論文 한국정밀공학회지 제12권 제7호 1995. 7
Journal of the Korean Society of Precision Engineering Vol. 12. No7. July. 1995.

신경회로망 방식에 의한 복잡한 포켓형상의 황삭경로 생성

신 양 수,* 서 석 환*

Neural network based tool path planning for complex pocket machining

Yang-Soo Shin,* Suk-Hwan Suh*

ABSTRACT

In this paper, we present a new method to tool path planning problem for rough cut of pocket milling operations. The key idea is to formulate the tool path problem into a TSP (Travelling Salesman Problem) so that the powerful neural network approach can be effectively applied. Specifically, our method is composed of three procedures: a) discretization of the pocket area into a finite number of tool points, b) neural network approach (called SOM - Self Organizing Map) for path finding, and c) postprocessing for path smoothing and feedrate adjustment. By the neural network procedure, an efficient tool path (in the sense of path length and tool retraction) can be robustly obtained for any arbitrary shaped pockets with many islands. In the postprocessing, a) the detailed shape of the path is fine tuned by eliminating sharp corners of the path segments, and b) any cross-overs between the path segments and islands. With the determined tool path, the feedrate adjustment is finally performed for legitimate motion without requiring excessive cutting forces. The validity and powerfulness of the algorithm is demonstrated through various computer simulations and real machining.

Key words : Pocket milling(포켓가공), Rough cutting(황삭), Tool path planning(공구경로계획), Neural network(신경망회로), SOM(자기조직화지도), CAD/CAM

1. 서 론

포켓가공은 외부윤곽의 내부에 있는 피삭재를 모두 제거하여 원하는 형상을 얻는 절삭가공 방식이다. 포켓 형상은 사각, 원형, 트랙과 같은 단순한 형상으로부터 라인과 아크의 조합으로된 자유형상, 포켓의 내부에 아일랜드가 존재하는 복잡한 형상까지 다양하다. 가공단계는 황삭, 중삭, 정삭으로 구분되며, 대부분의 피삭재가 황삭에서 제거되므로 가공시간에 미치는 영향이 매우 크고, 최근 황삭가공의 중요성이 인식되면서 대부분

* 포항공과대학교 산업공학과

외 금형 CAD/CAM 시스템은 황삭 절삭기능을 제공하고 있다.

기존의 경로생성 방식은 크게 양방향 방식과 옵셋 방식으로 구분된다. 양방향 방식은 옵셋윤곽과 경로간격(stepover)에 따라 생성된 스캔 라인(scan line)과의 교점을 구하여 직선으로 공구경로를 생성하는 것으로 지그재그(zigzag), 계단형(stair case), 해치(hatch), 방향평행 밀링(direction parallel milling)으로 일컬어 진다. 옵셋 방식은 내부가 모두 제거될 때까지 윤곽을 반복적으로 옵셋하여 경로를 생성하는 것으로서, 창 구조(window frame), 윤곽평행(contour parallel milling)으로 일컬어진다. 이상의 두 가지 방식을 강건성(robustness), 공구후퇴(tool retraction), 경로 최적화 및 절삭력 측면에서 분석을 하면 다음과 같다.

양방향 방식은 경로생성 방식이 옵셋에 비해 간단하므로 복잡한 형상에 대해서도 로보스트하게 경로생성을 할 수 있는 이점이 있기 때문에 대부분의 CAM 시스템의 경우에 아일랜드를 포함하는 포켓형상은 이 방식으로 경로를 산출하고 있다. 그러나 공구가 아일랜드를 만나는 경우에는 공구후퇴가 불가피하고, 아일랜드를 지나 포켓내부로 진입하는 부분은 드릴링 작업이 필수적이기 때문에 가공시간에 미치는 영향이 크다. 이 문제를 해결하는 방안으로 아일랜드를 돌아서 가는 우회경로(detour)를 생각할 수 있지만 우회길이는 아일랜드의 형상에 따라 다를 수 있으며, 매우 긴 우회경로가 생성될 수 있는 경우가 발생할 수 있다. 경로와 경로를 이어주는 방법에 있어서 무조건 공구가 새로운 시작점으로 갈 수 있지만, 선 가공된 지점부터 공구경로를 이어주는 방식을 씀으로써 공구후퇴는 발생하지만 드릴링 작업을 없애줄 수 있다. 따라서 양방향 방식은 공구후퇴와 드릴링 작업을 줄이는 것이 매우 중요하며, 형상이 복잡해지면 기존에 제시되었던 방법들이 잘 적용되지 않는 문제점이 있다.

경로 최적화 문제는 라인으로 된 볼록 다각형, 다면체 형상에 국한되어 많은 연구들이 행해졌다.[3,9,10,11] Wang[3]은 1) 계단형 방식의 경우 가장 긴 선분에 평행한 방향이 최단경로가 되며, 2) 절삭방향이 경로길이에 미치는 평균 변화폭은 대체로 5 - 10 %지만 경우에 따라서는 100 %까지 보였으며, 3) 계단형 방식이 창 구조 방식 보다 더 짧은 경로를 생성하고, 4) 창 구조 방식에서 시작점은 경로길이에 크게 영향을 미치지 않는다는 결과를 보였다. Held[1]은 최적의 경사각을 구하는 문제는 포켓형상이 복잡해 짐에 따라 그 효과가 급격히 감소되며 또한 최적의 경사각이 최소의 공구후퇴를 보장하지 못한다고 언급하고 있다. 따라서 아일랜드를 포함하는 복잡한 형상에서는 이러한 연구결과 적용이 불가능하게 되므로 경로 최적화를 꾀할 수 있는 새로운 알고리듬이 필요하다.

절삭력은 가공의 상태나 공구에 직접적인 영향을 줄 수 있는 것이므로, 일정하게 유지되는 것이 좋다. Kramer[2]는 경로간격 만큼의 절삭폭(radial depth)을 최소참여폭(minimal engagement)으로 정의하고, 양방향 경로에서 최소참여폭을 발견하여 이송속도를 조정하고 있다. 이 논문에서는 절삭폭을 슬롯(slot)과 최소참여폭으로 구분 짓고 있지만 좀 더 세분화될 필요가 있음을 밝히고 있다.

옵셋방식은 옵셋시 루프제거 문제로 인하여 로보스트한 시스템의 구현은 어려운 문제로 알려져 있다. Woodwark,[13] Preiss[14]는 "포켓팅은 매우 복잡한 상황이 많기 때문에 모든 문제점을 자동적으로 해결할 수 있는 시스템을 신뢰성 있게 구축하기는 매우 힘들다." 라고 언급하고 있다. Persson[4]은 bisector를 이용하여 포켓을 subarea로 구분 짓는 bisector skeleton을 가지고 옵셋을 하였고, Held[1]는 Voronoi diagram을 이용하여 pocket region을 볼록한 Voronoi polygon으로 구분하여 기존에 있던 옵셋시 루프제거의 어려움을 해결했다. 그러나 옵셋을 하는 것과 공구경로를 만드는 것과는 차이점이 있다. 예컨데 각각의 옵셋커브(offset curve) 연결 문제는 공구후퇴, 경로 최적화 문제를 고려해야하며, Held[7]는 근사맵(proximity map)을 이용하여 이러한 문제점을 해결하고 있다.

절삭력은 옵셋경로를 따라 움직이는 경우와 옵셋과 옵셋사이를 연결하는 경우로 구분 지어 이송속도를 보정함으로써 과다하게 걸리는 절삭력을 줄이고자 했지만, 공구가 옵셋경로를 따라 움직이는 동안 받는 절삭력은 다양하게 변할 수 있으므로 세분화된 이송속도 보정이 필요하다.

위에서 제시된 양방향, 옵셋 방식이외에도 공구의 가공경로를 수동으로 제시하는 수동교시 방법이 있다.[22] 공구간격으로 구분된 그리드 상에서 공구를 8가지 방향으로 움직여서 공구경로를 생성하지만 사용자가 일일이 지정해야 되는 번거로움과 생성된 경로의 최적화 문제가 있다. Bae[8]는 star-shaped pocket에 대해 그리드 상을 지역항해법(terrain navigation)을 이용하여 공

구경로를 생성하고 있고, 옵셋방식보다 짧은 경로가 생성됨을 보이고 있다.

