
방향성 정보를 이용한 최적 경로 탐색 알고리즘의 설계

김진덕* · 이현섭** · 이상욱***

A Design of Optimal Path Search Algorithm using Information of Orientation

Jin-Deog Kim* · Hyun-Seop Lee** · Sang-Wook Lee

이 논문은 2004년도 Brain Busan 21 사업에 의하여 지원되었음

요 약

텔레매틱스의 주요 응용분야인 차량 항법 시스템은 GIS의 지도기술과 GPS의 측위 기술에 융합되어 탄생한 것이다. 기존의 시스템들은 정점과 정점 간의 최단경로를 중심으로 한 경로 탐색을 제시하였지만 교통량의 변화와 교통사고 및 수시 병목 구간 등의 교통 변화에 대해서는 능동적인 대처를 할 수 없는 시스템이다. 바꿔 말하면 현실적으로 최단경로 탐색 보다는 교통 정보를 파악하며 최단시간에 목적지로 도착하는 최적경로 탐색 시스템이 필요하다. 이 논문에서는 교통 정보를 기반으로 최적 경로를 탐색하는 알고리즘을 제안한다. 구체적으로 최적 경로 탐색을 위한 시스템 구조를 제안하며 기존의 최단경로 탐색 알고리즘의 문제점을 살펴보고 수집되는 교통정보의 방향성을 활용하는 최적 경로 탐색 알고리즘을 새롭게 제안한다.

ABSTRACT

Car navigation system which is killer application fuses map management techniques into GPS techniques. Even if the existing navigation systems are designed for the shortest path, they are not able to cope efficiently with the change of the traffic flow and the bottleneck point of road. Therefore, it is necessary to find out shortest path algorithm based on time instead of distance which takes traffic information into consideration. In this paper, we propose a optimal path search algorithm based on the traffic information. More precisely, we introduce the system architecture for finding out optimal paths, and the limitations of the existing shortest path search algorithm are also analyzed. And then, we propose a new algorithm for finding out optimal path to make good use of the orientation of the collected traffic information.

키워드

텔레매틱스, 이동 객체, 최적 경로

1. 서 론

최근 텔레매틱스의 다양화된 보급과 함께 수요와 공급이 급격히 늘어가는 분야가 있다. 항공 및

*동의대학교 컴퓨터공학과 조교수

** 동의대학교 컴퓨터공학과 석사과정

*** 경상대학교 정보통신공학과

해상에서는 물론 내륙교통에서도 원하는 목적지에 정해진 시간 안에 도착하기 위하여 효율적인 경로 검색은 필수적인 요소가 되어 가고 있다.

목적지로 가는 경로가 초행이거나 어떠한 경로로 가는 것이 가장 가까운 거리인가 하는 정보들은 네비게이션의 가장 기본적인 조건이다. 이런 조건들을 모두 포함하고 교통량이 갑자기 증가하는 경우, 갑작스런 사고로 인한 도로 마비, 출퇴근의 교통상황 등의 예기치 못한 상황들을 계산하여 가장 빠른 시간 안에 목적지까지 도달하는 최적경로를 탐색하는 것이 가장 중요한 핵심이라고 할 수 있다. 이 논문에서는 수많은 교통 정보들을 획득하여 가장 빠른 시간 안에 목적지에 도달할 수 있는 최적경로 탐색 시스템을 제안한다. 최적경로 탐색을 위해 수집된 도로 교통 정보를 활용하는 알고리즘을 제안한다. 구체적으로 말하자면 방향성을 지니고 있지 않은 Dijkstra 알고리즘에 방향성을 부여하고 도로 교통망의 교차로 회전 정보와 각 간선의 평균 속도 정보를 기반으로 하는 최적 경로 탐색 알고리즘을 제안한다.

이 논문의 구성은 다음과 같다. 제 2장에서는 기존의 네비게이션 시스템들의 구동원리와 문제점 등의 관련 동향에 대하여 알아보고, 3장에서는 교통 정보 기반 최적경로 탐색을 위한 시스템 구조를 제안하며 최적경로 탐색을 위한 알고리즘을 제안한다. 마지막으로 4장에서 결론을 맺는다.

