韓國精密工學會 '96年度 秋季學術大會論文集 pp.572~577

광조형물의 효율적 성형을 위한 최적 지지대 구조 생성 알고리즘에 관한 연구

A Study on Generation Algorithm of Optimal Support Structure for Effective Building of Stereolithographic Parts

김 호 찬(부산대 대학원), 최 홍 태(부산대 기계기술연구소), 이 석 희(부산대 생산기계공학과)
Ho-Chan Kim(Graduate School, Pusan National Univ.), Hong-Tae Choi(Pusan National Univ. RIMT)
Seok-Hee Lee(Dept. of Mechanical and Production Engineering, Pusan National Univ.)

ABSTRACT

Stereolithography is a process used to rapidly produce stereolithographic parts directly from three dimensional CAD models. However, design methodologies necessary to create components to be built by stereolithography are different from those required by conventional machining processes. A case in point is the nescessity of support structures, which are used to support a component during the building the build but are removed once building and curing are complete. Support structures are required to anchor the component to the platform and to prevent sagging or distortion. This paper deals with the specially maintained SupportMap data structure to find some region which need support structures. Interferences between support structures and parts, as well as among support structures are checked and statically stable regions are searched to remove the surplus support structures. Cross shaped tooth profiles are designed for easy eliminating the support structures.

Key Words: Stereolithography(광조형법), Support Structure(지지대)

1. 서 론

급속조형장치는 소프트웨어와 하드웨어를 포함하여 매우 고도로 일체화된 기기이다. CAD 데이터에서 2차원 슬라이스 데이터를 도입할 뿐만 아니라 실제의 조형에서는 그 데이터에 많은 손질을 해야 한다. 특히 광조형법일 경우 액체상태의 광경화성 수지를 사용하기 때문에 조형물이 완성되어서 수지 통속의 플랫포옴으로부터 분리시킬 때 조형물의 파손 방지, 조형시 지붕형상과 돌출부분에서의 처짐 방지, 그리고 섬과 같이 떠다니는 층의 고정 등을 위해서 지지대가 반드시 필요하다. 종래의 지지대 설계는 CAD상에서 조형대상을 모델링할 때 동시에 이루어지기 때문에 사용자의 주관적인 판단과 경험에 의해 시행착오를 거치면서 이루어져 왔다. 따라서 정확히 지지되어야 할 부분이 간과되거나 불필요한 부분에서 지지되는 경우가 발생하여 조형물의 최종 정밀도에 좋지 않은 영향을 미치고, 후 처리 공정이 길어지는 단점이 발생하였다. 이러한 문제점을 해결하기 위한 많은 노력으로 현재 BridgeWorks[1], Magics SG[2]와 같은 지지대 설계를 위한 상용 소프트웨어가 시판되고 있다. 그러나 이러한 상용 소프트웨어는 개발비 및 상업적인 이유로 인해 상당히 고가로 시판되고 있는 실정이다.

본 논문에서는 신속한 검색을 위한 데이터 구조와 효율적인 알고리즘에 의해 최소한의 사용자 입력으로 지지대 데이터를 자동으로 생성하여 STL 포맷으로 출력하는 방법을 제시하고자 한다.

2. 관련연구

일본의 吉川浩一[3]등은 조형대상을 한층씩 적층할 때 이전 층에 없던 독립된 2차원 단면이 존재할 경우 이를 지지하기 위해서 로보트 동작계획법중 Backprojection법을 응용한 것으로 조형물과 간섭이 일어나지 않는 접지 가능원추를 구하고 지지할 곳으로부터 플랫포옴 방향으로 지지대를 수직으로 내릴 때 접지 가능원추와 만날 경우 간섭이 일어나지 않는 접지 가능영역으로 경로를 바꾼다. 그러나 조형물이 복잡할 경우 접지가능원추를 구하기가 어려운 문제점이 있다. Kirschman[4]등은 facet모델을 이루는 모서리들을 Base border, False border, Flat border, Beam border로 분류하고, 이러한 facet의 경계 모서리를 검색할 수 있도록 트리 구조의 자료구조를 이용하였다. 그러나 이러한 검색법만으로는 조형대상과 지지대와의 간섭 검사를 할 수 없으며, 단순 지지보와 같은 형상에서는 과도한 지지대가 생길 수가 있다. Webb[5]등은 Stereolithography를 위한 STL 파일로부터 지지대를 생성하는 MS Windows 프로그램인 Support Designer의 개발에 대해 발표하였지만 세부 알고리즘에 대한 설

떻이 없고, 직육면체에 구멍이 있는 간단한 형상의 적용예만 소개하고 있다.

