4차원 이동객체 데이터베이스를 위한 위상 변화 연산자 설계*

전성우⁰, 지정희, 김상호, 류근호
충북대학교 데이터베이스 연구실
e-mail : {swjun⁰, jhchi, shkim, khryu}@dblab.chungbuk.ac.kr

Design of Topological Development Operators for 4D Moving Object Database

Sung Woo Jun⁰, Jeong Hee Chi, Sang Ho Kim, Keun Ho Ryu
Database Laboratory, Chungbuk National university

요 약

지금까지 이동 객체들 사이의 위상 관계에 관한 연구는 주로 2차원 객체를 대상으로 수행되어 왔고 몇몇 3차원 객체를 대상으로 한 연구에서도 기존의 위상 관계 정의에서 단순히 차원만 확장하였기 때문에 3차원 이동 객체들 사이에서 가능한 특수한 위상 관계의 실제 응용에 적용하기 어렵다. 따라서 이 논문에서는 실세계의 3차원 이동 객체들 사이의 위상관계에 대한 표현력을 높임으로써 실제 응용에 적용할 수 있도록 3차원 공간 위상 관계 연산자와 4차원 위상 변화 연산자를 정의 하고 알고리즘을 설계하였다. 3차원 공간 위상 관계를 효율적으로 나타내기 위하여 3차원 객체들의 x, y 좌표와 z좌표를 분리해서 고려하는 기법을 제안하였고, 4차원 이동 객체들 사이의 가능한 위상 관계 변화를 가시적으로 보여주기 위하여 이동 부피 객체들 사이의 인접 위상 변화 그래프를 제안하였다.

1. 서론

최근 하드웨어 및 소프트웨어 기술의 발달과 Doppler radar, Sensor, GPS 등의 샘플링 기술과 장비의 발달로 인해 모의 전장 분석, 항공 관제 시스템, 위치 기반 서비스 등과 같이 이동객체의 위치정보를 이용한 응용 서비스의 연구가 활발히 진행되고 있다.

그러나 기존의 이동객체 위상관계 연산자에 관한 연구는 주로 2차원 기반에서 수행되어 왔기 때문에 실세계에서 가능한 위상관계를 표현하고 분석하는데 한계가 있었다. 최근에 몇몇 연구자들이 기존의 기본적인 공간 연산자들을 시간과 차원적으로 확장한 4차원 시공간 위상 관계 연산자들을 제안하였으나[1,2,3], 이 연산자들은 이력 객체를 대상으로 설계되거나 단순히 3차원으로 공간 확장과 기존의 시간 위상 개념이 그대로 적용되었기 때문에 4차원 이동 객체들 사이의 특정한 위상 관계에 적용 하기에는 한계가 있다. 예를 들어, 4차원 이동객체 데이터베이스에서는 어떤 객체가 다른 객체를 통과해서 지나가는 경우와 통과하지 않고 위로 지나가는 경우, 그리고 통과하지 않으면서 옆으로 지나가는 경우 모두 서로 다른 위상적 의미를 갖는다. 그러나 기존의 2차원 연산자 또는 단순히 차원을 확장한 연산자들로는 이러한 위상 관계의 차이를 표현 할 수 없다.

실세계에서 가능한 시나리오를 만들어 보면 그 표현의 한계는 더욱 명확하다. 예를 들어 3차원 항공 관제 시스템에서 태풍을 통과한 비행기들을 검색하는 경우 2차원 위상관계 연산자를 이용하면 실제로는 태풍을 통과하지 않고 태풍의 위로 지나간 비행기들도 모두 검색되게 되고, 기존의 4차원 시공간 연산자를 이용하면 태풍의 위로 지나간 비행기들은 모두 태풍의 영역을 지나가지 않은 것으로 분석된다.

