

## 실시간 곡면 가공에 관한 제어 알고리즘 및 하드웨어 연구

### Realtime Control Algorithm and Hardware for Machining Curved Surfaces

정 승 권, 권 옥 현

서울대학교 전기공학부 (Tel: 880-7314; Fax: 871-7010; E-mail: jsk@cisl.snu.ac.kr)

**Abstract** This paper describes an interpolation method for a parametric surface. A parametric surface is approximated to triangular mesh surfaces and then the basic paths are achieved. As the generated path is a series of linear segments, this algorithm can be easily adapted to general NC controllers. The generated paths have minimal transfer length and are gouge-free within the approximation tolerance. The problems, induced when the paths are represented by linear segments, are overcome without making any path deviation by this algorithm. This algorithm saves machining time by eliminating overdetermined tool paths and keeping the desired average feedrate, which improve productivity and lead to lower production costs.

**Keyword** CNC(Computerized Numerical Control), CL(Cutter Location), CC(Cutter Contact)

#### 1. 서론

고기능 CNC 컨트롤러의 여러 필요한 기능 중, 곡면 가공시 가공경로의 온-라인 계산과 계산된 경로의 실시간 처리가 크게 요구되고 있다. 현재 상용 수치 제어 공작 기계를 이용하여 곡면 가공을 위해서는 호스트 컴퓨터에서 CAD작업을 통해 곡면을 정의해야 한다. 곡면이 정해지면 곡면을 가공하기 위한 공구의 이동 경로가 호스트 컴퓨터에서 계산되고 계산된 이동 경로는 대개 표준 NC 코드로 표현되어 공작 기계에 입력된다. 이렇게 오프-라인(off-line)으로 이동 경로를 계산할 경우 계산된 데이터(CL data)량은 매우 많다. 따라서 공작 기계가 이동 경로에 대한 데이터를 모두 수용할 수 있으려면 매우 큰 메모리를 가져야 하며, 주어진 절삭 작업의 정밀도가 높아질수록 데이터량은 제곱배 이상 증가하게 된다. 그런데 만일 공작 작업 중 CL(Cutter Location) 데이터가 온-라인(on-line)으로 호스트 컴퓨터에서 공작 기계로 전송된다면 호스트 컴퓨터는 데이터 전송에 전용되므로 이것은 자원을 비효율적으로 이용하는 방법이 된다. 또한 오프-라인으로 계산된 공구 이동 경로는 공작 기계의 상태를 충분히 고려하지 못하므로 최적의 해를 제공하지 못한다. 따라서 이러한 문제를 해결하기 위하여 곡면 가공을 위한 공구 이동 경로를 공작 기계가 온-라인 상에서 계산할 수 있는 연구가 진행 중에 있다[1][2]. 온-라인 공구 이동 경로 계산에 관한 기존의 연구에서는 기본적인 알고리즘과 그것을 검증하기 위한 높은 성능의 하드웨어 시스템만이 제안되어 있다. 이것들이 실제로 상용 시스템에 적용되기 어려운 이유는 입력되는 곡면의 모양과 허용되는 가공 오차, 공구의 모양 등에 따라 요구되는 계산량이 달라지는데 그것에 대한 분석이 정확하게 제시되지 않았기 때문이다.

CNC 제어가 곡면식이 아닌 곡선식까지만 입력받을 수 있도록 하는 실시간 커브 인터플레이터에 대한 연구도 활발히 진행 중에 있다[4][5][6]. 곡선식을 입력받는 것은 곡면식을 입력받는 것보다는 조금 뒤진 기능이지만 직선 보간과 원호 보간만을 수행하는 일반적인 상용 CNC 제어기에 추가되어야 할 필요한 기능이다. 일반적인 자유곡선으로 공구를 움직이려면 그것을 선분의 조각으로 근사화하여 직선 보간을 수행하거나 또는 원호의 조각들로 표현하여 원호 보간을 수행하여야 한다[7]. 이러한

방법은 모두 곡선을 근사화시켜서 가공 경로를 찾는 것이므로 직접 곡선을 보간 하는 것보다 가공 오차가 클 수밖에 없다. 곡선식은 매개 변수식으로써 표현된다고 가정하는데 매개 변수 곡선식은 일반적인 모든 곡선을 표현할 수 있으며 곡선을 정의하기도 용이하다. 이렇게 표현된 곡선을 보간 하는 알고리즘들은 직선 보간과 원호 보간에 대한 비교를 제시하여 계산량과 가공 오차를 정량적으로 분석하였다. 그러나 형상이 곡면으로 정의되고 그것을 가공하는 경로를 얻을 때 곡면식에서 가공에 필요한 곡선식을 얻는 것이 어려운 일이므로 그것에 대한 해결이 있어야 한다.

