협동설계시스템의 솔리드 모델러를 위한 오브젝트의 Picking과 Concurrency

윤보열* · 김응곤**

Object Picking and Concurrency for Solid Modeler in Collaborative Design System

Bo-yul Yoon* · Eung-kon Kim**

요 약

여기서 제시된 협동설계시스템은 인터넷망과 웹브라우저를 이용하여 공유된 가상 공간에서 협동작업이 이루어진다. 협동설계 참여자들은 인터넷을 통해 솔리드 모델러 서버에 접근하여 원하는 3D 도형을 생성하고 조작한다. 이때 공유된 여러 오브젝트 중 임의의 오브젝트를 선택하는 picking 문제와 여러 참여자들이 한 오브젝트에 어떤 조작을 가할 때 제어하는 concurrency 문제가 중요하게 대두된다.

본 논문에서는 오브젝트의 picking이 마우스 포인터에서의 투사되는 선과 오브젝트간에 겹치는 부분을 계산하는 방법 외에 Java 3D를 이용하여 scene graph의 노드에 picking 속성을 주는 방법, bounds를 설정하는 방법, picking test의 범위를 한정하는 방법을 사용하여 계산하는 부담을 줄이고 효과적인 picking이 이루어지도록 한다. 이어서 picking된 오브젝트에 대해 협동설계에 참여한 사람이 공유작업공간에서 action에 따라 shared lock과 exclusive lock을 사용하여 효과적인 동시성제어가 이루어지도록 한다

ABSTRACT

We are able to work on the shared virtual space in Web-based Collaborative Design System using only Internet and Web browser. The users connect to the Solid Modeler Server through WWW and they create 3D shape and manipulate them variously. Then the users will share 3D objects and two problems can arise. The users must be able to pick the objects effectively which they want to manipulate. When one of the users manipulates a particular object, others should not disturb with the same object.

In this paper, picking is implemented not only by computing intersection of mouse pointer with the objects of the virtual world, but also by using capabilities and attributes of scene graph node, by setting bounds intersection testing instead of geometric intersection testing, by limiting the scope of the pick testing, using Java 3D. These methods can reduce the computation of picking and can pick 3D objects effectively and easily using the system of hierarchy. To have effective concurrency, we used shared lock and exclusive lock as the action in work space.

키워드: CAD, 솔리드모델러, 협동설계, picking

* 순천대학교 컴퓨터그래픽스 연구실
** 순천대학교 컴퓨터과학과

접수일자 2001. 5. 20

I. 서론

전통적인 협동작업은 일정한 시간에 일정한 장소에서 함께 만나 서로 의견을 주고받으며 진행되었다. 오늘날에는 컴퓨터와 통신 기술의 발달로 시간과 공간의 제약 없이 공유된 가상 공간에서 상호작용을 하면서 효율적으로 작업하는 새로운 시스템이 대두되고 있다[1,2].

이러한 협동작업시스템으로는 화상회의, 공동프로그래밍, 전자결재, 원격교육, 원격 진료시스템 등이 있다. 대부분의 시스템은 특정한 플랫폼과 그룹웨어를 사용하여 폐쇄적으로 공동작업이 이루어지는 경우가 많고, 윈도우의 탐색기 형태를 취하고 있어서 공유 오브젝트가 폴더나 문서, 메모 등에 그치고 있다[3].

우리가 제시한 협동설계시스템은 플랫폼에 구애받지 않는 인터넷 상에서 웹브라우저를 통해 서버에 접근하여 협동작업이 이루어지도록 되어 있으며, 가상공간에서 협동설계작업을 수행하기 때문에 공유 오브젝트가 3D 도형이 되고 있다. 협동설계서버에 솔리드 모델러가 있어서 설계에 참가한 사람들은 기본 오브젝트를 쉽게 생성하고 조작할 수 있다. 이때 공유된 여러 오브젝트 중 임의의 오브젝트를 선택하는 picking 문제와 여러 참여자들이 한 오브젝트에 어떤 조작을 가할 때 제어하는 concurrency 문제를 효과적으로 해결해야 한다.

