한국마린엔지니어링학회지 제32권, 제1호, pp. 192~199, 2008. 1 (ISSN 1226-9549)
Journal of the Korean Society of Marine Engineering

이동로봇의 맵 빌딩 기반 최적 주행 알고리즘

김종화† · 김진규* · 임재권** · 한승봉***

(원고접수일 : 2007년 12월 31일, 심사완료일 : 2008년 1월 14일)

An Optimal Traveling Algorithm Based on Map Building for Mobile Robots

Jong-Hwa Kim† · Jin-Kyu Kim* · Jae-Kwon Lim** · Seong-Bong Han***

Abstract : In order for a mobile robot to move under unknown or uncertain environment, it is very important to collect environmental information. This paper suggests a traveling algorithm which leads to the map building algorithm and the A* algorithm under the assumption that environmental information should already be collected. In order to apply the proposed traveling algorithm to a real mobile robot, this paper additionally discusses a path amendment algorithm. For the purpose of verifying the proposed algorithms, several simulations are executed based on a UI host program-based simulation interface and an experiment is executed using a mobile robot under a real unknown environment.

Key words : Traveling algorithm(주행 알고리즘), Map building algorithm(맵 빌딩 알고리즘), A* algorithm(A* 알고리즘), Path revision algorithm(경로수정 알고리즘)

1. 서 론

오늘날에는 수많은 로봇들이 사람들과 함께 공존하고 있다. 여러 산업 현장에서는 오래 전부터 로봇이 사람을 대신하여 많은 역할을 해오고, 현재에 이르러서는 비단 산업 현장뿐만 아니라 사람들의 일상 속에서도 로봇의 모습과 역할을 쉽게 찾아볼 수 있는데, 이렇듯 산업 현장에서부터 일상생활 속까지 널리 사용되는 이유는 로봇의 공통적 형태가 바로 이동로봇의 형태를 지니고 있기 때문이다. 다양한 형태의 이동로봇들은 공장자동화, 빌딩 감시 등의 일반적인 산업현장에서부터 우주 탐사, 원자로 등의 극한적인 분야와 청소대행 혹은 간호보조 등의 역할을 수행하는 서비스 분야에까지 다양하게 활용되고 있다. 그러나 아직도 이동로봇을 일상생활에 사용하기에는 여러 가지 극복해야할 문제들이 많다. 특히, 항상 다른 환경에서 임무를 수행해야 하는 경우 각 환경에 대한 인식이 선행되어야 하며, 이를 통해 이동로봇의 최적 주행을 기대할 수 있을 것이다.

이에 본 논문에서는 이동로봇의 주행환경에 대한 맵 빌딩 알고리즘과, 최단 경로와 시간을 고려한

† 교신저자(한국해양대 컴퓨터 · 제어 · 전자통신공학부), E-mail : kimjh@hhu.ac.kr, Tel : 051)410-4343
* 대전 · 충남지방중소기업청 기술지원과
** 한국해양대 박사과정
*** 삼성전자 DM 연구소

주행을 하기 위해 A* 알고리즘 및 경로수정 알고리즘을 적용한 주행 알고리즘을 제안하고 시뮬레이션과 실험을 통해 그 실효성을 검증해 보고자 한다.

2. 맵 빌딩 알고리즘

기존의 이동로봇의 환경 지도 작성 방법에는 특징 추출법과 격자식 방법으로 나눌 수 있다. 특징 추출에 의한 지도 작성은 주어진 환경의 평면(Plane), 구석(Corner), 모서리(Edge) 등을 검출하여 이를 트리(Tree) 형태로 저장하는 방법이다. 격자식 방법은 주어지는 환경을 2차원 또는 3차원 격자로 나눈 후 해당되는 셀(Cell)에 가중치를 부여하여 표현하는 방법이다. 본 논문에서는 위 두 가지 방법의 장점을 모두 응용한 방법을 이용하였다[1]. 최초 이동로봇의 초음파 센서를 통해 획득되는 거리 및 방향 정보를 바탕으로 평면, 구석, 모서리를 검출하고 이를 특징 추출법처럼 단순히 트리형태로 저장 공간에 저장하는 것이 아니라 격자식 방법처럼 환경을 2차원 격자로 나누고 그 2차원 격자의 각 셀에 가중치를 주어 저장하는 것이다. 이동로봇이 맵 빌딩을 수행하는 과정은 다음과 같다.

