공학 연구논문집 제 30권 1호 pp. 199~210. 울산대학교
Journal of Engineering Research Vol. 30, No. 1, pp. 199~210. 1999
University of Ulsan

제품구성 및 조립 애니메이션을 위한 VRML2.0코드 자동생성 시스템 개발

정덕호 (dhj@uou.ulsan.ac.kr), 서윤호 (yhs@uou.ulsan.ac.kr)
울산대학교 산업공학과

Automatic Generation of VRML2.0 Code for Animation of Product Structure and Assembly

Deokho Cheong, Yoonho Seo
Department of Industrial Engineering

<Abstract>

VRML(Virtual Reality Modeling Language) is a graphic modeling tool that is used for 3D representation and animation of objects through WWW (World Wide Web). Especially, VRML2.0 whose specifications were announced recently has excellent functions to model a virtual reality such as enhanced static worlds, interaction, animation, scripting, prototyping, etc. Through these capability of VRML2.0, the goal of this research is to develop a computer system to automatically generate VRML2.0 code for demonstrating product shape, product configuration and assembly process. The VRML code generated can be used to provide a product information through any browser on WWW. Specifically three algorithms are presented: one for converting CAD to VRML, another for converting BOM to VRML and the other for generating animation VRML.

1. 서론

가상공간에서의 제조기술이 발전함에 따라 제품에 관한 정보를 기업내외의 팀 또는 팀 구성원 사이에서 공유하는 것은 제조와 설계에 있어서 매우 중요하다. 지리적으로 분산된

제품 개발 관련 부서들이 제품의 형상정보, 구성정보 및 3차원 애니메이션을 통한 조립에 관한 정보를 쉽고 빠르게 생성, 획득할 수 있다면 이는 기업내외의 공동 설계팀이 제품 개발을 병렬적으로 진행할 수 있도록 해주는 동시공학(Concurrent Engineering)의 도구로서 이용 가능할 것이다. 이러한 제품에 관한 정보를 가장 효과적으로 인지하고 공유할 수 있는 방법중의 하나가 WWW(World Wide Web)을 통한 제품의 3차원 형상과 조립방법의 3차원 애니메이션을 들 수 있으며, 이러한 3차원 기능을 수행할 수 있는 가상현실 모델링 언어가 바로 VRML(Virtual Reality Modeling Language) [1]이다.

본 연구의 목적은 부품의 BOM (Bill of Material) 및 CAD 정보와 조립 방법에대한 입력으로부터 VRML2.0코드를 자동 생성하는 시스템을 개발하는 것이다. 자동생성된 VRML2.0코드는 Web Server 와 연계하여 WWW상에서 Browser를 통하여 그 제품의 구성, 형상 및 조립 등에 관한 정보를 제공하는데 이용될 수 있다. 이를 위하여 첫째 CAD정보를 VRML2.0 형상 정보로의 변환을 위한 시스템을 개발하였으며, 둘째 제품의 BOM정보를 이용하여 단품에서 중간 조립품, 최종 제품에 이르는 제품구성 표현을 위한 VRML2.0 코드를 생성할 수 있도록 일반적인 제품의 VRML2.0 코드구조를 정의하였다. 마지막으로 제품의 조립관계를 일반적으로 표현하기 위하여 Contact, Fit, Screw-Fit관계를 정의하였다. 이와 같은 부 시스템을 이용하여 제품구성 표현과 조립 애니메이션을 위한 VRML2.0 코드를 자동으로 생성하는 시스템의 프로토타입을 개발하였다.

현재 VRML을 통한 정보의 시각화에 관한 연구는 그 필요성이 점차 증대됨에 따라 여러 분야에서 진행되고 있다. Rohrer & Swing[2]은 추상화된 데이터 및 정보를 3차원 공간 데이터로 변환하기 위한 방법을 연구하였으며, 구체적으로 Hierarchical information, Networks, Content-based document clustering, Visual Web search, Information space metaphors 등의 프로토타입을 제시하였다. Casher et al. [3]은 VRML을 기반으로 한 분자결합구조의 3차원 도시 방법을 제시하였다. Ressler et al. [4]은 제조산업에서 발생하는 데이타를 VRML을 통하여 3차원 공간상에 표현하는 방법을 연구하였으며, Osawa et al. [5]은VRML2.0의 애니메이션 기능을 이용하여 병렬 컴퓨터 시스템에 대한 분석과 Debugging에 관하여 연구하였다. 김 영호 [6]는 형상정보를 공유하기 위한 월드 디자인 뷰 시스템을 개발하여 제품 설계를 위한 Visual 프로토타입을 제안하였다.