이 논문에서는 이러한 문제점을 해결하기 위하여 기본적인 3차원 공간 위상 관계 연산자와 4차원 이동객체 위상 관계 연산자를 정의하고, 이러한 기본 연산자들을 이용하여 4차원 위상 변화 연산자를 제안한다. 4차원 위상 변화 연산자란 어떤 시구간 동안 3차원 공간에서 3차원 이동 객체들 사이의 연속적인 위상관계 변화를 분석하기 위한 새로운 연산자를 의미한다. 4차원 위상 변화 연산자는 기존의 연산자로는 적용할 수 없었던 실세계 응용에 적용할 수 있고, 이동 객체 데이터베이스에서 더욱 정확한 분석을 위한 질의를 수행 할 수 있다.

2. 관련 연구

이 장에서는 관련연구로써 기존의 위상관계 정의 기법과 공간 위상 관계, 그리고 시공간 위상 관계에 대해 기술하고 문제점을 제시한다.

* 이 연구는 한국과학재단 지원 RRC(청주대ICRC)의 지원으로 수행되었음.

2.1 위상관계 정의 기법

위상 관계를 정의하는 방식은 크게 포인트 집합에 기반 한 공간 교차 방법(DE-9IM)[4]과 계산에 기반 한 방법(Calculus based Method)[5]으로 분류된다. 이중에서 널리 인정된 방법은 공간 교차 방법이다. 공간 교차 방법은 포인트 집합(point set)을 기반으로 하여 객체의 내부, 경계, 외부를 나타내는 *I(a), B(a), E(a)*들 사이의 교집합 결과를 8개의 패턴 값으로 나타내어 공간 관계 조건을 공식화하는 것이다. Egenhofer[4]는 공간 교차 방법을 이용하여 disjoint, meet, overlap, equal, covers, coveredBy, inside, contains 등의 8가지 기본적인 2차원 공간 위상 관계를 정의 하였다. 이들 8가지 위상 관계는 3차원 또는 4차원 객체를 대상으로 확장 되어 적용 될 수 있다. 그러나 단순히 차원만 확장할 경우 1장에서 기술한 바와 같이 3차원 또는 4차원 객체들 사이에서 고려될 수 있는 특수한 위상 관계를 표현할 수 없다.

2.2 공간 및 시공간 위상관계

기존의 3차원 공간 위상관계[6,7] 및 시공간(2차원+ 시간) 위상관계[8,9]에 관한 연구에서는 2차원 위상 관계를 단순히 3차원으로 확장하거나 2차원 객체를 대상으로 연구되었기 때문에 3차원 객체들 사이에서 가능한 특수한 위상관계가 필요로 되는 4차원 (3차원+ 시간) 기반의 실제 응용에 그대로 적용할 수가 없다.

지금까지 몇몇 연구자들에 의해서 4차원 시공간 위상관계 연산자에 관한 연구가 수행되었고[2,3], 한국전자통신연구원에서는 'EDGE-4차원 지리정보 엔진 컴포넌트'를 개발하여 상용화 시킨 상태이다[10]. 그러나 이들 연구는 지형이나 시설물 같은 이력 객체에 대상으로 하고 있기 때문에 이동 객체에 적용하기에는 한계가 있다. 또한 이들 연구에서는 3차원 공간 위상관계 연산자에 시간 차원을 추가하여 4차원으로 확장하였는데, 이때 Allen의 7가지 시간 위상 개념을 그대로 적용하여 정의하고 있다. 예를 들면, α.Within4D(β)라는 연산은 α가 β에 대해 공간적으로 포함되고, α의 유효시간은 β의 유효시간에 대해 α.tDuring(β)의 관계가 성립하면 참을 반환하는 것으로 정의되어 있다[2]. 즉 이 연산이 참이 되려면 시간 위상이 tDuring이 성립되어야 한다. 그러나 실제로 tDuring의 관계 뿐만 아니라 tOverlap, tEqual 등과 같은 시간 위상 관계에서도 포함관계는 성립할 수 있다. 즉 어떤 시간 구간에서 두 객체가 어떤 공통된 시간 구간을 갖는다면 연산은 가능하게 된다. 따라서 이 논문에서는 공통유효시간(CommonValidtime) 개념을 적용한다.