Swaelens[6]등은 벨기에 Materialise NV의 지지대 자동생성 프로그램인 MAGICS 소프트웨어에 관한 내용으로 STL 파일에서 지지대가 세워질 영역의 돌출부분, 길이, 면적 등을 고려하여 Block, Line, Point, Gusset 지지대중 하나를 결정하는 알고리즘을 제시하였다. 그러나 이러한 알고리즘으로는 비교적 단순한 다면체 형상에만 적용 가능하며 지지할 부분이 경사면일 경우와 조형대상과의 간섭이 있을 경우에는 정확한 지지대를 생성할 수 없다.

허정훈[7]등은 SLA를 이용한 시작작업을 위해서 돌체상의 지지대가 요구되는 부분을 검색한 후, 각 부분의 기하학적 조건을 반영하는 지지대 생성 알고리즘을 제시하였다. 그러나 정역학적으로 안정된 영역에서의 불필요한 지지대가 생성되는 점을 고려하지 않았으며 지지대 분리를 용이하게 하기 위한 Tooth profile과 최종 지지대 생성 결과인 STL 파일에 대해 언급하지 않았다.

3. 본 론

광조형법에서 지지대는 큰 단점 중의 하나이지만, 액체 광경화성 수지에 레이저를 주사하여 한 층씩 적층하는 특성 때문에 지지대는 반드시 필요하다. 점성이 큰 수지속에 엘리베이터의 상하운동 때문에 수지 표면영역이 파도가 일어나는 현상을 'Parachuting'이라 하고, 위 층의 하중으로 인해 처짐이 발생하는 현상을 'Sagging'라 하며, 수지가 경화되면서 수축에 의한 뒤틀림이 발생하는 현상을 'Warping'이라고 한다. 이상은 초기 광조형법의 대표적인 문제점으로 지적된 사항들이지만, 최근의 상용 광조형 장치 개발업체에서 조형성이 뛰어난 광경화성 수지 개발 및 수지 공급장치 개선 등으로 조형물의 정밀도를 많이 향상 시키고 있다. 그러나 여전히 지지대는 소프트웨어적으로 처리해야할 부분으로 조형의 정확성과 효율성 측면에서 고려하여야 할 사항이 많다.

3.1 지지대의 필요성

지지대가 필요한 조형대상의 자세와 지지 형태를 Fig. 1에서와 같이 5가지로 분류하였다. (a), (b)는 가장 일반적인 경우로 삼각형 facet의 법선벡터의 z 값이 음의 값을 가지는 경사면 중 일정한 각도(-0.7 ~ -1.0)이내에 드는 수평방향으로 돌출된 부분으로 처짐이 발생한다. (c)는 레이저 주사중 조형대상의 슬라이스가 이 전의 층에 없던 섬(island)과 같은 단면이 발생하는 경우로 지지대가 없으면 레이저 주사와 동시에 이 부분이 바닥면에 가라앉는다. (d)는 설계자가 지지대를 고려하지 않고 3차원 CAD에서 조형대상을 모델링할 때 흔히 발생한다. 이러한 경우 (a), (b)의 경우에서와 같이 지지대가 필요하지만, STL 파일을 대상으로 조형모델의 최적자세를 구하는 전처리 공정에 의해서 수정할 수 있다. (e)는 조형물이 완성된 후에 조형물 내부에 고립되어 있는 미경화 수지 제거, 표면에 묻어 있는 수지 세척, UV 오븐에서의 건조, 지지대 제거등과 같은 후처리 공정을 위해 플렛포옴 바닥면으로부터 조형물을 분리할 때 파손과 변형을 방지하기 위해 Base Support가 필요하다.

