

고정 그리드 기반 가변 휴리스틱을 이용한 최적경로탐색

이현섭 · 김진덕

동의대학교 컴퓨터공학과

Optimal Path Search using Variable Heuristic based on Fixed Grid

Hyoun-sup Lee · Jin-Deog Kim

Donguei University

E-mail : jdk@deu.ac.kr

요약

최적경로를 탐색하기 위해서는 출발지와 목적지간의 거리뿐만 아니라 탐색 되어지는 구간에 존재하는 많은 교통 상황들을 파악하고 이를 경로 탐색에 활용하여야 한다. 그러나 기존의 경로 탐색 알고리즘은 이러한 교통상황들을 적절히 이용하지 못하고 있다.

이 논문에서는 새로운 최적 경로 알고리즘을 제안한다. 알고리즘은 최적경로를 검색하기 위해 교통상황을 충분히 고려하고, 연산비용을 줄이기 위해 도로를 그리드형태로 나누어 각각의 평균 속도를 가지고 휴리스틱을 부여한다. 또한 알고리즘의 전체 수행시간, 노드 접근 횟수, 최적경로의 정확도를 항목으로 하는 실험을 수행하여 기존의 탐색 알고리즘인 Dijkstra 알고리즘과 A*알고리즘과의 성능평가를 실시하였다. 실험 결과 제안한 알고리즘이 타 알고리즘에 대해 좋은 성능을 보여주었다. 제안한 알고리즘은 향상된 응용을 지원하는 텔레매틱스 시스템에 유용하게 사용될 것으로 기대된다.

ABSTRACT

Optimal path search algorithm should consider traffic flows of the roads as well as the distance between a departure and destination. The existing path search algorithms, however, usually don't apply the continuously changed traffic flows.

In this paper, we propose a new optimal path search algorithm. The algorithm takes the current flows into consideration in order to reduce the cost to get destination. It decomposes the road networks into Fixed Grid to get variable heuristics. We also carry out the experiments with Dijkstra and A* algorithm in terms of the execution time, the number of node accesses and the accuracy of path. The results obtained from the experimental tests show the proposed algorithm outperforms the others. The algorithm is highly expected to be useful in a advanced telematics systems.

키워드

최적 경로, 텔레매틱스, 고정 그리드, 가변 휴리스틱

1.서론

도로위에서 정해진 목적지를 향하여 이동을 하기 위해서는 경로가 필요하다. 이러한 경로를 찾기 위해 여러 가지의 방법들이 존재한다. 이 방법들을 경로 탐색이라고 말한다. 과거에는 출발지에서 목적지 까지 단순히 이동 가능한 도로를 찾기 위한 경로 탐색이 존재 하였으나 현재에 이르러 단순히 경로만을 탐색하는 것을 벗어나서 어떻게 하면 최소한의 비용으로 가장 빠른 시간 내에 목적지를 찾아 갈수 있는가 하는 문제가 경로탐색에서의 핵심으로 떠오르고 있다. 여기서 말하는 비용을 줄이기 위해서는 두 가지

측면을 고려하여야 한다.

첫째, 탐색된 경로에 대하여 이동거리 및 이동시간이 감소해야한다. 두 가지 모두 최소에 가까운 값을 가지는 경로를 탐색하는 것이 중요하다. 이동거리는 짧은 경로지만 경로상의 교통흐름이 좋지 않다면 이동시간이 얼마나 늘어날지 모르는 위험을 안고 있으므로 이동시간이 비용감소 측면에서 볼 때 더 중요한 요소라고 할 수 있다 [4].

둘째, 시스템의 성능 측면에서 경로 탐색의 연산시간 및 탐색되는 노드의 수가 감소해야 한다. 서버에서 모든 연산이 이루어지는 경우 상당량의 클라이언트에서 요구하는 검색결과를 연산함에 있어서 연산시간과 노드수의 감소는 연산 비

용의 감소에 직접적인 영향을 준다. 반대로 시스템의 성능이 떨어지는 클라이언트에서 모든 연산이 이루어지는 경우에도 마찬가지로 검색 비용을 낮추는 것이 중요하다고 할 수 있다.

이 논문에서는 대표적인 경로 탐색 기법인 A* 알고리즘을 변형하여 최적경로를 탐색하는 시스템을 제안한다. 구체적으로 이동거리의 증가로 인한 검색범위의 증가를 효과적인 휴리스틱 가중치와 고정 그리드의 검색범위 분할을 적용하여 최적경로를 검색함과 동시에 연산범위를 최대한 줄여 빠른 시간 안에 검색할 수 있는 처리 방법을 제안한다.

