지리 데이타베이스 시스템에서의 효율적인 공간 데이타 수집

김 종 훈† 김 재 홍†† 배 해 영†††

요 약

지리 데이타베이스 시스템은 지도 형태의 데이타를 출력할 뿐만 아니라 공간 데이타와 비공간 데이타에 대해서 저장, 검색, 조작 및 분석 등을 처리하는 데이타베이스 시스템이다. 이러한 지리 데이타베이스 시스템에서 관리되는 공간 데이타는 그 크기가 매우 방대하므로 공간 데이타의 입력시 많은 시간이 요구되며 저장 공간을 많이 차지하게 된다. 따라서 지리 데이타베이스 시스템은 공간 데이타의 입력 시간을 줄이고, 저장 공간을 효과적으로 사용할 수 있는 공간 데이타 수집 시스템이 필수적으로 요구된다. 본 논문에서는 공간 데이타에 대한 입력 시간을 줄이기 위해 전체를 한번에 벡터화하는 기존의 방법에 대한 문제점을 분석하여 이를 개선한 벡터화 기법을 제안한다. 제안된 벡터화 방법은 도면 전체를 벡터화하지 않고, 공간 데이타의 특성을 고려하여 사용자가 속성 정보를 입력하는 시점에서 그 공간 데이타만을 벡터화한다. 이와함께 수집된 데이타의 저장 공간을 효율적으로 사용하기 위해 태그 비트를 사용한 압축 저장 형식을 제안한다.

Efficient Capturing of Spatial Data in Geographic Database System

Jong Hoon Kim†, Jae Hong Kim†† and Hae Young Bae†††

ABSTRACT

A Geographic Database System is a database system which supports map-formed output and allows users to store, retrieve, manage and analyze spatial and aspatial data. Because of large data amount, it takes too much time to input spatial data into a Geographic Database System and too much storage. Therefore, an efficient spatial data collecting system is highly required for a Geographic Database System to reduce the input processing time and to use the storage efficiently. In this paper, we analyze conventional vectorizing methods and suggest a different approach. Our approach vectorizes specific geographic data when the users input its aspatial data, instead of vectorizing all the map data. And also, we propose optimized vector data format using tag bit to use the storage that collected data efficiently.

1. 서 론

지리 데이타베이스 시스템은 지도 형태의 데이타를 출력할 뿐만 아니라 지리 데이타의 저장, 검색, 조작 및 분석 등을 처리하는 데이타베이스 시스템으로 지리적 요소의 공간 데이타와 이와 관련된 속성 데이타를 처리하는 시스템이다[3, 4, 11, 12]. 지리 데이타베이스 시스템에서 공간 데이타는 지도, 지리, 지역, 도로, 하천, 학교, 공공시설 등의 지리와 관계되는 점(point), 망(network), 영역(region)적인 성격을 가지는 데이타들로써 마우스(mouse), 수동 독취기(digitizer), 자동 독취기(scanner) 등을 통하여 입력한다[1, 6, 7, 9].

이러한 다양한 입력 장치의 선택은 지리 데이타베이스 시스템의 자료 구조에 크게 의존하는데, 공간 데이타의 형태별, 기능별로 분류되는

† 준 회 원 : 인하대학교 전자계산학과 박사과정
†† 정 회 원 : 인하대학교 전자계산공학과 전임대우
††† 종신회원 : 인하대학교 전자계산공학과 교수
논문접수 : 1994년 6월 20일,심사완료:1994년 9월 24일

업무의 특성을 고려하여 시스템 운용에 적합한 입력 시스템이 구성되어야 한다. 예를 들어, 도로 유지 관리를 위해서는 도로 현황 평면도, 지하 시설물도, 도로 용지 구성도, 종횡 단면도가 도로 유지 관리에 적합하게 활용될 수 있도록 입력 방법을 심도있게 검토하여야 한다[11]. 특히 지리 데이타베이스 시스템에서 취급하는 공간 데이타는 그 크기가 매우 방대하므로 입력시 많은 시간이 요구되고, 저장시 많은 공간을 차지하므로 공간 데이타의 입력 시간을 줄이고, 저장 공간을 효율적으로 사용할 수 있는 방법에 대한 연구가 필요하다.

본 논문에서는 지리 데이타베이스 시스템에서 공간 데이타를 입력하는 방법인 수동 독취기를 통한 수동 입력과 자동 독취기를 통하여 벡터화(vectorizing)한 자동 입력에 대하여 장단점을 분석하고, 공간 데이타의 입력 시간을 줄일 수 있는 공간 데이타 속성을 고려한 벡터화 방법을 제안한다. 그리고 제안된 벡터화 방법에 의해 표현되는 방대한 공간 데이타의 저장 효율을 향상시킬 수 있는 태그 비트(tag bit)를 이용한 압축 저장 형식을 제안한다.

2. 공간 데이타의 수집

지리 데이타베이스 시스템은 도시의 도형과 그 도형상의 속성치 그리고, 그 도형상의 주요 대상물인 도로 점용물, 즉 상하수도 시설물, 가스관, 전기 및 통신시설물 등을 수집, 저장, 처리, 발췌 그리고 출력하는 기능을 갖는다. 따라서 사용자가 많은 데이타를 용이하게 대화식으로 입력할 수 있도록 다양한 입력장치를 지원하여야 한다[13, 14]. 이러한 입력 장치를 사용하여 데이타를 수집하는 데이타 수집 시스템은 다음과 같이 속성 데이타 입력 시스템과 공간 데이타 입력 시스템으로 나뉜다.

1) 속성 데이타 입력 시스템

키보드를 이용하여 도로명, 도로길이, 도로 폭, 인구밀도, 교통량, 강우량 등과 같은 속성 정보를 입력할 수 있다.

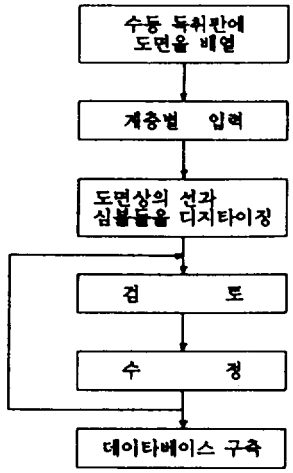
2) 공간 데이타 입력 시스템

공간 데이타의 입력 장치는 수작업으로 도형 좌표를 취득하는 수동 독취기, 레이져 빔을 사용하여 선의 방향을 감지하여 추적하는 선형 추적 독취기 그리고 래스터 데이타(raster data)를 벡터화하는 자동 독취기가 있다[1, 11].