오브젝트의 picking이 마우스 포인터에서의 투사되는 선과 오브젝트간에 겹치는 부분을 계산하는 방법뿐만 아니라 Java 3D API를 이용하여 scene graph의 노드에 picking 속성을 주는 방법, 오브젝트에 bounds를 설정하는 방법, 오브젝트의 picking test의 범위를 한정하는 방법을 사용하여 계산하는 부담을 줄이고 효과적으로 이루어지도록 한다.

이어서 picking된 오브젝트에 대해 협동설계에 참가한 사람이 공유작업공간에서 여러 가지 조작을 하게 되는데, action에 따라 shared lock과 exclusive lock을 사용하여 효과적인 동시성제어가 이루어지도록 한다.

본 논문의 구조는 II장에서 웹기반 협동설계시스템의 전체구조와 솔리드 모델러를 소개하고, III장에서는 intersection과 3D scene graph 노드를 이용한 효과적인 picking 방법을 제시하고, IV장에서는 적절한 locking을 사용한 concurrency를 다루고, V장에서는 요약 및 결론을 맺는다.

II. 협동 설계시스템의 구조

시스템은 클라이언트/서버 구조로 클라이언트는 자바 애플릿을 통해 웹 상에서 접근하고 서버는 자바 애플리케이션으로 접속을 통제하는 접속관리자, 작업 그룹의 동기화를 유지하며 공유된 작업 공간을 확보하는 작업관리자, 그리고 3차원 도형을 그릴 수 있는 솔리드 모델러로 이루어져 있다. 그림1은 협동설계시스템 전체 구조를 보여주고 있다[3].

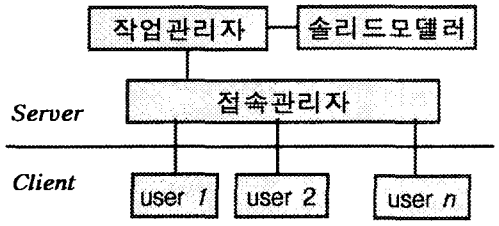

그림1. 전체 시스템의 구조

접속관리자는 웹서버를 통하여 들어온 클라이언트의 서비스 요청을 받아 분석하여 메시지로 세션관리자에게 보낸다. 사용자의 ID에 따라 접속을 설정하거나 해제할 수 있고, 작업하는 동안 클라이언트의 접속을 계속 유지시키는 역할을 한다.

작업관리자는 클라이언트의 작업 요청에 따라 실제적인 작업을 처리한다. 개인작업공간과 공유작업공간을 확보하도록 하고 도형 객체를 생성하거나 저장된 파일을 불러와 변형시킨다.

솔리드 모델러는 Java 3D를 이용하여 개발하며, 시스템 라이브러리는 여러 가지 Java 클래스들로 구성되며 시스템 레벨의 클래스는 그림2와 같다.

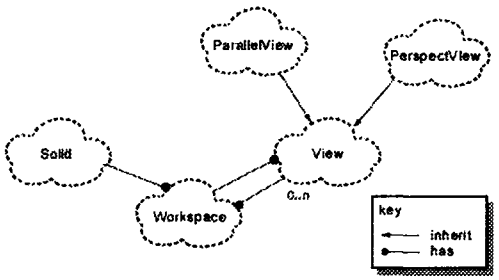

그림2. 솔리드 모델러의 시스템라이브러리

솔리드의 기본입체로는 육면체, 원기둥, 원뿔, 토러스, 피라미드, 구 등이다. 이들 기본입체에 대하여 INTERSECT, DIFFERENCE, UNION과 같은 부울리안 연산을 수행하여 솔리드 모델을 만들 수 있게 된다. 솔리드 객체에 대하여 TRANSLATION, SCALING, REFLECTION, ROTATION과 같은 변환을 수행할 수 있다. 솔리드 모델러에서 오브젝트를 만들거나 불러 와서 변형을 시키는데, 여러 개의 오브젝트 중 어느 것을 선택하여 조작하려고 할 때 오브젝트 picking이 이루어져야 한다.

III. 오브젝트의 Picking

1. intersection을 통한 오브젝트의 picking

picking이란 화면 내부의 물체 혹은 물체의 일부분을 구성하는 메쉬 등을 선택하는 것이다[4]. 이는 포인팅 디바이스 등으로 어떤 오브젝트를 선택하여 인터액션이 이루어지도록 하기 위한 것으로 그림3처럼 마우스 포인터 위치에서 내부 가상 세계에 광선을 쏘아 오브젝트와의 겹치는 부분(intersection)을 찾아내는 것이다[5].

