Journal of Korean Institute of Intelligent Systems, Vol. 22, No. 2, April 2012, pp. 161-167
http://dx.doi.org/10.5391/JKIIS.2012.22.2.161

무인 차량의 도로주행을 위한 경유점 생성 방법

A Via Point Generation Method for Road Navigation of Unmanned Vehicles

최혁두*, 박남훈**, 김종희***, 박용운***, 김은태*†

Hyukdoo Choi, Nam Hun Park, Jong Hui Kim, Yong Woon Park, Euntai Kim†

*연세대학교 전기전자공학부

**안양대학교 컴퓨터학과

***국방과학연구소

요 약

본 연구는 무인 차량의 도로에서의 자율주행을 위한 경유점을 지정하는 방법에 관한 것이다. 무인 차량이 현재 위치와 목표 위치의 전역 좌표를 알고 주어진 도로 지도 상에서 경로 계획을 한다면 어떤 도로의 어떤 차선을 거쳐 가야 하는지 미리 계획하여야 한다. 또한 이를 위해서는 도로의 위치 정보가 미리 체계적으로 저장되어 데이터베이스화 되어 있어야 한다. 본 논문에서는 도로를 데이터베이스로 저장하는 방법과 그 데이터베이스를 이용하여 최단거리 경로를 계획하고 이동방향과 차선을 고려하여 차량이 지나가야 할 상세한 경유점을 생성한다. 이후 시뮬레이션을 통해 제안한 알고리즘으로 도로에 최적화된 경로를 찾을 수 있음을 검증하였다.

키워드 : 경로 계획, 경유점 생성, 도로 데이터베이스

Abstract

This research deals with generating via points for autonomous navigation on a roadway for unmanned vehicles. When a vehicle plans a path from a starting point to a goal point, it should be able to map out which lane on which road it passes by. For this purpose, we should organize positional information of roads and save it as a database. This paper presents methods to save the database and to plan a shortest path to the goal by generating via points in consideration of the moving direction and the lane directions. Then we prove that the proposed algorithm can find the optimal path on the road through simulations.

Key Words : Path plan, Via points, Road database

1. 서 론

무인 차량의 자율주행 기술은 산업적, 군사적으로 중요한 의미를 가지고 있다. 그러나 기술의 특성상 다양한 환경에서 높은 수준의 정확도를 요구하기 때문에 아직 완성된 기술이 없어 현재 이와 관련한 많은 연구들이 진행되고 있다 [1-2]. 자율주행에 필요한 많은 기술 구성 요소 중 경로 계획은 자율주행의 핵심요소로서 크게 지역 경로 계획과 전역 경로 계획으로 나뉜다. 지역 경로 계획은 주로 차량에 설치된 센서로 주변 환경의 위험을 파악하여 장애물 등을 피해 안전하게 목적지까지 가는 기술을 말한다 [3]. 이에 반해 전역 경로 계획은 지도상에서 출발점과 목표점을 설정했을 때 도로와 지도에 표기된 장애물 등을 고려하여 미리 전체적인 경로를 설계하는 것을 말한다 [4].

이전 연구들은 주로 정적, 동적 장애물을 부드럽게 피해가는 지역 경로 계획에 집중되어 있었다 [5][6][7]. 전역 경로 계획 연구들도 지역 경로 계획의 연장선에서 트리 구조를 이용하여 지역 경로를 더 멀리 내다보고 계획하는 연구가 많았다 [8]. 본 논문에서는 단순한 장애물 지도가 아닌 구조화된 도로지도에서 전역 경로를 계획하여 차량이 지나가야 할 경유점들을 설정하는 것을 목표로 한다. 이를 위해서는 도로 지도가 경로 계획 모듈이 활용할 수 있는 형태로 미리 데이터베이스화 되어있어야 한다. 도로 데이터베이스는 크게 도로를 직선 단위로 구분하여 위치와 모양을 저장한 다각형(Polygon)과 차선 경계선을 나타낸 차선 경계선(LaneBorder)로 구성된다. 이를 통해 경로 계획 모듈이 도로 구조를 파악하여 출발점에서 목적지까지 가는 최단 경로를 연속된 다각형으로 계획하고 각 다각형 상에서 이동방향에 맞는 차선을 선택하여 구체적인 경유점

접수일자: 2011년 12월 22일
심사(수정)일자: 2012년 3월 21일
게재확정일자 : 2012년 3월 23일
†교신저자
본 연구는 국방과학연구소 과제 “자율주행용 주행정보 파일의 실시간 관리 시스템 연구”의 지원을 받았습니다.

들을 설정한다.

본 논문의 기여는 크게 두 가지이다. 도로 정보를 체계적으로 저장하여 데이터베이스화 한 것과 이를 이용해 차선 단위까지 구체적으로 이동 경로를 설정한 것이다. 이를 통해 대규모 도로에서 안정성, 적법성, 효율성을 고려한 경로상의 경유점들을 지정할 수 있다.