곡면의 온-라인 가공에 있어 효과적인 알고리즘뿐만이 아니고 알고리즘을 실시간에 풀기 위한 하드웨어 구조에 대한 깊이 있는 연구가 필요하다. 알고리즘에서 요구하는 계산량은 곡면의 모양과 가공 오차 한계, 공구 모양 등의 입력 조건에 의하여 결정되므로 하드웨어 구조 또한 고정된 결정적인 구조가 아니라 확장 가능한 유연성 있는 구조가 되어야 한다. 곡면 가공을 위한 기존의 연구에서 가공 경로를 온-라인 실시간으로 계산하는 체계적인 연구 결과는 아직 얻어지지 못하였다. 따라서 본 연구에서는 효과적인 공구 이동 경로를 구하는 알고리즘, 알고리즘의 오차 허용성에 대한 검증, 알고리즘의 실시간 계산을 위한 하드웨어 구조 등을 제안하고자 한다.

본 논문의 구성은 다음과 같다. 2절에서는 효과적인 가공 경로로 생성 알고리즘을 제안하며 이것의 온-라인 처리 방법을 3절에서 보인다. 제안된 알고리즘의 실제 구현 결과를 4절에서 제시하고 5절에서 논문을 결말짓는다.

#### 2. 곡면 가공 경로 생성

곡면을 수식으로 서술해야 할 경우에는 대개 매개변수식을 사용한다. 매개 변수에 의한 표현 방법은 곡선이나 곡면을 가장 잘 표현할 수 있으므로 CAD 소프트웨어에서는 널리 쓰이는 방법이다. 흔히 매개 변수 표현법으로 표현된 매개 변수식이라 하면 매개 변수 입방 곡선(Parametric Cubic Curve)과 이것을 확장한 매개 변수 제입방 곡면(Parametric Bicubic Surface)을 의미한다. 이

렇게 표현된 곡선 혹은 곡면은 모양이 콘트를 포인트에 의해 결정되므로 작성이 용이하며 함수식 자체가 항상  $C^2$  연속을 보장하므로 곡선 혹은 곡면의 모양을 변경하여도 부드러움이 유지되는 장점이 있다. 매개변수로 곡선을 표현할 때는 계수 설정 방법에 따라 Hermite 곡선, Bézier 곡선, B 스플라인(B-spline) 곡선 등의 종류가 있는데 이것들은 서로 다른 것으로 표현이 가능하며, 특히 B 스플라인은 가장 일반적인 표현방식이다. 따라서 본 연구에서는 곡선이 B 스플라인으로 표현되었다고 가정한다.

곡면을 가공하기 위한 공구 경로 생성에서는 첫번째로 드라이브 곡선에 대한 고려를 해야 한다[8]. 드라이브 곡선은 공구가 이동해야 할 경로를 미리 지정하여 공구축(일반적으로 Z축)에 수직인 평면(일반적으로 X-Y 평면)에 투영하여 표현한 것이다. 공구는 드라이브 곡선 위를 움직이며 공구축 방향으로 공구를 움직여 가공을 수행한다. 그런데 매개변수 함수식으로 표현된 곡면의 가공에서 드라이브 곡선을 직교좌표계상에서 지정하는 것은 매우 어려운 문제이므로 일반적으로 매개변수 영역에서 드라이브 곡선을 설정하는 방법을 많이 사용한다. 이것은 매개변수 영역에서 드라이브 곡선을 설정하고 곡선을 일정 간격으로 나누어 각 매개변수 포인트에 대응하는 직교좌표계에서의 값을 구하여 여기에 공구를 위치시키는 것이다. 이와 같은 방법의 문제점은 매개변수 영역에서 일정간격으로 나누었다고 해서 그것이 직교좌표계 영역에서 일정간격이 되지 않는다는 것이다. 그러므로 효율적인 가공 경로를 얻기 위해서는 드라이브 곡선은 직교좌표계상에서 총 공구 이동 길이가 가장 짧고 Cusp의 높이가 일정 범위 이하로 제한되도록 결정되어야 한다[9][10].