2.1 수동 독취기

수동 독취기란 도면상의 데이타를 전산화하기 위하여 수동 독취판(A0 size)위에 도면을 놓고 수작업으로 좌표화가 필요한 도형을 따라 커서(cursor)를 이동하여 도면상의 선과 기호들을 디지탈화하는 것이다.

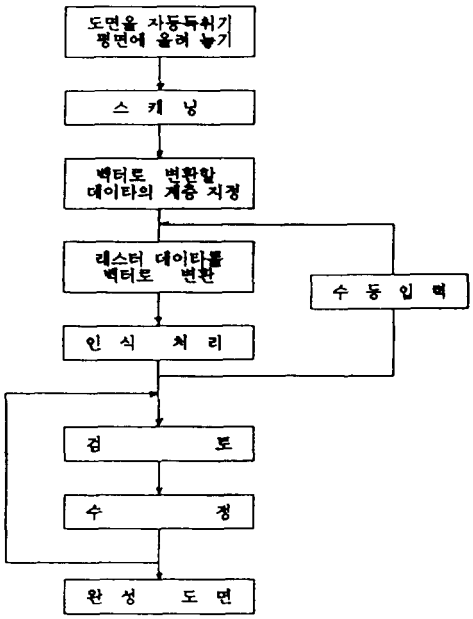
수동 독취기는 입력할 데이타를 입력하기 전에 계층(layer)별로 분류를 하고, 분류 정리된 각 계층은 수동 독취기에 의해 개별적으로 입력된다. 이렇게 입력된 데이타는 즉시 벡터값으로 저장된다. 수동 독취기의 특징은 필요한 데이타를 선별하여 입력하므로 메모리의 낭비를 막을 수 있으며, 다른 좌표 독취기보다 가격이 저렴하고, 작업을 하면서 작업 순서를 임의로 할 수 있다는 장점이 있다. 그러나 데이타 입력을 하는데 장시간이 소요되며, 데이타의 정확도가 입력자의 숙련도에 비례하는 단점이 있다.



(그림 1) 수동 독취기에 의한 공간 데이타의 입력
(Fig. 1) Spatial Data Input Using a Digitizer

2.2 자동 독취기

일반적으로 공간 데이타는 그 양이 방대하고 복잡하므로 수동 독취기에 의한 입력에는 많은 노력을 필요로 한다. 그러나 오늘날 광학 자동 독취기의 개발로 도면 전체를 자동 독취기에 의해 규칙적으로 투과(scan)하여 광밀도(optical density)의 래스터 데이타로 저장한 후 소프트웨어에 의해 벡터화하는 것이 가능해졌다. 이것은 공간 데이타를 자동으로 인식하여 디지탈화하여 주므로 공간 데이타를 입력하는 시간의 절감과 노력을 최소화할 수 있다는 장점이 있다. 그러나 자동 독취기는 데이타 입력을 컴퓨터가 처리하고자 하는 데이타만 입력하는 것이 아니라, 입력하고자 하는 도면의 공간을 포함한 전체를 읽어 들이므로 입력된 데이타에서 불필요한 데이타를 제거해야 하는 문제점이 있다[11].



(그림 2) 자동 독취기에 의한 공간 데이타의 입력
(Fig. 2) Spatial Data Input Using a Scanner

또한 입력된 데이타는 영상 개념의 래스터 값을 갖고 있기 때문에 벡터 형식의 데이타 변환(vector conversion) 작업을 하여야 하며, 시스템 운영에 가장 중요한 체계적인 분류를 자동으로 하지 못한다는 단점이 있다. 따라서 도면 보관 관리에는 적합하지만 체계적인 분류를 하기 위해서 분류체계를 다시 설정하여야 하며, 재차 데이타 변환을 하여야 하므로 많은 시간과 기억용량을 필요로 하게 된다. 또한 전통적인 지도는 인간이 읽기 편하도록 만들었기 때문에 자동 독취기가 자동으로 인식하기에는 많은 제약이 따른다. 그밖에도 선들은 문자열이나 다른 선에 의해 끊어짐이 빈번하게 발생하며, 또 영역의 경우 한 지도에서 영역으로 표시되는 도시가 다른 곳에서는 점으로 표시되는 경우가 있으며, 심볼이 통일되지 않아 자동 도면 독취(automatic map digitizing)시스템에서 어려움으로 남고 있다.

3. 다각형의 속성을 고려한 벡터화 기법

지리 데이타베이스 시스템에서 다루는 공간 데이타는 그 크기가 방대하므로 디지탈 정보를 표현하고 데이타를 조작하는 복잡한 처리를 사람이 아닌 컴퓨터가 대행하기를 원하게 되었다. 최근 자동 독취기와 리모트 센싱과 같은 입력 기술이 발달함에 따라 래스터 데이타를 벡터 데이타로 변환하여 다양한 공간 데이타를 처리할 수 있는 벡터화 기법에 관한 연구가 활발하게 진행되었다. 즉, 디지탈 형태의 공간 데이타를 읽어 래스터 구조로 저장한 후 벡터 데이타로 변환하는 벡터화 기법에 의해 자동 인식함으로써 디지타이징을 대신해 줄 수 있다[2]. 이러한 벡터화에 대한 연구는 기계 도면 인식, 문자 인식 등의 필요성에 힘입어 활발하게 진행되고 있으나, 아직은 사용자의 개입을 필요로 하지 않는 완전한 의미의 자동 입력 시스템은 구현되어 있지 않다.

본 논문에서는 전체를 벡터화하지 않고, 사용자가 속성 정보를 입력하는 시점에서 그 공간 데이타만을 벡터화하는 방법을 제안한다. 이로써 벡터화를 원하는 공간 데이타의 속성을 고려하여 정확하게 벡터화하여 입력 시간과 저장 공간을 줄일 수 있으며, 그 공간 데이타 특성에 맞도록 분류하여 저장할 수 있게 한다. 이를 위해 지적

도에서 가장 많이 나오는 다각형으로 구성된 공간 데이타를 벡터화하는 것만을 다루며, 공간 데이타들은 서로 교차 또는 포함 관계가 아니라고 가정한다.

