우 중요한 분야이다. 특히 다양한 물체의 표면 재질을 효과적으로 표현하기 위한 방법에는 텍스처 매핑(texture mapping)과 절차적 텍스처링(procedural texturing) 등이 있으며, 지금까지 많은 연구가 진행되어 왔다[1].

텍스처 매핑은 물체 공간(space)의 점들을 텍스처 공

간의 점들로 매핑하여 표현하는 방법이다. 여기서 텍스처 공간은 스캐닝이나 디지털 페인팅에 의해 이미지 파일로부터 얻어진다. 텍스처 공간은 이차원 사각형의 픽셀(pixel)로 이루어진 이미지 컬러값의 점들로 구성되며, 물체 공간상의 점에 대한 컬러 값은 텍스처 공간에 위치한 컬러값들을 보간하여 구한다. 그러나 이 방법은 다음과 같은 제한을 지니고 있다. 첫째, 이미지에 표현된 것보다 물체를 표현하기 위해 더 많은 픽셀들이 요구되는 경우, 낮은 해상도의 텍스처로 렌더링된 결과 이미지는 사실감이 떨어지게 된다. 둘째로 해상도가 고정된 원본의 이미지 샘플들은 추가적인 변환들에 의해 변형이나 왜곡된다. 셋째로 모든 이미지는 대부분 메모리에 적재되기 때문에 저장 공간과 메모리 상의 비용 문제가 존재한다.

절차적 텍스처링 방법들은 위와 같은 문제들을 해결

· 이 논문은 2003학년도 중앙대학교 학술연구비 지원에 의한 것이다

† 비회원 : 중앙대학교 첨단영상대학원 영상공학과
daemon@orgio.net

** 학생회원 : 중앙대학교 첨단영상대학원 영상공학과
hwangyongho@hotmail.com

*** 종신회원 : 중앙대학교 첨단영상대학원 영상공학과 교수
honghk@hotmail.com

논문접수 : 2004년 2월 10일

심사완료 : 2004년 7월 6일

하기 위해 알고리즘과 수학적 방법을 통해 컬러값을 결정한다. 이 방법은 텍스처에 대한 정보가 알고리즘들로 표현되기 때문에 해상도의 제한이 없으며, 이미지를 이용하지 않는 등의 많은 장점이 있다. 그러나 절차적 표현을 위해 암시적(implicit) 방법을 이해해야 하며, 실제 제작을 위해 복잡한 프로그래밍이 요구되는 어려움이 있다. 셰이딩(shading) 모델을 재규정(redefine)하여 사용하는 셰이딩 언어가 이미 제안되어 있지만[2], 프로그래밍에 익숙하지 않는 일반 사용자를 위해 절차적 텍스처의 셰이딩 언어를 자동으로 생성하는 연구는 없었다.

본 논문에서는 절차적 패턴을 자동으로 생성하는 인터랙티브 디자인 시스템이 제안된다. 사용자는 다중 레이어(multi-layer) 연산을 기반으로 패턴을 디자인하며, 디자인된 패턴은 절차적 표현을 위한 셰이딩 언어로 자동 변환된다. 표현하고자 하는 패턴을 디자인하는 전체 과정은 그래프 노드의 구성으로 변환되며, 전체 그래프 구조는 패턴의 특징을 나타내는 모양(shape)과 컬러, 그리고 다양한 연산(operation) 작업을 포함하는 그래프 노드들로 구성된다. 제안된 시스템은 패턴 특성의 연산을 통해 다양한 텍스처 패턴을 디자인할 수 있으며, 절차적 텍스처 코드를 자동 생성하여 프로그래밍 과정 없이 절차적 작업을 하도록 한다.

절차적 텍스처링을 2장에서, 텍스처링에 필요한 암시적 표현과 앨리어싱(aliasing) 처리 방법 그리고 임의의 텍스처 패턴을 생성하기 위한 연산들의 조합 방법을 3장에서 설명한다. 4장에서는 유연한 사용자 인터페이스를 위한 그래프 기반의 인터랙티브 디자인 시스템을 소개하고, 이를 이용해 제작된 텍스처 패턴들을 5장에서 보이며, 6장에서 결론을 맺는다.

