하이퍼볼릭 패턴 생성을 위한 백워드 매핑
(Backward Mapping Method for Hyperbolic Patterns)

조 청 운 †
(Jho, Cheung Woon)

요 약 일반적으로 하이퍼볼릭 공간상에서 대칭 패턴을 생성하는 알고리즘은 벡터표현 방식에 기반한 포워드 매핑 알고리즘을 사용한다. 기존의 알고리즘에서는 복사한 대칭 패턴을 표현하는 레이어의 증가에 따라 메모리의 사용이 기하급수적으로 증가한다. 이러한 문제점으로 인해 전체 패턴의 정밀한 표현이 불가능하다. 또한 기본 패턴으로 비트맵 이미지를 사용하기 어렵고 벡터형태의 결과를 이미지로 변환하는 추가의 처리를 필요로 한다. 본 논문에서는 하이퍼볼릭 공간에서 대칭 패턴을 생성하는데 있어 정밀하고도 효율적인 계산 방법인 백워드 매핑 알고리즘을 제안한다.

키워드 : 하이퍼볼릭 패턴, 백워드 매핑, 포워드 매핑, 타일링

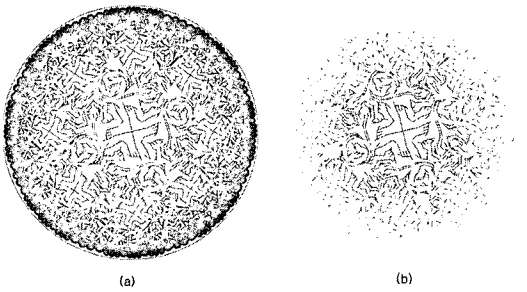
Abstract Most existing algorithms adopt the forward mapping method that is based on vector representation. Problem of existing algorithms is the exponential increase of memory usage with number of layers. This degrades the accuracy of the boundary pattern representation. Our method uses bitmap representation and does not require any additional post-processing for conversion of vector-form results to bitmap-form. A new and efficient algorithm is presented in this paper for the generation of hyperbolic patterns by means of backward mapping methods.

Key words : hyperbolic pattern, backward mapping, forward mapping, tiling

1. 서 론

하이퍼볼릭 패턴의 시각화에 대한 연구는 유클리디언(Euclidean) 평면에서의 패턴 시각화에 비해 비교적 많이 알려져 있지 않다. 수학자 Coxeter의 연구와 네덜란드 판화가 에셔(M. C. Escher)의 작업으로 하이퍼볼릭 패턴에 대한 연구가 시작되었다. Circle Limit 시리즈 I~IV로 예술가와 수학자들에게 하이퍼볼릭 패턴의 시각화 분야가 관심을 얻게 되었다. 유한한 원 안에 무한의 패턴을 시각화 하는 작업은 컴퓨터를 이용한 생성 알고리즘이 나오기 전까지는 다양한 연구가 이루어지지 않았다. 유클리디언 평면에서 패턴을 생성하는 경우와 달리 하이퍼볼릭 공간에서 패턴을 생성하기 위해서는 패턴의 복잡한 변환이 필요하고 이로 인해 하이퍼볼릭 공간상에서 다양한 패턴을 만들거나 타일링하는 방법에 대한 연구가 매우 제한적으로 이루어졌다.

2절에서는 관련된 연구내용에 대해 살펴보고 3절에서는 하이퍼볼릭 모델에 관련된 개념들을 정리한다. 4절에서는 본 논문에서 제안한 알고리즘과 이를 이용한 결과를 보여준다. 5절에서는 결론과 앞으로의 연구방향을 살펴본다.



(a) (b)

그림 1 기존의 생성 알고리즘(a)과 제안된 알고리즘에 의해 생성된 이미지(b)의 비교

2. 관련연구

Dunham은 컴퓨터의 프로그램으로 하이퍼볼릭 평면상에서 반복적인 패턴을 만드는 알고리즘을 개발했다[1].

† 정 회 원 : 중앙대학교 첨단영상대학원 영상공학과 교수
cwjho@dreamwiz.com
논문접수 : 2002년 8월 26일
심사완료 : 2003년 2월 7일

이후에 이와 관련된 지속적인 연구에서 기본적으로 벡터 방식의 기반을 한 포워드 매핑(forward mapping) 알고리즘을 이용했다[2,3,4,5,6,7]. Bruce et al.은 Affine 변환과 하이퍼볼릭 평면상에서의 동치 관계를 생성할 수 있게 확률적인 IFS(Iterated Function System) 방법으로 프랙탈 형태의 이미지들을 만들어냈다[8]. 그러나 기본적으로 포워드 매핑을 이용한 방법으로 일반적인 이미지에 대한 적용이 불가능하다. 따라서 생성되는 이미지는 프랙탈 패턴과 같이 확률적인 방법에 의해 생성되는 추상적인 형태만을 얻을 수 있다. [9]에서 Chung et al.은 Dynamics를 이용한 패턴 생성 방법을 제안했다. [10]에서 이러한 방법을 이미지 생성에 맞게 알고리즘을 개발했지만 $[p, q]^+$ 타입의 대칭군(Symmetry Group)에 대해서만 적용할 수 있는 알고리즘이라는 제한이 있다. 따라서 [p, q]나 $[p^+, q]$와 같은 일반적인 대칭군에 대해 적용할 수 있는 알고리즘이 필요하다.

3. 하이퍼볼릭 모델

3.1 하이퍼볼릭 기하와 관련된 모델

하이퍼볼릭 패턴은 하이퍼볼릭 기하 모델 중에서 주로 Poincare 모델로 표현한다. Poincare모델에서 평면의 점은 복소수 평면 상에서 단위원 안의 점들로 표현된다.

$$U=\{z\in C \| z | <1\}$$

또한 Poincare 모델에서 평면의 직선은 단위원에 직각으로 교차되는 원호이거나 중심점을 지나는 직경에 해당하는 직선이다. Poincare 모델에서 두 점을 지나는 원호를 구하기 위해서는 [11]의 방법을 사용해서 구할 수 있다. 하이퍼볼릭 정다각형은 등변의 하이퍼볼릭 직선으로 구성되는 영역이다. 본 논문상에서 표현되는 대부분의 이미지들은 이 모델 상에서 표현된 것이다.

