한국 CAD/CAM 학회 논문집
제 2 권 제 1 호 1997년 3월 pp. 28-34

평행 이동에 의한 스웹트 볼륨의 계산 방법

백낙훈*, 신성용**

Calculation of Translational Swept Volumes

Nakhoon Baek* and Sung-Yong Shin**

ABSTRACT

A swept volume is a useful tool for solving various types of interference problems. Previous works have concentrated on sweeping an object along an arbitrary path, that results in complex algorithms. This paper concerns the volume swept by translating an object along a linear path. After analyzing the structure of the swept volume, we present an incremental algorithm for constructing a swept volume. Our algorithm takes $O(n^2 \cdot \alpha(n)+T_c)$ time where n is the number of vertices in the original object and T_c is time for handling face cycles.

Key words : Swept volume, Translation, Arrangement, Computational geometry

1. 서 론

3차원 물체가 주어진 시간 간격 동안 공간 상을 이동하면서 지나간 모든 점들의 집합을 스웹트 볼륨(swept volume)이라 한다. 스웹트 볼륨의 한 응용 예를 들어보면, NC 시뮬레이션의 한 스텝은 공구의 이동에 대응하는 스웹트 볼륨과 피절삭체 사이의 집합 연산으로 표현할 수 있다[1].

Wang[2]은 미분기하학의 envelope theory에 기초하여 볼록 물체(convex object)가 미분 가능한 경로를 이동하는 경우의 스웹트 볼륨 계산 방법을 제안하였다. Blackmore[3]는 해석학적으로 스웹트 볼륨에 대한 미분 방정식을 유도하였고, 최근에는 이동 중에 변형이 일어나는 경우에 대한 이론을 제시하였다[4]. 이들 방법들은 이론적으로는 우수하나, 구현 과정이 복잡하고, 실제 수행시에 상당한 양의 계산이 필요하다.

Martin[1]과 Weld[5]는 물체 표면의 각 경계면들에 대한 스웹트 볼륨을 계산한 후, 이들의 합집합을 구함으로써 물체 전체의 스웹트 볼륨을 구할 수 있음을 보였다. Martin이 밝혔듯이, 이 방법은 모두 합집합의 계산 과정에서 많은 양의 계산 시간을 요구한다.

이제까지의 방법들은 모두 일반적인 이동 경로를 그 대상으로 하고, 이 때문에 복잡한 연산 과정이 요구된다. 반면에, NC 시뮬레이션 등의 응용 분야에서는 많은 경우에 평행 이동이나 평행 이동으로 근사된 형태로 이동 경로가 표현된다. 예를 들어, 3축 가공에서는 기준축과 평행하게 공구를 평행 이동시키는 것이 기본 연산이 되고[6], NC 프로그래밍에서는 상당한 양의 코드들이 평행 이동임이 밝혀져 있다[7]. 본 논문에서는 다면체(polyhedron)를 평행 이동시키는 경우의 스웹트 볼륨을 효율적으로 계산하는 방법을 제시하고자 한다.

2. 유향 직선에 기초한 스웹트 볼륨의 계산

3차원 물체는 여러가지 방법으로 표현할 수 있으나[8, 9], 본 논문에서는 그 중 널리 쓰이는 방법들 중의 하나인 B-rep(boundary representation)으로 표현한 경우를 그 대상으로 한다. B-rep에서, 다면체는 그 경계를 이루는 평면 다각형(planar polygon)들의 집합으로 표현되고, 각 평면 다각형의 법선 벡터는 항상 물체의 외부를 향한다. 대상이 되는 물체는 물체 내부에서의 자기 교차(self intersection)가 없는 유효 강체(valid solid)라고 가정한다. 설명의 편의를 위해,

*학생회원, 한국과학기술원 전산학과
**한국과학기술원 전산학과

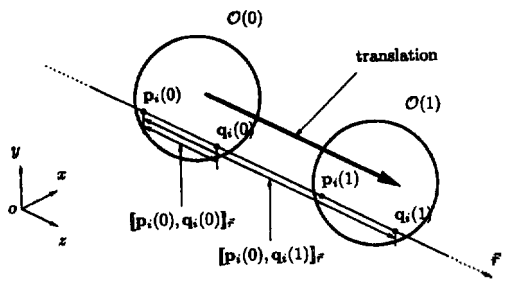


Fig. 1. Translation of an object O.

물체는 Fig. 1에서와 같이 3차원 좌표계의 z축과 평행하게 +z 방향으로 이동한다고 가정한다. 물체의 이동을 표현하기 위해서는 정규 시간 간격(normalized time interval) T=[0, 1]을 사용하고, 기하학적 형태(geometric entity) E의 시간 t에서의 위치를 $E(t)$로 표시한다. $E(0)$와 $E(1)$은 각각 E의 최초 위치와 최종 위치를 나타낸다.

본 논문에서는 z축에 평행하게 +z 방향으로 진행하는 직선들을 간단히 유향 직선(有向 直線, directed line)이라 하고, 하나의 유향 직선은 $\vec{r}$로 표시한다. $\vec{r}$ 상의 두점 $\mathbf{p}_i$, $\mathbf{p}_j$간의 우선 순위(precedence)는 다음과 같이 정의된다.