3. 3차원 공간 위상 관계

이 논문에서 제안하는 3차원 공간 위상 관계 정의 기법은 3차원 객체의 경계(B), 내부(I), 외부(E)들 사이의 교집합 관계를 이용하는 DE-9IM을 적용할 때 x, y 좌표와

z좌표를 구분하여 사용하는 것이다. 즉, 3차원 객체의 x,y 좌표상의 경계(B_{xy}), 내부(I_{xy}), 외부(E_{xy})들 사이의 교집합 관계를 이용 하고 z_{top}, z_{bottom} 좌표 개념을 도입하여 8개의 기본적인 3차원 공간관계 연산자와 한 객체가 다른 객체의 위쪽에 위치한 경우의 7가지 기본적인 3차원 공간관계 연산자, 그리고 한 객체가 다른 객체의 아래쪽에 위치한 경우의 7가지 기본적인 3차원 공간관계 연산자들을 정의한다. 제안하는 기법으로 3차원 공간 위상관계 연산자를 정의하면 다음과 같다.

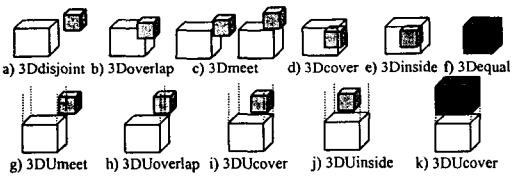
[정의 1] $a.3Doverlaps(b) \Leftrightarrow (I_{xy}(a) \cap I_{xy}(b) \neq \varnothing) \wedge (I_{xy}(a) \cap E_{xy}(b) \neq \varnothing) \wedge (E_{xy}(a) \cap I_{xy}(b) \neq \varnothing) \wedge (z_{top}(a) > z_{bottom}(b)) \wedge (z_{bottom}(a) < z_{top}(b))$

[정의 2] $a.3DUoverlaps(b) \Leftrightarrow (I_{xy}(a) \cap I_{xy}(b) \neq \varnothing) \wedge (I_{xy}(a) \cap E_{xy}(b) \neq \varnothing) \wedge (E_{xy}(a) \cap I_{xy}(b) \neq \varnothing) \wedge (z_{bottom}(a) > z_{top}(b))$

[정의 3] $a.3DLoverlaps(b) \Leftrightarrow (I_{xy}(a) \cap I_{xy}(b) \neq \varnothing) \wedge (I_{xy}(a) \cap E_{xy}(b) \neq \varnothing) \wedge (E_{xy}(a) \cap I_{xy}(b) \neq \varnothing) \wedge (z_{top}(a) < z_{bottom}(b))$

[정의 1]의 3Doverlaps 연산은 두 3차원 객체들이 서로 x, y, z 좌표상에서 모두 겹쳐지는 경우를 의미하고, 정의 2의 3DUoverlaps 연산은 한 객체가 다른 객체의 위에 위치하면서 x, y 좌표상으로는 겹쳐지고 z좌표 상으로는 떨어져 있는 경우, 그리고 정의 3의 3DLoverlaps 연산은 한 객체가 다른 객체의 아래에 위치하면서 x, y 좌표상으로는 겹쳐지고 z좌표 상으로는 떨어져 있는 경우를 의미한다.

[그림 1]은 제안된 22개의 3차원 공간 위상 관계 연산을 가시적으로 나타낸 것이다. 이 그림에서는 3Dinside/3Dcontains와 같이 연산 대상의 순서만 다르고 위상 관계는 유사한 것은 둘 중에서 하나만 나타내었다. 또한 3DLxxx 계열 연산자들은 3Duxxx 계열 연산자들과 위/아래 순서만 바꾸면 관계가 동일하므로 그림으로 표기하지 않았다. [그림 1]에서 a)부터 f)까지 3Dxxx 계열 연산자들은 기본적인 3차원 공간 위상 관계를 보여주고 있고, g)부터 k)까지 3Duxxx계열 연산 자들은 한 객체가 다른 객체의 위쪽에 위치한 경우를 보여주고 있다.