II. 관련연구

I현재 서비스 되고 있는 대표적인 네비게이션들의 특징될 만한 몇몇 스펙들을 살펴보면 다음과 같다. KT에서 서비스하는 KT Bizemeka Q-NAVI[2]라는 시스템이 있다. 현재의 자신의 위치를 계산하여 목적지까지의 경로를 내부에 저장되어 있는 지도 데이터에서 연산하여 최단거리를 화면에 디스플레이 해준다. 시간과 도로상황을 염두 하지 않은 단순한 최단거리일 뿐이다. 대신에 일부의 기능에서 회피지역을 지정하여 연산은 가능하므로 확실한 정보를 알고 입력하면 최적경로에 가까운 연산 결과를 얻을 수 있다. SK TELECOM에서 서비스하고 있는 NATE drive[3] 또한 네비게이션의 일종이라고 볼 수 있는데 중앙 처리기에서 지도 데이터를 분석하여 출발노드와 도착 노드가 지정되면 그 사이의 노드들을 계산하여 최단거리를 구해내는 방식을 채택하고 있다. 특징적으로 도로교통관리공단에서 제공하는 교통정보를 구해낸 최단거리에 반영하여 재연산한 결과를 각각의 요구 클라이언

트에게 보내준다. 그러나 교통사고, 긴급정체 등의 급작스런 도로상의 예외가 발생하게 되면 그 정보들을 즉각적으로 반영할 수 없다. 이 밖에도 이와 관련한 연구[4,5,6]가 진행되고 있지만 아직까지는 위의 두 가지 스펙에서 크게 벗어나지 않고 있다.

이 논문에서는 위에서 설명한 기존의 방식과는 달리 중앙서버에서 클라이언트의 교통 정보들을 수집하고 수집된 데이터를 분석하여 진행방향의 제한이나 U-턴 같은 특수한 도로상황과 현재 유입된 교통량을 탐색하여 최적의 경로를 클라이언트에게 전송하는 방식에 대하여 제안한다.

III. 방향성 기반 최적경로 알고리즘

3.1 최적 경로 탐색 시스템 구조

그림 1은 이 논문에서 제안하고 있는 교통 정보 기반 경로 탐색 시스템 구조를 도식화한 것이다. 상당량의 클라이언트로부터 많은 양의 데이터를 입력 받고 처리 하기위하여 대용량의 서버가 존재한다. 그리고 서버에서 처리된 데이터를 저장하기 위하여 도로 교통정보 DB를 만들고 주기적으로 데이터를 Update 시키는 구조이다. 각각의 클라이언트들은 서버와의 수신을 위한 텔레매틱스 단말기를 보유하고 있으며 자신의 위치와 움직이고 있는 속도 및 진행 방향과 클라이언트간의 분포도를 분석 할 수 있게 GPS 수신기를 설치하여 위성과의 통신으로 위치 정보를 획득 할 수 있다.

3.2 최단 경로 탐색 알고리즘

두 정점사이의 거리를 가중치로 갖는 그래프로 구성된 네트워크의 최단 경로 검색을 위해 주로 Dijkstra 알고리즘[1]이 사용된다. 이 알고리즘은 총 3개의 단계로 표현되어질 수 있다. 표 1은 각 단계에서 사용되는 요소들을 표로 나타낸 것이다.

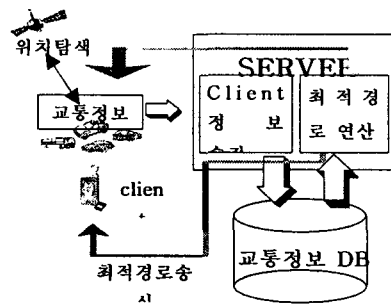


그림 1. 경로 탐색 시스템 구조
Fig. 1 Path Navigation System Architecture

표 1. Dijkstra 알고리즘의 표현 요소
Table 1. Symbols for Dijkstra algorithm

기호	설 명
N	정점들
s	출발 정점
e	도착 정점
T	알고리즘에 의해 지금까지 포함된 정점 집합
$w(i, j)$	정점 i 부터 j 까지의 가중치
$L(n)$	출발점에서 n 지점 까지 가는데 드는 가중치

Step 1

1. $T = \{s\}$
2. $L(s) = 0$
3. $L(n) = \infty$ for all $n \notin T$

Step 2

1. Find $x \notin T$ such that $L(x) = \min_{j \notin T} L(j)$
2. $T = T + x$
3. if $(L(x) = \infty)$ or $(x = e)$ then Finish.

