한국 CAD/CAM 학회 논문집
제 12권 제 5호 2007년 10월 pp. 395-404

학술논문

구속조건을 가진 디자인 피쳐의 수정

우 윤 환*

Editing Design Features Constrained by Feature Depedencies

Yoonhwan Woo*

ABSTRACT

Feature-based modeling and history-based modeling are the two main paradigms that are used in most of current CAD systems. Although these modeling paradigms make it easier for designers to create solid model, it may pose dependency constraints on features that are interacting one with another. When editing such features, these constraints often cause unpredictable and unacceptable results. For example, when a parent feature is deleted, the child features of the parent feature are also deleted. This entails re-generations of the deleted features, which requires additional modeling time. In order to complement this situation, we propose a method to delete only the features of interest by disconnecting the dependency constraints. This method can provide designers with more efficient way of model modification.

Key words : Feature modification, Feature deletion, Feature dependency, History-based modeling, Feature-based modeling

1. 서 론

현재 3차원 솔리드 모델은 제품의 개발 및 해석에 있어 실질적인 표준 (de facto standard)으로 자리잡아 가고 있다. 이러한 3차원 솔리드 모델을 보다 쉽고 효율적으로 생성할 수 있도록 지원하는 다양한 모델링 기법들이 개발되었으며, 그 중에서 피쳐기반 모델링 (feature-based modeling)과 이력기반 모델링 (history-based modeling)은 현재 사용되는 상용시스템에 있어서 가장 근본적인 모델링 방법으로, 대부분의 3차원 캐드 시스템에 적용되어 사용되고 있다. 피쳐기반 모델링 방법의 가장 큰 장점은 설계뿐만 아니라 이후 응용분야와 통합할 수 있도록 가능하게 해주는 추가적인 정보를 제공할 수 있다는 점이다[1].

피쳐를 생성하기 위해서는 먼저 이의 정의를 위한 정보들이 필요하다. 이러한 정보에는 피쳐 파라미터 (feature parameter), 포지셔닝 파라미터(positioning parameters), 그리고 포지셔닝 레퍼런스(positioning reference) 등이 있다. 여기서 피쳐 파라미터란 해당 피쳐의 실제 형상을 결정하는 값들로서, 예를 들면 드릴링 홀의 경우 깊이, 반경, 팁각도 등이다. 반면에 포지셔닝 파라미터는 피쳐의 위치를 결정하는 값들로서 기준형상으로부터의 거리 및 각도 등이 이에 해당한다. 여기서 말하는 기준형상이란 포지셔닝 레퍼런스로서 보통 면(face) 또는 엣지(edge)이며 이들을 기준으로 하여 작업중인 피쳐의 상대위치를 지정하게 된다. 예를 들어, 모델상의 한 평면은 다른 피쳐의 위치 기준면으로 사용될 수 있으며, 모델상의 한 직선을 이용하여 이 직선으로부터 피쳐의 상대적 위치를 지정할 수 있다.

이력기반 모델링이란 모델을 생성하는 데 수행되었던 작업들에 관한 순서 및 관련정보를 기록하는 것으로, 이와 같이 저장된 모델링 작업의 순서를 모델 이력(model history)이라 부르며, 이에 기록된 생성순서에 따라 피쳐에 불리안 연산을 적용함으로써 모델의 경계를 계산할 수 있다[2]. 이러한 피쳐기반 및 이력기반 시스템은 사용자에게 모델의 생성 및 수정을 용이하게 하였으며, 이를 통하여 보다 효율적인 솔리드 모델링을 가능하게 하였다.

*정회원, 한상대학교 기계시스템공학과
- 논문투고일: 2007. 04. 30
- 심사완료일: 2007. 09. 19

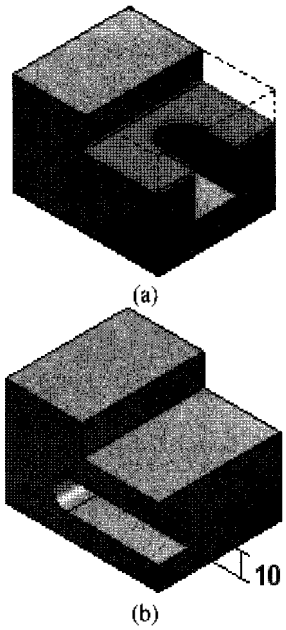

Fig. 1. Precedence constraint imposed on features: (a) the slot feature is attached on a face of step feature; (b) the position of the slot feature is measured from an edge of the step feature.

그러나, 이러한 장점에도 불구하고 피쳐기반 및 이력기반 시스템은 그 특성상 피쳐의 생성순서 및 레퍼런스에 의해 상호간 의존하게 되며, 이로 인하여 특정 피쳐의 수정이 다른 피쳐에도 영향을 주는 경우가 발생한다. 예를 들어, Fig. 1(a)에 보여진 모델은 먼저 스텝피쳐를 생성한 후에 그 스텝피쳐로 생겨난 평면을 기준면으로 하여 슬롯 피쳐를 생성한 모델이다. 이와 같은 생성순서 및 기준면의 사용은 두 피쳐간에 구속조건을 발생시키며, 이 경우 슬롯피쳐는 스텝피쳐에 의존하게 된다. 이러한 구속조건은 상용시스템인 Unigraphics NX의 경우 부모/자식 (parent/child) 피쳐로 표현된다. 또 다른 피쳐간의 구속조건은 Fig. 1(b)에 보여진 모델처럼 형상을 통한 구속조건은 없지만, 스텝피쳐로 생겨난 엣지를 슬롯 피쳐의 포지셔닝 레퍼런스로 사용함으로써 두 피쳐간에 구속조건이 발생한 예이다. 이러한 피쳐의 생성순서 및 레퍼런스를 통한 구속 때문에 경우에 따라 한 피쳐의 수정이 전체모델에 유효하지 않은 결과를 초래할 수 있으며, 또한 해당 피쳐의 수정이 불가능한 경우도 발생하게 된다.

