3차원 기하 모델링에서 곡면간의 교차곡선 추적 알고리즘

박 철 호[†] · 홍 성 수[††] · 심 재 홍[†††]

요 약

곡면간의 교차계산은 부울연산(Boolean operations)과 조각된 곡면들을 지원하기위한 기하 모델링과 솔리드에서 사용되는 기본적인 기하학 연산이다. 본 논문에서는 두 정규화된 곡면의 교차곡선을 따라 추적하기 위한 새로운 알고리즘을 제안한다. 그러므로 본 논문에서는 계산상의 간소화와 2차 연속성을 나타낸다. 따라서, 교차곡선의 한점이 주어지면 이 점을 초기점으로 하여 교차 곡선의 전체 곡선을 추적 한다. 그리고 각각의 교선들의 초기점들은 쿼드트리에서 DFS(Depth First Search)기법으로 검색되고 교선은 연속적인 형태로 자연스럽게 표현된다.

Curve Tracing Algorithm for Surface/Surface Intersection Curves in 3D Geometric Modeling

Chul Ho Park[†] · Sung Soo Hong[††] · Je Hong Sim[†††]

ABSTRACT

SSI(Surface/Surface Intersection)is a fundamental geometric operation which is used in solid and geometric modeling to support trimmed surface and Boolean operations. In this paper, we suggest a new algorithm for tracing along the intersection curve of two regular surfaces. Thus, in this paper, we present a simplicity of computing and second degree continunity. Given a point of intersection curve, it is traced to entire curve of a intersection curve as the initial point of its and the initial point of each of a intersection curve is detected to DFS (Depth First Search) method in the Quadtree and is naturally presented a continuous form.

1. 서 론

기하 모델링은 컴퓨터 그래픽스, CAD /CAM, 컴퓨터 비젼, 로보틱스등의 여러 응용분야들에 걸쳐 2차원 및 3차원 물체들을 컴퓨터에 표현하고, 이를 이용하여 여러가지의 기하학적 계산들을 효율적으로 수행하기위한 도구로서 중요하게 사용되어 오고 있다.

기하 모델링에서 가장 기본이 되는 계산은 두 곡선이 만나서 생기는 교차 곡선을 계산하는 것이다[2]. 일단 곡선의 교점들과 곡면의 교차곡선들이 정확하게 계산된 후, 2차원 평면이나 3차원 공간상에서의 입체물체에 대한 교집합, 합집합, 차집합등을 계산하는 문제들은 비교적 간단해 진다. 지금까지 기하모델링에서 중요한 연구과제가 되었던 문제들은 주로 이

† 준 회 원:광운대학교 전자계산학과
†† 종신회원:호서대학교 전자계산기공학부
††† 정 회 원:광운대학교 전자계산학과
논문접수:1996년 4월 9일, 심사완료:1997년 7월 30일

러한 곡선의 교점 및 곡면의 교차곡선을 가능한 한 오차가 적게 계산하는 것이었다. 이를 위하여 기하학적인 문제에 대한 수치해석학적 접근 방법들이 주로 이용되어 왔으며, 이중에서도 특히 Newton근사방식이 많이 이용되었다[4]. 곡선간의 교선을 계산하는 문제를 해결하기위해서는 두 가지 방법이 있다. 첫번째 방법은 비선형 연립방정식을 수치 해석적 기법을 풀어서 교선을 구하는 방법이다. 이 방법의 약점은 반복 계산을 위한 초기값을 적절히 선정하기가 어렵기 때문에 수렴성이 보장되지 않는 큰 단점이 있다[11]. 또 다른 하나는 분할이론(subdivision theory)에 기초를 둔 기하학적인 방법으로서, 주어진 두 곡면을 반복적으로 분할 하여 여러개의 평면 사변형(planar quadrilateral)으로 근사한 후, 근사된 평면 사변형들 간에 생기는 모든 교차 선분들을 찾아내는 방법이다. Peng[14]은 이 분할법에 의해 두 개의 3차 B-Spline 곡면간에 생기는 교선들을 구하는 알고리즘을 발표했다. 이 알고리즘은 깊이 우선 탐색법(Depth First Search)을 사용해서 하나의 교차 선분만을 먼저 구한 후, 쿼드트리 자료 구조[7]속에 저장되어 있는 분할된 각 패치들 간의 인접관계(adjacency relation)를 이용하여 나머지 교차선분들을 연속적으로 찾아 나간다. 그런데 이 방법은 두 곡면간의 교선이 하나인 경우에는 빠르고 정확하게 교선을 찾을 수 있으나, 둘 이상의 교선을 찾을 경우는 복잡한 알고리즘으로 인하여 다음에 언급되는 너비 우선 탐색법(Breadth First Search)보다 능률적이지 못하다. Lasser[5]는 분할법에 의해 베지에르 곡면간의 교선을 계산하는 방법을 제시했다. 이 방법은 Peng[14]과는 달리 먼저 두 곡면의 교차에 참여하는 평면으로 근사된 패치들의 쌍을 모두 찾은 후, 이들로 부터 교선을 근사하는 선분들을 순서에 관계없이 찾는 너비우선 탐색법을 사용하고 있다. 이 방법의 단점은 구한 교점을 순서대로 하기 위해 분류(sorting)과정이 필요하다는 것이다. 또 Koparkar[13]도 너비우선 탐색법을 이용해서 일반적인 매개변수화 곡면(parametric surface)간의 교선을 계산하는 알고리즘을 연구했는데, Lasser[5]과는 달리 패치간의 인접관계를 이용함으로써 분류과정을 없앴으며, 구한 교차점들에 대한 오차를 줄이기 위해 수치해석적 방법을 개발했다. 이들 분할법에 의한 교선계산방법은 수치해석적 방법에 비해 계산 시간이 오래 걸리는 단점이 있으나, 교선을 구하는데 실패할 염려가 거의 없다는 점에서 최근에 많이 연구, 개발되고 있다. 본 논문은 부울연산(Boolean operation)를 지원하는 모델러의 기하학적 범위에서 고려된다. 따라서 교선과 정규화된 3차원 곡면간의 경계계산은 부울연산을 이용하고, 이 때 곡면과 곡면들간에 교차 곡선인 잠재적인 에지들(potential edges)을 찾고, 안쪽과 바깥쪽에 곡면의 경계인 곡선 세그먼트들을 계산하여 이들 곡선들을 계층화한다. 이들 곡선 세그먼트들은 곡면과 곡면들간의 잠재적인 에지들에 교차점들에 의하여 한정된다. 이와 같은 조건에서, PCC(Piecewise Circular Curve)는 평면 교차 모델링에서 특히 관심이 있는 부분인 부드러운 공간 곡선(smooth space curve)들을 설계하는데 용이하다. 따라서 곡면과 곡면 혹은 곡선과 곡선간의 교선을 추적하는데 효과적이다. 본 논문의 구성은 다음과 같다. 제2절에서는 PCC의 생성방법을 나타내고 제3절에서는 두 곡면간의 교선추적 검사와 교선추적 알고리즘을 설명한다. 제4절에서는 제3절에서 제안된 알고리즘을 이용한 분석을 설명한다. 마지막으로 제5절에서는 본 논문에 대한 결론으로 구성된다.