Poincare모델과 관련된 다른 하이퍼볼릭 모델은 Weierstrass 모델이 있다.

$$W=\{(x,y,z)^T\in R^3 \mid z^2-x^2-y^2=1, z>0\}$$
$$z=\sqrt{x^2+y^2+1}$$

반쪽의 하이퍼볼로이드(hyperboloid) 상의 점들의 집합으로 표현되는 이 모델은 패턴의 대칭이동에 이용된다. 두 모델간의 변환은 다음과 같이 이루어진다.

$$g^{-1}:U\to W: g^{-1}\left(\begin{bmatrix}x\\y\\0\end{bmatrix}\right)=\frac{1}{1-x^2-y^2}\begin{bmatrix}2x\\2y\\1+x^2+y^2\end{bmatrix}$$

$$g:W\to U: g\left(\begin{bmatrix}x\\y\\z\end{bmatrix}\right)=\frac{1}{1+z}\begin{bmatrix}x\\y\\0\end{bmatrix}$$

Poincare모델의 좌표로 표현되는 패턴은 g^{-1}에 의해 Weierstrass모델로 표현되고 식 (2)~식 (5)에 나오는 행렬과 알고리즘 상에서 계산되는 행렬의 계산에 이 모델이 사용된다. 계산 결과로 나온 좌표는 g에 의해 다시 Poincare모델로 표현된다.

3.2 하이퍼볼릭 패턴과 관련된 대칭군(Symmetry Group)

하이퍼볼릭 평면 위에서 패턴의 대칭은 변환을 통해 자기 자신의 일치되게 하는 경우를 말한다. Poincare 모델에서의 대칭은 원호에 대한 반사의 조합으로 이루어진다. 패턴의 대칭군은 그 패턴의 모든 대칭의 집합을 말한다.

하이퍼볼릭 평면상에서 정분할(regular tessellation) {p, q}는 정p각형이 꼭지점에서 q개가 만나는 형태며 다음과 같은 조건을 갖는다

$$(p-2)(q-2)>4, \text{ where } p,q\in N \quad (1)$$

{6, 4}분할의 경우를 그림 6의 (a)에 보였다. 모든 6각형은 같은 하이퍼볼릭 영역(hyperbolic area)을 가지며 같은 하이퍼볼릭 거리(hyperbolic distance)를 가지게 된다. 이는 단위원의 바깥쪽으로 갈수록 길이가 줄어드는 Euclidean 거리로 표현된다.

하이퍼볼릭 군(group)의 생성자(generator)를 구성하는 변환에는 3가지가 있다. 그림 6의 (a)에서 각각 A, B, C에 해당하며 다음과 같이 표현된다.

$$A=\begin{bmatrix}-\cosh(2b) & 0 & \sinh(2b)\\ 0 & 1 & 0\\ -\sinh(2b) & 0 & \cosh(2b)\end{bmatrix},\ b=\cosh^{-1}\left(\frac{\cos\left(\frac{\pi}{q}\right)}{\sin\left(\frac{\pi}{p}\right)}\right) \quad (2)$$

$$B=\begin{bmatrix}1 & 0 & 0\\ 0 & -1 & 0\\ 0 & 0 & 1\end{bmatrix} \quad (3)$$

$$c=\begin{bmatrix}\cos\left(\frac{2\pi}{p}\right) & \sin\left(\frac{2\pi}{p}\right) & 0\\ \sin\left(\frac{2\pi}{p}\right) & -\cos\left(\frac{2\pi}{p}\right) & 0\\ 0 & 0 & 1\end{bmatrix} \quad (4)$$

$$D=C\cdot B,\ E=A\cdot C,\ F=B\cdot A \quad (5)$$

A, B, C는 각각 하이퍼볼릭 평면에서 반사(reflection)를 나타내며 그림 6의 (a)에서 나타낸 바와 같이 A는 중심 p각형의 에지를 기준으로 반대편으로 반사시키며 B는 x축을 기준으로 반대편으로 반사를, C는 단위원의 중심과 중심 다각형의 꼭지점을 지나는 사선을 기준으로 반대편으로 반사시키는 변환이다. 이들 변환을 조합해서 만들어지는 D, E, F는 각각 회전변환을 나타내며 D는 단위원을 중심으로 $\frac{2\pi}{p}$ 회전을 나타내며 E는 중심다각형의 꼭지점을 중심으로 $\frac{2\pi}{q}$ 회전을, F

는 +x축과 중심다각형의 교점을 중심으로 π만큼 회전을 나타내며 모든 회전은 반시계 방향으로 이루어진다.

하이퍼볼릭 군의 종류는 [p, q], [p$^+$, q], [p, q]$^+$ 등으로 나누며 p, q는 (1)을 만족한다. [p, q]의 경우 생성자는 다음과 같은 조건을 만족한다.

$$D^p=E^q=F^2=I \quad (6)$$

그림 2에서는 (1)을 만족하는 각각의 p, q에 대해 정분할 한 예를 보였다.

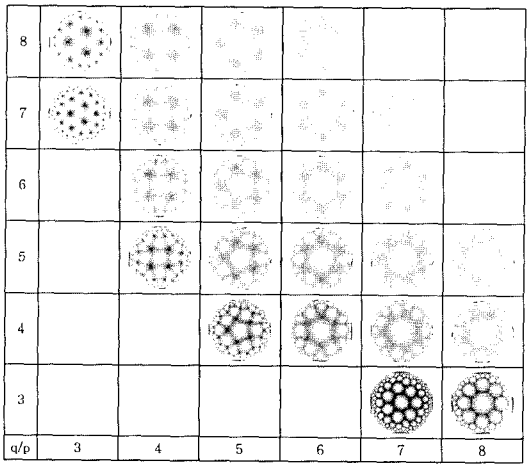


그림 2 p와 q값에 따른 {p, q} 분할의 관계

3.3 하이퍼볼릭 패턴 생성을 위한 L-시스템

하이퍼볼릭 패턴의 구조는 프랙탈 구조와 같이 무한히 반복되며 생성되는 성질이 있다. '자기 유사성(self-similarity)'의 형태나 나무의 구조와 같이 무한히 많아지는 패턴의 구조는 프랙탈의 생성방법을 이용할 수 있게 한다. 이에 착안해 본 논문에서는 하이퍼볼릭 타일링에서 패턴의 생성규칙을 L-시스템을 이용해서 기술하고자 한다[12]. Poincare 하이퍼볼릭 모델에서 정분할 [p, q]를 생성하는 생성규칙(production rule)은 다음과 같다.