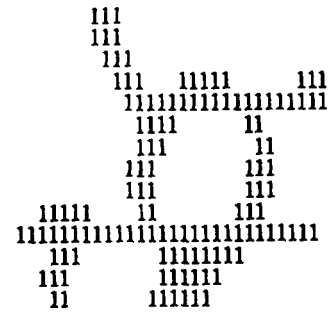
3.1 벡터화를 위한 기본 연산

일반적으로 래스터 데이타를 벡터 데이타로 변환하는 작업은 세개의 기본 연산으로 나뉘어 진다. 즉, 주어진 해상도내에서 선을 단일 두께로 줄이는 골격화 과정과 골격화된 선을 따라가면서 선의 좌표값을 식별하는 선 추출 과정, 그리고 선들의 인접성 관계를 고려하여 필요한 선분으로써 데이타를 구축하는 위상 재구성 과정으로 나뉜다[8].

(1) 골격화(line thinning)

현재 임의 형태의 직선들을 골격화하는 접근 과정은 반복적으로 선의 양쪽면을 줄여가는 peeling 방법과 하나의 상자내에서 여러 개의 풍선들이 부풀어 오르는 것처럼 선들의 공간을 확장하는 ballooning 방법, 그리고 각 선의 중심을 계산하는 medial axis 방법 등이 있다.

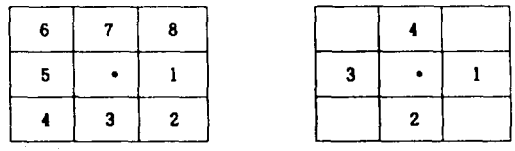
Peeling 방법은 골격화를 위한 가장 일반적인 알고리즘으로 각각의 두께를 갖는 선을 단일 두께가 될때까지 선의 양면을 한 해상도씩 줄여가는 방법이다. (그림 3)은 색이 있는 곳을 1로 표현하고, 색이 없는 곳을 0으로 표현하는 이진 행렬로 구성된 래스터 데이타의 한 예이다.



(그림 3) 자동 독취기로 읽은 이진 래스터 데이타
(Fig. 3) Binary Raster Data Scanned by a Scanner

이 과정은 0이 아닌 점에 대해 가운데 점과 (그림 4(a))와 같이 그와 인접한 8개의 점이 조사된다. 가운데 점은 반드시 다음 3가지 기준을 만족해야 하며 그렇지 않으면 0으로 바꾸어 선에서 제거한다. 즉, 가운데 점은 행렬내에서 유일하게 점유된 점이 아니며, 이웃점들이 연결되지 않도록 하며, 그 점의 제거는 선의 교차점이 아니므로 직선의 연속성을 변경시키지 않는다.

이 알고리즘은 선 단위로 처리하는 것이 아니라 점 단위로 하기 때문에 투사선 접근법을 이용하여 병렬 처리의 구현이 가능하다. 또한 선 교차 부근에 대한 정확하고 매끄러운 처리가 가능하고 구현이 쉽다는 장점이 있다. 그러나 점 단위로 처리하기 때문에 속도가 매우 느리고 직선의 두께가 매우 두꺼운 경우 중심점을 결정하는데 어려움이 있다.



(a) 8개의 이웃점 (b) 4개의 이웃점

(그림 4) 중심 점에 이웃하고 있는 점
(Fig. 4) Points Adjacent to a Center Point

Ballooning 방법은 선 사이의 영역을 경계선이 더 이상 감소될 수 없을 때까지 확장함으로써 선을 골격화한다. 이때 영역의 확장에 대한 기준은 peeling 방법의 3가지 기준과 동일하나 대각선상의 이웃점은 고려하지 않는다. 즉, (그림 4(b))와 같이 4개의 이웃점만을 비교함으로써 peeling 방법보다 빠른 골격화를 가능하게 한다. Balloning 방법은 선 자체보다는 선 사이의 공간을 처리하는 알고리즘으로 선의 한 방향으로만 접근함으로써 부정확한 결과를 초래할 수도 있다.

·Medial Axis 방법은 래스터 데이타의 모든 점에 대해 가장 자리로부터 최대 거리를 기반으로 한다. 즉, 래스터 데이타내의 각 점은 그것과 가장 가까운 바탕의 점으로부터의 점 갯수와 같은

값을 갖는다. 색을 갖는 모든 점은 다음과 같은 두 단계에 의해 거리 값을 할당받는다. 첫번째 단계는 모든 점들을 왼쪽에서 오른쪽, 그리고 위에서 아래 방향으로 전진하면서 그것의 바로 위 또는 왼쪽의 값중 최소 값에 1을 더하여 할당한다. 두번째 단계는 결과 행렬을 역순 즉, 오른쪽에서 왼쪽 그리고 아래에서 위로 올라가면서 처리되는데 각 점에 현재 할당된 값과 오른쪽의 값에 1 더한 값, 그리고 이웃 아래값에 1 더한 값 중 최소의 값으로 대치한다. (그림 5(a))와 (그림 5(b))는 두 단계를 보여준다.

```
111       111
122       122
 1211    1233
  122   12344
   121123
    12234
   1233
   12341
  1234521
 123   121
 123   122
 123   123
 123   123
123    123
!23    123
```

(a) 1단계 처리

```
111       111
121       121
 1211    1221
  121   12111
   121121
    12211
   1221
   12221
  1211121
 121   121
 121   121
 121   121
 121   121
121    121
121    121
```

(b) 2단계 처리

(그림 5) Medial Axis의 두 단계 과정
(Fig. 5) Two Passes of Medial Axis

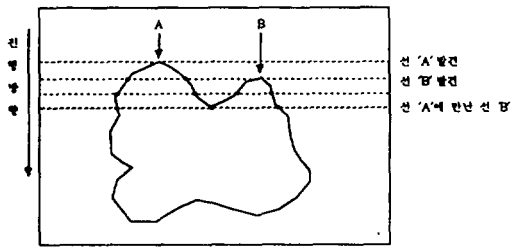
골격화는 두 단계에 의해 나온 결과 행렬에서 최대값을 갖는 것만을 취함으로써 이루어진다. 이 알고리즘은 매우 얇은 선에 대해 오류를 산출할 수 있다는 것과 병렬 처리가 어렵다는 단점이 있다.

(2) 선 추출(Line Extraction)

선 추출에 대한 접근 방법은 선 추적 방법과 모든 선을 동시에 추적하는 투사선(scan line) 방법이 있다.

선 추적 방법은 선을 따라가면서 선이 어떤 방향으로 진행했는지를 조사하여 순차적으로 재구성하는 방법이다. 한번 추적된 선은 더 이상 추적되지 않으며 위상을 고려하여 구성하는 개념적으로 매우 단순한 방법이다. 그러나 선이 교차한 곳에서 분기할 곳을 어떻게 결정해야 하는지의 문제가 발생하게 되고 후에 남은 분기 노선을 추적해야 한다. 또한 실행 시간이 선의 총 길이에 비례하기 때문에 래스터 데이타가 복잡할수록 실행 시간이 오래 걸리는 단점이 있다.