Step 3

1

$$L(n) = \min[L(n), L(x) + w(x, n)] \text{ for all } n \notin T$$

2. Goto Step2

3.3 새로운 최적경로 탐색 알고리즘

기존의 Dijkstra 알고리즘[1]을 교통 정보 시스템의 최적경로를 찾기 위한 알고리즘으로 사용하기에는 부적합한 부분이 존재한다. 도로에는 직진, 우회전, 좌회전, 일방통행 등의 진행 방향에 따른 진행 가능/불가능한 부분이 존재한다. Dijkstra 알고리즘에서는 연결되어 있는 정점은 모두 연산 대상에 포함하는데 위에서 설명한 도로의 특징에는 그대로 대입시켜 사용할 수 없다. 즉 정점을 연결

하는 각 간선에 대한 정보를 부여하고 이 정보들을 고려하여 최적경로를 탐색해야 한다. 또한 U턴과 교차로 등의 특수한 도로상황에서도 Dijkstra 알고리즘을 사용한 검색은 불가능하다. Dijkstra 알고리즘은 같은 정점은 두 번 탐색하지 않기 때문이다. 기존의 Dijkstra 알고리즘은 소스에서 임의의 정점들 모두에게 도달하는 가중치를 알아내고 있으나 본 논문에서 Dijkstra 알고리즘을 사용할 시에는 출발지에서 목적지까지의 가중치만을 알아내면 되기 때문에 기존의 알고리즘의 종료 조건에 ($x=e$)을 추가하여 목적지까지의 가중치를 구하였다면 알고리즘을 종료하도록 수정하여 사용하였다.

본 논문에서의 최적경로 알고리즘 역시 총 3개의 단계로 나타내어 질 수 있다. 표 2는 최적경로 알고리즘의 각 단계에서 사용되는 요소들을 표로 나타낸 것이다. 이 논문에서 제안한 최적 경로 탐색 알고리즘은 기존의 Dijkstra 알고리즘과는 달린 정점 i 에서 인접된 정점(를 경유한 정점 k (k 는 i 가 될 수도 있다) 즉, 각각 정점 i, j, k 를 연결하는 경로의 가중치 $w(i, j, k)$ 를 도로 구조에 따른 진행 불가능/가능 여부에 맞추어 가중치를 ∞ 또는 정점 j 와 k 사이의 가중치 $w(j, k)$ 둘 중 하나로 줄 수 있는 기능을 가지고 있으며 또, 출발점 s 에서 e 까지의 선분 (s, e) 에서 거리 l 이상 떨어진 정점 q 에 대해서는 검색의 대상에서 제외 하여 알고리즘의 성능을 개선할 수 있도록 하는 최적경로 알고리즘이다.

표 2. 최적 경로 알고리즘의 표현 요소
Table 2. Symbols for Optimal Path Algorithm

기호	설 명
N	정점들, 각 정점은 자신의 인접 정점과 연결된 방향의 수 n 에 따라 n 개씩 존재(본 논문에서는 $n=8$)
b	출발 지점으로 오기 직전의 정점
s	출발 정점
e	도착 정점
T	알고리즘에 의해 지금까지 포함된 각 정점들
$w(i, j, k)$	정점 i 에서 j 를 경유한 k 까지의 가중치 (이 값이 ∞ 이면 진행불가능 경로)

$w(i, j)$	정점 i 부터 j 까지의 가중치
$L(n, m)$	출발점에서 n 지점 까지 가는데 드는 가중치중 n 의 인접정점 m 을 경유한 가중치
l	선분 (s, e) 에서 검색을 제한할 거리 상수
$\Delta\{a, \overline{b}, \overline{c}\}$	정점 a 와 정점 b 와 c 가 이루는 선분 $(\overline{b}, \overline{c})$ 사이의 거리