2. PCC(Piecewise Circular Curve)의 구성 및 특성

2.1 PCC의 구성

PCC의 C^1 연속성에 보장을 위하여 원호들의 쌍을 일반 접선방향에 강제로 공유시킨다. 원호가 삼각제어에 의하여 접선일 경우, 부드러운 환경에서 삼각제어를 연결함으로서 두 개의 원호를 부드럽게 연결한다. 그러므로 두 개의 원호의 제어다각형, 즉 4변형(quadrilateral)으로 나타난다. 제어다각형의 4개의 꼭지점들이 동일 선상에 있지않을 때, 두 개의 원호는 꼬이는 형태로 생성된다. 그러므로 이 형태는 부드러운 비평면 곡선(nonplanar curve)이 된다.

· 제약조건

A_1, B_1, C_1, A_2, B_2, C_2 6개 점들의 8개 좌표들은 다음과 같은 제약조건들을 만족하는 보간된 두 개의 제어다각형을 정의한다.

1. 다각형은 경계 조건 위치를 만족한다.

$A_1 = P_1$, $C_1 = P_2$

2. 다각형은 두 동일한 이등변삼각형으로 분리시킨다.

$\|B_1 - A_1\| = \|C_1 - B_1\|$,
$\|B_2 - A_2\| = \|C_2 - B_2\|$

3. 두 개의 제어 삼각형들은 두 개의 제어 삼각형들에 점들과 자연스럽게 연결한다.

$C_1 = A_2$, $C_1 - B_1 = k(B_2 - C_1)$
(k: 외부 변수)

4. 다각형은 삼각구조 경계 조건들을 만족한다.

$B_1 = P_1 + a_1 \cdot T_1$, $B_2 = P_2 - a_2 \cdot T_2$
/* T_1, T_2: 접선 벡터 */
/* a_1, a_2: 비율(ration)을 나타내는 매개변수 */
$\|B_2 - B_1\| = a_1 + a_2$

$\|(P_2 - a_2 T_2) - (P_1 + a_1 T_1)\| = a_1 + a_2$

· 제어 다각형의 계산

두 개의 원호에 남아 있는 자유도를 결정하는 매개변수로서 a_1를 계산하고, a_1에 의하여 a_2를 계산한다.

$$a_1 a_2 (T_1 \cdot T_2 - 1) + \frac{\|S\|^2}{2} = a_1 (ST_1) + a_2 (ST_2)$$

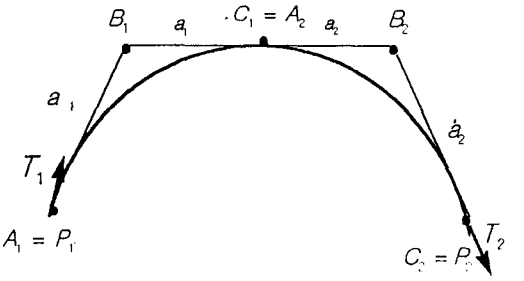
/* $S = P_1 - P_2$, $\|T_1\| = \|T_2\| = 1$ */

$$a_2 = \frac{a_1 ST_1 (\frac{\|S\|^2}{2})}{a_1 (T_1 \cdot T_2 - 1) - ST_2}$$

· 선택된 제어 다각형

앞에서 a_2를 계산하였고 a_1의 값과 접선 그리고 보간된 점들이 주어지므로서, 보간된 두 개의 원호에 제어 다각형을 계산하였다. 두 개의 원호에 모양은 a_1 값에 따른다.