2. 절차적 텍스처링

스캐닝 등에 의한 이미지를 이용하는 대신에 수학적 방법이나 알고리즘에 의해 텍스처를 생성하는 연구가 지금까지 많이 제안되었다[1]. Cook이 절차적 텍스처를 생성하는 “shade trees” 시스템을 처음으로 소개한 이후, 절차적 텍스처를 위한 언어가 Perlin에 의해 표현되었으며, Hanrahan 등이 Cook과 Perlin의 연구를 기반으로 렌더링 시스템에 사용되는 라이팅과 셰이딩 계산을 위한 새로운 셰이딩 언어를 디자인 하였다[3,4].

일반적으로 절차적 텍스처에 대한 연구는 Perlin의 노이즈(noise) 함수 등과 같이 자연스럽고 다양한 텍스처를 표현하기 위한 알고리즘[5-8]의 개선과 제안이 주로 이루어졌다. 또한 최근에는 게임이나 비행 시뮬레이션 등에서 사용되는 배경 또는 풍경(landscape) 등의 넓은 고해상도 텍스처 공간을 표현하기 위해 주어진 패턴을 비주기적으로 타일링하는 연구가 제안되었으며, 사용자

가 설정한 확률분포(probability distribution) 등에 따라 절차적으로 텍스처를 결정한다[9].

절차적 텍스처는 형태를 이루는 점들을 직접 생성하는 명시적(explicit) 방법과 임의의 시간상에 텍스처의 특정 위치에 대한 질문(query)에 대해 응답하는 암시적 방법이 있다. 명시적 방법은 고정된 순서에 따라 텍스처 패턴을 생성하기 때문에 암시적 방법이 렌더링 알고리즘에 보다 적합하다. 절차적 텍스처는 적은 저장 공간만을 필요로 하고 고정되지 않은 해상도를 지니며, 알고리즘의 수정 없이 관련된 매개 변수나 입력력을 조정하여 텍스처의 속성을 다양하게 변경할 수 있지만, 프로그래밍과 이에 대한 디버그가 쉽다. 또한 이미지를 이용하는 경우에 비해 절차적 방법은 최종적인 텍스처 패턴을 예측하기 힘들다. 본 논문에서는 사용자의 인터랙티브한 디자인 과정을 통해 쉽게 절차적 텍스처를 만드는 시스템이 제안되며, 언어인 텍스처 패턴은 다양한 활용을 위해 렌더맨 셰이딩 언어[10]로 표현된다.

3. 암시적 텍스처 디자인

텍스처 패턴은 영역을 정의하는 모양과 컬러 등의 속성에 의해 기본적으로 표현된다. 인터랙티브한 방법으로 기본적인 모양들을 만들고, 이를 불린(Boolean) 연산으로 조합하여 다양한 패턴을 효과적으로 디자인할 수 있다. 디자인 과정에서 텍스처 패턴은 프리뷰(pre-view) 창에서 계속 확인할 수 있으며, 내부적으로는 암시적 방법으로 표현된다. 본 장에서는 임의의 모양을 정의하기 위한 암시적 알고리즘과 텍스처 특성의 연산을 조합하여 새로운 패턴을 만드는 방법이 소개된다. 또한 절차적 텍스처링 방법의 앨리어싱 문제를 처리하기 위한 암시적 방법에 대해 기술한다.

3.1 모양 정의를 위한 암시적 알고리즘

텍스처 패턴의 모양은 사용자의 입력에 의해 디자인되기 때문에 명시적으로 정의된 임의의 모양을 암시적 방법으로 표현하는 알고리즘이 필요하다. 패턴의 모양은 경계(boundary)와 컬러 정보로 표현되며, 암시적 표현에서 모양은 임의의 점이 닫힌 모양(closed shape)상 내부 또는 외부 영역에 존재하는지를 분류하고 해당 영역의 컬러 값을 결정한다. 먼저, 어떤 점의 위치를 결정하는 점 분류(classification) 알고리즘 중에서 홀/짝수 교차 횟수로 내/외부 영역을 판단하는 Jordan 법칙을 사용한다. 선분으로 구성된 경우, 텍스처 좌표 상의 임의의 점에서 시작되는 레이(ray)를 2D 스캔(scan) 선분으로 이용하여 모양을 이루는 선분과 레이간의 교점을 구한다. 곡선인 경우, 베지어 클리핑 방법을 통해 교점과 그 위치를 구할 수 있다[11]. 스캔 선분과 닫힌 영역의 교차점과 교차 횟수에 따라 내부 및 외부 영역이 결

정되며, 닫힌 영역의 임의의 모양을 암시적으로 정의할 수 있다. 본 시스템에서는 선분과 곡선으로 구성된 2D 닫힌 모양으로 구성된 텍스처 패턴의 디자인을 목적으로 한다.