① 이동로봇은 환경을 인식하지 못한 공간에 위치하고 있다고 가정하기 때문에 그 공간의 가운데 위치하고 있다고 가정하고 출발한다. 출발과 동시에 초음파 센서를 통해 주변 환경 정보를 수집하여 가장 가까운 벽이나 장애물이 위치한 곳으로 이동한다.

② 이동로봇은 벽과 이동로봇의 좌우측면을 평행하게 유지한 상태에서 이동하게 된다. 이 때 벽을 따라 이동하면서 전방과 좌우측면에 위치해 있는 총 8조의 초음파 센서를 통해 주변 환경 정보를 수집하게 된다. ③ 벽을 따라 이동 및 방향 전환 등의 반복적 수행을 거치면서 최초 출발한 위치에 도달하게 된다. 이동로봇은 벽을 따라 이동하면서 일정 거리 간격으로 인덱스(Index)를 지정하고 벽을 따라 이동하면서 부여된 인덱스를 확인함으로써 폐곡선이 생성된 것을 확인할 수 있다.

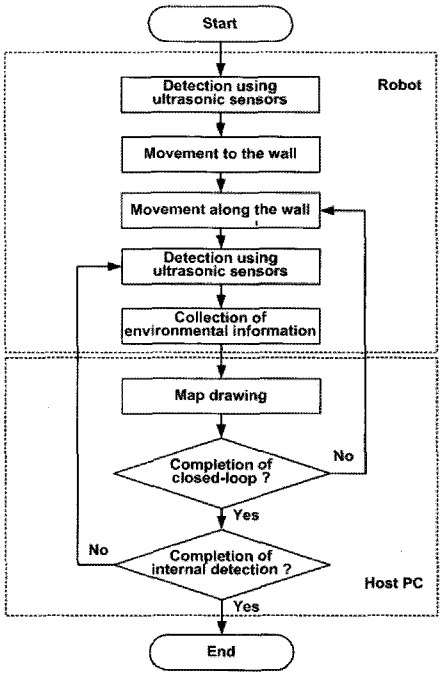

Fig. 1 Flowchart of the map building algorithm

④ 폐곡선이 생성된 후에는 폐곡선 내부의 탐색 유무를 점검한다. 벽을 따라 이동을 하면서도 벽과 반대 방향에 위치해 있는 초음파 센서는 내부 공간을 탐색하고 있기 때문에 작은 크기의 공간에서는 폐곡선 내부 공간에 대한 별도의 탐색이 필요 없지만, 큰 공간에서는 내부 공간에 대한 별도의 탐색이 필요하기 때문이다. 폐곡선 생성 후 내부 공간에 대한 탐색이 완료되어 있지 않다면 이동로봇은 폐곡선 내부로 이동하여 인식되지 못한 주변 환경에 대한 탐색을 수행한다.

⑤ 벽을 따라 이동하며 완성한 공간에 대한 윤곽과 내부 공간에 대한 탐색이 모두 완료되고 나면 일련의 맵 빌딩은 완료된다. Fig. 1은 지금까지 설명한 맵 빌딩 알고리즘의 수행 순서도를 나타낸 그림이다.

3. A* 알고리즘

맵 빌딩 알고리즘을 통해 이동로봇이 위치하는 환경에 대한 맵 빌딩이 완료되고 나면 이동로봇은 환경 정보를 기초로 하여 주행을 실시하게 된다.

시작 지점과 목표 지점 사이에 최단 경로를 계산하고 생성된 최단 경로로 주행하게 된다.

본 논문에서는 최단 경로 생성을 위하여 A* 알고리즘을 사용하였다[1]-[5]. A* 알고리즘은 범용적인 특성상 많은 분야에 적용시킬 수 있는 장점을 가지고 있어, 맵 빌딩을 통해 생성된 환경 정보를 바탕으로 최단 경로를 계산하는데 사용하였다.