정확한 곡선이 찾아졌을 때, a_1은 근사치를 계산하여 최소한의 오차를 선택한다. 만약 정확한 곡선이 못찾아졌을 때, a_1의 선택은 최소곡률을 요구함으로써 전체 원호 길이와 꼬인 혹은 두 개의 원호 결과에 다른 형태로 나타난다. 수치적인 방법을 사용함으로써, 최적 표준의 값을 조사한다.



(그림 2.1) PCC(Piecewise Circular Curve)의 생성
(Fig 2.1) The Generation of PCC

2.2 PCC의 특성

PCC는 또한 Bezier 곡선의 성질 중에 컨벡스 헐(convex hull)의 성질을 가지고 있다. 따라서 이 속성을 이용하여 효과적인 분할(subdivision)을 제안한다. 그리고 PCC는 Bezier 함수와 마찬가지로 벡터함수의 형태로 정의되므로, 매개변수 t에 의한 x, y의 방정식인 $t: \langle x, y \rangle = \langle f(t), g(t) \rangle$으로 표현된 매개변수 방법(parameterization)이 가능하다. 또한, PCC의 각 세그먼트는 또한 3차 달걀꼴 (cubic oval)이 가지는 볼록한 상태(convexity)와 단순성을 가진다. 그러므로 PCC안에서 연속된 점들의 유연성은 사용자에 의하여 조정된다. 또한 PCC는 구간적(interval)으로 구성함으로써 자료점의 갯수가 많거나 복잡한 곡선의 모양을 표현하는데도 역시 곡선의 차수는 3차이내로 제한할 수 있고, 변화율 감소(variation diminishing), 아핀 불변(affine invariance)등의 특징을 가진다. PCC의 변환과정은 Bezier 곡선에 변환과정과 유사하다[14]. 그러므로 본 논문에서는 계산상의 간소화를 위하여 다음과 같은 식을 이용한다.

$$Q_i = \frac{(i-1)P_{i-1} + (n-i+1)P_i}{n}$$

/* P_i, Q_i: 차수를 올리기 전과 올린 후의 조정점이다. */
/* n: 자료점의 개수 */

PCC 생성은 다음과 같다.

```
/* construct of PCCS */
check(x1, y1, x2, y2, x3, y3);
/* 제어점(control point) */
float x1, y1, x2, y2, x3, y3;
{
float len_ab, len_bc;
len_ab = sqrt((x2-x1)*(x2-x1)+(y2-y1)*(y2-y1));
len_bc = sqrt((x3-x2)*(x3-x2)+(y3-y2)*(y3-y2));
/* PCC를 제어하는 두 제어점들 간의
경계 조건 */
  if (len_ab==len_bc)
    return(TRUE);
  else
    return(FALSE);
    }
  seta1 =
(sqrt((x1-x2)*(x1-x2)+(y1-y2)*(y1-y2)))/R;
      seta2=atan(seta1);
    /* 두개의 원호가 되기 위한 조건 */
      seta2=seta2*180/M_PI;
        orx=(int)X_O;
        ory=(int)Y_O;
        rad=(int)R;
      s_a=trans_angle1(seta2);
      l_a=trans_angle2(seta2);
      a[0]=orx;
      a[1]=ory;
      a[2]=s_a;
      a[3]=l_a;
      a[4]=rad;
      return(*a);
```

3. 두 곡면간의 교선추적 알고리즘

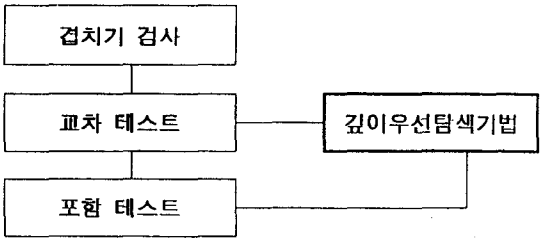
두 곡면간의 교선을 찾는 방법으로는 수치해석적인 방법과 기하학적 근사화를 이용한 방법이 있다. 본 연구에서는 후자의 방법으로 교선을 구한다. 이 방법을 곡면분할법이라 하는데 다음과 같은 종류가 있다.

3.1 관련된 교차추적 기법

1) 깊이 우선 탐색기법(Depth First Search Method) 교차할 가능성이 있는 하나의 패치쌍을 선정하여 깊이 우선으로 하나의 교차점을 구한뒤, 쿼드트리 자료구조에 저장된 각 패치의 인접관계를 이용하여 다른 교차선분을 계산하는 방법이다. 이 방법은 계산량은 적으나 그 알고리즘이 어렵다는 단점이 있다[6].