곡면은 매개변수식으로 표현되는 것이 좋지만 매개 변수식으로 표현할 경우 직교좌표계상에서 이루어지는 가공에 대한 공구 이동 경로를 구하기가 어려워진다. 이러한 문제를 해결하기 위해 공간상에서  $C^2$  연속인 곡면을 삼각형의 평면 조각들의 조합으로 표현하는 방법이 연구되었다[2][3]. 곡면을 삼각형의 조각들로 근사화시켜 표현할 경우 각 삼각형은 직교좌표계상에서 꼭지점의 좌표만으로 모양이 결정되므로 곡면을 더 이상 매개 변수 영역에서 고려하지 않아도 된다는 장점이 생긴다. 즉 공구 이동 경로를 구하는 작업이 모두 직교좌표계상에서 이루어지므로 계산이 쉬워지고 절삭 에러 역시 일정한 한계 안에서 효과적으로 제한할 수 있다. 또 하나의 장점은 곡면이 평면으로 표현됨으로써 얻어진다. 드라이브 곡선 위의 점에서 공구의 Z 위치를 구할 경우에 공구가 곡면에 접할 때의 위치를 구하는 것보다 평면에 접할 때의 위치를 구하는 것이 상대적으로 쉽기 때문이다. 이러한 곡면의 근사화는 곡면의 굴곡 정도에 따라 삼각형의 크기를 변화시켜 근사화하므로 전체곡면을 표현하는데 데이터 양에서 큰 부담을 주지 않는다.

곡면이 평면 조각 즉 삼각형의 조각들로 근사화되어 표현되었을 때 이를 가공하기 위한 공구 이동 경로는 각 삼각형을 가공하기 위한 공구 이동 경로에서 얻는다. 드라이브 곡선이 일정간격 떨어진 평행한 직선으로 설정되었을 때 하나의 삼각형을 가공하기 위한 각 드라이브 곡선마다의 공구의 시작 위치와 끝 위치를 결정하고 시작 위치와 끝 위치를 직선으로 이으면 삼각형을 가공할 수 있는 공구 이동 경로를 얻을 수 있다. 그림 2.1과 같이 하나의 삼각형을 가공하는 경로 즉 공구 중심의 이동 경로는 삼각형에 평행한 몇 개의 선분 조각으로 표현될 수 있다. 각 선분은 정해진 드라이브 곡선위로 투영될 수 있고 이에 해당하는 공구 접촉점의 경로는 삼각형 위에서 모서리와 모서리를 잇는 선분이 된다.

드라이브 곡선이 Y축에 평행한 직선으로 시작위치  $X_{st}$ 에서  $Dfx$ 의 간격으로 그림 2.1과 같이 설정되었다면 삼각형을 가공하기 위한 공구 이동 경로는 다음과 같이 구할 수 있다.

- (1) 세 꼭지점의 좌표를 X좌표 크기로 Sort한다. 크기순으로 각각 min, mid, max 인덱스를 붙인다.
- (2) 크기가 공구의 반지름이고 Z 성분이 음수인 삼각형의 법선 벡터 (a,b,c)를 구한다.
- (3)  $x_{min} < X_{st} + n \cdot Dfx + a < x_{max}$ 를 만족하는 자연수 n을 구하여 각 n에 대해 (4)-(5)의 과정을 반복한다.
- (4) 각 n에 대해  $x = X_{st} + n \cdot Dfx + a$  라는 평면과 삼각형의 모서리와와의 교점  $(X_{st} + n \cdot Dfx + a, y_{n1}, z_{n1})$ ,  $(X_{st} + n \cdot Dfx + a, y_{n2}, z_{n2})$ 를 구한다.  $y_{n1}, z_{n1}, y_{n2}, z_{n2}$ 는  $X_{st} + n \cdot Dfx + a$ 과  $x_{min}, x_{mid}, x_{max}$ 들과의 비례식에 의해 구할 수 있다.
- (5) 구해진 교점에서 (2)에서 구한 벡터를 뺀 값, 즉  $(X_{st} + n \cdot Dfx, y_{n1} - b, z_{n1} - c)$ 와  $(X_{st} + n \cdot Dfx, y_{n2} - b, z_{n2} - c)$ 를 잇는 선분이 n번째 드라이브 곡선에 대한 공구 이동 경로이다.

