|  |  |
| --- | --- |
| *2017* 한국자동차안전학회 추계학술대회 논문집 |  |
| 개선된 A\* 알고리즘을 적용한 자율주행 전략 개발 및  머신러닝을 이용한 교통 표지판 인식    김학준\*·양호준\*·고상환\*·박민수\*·나윤서\*\*·김민영\*\*·박주은\*\*\*·황성호\*\*    **Development of Autonomous Driving Strategy applying Improved**  **A\* Algorithm and Traffic Sign Recognition using Machine-Learning**)    Hak-Jun Kim, Ho-Jun Yang, Yun-Seo Na, Sang-Hwan Go, Min-Young Kim, Min-Su Park, Ju-eun Park, Sung-Ho Hwang    *Key Words : Improved A\* algorithm(*개선된 에이스타 알고리즘*), Near obstacle detection function(*근접 장애물 검출 함수*), Improved f-cost function(*향상된 *f-*비용 계산함수*), Goal function(*목표지점 함수*), Machine learning(*머신러닝*), Sign recognition(*표지판 인식*), Negative image(*배경이미지*), Positive image(*표지판 합성 배경이미지)    **ABSTRACT**  This paper proposes the improved A\* algorithm and machine learning algorithm based on sign recognition for autonomous driving system. The improved A\* algorithm is used to determine the driving paths and required goal positions. The drivable paths with appropriate goals are produced by applying Goal function and Improved f-cost function considering near obstacles produce. In addition, the traffic sign recognition is performed using a machine learning algorithm. The machine-learning algorithm is used for the comparison with the background images called negative images and the road sign images, which are combined the background images, called positive images. The proposed algorithm provides good performances to the autonomous driving system through driving tests. | |
|  | |

**1.** 서 론

|  |
| --- |
| \*  성균관대학교 전자전기공학부 |
| \*\* 성균관대학교 기계공학부  \*\*\* 성균관대학교 시스템경영공학과 |
| E-mail : rlagrkwns30@naver.com, hsh0818@skku.edu |

4차 산업혁명 시대가 도래하며 자율주행 자동차에 대한 연구가 활발히 진행되고 있다. 자율주행 기술은 Camera, Lidar, Gps 등의 센서를 이용해 사람을 대신하여 교통효율성을 높이고 운전자의 편의성을 증대시켜줄 기술이기 때문에 그만큼 안전하고 신뢰성 있는 기술개발이 필요하다. 이번 교통안전공단 자율주행 공모전에서 기본 알고리즘으로 보편적인 에이스타 알고리즘과 표지판 인식을 위한 객체추적 알고리즘을 사용하였다. 기본 에이스타 알고리즘은 최적경로를 탐색하기 위해 Lidar와 Vison 차선인식 결과를 합친 100 by 100 매트릭스에서 주어진 목표지점까지 각 지점에서 여덟 방향으로 경로탐색을 실시하고 경로 값을 돌려준다. 또한 객체추적 알고리즘으로 탬플릿 매칭과 ORB 알고리즘을 사용하여 실시간 영상으로 표지판을 인식할 수 있도록 하였다.

하지만 기본 에이스타 알고리즘을 적용했을 때 매트릭스 상에서 8방향 모두를 탐색하여 시간이 많이 소요되고 실시간으로 목표지점이 변할 뿐만 아니라 최단거리 경로가 차량이 주행하기 어려운 직각에 가까운 경우가 많았다. 또한 객체 추적 알고리즘 같은 경우는 2차원 평면으로 나타내는 영상 속 표지판을 실시간으로 인식하는데 어려움이 있었다.

따라서 개선된 에이스타 알고리즘 개발 및 머신러닝을 사용하여 실시간으로 주행에 최적화된 경로를 탐색해 제어함수로 전달하고 표지판 인식을 가능하게 할 필요성이 있었다.