제안된 데이터베이스와 경로 계획 알고리즘은 시뮬레이션을 통해 성능을 검증하였다. 가상의 도로 환경을 만들고 이를 직접 데이터베이스로 저장한 뒤 경로 계획 알고리즘을 적용하여 의도한 목적을 달성함을 확인할 수 있었다.

2. 도로 데이터베이스

도로 정보가 경로 계획에 활용되기 위해서는 도로와 차선의 모든 경계면이 표현될 수 있을 정도로 자세한 좌표들이 저장되어 있어야 한다. 도로의 형태는 크게 직선 구간, 곡선 구간, 교차로로 나눌 수 있으며 이들을 모두 하나의 형식으로 데이터베이스에 담을 수 있어야 한다. 경로 계획에 필요한 간단한 지리정보시스템(Geographic Information System)을 구현하기 위해 본 연구에서는 지리정보시스템의 세계 표준 기관인 Open Geospatial Consortium (OGC)를 참고하였다 [9]. 도로는 직선단위의 다각형(Polygon)으로 구성되며 각 다각형에서의 차선은 차선경계선(LaneBorder)으로 표현한다. 모든 다각형은 볼록 모양(Convex hull)을 가져야 한다. 그 이유는 경로를 계획할 때 각 다각형에 들어가는 진입점과 나가는 출구점을 설정하는데 임의의 진입점과 출구점을 잇는 선분이 다각형(도로)를 벗어나선 안 되기 때문이다. 이상의 기하학적 위치와 모양을 저장하기 위한 좌표(Coordinate)들도 또한 중복을 피하기 위해 따로 저장되어야 한다. 이러한 데이터베이스 구조를 나타낸 것이 그림 1이다.

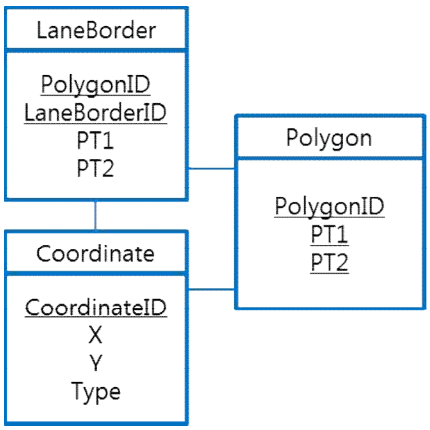

그림 1. 도로 데이터베이스 구조.

Fig. 1. A structure of a road database.

다각형은 여러 개의 선분으로 구성되며 하나의 다각형 레코드는 다각형을 이루는 하나의 선분을 나타낸다. PolygonID가 같은 레코드들이 여러 개 모여 하나의 다각형을 이룬다. 각 다각형 내부에서 차선 경계선은 차선사이에 그어진 도로선을 나타낸다. 각 차선 경계선은 양 끝점으로 표현된 하나의 선분을 표시한다. 그림 2의 예시에는 두 개의 다각형과 여섯 개의 차선 경계선이 나타나 있다. 파란 실선은 다각형의 외곽선을 나타내고 검은 점선은 차선 경계선을 나타낸다.

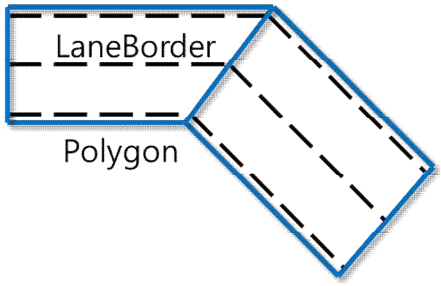

그림 2. 다각형과 차선 경계선의 구조 예시.

Fig. 2. An example of structures of a polygon and a lane border.

다각형과 차선 경계선 테이블에서 PT1, PT2는 선분의 양 끝점의 좌표번호를 나타내고 이것은 좌표 테이블의 CoordinateID와 연결된다. 해당 CoordinateID에 붙어있는 X, Y 좌표가 선분의 양 끝점의 좌표를 나타낸다. 좌표 테이블의 Type은 각 좌표가 다각형의 좌표인지 차선 경계선의 좌표인지를 구분하기 위해 사용된다.

다음은 간단한 예시를 통해 도로 데이터의 저장 방법을 설명한다. 그림 3과 같은 하나의 다각형에 2차선 도로가 있다고 가정하자. 원 안의 숫자는 선분의 끝점 좌표의 CoordinateID를 나타낸다. 다각형의 PolygonID는 1, 2, 3 등의 자연수로 입력순서대로 늘어나고 중앙선의 LaneBorderID는 언제나 0이다. 중앙선의 한쪽의 차선 경계선들을 중앙선에 가까운 것부터 +1, +2 순으로 지정하면 반대쪽은 -1, -2 순으로 지정한다. 차선은 직접 저장되어 있지는 않지만 차선 번호는 차선을 구분하는 두 경계선 중 바깥쪽 차선 경계선의 번호를 따른다. 예를 들어, 차선 -2는 차선 경계선 -1과 -2 사이에 있다. 또한 중앙선이 없는 1차선 도로의 경우엔 차선 번호를 0으로 한다.