A* 알고리즘은 1968년 AI 분야에서 개발된 범용 적이면서 효율적인 길 찾기 알고리즘 중 하나인데, A* 알고리즘을 이해하기 위해서는 f(Fitness), g(Goal), h(Heuristic)의 개념을 이해하여야 한다. g란 시작 지점에서 해당 지점까지 오는데 드는 비용을 의미하는데 이는 계산되는 값이다. h는 해당 지점에서 목표 지점까지 가는데 드는 비용을 의미하며, 아직 계산된 값이 아니라 추정된 값이다. 이러한 g와 h값을 계산하는데 있어 사용되는 방법은 사용자에 의존적이기 때문에 범용이라는 특징을 가지고 있다. f는 g와 h의 합, 즉 해당 지점을 거쳐 목표 지점까지 가는데 드는 비용을 말한다. 따라서 f값이 가장 작은 값이 최단 경로에 해당한다.

A* 알고리즘을 이해하기 위해서는 열린 목록(Open list)과 닫힌 목록(Closed list)에 대한 개념을 설명해야 하는데, 열린 목록이란 아직 탐색하지 않은 지점들로 구성되며 앞으로 탐색할 지점들을 의미한다. 여기서 탐색이란 각 지점에 대한 f, g, h값을 계산하는 것을 의미한다. 그리고 닫힌 목록이란 이미 탐색한 노드들을 의미하고 여기서 탐색했다는 의미도 그 지점에 연결된 모든 지점들에 대해 f, g, h값이 계산되었다는 것을 의미한다.

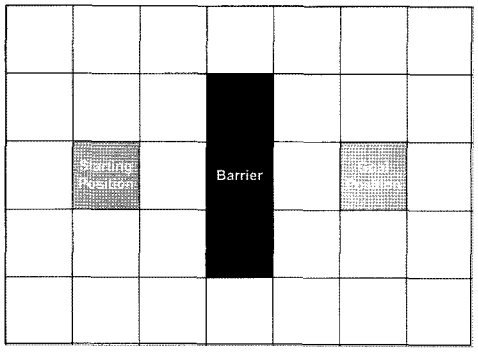

Fig. 2 An environment of A* search

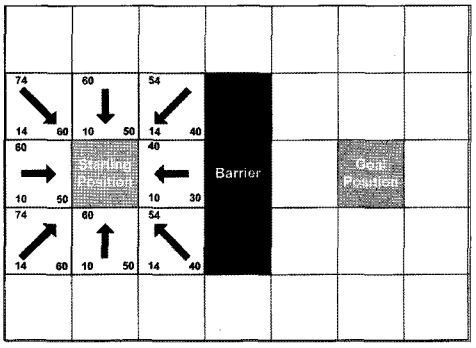

Fig. 3 Calculation of f, g and h cost

A* 알고리즘이 수행되는 과정을 살펴보기 위해 Fig. 2와 같이 간략하게 맵을 구성하였으며, 이를 기반으로 과정을 설명하면 다음과 같다. 이 때 Fig. 2에서 왼쪽이 시작 지점, 오른쪽이 목표 지점, 가운데 위치한 것이 장애물에 해당한다. 시작 지점과 목표 지점이 정해진 상태에서 우선 시작 지점을 닫힌 목록에 저장한다. Fig. 3과 같이 시작 지점에 접해있는 모든 지점들을 열린 목록에 저장하고, 저장한 지점들의 부모 지점으로 시작 지점을 지정해 준다. 그리고 열린 목록에 저장된 자점들에 대해 f, g, h값을 계산한 후 f값이 가장 작은 지점, 즉 최단 경로로 이동한다. 이렇게 하여 이동한 지점을 Fig. 4와 같이 다시 닫힌 목록에 저장하고, 이동한 지점 주위에 접해있는 모든 지점들을 열린 목록에 저장한다. 물론 저장된 지점들의 부모 지점으로는 이동한 지점을 지정해준다. 앞에서와 마찬가지로 열린 목록에 대해 f, g, h값을 계산하게 되는데, 열린 목록에 저장된 지점들 중 f, g, h값이