본 논문에서는 아일랜드를 가지는 복잡한 포켓형상에 대해서 신경망을 이용한 새로운 공구경로생성과 피드맵(feed map)을 이용한 이송속도의 조절을 통해 공구후퇴, 경로 최적화, 절삭력 문제를 해결하여 일정한 절삭력을 유지하면서 빠르게 가공을 할 수 있는 효율적인 황삭가공 알고리듬을 제시한다.

2. 포켓가공의 TSP 모델과 신경망

포켓 가공경로를 생성하는 방법은 크게 기하학적 형상을 가지고 수식적 알고리듬에 의해서 접근하는 자동포켓가공 형태와 사용자가 공구의 경로를 교시하는 수동교시 형태가 있다. FANUC 시리즈 15MF의 머시닝 센터를 위한 대화형 자동 프로그래밍(conversational automatic programming function for machining center)[6]에서는 자유형상 포켓가공을 위한 고기능의 하나로 자유 편집 경로(random edit path)라는 수동교시방식을 제시하고 있다. 이 기능은 아일랜드가 많이 존재하는 경우에 황삭 가공경로 생성을 사용자에게 의존함으로써 사용자의 지적능력을 이용하여 복잡한 형상에 대한 경로 생성의 강건성과 효율성을 꾀하고 있다.

공구의 직경과 경로간격이 주어지게 되면 Fig. 1과 같이 그리드가 생성이 되고, 사용자는 그래픽화면상에서 공구로 간주되는 커서를 8가지 방향(+, ×)으로 움직이면서 경로를 생성할 수 있고, 경로의 시작점(approach point)과 끝점(escape point)에서는 가공조건을 지정하도록 되어 있다. Fig. 1에서 그리드는 공구가 포켓을 가공하기 위해서 반드시 지나가야할 가공

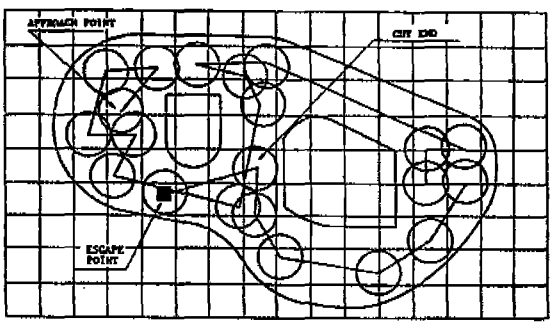


Fig. 1 그리드 상에서의 수동교시 방법

점으로 간주되며, 이러한 가공점들을 연결한 것이 포켓을 가공하기 위한 경로가 된다. 이 방식은 경로를 일일이 지정해주어야 된다는 불편함 뿐만아니라, 사용자의 지적능력에 경로의 효율성을 의존하고 있기 때문에 그리드 수가 많아지게 되면 효율적인 경로를 생성하는데 문제점이 있다.

포켓가공을 위한 가공점이 구해지면, 가공점을 연결한 것이 포켓가공경로가 됨을 알 수 있었고, 경로의 최적성을 부여하기 위해서 본 논문에서는 가공점들을 도시로 간주하여 도시들을 여행하는 TSP로 공구경로생성 문제를 모델링하였다.

TSP는 조합 최적화(combinatorial optimization) 문제로 일반적으로 알려져 있고, 문제의 크기가 커지게 되면 해를 찾는데 걸리는 시간이 기하급수적으로 증가함으로써, 최적해를 얻기 위해서는 계산상의 어려움이 있었고, 효율적으로 최적해에 근사하는 해를 구하는 휴리스틱 방법들이 많이 연구되었다. 최근 이런 문제점을 해결하는 방법으로 신경망을 이용한 방식이 대두되었고, 계산상의 문제점을 어느정도 극복하여 도시의 수가 증가해도 준최적화 해에 빨리 수렴하고 있다. 신경망을 이용하여 TSP를 푸는 방법으로는 Hopfield 와 Tank 모델에 의한 방법,[16] 설담금질(simulated annealing)에 의한 방법,[17] Elastic Net 에 의한 방법,[18] SOM(Self Organizing Map)에 의한 방법[19] 등이 있다.

TSP에 신경망을 처음으로 적용한 것은 Hopfield와 Tank가 1985년에 제시한 모델[16]로서, 도시의 수가 10인 경우에 상당히 좋은 결과를 보였고, 대상으로 했던 크기는 30정도 였다. 그러나 도시의 수가 30을 넘게 되면 불합리한(infeasible) 해로 네트웍이 수렴될 수 있고, 1988년 Wilson과 Pewley[21]는 이러한 점을 실제로 증명을 하였다. 설담금질 방법[17]에서는 온도를 heating, cooling, slow cooling등으로 조절하는 담금질 스케줄을 통해서 네트웍이 불합리한 해로 수렴하는 가능성을 줄이고 있지만 수렴시간이 오래 걸리는 단점이 있다.

Hopfield의 모델보다 조금 발전된 것이 1987년 Durbin, Willshaw의 Elastic Net 방법이고 도시의 수가 증가를 해도 상당히 좋은 해를 보여주고 있다. 기본적인 개념은 고무줄에 있는 예비 노드들을 거리를 최소화하면서 평면상에 있는 도시들로 사상(mapping)된다. Kohonen[20]이 1894에 제시한 SOM은 경쟁학습(competitive learning)의 원리를 약간 변형한 비감

독 학습(unsupervised learning) 알고리듬으로 형상의 특징을 묘사하는 형상지도(feature map)이다. 1988년에 Angeniol[19]은 SOM을 TSP에 적용하여 1000개의 도시를 대상으로 하여 해를 찾는데 20분 정도가 소요되었다. 본 논문에서는 형상을 묘사하는 성질과 빠른 수렴시간의 특징을 갖는 Angeniol의 SOM을 이용한 방법을 사용하여 TSP를 해결하고자 한다.

3. 가공점의 생성

공구의 경로간격이 주어지게 되면 주어진 형상을 감싸는 그리드맵을 생성할 수 있고, 그리드점들중에 포켓내부에 있는 점들을 선정하고, 선정된 포켓내부점 중에서 과절삭(overcut)을 일으키는 윤곽근처의 점들을 제거하면 가공점을 생성할 수 있다.

3.1 포켓내부점 생성

그리드맵상의 점들 중에서 포켓 내부에 있는 점들을 찾아내는 방법으로는 Fig. 2와 같이 스캔라인을 이용한 ray tracing 방법을 사용하고 있고, 다각형 형상의 경우 포켓내부점 판정 알고리듬은 다음과 같다.

[포켓 내부점 판정 알고리듬]

```
L:z의 왼쪽에서 스캔라인과 윤곽이 만나는 교점의 수
  Start L=0;
  for i=1 until N do {
    if(edge(i) is not horizontal) then
       if(edge(i) intersects l to the left of z)
       then L=L+1;
  }
  if(L is odd) then z is internal
  else z is external
  End.
```

위의 알고리듬에서는 교점의 수가 홀수이면 포켓내부로 판정하고, 짝수이면 포켓외부로 판정을 하고 있다. 그러나 스캔라인이 윤곽을 구성하는 선분(edge)들의 교차점 또는 수평선분들과 만나는 경우에는 위 알고리듬 적용이 어렵다.