투사선 방법은 투사선과 교차하는 모든 선을 동시에 처리하는 방법으로 각 투사선은 한번만 읽혀진다. 즉, (그림 6)과 같이 서로 연결되어야 하는 선을 찾아야 하며 선이 교차하는 지점이 발견되면 위상 재구성을 위해 기록될 수 있다.



(그림 6) 투사선에 의한 선 추출
(Fig. 6) Line Extraction via the Scan-Line

(3) 위상 재구성

위상 재구성은 보통 골격화 과정이나 선 추출 과정에서 부분적으로 수행될 수 있다. 특히 선 추출 과정에서 공간 상황으로서 인식되고 다루어져야 하므로 교차하는 지점에 대한 위치와 형태를 기억하고 있다가 후에 위상 재구성을 위해 쓰여져야 한다. 즉, 교차는 "T"형, "X"형, "+"형 또는 "Y"형 중에 하나로 인식되어야 한다.

기존의 벡터화 알고리즘은 지리 데이타베이스 시스템에서 사용하는 지도와 같은 복잡한 도면을 벡터화하는 경우 스캐닝한 도면 전체를 대상으로 위와 같은 방법으로 벡터화한 후 각 공간 데이타 단위로 속성 정보를 입력한다. 따라서 불필요한 데이타도 벡터화를 하게 되고, 글자나 다른 선에 의해 공간 데이타가 끊어지거나, 스캐닝시의 오류로 인한 선들을 자동으로 인식하지 못하는 경우가 발생한다. 또한 각 객체가 갖고 있는 속성을 고려하지 않은채 벡터화하여 추후 공간 객체 단위로 데이타를 수정해야 한다는 문제점이 있다.

3.2 제안 시스템의 알고리즘

N다각형은 평면에서 서로 다른 N개의 점(m1, m2, m3, ···, mn)이 주어지고 각 점들을 선분([m1, m2], [m2, m3], ···, [mn−1, mn], [mn, m1])으로 이어 만든 도형이다. 단, N은 3 이상이다. 즉, 다각형은 시작점과 끝점이 일치하며 다각형 내에 존재하는 내부 각이 N개 존재한다는 속성을 갖고 있다. 이와 같은 다각형의 속성을 고려하여 벡터화한다면, 중간에 선이 끊어져 점이 시작점으로 돌아오지 않을 경우를 고려하여 점 추적시 진행 방향으로 이웃한 점이 없더라도 그 다음 점이 있는지를 조사하여 끊어진 선에 대해서도 추적을 계속할 수 있으며, 점 추적시 이웃하는 점이 여러개 존재하는 경우 진행 방향으로부터 반 시계 방향으로 검사하여 먼저 나타나는 점으로 진행하면 되므로 위상 재구성 과정을 고려하지 않아도 된다는 장점을 갖는다.

본 논문에서 제안한 벡터화는 골격화를 위한 알고리즘으로 비교적 정확하게 골격화를 하는 peeling 방법을 사용하며, 선 추출 알고리즘으로는 공간 데이타의 속성을 고려하여 벡터화를 하기 때문에 공간 데이타별로 벡터화를 할 수 있는 선 추적 방법을 사용한다. 제안된 벡터화는 공간 데이타별로 벡터화를 함으로 선 추적 후 위상 재구성을 고려하지 않으며, 또한 기존의 방법으로 벡터화된 데이타가 실제의 공간 데이타 뿐만 아니라 필요하지 않은 많은 중간 데이타를 갖게 됨으로 이를 제거하기 위해 임계 기울기와 임계 거리를 계산하여 최적의 결과를 얻을 수 있도록 한다.

다음은 다각형 속성을 고려한 벡터화 과정이다.

(1) 골격화 알고리즘에 의해 공간 데이타의 다양한 굵기를 균일하게 만든다.

제안한 벡터화는 래스터 데이타에서 골격선을 추출하기 위하여 peeling 방법을 사용한다. 지리 데이타베이스 시스템은 일반적으로 정확한 데이타를 요구하므로 peeling 방법을 이용하여 데이타의 신뢰도를 높인다[5, 10].

(그림 7)은 점 P1과 이웃하고 있는 8점의 표시이다. 각 점은 0이나 1로 구성되어 있으며, 해당 점이 있는 경우 1을 갖는다.

P9 (i-1, j-1)	P2 (i , j-1)	P3 (i+1, j-1)
P8 (i-1, j)	P1 (i , j)	P4 (i+1, j)
P7 (i-1, j+1)	P6 (i , j+1)	P5 (i+1, j+1)

(그림 7) 점 P1과 이웃하고 있는 8점의 표시
(Fig. 7) Representation of an 8-Adjacent Matrix for Point P1

다음 단계로 래스터 데이타에 대한 골격선을 제외한 나머지 점을 제거해야 함으로 해당 점을 제거하기 위해 이웃하고 있는 점을 고려하여야 한다. 점 P1을 삭제하기 위한 조건은 다음과 같다.

삭제조건 : (조건 1 and 조건 2) and (조건 3 and 조건 4)

조건1 : $2 \le B(P1) \le 6$

여기서, B(P1)은 점 P1에 존재하는 이웃점의 수로 다음과 같다.

$B(P1) = P2 + P3 + \cdots + P9$

조건2 : $A(P1) = 1$

여기서, A(P1)은 (그림 7)에서 점 P1에 이웃하고 있는 8점의 순서화 된 집합 P2, P3, ···, P9 안에 0, 1형태의 수이다.

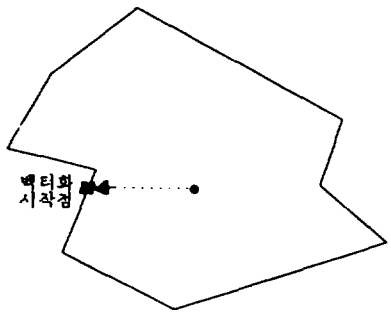
조건3 : $P2 * P4 * P6 = 0$

조건4 : $P4 * P6 * P8 = 0$

(2) 골격선이 추출된 정형화된 래스터 이미지를 추적하여 각 선을 추출한다.