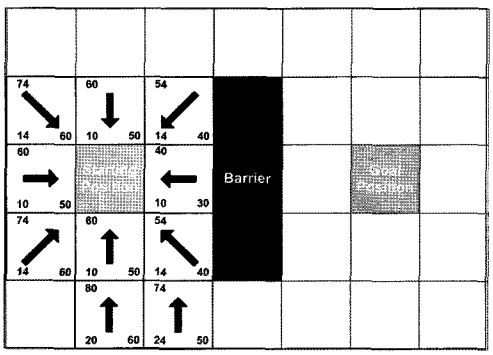

Fig. 4 Repetitive accomplishment of the A* algorithm

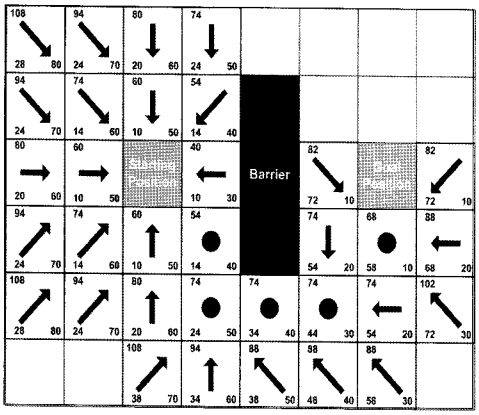

Fig. 5 Example of the shortest course creation

계산되어져 있는 지점들도 이동한 지점을 기준으로 다시 f, g, h값을 계산하게 된다. 계산된 f값이 기존의 f값보다 작으면 갱신한 후 부모를 해당 지점으로 재지정해주고, 계산된 f값이 기존의 f값보다 크면 무시하게 된다. 계산과 갱신이 완료되고 나면 또다시 f값이 가장 작은 지점, 즉 최단 경로로 이동하게 된다.

위와 같은 과정을 반복하게 되면 Fig. 5와 같은 결과를 얻을 수 있다. 그리고 이와 같은 과정은 목표 지점이 닫힌 목록에 속할 때까지 반복하게 된다. 목표 지점이 닫힌 목록에 속한 후에 최단 경로를 구하는 것은 매우 간단하다. 지금까지 이동하면서 지정해 두었던 부모 지점을 활용하게 되는데 목표 지점에서 부모 지점을 따라 역으로 이동하게 되면 그것이 바로 최단 경로가 된다. Fig. 6은 지금까지 설명한 A* 알고리즘을 정리하여 순서도로 나타낸 그림이다.

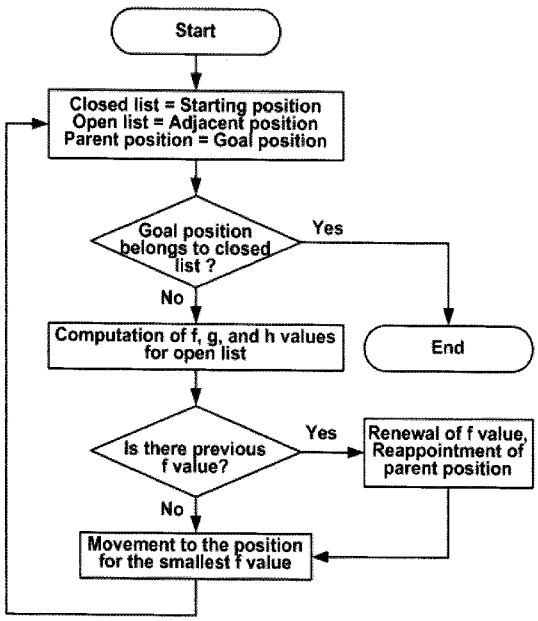

Fig. 6 Flowchart of the A* algorithm

A* 알고리즘은 매우 강력하면서도 범용 적이어서 다양한 분야에 적용이 가능하고 성능 또한 보장받을 수 있는 효율적인 길 찾기 알고리즘이다. 하지만 모든 알고리즘이 그렇듯이 장점만을 가지고 있는 것은 아니다. 몇몇 단점을 가지고 있고 이러한 단점을 보완함으로써 A* 알고리즘을 좀 더 최적화 시킬 수 있다.