본 연구는 VRML2.0의 형상표현과 애니메이션 기능을 제조분야에 적용하여 제품의 형상과 조립 과정을 3차원 애니메이션으로 표현할 수 있는 VRML코드를 자동 생성하기 위한 시스템을 개발하는 것이다. 제안된 시스템을 이용하면 CAD 및 BOM 등 기업의 기존 정보를 이용하여 제품에 관한 정보를 공중망을 통하여 쉽게 제공할 수 있다는 장점이 있다.

2. 시스템 설계

제안시스템은 제품의 BOM 정보와 각 구성 부품의 CAD정보를 이용하며, 부품의 초기 위치와 최종 조립형상을 입력으로 받아들여, 제품조립을 시각적으로 표현하는 VRML 코드를 생성하는 시스템이다. VRML 코드 자동생성 시스템은 입/출력을 위한 DB, 제품의 조립형상 VRML 코드 생성 모듈, 3차원 조립 애니메이션 VRML코드 자동 생성 모듈로 구성되어 있다. 시스템 전체 구성도 그림1에 나타나있다.

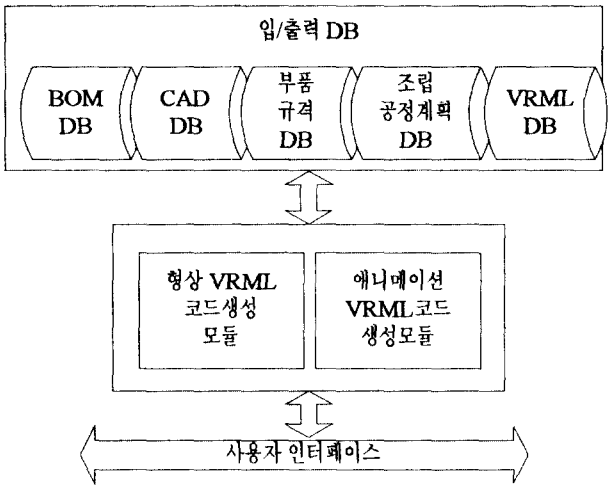


그림1. 시스템 전체 구성도

2.1 입력 데이타베이스

본 시스템은 CAD 및 BOM등 기업의 기존 정보를 입력으로 이용하기 때문에 입력 자료를 준비하기 위한 노력이 불필요하다는 특징을 갖고 있다. 본 시스템의 입력에 필요한 정보는 BOM, VRML, CAD, 부품의 규격, 조립 공정계획 등 5개의 데이타베이스로 구성되어 있다. BOM DB는 VRML코드의 구성형태를 나타내고 시스템이 VRML 코드구조를 생성할 수 있도록 해준다. VRML DB는 시스템에서 생성된 VRML코드를 저장하고 시스템의 호출에 따라 VRML 코드를 제공한다. CAD DB는 제품의 3차원 형상 DXF(Drawing Interchange File Format)가 저장되어 있어 필요에 따라 VRML데이터를 얻기 위하여 호출된다. DXF형식이란 모든 CAD 시스템간, 또는 모든 컴퓨터 기종간의 도면 정보 교환을 돕기 위한 중립형식의 Text 파일이다. 부품규격 DB에는 부품의 주요한 특징이 기술되어있다. 끝으로 조립 공정계획 DB에는 조립표현 즉 작업방법과 조립 순서가 저장되어있다.

2.2 형상 VRML 코드생성 모듈

BOM과 DXF정보를 이용하여 제품의 형상 VRML코드를 생성하는 모듈이다. 이를위하여 첫째 일반적인 BOM Tree로부터 각 부품의 종속관계를 표현할 수 있는VRML구조를 생성하기 위하여 그림2 처럼 BOM의 부품명과 VRML의 Transform Node와 일대일 대응관계를 가질 수 있도록 하였으며, 둘째 그 구조가 생성되면 각 단품들의 VRML 코드를 획득하기 위하여 해당 DXF 정보를 VRML코드로 변환시킨다.