2) 너비우선탐색기법(Breadth First Search Method) 교차할 가능성이 있는 모든 패치의 쌍을 구한후, 그 인접관계와는 무관하게 교차점들을 구해 이를 순서적으로 연결(sorting)해 주는 방법이다. 이 방법은 비교적 쉬운 알고리즘으로 되어 있으나 중복적인 계산으로 계산량이 많으며 연결과정이 추가되는 단점이 있다[9].

따라서 본 논문에서는 곡면 분할법중 깊이 우선탐색기법을 사용하였다. 여기서 사용한 깊이우선 탐색기법에 일반적인 과정은 다음과 같다. 또한 곡면을 데이타 구조로 표현하기 위해 쿼드트리 구조를 사용하였다. 그러므로 쿼드트리를 이용하여 이웃찾기를 조사하고 인접한 교선들 전체를 탐색할 수 있다.



(그림 3.1) 교선 추적을 위한 검사
(Fig 3.1) The test for tracing the intersection line

3.2 제안한 교선추적 기법들

(1) 겹치기 검사 (Overlap test)

겹치기 검사(overlapping)는 서로 다른 곡면에서 임의로 선택된 한쌍의 패치가 두 곡면의 교차(intersection)에 참여할 가능성이 있는지 없는지를 판단하는 것이다.

3차원 직교좌표 공간상에서 조정점들의 좌표를 비교하여 각 성분의 최대, 최소를 이루는 점에 의해 직육면체의 상자는 곡면을 포함하게 된다. 본 연구에서는 위의 성질을 이용하여 곡면을 포함하는 상자를 만들고 만들어진 상자를 이용하여 곡면간의 겹치기 검사를 수행한다.

(2) 패치의 편평도 검사 (Flatness test)

패치의 편평도 검사는 패치를 평면으로 간주할 수 있는지의 여부를 판단하기 위해 사용된다. 그런데 본 논문에서 사용하는 곡면상의 패치는 컨벡스 헐의 성질을 만족하므로 패치에 대한 편평도 검사는 조정점들이 평면으로 근사될 수 있는가를 판단하는 것만으로도 충분하다. 먼저 패치위에 있는 조정점들 중에 임의의 3개를 선택하여 평면을 구성한다. 그리고 나머지 조정점들로부터 평면까지의 거리를 계산한다.

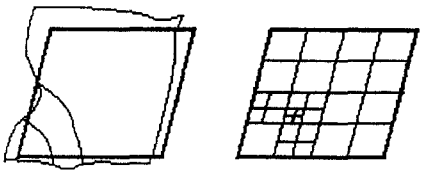
한점 Q에서 평면 $R = P + uA + v B$ $u, v \in [0, 1]$까지의 거리 d는 다음 식으로 계산된다.

$$d = (\vec{P} - \vec{Q}) \times \frac{\vec{A} \times \vec{B}}{|A \times B|}$$

이때 나머지 조정점들에 대해 구간 최대거리가 모두 주어진 공차안에 들면, 패치는 패치위의 가장 바깥쪽 4개의 조정점을 꼭지점으로 하는 평면 사변형(planar quadrilateral)로 근사된다. 이 때 어떤점 p에서 A와 B를 잇는 직선까지의 거리를 계산하는 식은 다음과 같다.

$$d = (\vec{P} - \vec{A}) \times \frac{\vec{B} - \vec{A}}{|\vec{B} - \vec{A}|}$$

/* d: 최단 거리(minimum distance) */



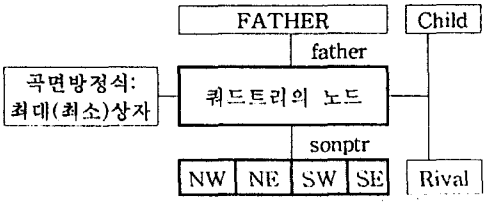
(그림 3.2) 곡면의 fitting과 쿼드트리 탐색
(Fig 3.2) Fitting of the Surface and Quadtree search

(3) 쿼드트리(quadtree)와 이웃 찾기(neighbor finding)

평면으로 곡면을 근사화하기 위해 반복적인 곡면의 분할을 수행하게 되는데 분할된 곡면을 저장하기 위한 구조로 쿼드트리 데이타 구조를 사용한다.

다음과 같은 쿼드트리 데이타 구조를 이용하면 인접한 곡면을 찾는 방법이 용이하여 본 연구에서는 매우 적합한 구조이다. 두 곡면간의 교선을 구하는 방법으로서 두가지 중요한 데이타 구조가 사용된다. 쿼드트리 데이타 구조와 이웃찾기 방법이다. 다음은 이웃 찾기 알고리즘인데, 첫 줄에나온 nodeprocedure EQUAL_ADJ_NEIGHBOR (P,D) 에서 P는 node이고 D는 찾아야 할 이웃방향인 WEST, EAST, SOUTH, NORTH중 의 하나이다.

이 알고리즘은 node P의 수평 또는 수직방향 D쪽에 있는 크기가 같거나 더 큰 이웃을 찾는 알고리즘이다.



(그림 3.3) 쿼드트리의 각 노드에 저장되는 정보
(Fig 3.3) The stored information in each node of Quadtree.