도로 예시인 그림 3의 도로를 저장한 다각형 테이블과 차선 경계선 테이블은 표 1과 표 2와 같다. 표 1에서 하나의 다각형은 네 개의 레코드로 저장되고 차선 경계선은 각각 하나의 레코드로 저장된다. 이와 같이 저장된 도로 정보는 다음 장에서 경로 계획 알고리즘에 사용된다.

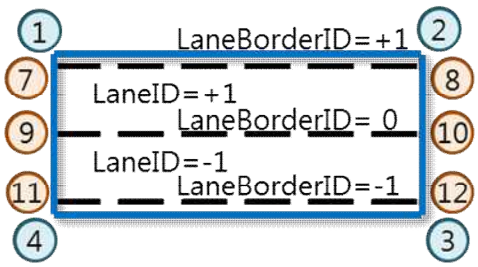

그림 3. 도로 데이터베이스 예시.

Fig. 3. An example of a road database.

표 1. 다각형 테이블 예시.
Table 1. An example of a polygon table.

PolygonID	PT1	PT2
1	1	2
1	2	3
1	3	4
1	4	1

표 2. 차선 경계선 테이블 예시.
Table 2. An example of a lane border table.

PolygonID	LaneBorderID	PT1	PT2
1	+1	7	8
1	0	9	10
1	-1	11	12

3. 경로 계획을 통한 경유점 생성 방법

본 장에서는 저장된 도로 데이터베이스를 이용하여 지정된 출발점에서 목표점으로 가는 최단 경로를 계획하고, 도로의 차선 수와 전후 연결 관계를 고려하여 무인 차량이 지나가야 할 경유점을 구체적인 좌표로 지정하는 방법을 제시한다. 최종적으로 경유점을 생성하기까지 Coarse to fine 기법을 사용하여 총 네 단계의 과정을 거친다. 먼저 도로 구조에서 지나가야 할 다각형들을 순서대로 찾고, 각 다각형에서 이동 방향에 따라 지나야 할 차선의 부호를 찾고, 각 다각형에서 진입 차선과 출구 차선을 정하고, 그 다음 각 다각형의 출구점과 진입점을 찾아 전체적인 경로의 경유점을 완성한다. 각 단계에 대한 상세 내용은 아래와 같다.

3.1 다각형 단위 경로 계획

볼록 모양의 다각형들의 집합으로 구성된 도로에서 경로를 계획하기 위해서는 먼저 출발점과 목표점 사이의 최단 경로를 구성하는 연속된 다각형들의 배열을 구해야 한다. 이를 위해 본 연구에서는 도로를 그래프 형태로 변형하여 다익스트라(Dijkstra)의 알고리즘을 적용한다 [10]. 각 다각형을 그래프의 정점(node)으로 설정하고 연결된 다각형의 정점 사이를 간선(edge)로 연결한다. 각 정점에는 해당 다각형의 중심점이 저장되고 각 간선에는 연결된 정점들 사이의 거리가 저장된다. 그림 4는 다각형을 그래프로 변환한 예시이다. 2장에서 저장된 도로 데이터베이스에서 연결된 다각형들은 한 면을 공유하므로 같은 끝점들을 가진 한 선분을 공유하는 두 다각형 사이에 간선을 연결한다.

다익스트라 알고리즘을 재귀적으로 구현한 의사코드(pseudo code)는 표 3과 같다. 이 알고리즘을 실행하기 위해서는 먼저 출발 정점 S와 목표 정점 G를 알아야 하고, Dist와 Pidx의 모든 엔트리를 -1로 초기화하고, Link에 연결된 모든 정점 사이의 거리를 저장해야 한다. Link(i,j)는 정점 i로부터 정점 j까지의 거리이다. 연결되지 않은 관계에는 0이 저장된다. 이 후 P=-1, C=S로 설정하여 알고리즘을 실행하면 Pidx에 각 정점으로 오는 최단경로상의 직전 정점들이 저장되므로 이를 목표 정점부터 출발 정점이 나올 때까지 따라가면 정점들로 이루어진 경로를 구할 수 있다. 또한 경로상의 정점들의 좌표들을 연결하면 개략적인 경로의 형태를 볼 수 있다.

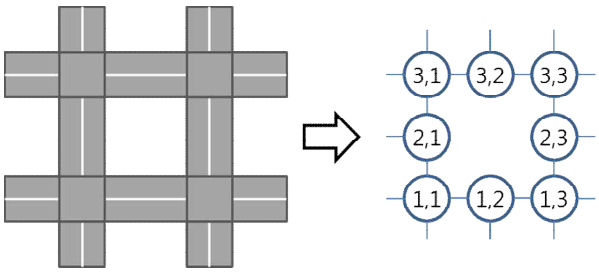

그림 4. 도로의 다각형들을 그래프로 변환한 예시.
Fig. 4. An example of a transformation of polygons to a graph.