피쳐를 수정하는 데는 크게 위치를 포함한 피쳐의 파라미터를 바꾸는 작업(change of feature parameters)과 피쳐를 삭제하는 작업(deletion of features)으로 구분할 수 있다. 피쳐의 파라미터 변경으로 인한 모델의 유효성 문제는 많은 연구가 진행되었으며, 이 들 대부분의 연구는 피쳐 네이밍(feature naming)을 통한 모델링의 유효화에 관련되어 있다[4-7]. 즉, 피쳐의 크기 및 위치를 변경함으로써 생기는 모델링 결과의 비일관성을 해결하는 방법으로 피쳐 네이밍이 주목을 받았으며, 아직도 이와 관련된 연구가 활발히 진행되고 있다[8]. 또한, 이와 더불어 해당 피쳐에 대한 수정작업이 모델의 유효화에 어떤 영향을 줄 것인가를 미리 사용자에게 알려주는 지원시스템에 관한 연구 등도 진행되고 있다[3].

이에 반해서 피쳐의 삭제에 의한 모델의 수정에 대한 연구는 상대적으로 적다. 피쳐의 삭제는 피쳐의 파라미터 수정에 비해 사용 빈도는 낮으나, 모델의 변경에 있어 없어서는 안 될 기능중의 하나이다. 제품 설계 과정에서 이미 존재하는 특징 피쳐의 기능이 필요치 않아 해당 피쳐를 삭제하는 경우가 많이 있다. 또는 모델링 과정에서 잘못 사용된 피쳐를 삭제 수정하는 경우도 빈번하다. 피쳐기반 시스템의 경우 모델링 단위가 피쳐이므로 해당피쳐를 쉽게 삭제할 수 있지만, 이 경우에도 역시 피쳐간 구속조건이 존재할 경우에는 문제가 발생한다.

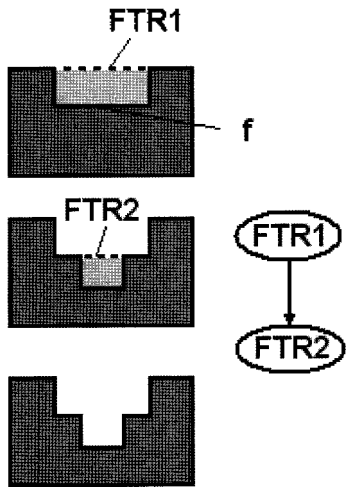

Fig. 2. A model created with feature dependency.

두 피쳐가 서로 구속되어 있지 않을 경우에, 어느 한 피쳐의 삭제에는 전혀 문제가 없다. 피쳐 히스토리에서 해당 피쳐를 삭제한 후, 다시 전체 히스토리를 실행하면 된다. 하지만, 두 피쳐가 서로 구속되어 있을 경우에는 좀 더 복잡한 결과가 나타난다. Fig. 2에 있는 모델은 FTR1을 생성한 후 이로 인해 생겨난 면

를 기준면으로 FTR2를 생성한 모델이다. 앞서 언급했듯이, 이러한 경우 FTR2는 FTR1에 구속되며, 이러한 구속조건을 화살표로 표시하였다. 만약 사용자가 FTR1을 삭제하고자 하는 경우를 보자. 세 가지 경우가 가능하다. 첫 번째, FTR1의 삭제가 이에 구속되어 있는 FTR2의 삭제까지 초래하는 경우로 Fig. 3의 A와 같은 경우이며, 현재 Unigraphics NX 및 CATIA V5가 이에 해당한다. 두 번째는 구속조건을 무시하고 FTR1만을 삭제한 후 다시 전체 히스토리를 실행시켜 B와 같은 결과를 얻는 경우이다. 마지막으로 세 번째는 FTR1의 삭제와 더불어, FTR2의 영역을 확장, 수정하여 C와 같은 결과를 얻는 경우이다. B와 C의 경우는 현재 상용 시스템에서 실행이 불가능하다.

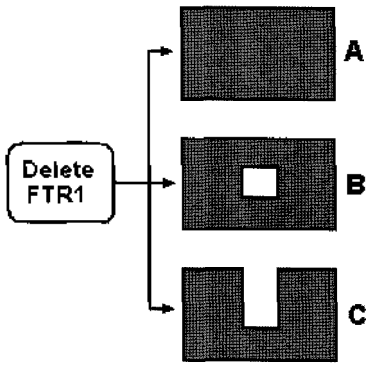

Fig. 3. Three possibilities caused by deleting FTR1 from the model in Fig. 2.

피쳐 및 이력 기반 CAD시스템에서 하나의 CAD모델은 여러 개의 피쳐들로 구성되어 있으며, 이들 피쳐는 설계자의 의도적인 또는 무의식적인 작업의 결과로 서로 구속되게 되며, 이로 인해 특정 피쳐의 정의가 다른 피쳐에 종속되게 된다. 만약 부품의 기능적인 변경 또는 설계자의 실수로 인한 CAD 모델의 변경이 필요하여, 종속되는 피쳐의 정의와 관련된 피쳐를 삭제할 경우, 이는 삭제하는 피쳐뿐만 아니라 이에 종속되는 모든 피쳐들에 영향을 주게 된다. 즉, Fig. 3의 C와 같이 모델을 수정하기 위해서는 FTR1을 삭제하여야 하며, 이는 현재 대부분의 상용 CAD시스템에서 FTR2의 삭제까지 초래하게 된다. 따라서 Fig. 2의 C와 같은 수정이 필요한 경우, 현재의 CAD시스템에서는 A와 같이 변경된 모델을 다시 C와 같이 수정하는 추가적인 작업이 필요하다. 그러나 이러한 작업은 서로 구속되는 피쳐의 개수가 증가하고 이들의 구속단계가 복잡해짐에 따라, 구속조건의 상위에 있는 피쳐의 삭제 시 재작업 시간이 증가하여 모델링의 용이성 및 효율성이 떨어지게 된다.