Step 1

1. $T = \{s\}$ (s with direction from b)
- 2

$$w(i, j) = \infty \text{ for all } i \text{ such that } \Delta\{i, \overline{(s, e)}\} > l, \text{ for all } j \in N$$

3. $L(s, b) = 0$
4. $L(n, m) = \infty$ for all $n \notin T$ for all m nearby n

Step 2

1. Find $x \in T$ such that $L(x, m) = \min_{j \in T} L(j, n)$ for any m, n nearby j, x
2. $T = T + x$
3. if $(L(x, m) = \infty)$ or $(x = e)$ then Finish.

Step 3

- 1
- $$L(n, m) = \min[L(n, m), L(x, k) + w(k, x, n)]$$
- for all $n \notin T$, for any m, k nearby n, x

2. Goto Step2

본 논문의 알고리즘 구현에서는 디스크에 정점들의 집합 N 의 한 정점 i, j, k 를 경유한 방향을 나타낼 때의 편의를 위해 인접한 정점들의 방향은 최대 8방향으로 하였다.

그림 2는 본 논문에서 제안한 최적 경로 알고리즘을 위한 도로 네트워크의 노드 구조를 나타낸다. 각각의 노드는 다음의 정보들을 저장하고 있다. 각 노드들의 식별번호와 그와 인접된 다른 노드들의 번호를 저장하고 있다. 또한 인접된 노드들을 통하여 진행할 수 있는 다른 노드들의 방향정보도 가진

다.

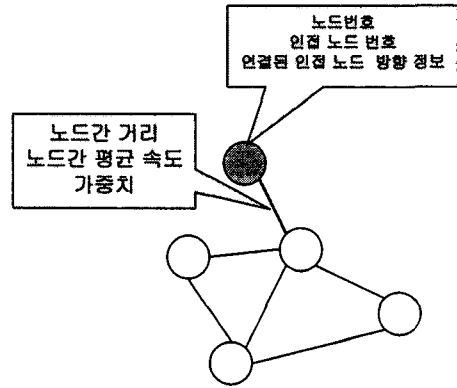


그림 2. 노드 구조
Fig. 2. Node Structure

각각의 노드들을 연결하고 있는 간선들은 노드와의 거리, 노드간의 평균속도, 거리와 속도를 조합하여 얻어낸 가중치의 값을 저장한다.

그림 3은 노드 데이터가 저장되는 방법이다. 노드들의 Index번호를 따라 그 노드와 연결되어 있는 인접노드들의 수와 이동할 수 있는 노드들의 방향데이터를 이진수 8비트형식으로 저장을 한다.

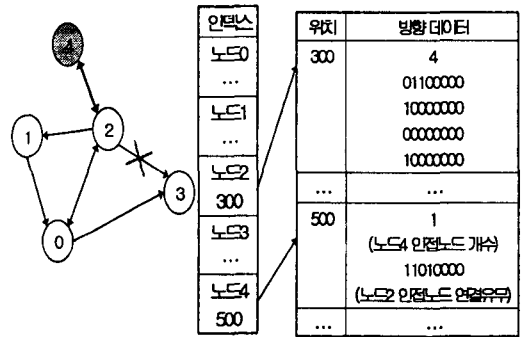


그림 3. 노드 데이터
Fig. 3. Node Data

그림 4는 진행 방향과 가중치 등을 정점과 간선으로 표현한 예이다. 진행방향이 있고 없음에 따라 Dijkstra 알고리즘을 사용해 출발지 1에서부터 목적지 6으로 갈 수 있는 노드를 표현하고 있으며 변형 시킨 알고리즘을 사용해 목적지 정점으로 갈 수 있는 가중치를 비교해 볼 수 있다. 기존의 Dijkstra 알고리즘을 사용하면 1->4->5->6의 경로를 탐색 결과로 나타낸다. 그러나 방향성을 고려하면 1->2->3->4->5->6의 탐색 결과가 도출 된다. 이

와 같이 일반 도로에서는 특수한 방향성이 있기 때문에 알고리즘의 수정이 불가피 하다.