Node Procedure

```
EQUAL_ADJ_NEIGHBOR(P,D)
/* 노드 P의 수평 혹은 수직 방향 D 쪽에 있는 크기가 같은 이웃을 찾는다. */
  begin
    value node P;
    value direction D;
    return(SON(ADJ(D, SONTYPE(P)) then
EQUAL_ADJ_NEIGHBOR(FATHER(P), D)
       else FATHER(P)
       REFLECTED(D, SONTYPE(P))));
  end;
end
```

(4) 교차 후보쌍 리스트(Rival)

겹치기 검사를 통과한 한쌍의 곡면을 교차후보쌍(rival) 이라 부른다. 본 연구에서는 현재 교차 후보쌍 리스트와 교차 결과 리스트는 겹치기 검사를 통과했지만 평면으로 근사화되지 않은 쌍을 가리키며, 교차 결과 리스트는 한쌍의 패치가 모두 평면으로 근사화

되어 평면간의 교선 찾기를 수행 할 수 있는 교차 후보쌍 리스트를 말한다.

현재 교차 후보쌍 리스트은 행렬을 사용하여 저장하는데 전체흐름도에서 보듯이 편평도 검사를 하여 평면으로 근사화가 않된 경우, 분할하여 패치들사이에 겹치기 검사를 수행하여 겹치는 쌍에 등록시키는 과정을 반복하게 된다.

3.3 기하학적기법을 이용한 교선 검색 알고리즘

PCC를 이용한 접근방법은 자연 곡면들에 의해 한정되는 솔리드 모델링에 유용하게 지원된다. 먼저, PCC에 의하여 교선(Intersection edge)들을 근사한다. 이 방법은 단지 하나의 형태에 에지만으로 SSI(Surface /Surface Intersection)실행의 과정을 단순화 한다. 또한 이차 곡면 교선들을 정확하게 표현하는데 사용되고, 특히 환형체인 경우 효과적이다. 이에 대한 알고리즘은 다음과 같다.

```
Procedure Surface/SurfaceIntersection_test
begin
  Overlap_test;
   Overlap compare to the two pathes;
    if two patches are overlaped
     then begin
     Intersection-edge = subdivision two
     patches;
   Overlap_test end
  Flatness_test;
new1 = MakeNewside(vertex.begin-edge,
vertex);
new2 = MakeNewside(vertex. end-edge,
vertex);
  if new1 and new2 are include in
  planar approximate tolerance than
   pass to Flatnesstest;
    compute to judge by
    approximation of the planar
    quadrilateral;
   Flatness_test end
end
  Node Procedure
    EQUAL_ADJ_NEIGHBOR(P,D)
    begin
     value node P;
     value direction D;
     return(SON(ADJ(D, SONTYPE(P)) then
EQUAL_ADJ_NEIGHBOR(FATHER(P), D)
      else FATHER(P)
       REFLECTED(D, SONTYPE(P))));
    end;
  end

Procedure IntersectionCurve
/* Intersecting the two surface1 and surface2 */
begin
  edge1 = first edge in the
          surface1.patch1.edges;
  edge2 = first edge in the
          surface2.patch2.edges;
  p = IntersectTwoEdges(edge1, edge2);
   subdivision edge1 and edge2 at p;
   Create and order three new
   vertices;
   Find the vertices into the DFS;
   Inset the three vertices into
   vertex-queue with sorted order;
  end

Procedure PCCInterpolation
/* execute intersection collection with one
edge and an IntersectionVertex
structure-intersection */
  begin
   edge1, edge2 = subedges of edge find
                  at the intersection;
   Insert edge1 into a correct position in
  the sorted list of intersect.subedges;
   next_edge = the edge next to
    edge1 intersect.subedges;
    find-vertex = make a new connected
    type vertex with edge1, edge2;
  if (edge1, next_edge) pair satisfy the
```

```
PCC constraint then repeat the above
steps to the subedges;
    intersection_curve = intersection_edge
                +PCC constraint;
end
```

4. 교선 추적 알고리즘의 분석

임의의 두 곡면간의 교선을 찾기 위한 기존의 알고리즘은 연속적으로 추적하여 나타낼 수 있는 곡선은 선분(line segment)들과 각 패치들에 관한 위상 정보(topological information)를 이용, 계산하여 얻어진 선분들을 연결함으로서 생성된다. 그리고, Newton-Raphson 방법을 사용하여 각 세그먼트들을 재정의한다. 이 방법은 보간방정식으로 계산하는데 있어서 수렴성이 보장된다. 또한 미분가능한 기울기를 가지는 매끄러운 곡선이 연속으로 결합되어지는 선분을 따라 구간 곡선을 검사한다.

4.1 교점과 교선의 계산

1. 교점(Intersection point)과 접선을 계산한다.

첫 번째 단계는 기본 자유면(natural primitives)들일 경우, 곡선과 곡면간에 교선 계산은 자유곡면을 가지는 원호들이나 혹은 선들의 교점을 변형시킨다. 이와 같은 계산은 효율적인 실행방법으로 [2]에서 설명되었다. 교선(Intersection curve)에서 접선은 교점에서 두 곡면간에 정상적인 교차에 생성을 추적한다.