이 논문의 구성은 다음과 같다. 제 2장에서는 기존의 경로 탐색 기법인 Dijkstra 알고리즘과 A*알고리즘의 장단점을 알아보고, 3장에서는 교통상황을 반영하는 고정그리드 기반 가변 휴리스틱을 적용한 최적경로 탐색시스템을 제안한다. 그리고 구현된 결과를 토대로 세 가지 알고리즘의 성능을 비교한다. 마지막으로 4장에서 결론을 맺는다.

II. 관련 연구

현재 경로 탐색을 위해 사용되고 있는 기법들로 여러 가지 탐색 알고리즘들이 사용되고 있으나 도로상에서 적용하는 최적경로 검색을 위한 알고리즘은 A*알고리즘[1]과 Dijkstra알고리즘[2]을 직접 사용하거나 이를 상황에 맞게 변형시켜 사용하는 경우가 가장 많다. 각각의 알고리즘 특징을 살펴보면 다음과 같다.

먼저 Dijkstra 알고리즘은 최단 경로 탐색을 위해 사용되는 알고리즘으로 출발지로부터 인접해있는 모든 노드들을 차례로 방문하여 목적지까지의 비용이 가장 작은 경로를 찾아내는 알고리즘이다.

이에 반해 A*알고리즘은 최단 거리 찾기에서 효과적으로 동작하는 알고리즘으로 Dijkstra 알고리즘이나 Best-First Search (BFS)[3] 알고리즘의 연산속도를 훨씬 앞지르고 있다. 그 이유는 기존의 그래프 탐색 알고리즘들과 비교하여 목표에 얼마나 근접하였는지를 평가하기 위해 휴리스틱 함수를 사용한다는 것이다. 휴리스틱에 의하여 목적지에 가장 근접한 방향으로 탐색을 하며, 그 방향이 실패하면 다른 경로를 찾게 된다. 이러한 과정을 통하여 연산 범위의 축소를 가져오므로 빠른 시간 내에 경로를 탐색 할 수 있다. 또한 휴리스틱 가중치는 고정적인 값이 부여되는 것이 A*알고리즘의 기본이다. 이 값으로 인하여 연산속도에 치명적인 약점이 생길수가 있다. 예를 들어, 현재 검색 대상이 되는 도로의 전체적인 교통흐름이 상당히 나쁜 상황인데 유독 한 구간에서 이동속도가 최대로 나오는 경우 휴리스틱 가중치는 이 구간의 속도에 맞추어 휴리스틱 가중치를 부여하게 된다. 따라서 전 구간

의 상황은 좋지 않은데 한 구간의 최대 속도 때문에 연산 범위는 증가하게 되는 상황이 발생한다.

III. 고정그리드 기반 가변 휴리스틱을 이용한 최적경로 탐색 시스템

3.1 최적경로 탐색 기법

연산속도와 노드 탐색 수를 줄이면서 최적경로를 찾아내기 위해서 본 논문에서는 가변적 휴리스틱[5]을 적용하는 방법을 제안한다. 노드간의 속도에 따라 휴리스틱 가중치를 다르게 주어 경로를 탐색하는 방법이다. 기본적인 알고리즘은 A*와 같지만 각각의 노드가 서로 다른 휴리스틱 가중치를 가지고 있다는 것이 차이점이다. 즉 탐색을 시작하면서 노드가 가지고 있는 휴리스틱 가중치의 정보를 이용하여 검색 범위의 감소를 가져온다.

따라서 가변적인 휴리스틱 가중치를 사용하면서 추가적인 연산을 최소화 하는 방법이 필요하다. 고정 그리드를 사용하는 방법은 이러한 문제점을 해결할 수 있다. 전체 탐색 범위를 일정한 크기의 그리드로 나누어 하나의 그리드안의 최대속도를 각각의 그리드에 휴리스틱 가중치로 사용하는 방법이다. 이 방법은 게임에서의 픽셀 단위 휴리스틱[6,7]을 확장한 방법이다. 그림 1은 A*알고리즘의 설명이다.