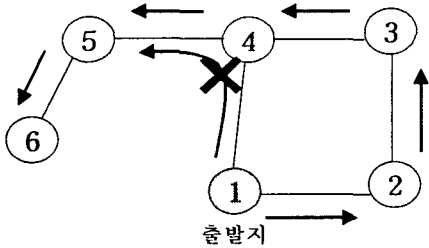


그림 4. 진행 방향이 존재하는 도로
Fig. 4. Road with Orientation

진행 방향에 따른 정보를 기존의 Dijkstra 알고리즘에 적용시키는 방법으로 이 논문에서 채택한 방법은 기존 Dijkstra 알고리즘 구현에서 $\omega(i,j)$ 가 정점 i 에서 정점 j 까지의 가중치를 나타내는 대신 $\omega(i,j,k)$ 를 사용하여 i 를 기준으로 했을 때 j 를 경유하여 k 까지 갈 때의 가중치를 나타내도록 하여 진행 방향에 따라서 j 에서 k 까지의 가중치를 달리 반환하여 진행 가능/불가능 한지를 나타내도록 한다. 그리고 각 정점들은 연결되어있는 정점들을 통하여 그 정점과 다시 연결되어있는 인근 정점으로 이동 할 수 있는지의 정보를 지니고 있다. 그리고 각 정점들 사이에 존재하는 간선에는 정점과 정점 사이의 가중치 값을 가지고 있다.

그림 5에서는 출발지 1에서 목적지로 진행할 때 U-턴을 통하여 진행하면 가중치가 단순 정점이동에 비하여 작은 것을 알 수 있다. 위와 같은 경우 정점 1을 출발하여 정점 3으로 진행 할 때 정점 2를 통하여 3으로 가는 경우 좌회전이 되지 않으므로 왼쪽 하단에 보이는 우회로를 선택 하여야 한다. 그러나 정점 2쪽으로 우회전을 한 뒤 U-턴을 통하여 정점 3으로 갈 수 있는 방법도 있다. 현실의 도로상황에서는 빈번히 일어나는 경우이다.

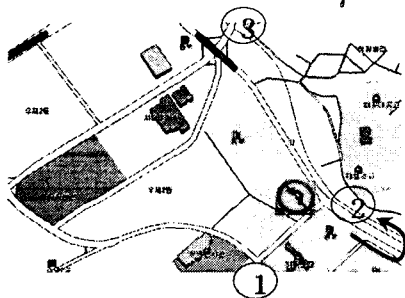


그림 5. U-턴 도로와 가중치
Fig. 5. U-turn and Weight

이 경우에 정점 2를 두 번 거쳐야 하는데 한 정점을 두 번 이상 통과해야하는 연산을 해야 할 경우에 Dijkstra 알고리즘을 그대로 사용하면 필요한 연산결과를 얻을 수 없는 경우가 발생한다. U-턴과 교차로 등을 통과해서 한 정점이 2번 이상 들어간 최적경로를 찾아내기 위해서는 기존 Dijkstra 알고리즘에서 한 정점에 대하여 한 개의 최소가중치의 합을 가지는 점을 수정하여 해당 정점과 연결된 모든 간선들을 통해 들어오는 모든 경우를 따로 관리하여 한 정점 당 1개의 경로가 아닌 한 정점에 대해 연결된 각 간선 당 1개의 경로가 존재하게 된다. 그림 6은 기존의 Dijkstra 알고리즘을 사용하여 최단경로 탐색하여 그 결과를 나타낸 그림이며 그림 7은 이 논문에서 제시한 알고리즘을 구현한 것으로 거리와 속도를 모두 사용하여 새로운 값을 연산하고 이동 방향과 U-턴을 고려하여 최적경로를 탐색한 수행한 결과이다. 그 결과 기존의 알고리즘에 의한 경로와는 달리 최적 경로 탐색을 위해 교차로의 회전 정보 및 방향정보를 이용하였으며, 또한 U-턴 경로를 택하여 시간적으로 최소인 경로를 유도하고 있다. 따라서 제안한 최적 경로 탐색 알고리즘은 기존의 Dijkstra 알고리즘과는 전혀 다른 경로가 선정됨을 알 수 있다.