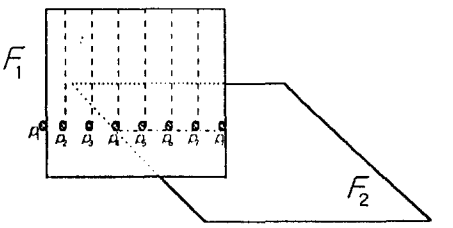
2. 에지들에 속한 점들을 정렬한다.

두 번째 단계는 [6][12]에서 제안된 알고리즘에서 효율적이지 못했던 부분들이다. 본 제안은 하나의 기본적인면인 2차원 매개변수들로서 원소들의 배열로 교점을 사상한다. 이 방법은 검색없이 실제적으로 연속적인 점들을 찾는다. 이 이유는 귀납적 방법으로 설명하기에는 불충분하므로 두 곡면이상이 서로 매칭하는 4개의 서브셀(subcell)들에 대해 어떤 셀들을 반복적으로 재분할하여 구현한다.

3. PCC에 점들을 보간한다.

세 번째 단계는 연속적인 교점들의 쌍과 꼬인 두 개의 원호(twist bi-arc)들을 가지는 접선들을 보간한다. 그러므로 모든 교점(Intersection point)들로서 에지들을 보간하는 PCC를 생성한다. 따라서 다음과 같은 그림에서 F_1과 F_2는 자유곡면 S_1과 S_2에 각각 포함되는 두 패치들일 경우, S_2에 S_1의 교점은 1차원의 집합에 있거나 교점이 없을 수 도 있다. 그러므로 특수한 경우인 생성기가 S_2상에 놓인 다는 가정이 없이도 적어도 4개의 교점들을 얻는다. 이 교점들은 S_2상에서 F_1에 놓이고 매개변수(u, v)를 사용하여 F_1을 매개변수화 공간(parametric space)에서 나타낸다. S_1상에 점 P가 주어질 경우, F_2의 바깥점들을 쉽게 제거되고 F_2의 매개변수화 공간에서 (u, v)값들이 계산된다. 또한 F_2의 경계선과 일치하는 u-한계와 v-한계를 가지고 (u, v)값들을 비교한다. 그러나 이점들에서 바깥쪽의 일부 점들과 F_2상에 놓인 점들은 두 개의 원호들로 정의되어야 한다. 두 개의 원호는 곡면 S_1과 S_2간에 교점 에지들의 세그먼트 E에 근사치를 구한다. 두 개의 원호의 일반성이 교차곡면에 꼭 필요하지 않을 경우, 세그먼트 E는 F_2에 일부분을 의미하고, 두 개의 원호는 F_2상에 일부분으로 나타낼 수 있다. 이는 F_1의 v-곡선들과 유사한 교점들을 계산하는데 필요하다.

그러나, F_1의 u-곡선들과 거의 평행한 큰 교차에지들은 잘못 계산되어질 수 있다. 그러므로 단지 F_1의 u-곡선들만 사용할 경우, 교차 에지들을 잘못 계산하는 경우가 발생된다. 교점에 접선은 근사값을 구하기 위한 추가적인 조건으로 사용한다. 교점 P에서 교선에 실제 접선은 S_1과 S_2상에 점 p에서 정상적으로 교차하여 생성함으로써 계산된다.



(그림 4.1) 곡면/곡면간의 교차점 계산
(Fig 4.1) Intersection point calculation between Surfaces

자유 곡면의 정상적인 면은 쉽게 계산될 수 있는데, 두 정상적인 면들이 동일 선상에 있을 경우, 두 곡면은 교차 에지의 자체교차(self-intersecting)이거나

점들 혹은 접선의 곡선중 하나를 나타내는 접선이다.

4.2 교선 추적알고리즘의 비교분석

제안된 알고리즘은 다음과 같은 방법으로 기존에 있는 방법과의 차이점이 있다.

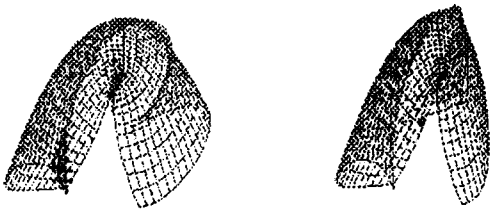
첫째로, 알고리즘은 divide-and-conquer 방법을 기초로 한다. 곡면의 각각에 패치들은, 서브패치들과 연결된 교차후보쌍(rival list)이 있다. 곡면분할과정에서, 교차쌍 정보는 계속연결되고 자동적으로 수정된다. 또한 깊이우선탐색(Depth-First Searching)방법으로가능한 한 효과적으로 두 곡면간에 교차들을 순서적으로 찾아낸다. 그러므로, 곡면분할방법의 사용과 두 곡면들에 패치를 기본으로 하여 불필요한 divide-and-conquer 단계[14] 들을 피한다. 따라서 효과적인 알고리즘을 제안한다.