정의 1 두점 $\mathbf{p}_i$, $\mathbf{p}_j$의 z 좌표값을 각각 $Z(\mathbf{p}_i)$, $Z(\mathbf{p}_j)$라 할때, 유향 직선 $\vec{r}$에 대하여 $\mathbf{p}_i \cap \vec{r} \neq \emptyset$, $\mathbf{p}_j \cap \vec{r} \neq \emptyset$이고, $Z(\mathbf{p}_i) \leq Z(\mathbf{p}_j)$이면, $\mathbf{p}_i$가 $\mathbf{p}_j$에 우선한다고 하고, $\mathbf{p}_i \leq_{\vec{r}} \mathbf{p}_j$로 표시한다.

하나의 유향 선분(directed line segment) $[\mathbf{p}_i, \mathbf{p}_j]_{\vec{r}}$은 다음과 같이 정의한다.

정의 2 유향 선분 $[\mathbf{p}_i, \mathbf{p}_j]_{\vec{r}}$은 $\vec{r}$에서 $\mathbf{p}_i \leq_{\vec{r}} \mathbf{p} \leq_{\vec{r}} \mathbf{p}_j$를 만족하는 모든 점 $\mathbf{p}$들의 집합이다. 즉, $[\mathbf{p}_i, \mathbf{p}_j]_{\vec{r}}$= {$\mathbf{p}$ | 점 $\mathbf{p}$는 $\mathbf{p} \in \vec{r}$이고, $\mathbf{p}_i \leq_{\vec{r}} \mathbf{p} \leq_{\vec{r}} \mathbf{p}_j$}이다.

시간 t에서의 물체 O의 위치를 $O(t)$라 하면, 하나의 유향 직선 $\vec{r}$은 $O(t)$의 경계면들과 짝수번 교차한다[10]. 각 교차점들을 z좌표의 증가 순으로 $\mathbf{p}_1(t)$, $\mathbf{q}_1(t)$, …, $\mathbf{p}_i(t)$, $\mathbf{q}_i(t)$, …, $\mathbf{p}_n(t)$, $\mathbf{q}_n(t)$라 하면, 교차점들 중 i번째 쌍 $\mathbf{p}_i(t)$, $\mathbf{q}_i(t)$는 $O(t)$의 내부에 해당하는 유향 선분 $[\mathbf{p}_i(t), \mathbf{q}_i(t)]_{\vec{r}}$을 형성한다. 유향 직선들 중에서 $O(t)$와 교차하는 것들을 특별히 $\mathbf{R}$={ $\vec{r}$ | $\vec{r} \cap O(t) \neq \emptyset$ }로 표현하면, $O(t)$는

$$O(t) = \bigcup_{\vec{r} \in \mathbf{R}} (\vec{r} \cap O(t)) = \bigcup_{\vec{r} \in \mathbf{R}} \left(\bigcup_i [\mathbf{p}_i(t), \mathbf{q}_i(t)]_{\vec{r}} \right)$$

로 계산할 수 있다. $O(t)$의 스웹트 볼륨 SV는 시간 간격 T=[0, 1]에 대한 $O(t)$의 적분이므로,

$$\begin{aligned} SV &= \int_0^1 O(t)\, dt \\ &= \int_0^1 \bigcup_{\vec{r} \in \mathbf{R}} \left(\bigcup_i [\mathbf{p}_i(t), \mathbf{q}_i(t)]_{\vec{r}} \right) dt \\ &= \bigcup_{\vec{r} \in \mathbf{R}} \bigcup_i \left(\int_0^1 [\mathbf{p}_i(t), \mathbf{q}_i(t)]_{\vec{r}}\, dt \right) \end{aligned} \quad (1)$$

이다. 평행 이동에서는 $\mathbf{p}_i(0) \leq_{\vec{r}} \mathbf{p}_i(t) \leq_{\vec{r}} \mathbf{p}_i(1)$, $\mathbf{q}_i(0) \leq_{\vec{r}} \mathbf{q}_i(t) \leq_{\vec{r}} \mathbf{q}_i(1)$이므로, interval arithmetic 기법[11]에 의하여,

$$\int_0^1 [\mathbf{p}_i(t), \mathbf{q}_i(t)]_{\vec{r}}\, dt = [\mathbf{p}_i(0), \mathbf{q}_i(1)]_{\vec{r}}$$

이다. 따라서, 식 (1)은

$$SV = \bigcup_{\vec{r} \in \mathbf{R}} \bigcup_i [\mathbf{p}_i(0), \mathbf{q}_i(1)]_{\vec{r}}$$

이 되고, B-rep에서 필요로 하는 SV의 경계면들은

$$B(SV) = B\left(\bigcup_{\vec{r} \in \mathbf{R}} \bigcup_i [\mathbf{p}_i(0), \mathbf{q}_i(1)]_{\vec{r}} \right) \quad (2)$$

이다. 이 때, $B(\cdot)$는 경계(boundary)를 의미하는 연산자이다.