[그림 1] 3차원 부피 객체들 사이의 공간 위상 관계

이와 같이 정의한 22개의 기본적인 3차원 공간 위상 관계 중에서 3Dmeet와 3Ddisjoint 연산은 x,y좌표와 z좌표의 기본적인 분리 기법과 다른 적용이 필요하다. 3Dmeet는 [그림 1]의 c)와 같이 객체와 객체가 직접 만나는 상태를 의미하는 것으로 x,y 좌표와 z좌표를 분리하지 않고 정의하고, 3Ddisjoint는 z좌표와 관계없이 x,y좌표를 기준으로 서로 떨어져 있는 경우를 의미하기 때문에 각각 [정의 4], [정의 5]와 같이 정의한다.

[정의 4] $a.3Dmeet(b) \Leftrightarrow (I_{xyz}(a) \cap I_{xyz}(b) = \varnothing) \wedge ((I_{xyz}(a) \cap B_{xyz}(b) \neq \varnothing) \vee (B_{xyz}(a) \cap I_{xyz}(b) \neq \varnothing) \vee (B_{xyz}(a) \cap B_{xyz}(b) \neq \varnothing))$

[정의 5] $a.3Ddisjoint(b) \Leftrightarrow (I_{xy}(a) \cap I_{xy}(b) = \varnothing) \wedge (I_{xy}(a) \cap B_{xy}(b) = \varnothing) \wedge (B_{xy}(a) \cap I_{xy}(b) = \varnothing) \wedge (B_{xy}(a) \cap B_{xy}(b) = \varnothing)$

4. 4차원 위상변화 연산

4.1 4차원 이동객체 위상 관계 연산자

이 논문에서는 제안된 3차원 공간 위상 관계 연산자를 4차원 시공간 위상관계 연산자로 확장 하기 위해 공통 유효 시간 개념을 적용한다. 일반적으로 특정 객체에 대하여 시공간 연산을 수행한다는 의미는 시간적으로 기준 객체(a)와 탐색 객체(b)가 동시에 존재하는 공통 유효 시간을 만족해야 하며 공통 유효 시간 동안에 공간 조건을 만족하는 탐색 객체의 선택을 의미하기 때문이다. 공통유효시간은 다음과 같이 정의된다.

[정의 6] CommonValidtime(a, b) = BEGIN(Validtime(a)) ≤ END(Validtime(b)) and END(Validtime(a))≥BEGIN(Validtime(b)))

공통 유효 시간을 적용하여 4차원 시공간 위상관계 연산자를 정의하면 다음과 같다.

[정의 7] 4DOverlap 연산은 두 3차원 이동 객체가 공통으로 존재하는 유효시간 구간동안 두 부피 객체가 서로 교차 하는 것을 의미하고 다음 식으로 정의한다.

$a.4DOverlap(b) \Leftrightarrow (I_{xy}(a) \cap I_{xy}(b) \neq \varnothing) \wedge (I_{xy}(a) \cap E_{xy}(b) \neq \varnothing) \wedge (E_{xy}(a) \cap I_{xy}(b) \neq \varnothing) \wedge (z_{top}(a) > z_{bottom}(b)) \wedge (z_{bottom}(a) < z_{top}(b)) \wedge$ CommonValidtime

4차원 이동객체 위상 관계 연산자는 [정의 7]과 같이 22개의 3차원 공간 위상 관계 연산자에 공통 유효시간을 적용하여 정의하므로 모두 22개의 4차원 이동객체 위상 관계 연산자가 정의 된다. 이렇게 정의한 4차원 이동객체 위상 관계 연산자는 다음 장에서 제시하는 4차원 위상 변화 연산자를 정의하는데 사용된다.