A*알고리즘은 각각의 노드 비용의 합이 가장 적은 경로를 탐색한다. 노드와 그리드가 동일

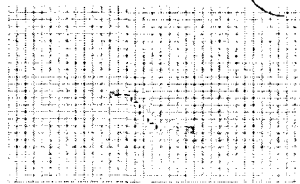


그림 1. A*알고리즘

논문에서 이야기하는 그리드 알고리즘과는 결정적인 차이가 존재한다. 일반적인 A*알고리즘에서는 탐색 경로 내에 존재하는 가장 큰 값을 휴리스틱 가중치로 주어 검색을 하기 때문에 현재 검색 범위의 전체 속도가 나쁘지만 하나의 노드가 속도가 빠른 경우 그 노드를 중심으로 경로를 검색하므로 검색 범위는 필요이상으로 증가하게 된다. 그리드 알고리즘은 이런 약점을 보완하여 각각의 그리드 범위 내에 존재하는 노드는 그 그리드가 가지고 있는 값을 휴리스틱 가중치로 두어 탐색을 한다. 따라서 최적경로의 최소 가능성이 있는 범위만을 대상으로 탐색을 진행하므로 연산속도와 노드 접근횟수는 일반적인 A*알고리즘에 비하여 성능이 좋다.

그림 2는 그리드를 적용하여 구현한 프로그램 결과이다. 구현은 Pentium PC에서 C# 프로그래밍 언어를 사용하였으며, 데이터로 사용된 도로

네트워크는 부산시 전체이며, 각 노드마다 좌회전, 우회전, U-턴 정보를 포함하고 있으며, 각 링크는 현재 시점의 평균 속도를 포함하고 있다.

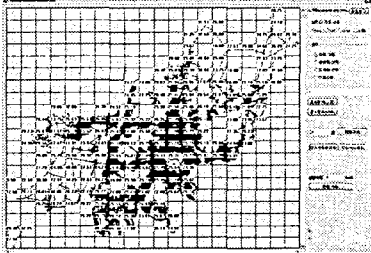


그림 2. 그리드기반 경로 탐색 프로그램

3.2 성능 평가

고정 그리드 A*알고리즘을 구현하여 다른 두 가지 알고리즘과 비교하였다. 모든 알고리즘은 동일한 조건에서 실행하였다. 100개의 경로를 임의로 설정하여 세 가지 알고리즘으로 탐색한 결과를 이어 나오는 표로 만들었다.

그림 3은 세 가지 알고리즘의 전체 케이스의 연산 수행시간 비교이다. 표에서 보면 Dijkstra 알고리즘과 A*알고리즘의 연산시간에 비하여 고정그리드 A*알고리즘의 연산시간이 상당히 향상되었다는 것을 알 수 있다.

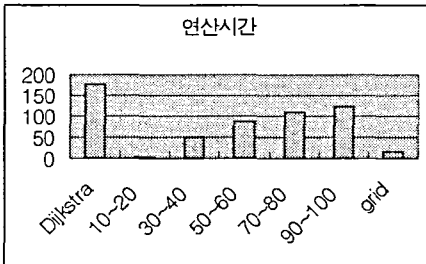


그림 3. 각 알고리즘의 연산 수행시간

특이할 점은 A*알고리즘에서 직진 성을 강하게 준 최대속도 10~20Km구간에서 가장 빠른 연산속도를 보이고 있다. 연산속도가 이렇게 빠른 이유는 주위에 다른 도로를 생각하지 않고 출발지에서 목적지 까지 일직선상의 도로만 검색 대상으로 여기고 진행하기 때문이다. 이런 경우 연산시간의 단축 효과는 있지만 최적경로의 정확성이 떨어진다. 이는 그림 5에 나타나 있다.

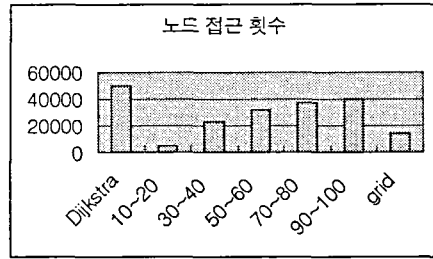


그림 4. 각 알고리즘의 노드 접근 횟수

그림 4는 세 가지 알고리즘의 탐색 노드 수의 비교이다. 여기서도 마찬가지로 고정그리드 A*알고리즘의 탐색 노드수가 두 가지 알고리즘에 비하여 상대적으로 적다는 것을 알 수 있다. 여기서도 A*알고리즘의 10~20Km구간이 가장 적은 수의 노드 탐색 횟수를 가지는데 이것은 위에서 설명한 것과 동일한 이유에서이다.