이러한 문제점을 해결하기 위해, 본 논문에서는 서로 구속하는 피쳐들을 사용자의 추가적인 작업 없이 Fig. 3의 C와 같이 수정할 수 있는 방법을 제시하고자 한다. 이를 통하여, Fig. 3에 있는 경우와 같이 구속조건을 가지고 있는 특정 피쳐를 삭제할 때 설계자가 상황에 맞는 삭제결과를 선택할 수 있는 지원 시스템을 구현하고자 한다.

2. 피쳐의 생성 및 피쳐간의 구속

현재 사용되는 3D캐드 시스템은 각 시스템마다 사용자 인터페이스는 조금씩 다르지만, 피쳐를 생성하는 기본 개념은 거의 유사하다. 특히 폼 피쳐(Form feature)의 경우 거의 모든 시스템에서 유사한 방법으로 정의되고 있다. 대표적인 폼 피쳐로는 protrusion 및 cut 등이 있으며 이들은 다시 그 특징에 따라 protrusion 피쳐는 보스(boss), 패드(pad) 등으로, cut 피쳐는 슬롯(slot), 포켓(pocket), 드릴링 홀(drilling hole)등으로 분류할 수 있다. 시스템에 따라 한 평면에서 스케치한 후 이를 익스트루드(extrude)나 리볼브(revolve)를 이용하여 폼 피쳐를 생성하기도 한다.

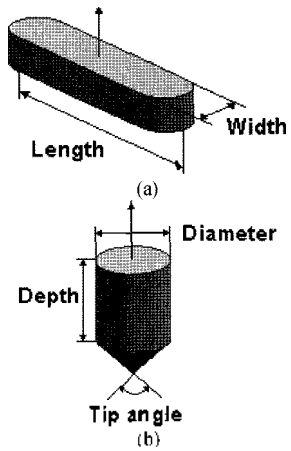

Fig. 4. Examples of form features; (a) step; (b) drilling hole.

이 들 폼 피쳐를 생성하는 데 있어서 공통점이 있다면 각 피쳐들을 생성할 때 기준면이 어디인가 지정해 주어야 한다는 점이다. 기준면(reference plane) 또는 위치면(placement plane)이라 불리는 이 평면은 새로 생성하고자 하는 피쳐의 시작면으로서 해당 피쳐 파라미터의 기준이 된다. 예를 들어, Fig. 5에 보여지는 것과 같이, 드릴링 홀의 경우 사용자가 지정한 기

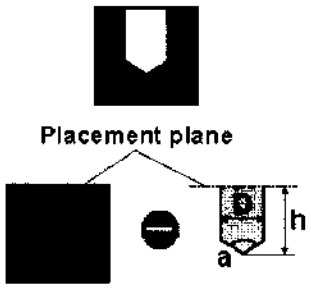

Fig. 5. Creation of drilling hole on a block.

준면을 기준으로 하여 드릴링 홀 모양의 폼 피처를 생성한 후 이를 블록으로부터 불리안 차 연산을 적용하여 원하는 형상을 생성하게 된다.

즉, 이미 생성된 피처의 한 면을 기준면으로 하여 새로운 피처를 생성하게 되면, 두 피처간에는 구속조건이 발생되며, 이렇게 발생된 구속조건은 피처의 수정에 영향을 주게 된다. 본 논문에서는 이러한 구속을 직접구속(direct dependency)이라 정의한다. 또한 이미 존재하는 피처를 부모피처(parent feature), 부모피처의 어느 한 면을 기준면으로 하여 생성되는 피처를 자식피처(child feature)라 정의한다. 직접구속의 예로, Fig. 3에서 FTR1을 삭제할 경우 A와 같은 결과가 나타나는 이유는 바로 FTR2의 기준면이 FTR1에 의해 생성된 면이기 때문이다. 즉, 부모피처인FTR1을 삭제하게 되면 이에 따라 이의 자식피처인 FTR2의 기준면이 삭제되고, 이로 인해 FTR2를 정의할 수 없기 때문에 대부분의 상용 CAD 시스템에서는 이와 같은 결과가 나타난다.

이와 달리, 하나의 피처가 다른 피처의 포지셔닝 레퍼런스로 사용되어 서로 구속되는 것을 간접구속(indirect dependency)이라 정의한다. 궁극적으로 본 연구는 두 피처가 직접 또는 간접 구속되어 있는 모든

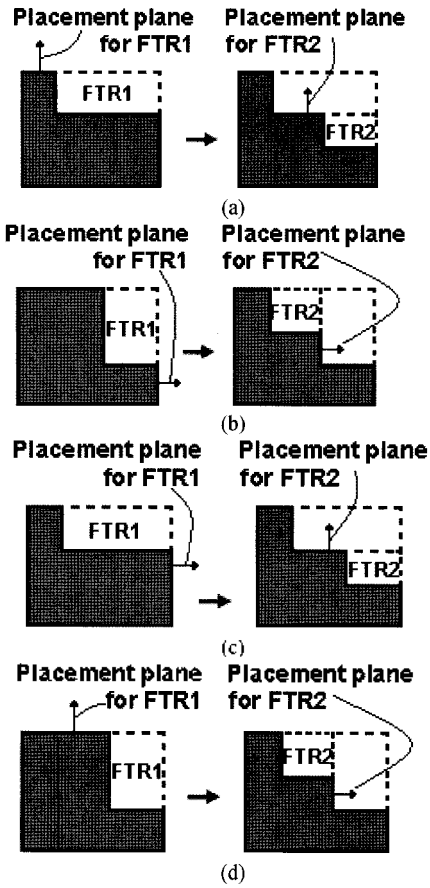

Fig. 6. Creation of the same model with different sets of directly-dependent design features; FTR2 is directly dependent on FTR1 in (a), (b), (c) and (d).