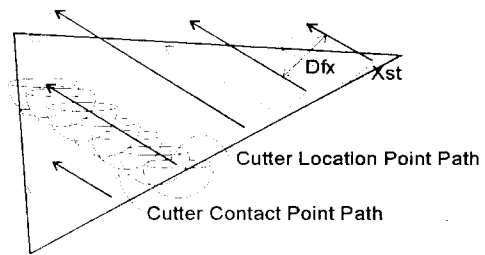


그림 2.1 삼각형 가공을 위한 이동 경로 조각

각 삼각형마다 위의 알고리즘을 실행하여 공구 이동 경로의 조각들을 구한다. 구해진 조각들은 드라이브 곡선별로 모아져진 체적인 공구 이동 경로를 형성하도록 한다. 그림 2.2는 드라이브 곡선을 포함하는 평면( $x = X_{st} + n \cdot Dfx$ )에서 공구 이동 경로 조각들을 표시한 것이다. 공구의 기하학적 모양과 삼각형의 공간상에서의 위치에 따라 그림과 같이 평면에는 여러 형태의 공구 이동 경로 조각들의 나타나게 된다. 그림 2.2에서  $\overline{ab}$ ,  $\overline{ca}$ 와 같이 이동 경로에 간섭에 생기게 되면 과잉 절삭을 방지하기 위해  $\overline{ae}$ ,  $\overline{ea}$ 를 공구 이동 경로로 하며,  $\overline{fg}$ ,  $\overline{gh}$ 와 같이 연결이 끊어진 부분은 두 선분의 연장선상에 있는  $j$ 를 지나  $\overline{fj}$ ,  $\overline{ji}$ 를 공구 이동 경로로 한다. 서로 다른 삼각형들에서 이동 경로 조각들이 생성되므로  $\overline{ku}$ 과 같은 데이터가 생길 수 있다. 과잉 절삭을 방지하기 위해 이 부분에서 이동 경로는  $\overline{ji}$ 가 되므로  $\overline{ku}$ 를 생성한 삼각형은 가공이 되지 못한다. 이때  $\overline{ku}$ 를 생성한 삼각형의 법선 벡터 중 크기가 공구의 반지름이고 Z 성분이 음수인 것을 (a,b,c)라 하면  $(X_{st} + n \cdot Dfx, y' + b)$ 과  $(X_{st} + n \cdot Dfx, y'' + b)$ 의 사이 구간에서 Under-Cut Error  $E_u$ 가 발생한다는 것을 계산할 수 있게 된다.

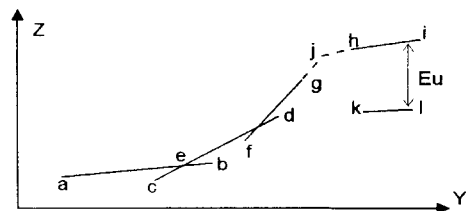


그림 2.2 드라이브 평면위의 공구 이동 경로

이와 같이 근사화된 삼각형에서부터 공구 이동 경로를 구하는 것은 Cusp 높이를 제한하는데도 잊점이 있다. 기존의 곡면을 평면화시켜 공구 이동 경로를 구하는 알고리즘에서는 효율적인

드라이브 곡선 설정에 관한 해결책을 제시하지 못했다. 즉 이동 경로를 따라 공구가 움직일때의 피삭체의 과잉절삭에 대한 문제는 어느정도 해결했으나 이동 경로 사이에서 생기는 Cusp의 제한에는 뚜렷한 해결책을 보이지 못했다. 일반적인 경우 드라이브 곡선은 직교좌표계상에서 등간격의 직선들로 설정되는데 그러할 경우 그림 2.3에서와 같이 가공면의 모양에 따라 다른 Cusp를 남기므로 가공되는 면의 드라이브 곡선 직각 방향으로의 기울기에 따라 드라이브 곡선 간격이 다르게 결정되어야 한다. 본 연구에서 제안된 알고리즘의 경우 평면화된 삼각형별로 가공 경로를 구하므로 삼각형의 기울기에 따라 주어진 드라이브 곡선 사이에 새로운 드라이브 곡선을 추가하여 가공 경로를 구할 수 있다.