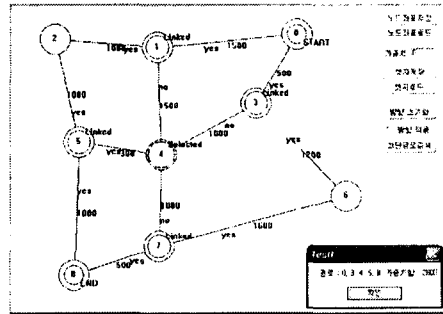


그림 6. Dijkstra 알고리즘 수행 결과
Fig. 6. Result of Dijkstra Algorithm

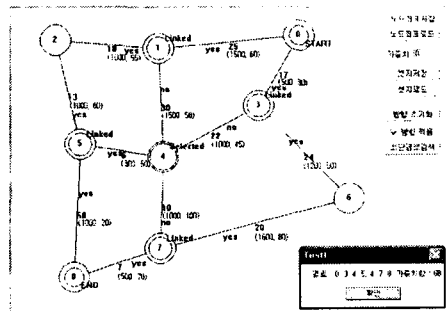


그림 7. 최적경로 탐색 알고리즘 수행 결과
Fig. 7. Result of Optimal Path Algorithm

3.4 성능 향상 방안

앞서 설명한 변형 알고리즘으로 최적경로를 탐색할 수는 있으나 거미줄처럼 연결되어있는 도로 상에서 이 알고리즘을 수행하면 연산시간이 늘어나는 결과를 초래한다. 따라서 이렇게 늘어난 연산 시간을 줄일 수 있는 방법을 제안한다.

첫째, 연산을 수행하였을 때 가장 먼저 도착하는 경로가 존재하면 즉시 연산을 종료하는 방법이다. 최적 경로 탐색을 위하여 많은 수정을 하였으나 기본적인 연산은 Dijkstra 알고리즘으로 수행하므로 Dijkstra의 가장 먼저 도착하는 간선의 경로가 최단거리라는 특징을 이용하여 연산할 수 있다.

둘째, 탐색경로가 상당한 거리 일 때는 별도의 연산을 수행하는 방법이다. 그림8은 부산의 어느 출발점에서 서울의 어느 목적지까지의 최적경로를 탐색한다고 가정하면 서울에서 부산까지의 모든 정점들을 계산하지 않고 현재 위치에서 고속도로와 대표국도의 시작점까지의 연산만 수행하는 방법이다. 즉 특정한 구간에서는 필수적으로 이용을 해야 하는 노드를 설정하는 것을 의미한다.

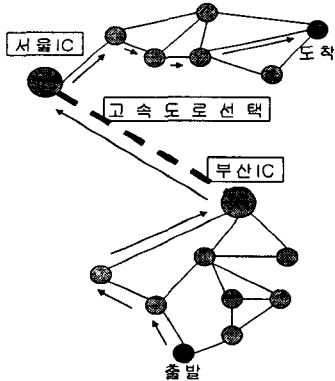


그림8. 탐색경로가 큰 거리 경로 검색
Fig. 8. Path Search with Long Path

셋째, 영역을 지정하여 탐색하는 방법이다. 검색이 진행되는 위치에서 일정한 거리 이상 목적지로부터 멀어 지게 되면 연산을 수행하지 않는 방법이다. 즉 목적지 방향으로만 검색 연산을 진행하며 다른 방향으로 검색 하는 것은 일정 영역 안까지만 검색을 하는 방법이다. 그림9는 설명된 방법을 그림으로 도식화 한 것이다. 하얀점선 범위로 검색 연산을 수행하며 오렌지 색의 범위는 검색 대상에서 제외를 하는 방법이다.