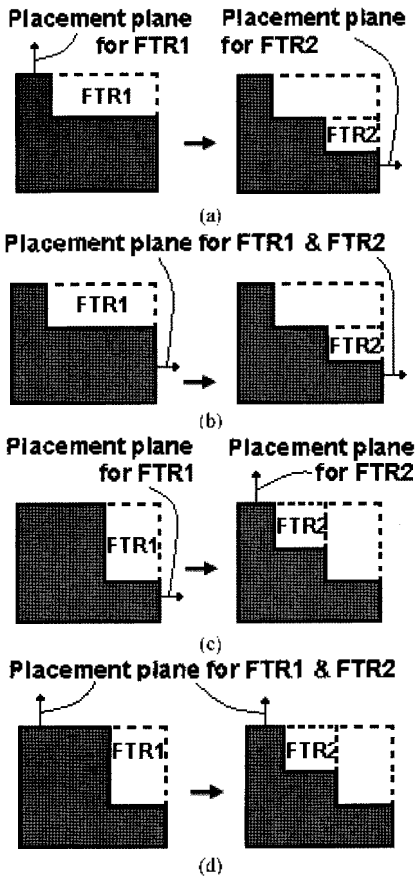

Fig. 7. Creation of the same model with different set of independent design features; FTR1 and FTR2 are independent in (a), (b), (c) and (d).

경우의 피쳐의 수정에 관해 진행될 계획이나, 현재는 연구의 첫 번째 단계로 기준면에 의해 직접 구속된 피쳐의 수정 방법만을 제시한다. 간접 구속의 경우에는 부모피쳐의 면뿐만 아니라 엣지 및 vertex 등에 구속되어 부모피쳐의 삭제 시 포지션의 재정의를 위한 새로운 레퍼런스를 생성하거나, 모델에 존재하는 다른 형상개체를 이용하여 지정하여야 하며, 이러한 문제는 향후 과제로 본 논문에서는 다루지 않는다.

여기서 또 한가지 고려할 점은 최종 형상이 동일한 모델이라 하더라도 사용자에 따라 사용한 피쳐가 다를 수 있다는 점이다. Fig. 6은 동일한 형상을 가진 모델을 직접 구속을 가진 서로 다른 피쳐를 사용하여 네 가지 방법으로 생성한 예이다. 즉, Fig. 6의 네 경우 모두 FTR2는 FTR1의 자식피쳐로서 직접 구속되며, 부모피쳐인 FTR1의 삭제는 FTR2에 영향을 미치게 된다. 반면에 Fig. 7은 동일한 모델을 서로 독립적인 피쳐들로 구현한 네 가지 경우를 보여주고 있다.

3. 직접 구속된 피쳐의 삭제

본 논문에서는 직접구속을 가진 부모 피쳐를 삭제하는 데 있어서 설계자가 추가적인 작업없이 보다 직관적인 수정 대안을 선택할 수 있는 모델링 방법을 제시한다. 직접 구속된 두 피쳐 중 부모피쳐를 삭제함으로써 발생할 수 있는 결과는, 앞서 언급한 바와 같이 세 가지 경우가 있다. 직접 구속된 두 피쳐는 자식피쳐의 기준면이 되는 평면을 통하여 접하고 있을 뿐 서로 교차하지는 않는다. 만약 두 피쳐가 서로 교차한다면 이 세 가지 외에 추가적인 경우도 가능할 수 있으나, 직접 구속된 부모피쳐를 먼저 삭제하는 데 있어 가능한 경우는 세 경우로 한정된다.

여기서 Fig. 3의 A의 경우와 같이 부모 피쳐와 자식피쳐가 같이 삭제되는 것을 동반삭제(coupled deletion), Fig. 3의 B 의 경우처럼 단순히 해당 피쳐만을 삭제하는 것을 단독삭제(single deletion), Fig. 3의 C의 경우처럼 부모 피쳐는 삭제하고 이에 구속되어 있는 자식 피쳐는 확장, 수정하는 것을 확장삭제(deletion with extension)라고 정의한다.

먼저 동반삭제는 부모피쳐의 삭제 시 자식피쳐도 같이 삭제되는 것으로, 이는 봄 피쳐를 생성하는 데 있어서 기준면이 자식 피쳐를 정의하는 레퍼런스로 사용되어 피쳐의 오리엔테이션, 가로, 세로, 깊이 등이 결정되기 때문이다. 따라서 직접구속의 경우 이러한 기준면을 가지고 있는 부모 피쳐를 삭제하다는 것은 이를 기반으로 하여 생성된 자식 피쳐의 정의가 소멸됨을 의미한다. 동반삭제는 새로운 기준면의 생성 및 설정, 새로운 구속조건의 추론 등 추가작업 없이 단순히 종속되는 피쳐를 순차적으로 삭제하는 가장 기본적인 방법으로, 현재 대부분의 상용 CAD 시스템에서 동반삭제가 기본(default) 작업으로 실행되며, 이에 본 논문에서는 동반 삭제에 대한 설명은 생략한다.