$$\omega \rightarrow \Omega\alpha, \quad T \leftarrow I$$
$$\alpha \rightarrow ([F\Omega\beta][E(E\Omega\gamma)^{q-3}]D)^p$$
$$\beta \rightarrow D(D[F\Omega\beta][E(E\Omega\gamma)^{q-3}])^{p-4}D[F\Omega\beta][E(E\Omega\gamma)^{q-4}]$$
$$\gamma \rightarrow D(D[F\Omega\beta][E(E\Omega\gamma)^{q-3}])^{p-3}D[F\Omega\beta][E(E\Omega\gamma)^{q-4}] \quad (7)$$

Ω : *pattern drawing procedure*

T : *current transformation matrix*

I : *identity matrix*

여기서 D, E, F는 (5)에 나타낸 바와 같이 각각 하이퍼볼릭 변환 행렬이며 현재 변환행렬 T에 곱해진다. [과]은 변환행렬에 대한 스택 연산자이다. [은 현재 변환 행렬을 스택에 넣으며]은 현재 행렬 T를 스택에서 꺼내온 행렬로 설정한다. Ω는 각각의 p-각형 내부에 해당하는 패턴을 그리는 연산자다. α에 대한 생성규칙은 최초의 중심 패턴으로부터 주변을 둘러싼 패턴들을 생성하는 규칙이다. β와 γ는 각각 다각형의 선분에 인접하는 패턴과 꼭지점에 인접하는 패턴을 생성하는 규칙이다. 괄호와 함께 사용된 위 첨자는 반복을 나타낸다. 생성규칙의 유도 길이가 증가함에 따라 생성되는 패턴이 단위 원의 테두리에 인접해 간다. 이때 패턴의 복사는 일반적으로 벡터에 기반을 둔 방법으로 그리게 되는데 다음 절에 설명하는 문제점들을 가지고 있다.

3.4 포워드 매핑(forward mapping) 알고리즘의 문제점

포워드 매핑 알고리즘을 사용하는 경우 벡터 방식을 사용해서 표현을 하게 되고 이로 인해 다음과 같은 문제점들이 발생한다. 첫째, 메모리의 사용이 레벨의 증가에 따라 지수승에 비례하여 증가한다. 레이어가 하나 증가할 때마다 복사되는 패턴의 수가 급수적으로 증가하므로 메모리 사용도 많아지고 따라서 그리려는 크기에 관계없이 제한된 수의 레이어만 가지고 근사(approximation)해서 표현해야 한다. 따라서 단위 원에 근접하는 테두리 부분을 완성시키기 어렵다. 특히 패턴전체에 대해 이동변환을 하는 경우 패턴이 한쪽으로 몰려 완성되지 않고 남는 부분이 커질 경우 더 큰 문제점이 발생한다. 그림 11에서 보면 세번째 레이어까지 차지하는 부분이 이동변환에 따라 급격히 감소함을 볼 수 있다. 이렇게 패턴이 채워지 못하고 비워진 채로 남는 공간이 발생한다는 문제점이 있다. 셋째, 선으로 표현하는 경우에 바깥으로 갈수록 어두워진다. 선의 굵기가 비례해서 가늘어져야 하는데 이를 조절하기 위해서는 선의 굵기를 위치에 따라 파라미터화 해서 각기 다르게 지정해야 하며 이를 래스터화 할 수 있는 후처리가 추가적으로 필요하다.

이 밖에도 복잡한 패턴을 만들기 위해서는 패턴 당 메모리의 요구량이 많아진다는 점도 벡터 표현을 하이퍼볼릭 타일링에 사용할 때 발생하는 문제점이다. 이를 해결하기 위해서 각 픽셀 단위의 처리가 가능한 이미지 기반의 타일링 알고리즘이 필요하다.

4. 백워드 매핑(backward mapping) 알고리즘

하이퍼볼릭 패턴을 하이퍼볼릭 평면에 채우는 기존의 방법에서는 각각의 다각형에 대한 변환행렬이 정해지고 이에 의해 패턴의 복사가 이루어 진다. 이러한 방법은

앞서 언급한 바와 같이 벡터 방식에 대해서만 적합하고 비트맵방식으로 표현하고자 할 경우에는 생성된 이미지의 임의의 위치에 대해서 그 화소(pixel)가 속하게 되는 다각형의 변환행렬을 구해야 한다. 이에 대한 분석적인 방법이 알려져 있지 않으므로 모든 하이퍼볼릭 평면을 검색하지 않고 해당 변환을 구할 수 있는 효율적인 방법이 요구된다. [p, q]군 뿐만 아니라 [p, q]*군이나 [p*, q]군의 패턴 생성에도 적합하면서 최소한의 비교를 통해 화소가 속하게 되는 다각형으로 변환하는 행렬을 얻을 수 있어야 한다. 본 논문에서는 이러한 변환행렬을 효율적으로 구하는 알고리즘을 제시하고자 한다. 이러한 방법의 경우 쉐이딩 언어와 같은 쉐이더 제작에 있어 절차적인 방법을 통한 텍스쳐의 생성을 가능하게 한다.

4.1 레이어와 에지

[1]에 설명되어 있는 바와 같이 원으로 표현되는 하이퍼볼릭 평면의 중심으로부터 외각으로 레이어가 증가한다. 이는 식 (7)에서 유도 길이가 1씩 증가함에 따라 증가되는 하이퍼볼릭 다각형의 집합과 같다. 따라서 가장 중심에 위치하는 하이퍼볼릭 다각형이 0번째 레이어에 해당하고 0번째 레이어의 꼭지점에 이웃하는 모든 하이퍼볼릭 다각형들이 1번째 레이어가 된다(그림 3).

하이퍼볼릭 다각형의 에지는 원호의 형태로 나타나는데 [11]에서 보인 바와 같이 에지를 포함하는 원의 중심과 반지름을 구할 수 있다. 본 논문에서는 이렇게 구한 원을 광선추적법의 바운딩 볼륨과 같이 이용하여 원하는 점이 속하는 하이퍼볼릭 다각형의 위치를 찾는 속도를 효과적으로 줄일 수 있는 방법을 제시한다.