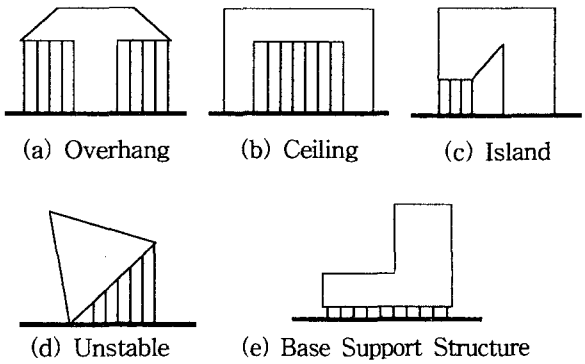

Fig. 1 Various Support Types

3.2 STL 포맷의 위상정보 재구성

Fig. 2에서와 같이 사면체를 구성하는 삼각형 facet수는 4개이고, vertex수는 4개이다. 그러나 이러한 STL 파일에서는 삼각형 facet수가 4개로 같지만, vertex수는 12개로 훨씬 많다. 이는 Vertex-to-Vertex 규칙을 따르는 STL 포맷의 특성 때문에 vertex 수가 중복되게 존재하기 때문이다.

주어진 STL 파일로부터 중복되지 않는 총 *vertex* 수와 *edge*수는 식(1), (2)와 같이 구할 수 있다.

중복되지 않는 총 *vertex* 수

$$= \frac{\text{총 } facet\text{수}}{2} + 2 \qquad (1)$$

중복되지 않는 총 *edge* 수

$$= \text{중복되지 않는 총 } vertex\text{ 수} \times 3 - 6 \qquad (2)$$

이러한 중복된 vertex 수를 줄이기 위해서 Vertex와 Facet 구조체 리스트를 작성하여 새로운 위상정보를 구성한다. Fig. 2에서 각 vertex를 공유하는 삼각형 facet 집합을 다음과 같이 구성할 수 있다. 이는 Vertex 구조체 리스트를 구성하는 기본 개념으로 검색 시간을 줄이기 위함이다.

v0 = { *f0, f1, f2* }
v1 = { *f0, f2, f3* }
v2 = { *f0, f1, f3* }
v3 = { *f1, f2, f3* }

Fig. 2에서 각 삼각형 facet의 세 이웃 facet 집합을 다음과 같이 구성할 수 있다. 이는 Facet 구조체 리스트의 요소로써 지지할 영역과 이웃한 미지지 영역과의 정역학적 관계를 검색하는데 도움을 주기 위해서이다.

f0 = { *v0, v1, v2* }

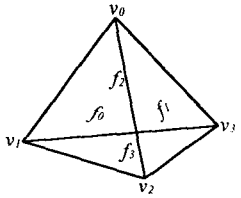

Fig. 2 Removed redundancy vertices

f1 = { *v0, v2, v3* }
f2 = { *v0, v1, v3* }
f3 = { *v1, v2, v3* }

3.3 Vertex와 Facet 구조체 리스트

위상 정보가 부족한 STL 포맷으로부터 지지대를 설계하기 위해서는 무엇보다 효율적인 데이터 구조가 중요하다. 따라서 본 연구에서는 Vertex 리스트와 Facet 리스트를 구성하여 지지대가 필요한 영역, 지지대와 조형대상과의 간섭 검사 및 과도한 지지대 방지를 위한 검색자료로 사용한다. Vertex 리스트는 정점의 x, y, z 좌표와 이를 공유하는 삼각형 facet의 법선벡터, 세 정점의 vertex 번호, 세 이웃 삼각형 facet 번호, 지지대 필요 여부 등의 정보들을 가진다. 이러한 위상 정보들은 Vertex 리스트와 Facet 리스트를 동시에 병행해서 구성함으로써 구해진다. Fig. 3에서는 Vertex 리스트와 Facet 리스트 구성 절차를 번호 순서대로 도식적으로 보여 주고 있으며 세부적인 설명은 다음과 같다.