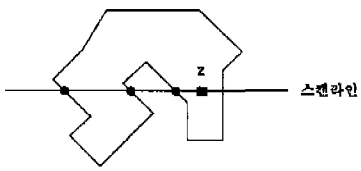


Fig. 2 다각형에서 포켓내부점 판정

스캔라인이 선분들의 교차점과 만나는 경우는 크게 2가지 경우로 구분된다. Fig. 3(a)의 경우와 같이 스캔라인과 만나는 두 선분이 모두 스캔라인 위 또는 아래에 있는 경우에는 두 선분의 교차점에서 스캔라인과 만나지만 교점이 없는 것으로 간주되어야 하고, (b)와 같이 스캔라인의 위와 아래로 선분이 존재하게 되면 교점이 있는 것으로 간주된다. 윤곽을 구성하는 선분이외의 아크에도 이러한 규칙이 적용된다.

스캔라인과 수평선이 만나는 경우에 무한개의 교점이 존재하지만 Fig. 4(a)의 경우는 수평선의 끝점중에 하나의 점을 교점으로 간주하고, (b)의 경우는 두 끝점을 모두 교점으로 간주한다. 따라서 스캔라인과 선분들과의 교점의 수를 계산하여 홀수면 포켓 내부로, 짝수면 포켓 외부로 판정을 하게 된다. 단, 위의 방식을 적용하게 되면 수평선 상에 있는 점들이 포켓내부로 판정이 될 수 있는 경우가 발생할 수 있다. 공구가 윤곽 위의

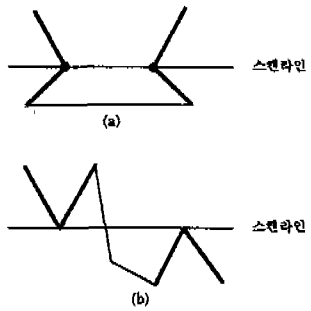


Fig. 3 스캔라인과 선분의 교차점이 만나는 경우

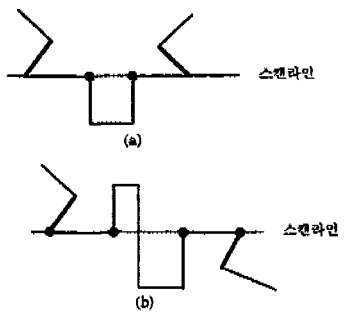


Fig. 4 스캔라인과 수평 선분이 만나는 경우

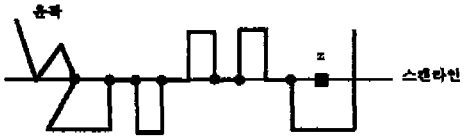


Fig. 5 윤곽과 스캔라인이 만나는 경우 교점판정

점에 오게 되면 과절삭이 발생하므로, 수평선상에 있는 점들은 포켓 내부점에서 제외된다.

따라서 스캔라인이 윤곽과 만나는 경우를 선분의 교차점과 수평선분으로 구분하여 위와 같은 교점판정 룰을 사용하여 ray tracing 방법에서와 같이 교점의 수에 따라 포켓 내부, 외부를 판정하고 있다. Fig. 5에서 선분의 교차점 또는 수평선분과 스캔라인이 만나는 경우에 위의 방법으로 교점을 판정하게 되면 Z의 왼쪽에서 교점수가 7개 이므로 Z는 포켓 내부점이 된다. 위에서 제시한 방법이외에도 스캔라인을 위 또는 아래로 조금 움직여서 위와 같은 경우를 피하는 방법도 있다.

3.2 가공점 생성

포켓내부에 있는 점들 중에 외부 윤곽이나 아일랜드에 근접한 점들은 공구의 볼롬이 있기 때문에 과절삭을 일으킬 수 있으므로 가공점에서 제외된다. 공구를 원이라 하면 과절삭 여부는 윤곽을 구성하는 선분(line segment), 아크와 원과의 교차 문제이다. Fig. 6(a)의 경우 선분과 원의 중심과의 최소거리(d)가 공구반경

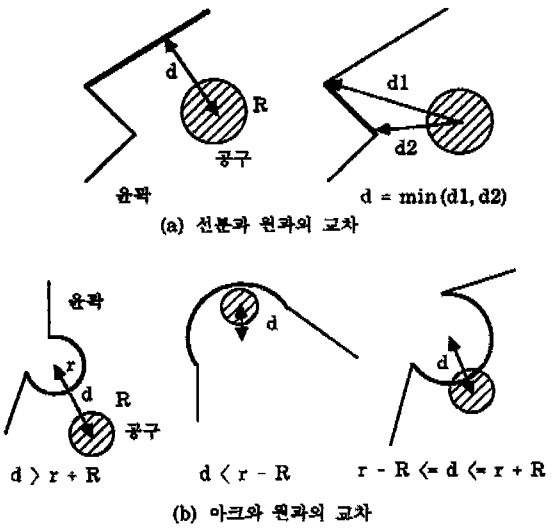


Fig. 6 원과 선분, 아크와의 교차

(R)보다 작으면 과절삭으로 판정하고, (b)의 경우 아크와 원이 접하거나 두 점에서 만나는 경우에 교점이 아크상에 있으면 과절삭으로 판정한다. 만약 포켓 내부점이 과절삭으로 판정되면 가공점에서 제외된다.

4. SOM을 이용한 경로생성

4.1 TSP에의 SOM 적용

Angeniol이 제시한 SOM 알고리듬의 기본적인 개념은 다음과 같다. 도시가 있는 평면상에 링이 있다고 보고, Fig. 7과 같이 링에 있는 노드들이 진화하여 도시로 움직이면서 도시와 사상이 되면, 연결된 노드들의 경로가 도시들을 여행하는 TSP의 근사된 최단 여행경로를 생성한다.

알고리듬에서 도시는 1에서 M까지, 노드는 1에서 N까지 순번 되어지고, 도시의 위치좌표는 (X_1^i, X_2^i), 노드의 위치좌표는 (C_1^i, C_2^i)로 표기된다. 링에 있는 노드들이 도시로 움직이기 위해서 도시 i는 식(1)과 같이 가장 가까운 노드 j_c를 승자노드(winner node)로 선택하고, 승자노드와 이웃노드들을 식(2)과 같이 도시방향으로 움직이며, 움직이는 거리는 함수f(G, n)을 따른다. 여기서 G는 gain parameter이고, n은 링에서 승자와 이웃노드 사이의 최소거리를 의미한다. G는 에 의해서 감소되며, G가 0에 가까워지면 승자노드 jc만이 도시 i로 움직이게 되고. 결국 링에 있는 노드와 평면상에 있는 도시가 일치된다.

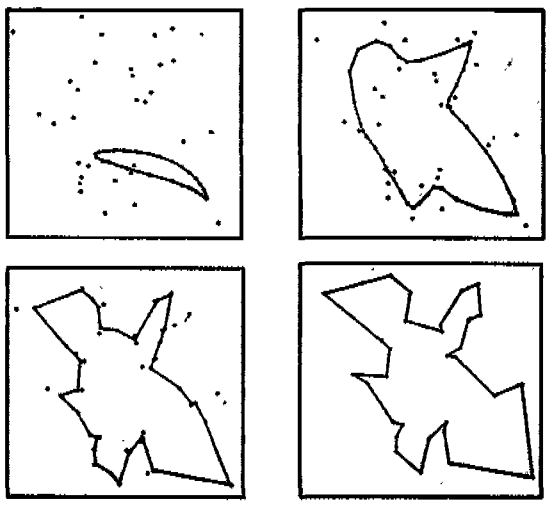


Fig. 7 Hopfield Tank가 사용한 30개의 도시를 대상으로 Angenniol의 SOM에 의한 방법에서의 링의 진화[19]

[SOM 알고리듬]
평면상에 위치한 도시의 좌표: (X_1^i, X_2^i)
링에 있는 노드의 좌표: (C_1^i, C_2^i)라 하면,
Step 1. city i에 가장 가까운 node j_c를 찾는다.

$$V_j = (X_1^i - C_1^i) + (X_2^i - C_2^i)$$
$$V_{j_c} = \min V_j \qquad (1)$$

Step 2. node j_c와 그 이웃노드들을 city i 방향으로 움직인다.

$$C_k^j = C_k^j + f(G, n) * (X_k^i - C_k^j) \qquad (2)$$
$$f(G, n) = (1/\sqrt{2}) * \exp(-n^2/G^2)$$
$$n = inf(j - j_c(\mathrm{mod}N),\ j_c - j(\mathrm{mod}N))$$
$$G = (1-\alpha) * G$$

위에서 제시된 알고리듬에서 parameter는 α이며, 에 따라 다른 해를 생성한다. α의 값이 크면 해를 찾는데 걸리는 시간이 빠르며, 실험을 통하여 값이 길이에 민감하지 않다는 결과를 보이고 있다. 그러나 가공경로라는 측면에서 α값에 따라 생성된 해는 다른 결과를 보일 수 있고, 적정 α를 자동적으로 선정하는 문제도 중요하다. 절삭력측면에서는 형성된 경로가 일정한 패턴을 갖도록 하는 것이 필요하며, 이러한 문제점을 해결하기 위해서 본 논문에서는 변형된 SOM 알고리듬을 제시한다.