2.2.1 제품구성의 VRML 표현

일반적으로 제품의 구성형태는 BOM에 기술되어있다. 그림 2에 표현되어있듯이 BOM

Tree는 각 부품간의 종속관계를 구조적으로 표현해준다. BOM Tree의 Leaf 노드는 부품을 의미하고 중간 노드는 중간 어셈블리, 루트노드는 제품을 의미한다. 이런 BOM의 구조적인 종속관계는 VRML 코드를 생성하는데 있어서 각 부품에 해당되는 VRML 코드간의 종속관계 구조 생성에 근간이 된다. 또한 BOM과 유사한 트리 형태를 갖는 VRML 코드 Tree구조는 조립 애니메이션 구현에 있어서 각 부품간의 이동이나 회전의 종속관계를 명확히 할 수 있는 장점이 있다.

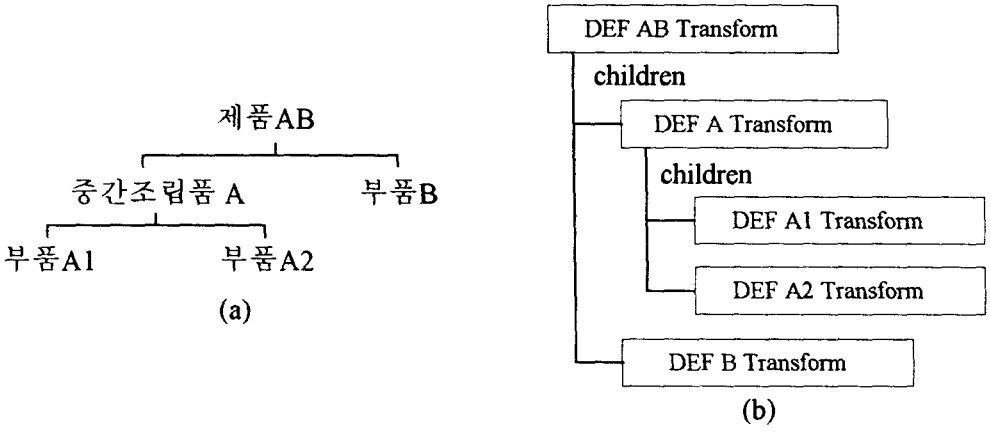


그림2. (a) BOM Tree와 (b) VRML Tree

일반적인 BOM 데이터베이스로부터 그림2와 같은 VRML Tree를 구하는 알고리즘을 기술하기 위하여 다음과 같이 변수, 객체클래스, 서브함수를 정의한다.

part: 자기자신의 부품명과 부모명 포함하는 Structure
List: part의 정보를 가지고있는 선형리스트 클래스의 인스턴스
position: List에서 현재 위치
List.AddTail(): 리스트에 원소를 추가하는 List클래스의 메소드 함수
List.GetAt(): 리스트 특정위치의 정보 획득(List 클래스의 메소드 함수)
List.GetFirst(): 리스트 처음에 위치한 정보 획득(List 클래스의 메소드 함수)
List.GetCount (): 리스트 개수(List 클래스의 메소드 함수)
List.DeleteFirst (): 처음 리스트 삭제(List 클래스의 메소드 함수)
Make_Node (): 노드 생성
Insert_ChildNode (): 자식으로 노드를 생성
Query_BOM_Database (): 매계 변수를 부모로 하는 부품 획득
Find_Parent_Position (): 생성된 Node 중에서 매계 변수와 일치하는 위치

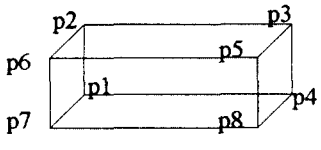
Algorithm to convert BOM to VRML

```
part = VRML 코드를 생성하고자 하는 제품명;
position = 0;
List.AddTail(part);
```

```
while (position == List.GetCount())
{
part = List.GetAt (postion);
List.AddTail (Query_BOM_Database (part));
position ++;
}
while (List.GetCount () != 0)
{
part = List.GetFirst ();
if (part.parent ==none) Make_Node (part);
else Insert_Child_Node (Find_Parent_Position (part),part);
        List.DeleteFirst ();
}
```