3.2 차선 부호 찾기

앞 절에서 나온 결과로 출발점과 목표점 사이에 지나가야 할 최단 경로 위에 존재하는 다각형의 배열을 얻을 수 있다. 하지만 무인 로봇의 안전하고 적법한 주행을 위해서는 차선까지 고려하여 보다 세밀한 계획이 필요하다. 이를 위해서는 양방향 차선들 중 어느 쪽 차선으로 가야하는지를 정해야 한다. 본 연구에서 저장된 도로 데이터베이스를 기준으로는, 중앙선을 기준으로 양의 방향인지 음의 방향인지를 결정해야 한다. 단, 단일 차선 도로는 이를 고려할 필요가 없다. 차량은 진행 방향을 기준으로 중앙선의 오른쪽 차선으로 이동해야 한다. 이를 수학적으로 정리하기 위한 변수들을 정리하면, 진행방향을 기준으로 중앙선의 양 끝점 중 가까운 점을 P1, 먼 점을 P2라 하고 임의의 양의 차선 경계선의 좌표를 P+, 임의의 음의 차선 경계선의 좌표를 P-이라 한다. 그리고 양의 차선의 상대 각도 θ_+와 음의 차선의 상대 각도 θ_-는 그림 5와 같이 정의된다.

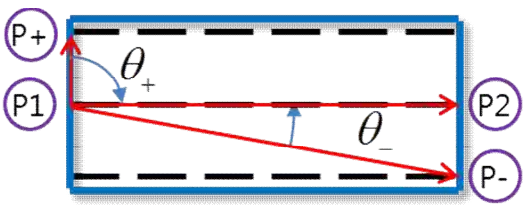

그림 5. 양의 차선과 음의 차선의 상대 각도 정의.
Fig. 5. A definition of relative angles of a positive lane and a negative lane.

표 3. 다익스트라 알고리즘의 의사코드.
Table 3. A pseudo code of the Dijkstra algorithm.

1:	**Dijkstra**(S, P, C, G, Dist, Link, Pidx) S : 출발 정점, P : 이전 정점, C : 현재 정점, G : 목표 정점, Dist[i] : 출발 정점으로부터의 정점

```
    i까지의 거리, Link[i][j] : 연결된 정점 i, j
    사이의 거리, Pidx[i] : 정점 i로 오는
    최단경로상의 직전 정점
2:  if(P<0)      // 이 함수에 처음 들어왔을 때
3:      path_dist = 0;
4:  else
5:      // 새 경로의 거리
        path_dist = Dist[P] + Link[P][C]
6:  endif
7:  // 새 경로의 거리가 더 짧을 때
    if(path_dist<Dist[C] or dist[C]<0)
8:      Dist[C]=path_dist  // 거리 갱신
9:      Pidx[C]=P          // 경로 갱신
10:     if(C==G) return
11: else  return // 경로를 갱신하지 못했다면 중지
12: endif
13: for i=1,...,N  // 다음 경로 탐색
14:     // 이전 정점로 돌아가는 길은 보지 않는다.
        if(i==S or i==P)    continue
15:     // C가 i와 연결됐다면 경로를 진행한다.
        if(Link[C][i]>0)
16:         Dijkstra(S, C, i, G, Dist, Link, Pidx)
17:     endif
18: endfor
```

$\theta_+ > 0$ 이고 $\theta_- < 0$ 라면 차량은 양의 차선으로 주행해야 하고 반대로 $\theta_+ < 0$ 이고 $\theta_- > 0$ 라면 차량은 음의 차선으로 주행해야 한다. 그림 5에서는 차량은 음의 차선으로 주행해야 한다. 최단 경로를 이루는 모든 다각형 도로 조각에서 위와 같은 검사를 시행하여 각 다각형에서 지나야 할 차선 부호를 정하여 LaneSign이라는 배열에 저장한다. 양의 차선은 +1, 음의 차선은 -1, 양방향 1차선은 0으로 저장하고 교차로에서는 JUNC이라는 임의의 상수로 저장한다.

3.3 정확한 차선 지정

도로에서 한 방향의 차선이 여러 개인 경우 도로의 전후 연결 관계에 따라 도로에 들어가는 차선과 나가는 차선을 정해야 한다. 다차선 도로에서는 교차로에서 우회전하는 경우 바깥 차선으로 나가서 다음 도로의 바깥 차선으로 들어가고, 직진은 가운데 차선으로 나가서 다음 도로의 같은 차선으로 들어가고, 좌회전은 안쪽 차선으로 나가서 다음 도로의 안쪽 차선으로 들어가는 것이 안전하다. 이를 위해서는 먼저 교차로에서 연결된 다각형들을 분석하여 차량이 우회전, 좌회전, 직진 중 어느 행동을 하는지를 알아내야 한다. 다음은 이를 알고리즘으로 정리한 것이다.