맵의 크기가 크면 열린 목록과 닫힌 목록에 수백, 수천의 지점들이 포함될 수 있고 이렇게 되면 많은 메모리를 차지하게 되고 지점을 검색하는데 많은 CPU 시간을 할당해야 한다. 그리고 시작 지점에서 목표 지점까지 경로가 없는 경우에는 이를 미리 인지하고 못하고 전체 맵에 대한 열린 목록과 닫힌 목록을 관리하고 모든 지점에 대한 f, g, h값을 연산하여야 하는 굉장히 비효율적인 작업을 수행하게 된다. 이와 같은 단점을 보완하고 최적화시키기 위해서 A* 알고리즘이 사용할 수 있는 상한 시간을 설정해 두어 특정 시간 이상이 걸렸지만 경로를 찾지 못했다면 알고리즘의 성능을 저하시키기 전에 검색을 중단하고 부분적인 경로만 돌려주는 것이다. 그렇게 하고 난 뒤에 다시 나머지 검색을 수행하면 알고리즘의 성능 저하를 막을 수 있다. 또, 웨이브 포인트(Wave point)를 이용하는 것도 굉장히 좋은 방법 중에 하나이다[6]. 웨이브 포인트를 둘 경우 큰 지형을 작은 부분 지형으로 분할하여 관리할 수 있으며 길 찾기 알고리즘을 수행할 경우 그만큼 탐색할 공간이 줄어들어 성능을 향상시킬 수 있다.

4. 경로수정 알고리즘

A* 알고리즘을 사용하여 최단 경로를 생성하게 된다. 하지만 이는 알고리즘 상에서 거리만을 고려

한 최단 경로이다. 본 논문에서 실험과 시뮬레이션에 이용한 이동로봇은 이동 중 방향 전환이 필요한 경우 15°를 기준으로 방향전환을 수행하게 된다. 방향전환을 수행할 때 정지 후 방향전환을 수행하기 때문에 방향전환을 수행하는데 많은 시간이 소요된다. 따라서 방향전환을 최소화 하게 되면 그만큼 정지하는 횟수가 줄어들게 되어 주행 시간을 절약할 수 있게 된다. 이러한 점들로 인해 최단 경로를 생성할 때 A* 알고리즘을 통해 거리만을 고려할 것이 아니라 시간적인 면에서의 최단 경로도 고려되어야 할 것이다. 그리고 A* 알고리즘 최적화 방법에서 언급하였듯이 A* 알고리즘의 성능을 최적화시키기 위해 웨이브 포인트를 사용하였다. 이러한 웨이브 포인트를 사용함에 있어 웨이브 포인트 간의 연결 지점에서 방향전환점이 추가적으로 발생하게 된다.

웨이브 포인트란 일정 크기 이상의 맵을 분할할 때 분할된 각 맵의 입구를 의미한다. 웨이브 포인트를 이용하여 최단 경로를 생성하고자 할 때, 분할된 각 맵에서 최단 경로를 생성한 후 맵 통합 시에 하나의 최단 경로로 통합하기 위해 이 웨이브 포인트를 사용하게 된다. 따라서 웨이브 포인트를 사용하여 분할된 각 맵에서 최단 경로를 구하게 되면 분할된 맵 내에서의 시작점과 목표점은 각각 웨이브 포인트의 출입구가 되는 것이다. 맵 통합 시 웨이브 포인트를 통해 각기 분할된 맵의 최단 경로를 연결하게 되면 많은 경우에 웨이브 포인트를 기준으로 방향 전환점이 발생하게 된다. 최단 경로 생성 알고리즘인 A* 알고리즘이 아닌 웨이브 포인트로 인해 방향 전환점이 발생하게 된다면 수정할 수 있는 방향 전환점이 많게 되는데 그 수정 방법은 다음과 같다. 하나의 웨이브 포인트에 접해 있는 두 개의 분할된 맵에서 각각 웨이브 포인트에 도달하기 전 마지막 방향 전환점을 찾는다. 그리고 두 개의 방향 전환점 좌표 값을 대각으로 둔 사각형을 그리게 된다. 만들어진 사각형은 다시 두 개의 방향 전환점을 잇는 대각선을 기준으로 두 개의 삼각형으로 나누어 볼 수 있다. 이 두 개의 삼각형 중 하나라도 벽이나 장애물에 방해를 받지 않고 존재할 수 있다면 웨이브 포인트로 인해 발생하는 불필요한 방향 전환점을 줄일 수 있게 되는 것이다.