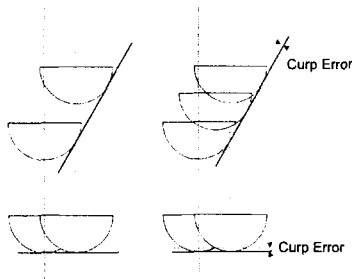


그림 2.3 드라이브 곡선 간격과 Cusp

### 3. 실시간 곡면 가공을 위한 분산처리

공작 기계에서의 실시간성 문제는 공구의 위치를 결정하는 연산 작업부와 모터를 제어하는 서보 제어부 사이에서 발생할 수 있다. 사용자에게 의해 가공을 명령하는 입력 즉 NC 프로그램이 입력되었을 경우 연산 작업부는 프로그램을 해석하여 서보 제어부에 일정 주기로 이동 명령을 주어야 한다. 명령을 주는 주기는 모터와 서보 제어부의 성능에 따라 결정되는데 가공 작업을 수행할 경우 연산 작업부에서 정해진 주기 내에 이동 명령을 서보 제어부에 주지 못하면 공구는 이동과 정지를 반복하므로 가공이 고르게 되지 않게 된다. 그러므로 연산 작업부는 항상 정해진 시간 한계(Time Bound)내에 다음에 이동해야 할 공구의 위치를 계산해야 하며 이것을 보장하기 위해 공작 기계 제어부는 연산 작업부에 항상 우선 순위를 두어 자원(Resource)을 할당해야 한다. 수치제어 공작기계가 가공면에 대한 곡면식을 입력받아서 이를 가공할 수 있도록 한다면 실시간성 문제를 고려하지 않을 수 없게 된다. 연산 작업부에서 곡면식을 분석하여 공구 이동 경로를 얻고 이것으로부터 미세 이동치를 서보 제어부에 넘겨주는 것은 많은 계산을 필요로 하기 때문이다. 따라서 곡면의 온라인 실시간 가공을 위해서 자원을 최소로 이용하면서도 빠르게 계산되는 공구 이동 경로 생성 알고리즘과 알맞은 하드웨어 구조가 필요하다.

곡면이 매개변수 함수식으로 정의되어 있다고 할 때 곡면을 가공하기 위해서는 크게 2절에서 제시된 두가지의 단계, 즉 곡면을 평면의 조합으로 근사화시키는 것과 근사화된 평면에서 공구의 이동경로를 구하는 두 단계를 거쳐야 한다. 만일 공작기계가 곡면의 함수식을 직접 입력받을 수 있다면 이러한 두 작업을 추가적으로 수행할 수 있어야 하는데 두 작업 모두 많은 계산량을 요구하므로 계산을 나누어 처리해야 할 필요가 생긴다. 따라서 본 연구에서는 공구 이동 경로를 실시간으로 얻기 위해 세단계의 파이프라인 구조를 제안한다(그림 3.1).

그림 3.1과 같은 분산 처리 구조에서는 첫 단계에서 곡면을

평면으로 근사화하는 작업을 수행하여 근사화된 삼각형의 데이터는 다음 단계로 전송된다. 두번째 단계에서는 전송되어 온 삼각형의 데이터를 통하여 공구의 이동 경로를 구하며 이동 경로 계산에서 더 이상 필요 없는 삼각형의 데이터는 메모리에 저장하지 않는다. 이와 같은 구조는 계산의 실시간성 문제와 메모리 오버플로우 문제를 모두 해결할 수 있다.