2.2.2 VRML 형상 표현

제품 구성 표현을 위한 VRML 구조가 완성되면 각 단말 노드에 해당하는 부품의 DXF정보로부터 VRML코드를 생성하여야 한다. DXF형식의 3차원 형상을 적절한 Module을 통해 CAD의 부품형상을 VRML형식으로 변환시킴으로서 기업이 소유하고 있는 기존 데이터를 재활용할 수 있다. VRML 형상에 관한 구조는 그림 3에 예시 되어있듯이 C언어의 구조체 형식과 비슷한 구조로 정의된다. 즉 Shape 이라는 Node에 형태를 담고 있는 geometry 필드가 있으며 그 필드는 IndexedFaceSet 노드로 정의 되어있다. 그리고 다시 이것은 coord와 coordIndex필드로 나뉘어져 있다. 아래 그림3은 임의의 직육면체의 DXF 형식으로 표현된 화일의 일부분과 DXF화일이 다시 VRML 형식으로 변환된 화일을 도시한 것이다.



(a)

```
.. .. ..
CONTINUOUS
62
3
10
p1x
20
p1y
30
p1z
 ..
CONTINUOUS
 ..
30
p4z
 ..
SEQEND
 ..
```

(b)

```
Shape {
    geometry IndexedFaceSet {
        coord Coordinate {
            point [p1,·····,p8]
        }
        coordIndex [ 0,1,2,3,-1,···-1]
    }
}
```

(c)

그림3. (a) 직육면체 와 (b) DXF화일 형식 그리고 (c) VRML 형식

그림 3(b)와 같은 직육면체 형상에 대한 DXF형식을 그림 3(c)와 같은 VRML형식의 정보로 바꾸기 위하여 DXF의 X,Y,Z좌표를 나타내는 Group Code 10, 20, 30 과 한면을 표현하는 SEQEND Entity 그리고 VRML2.0의 IndexedFaceSet 노드를 이용한 변환 알고리즘을 다음에 기술하였다.

dxf: DXF 텍스트 파일
dxf_index: XYZ 좌표와 SEQEND문자열을 가지는 선형리스트의 인스턴스
index_table: XYZ좌표를 가지는 선형리스트의 인스턴스
point: VRML의 Coordinate 노드의 point 필드를 나타내는 문자열
coord_index: VRML의 IndexedFaceSet노드의 CoordIndex필드를 나타내는 문자열
DXF_Filter (): dxf의 문자열로부터 X, Y, Z좌표와 SEQEND를 추출하는 함수
DXF_Coord_Filter (): dxf의 문자열로부터 중복된 좌표와 SEQEND를 삭제하는 함수
IndexTable_To_VRML_PointSet(): index_table로부터 VRML의 point 필드에 적합한 형태로 변환한다.
DXF_SEQEND_To_VRML_Index(): DXF의 XYZ좌표집합을 index_table과 비교하여 VRML의 coord_index 필드에 적합한 형태로 변환한다.
Make_Shape(): 매개변수를 통해서 VRML의 Shape 노드를 완성한다.

```
Algorithm to convert DXF to VRML
dxf_index = DXF_Filter (dxf);
index_table = DXF_Coord_Filter (dxf_index);
point = IndexTable_To_VRML_PointSet (index_table);
coord_index = DXF_SEQEND_To_VRML_Index (index_table, dxf_index);
Make_Shape (point, coord_index);
```

2.3. 애니메이션 VRML 코드생성 모듈

부품 조립 애니메이션을 위한 VRML코드를 생성하기 위해서는 부품의 초기위치, 최종조립 제품의 구성 및 조립관계, 조립순서, 제품규격 등의 정보가 명시적으로 표현되어야 한다. 본 시스템은 형상VRML코드생성모듈에서 유도된 제품의 형상VRML 코드와 조립 공정계획으로 부터 최종 조립제품의 위치, 조립관계, 조립순서에 관한 정보를 얻도록 하였으며, 부품의 초기위치는 사용자로부터 입력 받도록 구현하였다. 이러한 애니메이션을 구현하기위하여 VRML의 Interpolator NODE를 이용하였다.