4.2 α가 가공경로에 미치는 영향

신경망에서는 α의 변화가 경로길이를 향상시키는데 별로 영향을 주지 않고, 0.1 - 0.4의 가 좋은 해를 생성한다는 결론을 주었다. 그러나, 가공에 있어서는 경로길이도 중요하지만 형성된 경로자체가 일정한 절삭력를 유지하는 것이 좋다. Fig. 8은 사각포켓을 대상으로 α가 0.1, 0.2, 0.3일 경우에 생성된 가공경로를 보여주고 있고, 실험을 통해 0.05 - 0.2인 경우에 랜덤경로 보다는 안정된 경로형태를 취함을 알 수 있었다.

4.3 사상을 통한 α의 선정

경로길이나 경로형태를 최적으로 만드는 α를 선정하기 위해서는 많은 실험을 필요로 한다. α가 경로에 영향을 미치는 이유를 생각해 보면, α는 노드가 도시방향으로 이동하는 거리와 밀접한 관계가 있고, 이동량에 따라 지역최소점(local mimimum)에 빠져서 안 움직이는 경우와 전역최소점(global minimum)을 지나가는 경우가 생길 수 있다. 따라서 네트웍이 해를 찾아가는 동안 일정한 α를 유지하는 것은 바람직하지 못하다. 초기에는 노드들이 도시방향으로 자유롭게 움직일 필요가 있고, 노드가 도시에 가까워지면 움직이는 거리 또한 줄어들어야 한다. 네트웍이 지역최소점에 빠지는 것을 해결하기 위한 방법으로 설담금질 방법이 있다. 설담금질 방법에서는 온도변화를 이용하지만, SOM에서는 도시와 노드의 사상정도를 가지고 α를 설정할 수 있다. 사상정도는 식(3)과 같이 표현될 수 있다.

$$K = \sqrt{\sum_i (X_i - C^*(X_i))^2} \qquad (3)$$

여기서 X는 도시의 위치, C*(X)는 도시에서 가장 가까운 노드의 위치를 의미하며, 사상정도치인 K는 노드가 얼마나 도시에 근접했는 가를 표시하고 있다. Boltzmann Machine에서 담금질 온도변화는 식(4)와 같이 시간에 따라 감소를 하게 되지만, SOM에서는 노드와 도시가 근접하게 되어 K값이 작아짐에 따라 식(5)와 같이 α는 증가한다.

$$T(tn) = \frac{T_0}{\log(1+tn)} \qquad (4)$$

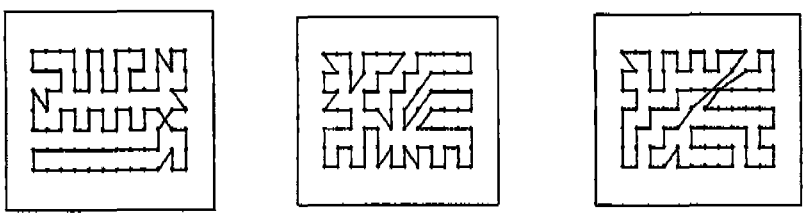


Fig. 8 α값에 따른 공구경로

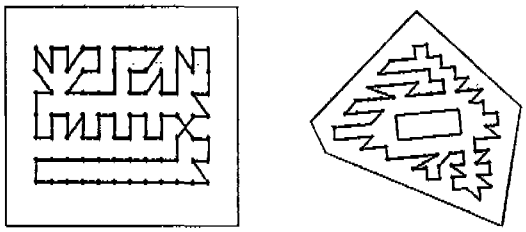


Fig. 9 사상을 이용한 공구경로

where T_0 = 상수, tn = 시간변수

$$\alpha = C * \log(1 + \exp(-K)) \quad (5)$$

where C = 상수

사상정도에 따라 α를 조정하여 실험해 본 결과, 대부분의 경우에 Fig. 9과 같이 안정된 경로형태를 보여주며, α를 고정시키는 경우보다 해를 찾는데 걸리는 시간이 줄어든다. 네트웍이 진화해 가면서 노드들이 도시에 근접하게 될 때 α는 지역경로(local path)를 형성하는데 큰 영향을 주게 되며, 사상을 이용하는 경우에는 지역경로 형태가 꼬이지 않고 잘 나오는 것 같다.

4.4 가중치를 준 경쟁학습을 통한 노드의 선정

SOM 알고리듬에서 승자를 선정하는 경쟁방법은 식(1)과 같이 도시와 노드의 거리를 기준으로 하고, 승자로 선택된 노드와 이웃노드는 도시의 방향으로 움직이게 된다. 절삭시 공구가 받는 힘을 일정하게 유지하기 위해서는 노드들이 일정한 방향성을 가져야 되는데, Euclidean distance를 그대로 적용을 해서 승자를 택하게 되면, 수평, 수직 방향의 노드들이 자유롭게 선정이 될 것이다. 수평방향으로 노드들을 조정하기 위해서는 수평방향의 노드들이 수직방향에 있는 노드들보다 승자로 선택될 수 있어야 된다. 이를 위한 한 방법으로 본 연구에서는 수직(또는 수평) 방향의 거리에 가중치를 줌으로써 수직(또는 수평) 방향의 노드가 선택될 경우가 높아지도록 한다.

식(6)에서 가중치(weight)는 수평방향 조정도를 의미하고, 가중치가 주어지게 되면 수평(수직)방향으로 편향된 공구경로가 생성됨을 알 수 있다. (Fig. 10) 가중치가 커지게 되면 편향정도는 좋지만, 에 따라 상한치가 존재하고, 현재까지 실험결과로는 α가 0.1인 경우 가중치를 4이상 주게 되면 생성된 경로 자체가 불합리한 경우가 발생한다. 이러한 특성을 갖는 가중치 인자

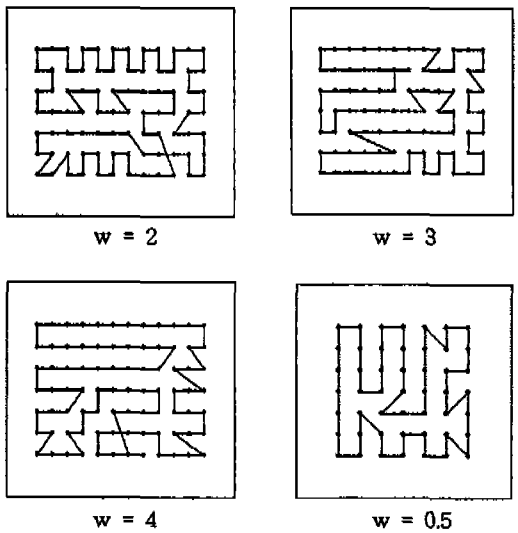


Fig. 10 가중치를 준 경로생성

는 α가 작아짐에 따라 상한치가 커지는 경향을 보인다.

$$V_j = (X_1^i - C_1^j)^2 + weight(X_2^i - C_2^j)^2 \quad (6)$$

4.5 수정된 SOM 알고리듬

평면상에 위치한 도시의 좌표: (X_1^i, X_2^i)

링에 있는 노드의 좌표: (C_1^i, C_2^i)라 하면,

Step 1. city i에 가장 가까운 j_c를 찾는다.

$$V_j = (X_1^i - C_1^j)^2 + w * (X_2^i - C_2^j)^2$$

$$V_{j_c} = \min V_j$$

Step 2. j_c와 그 이웃노드들을 city i방향으로 움직인다.

$$C_k^j = C_k^j + f(G, n) * (X_k^i - C_k^j)$$

$$f(G, n) = (1/\sqrt{2}) * \exp(-n^2/G^2)$$

$$n = inf(j - j_c(\mathrm{mod}N), j_c - j(\mathrm{mod}N))$$

if mapping node is selected then

$$K = \sqrt{\sum_i (X_i - C^*(X_i))^2}$$

$$\alpha = C * \log(1 + \exp(-K))$$

$$G = (1 - \alpha) * G$$

위에서 제시된 알고리듬을 이용하여 사용자는 C, w를 조정하면서 포켓형상에 맞는 경로를 디자인하여 공

구경로를 산출하게 된다.