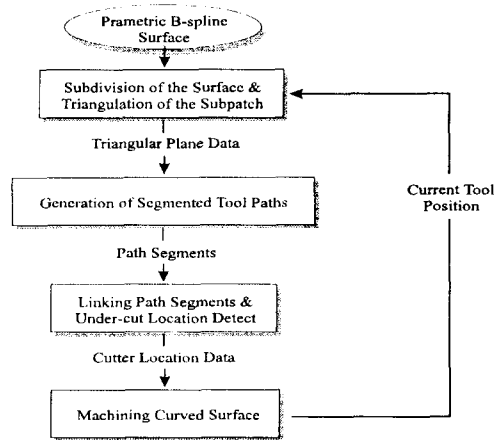


그림 3.1 알고리즘의 파이프라인 처리

이와 같은 파이프라인 구조에서는 2절에서와 같이 공구의 이동 경로를 구할 경우에 특정 방향으로 이웃한 삼각면들에 대한 데이터가 연속적으로 필요하게 된다. 드라이브 곡선 방향으로 이웃한 삼각면들의 데이터가 차례로 전송되어 올때 하나의 드라이브 커브에 해당하는 공구 이동 경로가 가장 빨리 얻어 질 수 있다. 평면화가 특정 방향으로 진행되도록 하기 위하여 트리에 기반한 알고리즘을 사용한다. 이것은 평면과 근사화된 곡면의 오차가 커서 영역을 4등분하였을 경우 드라이브 커브 방향으로 인접한 부분을 평면화시킬뿐 아니라 드라이브 필요한 가공 경로가 먼저 생성될 수 있도록 해당하는 부분을 먼저 평면화하여 가공 경로를 구하도록 한다.

### 4. 공구 경로 생성 알고리즘의 구현

개발된 공구 이동 경로 생성 알고리즘을 IBM-PC에 적용하여 시뮬레이션 하였다. 곡면식은 Uniform B-Spline곡면식으로  $0 \leq u \leq 4$ ,  $0 \leq v \leq 3$ 의 매개 변수 영역에 42개의 컨트롤 포인트를 사용한 곡면이다.

그림 4.1은 매개 변수 영역에 일정간격으로 라인을 그어서 그것을 직교좌표계 영역으로 변환시킨 것이다. 그림에 나타난 곡선은 곡면 위의 곡선들로 공구 접촉점의 경로가 되며 직교좌표계 상에서 곡선간의 간격이 일정하지 않음을 보여주고 있다.

그림 4.2는 곡면을 삼각면 조각으로 근사화시킨 것을 보인다. 곡면의 굴곡에 따라 다른 크기로 근사화되었음을 보인다. 그림 4.3은 평면으로 근사화된 곡면을 가공하기 위한 공구 이동 경로로서 2절에서 제안된 알고리즘을 이용하여 구한 것이다.

공구 이동 경로 생성 알고리즘의 수행에서 경로를 표현하기 위한 메모리 요구량은 매우 크다. 단 알고리즘이 온라인 가공을 위한 것이므로 가공한 경로를 메모리에서 지우기 때문에 주어진 시뮬레이션에서는 총 경로를 나타내기 위한 메모리 요구량의 1/20 수준의 메모리만을 할당함으로써도 메모리 오버플로우가 생기지 않았다. 총 경로를 나타내기 위한 메모리 요구량 대 온라인 가공에서 필요한 메모리 요구량의 비는 곡면이 더 작은 삼각형으로 근사화될수록 작아지며 궁극적으로는 총 요구량의 재곱 근사값의 메모리만을 가지고도 가공작업을 수행할 수 있게된다.