**2.** 개선된 에이스타 알고리즘과 표지판 인식 알고리즘 도입

**2.1** 개선된 에이스타 알고리즘 도입의 필요성

**2.1.1** 장애물과 차선을 곡선으로 회피하는 기능 필요

기존 에이스타 알고리즘을 사용하면 최단거리를 비교적 빠른시간 내에 알아낼 수 있으나 시작지점과 목표지점에 따라 장애물을 피하지 못하며 최단경로를 도출하기 위해 장애물과 차선에 붙는 경로를 도출하는 경우가 많았다. 이를 개선하기 위해 차량의 위치에서 일정한 크기의 원을 이용해 장애물·차선이 겹치는 면적을 계산하는 근접 장애물 검출함수를 도입하였다. 이 값을 f-비용 계산에 반영하여 장애물과 차선을 부드러운 곡선으로 회피할 수 있도록 하였다.

**2.1.2** 빠른 경로도출을 위한 **f-**비용 계산함수 변수조정

기존 f-비용 계산함수는 시작지점까지의 거리와 목표지점까지의 거리를 합산하여 최적 경로를 선택하는 방식이다. 이 과정에서 차량의 진행방향을 고려하지 않고 8방향을 모두 계산하기 때문에 효율이 떨어지고 f-비용계산 값이 커 매트릭스 상에서 값을 저장·사용할 때 연산속도가 느렸다. 이를 개선하기 위해 탐색시간 고정 및 탐색방향을 3방향으로 제한하고 차선 외부 지역을 탐색을 하지 않으며 거리 계산 값을 20배 축소시켜 빠른 연산이 가능하도록 하였다.

**2.1.3** 목표지점 함수를 통한 적절한 실시간 목표지점 설정

기존 에이스타 알고리즘은 미로와 같은 정적인 격자구조에서 주로 사용된다. 일반적으로 정적 경로탐색은 목표지점이 고정되어있지만 자율자동차에 적용하는 경우에는 Liadr와 Vison 차선인식 결과물이 계속 변하기 때문에 목표지점이 지속적으로 변해야 한다는 차이점이 있다. 따라서 적정한 목표지점을 검색하여 개선된 에이스타 알고리즘에 실시간 전달하기 위해 매트릭스에서 차량 위치를 기준으로 차선과 장애물을 고려하여 차량이 직선으로 도달할 수 있는 최대 지점을 검색하는 목표지점 함수를 도입하였다.

**2.2**머신러닝을 통한 표지판 인식 및 알고리즘 강화의 필요성

**2.2.1** 영상 속 표지판 이미지 왜곡 대비 인식 강화 필요

템플릿 매칭과 ORB 알고리즘을 이용하였을 때는 정적인 상황에서는 표지판을 잘 인식을 하였지만 동적인 상황에서는 실시간으로 인식을 하지 못하였다. 영상은 2차원 평면상에 이미지를 나타내기 때문에 원근법 등 이미지 왜곡이 발생하게 되고 위의 알고리즘은 이 부분에 취약하다고 판단하였다. 이에 이러한 왜곡상황을 대비할 수 있는 기계학습 알고리즘을 사용하였다.

**2.2.2** 곡선 구간 대비 인식 강화 필요

실제 대회에서 주행 중 곡선구간에서 다시 직선구간직선 진입할 때 다시 중심을 잡기 까지 차량제어가 불안정한 구간이 있었고 카메라를 통해 찍는 영상도 흔들림이 생겨 이미지 왜곡이 심해진다. 이에 대비하여 기계학습 알고리즘을 사용하여 인식을 강화하였다.

**2.2.3** 형태가 같은 표지판 구별 강화 알고리즘 필요

주차, 유턴 표지판 같은 경우는 사각형, 원형의 표지판 이였기 때문에 구별이 쉽게 가능하였다. 하지만 경고 표지판인 삼각형 표지판들 같은 경우는 형태가 비슷하여 인식 할 때 오차가 발생하였다. 이를 개선하기 위하여 표지판 속 경고 이미지를 탬플릿으로 따로 잡아 기계학습을 시켜주어 구별할 수 있도록 하였다.

**3.** 개선된 에이스타 알고리즘

**3.1** 근접 장애물 검출함수

**3.1.1 1**차 근접장애물 검출함수

장애물을 검출하기 위해 차량 위치를 기준으로 매트릭스가 100 by 100일 때, 지름이 18칸인 원을 도입하였다. 매트릭스 상에서 경로탐색 지점을 기준으로 하여 원에 포함되는 장애물·차선의 면적을 구하고 이 값을 f-비용 계산함수에 포함시킨다. F-비용 계산 값이 낮을 수록 우선순위 경로로 선택되기 때문에 최대한 장애물과 차선을 피하도록 경로를 내놓는 역할을 수행한다.