둘째로, 두 곡면들간에 교차선을 찾을 경우, 알고리즘은 다음 단계로 곡면들의 응집력 성질을 사용하여 교차선의 전체길이를 찾을때 까지 이미 얻어진교차선을 저장한다. 따라서 교차선은 후위정렬(post-sorting)에 방법을 피하고, 곡선들의 세그먼트들에 집합 보다는 자연스럽게 연속적인 점 으로 표현할 수 있다.

셋째로, 전체 교선을 결정한다.

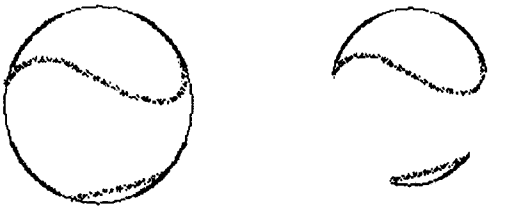
실험은 단지 어떤 하나의 에지 형태만을 취급하기 때문에 매우 간단하게 구현된다.

PCC는 적은 수 의 세그먼트로 확장된 에지들과 시각적으로 구분할 수 없는 PCC에 에지들에 대해 근사치를 구하여 계산한다. 따라서, 두 곡면들간에 교차가능한 교차선을 모두 찾는다.



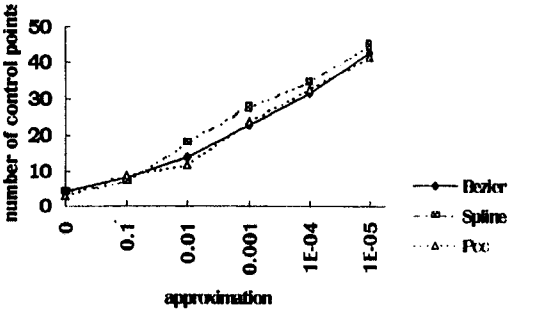
(그림 4.2) 자체 교차(self-intersection)을 가지는 곡면
(Fig. 4.2) The surface with self-intersection

그러나 교차선을 단순하게 추적만을 가지고 비교하면 비용이 많이 들게 된다. 그러므로 먼저 정확하게 두 곡면들간에 교선의 연속성(continuity)을 정의하고, 존재하는 교선(intersection curve)을 알고리즘을 사용하여 찾는다. 그러므로 실제의 교차점의 갯수가 많지 않고 교선분들이 교차점 부근을 제외하고는 거의 근접하지 않을 경우, 본 제안방법을 사용하여 위에서의 두 개의 문제점들을 쉽게 해결한다.



(그림 4.3) 교차선 추적 알고리즘을 이용한 결과
(Fig 4.3) The result using the intersection tracing Algorithm.

따라서 본 논문은 PCC에 접선 벡터와 보간된 점들을 이용한 교선을 생성하는 새로운 기법을 나타낸다. 이 방법들은 자연스런 곡면(natural surface)를 지원하는 모델러에서 교차 곡선들을 근사하여 생성하고 출력하는 데 용이하다. 또한 시간상의 결과는 비 2차 곡면(nonquadratic surface)들의 근사된 교차 에지들과 본 논문에서 계산된 교차 에지들간의 비교하였을 때, 시간상의 결과가 효과적이다.



(그림 4.4) 3차 곡면들의 교차 교선 추적에 대한 근사치에 대한 비교
(Fig. 4.4) The comparison with approximation for intersection curve tracing of cubic surfaces.

5. 결 론

기하 모델링에서 임의의 두 곡면과 곡면이 만나서

생기는 교차점과 두 곡면간에 교차 곡선에 계산은 가장 기본이 되는 작업이다. 이를 해결하기 위해서 기하학적인 문제에 대한 수치적 해법들이 많이 사용 되었다. 그러나 수치적 방법들은 반복계산을 위한 초기값을 선정하기가 어려울 뿐만 아니라 수렴성 문제가 대두된다. 이러한 문제를 해결하기 위한 다른 방법은 분할이론(subdivision theory)에 기초를 둔 기하학적 방법으로 주어진 임의의 두 곡면을 반복적으로 분할한 다음 여러개의 평행 사변형으로 근사한 후 근사된 평형 사변형간에 생기는 모든 교차 선분들을 찾아 내는 방식으로서 쿼드 트리의 자료구조를 사용하였다. 본 논문에서는 PCC 로서 접선 벡터(tangent vector)들과 보간된 점들로 교차곡선을 추적하는 새로운 방법을 제안한다. 이 방법은 자연 곡면(natural surface)들을 지원하는 시스템에서 교차 곡선들을 근사하는데 적절하다. 또한 PCC는 단지 자연 곡면(natural surface)들만 지원하는 형상 모델링에서 스위핑(sweeping), 블랜딩(blending)에 조건을 제공한다. 앞으로의 연구 과제는 두 곡면에 교차 계산의 좀 더 안정된 해를 찾기 위한 계산상의 오차를 줄이는 연구와 경계 계산의 한계범위를 효과적으로 찾는 방법이 요구된다.

참 고 문 헌

[1] A.A.G.Requicha "Representations for Rigid Solis : Theory, Methods, and System", ACM Surv.12. No.4. pp 437-464. December. 1980.