식 (2)를 직접 이용하려면, 집합 $\mathbf{R}$에 속하는 무한개의 유향 직선들을 다루어야 한다. 이를 피하기 위해, 우선 순위 관계를 다음과 같이 확장한다.

정의 3 유향 직선들의 집합 $\mathbf{R}_{(E_i, E_j)}$={$\vec{r}$ | $\vec{r} \cap E_i \neq \emptyset$, $\vec{r} \cap E_j \neq \emptyset$ }에 속하는 모든 유향 직선 $\vec{r}$에 대하여, $(\vec{r} \cap E_i) \leq_{\vec{r}} (\vec{r} \cap E_j)$이면, E_i가 E_j에 우선한다고 말하고, $E_i \leq E_j$로 표시한다.

$E_i \leq E_j$이고, $\mathbf{R}_{(E_i, E_j)} \neq \emptyset$이면, E_i, E_j로 둘러싸인 유향 선분들의 집합은 새로운 기하학적 형태인 $[E_i, E_j]$를 형성한다.

$[\mathbf{p}_i(t), \mathbf{q}_i(t)]_{\vec{r}}$의 끝점 $\mathbf{p}_i(t)$, $\mathbf{q}_i(t)$는 각각 유향 직선 $\vec{r}$이 $O(t)$의 경계면들과 교차하는 점들이다. 대응되는 경계면들을 각각 $f(t)$, $g(t)$라 하면, $\mathbf{p}_i(t) = \vec{r} \cap f(t)$, $\mathbf{q}_i(t) = \vec{r} \cap g(t)$로 표현할 수 있다. 따라서, 식 (2)를 경계면들의 관점에서 해석할 수 있다. 유향 직선 $\vec{r}$의 방향 벡터는 $\mathbf{T}$=(0, 0, 1)로 표현할 수 있다. 또, 각 경계면에서의 법선 벡터 $\mathbf{N}(f(t))$와 $\mathbf{N}(g(t))$는 항상 물체의 외부 쪽을 향한다. 따라서, $\mathbf{T}$와 법선 벡터들간의 내적을 이용하여, $O(t)$의 전체 경계면들은 다음 두 개의 서로 겹치지 않는 집합(disjoint set)으로 분류할 수 있다.

$$F^f(t)=\{f^f(t)\mid \mathbf{T}\cdot\mathbf{N}(f^f(t))\le 0, f^f(t)\in O(t)\}$$
$$F^b(t)=\{f^b(t)\mid \mathbf{T}\cdot\mathbf{N}(f^b(t))> 0, f^b(t)\in O(t)\}$$

$f^f(t)$와 $f^b(t)$를 각각 앞면(front face)과 뒷면(back face)이라 한다.

앞면과 뒷면이 공유하는 변들을 특별히 윤곽변(silhouette edge)이라 한다. 하나의 윤곽변 $e^s(t)$가 $f^f(t)$와 $f^b(t)$에 의해 공유된다면, $e^s(t)$상의 점 $\mathbf{p}_k(t)$는 유향 선분 $[\mathbf{p}_k(t), \mathbf{p}_k(t)]_r$를 형성한다. 따라서, $e^s(t)$는 평행 이동에 의해 $B(SV)$에 속할 수 있는 새로운 면

$$f^s=\int_0^1 e^s(t)\,dt=[e^s(0), e^s(1)]$$

을 생성한다. f^s는 $e^s(t)$의 평행 이동에 의한 결과이므로, $e^s(0)$, $e^s(1)$의 끝점들을 꼭지점으로 하는 평행사변형이다. 윤곽변들을

$$\varepsilon^s(t)=\{e^s(t)\mid e^s(t)=f^f(t)\cap f^b(t), f^f(t)\in F^f(t), f^b(t)\in F^b(t)\}$$

라 하면, 대응되는 윤곽면(silhouette face)들은

$$F^s=\{f^s\mid f^s=[e^s(0), e^s(1)], e^s(t)\in\varepsilon^s(t)\}$$

로 정의된다. 종합하면, $F^f(0), F^b(1), F^s$가 각각 SV의 경계면이 될 수 있는 후보들이고,

$$B(SV)\subseteq F^f(0)\cup F^b(1)\cup F^s$$

이다.

면 f가 $F^f(0)\cup F^b(1)\cup F^s$에 속할 지라도, 그 일부 또는 전부가 SV의 내부에 속할 수 있고, 이러한 경우는 해당 부분이 $B(SV)$에 포함되지 않아야 한다. 어느 부분이 $B(SV)$에 속하는 지를 알기 위해, 경계면들간의 우선 순위를 이용한다. $O(0)$는 유효 강체(valid solid)이므로, $O(0)$의 두 경계면 $f_i(0)$, $f_j(0)$는 서로 교차할 수 없고, 이들이 공유하는 유향 직선들의 집합 $\mathbf{R}_{(f_i(0), f_j(0))}$이 공집합이 아닌 경우에는 반드시 $f_i(0)\le f_j(0)$ 또는 $f_j(0)\le f_i(0)$의 관계가 성립한다. $f_i(0)$, $f_j(0)$가 공유하는 유향 직선들이 항상 존재하는 것은 아니므로, 경계면들간의 우선 순위는 부분 순서(partial order)이다.