**3.1.2 2**차 근접장애물 검출함수

1차 근접장애물 검출함수와 비슷한 역할을 수행한다. 1차장애물 검출함수가 부드러운 곡선을 탐색하는 역할을 하지만, 차량의 크기를 반드시 고려하지는 않기 때문에 최단경로를 검색할 때 차량이 장애물이나 차선에 겹칠 가능성이 아직 존재한다. 따라서 2차 근접장애물 검출함수는 100 by 100 매트릭스 기준 지름이 6으로 설정된 원을 도입하여 경로검색 시 원안에 포함되는 장애물과 차선이 없도록 하여 차량의 크기를 반드시 고려하는 경로가 탐색되도록 하는 역할을 수행한다.

**3.2**향상된 **f-**비용 계산함수

**3.2.1 f-**비용 계산함수 변수 조정

기존의 에이스타 함수가 8방향을 탐색하므로 효율성이 떨어졌기 때문에, 개선된 에이스타 함수에서는 f-비용 계산을 차량의 진행방향과 일치하도록 진행방향과 진행방향의 양 대각선인 3방향만을 탐색하도록 하였다. 또한 에이스타 경로탐색이 계산 리소스를 많이 사용하므로 제어 함수와 Vision 등 다른 함수의 리소스를 확보하기 위해 f-비용 계산함수의 계산시간을 0.2초 이하로 제한시켰다. 추가적으로 차선 이외의 지역의 탐색을 제한시키고 f-비용 계산 값을 20배 축소시켜 연산시간을 0.1초 대로 단축시킬 수 있었다. 이를 통해 실제 차량주행 시점에 가장 근접한 데이터를 통한 차량제어가 가능해졌다.

**3.3** 목표지점 함수

**3.1.1** 적절한 실시간 목표지점 설정

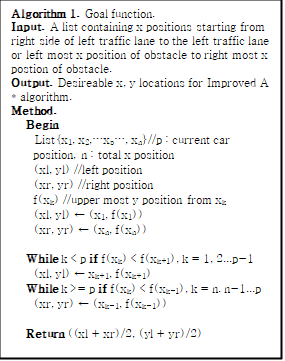


Fig. 1 Algorithm 1 Goal function algorithm

목표지점 함수를 통해 실시간으로 적절한 목표지점을 검출하기 위해 차량의 y위치를 기준점으로 인접한 차선 또는 장애물까지의 매트릭스 상의 x 좌표 값을 list에 저장한다. List에 저장된 x좌표의 시작점과 끝점에서 각각 출발하여 장애물이나 차선에 걸리지 않고 진행방향으로 직선 도달할 수 있는 최대 x, y 좌표를 직전 x, y 좌표의 y 값과 비교하여 y값이 큰 경우 좌표를 갱신하도록 한다. 최종적으로 구해진 두 x, y 좌표의 평균을 개선된 에이스타의 목표지점으로 설정한다.

**4.** 강화된 표지판 인식 알고리즘

**4.1.** 기계학습**(Machine learning)**

**4.1.1 Haar cascade** 알고리즘 이용

이미지에서 임의의 영역과 영역의 밝기 차를 이용하여 특징을 찾아내는[Viola2001] 논문을 인용하였다. 사람의 얼굴, 물체의 모양 등은 다양할 수 있지만 각 물체들 형상의 패턴은 비슷하므로 특정위치, 그리고 특정 영역의 픽셀 값들의 분포도 크게 차이가 없을 것이다. 이 영역을 경계 값(threshold)으로 잡는다.