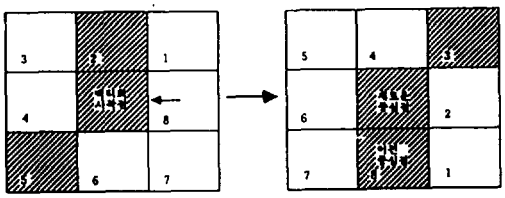
공간 데이타에 특성에 맞도록 벡터화하기 위해 지적도와 같이 다각형으로 구성된 공간 데이타를 벡터화한다. 이를 위해 다각형의 임의 내부 공간을 선택하게 되고, 그 공간 데이타를 찾기 위한 공간 탐색이 시작된다. 즉, 다각형을 벡터화하는 알고리즘은 우선 다각형 내의 임의점을 선택하면, (그림 8)과 같이 선 추적 방법을 이용하여

그 점의 좌측 방향으로 선이 나타날 때까지 탐색한다.

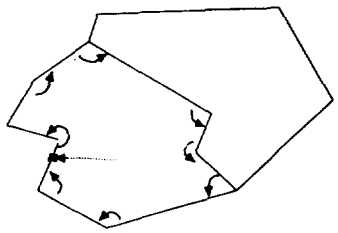


(그림 8) 다각형 내에서 공간 데이타 탐색
(Fig. 8) Spatial Data Search in a Polygon

연결된 점들을 시계 방향으로 추적하면서 선으로 만든다. 이를 위해 (그림 9)의 첫번째 그림과 같이 그 점과 이웃한 8개 점의 색을 반시계 방향으로 돌면서 중심 점의 색과 같은 색이 최초로 나온 곳으로 중심 점을 이동한다. 이때 처음으로 비교하는 점은 (그림 9)의 두번째 그림과 같이 바로 이전 중심점으로부터 반시계 방향으로 다음 점을 선택하여 비교한다. 즉 인접하여 있는 다른 공간 객체에 영향을 받지 않고, 원하는 다각형을 선 추적할 수 있다.



(그림 9) 이웃한 8점의 비교
(Fig. 9) Comparison of 8 adjacent points



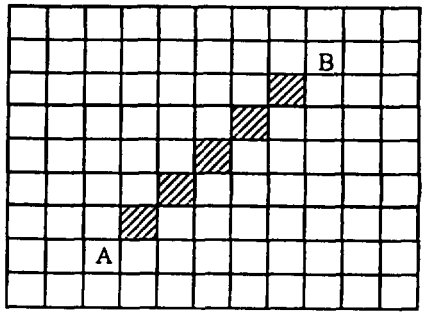
(그림 10) 반시계 방향으로의 탐색
(Fig. 10) Counter clockwise Search

따라서 한 시점에 하나의 공간 객체만을 벡터화하는 제안된 벡터화 방법은 선 추출 과정에서 공간 상황으로서 인식되고 다루어져야 할 교차 지점에 대한 위치와 교차 형태를 기억하여 위상 재구성을 하는 과정이 필요없다.

이는 다각형의 특성상 (그림 10)의 경우와 같이 다른 다각형과 접했을 경우 시계 반대 방향으로 돌면서 이전 중심점의 바로 다음 점부터 비교해 나감으로써 겹친 곳으로 잘못 추적하는 경우를 방지할 수 있다. 이는 하나의 공간 객체만을 벡터화함으로써 교차하는 지점에 대한 위치와 어떤 형태로 교차하는지를 기억하고 있다가 후에 위상 재구성을 하는 과정이 불필요하다. 이러한 과정을 처음 시작점으로 되돌아올 때까지 반복하여 모든 점을 기억한다. 이때 중간에 선이 끊어져 점이 처음으로 돌아오지 않을 경우를 고려하여 점 추적시 진행 방향으로 이웃한 점이 없더라도 그 다음 점이 있는지를 조사하여 끊어진 선에 대해서도 추적을 계속할 수 있도록 한다.

(3) 추출된 결과는 필요한 점보다 더 많은 점들을 가질 수 있으므로 필요없이 추가된 점들을 임계 기울기와 임계 거리를 고려하여 제거한다.

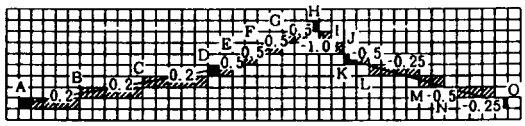
선 추적에 의해서 기억된 모든 점은 실제 필요한 점보다 더 많은 점들을 가지므로 각각의 점들을 연결하는 직선들의 기울기를 구하여 기울기의 차가 임계 기울기 이상으로 차이나면서 점과 점 사이의 거리가 임계 거리 이상으로 되는 선분으로 인식될 수 있는 필요한 점만을 다시 저장하여 벡터화를 마친다.



(그림 11) 임계 거리 미만의 선
(Fig. 11) A Line Under Threshold Distance

200 DPI(dot per inch)로 스캐닝한 1/500 지도에서 한 선이 적어도 0.5mm 이상이라면 구현시 (200dot*0.5mm)/(25.4mm)=4dot 이상의 길이를 단일 선으로 간주하여 저장하여야 한다. 즉, (그림 11)의 경우와 같이 각 점은 단일 선분으로 나뉘어지는 것이 아니라 전체를 하나의 선으로 이어서 저장하여야 한다. 본 논문에서는 이 값을 임계 거리라 하고, 선의 길이를 최소 0.5mm 즉, 임계 거리를 4dot로 하였을 경우로 구현한다.

임계 거리를 해결하여도 (그림 12)와 같이 한 개의 직선 AD가 AB, BC, CD의 3개의 직선으로 나뉘어 저장될 수 있다. 이는 저장 공간의 낭비를 초래하게 되며, 이후 처리에도 많은 시간을 요구한다. 따라서 본 논문에서는 두 직선의 기울기를 구하여 기울기 차에 대한 절대값이 임계값 미만인 경우 두 직선을 하나의 직선으로 처리하여 저장 효율을 높인다. (그림 12)는 다각형을 골격화한 예로 각 직선간의 기울기값을 계산하였다. 이때 직선 AB, BC, CD의 기울기는 모두 0.2로 하나의 선으로 처리되어야 하고, 직선 CD와 DE의 기울기는 각각 0.2와 0.5로 새로운 직선으로 간주되기 위해 점 D를 기억해야 한다. 그러나 직선 KL과 LM의 기울기는 각각 −0.5와 −0.25로 기울기 차에 대한 절대값이 0.25가 나지만 직선 KL, LM, MN, NO는 하나의 직선으로 다루어져야 한다. 즉, (그림 12)의 경우 기울기 차에 대한 임계값이 0.3 보다 크면 점 D를 인식하지 못한채 직선 AH를 생성하게 된다. 또한 0.25 미만으로 임계값을 정하게 되면 직선 KO가 하나의 직선임에도 불구하고 KL, LM, MN, NO의 4개의 직선을 생성하게 된다. 따라서 본 논문에서는 지적도에 가장 많이 나오는 다각형을 대상으로 실험을 한 결과 기울기 차에 대한 절대값, 즉 임계값을 0.3 정도로 하였을때 적합함을 확인했다. 기울기 차에 대한 임계값은 각 객체 특성에 따라 달라질 수 있으며, 등고선과 같이 보다 정밀하고 많은 점을 기억해야 하는 속성은 임계값을 더 작게하여 많은 점을 기억하게 해야 한다.