2.3.1 애니메이션 VRML코드 Structure

애니메이션은 형상표현과는 달리 각 부품의 이동경로도 함께 표현 되어야 한다. 이를 위해 VRML이 제공하는 Interpolator 기능을 사용하여 애니메이션을 구현하였다. 일반적으로 VRML에서 사용되는 애니메이션은 크게 세 부분으로 나누어 진다. 첫째로 시작과 애니메이션 시간을 관리하는 부분, 둘째로 각 구성품의 형상과 위치와 회전에 관한 분분, 마지막으로 각각의 구성품에 시간의 흐름과 이벤트를 전달해 주는 Route 부분으로 나누어진다. 그림4에 일반적인 조립 애니메이션 VRML코드 구조를 정의하였다.

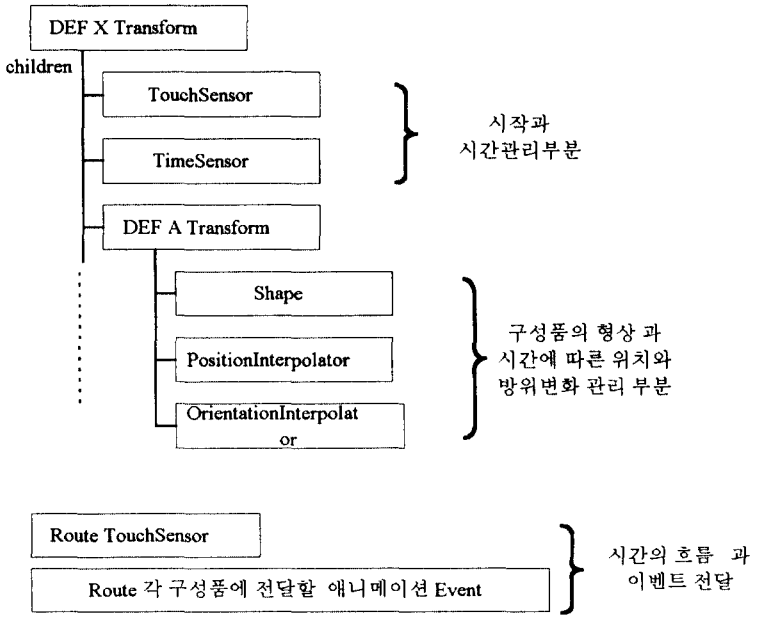


그림4. 애니메이션 VRML Structure

2.3.2 조립관계 표현

조립 관계는 최종 조립 제품의 구성품 사이의 접합 관계를 명시한다. 그 중 대표적인 것으로는 Contact, Against, Slot-Fits, Groove-Fits, Snap-Fits, Screw-Fits 등이 있다[7]. 본 연구에서는 이들 중 조립 애니메이션 관점에서 부품사이의 접합 관계를 표현하기 위하여 Contact, Fits, Screw-Fits등을 정의 이용한다. 아래에 각각에 대해 간단히 설명하였으며, 아래 그림5에 그림으로 표현하였다.

Contact (A, B): A와 B가 Contact관계에 있다는 것은 조립 시 A 와 B가 Contact면을 기준으로 움직일 수 있다는 것을 의미한다.

Fits (A,B): 두개의 부품 중 어느 하나는 오목, 다른 하나는 볼록 형태를 가진 부품으로 수직적인 조립형태를 갖는다.

Screw-Fits (A, B): 두개의 부품 중 어느 하나는 오목, 다른 하나는 볼록 형태를 가진 부품으로 수직적이며 회전의 조립형태를 갖는다.