4.2 알고리즘

하이퍼볼릭 평면을 나타내는 단위원 안의 임의의 점에 대해 이점을 포함하는 하이퍼볼릭 다각형을 구하고 이 다각형으로 변환하는 행렬을 구하면 임의의 하이퍼볼릭 대칭패턴을 생성하는데 이용할 수 있다. 본 논문에서 제안하는 알고리즘은 다음과 같이 정리된다.

(1) 현재 레이어에 하이퍼볼릭 다각형과 하부 레이어와 인접하는 에지를 이용해서 점을 포함하는 원을 구한 후 이 원에 포함관계를 이용해서 검색할 하이퍼볼릭 다각형을 줄인다. 어느 원에도 포함되지 않는 경우는 현재 하이퍼볼릭 다각형의 안에 점이 위치하는 경우이므로 검색을 마친다.

(2) 줄여진 범위 내에서 같은 레이어 안에 하이퍼볼릭 다각형이 여러 개 존재하면 공유하는 에지를 이용해서 하나의 하이퍼볼릭 다각형과 그 하부 레이어로 검색범위를 줄인다.

(3) (1)과 (2)의 과정을 임의의 점을 포함하는 하이퍼볼릭 다각형이 하나로 정해질 때까지 반복한다.

그림 4에서는 위의 적용 예를 보이고 있다. 그림 4에서 (a)와 (c), (e)가 각각 (1)에 해당하며 (b)와 (d)가 (2)에 해당한다. 그림 4의 (a)에서 0번째 레이어는 하나의 하이퍼볼릭 다각형만 있으므로 α_0가 현재 하이퍼볼릭 다각형에 해당한다. α_0의 에지를 포함하는 원을 4개(circle 1~circle 4)를 그릴 수 있고 점 P를 포함하는 원이 하나이므로 그 부분에 해당하는 하이퍼볼릭 다각형들과 하위 레이어에 포함된 하이퍼볼릭 다각형들로 범위가 줄어든다. 그림에서는 γ_0와 γ_2, β_0의 세 개의 하이퍼볼릭 다각형과 이들 하부 레이어의 다각형들이 범위에 들어간다. 점 Q의 경우는 두 개의 원(circle 1, circle 2)에 동시에 포함되므로 γ_1과 그 하위 하이퍼볼릭 다각형으로 범위가 줄어든다. 따라서 같은 레이어 내에서 줄이는 단

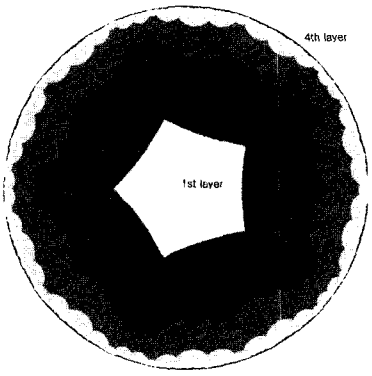


그림 3 레이어의 예 ([5, 4]분할)

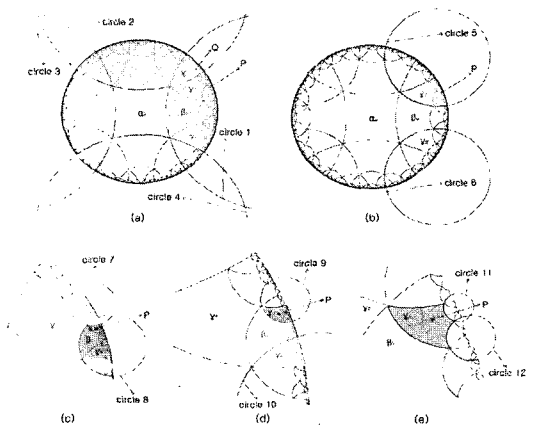


그림 4 점 P를 포함하는 하이퍼볼릭 다각형의 위치를 결정하는 예 ([4, 6]분할)

계 없이 바로 다음 단계로 넘어갈 수 있게 된다.

그림 4의 (b)에서 1번째 레이어에서 γ_0와 γ_2, β_2로 3개의 다각형으로 범위가 축소 되었고 이들 중 하나로 범위를 줄이기 위해 서로 경계를 이루는 부분의 에지를 이용해서 포함하는 원을 그린다(circle 5~circle 6). 점 P를 포함하는 관계에 의해 γ_0로 범위를 축소 시킬 수 있게 된다. 그림 4의 (c)에서는 (a)에서 처리한 방법과 마찬가지로 γ_0의 하위 레이어에서 점 P를 포함하는 원을 이용해 γ_3와 γ_4, β_1와 이들 하부 레이어의 다각형들로 범위를 축소한다. 그림 4의 (d)에서는 (b)에서 처리한 방법과 마찬가지로 2번째 레이어에서 하나의 다각형과 그 하부 레이어들로 범위를 축소 시킨다. 그림 4의 (e)에서는 (a)와 (c)에서 적용한 방법과 마찬가지로 하부 에지에 대한 점 P의 포함관계를 검사해서 어느 곳에도 포함되지 않는 경우 현재 하이퍼볼릭 다각형에 포함되는 경우로 판단하고 종료된다.

그림 7에 보인 바와 같이 q가 짝수인 경우엔 포함관계를 계산하기 위한 원과 하부 레이어를 포함한 에지가 일치하지만 q가 홀수인 경우는 에지와 정확하게 맞지 않는다. 그러나 (1)단계의 계산은 동일하고 (2)단계에서의 계산량이 레이어 내의 가장 바깥쪽 범위를 계산하기 위해 약간 증가할 뿐이고 기본적으로 동일한 방법으로 처리가 가능하다. 추가적으로 그림 4의 (a)에 보인 점 Q의 경우와 같이 두 개의 원에 동시에 포함되는 경우가 q가 짝수인 경우에 발생하며 이 경우 (2)의 단계를 건너뛰고 다음 반복으로 넘어갈 수 있다.