이 부분 순서는 경계면들과 그들의 우선 순위 관계를 각각 노드(node)와 유향 에지(directed edge)로 하는 그래프 G로 표현할 수 있다. 그래프 G를 위상 정렬하여 전체 경계면들에 대한 리스트

$$L=(f_1(0), f_2(0), \cdots, f_i(0)\cdots, f_n(0))$$

이 구해진다. 이 리스트 L 상에서는 면 $f_i(0)$보다 우선하는 모든 경계면들은 $f_i(0)$보다 앞에 위치한다. L상에서 $f_i(0)\le f_j(0)$의 관계가 성립하는 두 경계면 $f_i(0)$, $f_j(0)$에 대하여, $f_i(0)\le f_k(0)\le f_j(0)$를 만족시키는 경계면 $f_k(0)$가 존재하지 않는 경우는 특별히 $f_i(0)\le f_j(0)$로 표시한다. 그래프 G에 싸이클(cycle)이 존재하는 경우에는 위상 정렬이 불가능하다[12]. 위상 정렬의 과정에서 싸이클이 발견되면, 싸이클 내에서 서로 인접한 두 경계면 f_a와 f_b를 찾은 후, f_b의 한 변을 기준으로 f_a를 분리하여 싸이클을 제거할 수 있다[13].

$f^b(0)$와 $f^f(0)$가 $f^b(0)\le f^f(0)$를 만족한다고 하자. $f^b(0)$는 $f^b(1)$으로 이동하여 스웹트 볼륨을 형성한다. 이때, $f^b(1)$과 $f^f(0)$사이의 우선 순위는 다음의 3가지 경우로 나누어진다. Fig. 2는 각각의 경우에 대한 예를 보여주고 있다.

경우 1.1 ($f^b(0)\le f^b(1)\le f^f(0)$) $\mathbf{R}_{(f^b(0), f^f(0))}$에 속하는 유향 직선들은 $f^b(1)$을 만나는 순간에 SV의 외부로 나오게 되고, $f^f(0)$를 만나면 다시 SV의 내부로 들어간다. 따라서, $f^b(1)\cap\mathbf{R}_{(f^b(0), f^f(0))}$과 $f^f(0)\cap\mathbf{R}_{(f^b(0), f^f(0))}$에 해당하는 부분들이 $B(SV)$에 속한다.

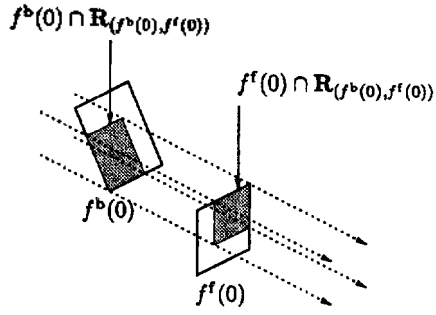
경우 1.2 ($f^b(0)\le f^f(0)\le f^b(1)$)$\mathbf{R}_{(f^b(0), f^f(0))}$에 속하는 유향 직선들은 $f^f(0)$, $f^b(1)$을 거치는 동안 계속 SV의 내부에 있고, $f^b(1)$을 지난 후에도 SV의 내부에 포함된다. 따라서, $f^b(1)\cap\mathbf{R}_{(f^b(0), f^f(0))}$과 $f^f(0)\cap\mathbf{R}_{(f^b(0), f^f(0))}$에 해당하는 부분들은 모두 $B(SV)$에 속하지 않는다.

경우 1.3 ($f^b(1)\cap f^f(0)\neq\emptyset$) $f^b(1)$과 $f^f(0)$는 각각 평면 다각형이므로, 이들의 교집합은 하나의 직선 l상에 위치한다. $\mathbf{R}_{(f^b(0), f^f(0))}$의 유향 직선들은 l을 기준으로 $f^b(0)\le f^b(1)\le f^f(0)$ 또는 $f^b(0)\le f^f(0)\le f^b(1)$ 중의 하나를 만족하고, 각각은 경우 1.1과 1.2에 해당한다.

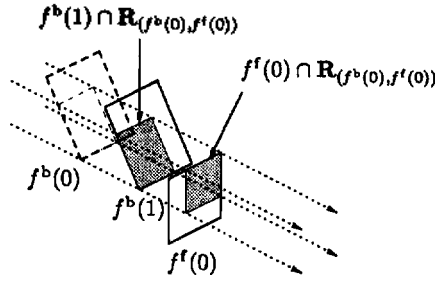
윤곽변들에서 $B(SV)$에 속하는 부분들을 찾기 위해, 한 윤곽변 $e^s(0)$가 $f^b(0)\le e^s(0)\le f^f(0)$를 만족한다고 하자. $f^b(0)$와 $f^f(0)$는 각각 L의 순서에 따라 결정할 수 있다. $f^b(0)\le e^s(0)\le f^b(1)$이면, $[e^s(0), f^b(1)]$에 해당되는 부분은 스웹트 볼륨의 내부에 속하므로, $[e^s(0), f^b(1)]\notin B(SV)$이다. 마찬가지로, $f^f(0)\le e^s(1)\le f^f(1)$이면, $[f^f(0), e^s(1)]\notin B(SV)$이다. 따라서, $[e^s(0), e^s(1)]$상에서 $B(SV)$에 속하는 부분은