- 차선이 하나인 경우는 차선 번호가 LaneSign과 같다.
- 차선이 둘 이상이면 진입 차선을 결정한다.
  1. 이전 다각형이 교차로인 경우 현재와 이전 두 다각형의 연결 관계를 고려하여 교차로에서의 진행 방향을 결정한다.
  2. 진입 차선은 우회전이면 바깥 차선, 직진이면 가운데 차선, 좌회전이면 안쪽 차선으로 정한다.
  3. 이전 다각형이 교차로가 아닌 경우 이전 다각형의 출구 차선을 현재 다각형의 진입 차선으로 한다.
- 차선이 둘 이상이면 출구 차선을 결정한다.
  1. 다음 다각형이 교차로인 경우 현재와 다음 두 다각형의 연결 관계를 고려하여 교차로에서의 진행 방향을 결정한다.
  2. 출구 차선은 우회전이면 바깥 차선, 직진이면 가운데 차선, 좌회전이면 안쪽 차선으로 정한다.
  3. 다음 다각형이 교차로가 아닌 경우 다음 다각형의 진입 차선을 현재 다각형의 출구 차선으로 한다.

3.4 경로 상의 경유점 생성

이상의 알고리즘으로 출발점에서 목표점으로 가는 경로상에 어떤 다각형들이 있는지, 각 다각형에서는 어떤 차선으로 들어가서 어떤 차선으로 나가야 하는지를 결정하였다. 이를 무인 차량이 활용하기 위해서는 각 다각형에서의 진입점과 출구점을 명확히 지정하여 주는 것이 좋다. 각 다각형마다 진입점과 출구점이 있지만 경로 상의 한 다각형의 출구점과 그 다음 다각형의 진입점이 같으므로 이들의 중복을 피해야 한다. 경로상의 지나가야 할 주요 좌표들을 저장한 배열은 표 4에 정리된 알고리즘으로 구할 수 있다. 표 4에서 진입점은 진입 차선 양쪽 차선 경계선들의 시작점의 중간점으로 구하고, 출구점은 출구 차선 양쪽 차선 경계선들의 끝점의 중간점으로 구한다. 이 경유점들을 잇는 선분들이 무인 차량이 대략적으로 따라야 할 경로이다. 여기서 대략적이라 함은, 일방 다차선 도로에서 진입 차선과 출구 차선이 다를 때 언제 차선 변경을 해야 하는지는 그때의 도로 상황에 따라 달라지므로 이를 미리 계획하지 못 하기 때문에, 경로를 각 다각형의 진입점과 출구점을 잇는 선분으로 밖에 표현하지 못 함을 나타낸다. 진입차선과 출구차선이 다를 경우 차선변경은 차선과 안전을 고려한 지역 경로 계획 알고리즘을 사용하여 실행한다.

4. 시뮬레이션 결과

표 4. 경유점 지정 알고리즘.
Table 4. A viapoint setting algorithm.

```
1:  SetViaPoints(LaneSign, LaneIn, LaneOut, Spt,
    Ept)
    LaneSign[i] : 다각형 i의 차선 부호, LaneIn[i] :
```

```
      다각형 i에 들어가는 차선, LaneOut[i] : 다각형
      i에서 나가는 차선, Spt[i]k : 다각형 i의 차선
      경계선 k의 시작점 좌표, Ept[i]k : 다각형 i의
      차선 경계선 k의 끝점 좌표
2:    Wp[1]=S          // 첫 좌표를 출발점
3:    m=2              // 경유점 개수
4:    for i=1,...,N    // 다각형 i에서 경유점 지정
5:      b_in=0, b_out=0
6:      // 교차로에서는 경유점을 정하지 않는다.
        if(LaneIn[i]==JUNC)   continue
7:      // 마지막 다각형에서는 진입점만 계산
        elseif(i==N)   b_in=1
8:      // 이전 다각형이 교차로면 둘 다 계산
        elseif(LaneIn[i-1]==JUNC)   b_in=1, b_out=1
9:      // 나머지 경우 출구점만 계산
        else   b_out=1
10:     endif
11:     if(b_in==1)   // 진입점 계산
12:       if(LaneSign[i]==0)
13:         Wp[m] = (Spt[i]-1+Spt[i]+1)/2
14:       elseif(LaneSign[i]>0)
15:         Wp[m]=Spt[i]LaneIn[i]+Spt[i]LaneIn[i]-1)/2
16:       else
17:         Wp[m]=Spt[i]LaneIn[i]+Spt[i]LaneIn[i]+1)/2
18:       endif
19:       m=m+1
20:     elseif(b_out==1)   // 출구점 계산
21:       if(LaneSign[i]==0)
22:         Wp[m] = (Ept[i]-1+Ept[i]+1)/2
23:       elseif(LaneSign[i]>0)
24:         Wp[m]=Ept[i]LaneOut[i]+Ept[i]LaneOut[i]-1)/2
25:       else
26:         Wp[m]=Ept[i]LaneOut[i]+Ept[i]LaneOut[i]+1)/2
27:       endif
28:       m=m+1
29:     endif
30:   endfor
31:   Wp[m]=G   // 마지막 좌표는 목표점
32:   return Wp
```