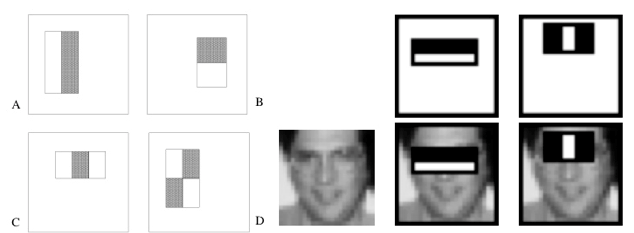


Fig. 2 Haar-like feature 검출 방법

Fig 2에서 사람의 얼굴 위에 흑백의 사각형을 겹쳐놓은 다음 밝은 영역에 속한 픽셀 값들의 평균에서 어두운 영역에 속한 픽셀 값들의 평균의 차이를 구한 뒤, 그 차이가 경계 값을 넘으면 Haar-like feature라고 판단하고 그 영역은 특징영역으로 잡는다. 다양한 흑백 사각형을 통하여 흑 영역에 속한 픽셀 값, 백 영역에 속한 픽셀 값들의 평균 차이를 다양하게 만들어 비교 분석하여 인식의 정확성을 높인다. 이렇게 미리 특징을 학습시켜놓은 데이터와 실시간으로 들어오는 영상 속 객체들의 특징을 비교 분석하여 인식여부를 판단한다. 개발환경은 리눅스를 사용하였고 Language는 Python 2.7를 사용하였다. 또한 Opencv에서 제공하는 Haar-training 라이브러리를 이용하였다.

**4.1.2 Positive, Negative image** 생성 및 두 이미지 비교를 통한 **Haar-training**

Image.net 에서 저장되어 있는 각 이미지들의 url을 이용하여 19157장을 Negative image로 받아왔다. 이미지들을 받아올때는 표지판이 있는 환경을 고려하여 도로, 교통상황 등을 표현하는 이미지들 위주로 저장하였다. Opencv\_createsamples 함수를 이용하여 각 표지판 이미지들을 Negative image에 합성시켰다. 단일이미지 방법을 활용하였고, 하나의 표지판을 19157장의 Negative samples에 합성시켰다. 합성 시킬 때는 함수 내에 저장되어있는 위상변조, 밝기변조 등의 변수들을 활용하여 이미지 왜곡현상을 대비할 수 있게 합성하였다. 이렇게 만든 Positive image들과 Negative image 들을 Opencv\_traincascade 함수를 이용하여Harr-training 비교학습을 시킨다.

**4.1.3** 동일한 형태의 표지판 구별

Positive image 생성 시 표지판 이미지의 테두리도기준값을 넘었다고 판단하여 특징영역으로 잡는다. 이러면 동일한 형태의 표지판을 구별할 때 오차가 발생한다. 그래서 이러한 경우에는 경고 표지판들의 안 쪽 경고 이미지들을 따로 sample 로 만들어 Positive image로 합성하였다. 이렇게 하여서 오차를 줄이는데 성공하였다.

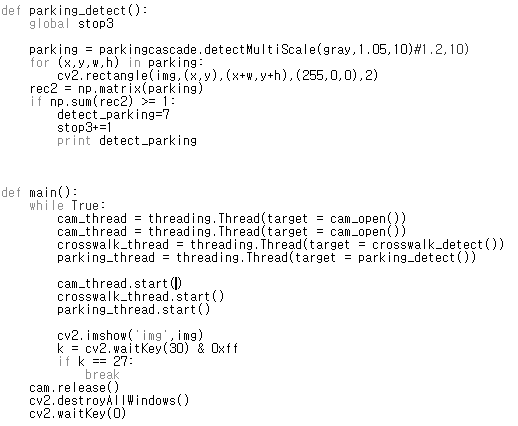


Fig. 3 part of sign detect & threading function code

**4.2** 표지판 인식 제어 함수

**4.2.1.** 표지판 인식 함수 동시제어

각 표지판을 인식하는 함수들을 병렬적으로 제어하여야 한다. Python의 동시성 모듈인 threading 모듈을 이용하여 대회 주행 중에는7개의 표지판을 동시에 검출할 수 있게 한다. Fig. 3은 주차표지판 검출함수이다. detectMultiscale 함수 의 scalefactor와 minneighbors변수 값으로 각 타깃 표지판들의 인식율을 결정한다. main함수에서는 주차표지판과 횡단보도 표지판의 thread를 생성하여 동시 검출을 시행하는 일부분 코드이다.