그림 1은 점 P 의 컬러를 정의하는 암시적 과정을 보인다. 렌더러가 P 의 컬러 값을 요구하면, 시스템은 패턴의 모양을 구성하는 선분 혹은 곡선들과의 교차점을 구하여 내부 혹은 외부 영역을 판단한다. 최종적으로 모양 정보를 바탕으로 해당 영역의 컬러 값을 계산하여 렌더러의 질문에 응답한다.

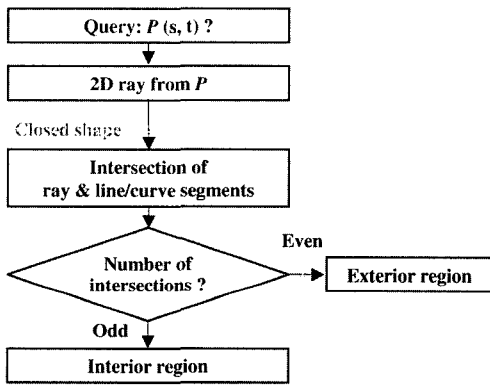


그림 1 모양을 정의하는 암시적 방법

3.2 텍스처 특성의 연산

암시적 방법으로 정의된 기본 모양들을 조합하여 표현하고자 하는 임의의 패턴을 효과적으로 생성할 수 있다. 이 과정에서 모양들에 대해 논리, 산술, 불린 및 2D 변환 등의 다양한 연산들이 적용되며[8], 각 영역에 대해 같은 컬러 값으로 채운다. 그림 2는 바탕 컬러 값을 가진 두 개의 모양(D, S)에 불린 연산을 적용하여 다른 패턴을 디자인하는 예이다. 그림 2의 디자인 과정은 데이터, 연산 및 출력 노드로 구성된 그래프 구조로 함께 표현되어 사용자로 하여금 직관적이며 동시에 효과적으로 텍스처 패턴을 디자인하도록 돕는다.

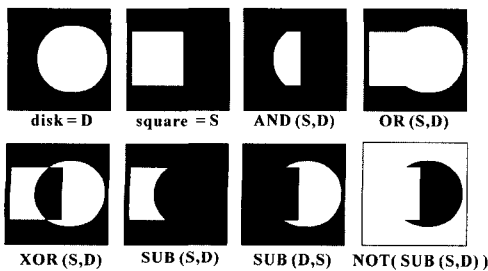


그림 2 불린 연산을 이용한 디자인 과정

3.3 앨리어싱 처리 방법

앨리어싱은 낮은 주파수 샘플링으로 정보가 왜곡되는 현상을 의미한다. 이에 대처하기 위한 안티앨리어싱 방법들이 사용되지 않는다면, 결과 텍스처에서 계단 현상 등이 발생한다. 이미지 텍스처의 경우, 확률론적(stochastic) 슈퍼샘플링(super-sampling) 방법으로 앨리어싱 문제를 해결한다. 그러나 절차적 텍스처링은 임의의 시간에 특정 점에 대한 질의에 응답하는 암시적 방법이기 때문에 특정점에 대한 주변 컬러 정보가 없다. 또한 절차적 텍스처링에 의해 표현되는 공간의 크기가 가변적이기 때문에 후처리(post-filtering)를 위한 이웃 영역의 크기를 결정할 수 없다. 이런 이유로 인해 암시적 표현 방법에서는 슈퍼샘플링 등의 방법들을 전혀 적용할 수 없다.