제안된 도로 데이터베이스와 경유점 생성 알고리즘을 검증하기 위해 가상의 도로에서 도로 데이터베이스를 구축하고 경유점을 생성하는 시뮬레이션을 수행한다.

서로 다른 성격을 가진 두 가지 도로를 설계하여 다양한 상황에서 경유점 생성 알고리즘을 검증하였다. 첫 번째 도로 환경에서는 2차선, 혹은 4차선의 도로를 복잡하게 연결하여 출발점에서 목표점까지의 경로 생성 능력을 검증하고자 한다. 첫 번째 환경에서 다각형은 22개가 있으며, 차선 경계선은 86개가 저장되어 있다. 3장에서 제안한 네 단계의 알고리즘들이 각각 정확히 작동하는지 확인하기 위해 각 단계별로 경로가 생성되는 과정을 그림 6, 7, 8을 통해 보여준다. 그림 6은 3.1의 다각형 단위 경로 계획 결과를 보여주고, 그림 7은 3.2에서 설명한 경로상의 다각형에서 차량이 주행할 방향에 따라 차선을 구별하여 순차선으로 경로를 계획한 결과이며, 그림 8는 차선의 방향뿐만 아니라 도로의 전후 연결 관계까지 고려하여 차량이 지나야 할 정확한 차선을 지정하여 경유점을 생성한 결과이다.

그림 6에서는 각 최단 경로상의 다각형의 중심점만을 이어놓았기 때문에 일부 구간에서는 도로를 벗어나기도 한다. 그러나 그림 7에서는 순차선과 역차선을 구별하여 순차선 중 중앙선에 가까운 차선으로만 지나가도록 한다. 그림 8에서는 다차선 도로의 교차로에서 이동 방향에 따라 다른 차선을 선택하는 모습을 보여준다. 경로에서 첫 번째 우회전과 두 번째 우회전에서 그림 7과 8에 차이가 있음을 볼 수 있다. 그림 9와 10은 첫 번째 환경에서 다른 출발점과 목표점을 설정하여 경로를 계획한 결과이다.

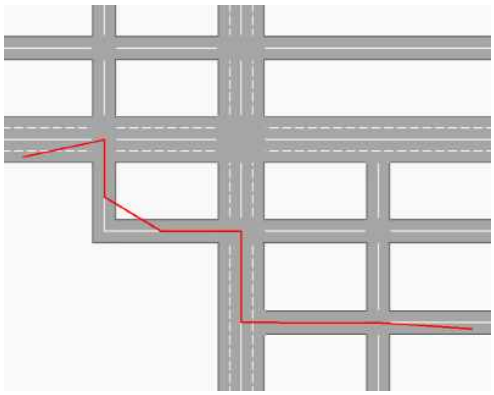

그림 6. 다각형 단위 경로 계획 결과.

Fig. 6. A result of path planning based on polygons.

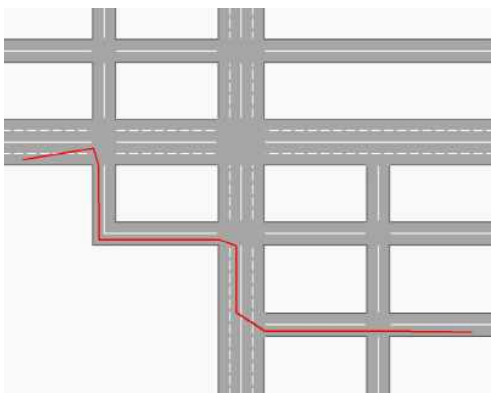

그림 7. 차선 구분 결과.

Fig. 7. A result of discrimination of lanes.

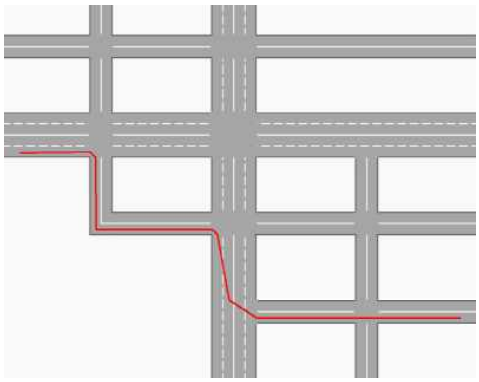

그림 8. 최종 경로 생성 결과.
Fig. 8. A result of final path planning.