5. 공구경로의 Smoothing

5.1 주평활 (Strong Smoothing)

공구경로가 외부윤곽이나 아일랜드를 지나는 경우가 발생할 수 있고, 과절삭을 막기 위해서 공구후퇴가 발생하게 되며, 다음 지점에서 공구가 진입을 하게 된다. 과절삭을 일으키는 문제는 선분을 따라 이동하는 공구의 볼륨과 윤곽을 구성하는 선분, 아크와의 교차문제이다. Fig. 11을 보면 공구볼륨과 선분의 교차는 두 선분이 만나 과절삭이 생기는 a)의 경우와 두 선분의 최소거리를 구해 반경보다 작은 경우에 과절삭을 판정하는 b)의 경우로 구분된다. 공구의 볼륨과 아크와의 교차는 두 원이 만나지 않아 과절삭이 발생하지 않는 c), d)의 경우와 공구반경안에 아크가 포함되어 과절삭이 일어나는 e)의 경우, 교점이 존재하는 할 때 공구경로의 옵셋 선분과 아크가 만나는 경우에만 과절삭이 발생하는 f)의 경우로 구분된다.

주평활작업은 알고리즘내에서 자동적으로 수행될 수 있으며, 현재 구현된 방식은, 과절삭이 탐색되는 구간에서 구간끝점에 드릴링동작을 NC 코드의 서두에 삽입하고, 이후의 밀링가공에서는 구간시작점에서 "공구뜨기" (급송이송), 구간끝점으로의 "다음점이송" (급송이송) 및 "공구하강" (피드이송) 코드를 삽입하도록 설계되어 있다.

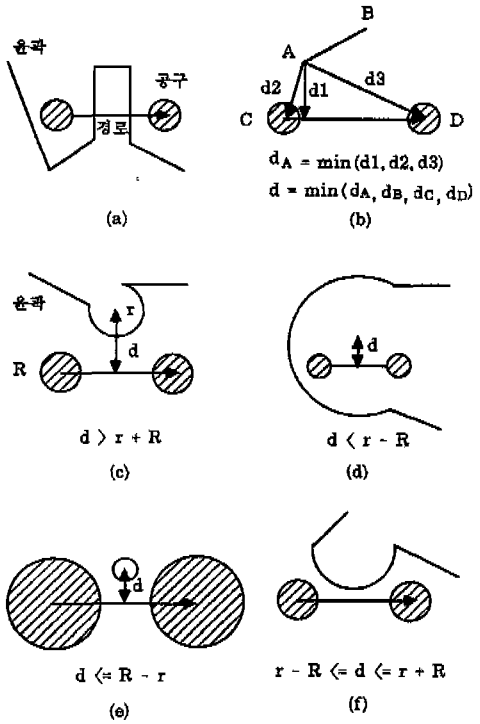


Fig. 11 주평활이 일어나는 경우

5.2 부평활 (Weak Smoothing)

Fig. 12(a)와 같이 경로가 꼬이거나 연이은 사선의 조합으로 된 경우에는 공구가 받는 절삭력의 변화가 다소 크므로, 전체 공구경로에서 이런 경우를 찾아내서 국부적으로 수직과 수평선의 조합으로 수정을 하면 경로길이도 줄어들고 공구가 받는 절삭력의 변화도 줄어들게 된다. 현재 구현된 방식은 Fig. 12(a)에 제시된 세가지 타입에 대해 Fig. 12(b)으로 자동적으로 평활시킨다. 그리드맵상에서 두점간의 직선이동경로를 $Pi = (S_{i,1}, S_{i,2}, E_{i,1}, E_{i,2})$라 하고 그리드간격을 g라 하면, 부평활을 거침으로서 경로길이는 타입 1, 2의 경우 $2g(1 + \sqrt{2})$에서 4g로 $2g(\sqrt{2} - 1)$만큼 줄어들며, 타입 3의 경우 $2g\sqrt{2}$에서 2g로 $2g(\sqrt{2} - 1)$만큼 줄어든다. 타입 1, 2를 평활하는 알고리즘과 타입 3을 평활하는 알고리즘은 다음과 같다.

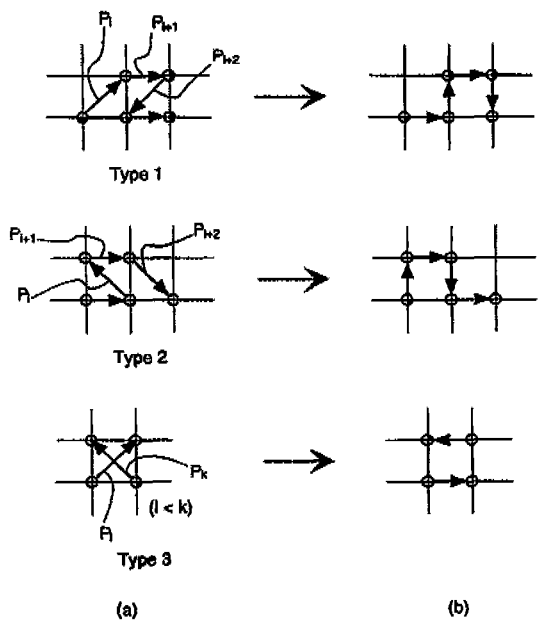


Fig. 12 부평활이 일어나는 경우

〈알고리즘 - 타입 1, 2 평활알고리즘〉

1. Read Pi, i∈[1:Q], and let i = 0.
2. i = i + 1
3. if i<L, stop, where L = Q + 1, if the entire path P is closed, otherwise L = Q - 2.
4. Check if Pi is a diagonal path:if $|S_{i,k} - E_{i,k}| = 1$, for all k∈(1:2), then goto Step 5, otherwise goto Step 2.
5. Check if P_{i+2} is a diagonal path and parallel to P_i:
   (Henceforth, if P is closed, segment index = MOD(segment, Q)
   if ($|S_{i,k} - E_{i+2,k}| = 1$) and ($|S_{i,k} - S_{i+2,k}| = 1$) for all k∈(1:2), goto Step 6, else goto Step 2.
6. Pattern matching
   if $(\Sigma_k |S_{i,k} - E_{i+1,k}| = 3)$ and $(\Sigma_k |S_{i,k} - E_{i+3,k}| = 1)$, goto Step 7, else if $(\Sigma_k |S_{i,k} - E_{i+1,k}| = 1)$ and $(\Sigma_k |S_{i+1,k} - E_{i-1,k}| = 1)$ then goto Step 8, else goto Step 2.
7. Smoothing for Type 1:
   $P_i = (S_{i,1}, S_{i,2}, E_{i+2,1}, E_{i+2,2})$,
   $P_{i+1} = (S_{i+3,1}, S_{i,1}, E_{i,1}, E_{i,2})$,
   $P_{i+2} = (S_{i,1}, S_{i,2}, E_{i,1}, E_{i,2})$,
   $P_{i+3} = (S_{i+2,1}, S_{i+2,2}, E_{i+3,1}, E_{i+3,2})$,
   goto Step 2.
8. Smoothing for Type 2:
   $P_{i-1} = (S_{i-1,1}, S_{i-1,2}, E_{i,1}, E_{i,2})$,
   $P_i = (S_{i+1,1}, S_{i+1,2}, E_{i+1,1}, E_{i+1,2})$,
   $P_{i+1} = (S_{i+2,1}, S_{i+2,2}, E_{i-1,1}, E_{i-1,2})$,
   $P_{i+2} = (S_{i,1}, S_{i,2}, E_{i+2,1}, E_{i+2,2})$,
   goto Step 2.