일반적인 렌더맨, BMRT 등의 렌더링 인터페이스에서는 셰이더 샘플링과 화소 샘플링이 분리되어 있기 때문에 암시적 방법으로 표현된 셰이더를 위한 안티앨리어싱 방법이 필요하다. 본 논문에서는 절차적 텍스처링의 앨리어싱 문제를 해결하기 위해 사전처리(pre-filtering) 기술을 이용한다. 제안된 방법은 질의된 임의의 점이 안티앨리어싱이 필요한 영역인지를 먼저 조사한다. 렌더맨 인터페이스는 임의의 점에 대한 표면의 매개 변수들과 미분 값을 사용하여 샘플링 비율(rate)을 결정하며[1], 본 논문에서도 안티앨리어싱 영역의 결정에 샘플링 비율이 사용된다.

만약 해당 점이 안티앨리어싱 영역 내에 존재하면, 경계로부터 내부 및 외부 영역의 거리 가중치에 따라 컬러 값을 선형 보간한다. 보간 영역들과 거리 가중치(weight)들을 그림 3에 나타내었다. 그림 3에서 앨리어싱이 필요 없는 외부 영역, 경계 및 내부 영역의 거리 가중치는 각각 0, 0.5, 1이며, 경계를 기준으로 외부 앨리어싱[0~0.5]과 내부 앨리어싱[0.5~1] 영역에서 안티앨리어싱을 위한 보간이 수행된다. 그림 3에서 굵은 실선은 모양의 경계이며, 이는 스캔 레이와 질의된 점과의 교차점에 해당한다. 제안된 알고리즘은 표현하고자 하는 물체 공간의 크기에 관계없이 절차적 텍스처링의 앨리어싱 문제를 해결할 수 있다.

4. 인터랙티브 디자인 시스템

제안된 인터랙티브 시스템은 프로그래밍 능력이 없는 사용자가 다양한 절차적 텍스처 패턴을 효과적으로 디자인할 수 있도록 하였다. 본 시스템은 비순환 그래프 구조를 기반이며, 그래프 노드들은 리스트 구조로 저장되고 각 그래프의 연결은 이진 트리 구조이다. 그림 4와 같은 그래프에 대한 일련의 연산과정은 이진 트리의 깊이 우선 탐색을 통해 절차적으로 수행된다. 텍스처 패턴

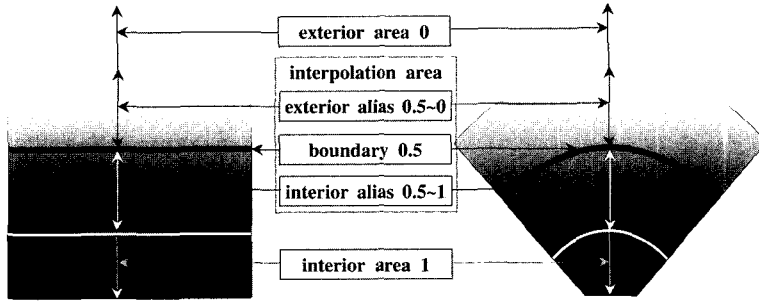


그림 3 엘리어링 영역과 거리 가중치

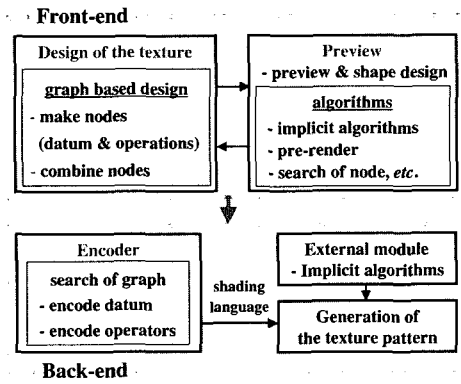


그림 4 시스템 구조

의 특성은 데이터, 연산 및 출력 노드 등을 포함한 그래프 노드들로 표현되며, 이런 노드들을 절차적으로 연결하여 표현하려는 텍스처 패턴을 생성한다.

시스템은 사용자가 인터랙티브하게 텍스처 패턴을 디자인하는 전반(front-end) 부분과 텍스처 패턴을 셰이딩 언어로 생성하는 후반(back-end) 부분으로 구성된다. 3.1절의 암시적 알고리즘과 엔티엘리어링 모듈이 컴파일되어 절차적 텍스처 언어로 사용된다. 결과 텍스처의 다양한 응용을 위해 랜덤한 셰이딩 언어를 기반으로 하며, 알고리즘 모듈은 C-언어로 제작되어 사용된다.