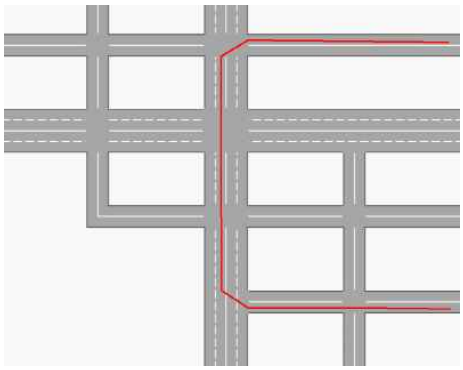

그림 9. 첫 번째 환경에서의 첫 번째 경로 계획 결과.
Fig. 9. The first path planning result in the first environment.

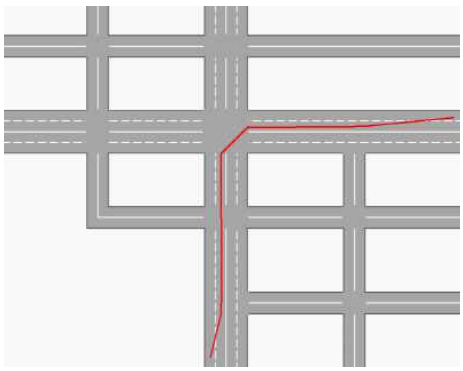

그림 10. 첫 번째 환경에서의 두 번째 경로 계획 결과.
Fig. 10. The second path planning result in the first environment.

두 번째 환경에서는 도로를 좀 더 단순화 하는 대신 도로의 차선 수를 다양화 하여 각 다각형의 진입 차선과 출구 차선이 교차로에서의 이동 방향에 따라 달라지는 모습을 보여주고자 한다. 그림 11과 그림 12는 두 번째 환경에서 서로 다른 경로를 계획한 결과이다. 두 번째 환경에서는 6차선 도로가 3개, 2차선이 1개, 1차선이 1개 있다. 6차선 도로의 교차로에서 경로가 좌회전일 때는 안쪽 차선을, 우회전일 때는 바깥 차선을, 직진일 때는 가운데 차선을 통하여 생성됨을 볼 수 있다. 다만 앞서 말했듯이, 경로는 각 다각형의 진입점과 출구점만 고려하여 만들어지기 때문에 중간에 어디서 어떻게 차선 변경을 해야 하는지는 경로 상에 나타나 있지 않다. 이는 지역 경로 계획에 속하는 것으로서 본 연구의 영역인 전역 경로 계획에서는 고려의 대상이 아니다.

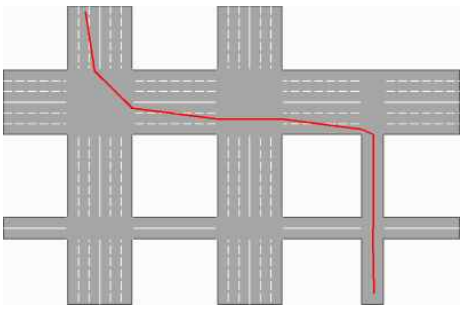

그림 11. 두 번째 환경에서의 첫 번째 경로 계획 결과.
Fig. 11. The first path planning result in the second environment.

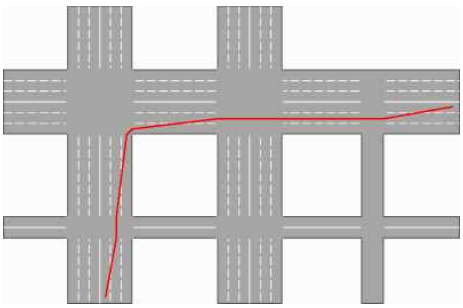

그림 12. 두 번째 환경에서의 두 번째 경로 계획 결과.
Fig. 12. The second path planning result in the second environment.

그림 9~12에 나타난 시뮬레이션 결과를 종합하였을 때 본 연구의 경로 생성 알고리즘이 차량이 지나가야 할 정확한 경유점들을 생성할 수 있음을 증명하였다.

5. 결론

본 연구에서는 도로에서 무인 차량의 자율주행을 위하여 도로를 데이터베이스화 하고 이를 이용해 출발점으로부터 목표점까지 이르는 경로를 계획하여 이를 경유점 형태로 출력하는 시스템을 개발하였다. 기존의 위험 회피를 위한 지역 경로 계획이나 큰 스케일의 도로 단위 경로 계획에서 벗어나 무인 차량이 실질적으로 이용할 수 있는 중간 좌표들을 제공하고자 하였다. 그리하여 다각형과 차선 경계선과 좌표 테이블로만 이루어진 간단한 데이터베이스를 이용하여 네 단계에 걸친 경로 계획 알고리즘으로 도로의 차선과 연결 관계까지 고려한 경유점 생성 알고리즘을 제안하였다. 두 가지 도로 환경에서의 시뮬레이션을 통해서 제안한 경유점 생성 방법이 빠르고 안전한 경로를 생성함을 확인 할 수 있었다.