〈알고리즘 - 타입 3 평활알고리즘〉

1. Read P_i, i∈(1:Q), and let i = 0.
2. i = i + 1
3. if i>Q, stop.
4. Check if P_i is a diagonal path:
   if $|S_{i,k} - E_{i,k}| = 1$, for all k∈(1:2), then goto Step 5, otherwise goto Step 2.
5. Find P_k, where k>i+1, which is diagonal path and crossing P_i:
   for k = i+2 to Q {
   if $P_k = (S_{i,1}, E_{i,2}, E_{i,1}, S_{i,2})$ or $P_k = (E_{i,1}, S_{i,2}, S_{i,1}, E_{i,2})$ goto Step 6
   }
   goto Step 2.
7. Smoothing for Type 3:
   $P_i = (S_{i,1}, S_{i,2}, E_{k,1}, E_{k,2})$,
   $P_k = (E_{i,1}, E_{i,2}, E_{k,1}, E_{k,2})$,
   goto Step 2.

6. 절삭력에 따른 이송속도 보정

6.1 소재제거율을 이용한 절삭력 근사

Smith, Tlusty[12]는 평균절삭력(F)를 식(7)과 같이 정의하였다. 공구와 피삭재의 특징에 따라 P_{sv}, v가 결정되므로 공구가 받는 평균절삭력은 소재제거율(MRR:Material Removal Rate)에 비례함을 알 수 있다. MRR은 식(8)과 같이 정의되며, 절삭깊이가 일정하다고 가정하면 절삭폭에 따른 이송속도 조정으로 일정한 MRR을 유지할 수 있다. MRR을 일정하게 유지함으로서 공구가 받는 절삭력 또한 일정하게 유지할 수 있다.

$$F = \frac{P_{sp}(MRR)}{v} \tag{7}$$

where F = cutting force
P_{sp} = specific power
v = cutting speed

$$\mathbf{MRR = bamcn} \tag{8}$$

where b = axial depth of cut
a = radial depth of cut
m = number of teeth on the utter
c = chip load(feed per tooth)
n = spindle speed

6.2 피드맵을 통한 MRR 근사

6.1절에서 절삭력은 MRR에 비례함을 알 수 있었고, MRR은 절삭폭과 이송속도의 조절로 일정하게 유지될 수 있다. 일반적으로 양방향 방식에서 절삭폭은 수식적으로 계산되어지며, 슬롯과 경로간격에 해당되는 절삭폭에 대해 각각 이송속도를 조절해 줌으로써 절삭

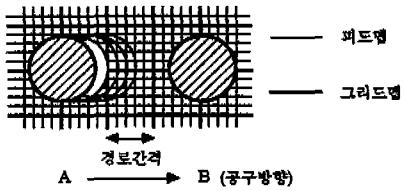


Fig. 13 피드맵상에서의 절삭폭 근사

력을 어느정도 유지하고 있으나, 좀 더 세분화될 필요가 있음을 밝히고 있다.[2]

신경망을 이용한 경로는 규칙성이 없기 때문에 수식적으로 절삭폭을 구할 수 없으므로 공작물을 작게 나누어서 형성된 피드맵(feed map)을 가지고 근사된 절삭폭을 구하는 방식을 사용하고 있다. 초기의 피드맵은 피삭재가 가공이 되지 않았기 때문에 0 상태의 맵으로 존재하다가 공구가 경로를 따라 움직이는 동안 공구의 볼륨에 해당되는 피드맵 상의 점들은 가공이 된 1의 상태로 변한다. 따라서 피드맵은 공구가 경로를 따라 움직이는 동안 공작물의 가공상태를 근사하게 묘사한다.

가공점을 생성하기 위해서 사용자가 입력한 공구의 경로간격을 가지고 그리드맵을 만들었고, 그리드맵으로는 절삭폭을 근사할 수 없기 때문에 더 작은 단위의 피드맵을 형성하였다. 그리드맵상에서 형성된 공구경로에 의해 공구는 Fig. 13과 같이 A에서 B로 움직이지만 피드맵상에서 한칸씩 움직여서 가는 것으로 간주된다. 가공점 A, B의 거리가 경로간격일 수도 있지만 경로간격보다 클 수 있으므로, 만약 경로간격보다 큰 경우에는 다시 경로간격 만큼씩 구간을 나누어, 각 구간에서의 절삭폭을 구하게 된다. 절삭폭은 공구가 피드맵을 한칸씩 움직이는 동안 새롭게 포함되는 그리드수로 근사될 수 있다. 한구간은 몇개의 피드맵점으로 나누어져 있고, 각각의 피드맵점에서 공구가 받는 절삭폭은 변할 수 있으므로, 구간에서의 최대 그리드수를 절삭폭으로 간주한다. 최대 그리드수를 절삭폭으로 선택한 이유는 공구의 휨 또는 파손없이 안전하게 가공을 하기 위함이다. 위와 같은 방식으로 절삭폭을 구하며, 일정한 범위의 MRR을 유지하기 위해서 이송속도를 보정하게 된다.

6.3 이송속도보정 알고리듬

각 구간마다의 절삭폭이 피드맵에 의해 근사되면 식(9)에 의해 이송속도(f)를 계산할 수 있다.

피드맵의 그리드간격:n,

Slot에 해당하는 이송속도:f_1,

1/2 Slot 에 해당하는 이송속도:f_2,

경로간격에서 최대의 그리드수:n_3이라 하면,

$$f = f_2 + \frac{(f_2 - f_1)}{(n_2 - n_1)} * (n_3 - n_2) \qquad (9)$$

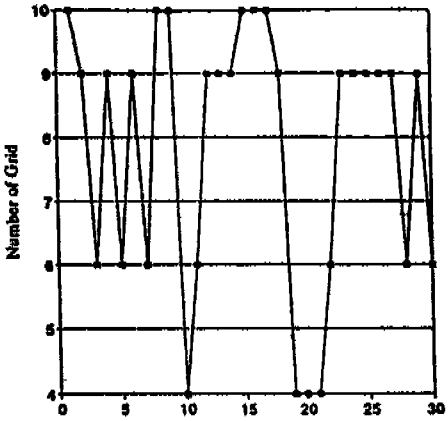
where n_1 = 공구직경/n, n_2 = 공구반경/n

식(9)는 절삭폭이 공구직경과 공구반경일 경우에 해당되는 이송속도를 가지고, 미소 절삭폭인 피드맵 그리드 간격에 대한 이송속도의 평균 변화량을 계산하여, 새롭게 포함되는 그리수에 평균변화량을 적용하여 피드를 보정한다. Fig. 14는 신경망을 이용해서 나온 공구경로에 피드맵을 이용하여 절삭력을 일정범위에 들도록 보정한 결과를 보이고 있다. 공구경로는 Fig. 17의 경로중 30개 구간을 대상으로 하고, 공구직경 = 40mm, 피드맵 그리드간격 = 4mm, n_1 = 10, f_1 = 50mm/min, n_2 = 5, f_2 = 100mm/min으로 가정한 결과이다. Fig. 14(a)은 30개 구간에서 피드맵을 통해서 얻어진 그리드수를 도시한 것으로, 그리드수가 4-10으로 변화한다. 그리드수 = 10은 공구직경과 같은 절삭폭을 의미하므로, 공구경로상에서 절삭폭의 변화는 공구직경의 40-100% 임을 알 수 있다. (a)의 그리드수를 식(9)에 적용하여 이송속도를 보정한 것이 (b)이며, 절삭폭이 증가하면 이송속도는 감소되는 것을 볼 수 있다. (c)는 절삭깊이를 1mm로 가정하고, 절삭폭과 이송속도를 곱해서 나온 MRR을 도시한 것으로, ■은 피드맵을 이용해서 이송속도를 보정한 것이고, + 는 slot 이송속도를 계속 유지한 것이다. 결과적으로 피드맵을 이용해서 이송속도를 절삭폭(그리드수)에 따라 보정함으로써 MRR을 일정한 범위로 유지할 수 있고, 일정 이송속도로 가공하는 것보다 가공시간을 줄일 수 있다. (c)는 MRR을 이론적으로 계산한 값이며 실험을 통한 검증이 필요하다.