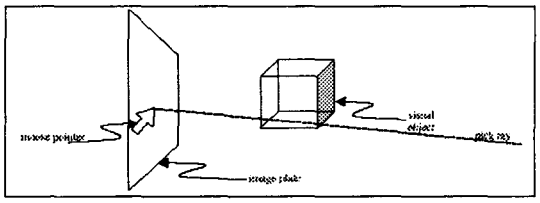

그림3. object에 대한 picking ray의 투사

마우스 포인터(x,y)와 오브젝트(x1,y1)와의 거리(D)를 구해 일정한 범위(APERTURE) 내에 들어있는지를 확인한다. 그림4의 왼쪽은 점(x1,y1)이 i)의 경우로 picking이 안되고, 오른쪽 그림은 ii)의 경우로 점(x1,y1)이 picking이 된다[6].

$$D2=(x-x1)2 + (y-y1)2$$

i) $(x-x1)2 + (y-y1)2 > APERTURE2$

ii) $(x-x1)2 + (y-y1)2 < APERTURE2$

앞에서 살펴본 D < APERTURE 를 확인하는 것은 셈이 많고 제곱근을 계산해야 하므로 연산시간이 많이 걸린다. 따라서 그림5처럼 점(x,y)를 중심으로 한 변의

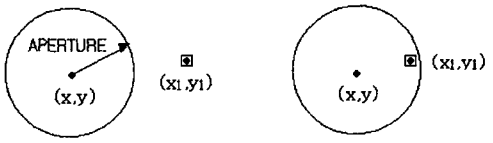

그림4. 오브젝트가 점인 경우의 picking

길이가 2×APERTURE인 정사각형의 내부에 점(x1,y1)이 들어오는 가를 확인한다.

$$|x-x1| + |y-y1| < APERTURE$$

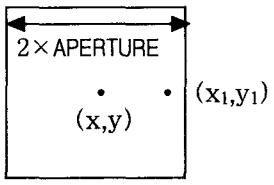

그림5. 정방형에 의한 테스트

다음은 한 점과 선분과의 거리를 구해 일정한 영역 안에 있으면 picking한다. 주어진 한 점에서 선분에 수선을 그어 그 교점과 주어진 점과의 거리를 구한다. 한 점(x1,y1)과 선분 ax + by = 0 와의 거리 D는 다음과 같다.

$$D=\frac{|ax_1+by_1+c|}{\sqrt{a^2+b^2}}$$

또 주어진 한 점(x,y)과 두 점 (x1,y1), (x2,y2)를 양 끝점으로 갖는 선분과의 거리 D는 다음과 같다.

$$D=\frac{|(x-x_1)(y_2-y_1)-(y-y_1)(x_2-x_1)|}{\sqrt{(x_2-x_1)^2+(y_2-y_1)^2}}$$

따라서 picking이 되기 위해서는 그림6처럼 선분이 원 안으로 들어와야 하므로 다음과 같은 조건을 만족하여야 한다.

$$\frac{((x-x_1)(y_2-y_1)-(y-y_1)(x_2-x_1))^2}{(x_2-x_1)^2+(y_2-y_1)^2}<A^2$$

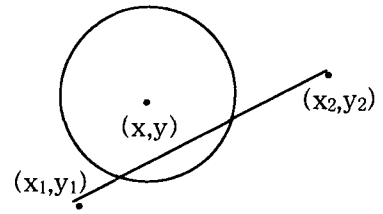

그림6. picking된 선분

많은 곱셈 연산을 피하기 위해 그림7은 원 대신에 정사각형을 사용하였다.

$$\min(|m(x-x_1)+y_1-y|,|\frac{(y-y_1)}{m}+x_1-x|)<A$$

(단, $m=\frac{y_2-y_1}{x_2-x_1}$)