$$[e^s(0), e^s(1)]-[e^s(0), f^b(1)]-[f^f(0), e^s(1)] \qquad (3)$$

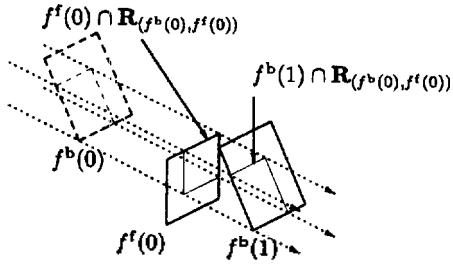
이다. 다음 절에서는 $B(SV)$에 속하는 모든 부분들을 찾는 알고리즘을 제시한다.



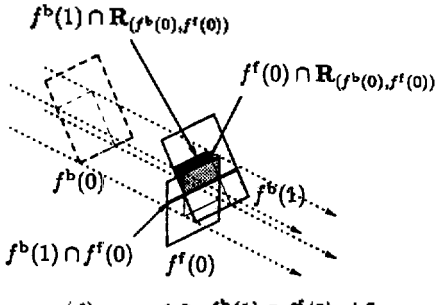
(a) initial positions



(b) case 1.1: $f^b(0) \preceq f^b(1) \preceq f^f(0)$



(c) case 1.2: $f^b(0) \preceq f^f(0) \preceq f^b(1)$



(d) case 1.3: $f^b(1)\cap f^f(0)\neq\emptyset$

Fig. 2. A front face and a back face.

3. 스웹트 볼륨 계산을 위한 알고리즘

xy평면에 평행한 임의의 평면을 기준 평면(reference plane) P_{ref}이라 하자. 기하학적 형태 E에 대하여, $\mathbf{R}_E=\{\vec{r}\mid \vec{r}\cap E\neq\emptyset\}$이라 하면, E를 P_{ref}에 사영(projection)시킨 결과는 $Proj(E)=\bigcup_{\vec{r}\in\mathbf{R}_E}(\vec{r}\cap P_{ref})$이다. 면 $f(t)$의 경우, $Proj(f(t))$는 2차원 다각형이다. $Proj(f(t))$와 교차하는 유향 직선들은 $\mathbf{R}_{f(t)}=\{\vec{r}\mid Proj(\vec{r})\in Proj(f(t))\}$이고, $0\le\tau\le1$인 임의의 시간 τ에 대하여, $Proj(f(t))=Proj(f(\tau))$이다.

위상 정렬된 면들의 리스트 L에서, 처음 $(j\text{-}1)$개의 경계면들 $f_i(0)$, $1\le i\le j\text{-}1$이 P_{ref}상에 사영되어 있다고 하자. 이때, $Proj(f_i(0))$, $1\le i\le j\text{-}1$의 변들은 2차원 선분들의 어레인지먼트(arrangement) A_{j-1}을 형성한다[11]. A_{j-1}에 속하는 하나의 영역 a_k에서의 유효면(active face)은 다음과 같이 정의된다.

정의 4 면 $f_p(0), 1\le p\le j-1$이 $a_k\subseteq Proj(f_p(0))$를 만족하고, $f_p(0)\le f_i(0)$이면서 $a_k\subseteq Proj(f_i(0))$를 만족하는 면 $f_i(0), 1\le i\le j-1$이 존재하지 않으면, $f_p(0)$를 $a_k\in A_{j-1}$의 유효면이라 한다.

유향 직선 $\vec{r}$이 $Proj(\vec{r})\in a_k$를 만족하면, a_k의 유효면 $f_p(0)$에 대하여 $\vec{r}\cap f_p(0)\neq\emptyset$이다.

L상의 처음 j개의 경계면들에 대한 새로운 어레인지먼트 A_j는 A_{j-1}에 j번째 면 $f_j(0)$의 사영 결과를 추가시켜서 계산할 수 있다. 영역 $a_k\in A_{j-1}$이 $Proj(f_j(0))$에 일부(또는 전부) 포함된다고 하자. 이때, a_k는 $a_k^{in}=a_k\cap Proj(f_j(0))$와 $a_k^{out}=a_k-a_k^{in}$의 두 부분으로 분리된다. $\mathbf{R}_{a_k^{in}}=\{\vec{r}\mid Proj(\vec{r})\in a_k^{in}\}$이라 하고, a_k의 유효면이 $f_p(0)$이면, $\mathbf{R}_{a_k^{in}}$에 속하는 모든 $\vec{r}$은 $f_p(0)$와 교차한다. 또, $a_k^{in}\subseteq Proj(f_j(0))$이므로 $\vec{r}\cap f_j(0)\neq\emptyset$이다. 따라서, a_k^{in}에 속하는 모든 $\vec{r}$은 $f_p(0)$, $f_j(0)$ 양쪽 모두와 교차하고,

$$\mathbf{R}_{a_k^{in}}\subseteq\mathbf{R}_{(f_p(0),\,f_j(0))} \tag{4}$$

이다. $f_p(0)$가 a_k의 유효면이므로, $f_p(0)\le f_i(0)\le f_j(0)$를 만족하는 면 $f_i(0)$는 존재하지 않는다. 따라서,

$$f_p(0)\preceq f_j(0) \tag{5}$$

이다.