① 입력 STL 파일로부터 한 삼각형 facet 단위로 vertex 좌표를 읽여 Vertex 리스트에 등록한다. 만일 동일한 점이 이미 등록되어 있다면 이 vertex의 색인 번호(StlVertex[i]의 i)를 리턴하고, 동일한 점이 등록되어 있지 않으면 새로운 vertex 색인 번호를 추가하고 i를 리턴한다.

② 입력 STL 파일로부터 읽어들인 법선 벡터의 x, y, z 좌표를 Facet 리스트 배열인 StlFacet[i]의 Normal.x, Normal.y, Normal.z에 할당한다. 그리고 Normal.z값이 -1.0~-0.7 사이에 있으면 지지대가 필요한 것으로 판별하여 NeedSupport에 'YES' 플래그를 설정하고, 그렇지 않으면 'NO' 플래그를 설정한다.

③ ①에서 리턴된 세 StlVertex[i]의 vertex 색인 번호 i를 할당한다.

④ StlFacet[i]의 facet 번호 i를 StlVertex[i]에 연결시킨다.

이상과 같은 단계를 입력 STL 파일의 삼각형 facet 개수 만큼 반복해서 수행함으로써 일차적인 위상정보를 구성하게 된다. ⑤, ⑥은 각 삼각형 facet의 세 이웃들을 찾아내고, 아래쪽으로 향한 첨점 여부를 판별하는 단계이다.

3.4 이웃 삼각형 facet 및 아래 방향 첨점 검색

중복되지 않는 vertex와 facet들의 위상정보들이 완성되면 이를 바탕으로 삼각형 facet당 세 이웃 facet들을 찾는 문제는 다음과 같은 방법으로 쉽게 구할 수 있다.

Fig. 4에서 삼각형 facet *f0*의 세 이웃을 찾기 위해서 먼저, Facet 구조체 리스트에서 *f0*를 이루고 있는 세 정점의 vertex 번호(StlVertex[i]의 i)를 찾아

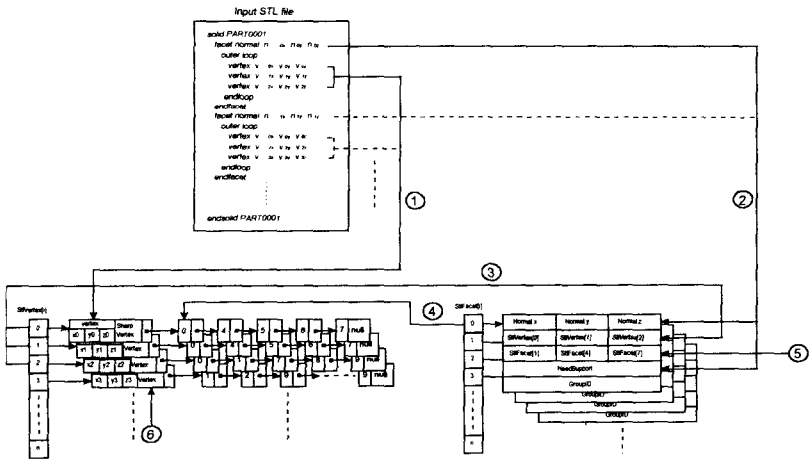

Fig. 3 construction of Vertex and Facet list

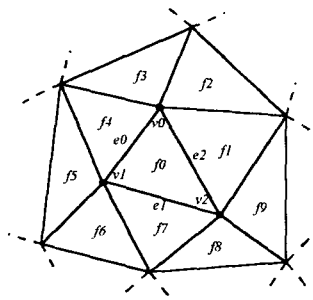

Fig. 4 Searching of three neighboring facets

서 해당 vertex를 공유하는 facet 번호(StlFacet[i]의 i)를 Vertex 구조체 리스트에서 검색한 결과는 다음과 같다. 여기서 이해를 돕기 위해서 StlVertex[0]는 *v0*로, StlFacet[0]는 *f0*로 표기한다.