Fig. 7은 경로 수정 알고리즘의 결과를 보여주고 있다. 굵은 선으로 표시된 경로는 A* 알고리즘을 통해 계산된 최단 경로이다. Fig. 7에서 확인할 수 있듯이 거리만을 고려한 최단 경로로써 6개의 방향전환점을 가지고 있다. 그리고 웨이브 포인트로 인해 불필요한 방향전환점이 발생한 것도 확인할 수 있다. 여기서 거리만을 고려한 경로에 경로 수정 알고리즘을 적용하여 Fig. 7에서 가는 선으로 표시된 경로로 수정할 수 있다. 가는 선으로 표시된 경로를 확인해보면 기존의 방향전환점이 6개에서 3개로 줄어든 것을 확인할 수 있다.

마지막으로 Fig. 8은 지금까지 설명한 경로 수정 알고리즘을 정리하여 순서도로 나타낸 그림이다.

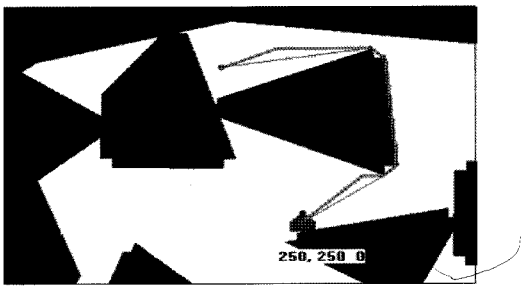

Fig. 7 Example of the path amendment algorithm

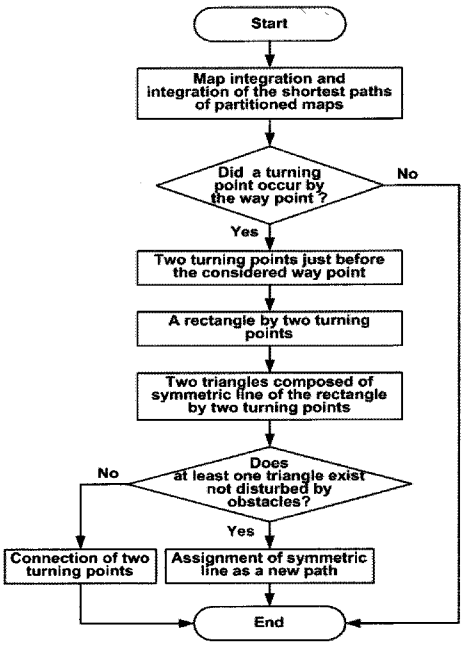

Fig. 8 Flowchart of the path amendment algorithm

5. 주행 알고리즘

주행 알고리즘은 앞에서 설명된 A* 알고리즘을 기본으로 하고 경로 수정 알고리즘을 추가하는 형태로 이루어지고 있다.