j번째 면이 앞면 또는 뒷면이냐에 따라, 다음의 두 경우로 나누어진다.

경우 2.1 (j번째 면이 앞면 $f_j^f(0)$인 경우) $O(0)$가 유효 강체이므로, a_k의 유효면은 뒷면 $f_p^b(0)$이다. 식 (4), (5)에 따라, $\mathbf{R}_{a_k^{in}}\subseteq\mathbf{R}_{(f_p^b(0),\,f_j^f(0))}$이고, $f_p^b(0)\preceq f_j^f(0)$이다. 따라서, 2절에서의 경우 1.1, 1.2, 1.3에 따라,

```
Algorithm Calculate the swept volume.
  Input: 물체 O, 평형 이동에 대한 정보 M
  Output: 스웹트 볼륨 SV
Object function SweptVolume(Object O, MotionDesc M)
  { step 1. 앞면/뒷면의 분류}
    O(0)의 경계면들을 F^f(0)와 F^b(0)로 분류;
    O(0)의 변들에서 ε^s(0)를 분류;
  {step 2. 경계면들을 위상 정렬}
    L←(F^f(0)∪F^b(0))의 면들을 위상 정렬한 리스트;
  {step 3. B(SV)를 계산}
    Arrangement A←∅;
    for each Face f∈L do
      A←A∪Proj(f)
      R←Proj(f)의 내부에 속하는 a_k^in들;
      if f∈F^f(0) then
        for each a∈R do
          g←a의 유효면;
          g와 f 상의 B(SV)에 속하는 부분들을 경우 1.1, 1.2, 1.
              3에 따라 계산;
          식 (4)에 따라, 윤곽변 상의 B(SV)에 속하는 부분들을
              계산;
        end for
      end if
      a_k^in 들을 유효면이 f인 새로운 영역 a_j로 합침;
    end for
  {스웹트 볼륨을 반환}
      SV←B(SV)을 이용한 SV의 B-rep 표현;
      return SV;
end function
```

Fig. 3. Algorithm *SweptVolume*.

주어진 a_k에 대하여 $f_p^b(0)$와 $f_i^f(0)$ 상에서 $B(SV)$에 속하는 부분들을 찾을 수 있다.

경우 2.2 (j번째 면이 뒷면 $f_j^b(0)$인 경우) 이후의 어레인지먼트 계산 과정에서, $f_j^b(0)$를 대응되는 뒷면으로 하는 앞면 $f_i^f(0)$, $i>j$가 찾아질 것이다. 이 $f_i^f(0)$는 경우 2.1을 만족하므로, 이러한 $f_i^f(0)$가 찾아질 때까지 처리를 미룬다.

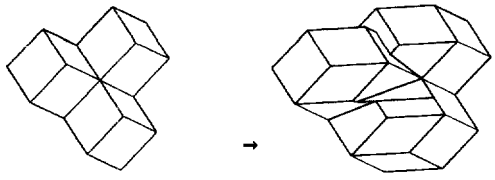
어느 경우이든, $a_k^{in} \subseteq Proj(f_i(0))$를 만족하는 영역 a_k^{in}들의 유효면들은 추후의 처리 과정을 위하여 수정된다. 이들 a_k^{in}들은 하나의 영역 a_j로 합쳐진 후, 이 새로운 영역의 유효면을 $f_j(0)$로 수정한다. a_k^{out}들은 $a_k^{out} \cap Proj(f_i(0))=\emptyset$이므로, 원래의 유효면들을 그대로 유지한다. 이러한 영역들 간의 합병은 어레인지먼트 A_i의 관리를 단순하게 한다.

이제 스웹트 볼륨은 2차원 선분들의 어레인지먼트를 점진적으로(incrementally) 형성하는 작업과 경계면들의 쌍에 대한 처리 작업을 통하여 계산된다. 전체 계산 과정을 정리하면, Fig. 3에 제시된 알고리즘 *SweptVolume*과 같다. 알고리즘 *SweptVolume*의 복잡도(complexity)를 분석하기 위하여, O의 꼭지점 수를 n이라 하자. 이때, O의 변과 면의 갯수들은 각각 $O(n)$이다. 단계 1에서, $O(0)$의 모든 경계면들이 $F^f(0)$와 $F^b(0)$으로 구분되고, 동시에 $\varepsilon^s(0)$들도 찾는다. 각 경계면들과 변들에 대한 처리가 상수 시간(constant time) 내에 수행될 수 있으므로, 이 단계의 시간 복잡도는 $O(n)$이다. 다음 단계로, $O(0)$에 속한 경계면들을 그 우선 순위에 따라 위상 정렬하여 리스트 L에 저장한다. 우선 순위에서의 싸이클을 처리하는 시간을 t_c라 하면, 이 단계는 $O(n^2+t_c)$의 처리 시간을 필요로 한다.