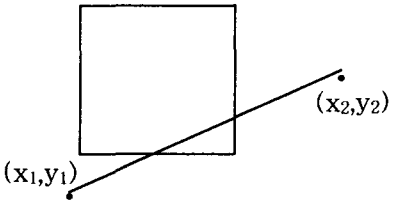

그림7. 정사각형을 통한 picking

어떤 오브젝트의 내부에 마우스를 클릭하면 그 오브젝트가 선택되었다고 하는데, 하나의 레이어에서 두 오브젝트의의 중첩된 부분을 클릭하였을 때는 어떤 오브젝트를 picking하게 되는지 문제가 된다. 이 경우에는 클릭한 좌표에서 가장 가까운 물체가 picking되도록 한다. 그림8에서는 점과 가까운 거리에 빗변이 놓인 B가 선택된다. 그림9는 pick window를 사용하여 picking 하고자하는 부분을 쉽게 볼 수 있도록 한다[4].

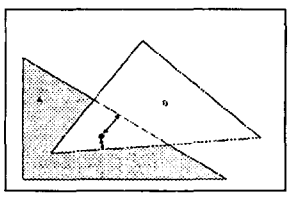

그림8. 중첩된 부분의 picking

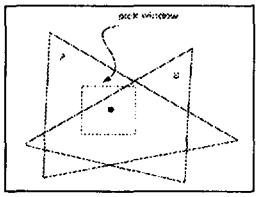

그림9. pick window

3.2. 노드를 이용한 3D 오브젝트의 picking

3D 공간에서의 picking은 Z축으로 가장 앞에 놓인 오브젝트를 picking하도록 한다. 그림10은 마우스에서 ray를 쏘아 첫 번째 비행기의 오브젝트를 picking하는 것을 보여 준다[7].

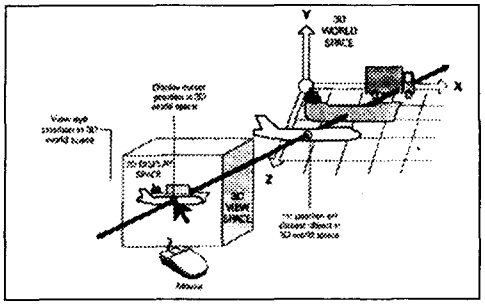

그림10. 3D World의 object picking

앞에서 언급한 방법들은 컴퓨터 내부적으로 많은 산술연산을 요구하고 있다. Java 3D에서는 그림11처럼 scene graph를 사용하므로 어떤 노드를 picking 할 수 있다.

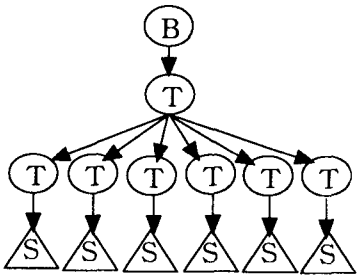

그림11. Java 3D의 scene graph의 예

노드를 picking하는 경우에는 많은 계산을 필요로 하지 않으며 그림12처럼 하위 오브젝트도 따라서 picking되는 효과를 가져오게 된다[8].

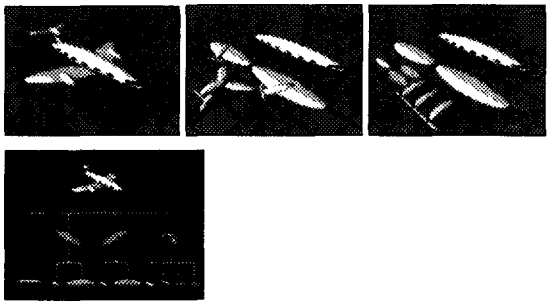

그림12. 하위 오브젝트의 picking

Java 3D를 이용하여 3차원 가상 공간에서 효과적으로 오브젝트를 picking하기 위해 다음과 같은 방법을 사용한다.

첫째 방법은 scene graph node의 attribute와 capability를 이용하는 것으로, 노드에 setPickable() 메소드를 사용하여 true와 false값에 따라 선택한 노드의 하위구조까지도 intersection연산을 하지 않도록 한다.

Node Method

extends: SceneGraphObject
subclasses: Group, Leaf
void setBounds(Bounds bounds)
//노드의 geometric bounds를 설정한다
void setBoundsAutoCompute(boolean autoCompute)
//노드on/off의 geometric bounds를 자동계산하게 한다
setPickable(boolean pickable)
//true로 설정하면 이 노드는 picking이 되고, false면 이 //노드와 하위노드는 모두 picking되지 않는다

또한 특정 Group Node에만 적용하는 capability설정이 있다.