그림 5에 전체 알고리즘에 대한 세부 의사코드를 나타내었다. 그림 5의 (1)은 중앙에 위치하는 하이퍼볼릭 다각형의 꼭지점 위치를 계산하는 부분으로 [11]의 내용을 이용한 것이다([11]에 나오는 (3)식은 잘못되어 있어 이를 본 논문에서와 같이 r값을 계산해야 올바른 결과를 얻을 수 있다). 그림 5의 (2)는 하이퍼볼릭 평면 내의 두 점 **a**, **b**로부터 원의 중심을 구하는 식이다. 그림 5의 (3)은 두 개의 원에 동시에 포함되는 경우를 처리한다. 그림 4의 (a)에서 Q점과 같은 경우를 처리한다. 그림 4의 (4), (5), (6)은 각각 그림 8의 가운데, 오른쪽, 왼쪽을 처리하는 부분으로 같은 레이어에 여러 개의 하이퍼볼릭 다각형이 포함된 경우 하나로 줄이기 위한 처리부분이다.

4.3 변환행렬의 계산

(6)에서 표현한 바와 같이 각각의 다각형의 위치로 패턴을 복사하기 위한 변환 행렬이 있고 이 위치로의 역변환을 구해야만 역방향 매핑 알고리즘에 이용할 수 있다. 식 (7)에서 임의의 p, q에 대해서 각각의 생성규칙으로부터 얻어지는 패턴의 개수를 보면 생성규칙 ω에 의해서는 항상 1개의 패턴이 생성되고 α에 의해서는 (q-2)*p개의 패턴이, β에 의해서는 (q-2)*(p-3)-1개, γ에 의해서는 (q-2)*(p-2)-1개가 생성된다. 일반적으로 다각형의 에지를 이루는 원호를 보면 원호에 의해 포함되는 다각형이 $2*\left[\frac{q-3}{2}\right]+1$개임을 알 수 있다. 이는 q가 홀수와 짝수인 경우 모두에 해당되며 q가 짝수일 때 이웃하는 에지에 의해 동시에 포함되는 경우를 제외한 것이다. 동시에 포함되는 경우를 따로 처리하는 이유는 원호에 의해 포함되는 관계로만 쉽게 검사회수를 줄일 수 있기 때문이다. 일반적인 p, q에 대해 하위 레이어에 대한 변환을 계산하는 방법을 그림 8에 보였다. T행렬이 순방향으로의 변환을 M이 역방향으로의 변환을 나타낸다.

4.4 [p, q]$^+$군과 [p$^+$, q]군에 대한 처리

Dunham은 [1]에서 p>3이고 q>3인 경우의 [p, q]군과 [p, q]$^+$군의 패턴 생성을 위한 알고리즘만을 설명하고 있다. 이 밖의 p=3이거나 q=3인 경우의 [p, q]군과 [p, q]$^+$군 생성을 위한 알고리즘과 [p$^+$, q]군의 패턴 생성을 위한 알고리즘에 대해서는 좀 더 복잡해진다고 만 언급하고 있을 뿐 자세한 설명을 하고 있지 않다. 이에 대해 본 논문에서 정리해 보면 다음과 같다. 그림 6의 (c)에서 보인 형태의 변환은 기본적으로 [p, q]$^+$군을 표현하는데 사용되며 이는 [p, q]군을 표현하는데도 동일하게 쓰일 수 있다. [1]에서도 [p, q]군과 [p, q]$^+$군에 대해 동일한 알고리즘을 사용한다. [p$^+$, q]군을 표현하기 위해서는 에지에서 인접하는 다각형과 반사 대칭(reflective symmetry)을 이뤄야 하고 그림 6의 (e)와 같이 행렬 B가 부분적으로 이용된다. 이러한 [p$^+$, q]군 생성을 위한 알고리즘은 [p, q]에 대해서도 동일하게 사용될 수 있다. 본 논문에서 제시한 알고리즘으로 각각의 군에 대한 생성 예를 (b), (d), (f)에 각각 보였다. 기본적으로 식 7에 제시한 생성방법을 기반으로 한 기본 알고리즘으로 (b)와 (d)를 생성할 수 있으며(그림 6의 (c)) 이를 그림 6의 (e)에 보인 바와 같이 기본 알고리즘을 수정하여 (b)와 (f)를 생성할 수 있다.

4.5 결과

제안한 알고리즘을 가지고 타일링에 적용한 결과를 그림 9와 그림 10에 보였다. 그림 9는 Escher의 자화상 작품을 이용하여 하이퍼볼릭 타일링을 한 결과다. 기존의 포워드 매핑 알고리즘으로 하기 힘든 일반적인 이미지를 사용하여 타일링이 가능함을 보여주는 결과로 특히 바깥 테두리 부분을 항상 깨끗하게 완성할 수 있다

$\mathbf{v}_i \leftarrow [d\cos(\frac{2\pi i+\pi}{p}), d\sin(\frac{2\pi i+\pi}{p})], where\ i \in Z, 0 \le i < p, \quad //\ \mathbf{v}_i\ is\ polygon\ vertices$

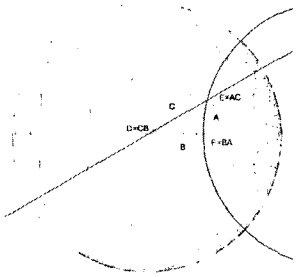
$$d = \frac{e^r - 1}{e^r + 1}, and\ r = \cosh^{-1}\frac{\cos\frac{\pi}{p}\cos\frac{\pi}{q}}{\sin\frac{\pi}{p}\sin\frac{\pi}{q}} \quad \cdots\cdots (1)$$

$\mathbf{p} \leftarrow [x, y]$ // $\mathbf{p}$ *is evaluation point*
$T \leftarrow I$ // T *is transformation matrix*
$M \leftarrow I$ // M *is inverse of* T
$S \leftarrow I$ // S *is temporal*
do {
$\mathbf{v}'_i \leftarrow T \cdot \mathbf{v}_i, where\ i \in Z, 0 \le i < p.$

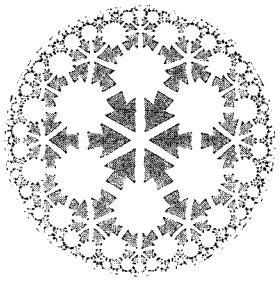
$$\mathbf{o}_i \leftarrow [\frac{a_y(1+\mathbf{b}\cdot\mathbf{b}) - b_y(1+\mathbf{a}\cdot\mathbf{a})}{2(a_y b_x - a_x b_y)}, \frac{b_x(1+\mathbf{a}\cdot\mathbf{a}) - a_x(1+\mathbf{b}\cdot\mathbf{b})}{2(a_y b_x - a_x b_y)}], \quad \cdots\cdots (2)$$