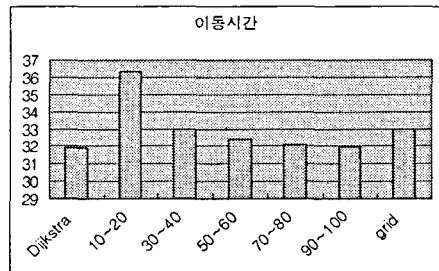


그림 5. 각 알고리즘의 최적경로 근접성

그림 5는 그리드 알고리즘과 A*알고리즘의 최고 속도 변화에 따른 탐색된 평균시간이다. A*알고리즘과 그리드알고리즘의 차이점을 볼 수 있는데 먼저 이동시간을 비교해 볼 때 근소한 차이로 A*알고리즘의 50~100Km구간이 최적경로에 그리드 알고리즘보다 가깝게 접근한다. 하지만 이 부분의 노드 탐색 속도와 접근횟수는 앞서 살펴본 표에서와 같이 그리드 알고리즘의 속도에 비하여 상당히 떨어지는 성능을 보인다. 10~20Km구간은 그리드 알고리즘과 A*알고리즘의 나머지 모든 구간보다 뛰어난 연산속도와 최저의 탐색 노드 횟수를 가진다. 그러나 표3의 결과를 보면 이동시간은 가장 크게 나오는 구간이기 때문에 이 구간이 최적경로라고 말할 수 없다. 그리고 현실적으로 도로에서 이동 가능한 속도는 정체가 빈번할 경우 시속 10~20Km에 가깝고 소통이 원활한 이동속도 또한 교차로 및 출발과 정지의 속도를 다 계산하면 30~60Km 사이라고 보는 것이 실제 교통상황에 가깝기 때문에 70~100Km 구간은 현실적으로 검색 대상에서 제외 될 수 있다고 사료된다.

그림 6은 A*알고리즘의 10~20Km 휴리스틱 구간과 그리드 알고리즘의 15Km 이하 구간과 초

과의 이동시간을 비교한 표이다. 10~20Km 휴리스틱의 연산 속도, 노드 접근 횟수의 성능이 그리드 알고리즘의 성능보다 떨어진다. 탐색거리가 15Km 이하로 짧을 때는 근소한 차이가 나타나는데 이동시간(최적경로)이 거리가 15Km를 초과하면서 그리드의 성능과 조금씩 큰 차이를 보이는 것으로 나타났다. 즉 거리가 멀어지면 멀어질수록 그리드 알고리즘이 최적경로를 탐색하는 데 유리한 것을 알 수 있다.

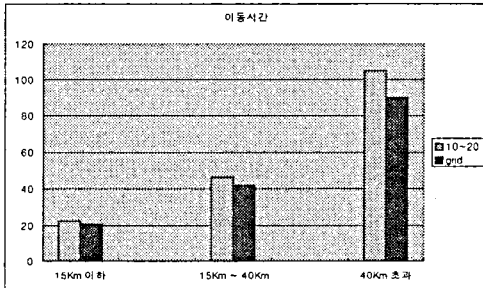


그림 6. 이동거리의 증가에 따른 그리드와 가변적 휴리스틱 10~20구간 비교

그리고 짧은 거리 구간에서도 그리드 알고리즘이 최적경로에 더욱 가깝게 탐색하는 경우들이 종종 발생하는 것을 확인 할 수 있었는데 이는 아마도 탐색범위의 교통상황이 나쁘면서 주위에 이동속도가 빠른 우회도로가 존재하였을 가능성을 보여주는 데이터이다.

Dijkstra 알고리즘의 결과 값을 100으로 봤을 때 그리드 알고리즘의 결과 값은 90% 가가운 근사 값을 이룬다. 모든 결과 값을 비교해보면 그리드 알고리즘은 연산 속도와 탐색노드 수는 상당히 줄이면서 최적경로에 가깝게 접근 하는 것을 확인 할 수 있다. 주목할 점은 탐색 거리가 늘어나거나 줄어들에 상관없이 그리드 알고리즘의 경로는 최적에 가까운 값이라는 것을 알 수 있다.