직접 구속된 피쳐의 단독삭제는 기준면을 제공하는 피쳐를 삭제하기 전에 먼저 이 기준면을 복사하여 이를 모델의 형상과는 별도의 독립적인 면으로 만든 후, 이를 종속되는 피쳐의 기준면으로 정의함으로써 두 피쳐를 직접 구속으로부터 분리할 수 있다. 일단 분리가 된 피쳐의 삭제는 서로 영향을 미치지 않으므로 원하는 피쳐만을 단독적으로 삭제할 수 있다. 이러한 작업은 Unigraphics의 경우에 새로운 데이텀 플레인(datum plane)을 생성하고 이를 기준면으로 사용하는 것과 동일한 작업으로 생각할 수 있다(Fig. 8).

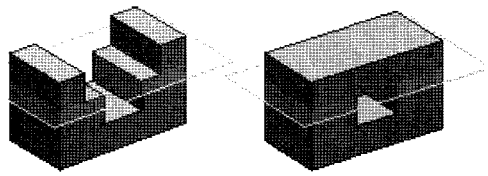

Fig. 8. Use of the plane copy for single deletion.

확장삭제의 경우에는 피쳐를 정의하고 있는 형상 및 파라미터들을 재정의하여야 하며, 이를 위해서는 구속되어 있는 피쳐간의 형상을 해석하고 이를 이용하여 새로운 피쳐를 정의하여야 한다. 이에 관련된 보다 자세한 내용은 다음 절에서 설명한다.

3.1 확장 삭제

확장삭제의 경우는 단독 삭제와는 달리 남아있는 피쳐의 확장으로 인한 형상의 변화가 일어나는 작업으로, 직접 구속된 피쳐와의 전체적인 관계를 파악하고 이로부터 새로운 형상을 만드는 방법이다. 이 방법은 구속되어 있는 피쳐중의 하나를 삭제할 경우, 남아있는 피쳐가 본래의 피쳐의 특징을 유지하면서, 삭제되는 피쳐가 차지하고 있던 영역까지 확장되는 기능을 제공한다.

두 피쳐가 서로 평면을 통하여 접촉하고 있을 때 이들 간의 관계에는 Fig. 6과 Fig. 7에 보여진 것과 같이 모두 여덟 가지 경우가 있다. 본 논문에서 제안하는 확장 삭제의 방법은 이 중 Fig. 6과 같이 두 피쳐가 직접 구속되어 있을 경우에 한한다. Fig. 7과 같이 두 피쳐가 독립적인 경우에는 항상 단독삭제가 가

능하기 때문에 남아있는 피쳐의 파라미터를 변경함으로써 확장삭제와 같은 효과를 쉽게 얻을 수 있다.

여기서 Fig. 6과 Fig. 7에서 우리는 두 가지 중요한 점을 발견할 수 있었으며 이는 다음과 같다.

• **Observation 1:**

두 피쳐가 직접 구속되어 있는 경우(Fig. 6), 두 피쳐 중 부모피쳐를 삭제할 때, 자식피쳐의 파라미터 중 높이(depth) 파라미터를 변경함으로써 확장삭제를 할 수 있다.

• **Observation 2:**

두 피쳐가 서로 독립적일 경우(Fig. 7), 먼저 생성된 피쳐를 삭제할 때, 나중에 생성된 피쳐의 높이(depth) 파라미터가 아닌 다른 파라미터를 변경함으로써 확장삭제와 같은 효과를 얻을 수 있다.

기준면을 이용하여 생성하는 피쳐의 경우는 항상 그 기준면에 피쳐의 프로파일을 생성하고 이를 기준면과 수직인 방향으로 스위핑(sweeping)하여 솔리드로 생성하므로, 스위핑을 하는 거리가 그 피쳐의 높이(depth) 파라미터가 된다. 이러한 이유로 두 피쳐가 직접 구속된 경우는 항상 자식피쳐의 높이 파라미터를 변경함으로써 확장삭제를 구현할 수 있다는 것이 observation 1의 내용이다.

반면에 observation 2는 두 피쳐가 독립적일 경우에 Fig. 7에서 볼 수 있듯이, 확장삭제와 같은 효과를 얻기 위해서는 높이 파라미터가 아닌 다른 파라미터를 변경해야 한다는 내용이다. 즉, 슬롯, 포켓 피쳐와 같은 경우는 길이(length)나 넓이(width) 파라미터를 변경함으로써 확장삭제와 같은 효과를 얻을 수 있다. 하지만 앞서 언급했듯이 두 피쳐가 직접구속 관계에 있지 않으면 어느 한 피쳐의 삭제가 동반삭제를 일으키지 않으므로 남아 있는 피쳐의 파라미터를 변경하여 쉽게 변경할 수 있어 본 논문에서는 직접 구속된 피쳐의 확장삭제만 고려한다.

직접 구속된 피쳐의 확장삭제는 또한 다음과 같은 두 경우로 분류할 수 있다.

- 직접 구속된 두 피쳐의 기준면의 법선 방향이 동일한 경우(Fig. 6(a), Fig. 6(b))
- 직접 구속된 두 피쳐의 기준면의 법선 방향이 서로 평행하지 않을 경우(Fig. 6(c), Fig. 6(d))

평면인 기준면을 통하여 두 피쳐가 직접 구속된 경우는 위의 두 경우만 존재하며, 각각의 경우에 대한 확장 삭제는 다음절들에서 설명한다.