v0 = { *f0, f1, f2, f3, f4* }
v1 = { *f0, f4, f5, f6, f7* }
v2 = { *f0, f1, f7, f8, f9* }

위 세 정점 *v0, v1, v2* 집합의 원소들은 삼각형 facet *f0*의 이웃이 될 후보 facet들이다.
이를 기초로 하여 삼각형 facet *f0*의 세 모서리 *e0, e1, e2*를 공유하는 facet들은 다음과 같이 각 모서리를 이루는 정점의 교집합으로 구할 수 있다.

e0 = *v0*∩*v1* = { *f0, f4* }
e1 = *v1*∩*v2* = { *f0, f7* }
e2 = *v2*∩*v0* = { *f0, f1* }

위 세 모서리 *e0, e1, e2* 집합의 원소들은 삼각형 facet *f0* 자신을 포함한 이웃 facet들을 가리킨다. 따라서 자기 자신을 제외한 세 이웃 facet들은 다음과 같이 구할 수 있다.

*f0*의 세이웃 facet집합={*e0*∪*e1*∪*e2*}-{*v0*∩*v1*∩*v2* }
={*f0, f1, f4, f7*}-{*f0*}
={*f1, f4, f7*}

Fig. 5에서 아래쪽으로 향한 첨점은 경사면의 기울어진 각도만으로는 지지대가 요구되지 않는 면이지만, 지지대가 없으면 조형되지 않는 아주 민감한 부분이다. 이러한 뾰쪽한 정점을 공유하는 모든 facet의 법선벡터 z값은 지지대가 요구되지 않는 음의 값(-0.699~-0.01)을 가진다. 이 부분에 대한 지지대의 설계를 위해서는 Vertex 구조체 리스트(StlVertex[i])에 연결되어 있는 facet 번호에 해당하는 Facet 구조체 리스트(StlFacet[i])를 찾아서 facet의 법선벡터의 z값을 검색하여 일정한 범위내의 음의 값을 가지면 Sharp Vertex로 판별하고, Vertex 구조체 리스트의 Sharp Vertex flag를 'ON' 시킴으로써 Support Map 생성시에 반영한다.

3.5 지지대와 조형물 사이의 옵셋

Fig. 6의 (a)는 수평으로 돌출된 외팔보와 같은 형상이고, (b)는 단순 지지보와 같은 형상으로 지지대가 필요한 영역은 삼각형 facet F1, F2 부분이다.

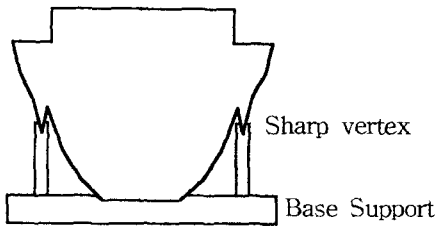

Fig. 5 Sharp vertex support

그러나 이 부분 전체를 바닥면에서 조형 방향으로 지지대를 만들게 되면 조형대상의 벽과 지지대가 붙게되어 후처리 공정에서 지지대를 분리하는 작업이 어렵게될 뿐만 아니라 조형물의 표면정도에 좋지 않은 영향을 미친다. 따라서 본 연구에서는 Support Map에서 δ량 만큼 옵셋된 지지대를 설계한다. 여기서 δ값은 사용자 입력으로 받아들이며 내정값은 6㎜로 설정하였다.

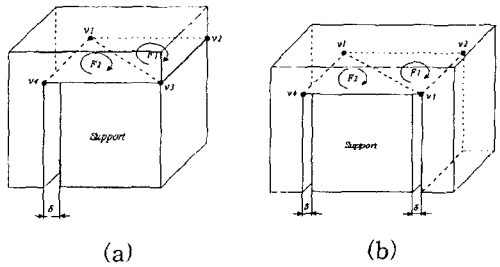

Fig. 6 Support structure of cantilever and simple beam with offset

3.6 지지대 결정 알고리즘

3.6.1 동일한 지지면의 삼각형 facet 그룹화

기본적인 자료구조가 완성이 되면 Support Map상에서 지지대가 필요한 영역중 정역학적으로 안정된 부분의 지지대를 제거하기 위해 지지대가 필요한 삼각형 facet별로 그룹화 시킨다.

