인공지능 기반 메카넘휠 주차로봇 연구

박현규, 최상현, 이준경, 채원호*
경북대학교 전자공학부*
us04128@naver.com, ung9809@knu.ac.kr, chae8787@naver.com, tkdgus6147@gmail.com

A Study on Mecanum wheel Parking Robot Using Artificial Intelligence

Hyun-Gyu Park, Jun-Gyeong Lee,Won-ho Chae, Sang-Hyun Choi*
School of Electronics Engineering, Kyung-Pook National University*

요 약

주차 공간의 부족으로 인해 생기는 문제점들을 해결하기 위해 기존 기계식 주차타워가 아닌 새로운 주차 방식으로 자동 주차 로봇을 만들어보고자 한다. A* 알고리즘과 QR코드, 각종 센서들을 통해 인공지능 프로그램을 구현하고, 회전반경을 줄여 공간의 효율성을 극대화할 수 있는 메카넘휠을 사용해 모바일 주차로봇을 만들어본다.

1. 서 론

차량은 지속적으로 증가하고 주차 공간 부족현상으로 인한 문제들이 발생하고 또한 주차 중 접촉사고, 개문 사고 등 갈등이 지속적으로 발생한다. 이를 해결하기 위한 방법으로 메카넘휠[1] 주차로봇을 만들고자 한다. 메카넘휠은 로봇을 전 방향 이동을 가능하게 하며 주차 회전반경을 줄일 수 있기 때문에 주차장의 면적을 보다 효율적으로 사용할 수 있다.

기존의 위치 인식 자율 주행 로봇은 슬램 방식 또는 마그네틱 라인에 의존하여 주행하는 로봇으로서 자율성이 보장되면 정확도가 떨어지고, AGV 방식과 같이 마그네틱 라인을 인식하여 정확도는 일부 상승하였지만, 실내이동로봇으로서의 자유도는 매우 떨어진다. 또한 주차 로봇은 주차 공간과 로봇이 다니는 길이 정해져 있으므로 일정 수준 이상의 자유도만 가지면 된다. 이에 본 연구에서는 QR 코드를 사용하여 이전 AGV 방식에서 벗어나 자율 주행 및 위치 인식이 가능한 연구 결과를 제안하고자 한다.

2. 설계 및 구현

2.1 메카넘휠

메카넘휠은 바퀴 테두리 전체에 여러 개의 롤러가 바퀴 회전축과 45도 각도를 이루면서 대각선 방향으로 붙어있다. 이 롤러는 4개의 메카넘휠이 올바르게 조립되어 있는 상황에서만 구동한다. 각각의 회전 속도와 구동 방향을 달리하면 바퀴에 달린 롤러들의 진행 방향이 벡터의 조합을 이루게 되어 차량의 좌우 이동, 회전, 사선이동이 가능하게 된다.
로봇에 메카넘휠의 특성을 이용하면 높은 자유도를 가지므로 현재 다양한 산업 분야에 적용되고 있다.

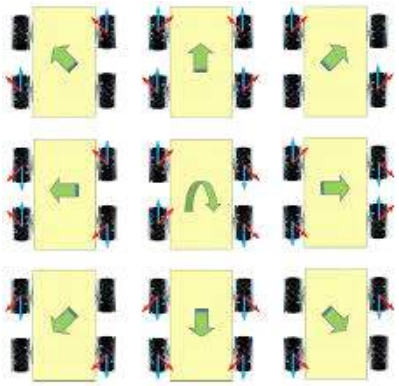

그림 2. 메카넘휠 구동 방향

2.2 QR코드

로봇의 제어를 위해 환경지도가 필요하다. 주차장은 이동 경로와 주차 칸이 정확히 나누어져 있기 때문에 격자 좌표 형태로 환경지도를 구성할 수 있다. 좌표 형태로 환경지도를 구성하게 되면 구현 코드 내에서 좌표를 배열로 표현할 수 있다. 배열로 표현하면 A* 알고리즘으로 최적의 경로를 구현할 수 있게 된다. 이 때 QR코드에 좌표 정보를 저장하여 주차장 바닥에 붙여 환경지도를 구성한다. 바닥에 부착된 QR코드를 로봇에 있는 리더기로 스캔하여 좌표 데이터를 얻고 지도상에서 로봇의 현재 위치를 파악한다. 현재 위치

에 따라서 이동할 방향이 결정되고 목적지까지 이동할 수 있게 된다. 이는 이동시 높은 위치 정밀도를 갖는 장점이 있다.

2.3 A* 알고리즘

A* 알고리즘은 출발 지점에서 목표 지점까지 최단 경로를 찾아내는 알고리즘이다. Dijkstra[2] 알고리즘의 확장이다. Dijkstra 알고리즘은 가중치 그래프로 출발 노드 기준으로 모든 노드의 최단 경로를 구하는 알고리즘이다. 따라서 모든 노드를 검색하기 때문에 시간 비용이 많이 든다는 단점이 있다. 반면 A* 알고리즘은 다음 탐색 노드를 선정하는 방식이 Dijkstra 알고리즘과 조금 다르다. A* 알고리즘은 휴리스틱 추정값을 사용하는데, 이 값은 탐색 우선순위가 생기기 때문에 검색할 노드가 줄어들어 시간 비용의 효율이 높아진다. 시작 노드부터 현재 노드까지의 가중치를 $g(x)$, 현재 노드에서 목표 노드까지의 추정 경로 가중치를 $h(x)$라 할 때, 이 두 값을 더한 $f(x)=g(x)+h(x)$가 가장 최소가 되는 노드를 다음 탐색 노드로 선정한다.