4.1 그래프 구조의 노드들

사용자의 디자인 과정은 내부적으로 그래프 노드들로 표현된다. 제안된 시스템의 그래프 노드는 입력과 출력의 속성에 따라 표 1과 같이 데이터, 연산 및 출력 노드들로 구분된다. 데이터 노드는 임의 패턴의 컬러 및 모양에 대한 정보를 저장하며, 이들 데이터 노드를 연산하여 새로운 모양을 모델링하거나 해당 컬러를 변경할 수 있다. 여기서 그레이 및 컬러 영상에 따라 별도의 연산 과정을 정의할 필요가 없이 동일한 연산이 적용된다. 연산 노드에는 다중 레이어 기반의 이미지 합성에 일반적으로 사용되는 불린, 산술 연산 등[12]이 포함되며, 출력 노드는 입력된 노드의 패턴 결과를 셰이딩 언어 파일로

표 1 그래프 노드의 종류

노드 종류	입력	출력	내용
데이터	0	없음	모양, 컬러
연산	1 또는 2	없음	불린, 논리, 산술
출력	1	0	텍스처 패턴, 프리뷰 영상

출력한다

본 시스템에서 연결된 노드들 사이의 화살표는 데이터 흐름을 의미하고, 세 종류의 노드들은 다른 노드들에 의해 횡수에 제한 없이 참조될 수 있다. 사용자에게 의해 디자인된 텍스처 속성이 데이터 노드에 저장되며, 연산 노드는 연산의 종류에 따라 하나 혹은 두개의 노드를 참조하여 각 노드의 데이터에 대한 연산을 수행한다. 시스템은 임의의 모양을 만들기 위해 바탕 컬러 값은 검은색(RGB(0,0,0))으로, 객체(object)는 흰색(RGB(255, 255, 255))을 기본으로 설정하였으며, 3.1절에서 제시된 암시적 알고리즘이 적용된다.

4.2 인터랙티브 디자인(전반 부분)

다중 레이어를 기반으로 한 절차적 연산들로 패턴을 만드는 과정을 그림 5에 보였다. 그래프 구조에서 각 레이어는 하나의 그래프 노드로 표현되며, 일련의 연산 순서와 데이터의 흐름을 효과적으로 제공한다. 각 그래프 노드에서는 현재까지 디자인된 패턴을 확인하고 필요한 경우는 직접 수정할 수 있으며, 출력노드를 연결하여 바로 셰이딩 언어를 생성할 수도 있다. 그래프 구조로 표현된 각 패턴은 이후 그래프 노드들의 절차적인 연산을 통해 최종적으로 디자인 하려는 패턴을 완성할 수 있다.

그래프 구조를 구성하는 그래프 노드들을 이용하여 패턴을 디자인하는 과정을 그림 6에 보였다. 선택된 그래프 노드에서의 결과 패턴을 프리뷰에서 미리 확인할 수 있다. 그림 6의 점선 박스 안의 패턴들은 사용자의 디자인 과정에 따라 각 그래프 노드들의 다양한 연산에 의해 순서적으로 얻어진 결과들이다.