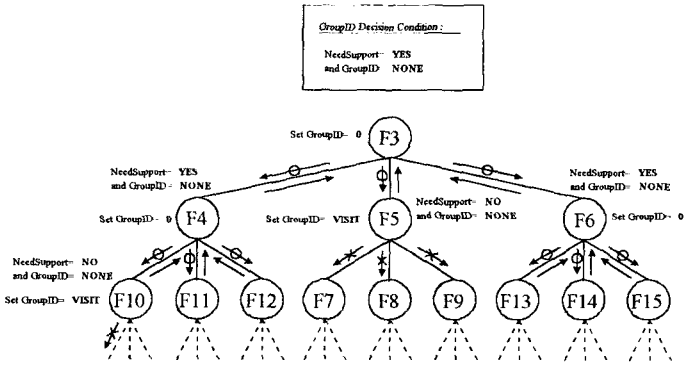

Fig. 7 Tree structure for GroupID decision

Facet 구조체 리스트의 NeedSupport 멤버값이 'YES'이고 GroupID가 'NONE'인 facet을 찾으면 같은 GroupID를 가질 삼각형 facet을 검색하기 시작한다. 여기서 GroupID는 'NONE'으로 초기화 되어있다. Fig. 7에서는 각 삼각형 facet의 세 이웃 facet들을 검색하기 위해 재귀 호출하는 상태를 나무구조로 보여주고 있다. 이들 이웃한 삼각형 facet들을 검색하고 NeedSupport와 GroupID를 검사하여 GroupID를 결정한다.

3.6.2 Support Map 작성

Vertex와 Facet 구조체 리스트가 완성되면 실제 지지대 생성에 필요한 데이터를 작성하기 위해 Fig. 8와 같은 Support Map이란 개념을 도입하였다. 여기서 Support Map은 바둑판 모양의 격자로 이루어져 있으며 격자점마다 SupportMap(i, j)와 같은 자료구조를 가진다. Support Map의 크기는 플렛포옴 바닥면의 넓이와 같기 때문에 최대 조형범위와 일치한다. SupportMap 구조체 리스트 작성을 위해 먼저 삼각형 facet *f1*~*f8*을 순서대로 Support Map상에 투영을 시킨다. 이 때 만나는 격자점에 해당하는 SupportMap(i, j)에 삼각형 facet 번호, 지지대 유·무, Support Map 바닥면에서 투영된 삼각형 facet까지의 높이, 지지대 바닥면과 꼭대기 *z*좌표를 구하여 할당한다. Fig. 8은 삼각형 facet *f8, f1, f5, f4*를 Support Map상에 투영시킬 때 SupportMap(6, 3)에서 만나는 예와 SupportMap 데이터 구조를 보여주고 있다. 여기서, GroupID는 같은 영역을 지지할 삼각형 facet끼리 묶기 위해서 부여하는 번호이다. 예를 들면 삼각형 facet *f5, f6*이 같은 그룹으로 GroupID가 '0'으로 설정되고, 삼각형 facet *f7, f8*은 GroupID가 '1'로 설정된다.

3.6.3 불필요한 지지대 제거

Fig. 9에서 삼각형 facet *f7, f8*은 지지대가 필요한 영역으로 GroupID가 0으로 설정되어 있다. 따라서 *f7, f8*과 접해 있는 facet *f9, f10, f11, f12*를 검사하여 GroupID가 0인 삼각형 facet *f7, f8*과의 정역학적 관계를 조사한 후 안정 영역 또는 불안정 영역으로 판별하여 해당 SupportMap(i, j)상의 SafeFactor에 제거해야할 길이값을 할당한다. Fig. 9의 예에서는 모서리 *e1*에서 *f7*과 *f12*는 정역학적으로 안정한 영역으로 판별하여 SafeFactor에 스스로 지지할 수 있는 단순 지지보의 길이의 반인 6㎜를 할당한다. 나머지 e2~e4는 불안정한 영역으로 판별하여 0㎜를 할당한다.