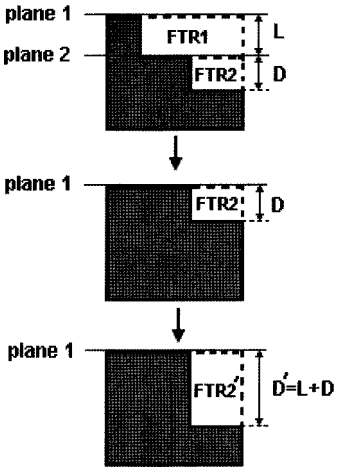

Fig. 9. Deletion with extension for the directly-dependent features in Fig. 6(a); the vectors normal to the placement planes are parallel.

3.2 기준면의 법선 방향이 동일한 경우의 확장 삭제

부모피쳐와 자식피쳐의 기준면의 법선 방향이 동일할 경우, 두 피쳐의 높이 파라미터는 기준면의 법선 방향으로 정의된다. 즉, 부모피쳐가 삭제되면, 자식피쳐의 높이 파라미터를 부모피쳐의 높이만큼 더하여 변경하여 확장삭제를 수행할 수 있다. CAD시스템은 모든 피쳐에 대한 정보를 가지고 있으며, 이 정보에는 피쳐의 기준면도 포함된다. 이 정보로부터 삭제할 피쳐(부모피쳐)와 확장할 피쳐(자식피쳐)의 기준면을 쉽게 얻을 수 있으며 이 두 평면간의 거리(L)은 쉽게 계산할 수 있다. 이를 이용하여 아래와 같은 방법으로 확장삭제 작업을 수행한다.

1. 삭제할 피쳐와 확장될 피쳐의 기준면을 각각 파악한다.
2. 파악한 두 기준면간의 거리(L)를 계산한다.
3. 확장될 피쳐의 기준면을 삭제할 피쳐의 기준면으로 변경한다.
3. 확장할 피쳐의 높이 파라미터를 L만큼 더한 값으로 수정한다.
4. 삭제하고자 했던 피쳐를 삭제한다.

예를 들어 Fig. 9에서, 두 피쳐의 기준면을 각각 plane1과 plane2라 하자. plane1은 단위 법선 방향 벡터 n1과 평면상의 한 점 p1, plane2는 단위 법선 방

향벡터 n2와 평면상의 한 점 p2로 정의된다고 가정한다. 여기서 n1과 n2는 동일한 벡터이므로 두 기준면간의 거리 L은 $|(p1-p2)\cdot n|$이며 이는 FTR1의 높이 파라미터 값과 같다. FTR1을 삭제하기 전의 FTR2의 높이 파라미터는 D이다. 먼저 FTR2의 기준면을 FTR1의 기준면으로 변경한 후, FTR2의 높이 파라미터를 D+L로 변경하여 확장삭제를 완성한다.

3.3 기준면의 법선 방향이 동일하지 않은 경우의 확장 삭제

이 경우는 두 피쳐가 직접구속의 관계에 있으면서, 기준면의 법선 방향이 서로 평행하지 않은 경우이다. 이러한 경우에도 부모피쳐의 확장삭제를 위해서는 자식피쳐의 높이 파라미터 값을 변경하여야 한다. 다만, 두 기준면의 법선 벡터의 방향이 동일하지 않으므로 부모피쳐의 기준면을 수정된 자식피쳐의 기준면으로 사용할 수 없다는 차이가 있다. 즉, 새로운 기준면을 생성하고 이를 확장될 자식피쳐의 수정된 기준면으로 설정하여 주어야 한다. 기준면의 법선 방향이 동일하지 않은 경우의 확장 삭제는 아래와 같은 방법으로 수행한다.

1. 확장될 피쳐의 기준면을 파악하고, 이 기준면의 법선 벡터(N)을 구한다.
2. 삭제할 피쳐의 바운딩 박스(bounding box)를 법선 벡터(N) 방향으로 생성한다.
3. 생성된 바운딩 박스의 법선 벡터(N) 방향의 길이(L)를 구한다.
4. 삭제할 피쳐의 기준면을 법선 방향으로 L만큼 이동한 새로운 기준면으로 변경한다.
5. 확장할 피쳐의 높이 파라미터를 L만큼 더한 값으로 수정한다.
6. 삭제하고자 하는 피쳐를 삭제한다.

제시된 방법의 두 번째 단계에 있는 삭제할 피쳐의 바운딩 박스를 법선 벡터 방향으로 생성한다는 의미를 좀 더 자세히 설명하면, Fig. 10에서 볼 수 있듯이 부모피쳐인 FTR1의 바운딩 박스를 FTR2 기준면의 법선 벡터(N) 방향으로 생성하는 작업이다. 바운딩 박스가 생성되면 박스의 높이를 구하는 작업은 간단하다.

전체적인 방법을 Fig. 11의 모델로 예를 들어 설명한다. 먼저, 두 피쳐의 기준면을 각각 plane1과 plane2라 하자. plane2의 단위 법선 방향 벡터를 N이라 하고 이를 이용하여 생성한 FTR1의 바운딩 박스의 N방향 길이가 L이며, FTR1을 삭제하기 전의

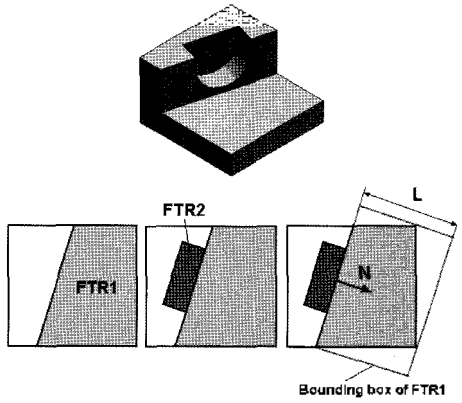

Fig. 10. Generation of bounding box of a parent feature.