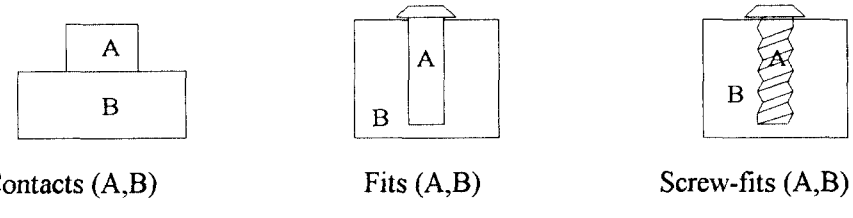
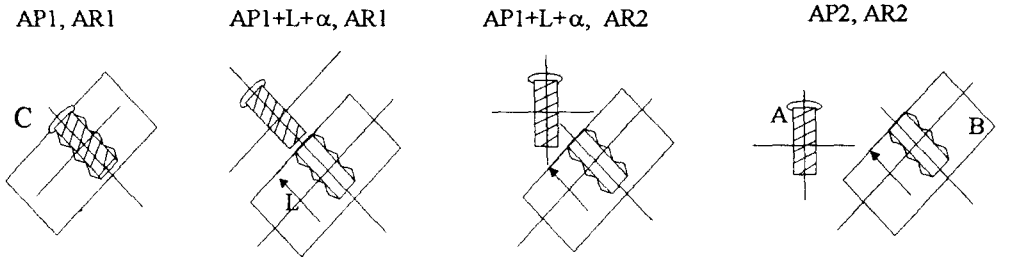


그림5. 조립 관계 표현

2.3.3 애니메이션 VRML 코드생성

애니메이션에 관련된 자료는 조립 순서, 조립 관계, 제품의 규격, 최종 제품의 구조적인 VRML 코드 그리고 각 부품의 초기상태로부터 유추해낼 수 있다. 본 절에서는 VRML의 Interpolator NODE를 이용하여 부품 조립의 중간 경로생성방법을 살펴보고 이를 바탕으로 구체적인 알고리즘을 기술하기로한다.

아래 그림 6에 Contact와 Fits의 특징을 모두 포함하는 조립을 고려하여 보자. 시간 t1부터 시작하고 T 시간의 조립시간을 갖는 Screw-Fits<A,B>의 제품 C 경우를 생각해보자. 제품 C 애니메이션에 관한 자료를 얻기 위해 제품을 분해 순으로 생각해보면, 첫째 제품 C의 조립 VRML Code로부터 각 부품의 조립된 위치와 각도를 알 수 있다. 둘째 조립 관계가 Screw-Fits<A,B>인 것 으로 보아 나사산이 존재함을 알 수 있으므로 조립 시에 부품 A의 회전이 필요함을 알 수 있다.



*Legends: A, B, C는 부품명, P는 Position, R은 Orientation

그림 6 조립 애니메이션VRML 코드 유도

셋째 그 회전량과 이동거리는 제품의 규격 L로부터 알 수 있다. 넷째 조립기구(예 : 로봇 이나 크레인)에 따른 부품의 이동경로를 알 수 있다. 따라서 위에 기술한 정보를 역순으로 이용하면 초기상태로부터 최종 제품의 형상으로 변해가는 경로를 유추해낼 수 있으며 VRML의 Interpolator NODE를 이용하여 코드를 생성할 수 있다.

그림 6(a)와 (d)의 초기 입력 조건 (즉, 초기위치와 최종위치)으로부터 아래 그림 7(a)와 같은 애니메이션 코드를 생성할 수 있으며, 그림6(b)와 (c)의 중간 조립 경로 1과 2로부터 얻은 정보를 추가하여 그림 7(b)와 같은 애니메이션 코드를 생성할 수 있다.

```
PositionInterpolator
{
    key[t1, t4]
    keyValue[AP2, AP1 ]
}
OrientationInterpolator
{
    key[t1, t4 ]
    keyValue[AR2, AR1 ]
}
```

(a) 초기 입력상태

```
PositionInterpolator
{
    key[t1,t2,t3,t4]
    keyValue[AP2, AP1+L+  α, AP1+L+ α, AP1 ]
}
OrientationInterpolator
{
    key[t1, t2, t3, t4 ]
    keyValue[AR2, AR2,  AR1, AR1 ]
}
```

(b) 조립경로 추가

그림 7 Interpolator를 이용한 애니메이션 코드

Interpolator를 이용하여 애니메이션 코드를 생성하는 알고리즘을 다음에 자세히 기술한다. 또한 알고리즘을 기술하기 위하여 요구되는 변수와 서브함수들에 대하여도 정의하였다.

pos_ori : 위치와 방위를 갖는 구조체배열

t: 조립시작시각

time : 조립시간

Read_VRML_Final(): 조립될 부품의 최종 위치를 읽는다.

Read_Relation(): 에니메이션에 이용될 조립 관계를 읽는다.

Read_Regulation(): 제품의 규격을 읽는다.