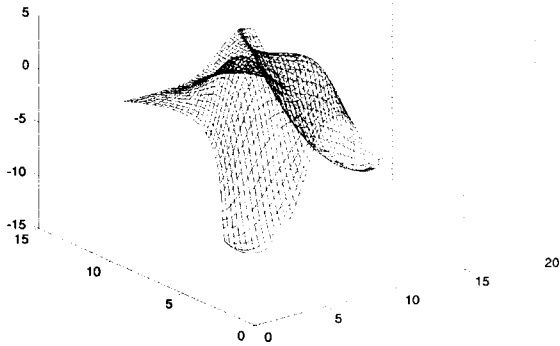


그림 4.1 Uniform B-Spline Surface

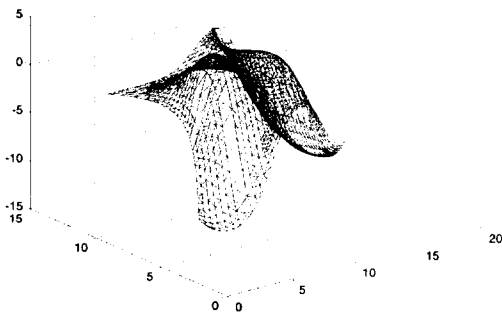


그림 4.2 Triangulation of the Surface

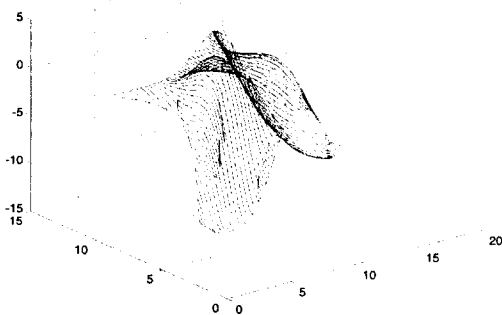


그림 4.3 Generated Tool Paths

## 5. 결론

발전된 CNC의 기능으로 곡면을 효과적으로 가공할 수 있는 방법을 보였다. 곡면 데이터를 받아서 가공에 필요한 공구 이동 경로를 온-라인으로 실시간에 계산하게 되면 공작 시간을 단축할 수 있고 이동 경로 계산에 공작 기계의 상태를 충분히 고려할 수 있으므로 이동 경로 계산에서 발생하는 오차를 줄일 수 있는 이점을 얻을 수 있다. 곡면의 실시간 가공에서 증가되는 계

산량과 메모리 문제를 해결하기 위해 하드웨어의 분산처리 구조를 제안하였다. 이 구조는 곡면 가공에 필요한 계산 작업을 평면 근사화와 공구 이동 경로 산출의 두 단계로 분리하여 병렬적으로 수행하는데 필요에 따라 공구 이동 경로 산출 작업이 또 다시 분산처리 될 수 있도록 하였다. 필요한 알고리즘 역시 이 구조에 적합하도록 구성하였는데 새롭게 구성된 알고리즘은 과잉 절삭을 방지하는 완전한 공구 이동 경로를 생성할 뿐 아니라 가공 후 생기는 과소 절삭 부위를 알아냄으로써 적은 공구를 사용하여 재가공을 할 경우에 효율적으로 과소 절삭 부위를 제거할 수 있게 하였다.

## 참고 문헌

- [1] Y. D. Chen, J. Ni, S. M. Wu, "Real-time CNC Tool Path Generation for Machining IGES Surfaces", ASME Trans. Engineering for Industry, vol. 115, pp.480-486, 1993.
- [2] S. Bedi, I. Ali, N. Quan, "Advanced Interpolation Techniques for N.C. Machines", ASME Trans. Engineering for Industry, vol. 115, pp.329-336, 1993.
- [3] A. Hansen, F. Arbab, "Fixed-Axis Tool Positioning with Built-in Global Interference Checking for NC Path Generation", IEEE Journal of Robotics and Automation, vol. 4, No. 6, 1988.
- [4] D. C. H. Yang and T. Kong, "Parametric interpolator versus linear interpolator for precision CNC machining", Computer-aided design, vol. 26 number 3 March, pp. 225-234, 1994.
- [5] M. Shpitahni, Y. Koren and C. C. Lo, "Realtime curve interpolators", Computer-aided design, vol. 26 number 11 November, pp. 832-838, 1994.
- [6] D. Kiritsis, "High precision interpolation algorithm for 3D parametric curve generation", Computer-aided design, vol. 26 number 11 November, pp. 850-856, 1994.
- [7] M. K. Yeung and D. J. Walton, "Curve fitting with arc splines for NC toolpath generation", Computer-aided design, vol. 26 number 11 November, pp. 845-849, 1994.
- [8] S. Marshall and J. G. Griffiths, "A new cutter-path topology for milling machines", Computer-aided design, vol. 26 number 3 March, pp. 204-214, 1994.
- [9] J. G. Griffiths, "Toolpath based on Hibert's curve", Computer-aided design, vol. 26 number 11 November, pp. 839-844, 1994.
- [10] J. J. Cox, Y. Takezaki, H. R. P. Ferguson, K. E. Kohkonen and E. L. Mulkay, "Space-filling curves in tool-path applications", Computer-aided design, vol. 26 number 3 March, pp. 215-224, 1994.