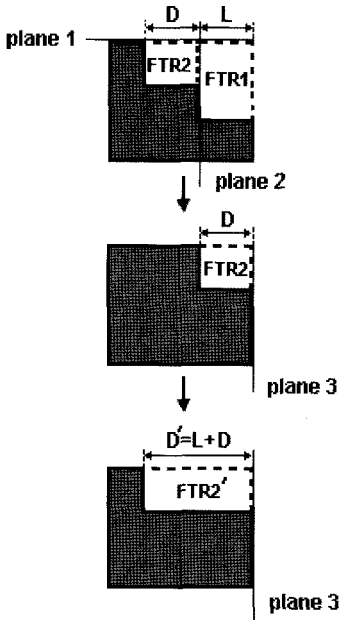

Fig. 11. Deletion with extension for the directly-dependent features in Fig. 6(d); the vectors normal to the placement planes are not parallel.

FTR2의 높이 파라미터는 D이다. 먼저 FTR2의 기준면을 N 방향으로 L만큼 이동하여 새롭게 생성된 기준면(plane3)으로 변경한 후, FTR2의 높이 파라미터를 D+L로 변경하여 확장삭제를 완성한다.

3.4 가법 피쳐의 확장 삭제

앞의 절들에서는 직접 구속된 두 피쳐간의 확장 삭제를 감법 피쳐에 적용한 예들만 제시되었으나, 본 논문에서 제시된 확장 삭제의 방법은 가법 피쳐(additive feature)의 경우에도 그대로 적용할 수 있다.

감법 피쳐와 마찬가지로 직접 구속된 두 가법 피쳐의 기준면의 법선 방향이 동일한 경우와 그렇지 않은 경우 두 경우로 분류할 수 있으며, 이를 위한 삭삭의 방법은 감법 피쳐의 경우들과 동일하다. Fig. 12는 앞서 설명한 확장 삭제의 방법을 가법 피쳐에 적용한 예이다.

이처럼 본 방법은 직접 구속된 두 개의 감법 피쳐 또는 두 개의 가법 피쳐들간의 확장 삭제에 적용할 수 있으며, 또한 Fig. 13에 나타난 경우처럼 하나의 가법 피쳐와 하나의 감법 피쳐가 서로 직접 구속되어 있는 경우에도 적용할 수 있다. 다만 예에서 볼 수 있듯이 그 사용면에 있어서 감법 또는 가법 피쳐들만이 직접 구속되어 있는 경우에 비해 덜 유용하다 할 수 있다.

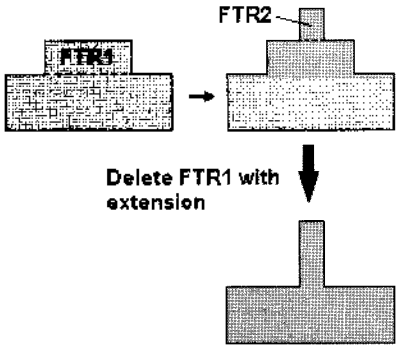

Fig. 12. An example of deletion with extension for additive features.

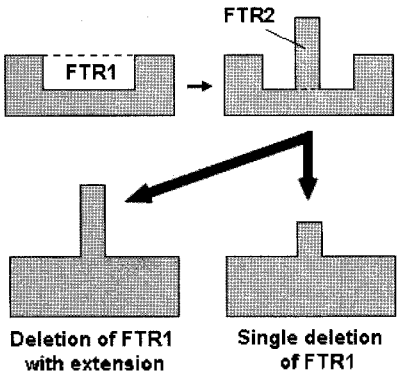

Fig. 13. An example of deletion with extension when additive and subtractive features directly constrained.

3.5 확장 삭제의 적용 예

앞서 설명한 방법들은 Spatial사의 ACIS를 기반으로 C/C++를 이용하여 윈도우 PC플랫폼 상에서 구현하였다. Table 1은 본 방법을 구현하는데 사용된 하드웨어와 소프트웨어를 나타난 테이블이다.

Table 1. H/W and S/W used for implementation of the method proposed in this paper

Hardware	PC Intel Pentium 4 CPU 3.0 GB, 1 GB RAM
Operating system	Microsoft Windows XP Ver. 2002, SP2
Development environment	Microsoft Visual C++ 6.0
Geometric kernel	Spatial Corp. ACIS 8.0

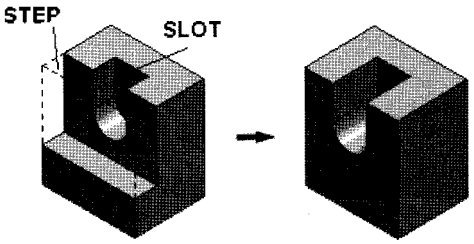

Fig. 14. Example of deletion with extension for STEP and SLOT features.

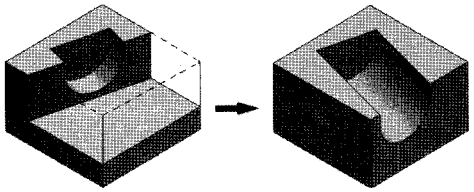

Fig. 15. Example of deletion with extension for the part shown in Fig. 10.

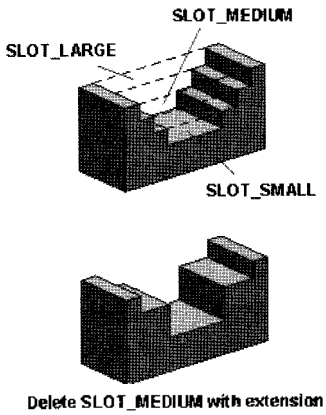

Fig. 16. Examples of deletion with extension for multiply-dependent features.