[2] A.A.G.Requicha and H.B Voelcker, "Boolean Operations in Solid Modeling : Boundary Ebaluation and Merging Algorithms", Proc. IEEE 73, No. 1 pp 30-44. January. 1985.

[3] B.Hamann and J.L.Chen, "Data point selection for piecewise linear curve approximation", CAGD II. pp 289-301. 1994.

[4] Bajaj, c., Hoffmann, C., Hopcroft, J., and Lynch, R., "Tracing Surface Intersections", Computer Aided Geometric Design, Vol.5, pp. 285-307, 1988.

[5] D.Lasser, "Intersection of parametric surfaces in the Bernstein Bezier representation", CAD, Vol. 18, pp 186-192. 1988.

[6] Hiroaki Chiyokura, "Solid Modeling with DESIGNBASE", ADDISON WESLEY, 1989.

[7] Hanan. Samet, "Application fof Spatial Data Structures", ADDISON WESLEY. 1989.

[8] J.M.Lane and R.F.Riesenfeld, "A theoretical developement for the computer generation of piecewise polynomial surface", IEEE Trans. PAMI 2, pp 35-46. 1980.

[9] Jaroslaw R.Rossignac, Aristides A.G.Requicha, "Piecewise Circular Curves for Geometric Modeling", IBM Journal. No 5, pp 177-190. July. 1988.

[10] K.Harada and E. Nakamae, "Application of the Bezier Curve to Data Interpolation", Comput. Aided Design 14, No 1, pp 55-59. January. 1982.

[11] M.Sabin, "Contouring-A Review of Methods for Scattered Data", Mathematical Methods in Computer Graphics and Design, K.W.Brodlie, Ed., Academic Press, New York. 1980.

[12] N.Beckmann, H. Kriegel. R. Schneider, and B. Seeger, "The R^*-tree : An efficient and robust access method for points and rectangles", Proc. ACM SIGMOD Conf., May, 1990.

[13] P.A.Koparkarand S.P, Mudur "Computational Techniques for Processing Parametric Surfaces", Comput. Vision. Graph. & Image Proc. 28, No 3 pp 303-322. December. 1984.

[14] Q.S.Peng, "An algorithm for finding the intersection lines between two B-Spline surfaces", CAGD. Vol 16. No 4, pp 191-196. July. 1984.

[15] R.B.Tilove, "Set Membership Classification : A Unified Approach to geometric Intersection Problems", *IEEE Trans. Computers C-29*, No. 10, pp 874-883. October. 1980.

[16] R.N.Wolfe, M.A. Wesley, J.C.Kyle, F.Gracer, and W.J.Fitzgerald, "Solid Modeling for Production Design", IBM J.Res.31, pp 277-295. May. 1987.

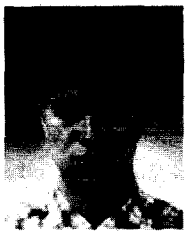
[17] T.Brinkhoff, H.P. Kriegel. R Schneider and B. Seeger, "Multi-step processing of spatial joins", Proc. ACM SIGMOD., 1994.

[18] Toussaint, G.T., "On the Complexity of Approxi-

mating Polygonal Curves in the Plane", Proc. of IASTED Int'l Symp. *on Robotics and Automation*, Lugano. Switzerland, 1985.

[19] W.Tiller, "Rational B-Splines for curve for andSurfaceRepresentation", *IEEE Comput. Graph. & Appl. 3*, No. 6. pp 61-69. September. 1983.

[20] Yihong Gong, Hongjiang Zhang, H.C Chuan, M. Sakauchi, "An Image Database System with Content Capturing and Fast Image Indexing Abilities", *Proceedings of the International conference on Multimedia computing and Systems*, pp 121-130. 1994.



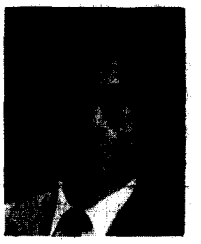
박 철 호

1992년 광운대학교 이과대학(이학사)

1994년 광운대학교 대학원 전자계산학과(이학석사)

1995년~현재 광운대학교 대학원 전자계산학과 박사과정

관심분야:컴퓨터 그래픽스, 계산기하학, 이미지 매칭, 알고리즘



홍 성 수

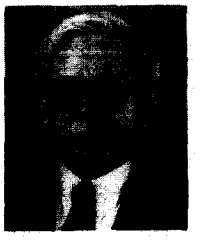
1979년 광운대학교 전자계산학과(이학사)

1983년 광운대학교 대학원 전자계산학과(이학석사)

1990년 광운대학교 대학원 전자계산학과(이학박사)

현재 호서대학교 전자계산기공학부 교수

관심분야:알고리즘, 이미지 검색, 자연어 처리



심 재 홍

1967년 서울대학교 수학과(이학사)

1980년 고려대학교 대학원 수학과(이학석사)

1988년 경희대학교 대학원 수학과(이학박사)

1984년~1986년 정보과학회 부회장 역임

1984년~현재 광운대학교 전자계산학과 교수

관심분야:컴퓨터 그래픽스, 수치해석학, 알고리즘