$where\ i \in Z, 0 \le i < p, and\ \mathbf{b} = \mathbf{v}'_i, \mathbf{a} = \mathbf{v}'_i$
$flag_i \leftarrow |\mathbf{p} - \mathbf{o}_i| < |\mathbf{a} - \mathbf{o}_i|$
for (*i*) {
if ($flag_i$) {
if ((*q*%2 == 0) && $flag_{(i+1)\%p}$) { ····· (3)
$T' \leftarrow T \cdot D^i \cdot E^{\frac{q}{2}}$
$M' \leftarrow E^{\frac{q}{2}} \cdot D^{(p-i)\%p} \cdot M$
continue do loop
}
$S \leftarrow T \cdot D^i \cdot F$
$\mathbf{v}''_1 \leftarrow S \cdot \mathbf{v}_1$
$r_1 \leftarrow$ *inside test* $\mathbf{p}$ *about* $\mathbf{v}'_{(i+p-1)\%p}$ *and* $\mathbf{v}''_1$.
$r_1 \rightarrow$ *inside* :
$\mathbf{v}''_{p-2} \leftarrow S \cdot \mathbf{v}_{p-2}$
$r_2 \leftarrow$ *inside test* $\mathbf{p}$ *about* $\mathbf{v}'_i$ *and* $\mathbf{v}''_{p-2}$.
$r_2 \rightarrow$ *inside* : ····· (4)
$T' \leftarrow S$
$M' \leftarrow F \cdot D^{(p-i)\%p} \cdot M$
$r_2 \rightarrow$ *outside* :
for $(0 \le j < \frac{q-3}{2})$ { ····· (5)
$S \leftarrow T \cdot D^i \cdot E^{j+2}$
$\mathbf{v}''_{p-1} \leftarrow S \cdot \mathbf{v}_{p-1}$
$r_3 \leftarrow$ *inside test* $\mathbf{p}$ *about* $\mathbf{v}'_i$ *and* $\mathbf{v}''_{p-1}$.
$r_3 \rightarrow$ *inside* :
$T' \leftarrow S$
$M' \leftarrow E^{q-j-2} \cdot D^{(p-i)\%p} \cdot M$
}
$r_1 \rightarrow$ *outside* :
for $(0 \le j < \frac{q-3}{2})$ { ····· (6)
$S \leftarrow T \cdot D^{(i+p-1)\%p} \cdot E^{q-j-2}$
$\mathbf{v}''_1 \leftarrow S \cdot \mathbf{v}_1$
$r_4 \leftarrow$ *inside test* $\mathbf{p}$ *about* $\mathbf{v}'_{(i+p-1)\%p}$ *and* $\mathbf{v}''_1$.
$r_4 \rightarrow$ *inside* :
$T' \leftarrow S$
$M' \leftarrow E^{j+2} \cdot D^{(p+1-i)\%p} \cdot M$
}
}
}
}

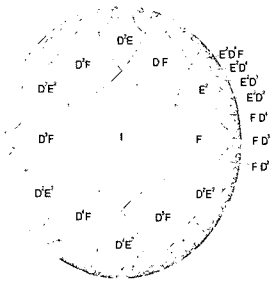
그림 5 알고리즘에 대한 의사코드(pseudo code)



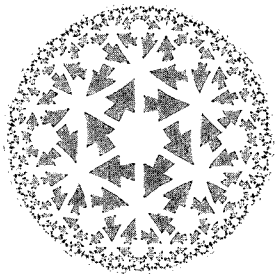
(a)대칭군의 생성자를 구성하는 기본 변환들



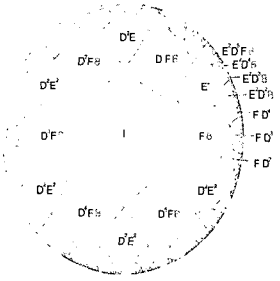
(b) 화살표 패턴을 이용한 [6,4]의 예



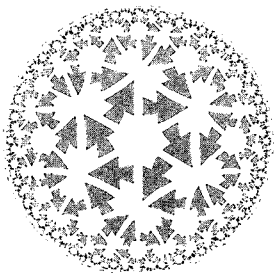
(c) $[6,4]^+$패턴을 위한 변환행렬



(d) 화살표 패턴을 이용한 $[6,4]^+$의 예

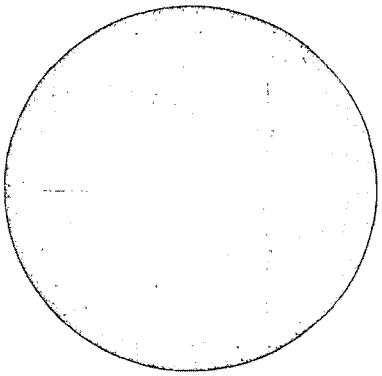


(e) $[6^+,4]$패턴을 위한 변환 행렬

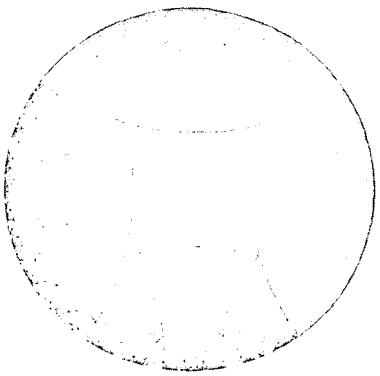


(f) 화살표 패턴을 이용한 $[6^+,4]$의 예

그림 6 [6, 4]분할과 패턴에 생성 예



(a) q가 홀수인 경우([5, 5] 타일링)



(b) q가 짝수인 경우([4, 6] 타일링)