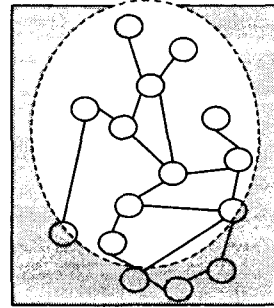


그림 9. 영역 지정 탐색
Fig. 9. Path Search with Restricted Region

3.5 예외상황 대체 방안

만약 탐색한 최적경로 대로 이동을 한다면 이상은 없지만 그렇지 못한 예외 상황이 발생할 가능성 또한 적지 않다. 일어날 수 있는 예외 상황들에 대한 대체 방안을 제안한다.

첫째, 발생할 수 있는 예외 상황은 이용자가 탐색 경로를 벗어나서 주행을 할 때이다. 경로를 벗어나면 이용자에게 그 상황을 알리고 현재위치들 토대로 서버에게 재 연산을 요청한다.

둘째, 사용자가 회피지역과 필수경유 지역을 선정하고자 하는 경우가 발생할 수 있다. 이 경우에는 사용자가 회피 및 필수 경유 정점을 선택하면 서버에서 그에 대응한 연산을 실행하여 전송한다. 먼저 회피할 정점은 전체 연산에서 각 정점에서 회피할 정점으로 들어가는 정점들의 간선에 대한 가중치 값을 무한대로 설정하여 연산하면 된다. 그리고 필수 경유 지역은 서버에서 필수경유 정점을 목적지로 잡고 제 1연산을 수행하고 경로 탐색이 끝나면 목적지로 잡았던 정점을 다시 출발지로 선택하여 최종 목적지와 의 제 2연산을 수행하여 두 번의 연산을 복합하여 탐색경로를 전송한다.

셋째, 운행 중 도착 예상 시간 안에 목적지에 도착을 못했을 경우도 예외 상황이라고 할 수 있다. 이 경우 시간의 경과로 인하여 근처의 도로상황이 많이 변경 되었을 가능성도 있고 진행 방향에 사고가 발생 하였을 수도 있다.

그림10과 그림11은 제안한 성능 향상 방안중에서 영역 지정 탐색을 구현한 것이다. 성능 개선 전의 최적경로 검색과 성능 개선 후의 최적경로 검색을 비교한 것으로 성능 개선후의 경로가 짧은 것을 알 수 있다.

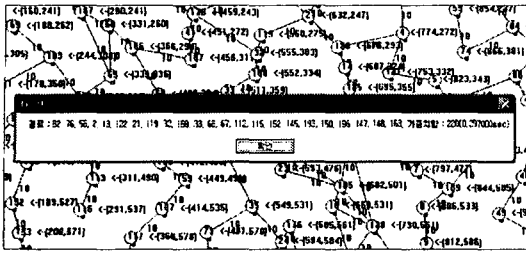


그림10 성능 개선 기법 적용 전
Fig. 10. Before Performance Enhancement

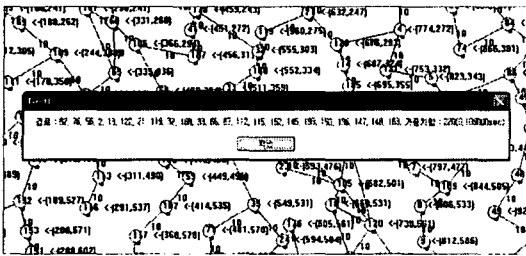


그림11. 성능 개선 기법 적용 후
Fig. 11. After Performance Enhancement

표 3은 탐색기법들을 비교한 것이다. 기존의 단순 최단 경로 검색인 Dijkstra 알고리즘과 현재 정보통신업체에서 서비스 중인 상용화 알고리즘에 비해 이 논문에서 제안하고 구현한 최적경로 알고리즘은 보다 최신의 교통정보를 이용함으로써 이동시간이 최소화되는 경향이 있는 반면, 최적 경로 알고리즘을 선정하기 위한 검색 시간은 증가할 것으로 판단된다. 그러나 경로 선정 검색은 주로 대형 서버에서 수행되며 그 시간 또한 각 차량 운전자의 이동 시간에 비하여 극히 짧은 시간이므로 본 논문에서 제안한 최적 경로 알고리즘은 텔레매틱스 응용과 ITS 등에 효과적으로 사용될 수 있다.