Cal_Contact(): 기울어진 방위와 제품의 규격으로부터 충돌이 생기지 않을 만큼 이동한다.

Cal_Fit(): 기울어진 방위와 제품의 규격으로부터 규격만큼 이동된 위치를 구한다.

Cal_Screw(): 기울어진 방위와 제품 규격으로부터 규격만큼 이동된 위치와 회전정도를 구한다.

Cal_Orientation(): 조리될 부품의 방위를 Root Node제품의 방위와 일치 시킨다.

Read_Root_ Orientation ():Root Node 제품의 방위 방위를 읽는다.

Read_ VRML_Init(): 조립될 부품의 초기 위치를 읽는다.

Add_P&O(): 시간과 위치, 방위를 Interpolator노드에 추가시킨다.

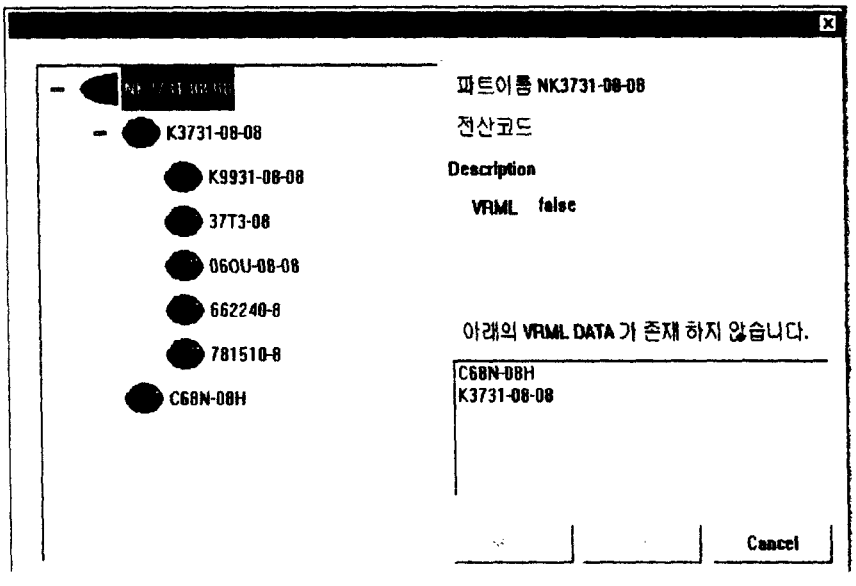
Algorithm to Generate Animation VRML

```
index = 0;
pos_ori[index++] = Read_VRML_Final()

if(Read_Relation() == Contact) pos_ori[index++ ] = Cal_Contact(pos_ori [index-1],
Read_Regulation());
if(Read_Relation() == Fit) pos_ori [index++ ] = Cal_Fit(pos_ori [index-1],
Read_Regulation());
if(Read_Relation() == Screw_fit) pos_ori [index++] = Cal_Screw(pos_ori [index-1],
Read_Regulation());
```

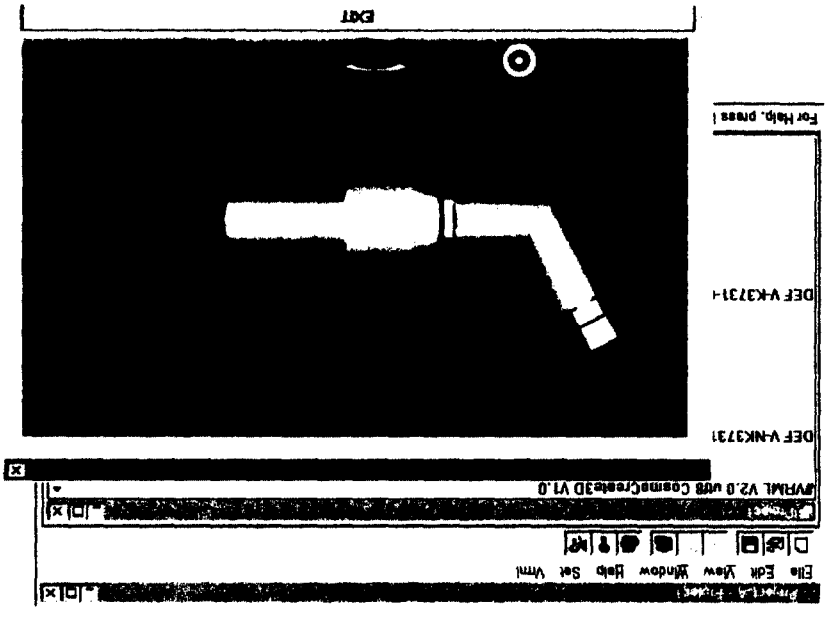
```
pos_ori[index++] = Calculate_Ori(pos_ori[index-1]);
pos_ori[index] = Read_ VRML_Init():

While(index>=0) {
Add_P&O(pos_ori[index--],Time, t);
}
```