표 1. 알고리즘 최종 성능 비교

	Dijkstra	A*	그리드
최적경로	100	80~95	90
연산속도	50	70~100	95
탐색노드수	20	30~100	95
특징	근거리 탐색에 사용하는 것이 가장 효과적	구간 이동속도 변화가 없을 경우 유리	교통상황과 같이 가변적인 상황에 유리

표 1에서 나타나듯이 모든 결과를 토대로 볼 때 Dijkstra 알고리즘과 A* 알고리즘 그리고 그리드 알고리즘을 경로탐색에 구간별로 대입하여 사용하는 것이 가장 효과적인 경로 탐색이 될 수 있다는 결과를 얻을 수 있었다. 먼저 15Km미만의 구간에서는 Dijkstra 알고리즘 및 A* 알고리즘의 직진 성능 강하게 주어 탐색을 하거나 연

산속도의 큰 차이가 없으므로 Dijkstra 알고리즘을 사용하여도 무방하다. 그러나 15Km 이상의 구간에서는 그리드 알고리즘으로 탐색을 하여 탐색노드를 줄이고 빠른 연산시간으로 최적경로에 가까운 경로를 찾아내는 방법이 가장 최적의 방법이 될 수 있을 것이다. 1000번의 실험으로 모든 케이스를 구분했다고는 할 수 없으므로 좀더 세밀한 케이스를 두어 실험을 진행하여 최적의 값을 가질 수 있는 그리드의 범위를 산출해야 한다. 실험을 하면서 모든 데이터가 앞서 설명한 케이스 안에 들어가지는 않았다. 크게 차이가 나는 부분도 몇몇 존재하였고 반대의 결과가 나타나는 경우도 존재 하였으므로 더욱더 실제 교통상황과 근접하는 데이터를 가지고 실험을 진행 할 것이다.

IV. 결론

교통상황은 항상 가변적으로 바뀐다. 이런 상황들을 고려하여 경로 검색의 방법 또한 가변적으로 바뀌어 질 수밖에 없다. 많은 트래픽을 유발하는 도심의 교통 속에서 가장 빨리 도착할 수 있는 경로를 탐색하는 것은 경로 탐색 방법에서 가장 핵심적으로 다루어야 하는 요소이다. 아울러 시스템의 비용을 줄이기 위하여 경로를 탐색 하면서 연산 범위와 연산시간을 줄이는 것도한 충분히 고려를 해야 한다. 고정그리드와 가변적인 휴리스틱 가중치를 사용한 A* 알고리즘은 위에서 제시한 두 가지 조건을 가장 근접하게 충족시키는 알고리즘이라 할 수 있다. 이 논문에서 제안한 기법은 많은 교통상황들을 신속히 대입하여 연산이 가능하며 탐색 범위를 최소화 시켜 시스템의 부담을 덜어 줄 수 있다. 이 논문에서는 고정그리드를 적용한 A* 알고리즘에 대하여 서술하고 구현한 결과를 다른 탐색 알고리즘과 비교하였다.

구체적으로 세 가지 알고리즘을 구현하여 연산 시간과 노드 탐색 횟수를 비교하였고, 각각의 알고리즘들이 최적경로에 얼마나 가깝게 접근 하는지를 확인하였다. 앞으로는 정확한 데이터를 위하여 그리드의 크기를 변형하면서 가장 이상적인 그리드의 크기를 계산하고 최대한으로 최적경로에 접근하는 그리드의 범위 및 탐색 방법을 찾기 위하여 여러 가지의 교통상황을 만들어서 많은 데이터를 가지고 입력하여 좀 더 세밀한 결과를 확보하여 산출할 것이다.

참고문헌

[1] en.wikipedia.org/wiki/A%2A_search_algorithm
 [2] en.wikipedia.org/wiki/Dijkstra%27s_algorithm
 [3] en.wikipedia.org/wiki/Breadth-first_search
 [4] Stefano Pallottino, Maria Grazia Scutella, "Shortest Path Algorithms in Transportation Models : Classical and Innovative Aspects", TR,

Univ. of Pisa, 1998

- [5] 이현섭, 안준환, 김진덕. "가변적 휴리스틱을 이용한 최적 경로 탐색", 해양정보통신학회 춘계학술대회, 2005
- [6] theory.stanford.edu/~amitp/GameProgramming
- [7] Woo Young Kwon; Sanghoon Lee; Il Hong Suh, "A reinforcement learning approach involving a shortest path finding algorithm" Intelligent Robots and Systems, Proceedings of IEEE/RSJ International Conference, 2003