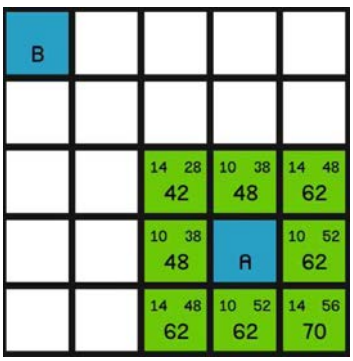

그림 2 노드 및 가중치 예시

그림 3을 보면 출발 노드(A)와 목표 노드(B)가 있고, 한 칸의 길이는 10으로 둔다. 가중치를 측정할 때에는 Manhattan distance 측정법을 사용하는데, 이는 장애물을 무시한 거리이며 다음과 같이 표현된다.

$$d(A,B)=\sum_{i=1}^{n}|a_i-b_i|$$

따라서 수평, 수직노드로 움직일 때에는 가중치가 10, 대각선 노드로 움직일 때에는 $\sqrt{2}$를 곱한 약 14가 된다. 이러한 조건에서 현재 출발 노드에서 인접 노드의 $f(x)$를 구하면 그림과 같이 계산 값이 나온다. 왼쪽 위의 숫자는 $g(x)$, 오른쪽 위의 숫자는 $h(x)$, 중앙에는 총합인 $f(x)$의 값이 나오게 된다. 그림 3에서는 총합이 42로 제일 작은 왼쪽 위 노드를 선택하게 되고, 그 노드에서 다시 $f(x)$를 계산하고 찾아가는 것이 A* 알고리즘의 핵심이다. 알고리즘을 구현할 때에 이동 가능한 노드의 집합인 Open List와 이동 불가능한 노드의 집합인 Close List라는 보조 데이터를 사용한다. 이 보조 데이터들을 사용하는 과정에 대해 간략히 설명해 보면, Open List에서 가장 낮은 $f(x)$를 찾아 현재 노드로 선택하고, 이 노드를 Close List로 넣는다. 선택된 노드는 다시 인접한 4개의 노드에 대해 $f(x)$를 탐색한다.(주차 로봇의 경우 대각선 주행의 필요성이 없기에 대각선 주행을 제외한 좌, 우, 아래, 위 4가지 노드만 탐색한다.) 이 때 인접한 노드가 가지 못하거나 Close List에 있다면 무시하고, 그렇지 않으면 계속 진행한다. 이 때 노드가 Open List에 있지 않다면 이것을 Open List에 추가한다. 그리고 이 노드의 부모를 현재 노드로 지정하여 $f(x),g(x),h(x)$ 값을 기록한다. 만약 노드가 이미 Open List에 있다면, $g(x)$를 이용하여 가장 작은 값을 가지는 노드를 부모 노드로 바꾸고 다시 $g(x),f(x)$ 값을 계산한다. 이러한 과정을 반복하면 목표 노드가 Open List에 포함될 경우가 나온다. 이러한 경우 목표지점에 대한 최적 경로가 나오게 된다.

```
[(6, 1), (5, 1), (4, 1), (3, 1), (2, 1), (2, 0)]
출발 (6, 1) -> 목적지 (2, 0)
now :6,1
(6, 1) --> (5, 1)
ADVANCED
STOP
b'F'
b'i'
now :6,1
(6, 1) --> (5, 1)
ADVANCED
b'n'
now :4,1
(4, 1) --> (3, 1)
ADVANCED
b'd'
now :2,1
(2, 1) --> (2, 0)
Left_2
STOP
```

그림 4 최적 경로와 모터 제어

위 과정을 거쳐 만들어진 최적 경로를 바탕으로 로봇의 주행 방향이 정해진다. 현재 노드와 다음 노드를 비교하여 앞, 뒤, 좌, 우 방향을 계산하고 Serial 통신으로 모터가 연결된 Arduino에 보내게 된다. 계산된 방향으로 로봇을 주행하고 이를 통해 로봇의 자율주행이 가능해 진다. 기존의 AGV는 하나의 길로만 다닐 수 있기에 장해물이나 지도가 변한다면 시스템을 새롭게 구성하여야 하지만 본 방식을 이용하면 차량이 주차된 곳을 알아서 계산하고 자유롭게 주행하여 주차할 수 있게 된다.