**5.** 개선된 에이스타 알고리즘과 머신러닝 도입의 결과

**5.1** 개선된 에이스타 알고리즘 도입의 결과

**5.1.1** 기본 에이스타 알고리즘

전체 경로지정에 소요된 시간:0.489 초, Fig 4. 인용

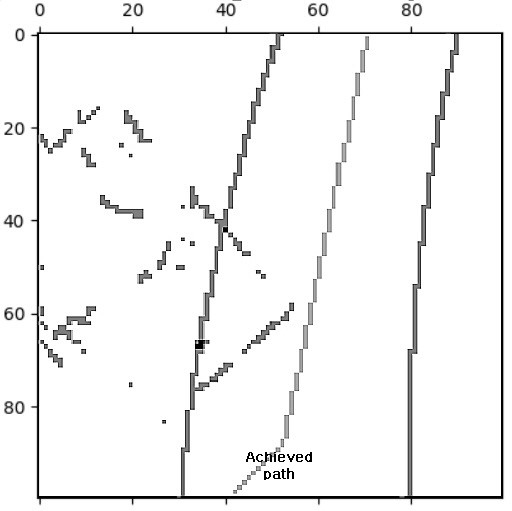


Fig. 4 A \* algorithm with fixed goal point

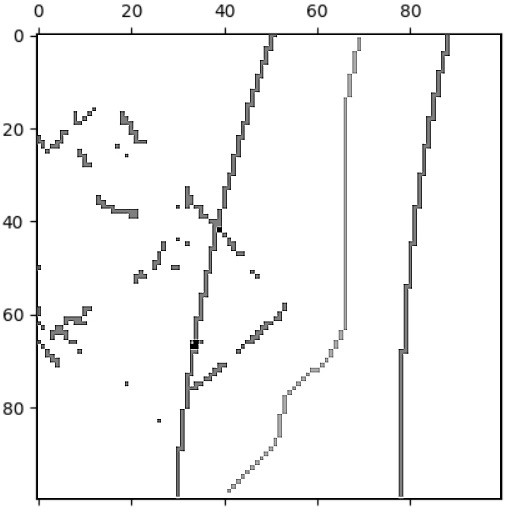


Fig. 5 A \* algorithm with fixed goal point and near obstacle detection function

**5.1.2** 근접 장애물 검출함수가 추가된 에이스타 알고리즘

전체 경로지정에 소요된 시간:0.349 초, Fig 5. 인용

기본 에이스타 알고리즘에 근접 장애물 검출함수를 적용하였다.

**5.1.3 f-**비용 변수 조정으로 개선된 에이스타 알고리즘

전체 경로지정에 소요된 시간:0.2 초, Fig. 6 인용

근접 장애물 검출함수, f-비용 계산함수 조정, 목표지점 함수를 모두 적용시킨 경우이다.

**5.2** 머신러닝을 이용한 표지판 인식 결과

**5.2.1 Positive image** 생성 및 **Haartraining**을 통해 생성된 **xml** 파일

Fig. 7 인용

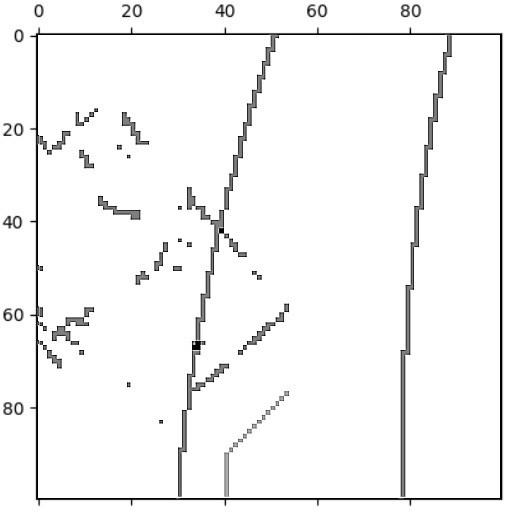


Fig. 6 Improved A \* algorithm

이미지 왜곡되는 현상을 대비할 수 있다. 학습은 30stage를 적용하였고, 각 단계마다 물체가 없는데 있다고 검출한 영역을 다시 Negative 이미지로 넣는 형식이라 마지막stage에 가까워질수록 오검출율이 줄어들고, Negative 이미지를 생성하는 시간이 오래 걸린다. 노트북으로는 30stage까지가 한계였고, 컨트롤러를 사용하여 40stage 학습을 시켜 곡선 구간 등 제어가 불안정한 상황에서도 인식율을 높였다.