표 3. 탐색기법 비교
Table 3. Comparison of Path Search Methods

	Dijkstra 알고리즘	상용화 알고리즘	최적경로 알고리즘
검색 방법	방향성 제외	불완전한 교통정보	정확도 높은 교통정보
장 점	빠른 검색	이동시간감소	이동시간 최소
단 점	이동시간 파악 불가	검색결과 항상 동일	검색 시간 증가

IV. 결론

현재 우리나라에서는 각종 네비게이션 시스템에서 단지 거리상의 최단 경로 검색만을 지원하고 있으며, SK 등 일부 무선 통신 회사에서 제공하는 최적 경로 검색 또한 진정한 의미의 최적 경로가 아닌 단순한 교통 체증 지역, 사고 지역만을 주기적으로 입력하여 운영자의 개입아래 주관적으로 정해진 경로를 제공하고 있다.

그러나 일반적인 도로망 정보를 기반으로 계산된 가까운 경로일지라도 교통정보가 개입이 되면 그 경로가 항상 최선의 경로가 되는 것은 아니다. 예를 들어 최단경로를 검색 하여 나온 결과로 이동을 시작하였는데 검색 결과 구간 위에 교통사고가 발생하였으면 얼마나 많은 시간이 소요될지는 모르는 경우가 발생할 수도 있다. 따라서 이 논문에서는 교통 상황과 도로의 특성 등의 교통 정보를 기반으로 한 최적경로 탐색 알고리즘에 대하여 서술하였다.

구현 결과 제안한 알고리즘은 도로 교통 정보 중에 각 구간별 평균 속도와 각 교차로의 회전 허용 정보 및 U턴 정보를 감안하여 최적의 경로를 탐색함을 알 수 있었다.

이 논문의 알고리즘은 교통 상황의 빠른 변화에 능동적으로 대처할 수 있는 알고리즘이라고 판단되며, 각각의 클라이언트로부터의 교통 흐름 정보를 이용하여 일정 지역 내의 교통량 흐름을 판단할 수 있는 것도 가능하다.

앞으로는 업체에서 사용 중인 GPS가 내장된 텔레매틱스 단말기와 서버간의 CDMA 통신을 이용하여 영업용 차량으로부터 수집되는 교통 정보를 가공하는 방안도 연구하고자 한다. 그리고 회전에만 따른 지연[7]을 알고리즘에 반영하는 방안을 연구하고자 한다.

참고문헌

[1] William Stallings. "Data & Computer Communications, sixth Edition", Prentice Hall, Inc, 2001
 [2] <http://qnavi.bizemeka.com>
 [3] <http://drive.nate.com>
 [4] <http://inavi.co.kr>
 [5] <http://visionjoy.co.kr>
 [6]Stefano Pallottino, Maria Grazia Scutella, "Shortest Path Algorithms in Transportati on Models : Classical and Innovative Aspects"

TR, Univ. of Pisa, 1998

- [7] Stephan Winter, "Modeling Costs of Truns in Route Planning", GeoInformatica, Vol. 6, No. 4, pp. 345-360, 2002

저자소개

김진덕(Jin-Deog Kim)



1993년 부산대 컴퓨터공학과(공학사)
1995년 부산대 대학원 컴퓨터공학과(공학석사)
2000년 부산대 대학원 컴퓨터공학과(공학박사)

1998.3~2001.2 부산정보대학 정보통신계열 전임강사

2001.3~ 현재 동의대학교 컴퓨터공학과 조교수

※관심분야 : 객체 지향 DB, 지리정보시스템, 공간 질의, 공간 색인, 모바일 데이터베이스, 텔레매틱스



이현섭(Hyun-Seop Lee)

2004년 동의대 컴퓨터공학과(공학사)

2004년 동의대학교 대학원 컴퓨터공학과 석사과정

2003년 동의대 대학원 컴퓨터공학

과 박사 수료

이상욱(Sang-Wook Lee)



부산대학교 전자공학과 공학사
부경대학교 대학원 전자공학과(공학석사)

부경대학교 대학원 전자공학과(공학박사)

1995년 8월~ 현 경상대학교 정보통신공학과 교수, 해양산업연구소 연구원

※관심분야 : Computer Vision moving