Node Capabilities

ENABLE_PICK_REPORTING
//디폴트 값으로는 false를 가지며, group노드에서
//사용되고 leaf노드에서는 쓸 수 없다
ALLOW_BOUNDS_READ | WRITE
//bounds information을 read(write)할 수 있게 한다
ALLOW_PICKABLE_READ | WRITE
//pickability state를 read(write)할 수 있게 한다

둘째는 bounds를 설정하는 것으로 복잡한 도형에 대해 일정한 영역을 설정하여 연산을 빠르고 간단하게 하도록 한다.

PickBounds

PickBounds()
// PickBounds 생성
PickBounds(Bounds boundsObject)
// 특정한 파라미터를 주어 PickBounds 생성
Bounds get()
// PickBounds로부터 boundsObject를 얻는다
void set(Bounds boundsObject)
// boundsObject를 이 PickBounds 안으로 설정한다

셋째는 pick testing의 scope를 한정하여 intersection을 찾기 위해 모든 부분에 대해 연산하는 것을 줄일 수 있다.

다음은 마우스를 이용하여 3D ColorCub의 scene graph의 노드를 picking하는 Java 3D 소스코드의 일부분이다[5]. 13-15번 줄에서 capability를 세팅하는데 picking이 가능하도록 ENABLE_PICK_REPORTING 한다. 20번 줄에서 root, canvas, bounds를 파라미터로 주고 picking behavior object를 생성한다. 21번 줄에서 behavior object를 scene graph에 붙인다. 27-29번 줄에서 scene graph object를 위하여 적절한 capability를 가능하게 해준다.

```
1.  public BranchGroup createSceneGraph(Canvas3D canvas) {
2.  // branch graph의 root 생성
3.  BranchGroup objRoot = new BranchGroup();
4.
5.  TransformGroup objRotate = null;
6.  PickRotateBehavior pickRotate = null;
7.  Transform3D transform = new Transform3D();
8.  BoundingSphere behaveBounds = new BoundingSphere();
9.
10. // ColorCube와 PickRotateBehavior objects 생성
11. transform.setTranslation(new Vector3f(-0.6f, 0.0f, -0.6f));
12. objRotate = new TransformGroup(transform);
13. objRotate.setCapability(TransformGroup.ALLOW_TRANSFORM_WRITE);
14. objRotate.setCapability(TransformGroup.ALLOW_TRANSFORM_READ);
15. objRotate.setCapability(TransformGroup.ENABLE_PICK_REPORTING);
16.
17. objRoot.addChild(objRotate);
18. objRotate.addChild(new ColorCube(0.4));
19.
20. pickRotate = new PickRotateBehavior(objRoot,canvas, behaveBounds);
21. objRoot.addChild(pickRotate);
22.
23. // 두 번째 ColorCube를 scene graph에 붙인다
24. transform.setTranslation(new Vector3f( 0.6f, 0.0f, -0.6f));
25. objRotate = new TransformGroup(transform);
26. objRotate.setCapability(TransformGroup.ALLOW_TRANSFORM_WRITE);
27. objRotate.setCapability(TransformGroup.ALLOW_TRANSFORM_READ);
28. objRotate.setCapability(TransformGroup.ENABLE_PICK_REPORTING);
29.
30. objRoot.addChild(objRotate);
31. objRotate.addChild(new ColorCube(0.4));
32.
33. // scene graph에서 최적화한다
34. objRoot.compile();
35.
36. return objRoot;
37. }
```

IV. Action에 따른 Concurrency

공유작업공간에서는 협동작업의 일관된 상태를 유지하면서 그룹의 참여자들이 작업을 수행할 수 있도록 지원해야 한다. 이런 협동 작업의 일관성을 유지하기 위해서는 동기화기법이 필요하다. 멀티 뷰를 통한 협력시스템에서의 뷰 동기화(View Synchronization) 유

지를 위해 참여자의 조작은 서버로 전달되고 서버는 그룹의 각 참여자들에게 브로드캐스팅하므로 WYSI WIS(What You See Is What I See)를 유지하도록 한다[9].