먼저 사용자는 데이터 노드 중에 기본 모양 노드를 생성하여 곡선 또는 선분 성분으로 패턴의 모양을 디자

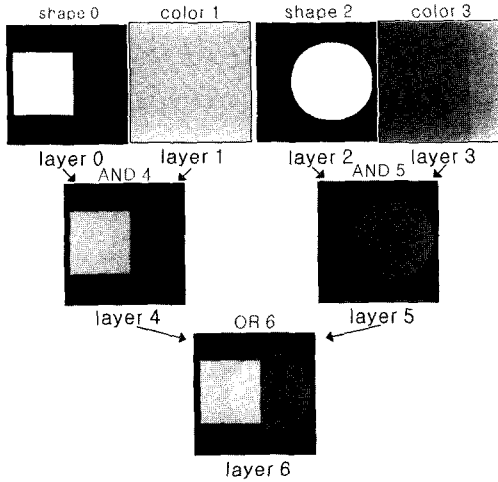


그림 5 다중 레이어에 기반한 연산과정

인한다. 그리고 컬러 값을 기반으로 다양한 연산 노드를 조합하여 패턴을 절차적으로 생성한다. 표현하려는 패턴의 노드에 출력 노드를 연결함으로써 셰이딩 언어로 작성된 절차적 텍스처 코드 파일을 최종적으로 생성한다. 이런 디자인 과정에는 프로그래밍 작업이 전혀 포함되지 않으며, 따라서 암시적 방법에 대한 이해와 프로그래밍 능력이 없어도 인터랙티브한 디자인 과정을 통해 절차적 텍스처 방법을 위한 셰이더를 제작할 수 있다. 본 시스템은 데이터 노드와 연산 노드를 조합하여 패턴을 생성하는 그래프를 디자인하며, 이런 인터랙티브한 디자인

인 작업 과정들은 이미지 합성 소프트웨어인 Shake와 유사하다[13].

4.3 셰이딩 언어 생성(후반 부분)

디자인된 패턴을 표현하는 그래프 구조는 이진 트리의 깊이 우선 탐색 알고리즘 기반으로 하며, 각 노드는 셰이딩 언어로 변환되어 최종으로 절차적 텍스처링을 위한 셰이더 파일이 생성된다. 후반부분은 텍스처 속성들에 대한 인코드(encode) 부분과 많은 계산량을 요구되는 알고리즘 부분으로 구성된다. 다양한 텍스처 속성들이 저장된 데이터 노드와 일련의 연산 노드들은 셰이딩 언어로 인코드된다. 그리고 계산량이 많은 암시적 알고리즘과 안티앨리어싱 처리는 C-언어로 작성된 외부 모듈로 제작되어 사용되고 셰이딩 언어로 인코드 되지 않는다. 디자인된 패턴을 3D 콘텐츠 제작 등의 분야에 다양하게 활용하기 위해 널리 사용되고 있는 랜더맨 셰이딩 언어로 작성하였다. 다양한 랜더맨 셰이딩 언어는 외부 모듈을 제작하여 사용할 수 있으며, 외부 모듈의 사용은 반복되는 코드 생성을 피할 수 있는 장점이 있다. 따라서 셰이딩 언어는 데이터 노드들의 텍스처 속성과 연산 노드들의 연산에 대한 인코딩 부분을, C-언어로 작성된 모듈은 암시적 알고리즘과 안티앨리어싱 처리를 각각 담당한다. 또한 컬러나 불투명도(opacity)를 매개 변수로 제공하여 텍스처의 특징을 쉽게 조절할 수 있다. 랜더맨 셰이딩 언어로 제작된 셰이더는 표면 셰이딩 모델로 생성되며, 제작된 셰이더는 랜더링 시 요구되는 텍스처 정보를 효과적으로 표현할 수 있다.

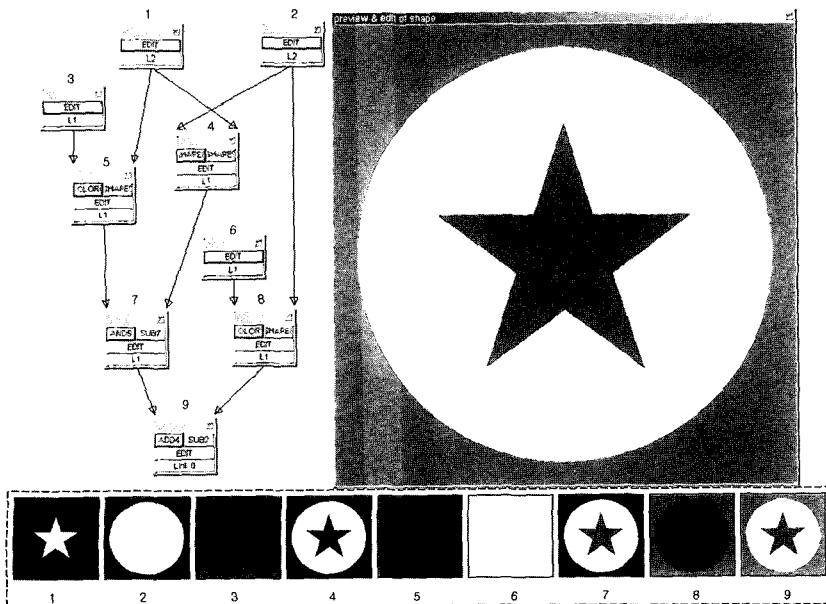
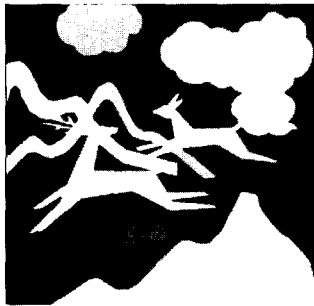