(그림 12) 기울기의 차가 임계값 미만의 선
(Fig. 12) A Line which the Difference of Slope under Threshold

4. 수집 데이타의 저장 형식

4.1 태그 비트를 이용한 압축 저장 형식

지리 데이타베이스 시스템은 지도, 항공 측량, 인공 위성 등으로부터 유도된 공간 데이타를 디지탈 형태로 수집하여 효율적으로 저장 및 관리를 해야 한다. 지리 데이타베이스 시스템은 래스터 데이타로 가장 많이 사용되고 있는 TIFF (Tagged Image File Format) 형식, GIF (Graphics Interchange Format) 형식, PCX 형식 등을 제공해야 하며, 벡터 데이타의 저장 관리를 위해 DXF(Data eXchange Format) 형식을 기본으로 제공해야 한다.

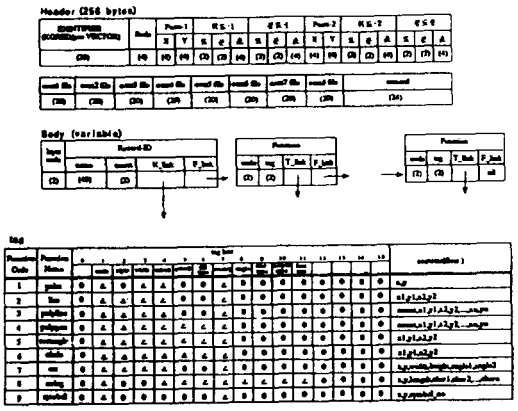
DXF 형식은 서로의 데이타 호환성을 고려하여 만든 형식으로 많은 그래픽 시스템에 의해서 해독되고 사용된다. DXF 형식은 사실상 거의 모든 소형 컴퓨터와 일부 중형 컴퓨터의 그래픽 시스템에서 표준으로 사용하는 벡터 형식으로 되어있다. 따라서 지리 데이타베이스 시스템은 데이타 호환성을 위해 반드시 DXF 형식을 지원해야 한다.

DXF 형식의 구조는 머리 부분(header section)과 몸체 부분(body section)으로 나뉘고, 각 부분은 한 라인에 한 항목씩 ASCII 형식으로 저장되며, 첫 라인에 그룹 코드(Group Code)가 들어가고 두번째 라인에는 그 그룹의 값이 들어간다. DXF 형식은 서로의 데이타 호환성을 고려하여 만든 형식으로, 많은 그래픽 시스템에 의해서 해독되고 사용되며 그래픽 시스템에서 표준으로 채택되는 벡터 형식이다. 따라서 지리 데이타베이스 시스템은 데이타 호환성을 위해 반드시

DXF 형식을 지원해야 한다.

그러나 DXF 형식은 일반적으로 많은 저장 공간을 필요로 하므로 방대한 공간 데이타의 처리 성능과 효율적인 저장 공간을 고려할때 공간 데이타에 대한 자체 관리를 위해서 반드시 효율적인 저장 형식이 필요하다.

(그림 13)은 본 논문에서 제안하고 있는 공간 데이타의 효율적인 저장 관리를 위해 태그 비트를 이용한 압축 저장 형식이다.



(그림 13) 태그 비트를 이용한 압축 저장 형식
(Fig. 13) Optimized Vector Data Format Using Tag Bit

태그 비트를 이용한 압축 저장 형식은 이진 데이타 형식으로 저장되며, 크게 256 바이트(byte)의 머리 부분과 가변 길이의 몸체 부분으로 나뉜다. 머리 부분은 전체 공간 데이타에 대한 지도 상의 축척과 좌표 그리고 연계될 벡터 데이타를 지칭한다.

몸체 부분은 각 공간 데이타를 객체별로 기술하는 부분으로 계층별 코드와 식별자 그리고 여러 개의 타입으로 구성된 출력 형태가 정의된다. 출력 형태는 태그 비트에 의한 가변 길이의 필드로 표현된다. 즉 태그 비트가 1로 표시되면 다음 필드로 각 비트 열에 대응되는 출력 형식이 정의된다.

DXF 형식의 구조와 비교하여 제안한 태그 비트를 이용한 압축 저장 형식의 가장 큰 장점은 전술한 바와 같이 이진 데이타 표현 및 태그 비트의 사용에 의한 데이타 크기를 줄여 저장 공간의 낭비를 줄인다는 것이다. 즉 x, y 좌표값에 대해 가질 수 있는 다양한 인자값을 선택적으로 기술해 줌으로써 저장할 데이타의 양을 줄인다.

4.2 저장 효율

지리 데이타베이스 시스템에서 사용하는 공간 데이타는 텍스트 데이타와 비교하여 그 크기가 방대하고 가변 길이를 갖는 것이 일반적이다. 1/500 지적도 한 도엽을 저장할 경우, 지도와 관련된 공간 데이타의 크기가 수 백 킬로 바이트(Kilo Byte)에서 수 메가 바이트(Mega Byte)까지 달하며, 한 시의 모든 지적도를 저장할 경우에는 수 기가 바이트(Giga Byte)의 저장 공간이 요구된다. 이러한 방대한 데이타의 저장 효율을 고려하는 것은 지리 데이타베이스 시스템의 성능에서 상당히 중요한 문제로 다루어진다.

본 논문에서는 DXF 형식과 제안한 태그 비트를 이용한 압축 저장 형식의 특징을 비교하여 DXF 형식의 데이타 저장율에 대한 제안한 태그 비트를 이용한 압축 저장 형식의 데이타 저장율을 평가해 본다.