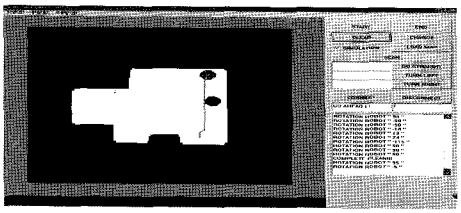

Fig. 9 Shortest path creation and traveling 1

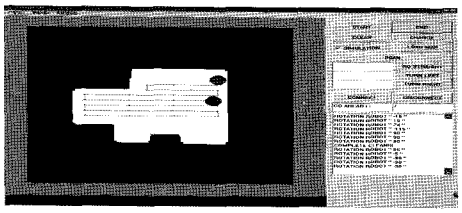

Fig. 10 Shortest path creation and traveling 2

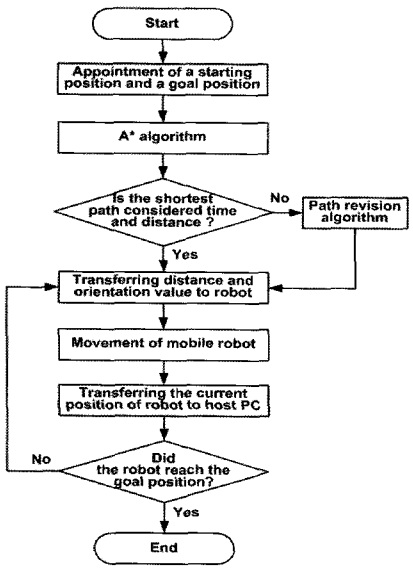

Fig. 11 Flowchart of the traveling algorithm

사용자에 의하여 혹은 이동로봇의 스스로의 필요성에 의하여 시작 지점과 목표 지점이 지정되고 나면 이 두 지점을 가지고서 A* 알고리즘을 통해 최단 경로를 생성하게 된다. 최단 경로를 생성한 후에는 생성된 경로가 시간과 거리를 모두 고려한 최단 경로인지를 점검한 뒤 그렇지 않다면 경로 수정 알고리즘을 수행하여 시간과 거리를 모두 고려한 경로로 수정하게 된다. 경로가 완성되고 나면 이동로봇은 방향전환점이 있는 지점까지의 거리 및 방향 값을 반복적으로 전달받아 이동하게 되고 방향전환점에 도착할 때마다 목표 지점 도달 여부를 확인시켜주게 된다.

Fig. 9와 Fig. 10은 주행 알고리즘이 수행되는 시뮬레이션 결과를 나타낸 것이고, Fig. 11은 이와 같은 주행 알고리즘이 수행되는 과정을 정리하여 순서도로 나타낸 그림이다.

6. 시뮬레이션 및 실험

시뮬레이션을 수행할 대상은 90°의 방향 전환점만을 가지고 있는 정사각형 형태이고 내부에는 다각형의 장애물 5개가 존재하는 다소 복잡한 맵으로서 이동로봇이 출발할 때 가장 먼저 탐색하게 되는 것은 벽이 아니라 장애물도 될 수 있는 일반적인 경우를 대상으로 하였다. 먼저 이동로봇은 맵 빌딩을 시작하기 위하여 가장 가까운 벽으로 이동하게 되는데 이것은 실제 벽이 아니라 벽과 이동로봇 사이에 존재하는 장애물중의 하나이다. 장애물을 따라 이동로봇이 이동하고 난 후 Fig. 13과 같이 장애물에 대한 폐곡선이 완성되고 나면 이동로봇도 벽이 아니라 장애물임을 확인하게 된다. 장애물 확인 후 맵 빌딩을 계속 수행하기 위해 다시 가까운 벽으로 이동하게 된다. 물론 다시 이동한 벽도 장애물일 가능성은 충분히 가지고 있다. 벽으로 이동한 후부터 다시 맵 빌딩을 시작하게 되는데, 맵 빌딩 과정 중 인덱스 확인을 통해 폐곡선의 생성 유무를 확인하는 과정, 탐색되지 않은 내부의 공간을 추가적으로 탐색하는 과정들을 거쳐 맵 전체를 탐색하고 Fig. 14와 같이 맵 빌딩을 완료하게 된다. 이렇게 맵이 완성된 후 완성된 맵을 바탕으로 하여 주행 시뮬레이션을 실시하였으며 그 결과가 Fig. 15이다.

본 논문에서는 실제 환경에서 제안한 알고리즘의

유효성을 확인하기 위하여 실험을 수행하였다. 실험에 사용된 자율로봇은 본 논문을 위하여 자체 제작한 모바일 로봇이며 전방에 4조, 좌우에 각각 2조 총 8조의 초음파 센서를 장착하여 환경에 대한 정보를 획득하고 있다.

Fig. 16은 사진과 같이 주어진 미지의 주행환경에서 로봇이 환경인식을 통한 맵 빌딩과정을 수행하고 있는 정보를 Bluetooth를 이용한 무선 RS232 통신을 이용하여 획득한 후 UI 인터페이스 프로그램으로 전송한 결과와 함께 순차적으로 나열한 것이다. 초기에 중간에 놓여 있던 로봇이 벽 쪽으로 이동하여 벽과 나란히 주행함으로써 폐루프를 구성한 후, 장애물을 포함한 내부를 탐색하는 방법으로 맵 빌딩과 주행을 동시에 수행해 가는 과정을 확인할 수 있다.