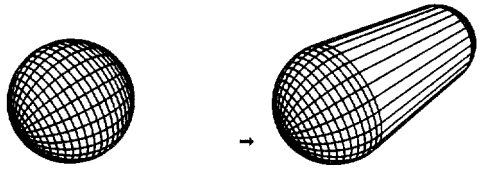
단계 3에서는 리스트 L의 순서에 따라, 각 경계면을 추가하는 방식으로 어레인지먼트 A를 점진적으로 형성하면서 $B(SV)$에 속하는 부분들을 찾아낸다. 어레인지먼트 A는 $O(0)$의 변들을 이용한 2차원 선분들의 어레인지먼트이다. 어레인지먼트를 형성하기 위해 입력되는 선분의 수는 $O(0)$의 변의 갯수와 같으므로, $O(n)$개가 된다. 영역들 간의 합병이 없는 경우에 어레인지먼트 A에는 $O(n^2)$개의 영역이 존재한다. 실제로는 영역 a_k^{in}들이 새로운 영역 a_j로 합쳐지고, 이후로는 a_k^{in}에 해당되는 영역들이 사용되지 않는다. 따라서, 각 영역은 최대 한 번의 합병에 사용된다. 새로이 생성되는 영역들은 이미 전체 영역 갯수 $O(n^2)$에 계산되어 있다. 그러므로, 영역들 간의 합병과 새로운 영역의 생성을 포함하여도 단계 3에서 사용되는 영역의 총 갯수는 $O(n^2)$이다.

단계 3은 각각의 작업을 차례로 수행하는 루프(loop) 구조를 가지므로, 각 작업 단계의 시간 복잡도와 공간 복잡도는 루프 전체를 수행 완료했을 때를 기준으로 계산하겠다. 점진적으로 어레인지먼트를 형성하는 알고리즘에서는 현재의 어레인지먼트에 선분 하나를 추가하는 데에 $O(n\cdot\alpha(n))$의 처리 시간이 필요하다[15, 16]. 여기서, $\alpha(n)$은 Ackermann 함수의 역함수이다[12]. $O(n)$개의 선분을 어레인지먼트 A에 첨가하기 위해서는 총 $O(n^2\cdot\alpha(n))$의 시간이 소요된다. 단계 3의 다른 작업들은 모두 하나의 영역 처리에 상수 시간을 필요로 하고, 총 $O(n^2)$개의 영역을 처리한다. 따라서, 단계 3은 총 $O(n^2\cdot\alpha(n))$시간에 수행된다.

종합하면, 우선 순위들 사이에 싸이클이 없는 경우에 알고리즘 *SweptVolume*은 $O(n^2\cdot\alpha(n))$시간에 수행된다. 싸이클에 의한 부가 처리에 필요한 모든 시간을 T_c라 하면, 알고리즘 *SweptVolume*의 수행 시간은 $O(n^2\cdot\alpha(n)+T_c)$이다. 주어진 면들의 우선 순위를 따지는 비슷한 유형의 알고리즘으로 은면 제거 알고



(a) example #1



(b) example #2

Fig. 4. Translational swept volumes.

리즘들[17, 18]을 들 수 있다. 이들 알고리즘들에서 이미 밝혀진 바와 같이, 실제 응용 예에서는 우선 순위들 사이에 싸이클이 존재하는 경우는 거의 발생하지 않는다. 평행 이동에 의한 스웹트 볼륨의 계산 시에는 유효 강체의 경계면들이 사용되므로, 나선 형태(spiral) 등의 심하게 꼬인 경우가 아니라면 싸이클의 발생은 매우 드물게 일어난다.

사용된 기억 공간들을 살펴보면, 우선 어레인지먼트 A에서 $O(n^2)$개의 영역이 생길 수 있으므로, $O(n^2)$의 공간이 필요하다. 또, 위상 정렬 시에 우선 순위 관계를 그래프로 표현하기 위해 $O(n^2)$의 기억 공간이 필요하다. 따라서, 전체 공간 복잡도는 $O(n^2)$이다. Fig. 4는 평행 이동에 의한 스웹트 볼륨의 예제이다.

4. 결 론

기존의 연구 결과들에서는 임의의 경로를 따라 움직이는 물체의 스웹트 볼륨을 구하고자 하였다. 반면에, 본 논문에서는 좀더 실용적인 알고리즘의 개발을 목표로 하여, 물체의 이동 경로를 평행 이동으로 제한하였다. 그 결과로, 2절에서와 같이, 물체의 경계면을 단위로 하여 어레인지먼트를 이용하는 처리 방법을 제시하였고, 최종적으로 주어진 물체의 꼭지점 개수가 n일 때, $O(n^2 \cdot \alpha(n))$의 처리 시간과 $O(n^2)$의 기억 공간으로 스웹트 볼륨을 계산하는 알고리즘을 제시하였다.