제안한 태그 비트를 이용한 압축 저장 형식은 각 객체별로 태그 비트를 두어 그 객체에 대한 필요한 정보만을 기술하여 데이타 저장에 대한 효율을 높인다.

〈표 1〉은 점, 선, 원에 대한 제안한 태그 비트를 이용한 압축 저장 형식과 DXF 형식의 저장 구조 예와 각각에서 요구되는 저장 공간의 크기 그리고 DXF 형식에서의 데이타 저장 크기에 대한 제안한 태그 비트를 이용한 압축 저장 형식의 데이타 저장 크기에 대한 저장 효율을 나타낸다.

이는 제안한 태그 비트를 이용한 압축 저장 형식으로 데이타를 저장할 경우 DXF 형식으로 저장하는 것보다 저장 공간면에서 70~80%의 성능이 향상됨을 보여준다. 즉 DXF 형식으로 저장할 경우 요구되는 저장 공간의 약 1/3~1/4정도의 저장 공간으로 제안한 태그 비트를 이용한 압축 저장 형식으로 저장하여 저장 공간의 낭비

를 막을 수 있으며 빠른 데이타 접근을 가능하게 한다.

〈표 1〉 제안한 태그 비트를 이용한 압축 저장 형식과 DXF 형식의 비교(I)

〈Table 1〉 Comparison between Proposed Vector Data Format using Tag Bit and DXF Format(I)

		제안한 태그 비트를 이용한 압축 저장 형식	DXF 형식
저장 형태		Binary file	ASCII file
점	형식 예	(00 01) (28 00) (00 0F) (00 01) (3E D9 2A 73) (80 00 00 00) => code(2 bytes), tag(2 bytes), 별명(2 bytes), 라인 타입(2 bytes), x 좌표(4 bytes), y 좌표(4 bytes)	0 /* 엔티티 시작 표시 */ VERTEX /* 엔티티 타입 */ 8 /* 레이어 명 시작 */ 6 /* 레이어 번호 */ 62 /* 컬러 번호 시작 */ 15 /* 컬러 번호 */ 6 /* 라인 타입 시작 */ CONTINUOUS/* 정의된 라인 타입 */ 10 /* x값 시작 */ 0.000006 /* x값 */ 20 /* y값 시작 */ -0.008191 /* y값 */
	저장크기	16 bytes	80 bytes(CR+LF 포함)
	효 율	80 % 개선	
선	형식 예	(00 02) (28 00) (00 0F) (00 01) (bF 00 40 BF) (E0 00 00 00) (BF 14 7Z 7D) (C0 00 00 00) => code(2 bytes), tag(2 bytes), 별명(2 bytes), 라인 타입(2 bytes), x1 좌표(4 bytes), y1 좌표(4 bytes), x2 좌표(4 bytes), y2 좌표(4 bytes)	0 /* 엔티티 시작 표시 */ LINE /* 엔티티 타입 */ 8 /* 레이어 명 시작 */ 6 /* 레이어 번호 */ 62 /* 컬러 번호 시작 */ 15 /* 컬러 번호 */ 6 /* 라인 타입 시작 */ CONTINUOUS/* 정의된 라인 타입 */ 10 /* x1값 시작 */ -0.000031 /* x1값 */ 20 /* y1값 시작 */ -0.000078 /* y1값 */ 11 /* x2값 시작 */ 0.000136 /* x2값 */ 21 /* y2값 시작 */ -0.000003 /* y2값 */
	저장크기	24 bytes	110 bytes(CR+LF 포함)
	효 율	78 % 개선	
원	형식 예	(00 06) (28 00) (00 0F) (00 01) (00 00 00 00) (00 00 00 00) (3E FF FB 4B) (3E FF FB 4B) => code(2 bytes), tag(2 bytes), 별명(2 bytes), 라인 타입(2 bytes), x1 좌표(4 bytes), y1 좌표(4 bytes), x2 좌표(4 bytes), y2 좌표(4 bytes)	0 /* 엔티티 시작 표시 */ CIRCLE /* 엔티티 타입 */ 8 /* 레이어 명 시작 */ 7 /* 레이어 번호 */ 6 /* 라인 타입 시작 */ CONTINUOUS/* 정의된 라인 타입 */ 10 /* 원 중심점 x 시작 */ 0.000000 /* x값 */ 20 /* 원 중심점 y 시작 */ 0.000000 /* y값 */ 40 /* 반경 시작 */ 0.000061 /* 반경값 */
	저장크기	24 bytes	84 bytes(CR+LF 포함)
	효 율	71 % 개선	

〈표 2〉는 인천 남구의 일부 지역인 1/500 도면 9개에 대해서 각각 DXF 형식과 제안한 태그 비트를 이용한 압축 저장 형식으로 저장한 후 저장된 데이타 크기를 확인한 것으로 약 3/10정도의 저장 공간으로 저장될 수 있음을 보여준다.

〈표 2〉 제안한 태그 비트를 이용한 압축 저장 형식과 DXF 형식의 비교(II)

〈Table 2〉 Comparison between Proposed Vector Data Format using Tag Bit and DXF Format(II)

e1318p.dxf	*716071*	Apr	5	12:10	e1318p.vet	*207189*	Aug 22 13:52
e1319p.dxf	*733417*	Apr	5	12:10	e1319p.vet	*215331*	Aug 23 16:51
e1320p.dxf	*876162*	Apr	5	12:10	e1320p.vet	*211732*	Aug 23 16:54
e1323p.dxf	*333021*	Apr	5	12:10	e1323p.vet	*83038*	Aug 23 17:04
e1324p.dxf	*853463*	Apr	5	12:10	e1324p.vet	*167272*	Aug 24 09:21
e1325p.dxf	*783162*	Apr	5	12:10	e1325p.vet	*209404*	Aug 23 17:12
e1803p.dxf	*434257*	Apr	5	12:10	e1803p.vet	*128809*	Aug 23 17:16
e1804p.dxf	*356345*	Apr	5	12:10	e1804p.vet	*125816*	Aug 23 17:18
e1805p.dxf	*1121230*	Apr	5	12:10	e1805p.vet	*270268*	Aug 23 17:40

5. 결 론

우리 나라는 미국이나 호주와 같이 도시 계획 초기 단계부터 컴퓨터에 의해 설계되지 않아 도시에 관한 데이타베이스가 구성되어 있지 않다. 그러나 선진국은 도시 계획이 잘되어 있어 단위 도면당 적은 데이타를 수동식 입력 방법으로 처리할 수 있는 반면 인구 밀도가 높은 우리나라에서는 단위 도면당 많은 데이타를 입력해야 하므로 수동식 입력 방법을 사용하게 되면 많은 인력과 예산이 소요된다. 따라서 국내 지리 환경에 적합한 공간 데이타의 자동 입력 방법과 수집된 데이타의 효율적인 저장 방법이 요구되었다.