다음은 제안한 주요 연산자들의 구체적인 알고리즘을 보여주고 있다.

```
Algorithm 4DOverlap(4DsourceObject, 4DtargetObject, TimeInterval)
Input : Box타입의 4DsourceObject, 4DtargetObject, TimeInterval
Output : Boolean(true or false)
Boolean 4DOverlap(4DsourceObject, 4DtargetObject, TimeInterval)
{   get Oid of 4DsourceObject and 4DtargetObject)
    If(주어진 TimeInterval내에 두 개 이상의 객체가 존재){
      for(두 객체가 모두 존재하는 공통 유효시간 동안){
        If((z_top(4DsourceObject)>z_bottom(4DtargetObject))∧
          (z_bottom(4DsourceObject) < z_top(4DtargetObject))){
            두 객체의 x,y 좌표를 검색해서 배열에 저장;
            If(x,y 좌표상에서 저장된 두 객체의 Overlap이 성립){
                return true;   }
            Else { return false; } }
        Else{ return false;  } } }
    Else { return false;   }
}
```

[알고리즘 1] 4DOverlap 연산

```
Algorithm 4DUOverlap(4DsourceObject, 4DtargetObject, TimeInterval)
Input : Box타입의 4DsourceObject, 4DtargetObject, TimeInterval
Output : Boolean(true or false)
Boolean 4DUOverlap(4DsourceObject, 4DtargetObject, TimeInterval)
{   get Oid of 4DsourceObject and 4DtargetObject)
    If(주어진 TimeInterval내에 두 개 이상의 객체가 존재){
      for(두 객체가 모두 존재하는 공통 유효시간 동안){
        If(z_bottom(4DsourceObject) > z_top(4DtargetObject)){
            두 객체의 x,y 좌표를 검색해서 배열에 저장;
            If(x,y 좌표상에서 저장된 두 객체의 Overlap이 성립){
                return true;   }
            Else { return false; } }
        Else{ return false;  } } }
    Else { return false;   }
}
```

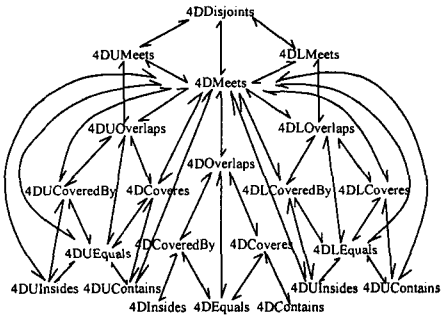
[알고리즘 2] 4DUOverlap 연산

[알고리즘 1]과 [알고리즘 2]에서의 가장 큰 차이점은 z좌표의 처리에 있다. 시간 처리에 있어서는 모두 같은 방식으로 설계하였다. 즉, 주어진 시간 구간 내에 객체가 존재하는지 먼저 검사하고, 존재하지 않으면 false를 반환하게 된다. 주어진 시간 구간 내에 객체가 존재하는 경우 다시 한번 객체들 사이에 공통 유효시간이 존재하는지 여부를 검색한 후 존재할 경우 공간 연산 과정이 수행된다.

4.2 4차원 위상 변화 연산자

이 장에서는 이 논문의 최종 목표인 4차원 위상 변화 연산자를 정의한다. 4차원 위상 변화 연산자는 3장에서 제안한 4차원 이동객체 연산자들을 기호 "▷"을 사용하여 조합함으로써 간단하게 나타낼 수 있다. 그러나 가능한 위상 변화 관계의 경우의 수가 너무 많기 때문에 모든 가능한 위상 변화 관계 마다 이름을 명명하거나 정의를 제시하는 것은 불가능하다. 따라서 이 논문에서는 실제 응용에서 빈번하게 사용

가능한 몇몇 연산자에 대해서만 기술하고, 가능한 모든 위상 변화 관계 변화를 쉽게 보여주기 위하여 [그림 2]와 같이 이동 부피 객체들 사이의 인접 위상 변화 그래프를 제시하였다. 사용자는 이 그래프를 보고 필요에 따라 원하는 위상 변화 관계 연산자를 정의하고 이름을 명명하여 사용할 수 있다.



[그림 2] 이동 부피 객체들 사이의 인접 위상 변화 그래프

다음은 실제 응용에서 빈번하게 사용 가능한 주요 4차원 위상 변화 연산자의 정의를 기술한 것이다.