협동 작업의 동시성 제어를 위해 ordering과 locking을 이용하는 방법이 일반적이다. ordering방법은 여러 참여자의 작업을 일정한 순서대로 계속 적용시켜 나간다. locking방법은 공유객체의 작업 권한을 한 참여자에게 줌으로써 일관성을 유지하는 방법이다. 본 시스템에서는 협동 작업의 객체를 순서대로 제어하는 것이 아니라 동시에 제어해야 하므로 후자인 locking방법을 사용한다.

locking은 locked와 unlocked의 두 가지의 상태로 1과 0의 값 중 어느 하나를 가진다. 하나의 lock은 하나의 오브젝트와 연관된다. 오브젝트 X에 관해 lock(X)=1이면 이 오브젝트를 요청한 action은 오브젝트 X에 접근 할 수 없다. lock(X)=0이면 오브젝트 X의 접근이 가능하다.

두 개의 연산 lock과 unlock은 오브젝트에 어떤 action을 취할 때 반드시 포함되어야 한다. 참여자의 어떤 action이 오브젝트 X에 접근하려 할 때에는 먼저 lock(X)연산을 실행한다. 이 때 lock(X)=1이면 이 action은 기다리고, lock(X)=0이면 이 action은 lock(X)의 값을 1로 고정시키고 오브젝트 X에 접근한다. 이 action이 오브젝트 X의 사용이 끝나면 unlock(X)연산을 실행하여 lock(X)의 값을 0으로 고정시킨다. 이 때 다른 참여자나 다른 action이 오브젝트X에 다시 접근할 수 있게 된다[10].

하나의 action만이 특정 오브젝트에 대해 lock을 걸고 사용하기 때문에 협동작업에 있어서는 너무 제약적이다. 어떤 action이 오직 읽기 위해 한 오브젝트에 접근한다면 여러 action이 이 오브젝트에 병행적으로 접근하더라도 문제가 없을 것이다. 그러나 어떤 action이 오브젝트에 대해 변형을 가하거나 저장하려고 한다면 그 action이 오브젝트X에 대해 독점적 접근을 가져야 한다. 읽기만을 위해 병행접근을 허용한다면 두 가지 형태의 lock연산이 필요하다.

① 공용로크 (Shared Lock (lock-S)): 어떤 action이 오브젝트X에 대해 공용로크(Lock-S)를 걸면 이 오브젝트에 대해 읽어 올 수 는 있으나 기록할 수는 없다. 이 때 이 오브젝트X에 대해 다른 action도 공용lock를 걸 수가 있다.

② 전용로크 (Exclusive Lock(lock-X)) : 어떤 action이 오브젝트X에 대해 전용로크(Lock-X)를 걸면 참여자는 이 오브젝트에 대해 읽고 기록할 수 있다. 이 때 이 오브젝트X에 대해 다른 action은 어떤 lock도 걸 수가 없다.

오브젝트를 생성한 사람만이 삭제하거나 저장할 수 있도록 한다. 오브젝트의 action에 따른 lock table은 표1과 같다.

표1. Object의 Action에 따른 Lock Table

ACTION \ LOCK	lock-S	lock-X
create	○	×
open	○	×
delete	×	○
translation	○	×
zooming	○	×
rotation	○	×
save	×	○

오브젝트 X에 대해 공용lock이 필요한 action은 lock-S(X)연산을 실행해야 하고, 전용lock이 필요한 action은 lock-X(X)연산을 실행해야 한다. 오브젝트 X에 대한 잠금을 풀기 위해서는 unlock(X)연산을 수행하면 된다. 참여자가 오브젝트를 사용할 때 다음과 같은 locking규정을 따르도록 한다.

1. 참여자가 오브젝트 X에 대해 읽거나 조작하고자 할 때는 lock-S(X)나 lock-X(X)연산을 먼저 실행한다.
2. 참여자가 오브젝트 X에 대해 저장하고자 할 때는 lock-X(X)연산을 먼저 실행한다.
3. 참여자가 lock-S(X)나 lock-X(X)연산을 하려고 할 때, 오브젝트 X가 이미 양립될 수 없는 lock이 걸려 있으면 그것이 unlock이 될 때까지 기다려야 한다.
4. 참여자가 모든 action의 실행을 종료하기 전에 lock을 건 오브젝트에 대해 반드시 unlock(X)를 실행해야 한다.
5. 참여자가 lock을 걸지 않은 오브젝트 X에 대해 unlock(X)를 실행할 수 없다.