기존의 벡터화 알고리즘은 지리 데이타베이스 시스템에서 사용하는 지도와 같은 복잡한 도면을 벡터화하는 경우 스캐닝한 도면 전체를 대상으로 벡터화하므로 불필요한 데이타도 벡터화를 하게 되고, 글자나 다른 선에 의해 공간 데이타가 끊어지거나, 스캐닝시의 오류로 인한 선들을 자동으로 인식하지 못하는 경우가 발생한다. 또한 각 객체가 갖고 있는 속성을 고려하지 않은채 벡터화하여 추후 공간 객체 단위로 데이타를 수정해야 한다는 문제점이 있다.

본 논문에서는 정확한 공간 데이타를 수집하기 위해서, 도면 전체를 벡터화하지 않고 사용자가 속성 정보를 입력하는 시점에서 그 공간 데이타만을 벡터화하는 방법을 제안하였다. 이로써 공간 데이타의 속성을 고려하여 벡터화를 원하는 공간 데이타만을 정확하게 벡터화하여 입력 시간의 절감 뿐만 아니라 많은 인력과 예산을 절감할 수 있게 되었다. 또한 수집된 방대한 공간 데이타에 대해서 자체 공간 데이타 저장 형식인 태그 비트를 이용한 압축 저장 형식을 제안하므로써 방대한 공간 데이타를 효율적으로 관리할 수 있게 되었다.

앞으로 등고선 등 다양한 공간 데이타의 벡터화 기법에 대한 연구가 필요하며, 또한 공간 데이타 저장 형식도 이러한 공간 데이타의 특성에 따라 보다 효율적으로 저장할 수 있는 기법이 요구된다.

참 고 문 헌

[1] P. A. Burrough, 'Principles of Geographical Information System for Land Resources Assessment', *Clarendon*, pp. 57-80, 1986.

[2] K. C. Clarke, 'Analytical and Computer Catography', Prentice-Hall, pp. 186-197, 1990.

[3] J. H. Kim, H. Y. Bae, "Rr-tree:The Design of Efficient Access Method for Spatial Objects", *Proc. of the Far East workshop on GIS, World Scientific*, pp. 91-105. 1993.

[4] H. P. Kriegel, T. Brinkhoff and R. Schneider, "The Combination of Spatial Access Methods and Computational Geometry in Geographic Database Systems", *Proc. of 2nd Symposium SSD'91*, pp. 5-21, 1991.

[5] L. Lam, S. W. Lee, and C. Y. Suen, "Thinning Methodologis-A Comprehensive Survey", *IEEE Trans. Patt. Anal. Machine Intell.*, Vol. 14, No. 9, pp. 869-885, 1992.

[6] B. C. Ooi, R. S. Davis and K. J. McDonell, "Extending A DBMS for Geographic Applications", Proc. of IEEE Data Engineering, pp. 590-597, 1989.

[7 J. Palimaka, et al., "Integration of a Spatial and Relational Database Within a Geographic Information System", Proc. of Int. Workshop on GIS, 1987.

[8] D. J. Peuquet, "An Examination of Techniques for Reformatting Digital Cartographic Data/Part 1:The Raster-To-Vector Process", *Cartohrraphica*, Vol. 18, No. 1, pp. 34-48, 1981.

[9] D. J. Peuquet, "Data Models for Very Large Geographic Databases", *Proc. of Int. Workshop on GIS*, 1987.

[10] T. Y. Zhang and C. Y. Suen, "A Fast Parallel Algorithm for Thinning Digital Patterns", Comm. ACM, Vol. 27, No. 3, pp. 236-239, 1984.

[11] 김재홍, "KORED/GEO:지리 정보 시스템을 위한 관계 데이타베이스 시스템의 확장", 인하대학교 박사학위논문, 1994.

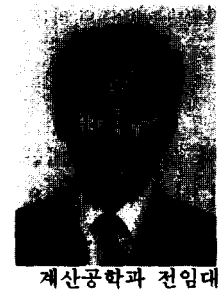
[12] 김재홍, 배해영, "공간 데이타 처리를 위한 확장된 저장 관리 시스템", 한국 GIS 학회지, Vol. 1, No. 1, 1993.

[13] 김종훈, 김재홍, 배해영, "KORED/GEO의 공간 데이타 처리기의 설계 및 구현", 한국정보과학회 '92 가을 학술발표논문집, Vol. 19, No. 2, pp. 55-58, 1992.

[14] 박동선, 김재홍, 배해영, "KORED/GEO의 공간 데이타 연산", 한국정보과학회 '93 봄 학술발표논문집, Vol. 20, No. 1, pp. 19-22, 1993.

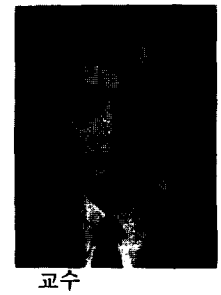
김 종 훈

1991년 인하대학교 전자계산공학과(공학사)
1993년 인하대학교 대학원 전자계산학과(공학석사)
1993년~현재 인하대학교 대학원 전자계산학과 박사과정
관심분야 : 데이타베이스(특히, 멀티미디어 데이타베이스 시스템, 지리 정보 시스템, 가상 현실)



김 재 홍

1988년 인하대학교 전자계산학과(이학사)
1990년 인하대학교 대학원 전자계산학과(이학석사)
1994년 인하대학교 대학원 수학과 전자계산학(이학박사)
1993년~현재 인하대학교 전자계산공학과 전임대우
관심분야 : 데이타베이스(특히, 멀티미디어 데이타베이스 시스템, 지리 정보 시스템), 컴퓨터 그래픽스



배 해 영

1976년 인하대학교 응용물리학과(공학사)
1986년 연세대학교 대학원 전자계산학과(공학석사)
1990년 숭실대학교 대학원 전자계산학과(공학박사)
1985년 Univ. of Houston 객원교수
1982년~84년 인하대학교 전자계산소 소장
1982년~현재 인하대학교 전자계산공학과 교수
관심분야 : 데이타베이스(특히, 멀티미디어 데이타베이스 시스템, 실시간 데이타베이스 시스템, 지리 데이타베이스 시스템)