Fig. 10에서는 GroupID가 같은 삼각형 facet들, 다시 말해서 동일한 영역을 지지해야할 삼각형 facet들과 이들의 경계에 인접해 있는 facet들이 서로 안정한지, 불안정한지에 대한 정역학적 관계를 판별하기 위해 사용되는 원을 나타내고 있다.

Fig. 11에서는 지지대가 필요한 삼각형 facet은 *f7*

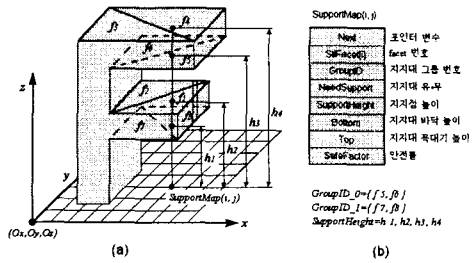

Fig. 8 Structure list of support map

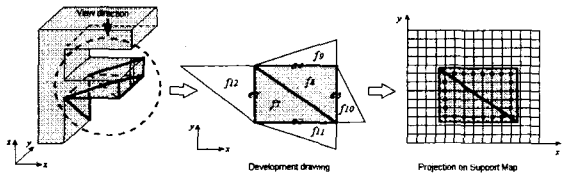

Fig. 9 Searching of removing support map

이며 이웃한 삼각형 facet은 *f10*과 *f12*이다. 여기서 *f7*이 존재할 수 있는 범위는 θ_3 이내이고, 이웃한 삼각형 facet *f10*과 *f12*는 θ_1 또는 θ_2 이내에 존재하게 된다. 따라서 *f10*은 θ_1에 위치하기 때문에 수평방향으로 돌출된 형상으로 모서리 *e3* 쪽은 불안정하다. *f12*는 θ_2에 위치하기 때문에 *f7*에 대해서 벽과 같은 상태로 지지하므로 모서리 *e1* 쪽은 안정하다고 판별한다.

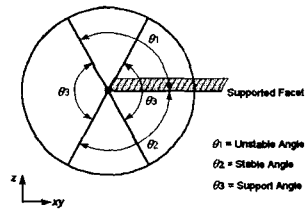

Fig. 10 Stable or unstable decision circle

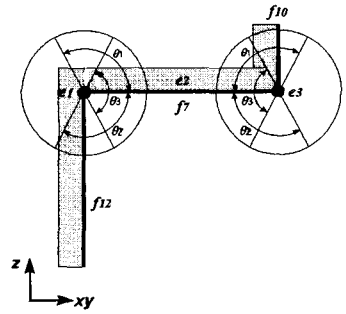

Fig. 11 Application of stable or unstable decision circle

3.7 Tooth Profiles 및 지지대 데이터 출력

이상과 같이 SupportMap 데이터가 완성되면 지지대를 STL 포맷으로 출력하게 된다. 조형물의 지지면과 지지대 사이가 단순히 접해있는 것만으로는 조형물을 확실히 지지할 수 없기 때문에 Fig. 12에

서와 같이 δ_z만큼 지지대의 일부가 조형물의 내부에 박혀 있어야 한다. 본 시스템에서는 δ_z값을 사용자가 입력하도록 하였지만 내정값으로 0.6㎜를 가진다. 그러나 Fig. 13의 (a)와 같이 지지대 몸체 부분이 지지면 안쪽으로 박혀 있을 경우 조형후 지지대 분리가 어렵게 된다. 그래서 Fig. 13의 (b), (c)와 같은 십자형 모양의 Tooth Profiles가 지지대의 몸체와 지지면을 연결함으로써 후 처리 공정중 지지대 제거를 용이하게 할 수 있다.