V. 요약 및 결론

협동설계시스템에서의 솔리드 모델러는 인터넷 가상공간에서 설계 참여자들이 3D도형을 쉽게 생성하고 조작할 수 있도록 해준다. 이 때 많은 오브젝트 중 임의의 오브젝트를 선택하는 picking이 이루어지고, 다수의 설계 참여자가 그 오브젝트에 어떤 조작을 취할 때 여러 사용자의 조작을 제어하는 concurrency 문제가 중요하게 대두된다.

본 논문에서는 Java 3D를 이용해 오브젝트의 picking을 scene graph의 노드에 picking 속성을 주는 방법, bounds를 설정하는 방법, picking test의 범위를 한정하는 방법을 사용하여 computation의 부담을 줄이고 효과적인 picking이 이루어지도록 했다. 이는 계층구조를 가진 3D 도형에서 하위 계층의 모든 노드를 제어할 수 있는 큰 장점을 보여준다.

그리고 picking된 오브젝트에 대해 어떤 조작을 가할 때 협동작업에 참여한 사람의 action에 따라 오브젝트에 공용로크(shared lock)와 전용로크(exclusive lock)를 적절하게 사용하도록 하였다. 이는 참여자들이 여러 오브젝트를 서로 공유하면서 방해받지 않고 효과적인 협동설계가 가능하게 해준다.

감사의 글

본 연구는 과학기술부, 한국과학재단지정 여수대학교"설비자동화 및 정보시스템연구개발센터"지원에 의해 연구되었으며 이에 감사드립니다.

참고문헌

[1] F. Faure, C.Faisstnauer, G.Hesina, "Collaborative animation over the net", IEEE p107-116, 1999.

[2] Icgiro Hagiwara and Shinsuke Noda, "Homotopical Modeling as the Basis of New CAD Standard Homotopy CAD for Collaboration Engineering", IEEE p231-237, 1999.

[3] 윤보열, 김응곤, "웹 기반 협동CAD시스템에 관한 연구", 한국해양정보통신학회 논문지 Vol .4-4, 2000.

[4] 주우석, "3차원 컴퓨터그래픽스", 도서출판 그린, 1999.

[5] Denis J Bouvier, "Getting started with the Java 3D API", Sun microsystems, http://java. sun. com/products/java-media/3D/collateral/, 2000.

[6] 심재홍, "컴퓨터그래픽스",이한출판사, 1996.

[7] Jon Barrilleaux, "3D User Interfaces with Java 3D", manning, p.247-256, 2001.

[8] Henry A. Sowizral, David R. Nadeau, "An Introduction to Programming AR and VR Applications in Java 3D", http://www.sdsc.edu /~nadeau/Courses/VR99/, 1999.

[9] Jonathan Munson and Prasun Dewan, "A concurrency control framework for collaborative system", Proceedings of the ACM 1996 Conference on CSCW, 1996.

[10] 이석호, "데이타베이스 시스템", 정익사, 1995.

윤보열(Bo-yul Yoon)
1984년 전남대학교 학사
1988년 조선대학교 석사
1998년-현재 순천대학교 컴퓨터과학과 박사과정
1984년- 순천매산고등학교 교사
※ 관심분야 : 컴퓨터그래픽스, 멀티미디어, WBI

김응곤(Eung-kon Kim)
1980년 : 조선대학교 공학사
1987년 : 한양대학교 공학석사
1992년 : 조선대학교 공학박사
1984년~1986 : 금성반도체(주) 연구원
1987년~1991년 : 국방과학연구소 선임연구원
1991년~1993년 : 여수대학교 컴퓨터공학과 전임강사
1997년~1998년 : 미국 University of California, Santa Cruz, Post Doc.
1993년~현재 : 순천대학교 컴퓨터과학과 부교수
※ 관심분야 : 컴퓨터그래픽스, 영상처리