[정의 8] 4DEnters=: 4DDisjoints ▷ 4Dmeets ▷ 4Doverlaps ▷ 4DcoveredBy ▷ 4DInsides

[정의 9] 4DUEnters =: 4DDisjoints ▷ 4DUMeets ▷ 4DUOverlaps ▷ 4DUCoveredBy ▷ 4DUInsides

[정의 10] 4DUCrosses=:4DDisjoints ▷4DUMeets ▷4DUOverlaps ▷4DUInsides ▷ 4DUOverlaps ▷ 4DUMeets ▷ 4DDisjoints

[정의 11] 4DIntersects =: 4DEnters or 4DCrosses or 4DLeaves

4DEnters 연산은 시간 구간 동안 점 객체 또는 부피 객체가 다른 부피 객체의 내부로 들어가는 동안의 위상 변화 과정을 의미하는 것이고, 4DUEnters 연산은 점 객체 또는 부피 객체가 다른 부피 객체의 위에 위치하면서 x,y 좌표상으로는 내부로 들어가는 동안의 위상 변화 과정을 의미하며, 4DUCrosses 연산은 한 객체가 다른 부피 객체의 위로 지나가는 경우의 위상 변화 과정을 의미하는 것이다. 4DIntersects 연산은 하나의 위상 변화 연산자로는 나타낼 수 없는 경우로 4차원 위상 변화 연산자 4DEnters, 4DCrosses, 4DLeaves 중에서 어느 하나 이상을 만족하면 성립한다.

5. 4차원 위상 변화 연산 질의 시나리오

이 장에서는 제안된 4차원 위상 변화 연산자가 어떻게 적용될 수 있는지 항공 관제 시스템에 적용한 시나리오를 통하여 살펴본다. 항공 관제에 있어서 비행기와 승객의 안전에 있어서 날씨에 관한 정보는 매우 중요한 역할을 한다. 따라서 비행기의 이동 경로와 태풍, 안개, 눈보라, 구름 등의 날씨 정보가 데이터베이스에 저장된다고 가정하고, 비행기와 날씨 객체 사이의 위상 변화 연산을 통한 질의를 살펴본다. 질의를 위하여 다음과 같은 릴레이션을 사용한다.

flights(id:string, alocation:4Dmpoint), typhoons(id:string, wlocation:4Dmbox)

여기서, id는 비행기와 태풍의 식별자를 의미한다. 또한 alocation은 공간 좌표 x,y,z와 시간 t를 갖는 4차원 이동 점으로 표현되는 비행기의 이동 경로를 의미하고, wlocation은 4차원 이동 부피로 표현되는 날씨 객체의 이동 경로를 의미한다.

제안된 위상 변화 연산자들을 이용하여 다음과 같이 태풍과 비행기 사이의 위상 관계에 관한 질의들이 수행될 수 있다.

[질의 1] "8월 31일 오전 9시부터 오후 12시 사이에 태풍 루사(id:1015)를 통과하여 지나간 모든 비행기를 검색하시오."

```
SELECT f.id          FROM flights f, typhoons t
WHERE t.id = 1015 and 4DCrosses(f, t, 8.31.9:00-24:00)
```

[질의 2] "8월 31일 오전 9시부터 오후 12시 사이에 태풍 루사(id:1015)의 위를 지나간 모든 비행기를 검색하시오."

```
SELECT f.id          FROM flights f, typhoons t
WHERE t.id = 1015 and 4DUCrosses(f, t, 8.31.9:00-24:00)
```

[질의 1]은 주어진 시간 구간 동안에 태풍의 직접적인 영향을 받으며 태풍 속을 통과한 비행기들을 검색하는 질의로서 조종사가 위험한 비행을 했음을 알 수 있다. [질의 2]는 비행기가 태풍의 영역을 통과하긴 했지만 태풍의 영향을 받지 않고 태풍 위로 지나간 비행기들을 검색하는 질의로서 안전한 운행을 했음을 알 수 있다.