그림 6 디자인된 그래프 구조(왼쪽), 프리뷰(오른쪽) 및 결과 텍스처 패턴(아래)

표 2 텍스처 패턴의 그래프 노드 구성

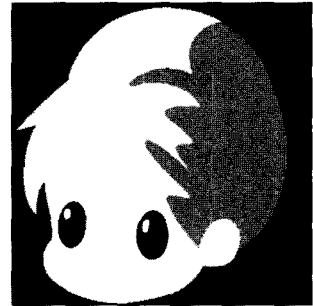
텍스처 패턴	그래프 노드									
	데이터 노드		연산 노드							
	Shape	color	AND	OR	XOR	NOT	ADD	SUB	MUL	DIV
사슴과 산	11	11	12	9		4				
펍시 로고	9	6	2	7				2		
소녀 얼굴	9	4	6	7	2			1		1
스페이드 에이스	8	2	2	5				3		
CAU 로고	6	3	3	5	1					
Luxo's ball	2	2	3					2		



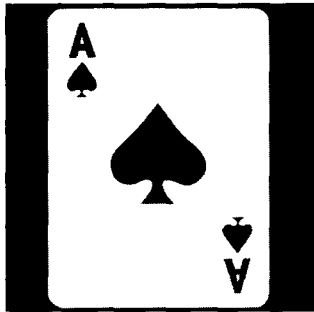
(a) 사슴과 산



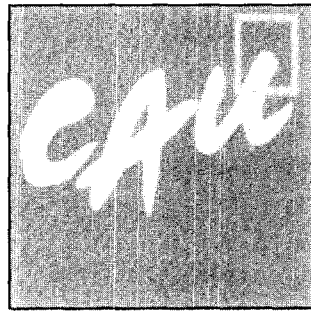
(b) 펍시 로고



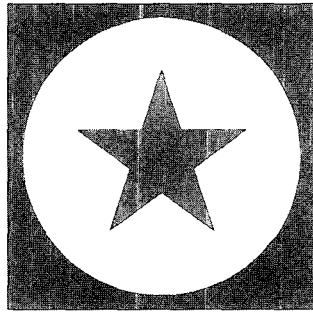
(c) 소녀



(d) 스페이드 에이스



(e) CAU 로고



(f) Luxo's ball

그림 7 디자인된 텍스처 패턴들

5. 시뮬레이션 및 결과 패턴

제안된 시스템 상에서 제작된 6개의 패턴을 그림 7에 보였으며, 이를 위한 사용자의 인터랙티브한 디자인 과정은 데이터, 연산 및 출력 노드 등의 그래프 노드로 표현된다(표 2 참조). 본 시스템은 노이즈 함수 등에 의해 표현될 수 있는 자연스럽고 다양한 텍스처를 목적으로 하지 않으며, 임의의 형태를 가지는 패턴을 다중 레이어 기반의 그래프 작업을 통해 자동으로 생성한다. 즉, 비교적 단순하지만 다양한 해상도로 사용될 수 있는 회사 로고나 마크 등의 디자인에 효과적이다. 다중 레이어 연산에 익숙한 사용자는 인터랙티브한 디자인 과정을 통해 몇 분 내로 질 좋은 패턴을 만들 수 있다. 또한 절차적 텍스처이기 때문에 컬러, 크기 등의 텍스처 매개 변수도 매우 쉽게 수정할 수 있는 장점이 있다. 일반적으로

로 표현하고자 하는 패턴이 매우 복잡한 경우에는 노이즈 함수를 활용하는 기존의 절차적 알고리즘이나 이미지 텍스처를 사용하는 것이 보다 적합한 방법이며, 디자인의 복잡한 정도에 의해 요구되는 비용에 따라 사용자가 텍스처링 방법을 선택해야 한다.