3 시스템 구현

본 시스템은 사용자가 제품에 대한 형상 또는 조립 애니메이션 VRML 코드를 얻고자 할 때 기업에서 소유하고 있는 기존의 데이터를 활용 및 변환하여 사용자가 원하는 VRML2.0코드를 제공할 수 있도록 구현되었다. 이 시스템은 크게 세 부분으로 구현되어 있다. 첫번째 MSSQL Server를 이용한 데이터 입력부, 두번째 제품 형상 VRML코드 생성, 세번째로 제품의 애니메이션 VRML코드 생성으로 이루어져있다. 이는 윈도우 NT 4.0 환경에서 VISUAL C++ 5.0, Netscape Internet Control, Cosmo Player 1.0 Plugin그리고 MicroSoft SQL Server 6.5를 이용하여 구현하였다 그림 8(a)는 VRML Tree를 나타내고 있으며 그림 8(b)는 조립 애니메이션 코드와 애니메이션 실행 장면이다.



(a) VRML Tree



(b) VRML 조립 애니메이션
그림 8. 시스템 구현 화면

4. 결론 및 향후 연구과제

이 시스템은 제품의 종류나 형상에 관계없이 제품형상의 CAD, 제품구성의 BOM, 조립방법의 조립 공정계획을 이용하여 제품형상 표현과 조립 애니메이션을 위한 VRML코드를 자동생성하는 시스템을 개발하였다. 이는 가상공간에서 제품 관련 팀 또는 팀 구성원들의 제품 형상정보, 구성정보 및 3차원 애니메이션을 통한 제품 형상 및 조립에 관한 정보를 쉽고 빠르게 생성, 획득할 수 있게 하여 병렬적인 제품 개발이 가능하게 한다. 그러므로 동시공학(Concurrent Engineering)의 도구로서도 이용 가능하며 또한 실제 실험이 불가능한 선박, 플랜트 등의 대형 사업장에서부터 자동차를 포함하는 정밀기계의 조립방법에 대한 시각적인 평가도구인 그래픽 시뮬레이터로 활용될 수 있을 것이다.

좀더 나은 정보공유와 시스템 효율을 높이기 위해서는 Java와 STEP과의 연계를 통한 직접적인 Internet진출과 폭 넓은 개방성에 대한 연구가 필요하다.

참고문헌

[1] Silicon Graphics, The Virtual Reality Modeling Language Specification http://vrml.sgi.com/moving-world/spec/, 1997.

[2] Rohrer, Randall M. and Swing, Edward, Web-Based Information Visualization IEEE Computer Graphics and Application Information Visualization, pp52-59, July/August 1997

[3] Casher, O., Leach, Christopher, Page, Christopher S. and Rzepa, HenryS.,Advanced VRML-Based Chemistry Applications : 3D Molecular Hyperglossary 2nd ECC Conf., http://www.ch.ic.ac.uk/eccc2/, 1995.

[4] Ressler, S. et al. Using VRML to Access Manufacturing Data, Proc.of VRML97, ACM Siggraph, ACM Press, New york 1997.

[5] Osawa, N., Morita, H., and Yuba, T.,Animation for Performance Debugging of Parallel Computing Systems, Proc.of VRML97, ACM Siggraph, ACM Press, New york 1997.

[6] 김영호 제품 설계를 위한 Visual Prototyping 기술 제조분야의 효과적인 CALS 구축 위한 기술세미나 발표집, 1997.

[7] Ambler, A. P, and Popplestone, R. J., Inferring the Position of Bodies from Specified Spatial Relationships, Artificial Intelligence