앞으로의 연구에서 이를 지역 경로 계획과 결합한다면 실제 차량이나 로봇을 이용하여 실제 환경에서 자율주행을 할 수 있을 것이다.

참 고 문 헌

[1] N. K. Yilmaz, C. Evangelinos, P. F. J. Lermusiaux, and N. A. Patrikalakis, "Path Planning of Autonomous Underwater Vehicles for Adaptive Sampling Using Mixed Integer Linear Programming," *IEEE Journal of Oceanic Engineering*, vol. 33, pp. 522-537, Oct. 2008.

[2] S. Bhattacharya, R. Murrieta-Cid, and S. Hutchinson, "Optimal paths for landmark-based navigation by differential-drive vehicles with field-of-view constraints," *IEEE Transactions on Robotics*, vol. 23, pp. 47-59, Feb. 2007.

[3] Y. Kim, D. W. Gu, and I. Postlethwaite, "Real-time path planning with limited information for autonomous unmanned air vehicles," *Automatica*, vol. 44, pp. 696-712, Mar. 2008.

[4] T. M. Howard and A. Kelly, "Optimal rough terrain trajectory generation for wheeled mobile robots," *International Journal of Robotics Research*, vol. 26, pp. 141-166, Feb. 2007.

[5] 정경훈, 김정민, 전태룡, 김성신, 김광백, "퍼지 추론 시스템을 이용한 다중경로계획의 충돌회피 방법," *한국지능시스템학회 2009년도 춘계학술대회 학술발표논문집*, pp. 11-14, 2009.

[6] 이기성, 조현철, "유전 알고리즘을 이용한 자율 주행 로봇의 장애물 회피," *한국지능시스템학회 논문지*, vol. 8, pp. 27-35, 1998.

[7] 허정민, 김정민, 정승영, 김성신, and 김광백, "유전자 알고리즘, 퍼지 룰을 이용한 다중 경로 계획," *한국지능시스템학회 2008년도 춘계학술대회 학술발표논문집*, pp. 60-63, 2008.

[8] M. Barbehenn and S. Hutchinson, "EFFICIENT SEARCH AND HIERARCHICAL MOTION PLANNING BY DYNAMICALLY MAINTAINING SINGLE-SOURCE SHORTEST PATHS TREES," *IEEE Transactions on Robotics and Automation*, vol. 11, pp. 198-214, Apr. 1995.

[9] http://www.opengeospatial.org/

[10] E. W. Dijkstra, "A Note on Two Problems in Connexion with Graphs," *Numerische Mathematik*, pp. 269-271, 1959.

저 자 소 개

최혁두(Hyukdoo Choi)

2009년 : 연세대학교 전기전자공학부 졸업
2009년～현재 : 동 대학원 전기전자공학과 통합과정
관심분야 : 로보틱스, 인공지능, SLAM
Phone : 02-2123-7729
Fax : 02-313-2879
E-mail : goodgodgd@yonsei.ac.kr

박남훈(Nam Hun Park)

2002년 : 연세대학교 대학원 컴퓨터과학과 공학석사
2007년 : 연세대학교 대학원 컴퓨터과학과 공학박사
2007년~2008년 : 연세대학교 데이터베이스 국가지정 연구실 연구원
2008년~2010년 : Worcester
Phone : 032-930-6021
Fax : 032-930-6215
E-mail : nmhnpark@anyang.ac.kr

김종희(Chong Hui Kim)

2007년 : 한국과학기술원 전자전산학부 전기및전자공학전공 박사
2008년～현재 : 국방과학연구소 선임연구원
관심분야 : 최적제어, 자율주행
Phone : 042-821-2637
Fax : 042-821-3400
E-mail : chonghui.chkim@gmail.com

박용운(Yong Woon Park)

1994년 : University of Utah 기계공학과 박사
1982년～현재 : 국방과학연구소 책임연구원 무인자율화기술 부장
관심분야 : 자율주행, SLAM, 차량제어
Phone : 042-821-3201
Fax : 042-821-3400
E-mail : yongwoon5901@gmail.com

김은태(Euntai Kim)

1992년 : 연세대학교 전자공학과 졸업 공학사
1994년 : 연세대학교 전자공학과 석사과정 졸업, 공학석사
1999년 : 연세대학교 전자공학과 박사과정 졸업, 공학박사
1999년 3월～2003년 2월 : 국립한경대학교 제어계측공학과 조교수
2002년 3월～현재 : 연세대학교 전기전자공학부 교수
관심분야 : Computational Intelligence, 지능형 로봇
Phone : 02-2123-7729
Fax : 02-313-2879
E-mail : etkim@yonsei.ac.kr