그림 7 다각형의 한 변에 대해 포함되는 영역

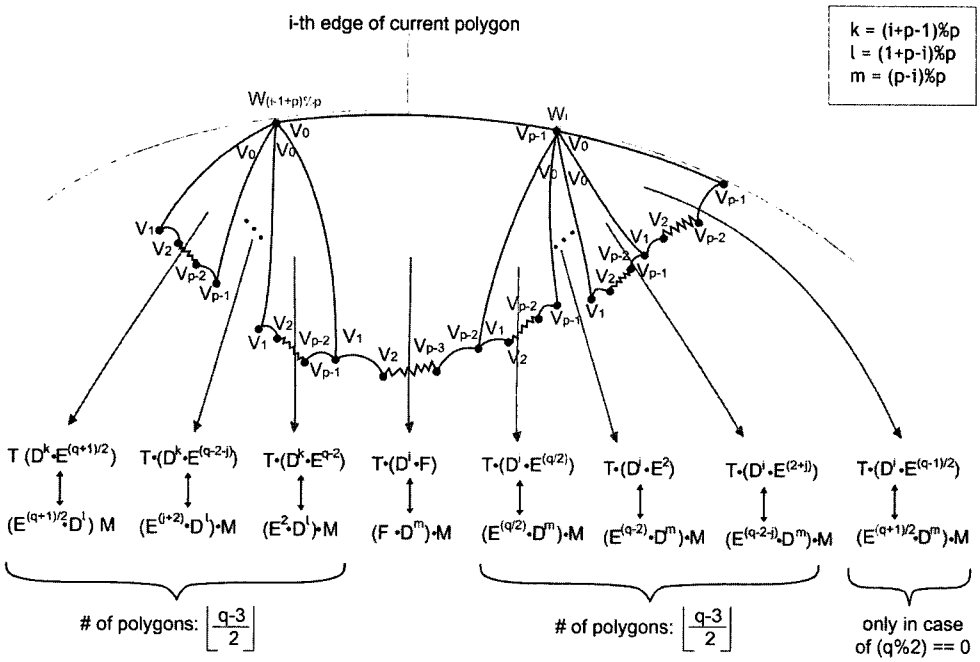


그림 8 하위 레이어의 변환 행렬의 변환관계

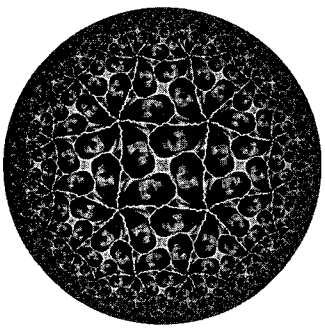


그림 9 Escher의 자화상 작품(Self-Portrait, November 1929, Lithograph, 264mm×203mm)을 이용해서 만든 $[4, 5]^+$ 타일링의 예

는 점은 제안한 알고리즘의 우수성을 보여주는 좋은 예다. 메모리의 사용양도 기본 패턴에 대한 사용량이 고정적이므로 레이어 수와 무관하다.

그림 10은 절차적인 쉐이딩 프로시져에 의해 패턴을 생성한 예다. 동심원 형태의 텍스쳐를 이미지를 사용하지 않고 코딩의 형태로 생성하였다. 이러한 예는 제안한 알고리즘이 쉐이딩 언어를 사용하는 렌더링 방식에 적

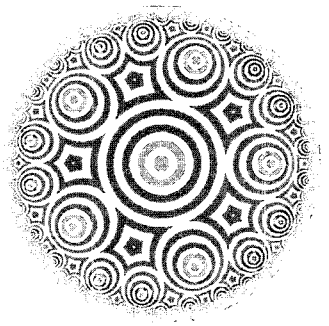


그림 10 쉐이더를 이용한 텍스쳐 생성의 예 ([5, 5]분할).

합함을 보여준다[13,14].

그림 11에서는 하이퍼볼릭 패턴을 하이퍼볼릭 평면 안에서 이동하여 그리는 경우를 적용한 예다. 오른쪽에 결과 이미지를 왼쪽에 이에 해당하는 레이어 형태를 나타냈다. 이동변환이 없는 경우 (a)의 그림처럼 중앙에 대칭인 형태로 패턴이 생성되지만 (b)와 (c)에 그림처럼 패턴 전체의 이동이 있는 경우에도 모든 부분을 동일하게 완성시켜준다. +모양의 위치가 패턴이 생성되는 맨 처음 레이어의 중심을 나타낸다.