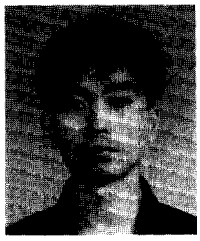
이들 알고리즘의 즉각적인 응용 예로는 NC 가공의 시뮬레이션을 들 수 있다. 3축 NC 가공에서는 대부분의 작업이 평행 이동이거나, 평행 이동으로 근사될 수 있으므로, 효율적인 NC 시뮬레이션을 위해 본 논문이 제시한 알고리즘들을 사용할 수 있다.

평행 이동은 회전 반경이 무한대인 회전 이동으로 볼 수 있다. 따라서, 본 논문에서는 회전 반경이 무한대인 회전 이동을 위한 스웹트 볼륨 계산 방법을 제시한 것이기도 하다. 본 논문에서의 기본 개념들을 이용하여 일반적인 회전 이동에 대한 스웹트 볼륨 계산 방법을 개발할 수 있을 것으로 기대된다.

참고문헌

1. Martin, R.R. and Stephenson, P.C., "Sweeping of three-dimensional objects", *Computer-Aided Design*, Vol. 22, No. 4, pp. 223-234, May. 1990.
2. Wang, W.P. and Wang, K.K., "Geometric modeling for swept volume of moving solids", *IEEE Computer Graphics and Applications*, Vol. 6, No. 12, pp. 8-17, Dec. 1986.
3. Blackmore, D. and Lue, M.C., "Analysis of swept volume via lie groups and differential equations", *The International Journal of Robotics Research*, Vol. 11, No. 6, pp. 516-537, Dec. 1992.
4. Blackmore, D., Leu, M.C., and Shih, F., "Analysis and modelling of deformed swept volumes", *Computer-Aided Design*, Vol. 26, No. 4, pp. 315-326, Apr. 1994.
5. Weld, J.D. and Leu, M.C., "Geometric representation of swept volumes with application to polyhedral objects", *The International Journal of Robotics Research*, Vol. 9, No. 5, pp. 105-117, Oct. 1990.
6. Elber, G., "Accessibility in 5-axis milling environment", *Computer-Aided Design*, Vol. 26, No. 11, pp. 796-802, Nov. 1994.
7. Childs, J.J., *Numerical Control Part Programming*, Industrial Press Inc., New York, N.Y., 1973.
8. Mantyla, M.J., *An Introduction to Solid Modeling*, Computer Science Press, Rockville, MD, 1988.
9. Chiyokura, H., *Solid Modeling with DesignBase*, Addison-Wesley Publishing Company, 1988.
10. Shamos, M.I., *Computational Geometry*, Springer Verlag, New York, 1979.
11. Snyder, J.M., "Interval analysis for computer graphics", *In Computer Graphics (SIGGRAPH '92 Proceedings)*, Vol 26, pp. 121-130, Jul. 1992.
12. Aho, A.V, Hopcroft, J.E., and Ullman, J.D., *Data Structures and Algorithms*, Addison-Wesley, Reading, MA, 1983.
13. Foley, J.D., Dam, A.V., Feiner, S.K., and Hughes, J.F., *Computer Graphics, Principles and Practice*, Second Edition, Addison-Wesley, Reading, Massachusetts, 1990.
14. O'Rourke, J., *Computational Geometry in C*, Cambridge University Press, 1993.

15. Edelsbrunner, H., Guibas, L., Pach, J., Pollack, R., Seidel, R., and Sharir, M., "Arrangements of curves in the plane: Topology, combinatorics, and algorithms", *Theoret. Comput. Sci.*, Vol. 92, pp. 319-336, 1992.
16. Wiernik, A. and Sharir, M., "Planar realizations of nonlinear Davenport-Schinzel sequences by segments", *Discrete Comput. Geom.*, Vol. 3, pp. 15-47, 1988.
17. McKenna, M., "Worst-case optimal hidden-surface removal", *ACM Trans. Graph.*, Vol. 6, pp. 19-28, 1987.
18. Schmitt, A., "Time and space bounds for hidden line and hidden surface algorithms", In *Proc. Eurographics 81*, pp. 43-56, Amsterdam, Netherlands, 1981.



백 낙 훈

1990년 한국과학기술원 과학기술대학 전산학 학사
1992년 한국과학기술원 전산학 석사
1992년 ~ 현재 한국과학기술원 전산학과 박사과정
관심분야 : Computer Graphics, Computational Geometry, Geometric Modeling

신 성 용

1970년 한양대학교 산업공학과 학사
1983년 University of Michigan, 산업공학 석사
1986년 University of Michigan, 산업공학 박사
1987년 ~ 현재 한국과학기술원 전산학과 교수
관심분야 : Computer Graphics, Algorithm Design and Analysis, Computer-Aided Geometric Design