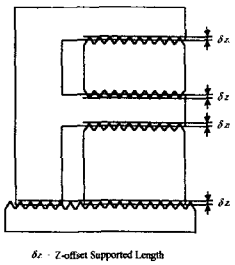

Fig. 12 Z-offset supported length

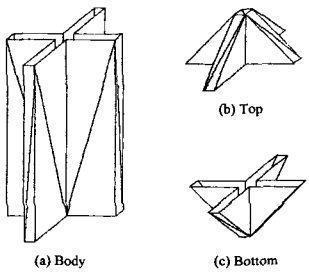

Fig. 13 Components of support structure

4. 적용예

Fig. 14와 15는 거미 형상의 STL 파일에 대한 지지대를 자동 생성하여 보여주고 있다.

Intel사의 3DR 실시간 렌더링 라이브러리를 사용하여 STL 파일 가시화 프로그램을 작성하였다. 이를 통해 거미의 다리 부분과 몸통 부분에 지지대가 정확히 생성되는 것을 확인할 수 있다.

5. 결 언

본 연구에서는 광조형물의 효율적인 성형을 위한 지지대 자동생성 시스템을 개발하였다. 지지대가 필요한 영역에 정확히 지지대를 생성시키기 위해 독자적인 SupportMap을 구성하여 조형물과 지지대 사이의 간섭, 지지대끼리의 간섭, 그리고 불필요한 영역에서 지지대 생성 등과 같은 기하학적 조건을 고려하여 이를 방지할 수 있도록 하였다. 그리고 지지대와 조형물 표면 사이의 접촉 부분에 Tooth profiles를 사용함으로써 조형물이 완성된 후 지지대의 제거를 쉽게 하도록 하였다. 생성된 지지대 데이터는 이 후의 슬라이싱 공정이 일관되게 하기 위해서 STL 파일 포맷으로 출력하였다.

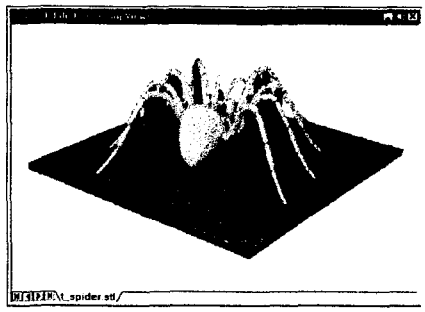

Fig. 14 View of supported spider

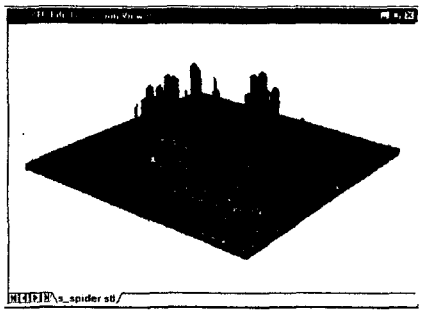

Fig. 15 Generation of supported structure for spider

참고문헌

(1) BridgeWorks, Solid Concepts Inc., http://solid-concepts.com/bridgewk.htm.

(2) Magics SG(Support Generation), Materialise, http://www.materialise.com/technic/magsg.htm.

(3) 吉川浩一, 鈴木宏正, 木村文彦, "光造形法におけるサポート形狀成形法", 日本精密工學會春季大會學術講演會講演論文集 C61, pp. 819~820, 1995.

(4) C.F. Kirschman, C. Namboodri, C.C. Jara-Almonte, A. Bagchi, R.L. Dooley, A.A. Ogale, "Stereolithographic Support Structure Design for Rapid Prototyping", Proceedings of the Third International Conference on Rapid Proto-typing, Dayton Ohio, pp. 115~121, 1992.

(5) D. Webb, V. Gerdes, C. Chassapis, "Computer-Aided support-structure design for stereo-lithography models", Proceedings of the Fifth International Conference on Rapid Prototyping, Dayton Ohio, pp. 221~228, 1994.

(6) B. Swaelens, J. Pauwels, W. Vancraen, "Support generation for rapid prototyping", Proceedings of the Sixth International Con-ference on Rapid Prototyping, Dayton Ohio, pp. 115~121, 1995.

(7) 허정훈, 이건우, "SLA를 이용한 신속시작작업을 위한 지지대 자동 생성 시스템의 개발", 대한기계학회 추계학술대회논문집, pp.788~794, 1995.