6. 결론

본 논문에서는 다양한 패턴을 절차적으로 디자인하고 이에 대한 웨이더를 자동으로 생성하는 새로운 인터랙티브 시스템이 제안되었다. 사용자의 디자인 과정은 기존의 절차적 텍스처링 알고리즘을 바탕으로 그래프 구조상의 데이터, 연산 및 출력 노드들로 표현된다. 제안된 시스템은 프로그램에 익숙하지 않은 사용자에게 직관적이면서도 유연한 인터페이스를 제공하며, 인터랙티

브한 디자인 과정들을 통해 다양한 텍스처 패턴을 절차적으로 제작할 수 있는 장점이 있다. 보다 다양한 패턴을 표현할 수 있는 새로운 암시적 알고리즘의 구현과 최적화에 대한 고려가 이후에 연구될 예정이다.

참 고 문 헌

- [1] D. S. Ebert, F. K. Musgrave, D. Peachey, K. Perlin, and S. Worley, *Texturing & Modeling A Procedural Approach Second Edition*, AP Professional, 2003.
- [2] RenderMan Document "Adding C Functions to Shading Language with DSOs," Sep. 1999.
- [3] Cook, R.L., "Shade trees," Proc. ACM SIGGRAPH, Vol.18, pp. 223-231, Jul. 1984.
- [4] P. Hanrahan and J. Lawson, "A language for shading and lighting calculations," Proc. ACM SIGGRAPH, Vol.24, pp. 289-298, Aug. 1990.
- [5] S. Worley, "A cellular texture basis function," Proc. Conf. on Computer Graphics and Interactive Techniques, Vol. 23, pp. 291-294, Aug. 1996.
- [6] Z. Joseph, R. Yaniv, D. Lischinski, and M. Werman, "Texture mixing and texture synthesis using statistical learning," IEEE Trans. on Visualization and Computer Graphics, Vol. 7, No. 2, pp. 120-135, 2001.
- [7] K. Perlin, "Improving noise," Proc. ACM SIGGRAPH, Vol. 21, pp. 681-682, Jul. 2002.
- [8] K. Perlin, "An image synthesizer," Proc. ACM SIGGRAPH, Vol.19, pp.287-296, Jul. 1985.
- [9] S. Lefebvre and F. Neyret, "Pattern based procedural textures," Proc. ACM Symposium on Interactive 3D Graphics, Vol. 15, pp. 203-212, April 2003.
- [10] A. A. Apodaca, Larry Gritz, *Advanced RenderMan*, pp. 170-194, 2002.
- [11] T. Nishita, T. W. Sederberg, and M. Kakimoto, "Ray tracing trimmed rational surface patches," Proc. ACM SIGGRAPH, Vol.24, pp. 337-345, Aug. 1990.
- [12] S. Wright, "Digital compositing for film and video," Focal Press, 2002.
- [13] Nothing Real Shake ver. 5.1080 (<http://www.apple.com/shake/>).



황 용 호

1996년 명지대학교 전자공학과 공학사
 1998년 명지대학교 대학원 전자공학과 공학석사. 1998년 1월~2000년 3월 (주) 윤의 C&C 부설연구소 연구원. 2000년 9월~2003년 7월 (주)콤포텍시스템 기술연구소 주임연구원. 2003년 9월~현재, 중앙대학교 첨단영상대학원 영상공학과 박사과정 재학 중. 관심분야는 컴퓨터그래픽스, 컴퓨터 비전 등임



홍 현 기

1993년 중앙대학교 전자공학과 공학사
 1995, 1998년 중앙대학교 대학원 전자공학과 공학석사, 공학박사. 1998년 9월~1999년 8월 서울대학교 자동제어특화연구센터 연구원. 1999년 9월~2000년 2월 중앙대학교 정보통신연구소 연구교수. 2002년 2월~2003년 1월 Univ. of Colorado at Denver 방문 연구원. 2000년 3월~현재, 중앙대학교 첨단영상대학원 영상공학과 조교수 재직 중. 관심분야는 컴퓨터그래픽스, 컴퓨터 비전, 멀티미디어 응용 등임



강 재 구

2002년 건양대학교 컴퓨터공학과 공학사
 2004년 중앙대학교 첨단영상대학원 영상공학과 공학석사. 관심분야는 컴퓨터그래픽스, 멀티미디어 응용 등임