6. 결론 및 향후연구

기존의 2차원 이동객체 연산자나 4차원 시공간 연산자로는 시간에 따라 이동하는 3차원 이동객체들 사이에 나타날 수 있는 복잡하고 특수한 위상관계들을 분석할 수 없었고 따라서 실제 응용에 적용하는데 한계가 있었다.

이 논문에서는 이러한 문제점을 해결하기 위하여 4차원 위상 변화 연산자를 정의하였고, 4차원 이동객체들 사이에서 가능한 모든 위상 변화 관계를 가시화하기 위하여 인접 위상 변화 그래프를 제안하였다. 4차원 위상 변화 연산자 정의를 위하여 기본적인 4차원 이동객체 위상 관계 연산자를 정의하고 주요 알고리즘을 기술하였다. 또한 4차원 이동객체 위상 관계에 기반이 되는 22가지 기본적인 3차원 공간 위상 관계를 정의하였고, 이러한 3차원 공간 관계를 정의하는 방법으로 x,y좌표와 z좌표를 분리하여 공간 교차 방법을 사용하는 기법을 제안하였다.

위와 같은 연구를 수행한 결과 시간 구간 동안 3차원 이동객체들 사이의 위상변화에 관한 표현력을 높일 수 있었고, 제안한 인접 위상 변화 그래프를 통하여 가능한 모든 위상 변화 관계를 표현할 수 있음을 보였다. 또한 좌표를 분리한 공간 교차 방법을 사용하여 3차원 객체들 사이의 특수한 공간 위상 관계를 정의함으로써 기존의 연산자로는 분석할 수 없는 위상 관계를 분석 할 수 있었다. 아울러 제안한 4차원 위상 변화 연산자를 항공 관제 시나리오에 적용하여 복잡하고 다양한 질의가 간단하게 수행 될 수 있음을 보였다.

향후에는 지금까지 제안한 정의와 알고리즘에 따라 연산자들을 구현하고 구현된 연산자들을 실제 응용 시스템에 적용하는 연구를 수행할 것이다.

참고문헌

[1] R.H.Guting, M.H.Bohlen, M.Erwig, C.S.Jensen, and N.A.Lorentzos, M.Schneider, and M.Vazirgiannis, "A Foundation for Representing and Querying Moving Objects," ACM Transactions on Database Systems, Vol.25, No.1, 2000.
[2] 강구, "컴포넌트 기반의 4차원 시공간 위상관계 연산자의 설계 및 구현", 충북대학교 대학원 전자계산학과 이학석사 학위논문, 2002년 2월.
[3] 이성호, 김상수, 김경호, 박종현, "4차원 지리정보시스템에서의 데이터 제공 서비스", 한국정보과학회 봄 학술발표논문집(A), Vol. 30, No. 1, pp. 632-634, 2003.
[4] M.J. Egenhofer and R.D. Franzosa. "Point-set Topological Spatial Relations," International Journal of Geographical Information Systems, Vol. 5, pp. 161-174, 1999.
[5] E. Clementini and P. Di Felice, A Comparison of Methods for Representing Topological Relationships, Information Sciences, pp. 149-178, 1995.
[6] S. Zlatanova, "On 3D Topological Relationships," 11th International Workshop on Database and Expert Systems Applications(DEXA'00), pp. 913, Sept 2000.
[7] 김상호, 강구, 류근호, "3차원 공간 위상 관계 연산자의 설계", 한국정보처리학회 논문지D, Vol.10, No. 02, pp. 211-220, 2003년 4월.
[8] Yoshifumi Masunaga, Noriko Ukai, "Toward a 3D Moving Object Data Model – A Preliminary Consideration –," In Proceedings of the 1999 International Symposium on Database Applications in Non-Traditional Environments (DANTE'99), 1999.
[9] M.Erwig and M.Schneider. "Spatio-Temporal Predicates," Technical Report, FernUniversity Hagen, 1999.
[10] 강구 , 이현아, 김상호, 류근호, 이성호, "컴포넌트 기반의 2차원 시공간 위상관계 연산자의 설계", 한국정보과학회 추계학술대회, Vol. 28, No. 02, pp. 79-81, 2001.