2.4 실시간 방향 제어

주차 로봇이 정해져 있는 길에서 벗어나지 않기 위해 IR센서와 검정색 전기 테이프를 이용해 라인 트레이싱을 하였다. 기존의 AGV 방식과 동일하며 높은 정확도를 가질 수 있게 된다. 전 방향 이동이 가능한 메카넘휠의 특성상 2륜 구동으로 움직이는 로봇과 달리 4개의 IR센서가 필요하다. 좌우와 앞뒤 주행 시 알고리즘을 달리하여 올바른 방향으로 주행할 수 있게 하였다. 정해진

길을 벗어날 경우 또는 올바른 각도로 주행하지 못하고 방향이 틀어졌을 경우 IR센서로 색을 감지해 자세를 조정하게 되고 이는 높은 정확성을 가지고 주행을 할 수 있게 한다.

2.5 로봇의 구성

아래는 주차 로봇의 하드웨어 구성도이다. 주 제어기로 Raspberry PI를 사용하였고 모터 구동으로 Arduino를 사용하였다. 현재 위치 파악을 위한 QR코드 리더기는 라즈베리파이에 연결하였고 실시간 방향 제어를 위한 4개의 IR센서는 Arduino에 연결하였다. 모터는 모터 드라이버를 통해 Arduino에 연결하였다. 라즈베리파이로 최적경로 계산, Arduino와 Serial 통신으로 모터제어, APK과 통신을 하였고 Arduino에서 실시간 방향 제어가 이루어진다.

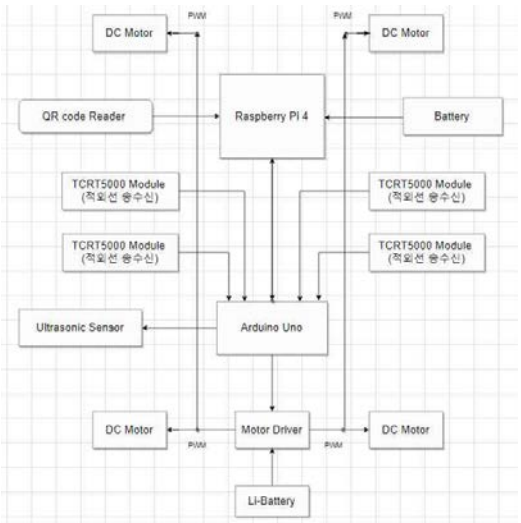

그림 5 H/W 설계도

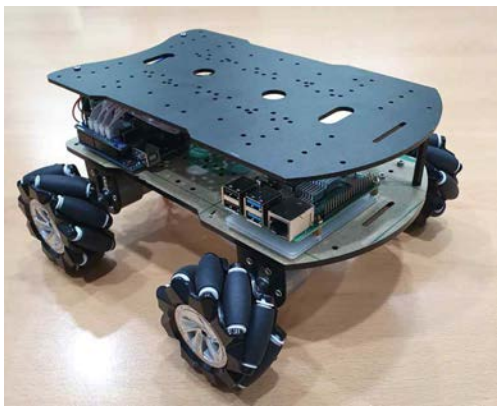

그림 6 구현한 주차로봇

사용의 편의성을 위해 직접 모바일 어플리케이션을 구현하였다. 사용자가 어플리케이션을 통해 차량 정보를 입력하고 주차 버튼을 클릭하면, 로봇은 A* 알고리즘을 통해 최적 경로를 생성하고 QR코드를 읽어 실시간으로 위치파악을 하며 메카넘휠을 통해 최소한의 회전반경으로 주차공간에 도달하게 된다.

그림 7 주차로봇 App

3. 결론

본 논문은 QR코드를 이용한 메카넘휠 주차 로봇의 위치 인식 및 자율주행 방법에 대해 제안하였다.

기존 AGV 방식의 문제점 해결을 위해 QR코드를 추가하여 정확도는 유지하며 자율성 또한 보장하였다. 또한 슬램 방식이나 인공 지표를 이용한 위치 인식 방식보다 높은 정확도를 가지며 낮은 시공비용이 발생한다.

일반적인 주차장에 비해 주차 공간을 40%이상 늘렸으며 운전 미숙으로 인한 사고 또한 발생하지 않았다. 주차타워가 기계설치 등의 비용이 많이 발생하는 것에 비해 스티커 부착 및 로봇의 비용만 추가되기 때문에 가격적인 면에서도 우수하다.

향후 본 논문에서 제안한 QR 코드 기반 위치인식 방법으로 주차 로봇뿐만 아니라 다양한 실내 이동 로봇에 적용하여 개발된다면 로봇 위치 인식 기술은 더욱더 발전할 수 있을 것이다. 이를 통해 다양한 서비스 로봇들이 우리의 삶의 질을 향상 시킬 것을 기대한다.

※ 본 논문은 과학기술정보통신부 정보통신창의인재양성사업의 지원을 통해 수행한 ICT멘토링 프로젝트 결과문 입니다.

참고문헌

[1] 주백석, Design and Kinematical Discussion of an Omni-directional Mobile Robot based on Mecanum Wheel, 제어로봇시스템학회 국내학술대회, 2012, p376-377

[2] 문지혜, Dijkstra* Algorithm: Minimum Search Cost Optimal Path Exploration, 한국정보과학회, 2018, p323-325