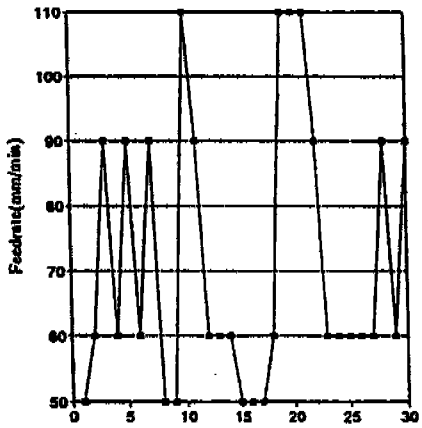
7. 시뮬레이션과 및 분석

7.1. 간단한 형상에 뉴럴을 적용한 결과

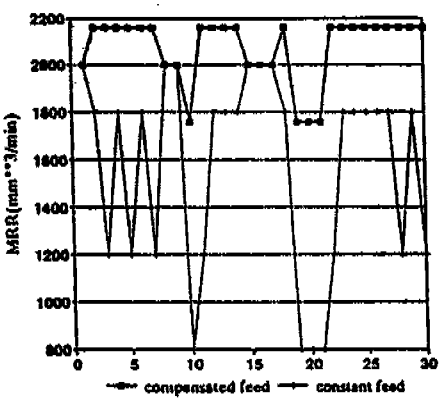
간단한 형상인 경우에는 양방향 가공방식을 사용하는 것이 효율적인 공구경로를 산출한다고 알려져 있다. 그러나, Fig. 15의 경우 외부윤곽은 사각형의 간단한 형상이지만 내부에 4개의 원모양의 아일랜드가 존재함으로해서 단일 가공영역이 분할 되어진다. Fig. 15와 같은 형상에 양방향 방식을 적용해본 결과, 아일랜드가 없으면 공구가 뜨지 않고 한번에 가공할 수 있은데 반



(a) 경로구간에서의 그리드수 변화



(b) 경로구간에서의 이송속도 변화



(c) 경로구간에서의 MRR 변화

Fig. 14 이송속도 보정후의 MRR 변화

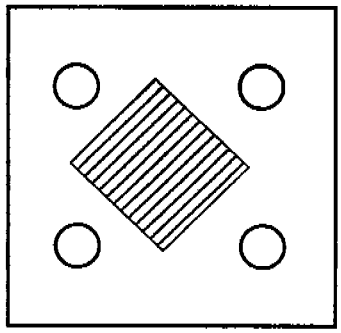


Fig. 15 아일랜드에 의한 단일 가공영역의 분할

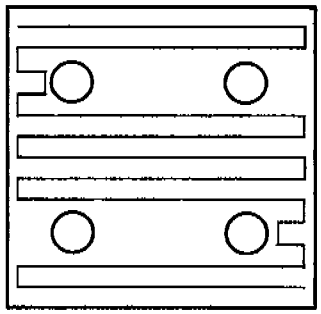


Fig. 16 아일랜드가 있는 간단한 형상에서의 양방향 가공경로

해, 간단한 아일랜드의 존재로 인하여 Fig. 16과 같이 단일 컷으로 가공을 할 수 없으며, 여러번의 공구후퇴가 발생하게 된다. 이런 문제점을 해결하고자 아일랜드를 우회하는 경로를 삽입하여 공구후퇴 수를 줄이고자 하고 있지만, 기본적인 양방향 알고리듬은 공구후퇴를 피할 수 없다. 옵셋방식을 적용을 하면 Fig. 15의 빗금친 영역에서 4개의 원이 만나므로 루프를 제거하여 생성된 옵셋경로는 가공을 위한 효율적인 경로라고 볼 수 없다. 신경망을 이용하여 공구경로를 산출하게 되면 아일랜드로 인해 발생했던 단일 가공의 어려움을 해결해 주고 있음을 Fig. 17를 통해 알 수 있다. SOM은 형상지도(feature map)라는 특성을 갖으면서, 지능적으로 경로를 산출하기 때문에 수식적으로 접근해서 풀기 어려운 문제점을 쉽게 해결해 주는 장점이 있다. Fig. 18은 외부윤곽은 원이고 아일랜드가 복잡한 형상에 신경망을 적용한 결과를 보여주고 있다. 음영이 진 부분은 실제로 공구가 가공한 영역을 도시한 것이다. 물론 순수한 황삭가공 경로이기 때문에 미절삭되는 부분이 남게 되는데 이러한 부분은 윤곽을 따라 중삭을 거치게

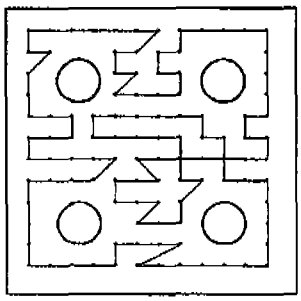


Fig. 17 아일랜드가 있는 간단한 형상에 신경망을 이용한 공구경로(α = 0.1, w = 4)

Fig. 18 복잡한 윤곽을 갖고 있는 아일랜드에서의 공구경로 (α = 0.2, w = 1)

되면 정삭여유만을 남길 수 있다.

7.2 Monotonous Area 에서의 뉴럴경로

Fig. 19와 같이 가공영역이 분할 되어질 수 밖에 없는 형상을 Held[1]는 Monotonous Area 라고 정의하고 있고, Voronoi diagram을 사용하여 이런 영역을 구분해 주고 있다. 그러나, 신경망을 사용하게 되면 공구후퇴를 줄이면서 가공을 할 수가 있다. 만약 이런 형상에 양방향 방식을 적용하게 되면 많은 공구후퇴 발생할 것이고, 옵셋 방식을 적용한다고 해도 각각의 옵셋 커브를 이어주는 문제점이 발생하게 된다. Fig. 19에서 점선은 공구후퇴가 발생하여 움직이는 부분이고, 가공영역이 분할되는 지역을 가공한 후에 다른 가공지역으로 경로가 이어짐을 볼 수 있다. 즉, 가공점과 신경망을 이용한 해법을 통해서 공구후퇴 및 경로연결 문제점이 해결됨을 알 수 있다.

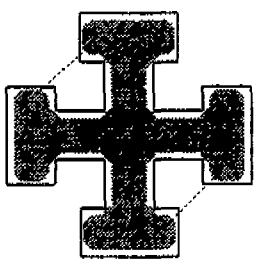


Fig. 19 Monotonous Area에서의 뉴럴경로 (α = 0.1, w = 1)

7.3 복잡한 형상에 뉴럴을 적용한 결과

형상이 복잡해 지면 경로 최적화 측면보다는 경로생성의 강건성이 더 중요시 된다. Fig. 20과 같이 8개의 아일랜드가 존재하는 형상에 뉴럴을 적용해본 결과, 형상의 복잡도에 상관없이 로보스트(robust)하게 공구경로를 생성함을 알 수 있고, 생성된 공구경로는 길이나 공구후퇴, 또는 실제 가공을 위한 경로의 측면에서 좋은 경로임을 알 수 있다.