**5.2.1** 카메라를 통한 유턴 표지판 인식

Fig. 8 인용

실시간으로 계속 인식되는 사실을 알 수 있다.

**5.2.3** 동일 형태의 표지판 구별

Fig. 9 인용

왼쪽그림은 표지판 전체를 학습시킨 것이고, 오른쪽 표지판은 내부 경고이미지를 따로 샘플로 만들어서 학습시킨 결과이다.



Fig. 7 Positive image with phase, brightness modulation applied



Fig 8. Uturn sign detection through a camera





Fig 9. Distinguish signs of same shape

**6.** 향상된 에이스타 알고리즘과 머신러닝의 이점

**6.1** 향상된 에이스타 알고리즘의 이점

**6.1.1** 장애물 회피의 이점

기본 에이스타 알고리즘이 장애물이나 차선에 근접한 주행경로가 나올 수 있다면 개선된 에이스타 알고리즘을 통해 미리 장애물을 인지하고 제어 함수에 전달해줄 수 있는 이점이 있다. Fig2와 Fig3를 비교한 경우에 개선된 에이스타 알고리즘이 장애물을 인지하고 회피방향을 유지하는 반면, 근접 장애물 검출함수만을 적용한 경우 최단경로 이득과 장애물 회피 이득의 중간영역이 존재하여 상대적으로 늦은 장애물 회피와 오실레이션을 보이는 것을 알 수 있다.

**6.1.2** 시간 단축의 이점

기존 에이스타 알고리즘의 0.489초에서 개선된 에이스타 알고리즘을 사용하여 0.2초대로 경로탐색을 줄여 약 60%의 시간 이득을 도출할 수 있었다. 시간단축을 통해 개선된 리소스를 Vison등에 배분하여 지연이 개선된 데이터를 바탕으로 자율주행을 실시할 수 있는 이점이 있다.

**6.1.3** 적절한 목표설정의 이점

Fig1과 Fig2를 고려하였을 때 반드시 먼곳의 목표지점을 잡는 것이 도움이 되는 것은 아니라는 것을 알 수 있다. 리소스를 사용하여 먼 지점까지 경로계산을 하여도 모든 경로를 제어에 활용할 수 없기 때문이다. 따라서 목표지점 함수를 통해 유의미한 거리까지의 경로계산을 한 후 즉각적인 방향제어를 유지하는 개선된 에이스타 알고리즘이 리소스·시간 대비 큰 이득을 준다는 것을 알 수 있다.

**6.2** 머신러닝을 이용한 표지판 인식의 이점

**6.2.1** 영상 속에서 실시간으로 표지판 인식 가능

특징점으로 물체를 찾는 학습을 반복하여 인식 알고리즘을 강화하였고 , Fig. 8과 화면 속에서 표지판 이미지가 사라지지 않는다면 계속 인식하고 있다.

**7.** 결 론

본 연구의 결과를 요약하면 다음과 같다.

1) 에이스타 알고리즘 개선을 위한 방식을 알아보았고 개선 전의 알고리즘의 결과들과 비교하여 장애물 회피, 시간, 리소스 대비 측면에서 이득이 된다는 것을 살펴보았다.

2) 머신러닝 알고리즘을 통해 표지판 인식이 가능한 점을 알아보았고 이미지 왜곡, 햇빛 변화 등에 대비 positive 이미지 생성 시 위상, 밝기 변화등을 적용시켜 이미지 합성을 하였다. 영상인식 쪽에서 취약한 부분을 강화할 수 있었다.

그러나 실제 주행에서는 GPS값의 오차가 있어 에이스타 알고리즘을Global map과 활용하는 부분까지는 못하였다. 또한 조도의 변화에 따른 표지판 인식율 차이를 해결하지 못하였다. detectMultiscale 함수의 변수 값들을 바꿔주면서 대회를 준비하였다.  향후에는 딥러닝 방법 중 뉴럴 네트워크 방법을 통해서 제어가 힘든 상황에서도 표지판이 실시간으로 인식이 되는 알고리즘을 생각해보겠다.

참고문헌

1. P. Viola and M. J. Jones, 2001, “Rapid Object Detection using a Boosted Cascade of Simple Features”,
2. 김영훈, 이상원, 2015, A스타 알고리즘에 방향 가중치를 적용한 최단거리 경로탐색에 관한 연구