Fig. 14에 보여진 모델은 스텝 피쳐를 생성한 후, 스텝 피쳐로 생성된 면을 기준면으로 하여 슬롯피쳐를 생성한 모델이다. 스텝피쳐는 슬롯피쳐의 부모피쳐이며 이를 확장 삭제한 결과를 보여주고 있다. Fig. 15는 Fig. 10에 예로 보여진 모델을 확장 삭제한 결과이다.

Fig. 16은 서로 구속되어 있는 세 개의 슬롯 피쳐들 중 가장 큰 슬롯과 중간 크기의 슬롯을 확장 삭제했을 경우를 보여 주는 예이다. 이 두 경우 모두 모델을 수정할 때 사용자에 의해 자주 요구되는 작업들로 본 방법의 통하여 보다 효율적으로 작업이 가능하였다.

본 논문에서 제시한 방법은 하나의 피쳐가 여러 피쳐와 직접 구속된 경우에도 적용할 수 있으며, 구속된 피쳐를 하나씩 순차적으로 적용시킴으로써 확장삭제 작업을 완성할 수 있다. Table 2는 Unigraphics NX4와 CATIA V5에서 지원하는 피쳐삭제 가능과 본 논문에서 제시된 방법을 기반으로 구현된 시스템과의 비교를 보여주는 표이다.

Table 2. Comparisons of feature deletion functionalities in commercial systems and the system implemented with the proposed method

	UGS NX4	CATIA V5	Proposed method
Single deletion of child feature	○	○	○
Single deletion of parent feature	×	×	○
Coupled deletion	○	○	○
Deletion with extension	×	×	○

4. 결 론

본 논문에서는 피쳐 및 이력 기반 모델링으로 인한 폼 피쳐간의 구속과 이에 따른 모델 수정의 비효율성 등에 대해 논의하였으며, 이에 대한 해결 방법의 일환으로 단독삭제 및 확장삭제에 대한 개념을 정의하고 이를 구현하는 방법을 제시하였다. 이를 통하여 설계자에게 더 많은 모델 수정 기능을 제공하고, 설계자는 이를 이용하여 보다 신속하고 효과적인 모델의 수정을 할 수 있다.

본 논문에서 제시한 확장삭제의 방법은 감법 피쳐들(subtractive features)간의 삭제뿐만 아니라 가법피쳐들(additive features)간 그리고 가법피쳐와 감법 피쳐가 같이 모델링 되어 있는 상황에서도 적용할 수 있다. 다만, 본 연구는 두 피쳐가 기준면을 통해 구속된 경우만 고려하였으며, 확장삭제로 인한 피쳐의 파라미터 수정도 기준면의 법선 방향과 평행한 피쳐의 높이(depth)에 해당하는 경우로 국한되었다. 또한, 상용 모델러에서 구현되는 모델들은 직접 구속 및 간접 구속을 함께 포함한 경우가 많이 있어, 실제 상용 모델러에서 채택하여 즉각적인 사용에 있어서는 제한적일 수 있다.

따라서 향후 과제로 더 많은 수의 모델을 테스트함으로써 완성도를 높임과 동시에, 독립적인 피쳐간의 확장 삭제 및 치수 부여 등에 의한 간접 구속을 가진 피쳐의 확장 삭제에 대한 연구를 진행하여 실제적인 활용 범위를 높일 계획이다.

감사의 글

본 연구는 2007년도 한성대학교 교내연구비 지원과제로 수행되었으며 이에 감사 드립니다.

참고문헌

1. Shah, J., "Conceptual Development of form Features and Feature Modelers", *Research in Engineering Design*, Vol. 2, pp. 93-108, 1991.
2. Shah, J. and Mantyla, M., Parametric and Feature-based CAD/CAM, John Wiley & Sons, 1995.
3. Bidarra Rafael, "Validity Maintenance in Semantic Feature Modeling", PhD Dissertation, University of Coimbra, Portugal, 1999.
4. Chen, X. and Hoffman, C. M., "On Editability of Feature Based Design", *Computer Aided Design*, Vol. 27, No 12, pp. 905-914, 1995.
5. Rossignac, J. R., "Issues in Feature-based Editing and Interrogation of Solid Models", *Computer Graphics*, Vol. 14, pp. 149-172, 1990.
6. Kripac, J., "A Mechanism for Persistently Naming Topological Entities in History-based Parametric Solid Models", *Computer Aided Design*, Vol. 29, No. 2, pp. 113-122, 1997.
7. Capoyleas, V., Chen, X. and Hoffmann, C. M., "Generic Naming in Generative Constraint-based Design", *Computer Aided Design*, Vol. 28, No. 1, pp. 17-26, 1996.
8. Wang, Y. and Nnaji, B., "Geometry-based Sematic ID for Persistent and Interoperable Reference in Feature-based Parametric Modeling", *Computer Aided Design*, Vol. 37, pp. 1081-1093, 2005.

우 윤 환

1993년 한양대학교 정밀기계공학과 학사
1995년 Illinois Institute of Technology 기계공학과 석사
1999년 Colorado State University 기계공학과 박사
1999년~2002년 미국 Spatial Corp/Dassault Systems ACIS 개발팀 소프트웨어 엔지니어
2002년~2004년 국민대학교 자동차공학전문대학원 연구교수
2004년~2005년 성균관대학교 기계기술연구소 연구교수
2006년~현재 한성대학교 기계시스템공학과 전임강사
관심분야: 3D geometric modeling, Feature recognition, CAPP