7.4 가공실험

신경망에 의한 공구경로는 일반적으로 규칙성이 없기 때문에 이송속도의 보정 과정이 필수적이며, 제안된 방식에서는 두개의 입력피드를 기준으로 하여 경로구간별 보정을 한다. 제시된 보정알고리즘의 유효성을 검증하기 위해 실제의 가공실험을 수행하였다. Fig. 21(a)의 형상을 황삭가공한 결과 Fig. 21(b)와 같이 나타났다. 실험에서는 절삭력을 측정하지는 았았으나, 전반적으로 균일한 절삭소음을 감안할 때, 신경망 황삭경로의 피드 조정후의 집행은 무리가 없다고 판단된다. 구체적인 실험조건 및 분석결과는 다음과 같다.

〈실험조건〉

가공기계:3축, 2마력 CNC기계(Bridgeport Interact II 모델)

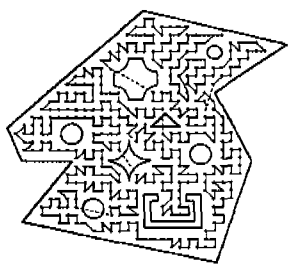
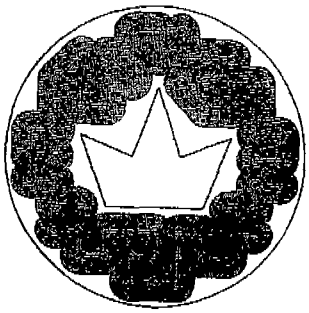
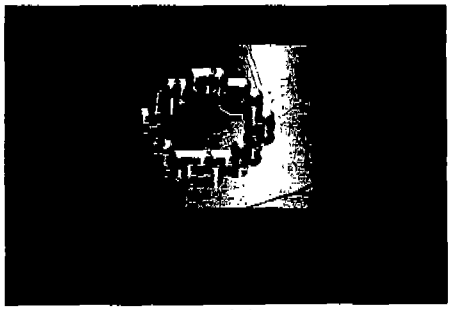


Fig. 20 복잡한 형상에서의 뉴럴경로 (α = 0.1, w = 1)



(a) 가공형상 및 시뮬레이션



(b) 가공시편

Fig. 21 가공실험 결과

사용공구:고속도강, 직경 15 mm, 4날 엔드밀

공작물재질:알루미늄

입력피드:슬롯이송속도 f_1 = 50 mm/min, 반슬롯 이송속도 f_2 = 100 mm/min

〈분석결과〉

생성된 NC코드에서, 이송속도는 50 - 120 mm/min으로 변화되었는 바, 고정속도(70 mm/min)를 사용하는 것에 비해 가공시간은 줄어든다. 아울러, 본 예제의 경우 산출된 경로는 공구뜸이 없는 연속경로로서, 공구뜸이 존재하는 기존의 알고리즘 (예:지그재그 패턴)에 비해 가공시간의 단축효과 (선드릴링 작업, 공구교환 시간 등)는 더욱 크다고 볼 수 있다.

8. 결론 및 추후연구

기존의 공구생성 방식을 사용하면 일반적으로 간단한 형상은 양방향 방식, 복잡한 형상은 옵셋방식이 좋은 경로를 생성한다고 알려져 있다. 그러나 기존방식은 특징형상 및 복잡한 형상에 대해서 좋은 경로를 생성해 주는데 문제점이 있었다. 신경망을 이용해서 공구경로를 생성하면, 1) 몇개의 구역으로 나누어질 수 밖에 없는 monotonous Area를 갖는 형상, 2) 비교적 간단하지만 아일랜드로 인해 가공영역이 분할되는 형상, 3) 복잡한 외부윤곽과 많은 아일랜드가 존재하는 매우 복잡한 포켓형상에 대해서, 공구후퇴 및 경로길이가 sub-optimal인 경로를 로보스트하게 생성해 준다. 생성된 경로는 평활과 피드맵을 이용하여 절삭력을 일정 범위로 유지하도록 하는 가공조건을 고려하였다. 신경망을 이용하는 경우 가공점의 수가 많게 되면 경로생성 시간이 오래 걸리는 단점도 있다. 그러나 가공점의 수는 포켓형상의 복잡도하고는 상관이 없기 때문에 형상이 복잡해도 가공점의 수가 작으면 빨리 경로를 생성해 준다. 추후연구로서 공구후퇴시 드릴링 작업을 피하기 위해서 선가공 지점에서 가공영역으로 바로 접근하는 방식에 대한 보완이 필요하다.

참고문헌

1. Held, M., On the Computational Geometry of Pocket Machining, Spinger-Verlag, 1991.
2. Kramer, T. R., "Pocket Milling with Tool Engagement Detection," Journal of Manufacturing Systems, Vol. 29, No. 11, pp. 370-376, 1991.
3. Wang, H. P., Chang, H., Wysk, R. A., Chandwarkar, A., "On the efficiency of NC tool path planning for face milling operations," Journal of Engineering for Industry, Vol. 109, pp. 370-376, 1987.
4. Persson, H., "NC machining of arbitrarily shaped pockets," Computer Aided Design, Vol. 10, No. 3, 1978.
5. Suh, Y. S., Lee, K., "NC milling tool path generation for arbitrary pockets defined by sculptured surfaces," Computer Aided Design, Vol. 22, No. 5, 1990.
6. Preiss, K., "Quality Evaluation of NC Modules for Automatic Mill Pocketing".
7. Held, M., Lukacs, G., Andor, L., "Pocket machining based on contour par-

allel tool paths generated by means of proximity maps," Computer Aided Design, Vol. 26, No. 3, 1994.

8. Bao, H. P., Yim, H., "Tool pathdetermization for end milling of non convex-shaped polygonal pockets," Transactions of NAMRI/ SME, Vol. XX, 1992.
9. Wang, H. P., Chang, H., Wysk, R. A., "An Analytical Approach to Optimize NC Tool Path Planning for Face Milling Flat Convex Polygonal Surfaces," IIE Transactions, Vol. 20, No. 3, 1988, pp. 325-332.
10. Bala, M., Chang, T. C., "Automatic cutter selection and optimal cutter path generation for prismatic parts," International Journal of Production Research, Vol. 29, NO. 11, 1991, pp. 2163-2176.
11. Prabhu, P. V., Gramopadhye, A. K., Wang, H., "A general mathematical model for optimizing NC tool path for face milling of flat convex polygonal surfaces," International Journal of Production Research, Vol. 28, 1990, pp. 101-130.
12. Smith, S., Tlusty, J., "An Overview of Modeling and Simulation of the Milling Process," Journal fo Engineering for Industry, Vol. 113, 1991, pp. 169-175.
13. Woodward, J., Computing Shape, Butterworths, Guildford, UK, 1986.
14. Preiss, K., Automated Mill Pocketing Computations, In Advanced Geometric Modeling for Engineering Applications, North-Holland, Amsterdam, NL, 1989.
15. Preparata, F. P., Shamos, M. I., Computational Geometry, Springer-Verlag, 1985.
16. Hopfield, J. J., Tank, D., "Neural computation of decisions in optimization problems," Biological Cybernetics, Vol. 5, 1985, pp. 141-152.
17. Kirkpatrick, S., Gelatt, C. D., and Vecchi M. P., "Optimization by Simulated Annealing," Science, Vol. 220, 1983, pp. 671-680.
18. Durbin, R., Willshaw, D., "An analogue approach to the travelling salesman problem using an elastic net method," Nature, Vol. 326, 1987, pp. 689-691.
19. Angeniol, B., et al., "Self-Organizing Feature Maps and the Travelling Salesman Problem," Neural Networks, Vol. 1, 1988, pp. 289-293.
20. Kohonen, T. E, Self Organization and Associative Memory, Springer-Verlang, 1984.
21. Wilson, G. V., Pawley, G. S., "On the stability of the travelling salesman problem algorithm of Hopfield and Tank," Biol. Cybernet., Vol. 58, 1988.