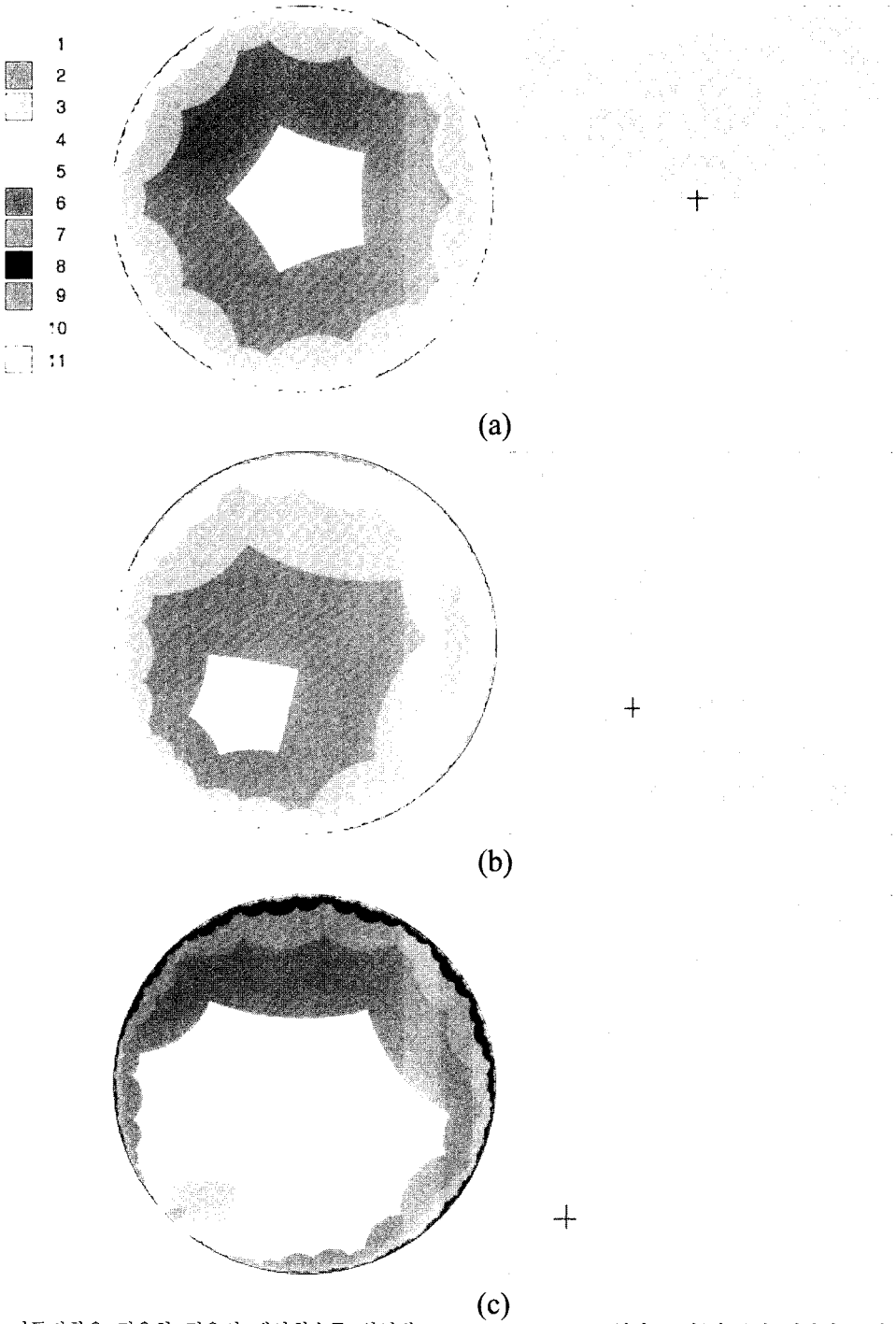


그림 11 이동변환을 적용한 경우의 계산횟수를 위치별로 pseudo color로 표현한 그림(왼쪽)과 해당하는 하이퍼볼릭 패턴(오른쪽). (패턴 그림에서 +표시로 나타낸 부분이 중심이 이동된 위치를 나타낸다) [5, 4]분할.

5. 결 론

본 논문에서는 기존의 하이퍼볼릭 패턴 생성 알고리즘이 벡터 데이터에만 적합한 한계를 극복하기 위해 이미지 데이터에 적합한 새로운 알고리즘을 제시했다. 결과에서 보인 바와 같이 기존의 포워드 매핑 방법으로 하기 힘들었던 부분을 해결할 수 있게 되었다. 이러한 알고리즘은 하이퍼볼릭 공간에서의 자동 타일링 방법에 대한 연구의 기여가 될 것이며 좀 더 다양한 하이퍼볼릭 패턴생성 알고리즘의 개발에 이용될 수 있을 것이다. 또한, 최근 Euclidean 평면에서 타일링의 자동화에 관한 연구[15]와 관련하여 하이퍼볼릭 평면에서의 자동화된 타일링의 연구에 기여할 수 있을 것이다.

감사의 글

본 논문은 교육부 두뇌한국21(BK21)사업과 정보통신부 국가지정연구실 사업(No.200-N-NL-01-C285)의 지원을 받아 수행되었습니다.

참 고 문 헌

[1] Douglas Dunham, John Lindgren, and David Witte, "Creating Repeating Hyperbolic Patterns," SIGGRAPH 1981 Proceedings, pp. 215-223.

[2] Douglas Dunham, "Hyperbolic Symmetry," Computers & Mathematics with Applications, (1/2), 12B, pp. 139-153.

[3] Douglas Dunham, "Creating Hyperbolic Escher Patterns," In: M. C. Escher: Art and Science, H. S. M. Coxeter et al., eds., Amsterdam: North-Holland, pp. 241-248.

[4] Douglas Dunham, "Transformation of Hyperbolic Escher Patterns," Visual Mathematics, Volume 1, No. 1, 1999.

[5] Douglas Dunham, "Artistic Patterns in Hyperbolic Geometry," In Bridges: Mathematical Connections in Art, Music, and Science; Conference Proceedings, Edited by Reza Sarhangi, pp. 239-249, 1999.

[6] Douglas Dunham, "Hyperbolic Celtic Knot Patterns," In Bridges: Mathematical Connections in Art, Music, and Science; Conference Proceedings, Edited by Reza Sarhangi, pp. 13-22, 2000.

[7] Douglas Dunham, "Hyperbolic Islamic Patterns -- A Beginning," In Bridges: Mathematical Connections in Art, Music, and Science; Conference Proceedings, Edited by Reza Sarhangi and Slavik Jablan, pp. 247-254, 2001.

[8] Bruce M. Adcock, Kevin C. Jones, Clifford A. Reiter, and Lisa M. Vislocky, "Iterated Function Systems with Symmetry in the Hyperbolic Plane," Computer & Graphics, Volume 24, Issue 5, Pages 791-796, October 2000.

[9] K. W. Chung, H. S. Y. Chan, and B. N. Wang, "Hyperbolic Symmetries from Dynamics," Computers & Mathematics with Applications, Vol. 31, No. 2, pp. 33-47, 1996.

[10] K. W. Chung, H. S. Y. Chan, and B. N. Wang, "Efficient Generation of Hyperbolic Symmetries from Dynamics," Chaos, Solitons & Fractals, Volume 13, Issue 6, Pages 1175-1190, May 2002.

[11] John C. Hart, "On the Hyperbolic Plane and Chinese Checkers," Proceedings of the Western Computer Graphics Symposium, pp. 69-72, Mar. 1996.

[12] Przemyslaw Prusinkiewicz, and Aristid Lindenmayer, The Algorithmic Beauty of Plants, Springer-Verlag, 1996.

[13] Darwyn Peachey, "Texture Generation," In Writing RenderMan Shaders (SIGGRAPH 1992 Course #21 Course Notes), pp. 32-56, July 1992.

[14] David S. Ebert, F. Kenton Musgrave, Darwyn Peachey, Ken Perlin, and Steven Worley, Texturing and Modeling: A Procedural Approach, 2nd ed., Academic Press, San Diego, CA., 1998.

[15] Craig S. Kaplan, and David H. Salesin, "Escherization," SIGGRAPH 2000 Proceedings, pp. 499-510.

조 청 운

1992년 중앙대학교 공과대학 전자계산학과 졸업. 1994년 중앙대학교 대학원 컴퓨터공학과 석사. 2000년~현재 중앙대학교 첨단영상대학원 영상공학과 연구교수. 관심분야는 컴퓨터 그래픽스