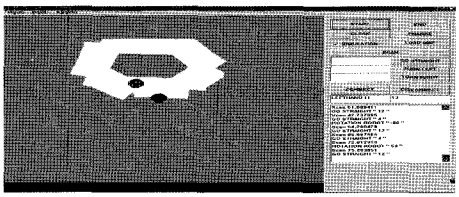

Fig. 12 an obstacle

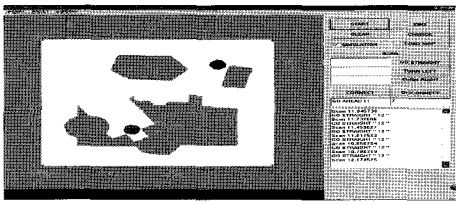

Fig. 13 an obstacle

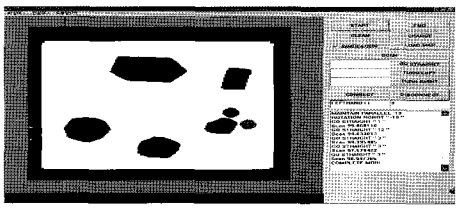

Fig. 14 the map building

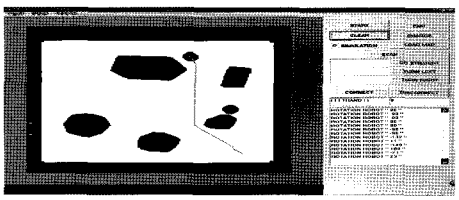

Fig. 15 Execution example of the traveling algorithm

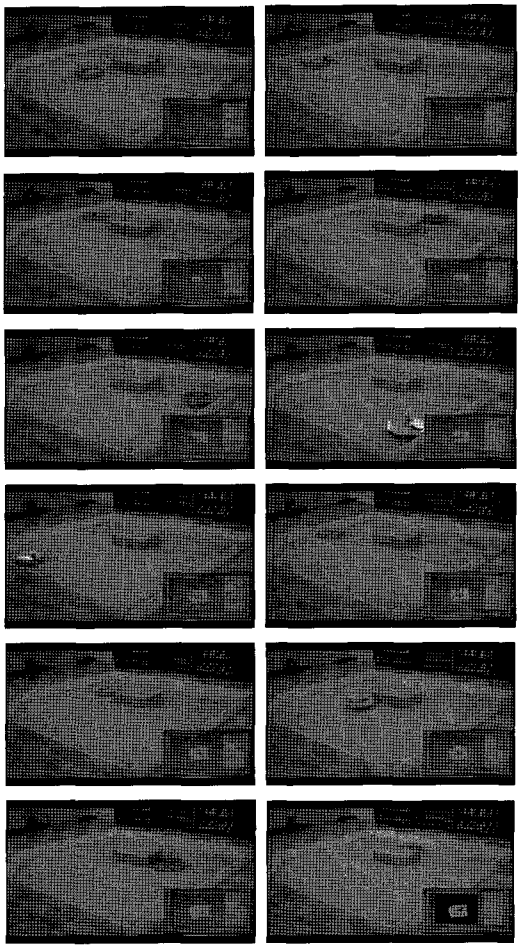

Fig. 16 Map building and traveling procedure under an actual unknown traveling environment

6. 결 론

본 논문에서는 초음파 센서를 이용하여 다양한 형태의 이동로봇에 적용할 수 있는 맵 빌딩 알고리즘을 통해 획득한 환경정보를 바탕으로 하여 A* 알고리즘과 경로 수정 알고리즘을 이용한 최단 경로를 생성함으로써 이동로봇의 주행 효율성을 높일 수 있는 주행 알고리즘을 제안하였다.

제안한 맵 빌딩 알고리즘과 주행 알고리즘을 시

뮬레이션 프로그램과 실제 주행을 통해 확인함으로써 알고리즘의 성능과 타당성을 검증하였다.

향후에는 맵 빌딩과정과 탐색과정에서 CPU의 연산시간을 보다 더 줄이면서도 효율을 보장할 수 측면에서 알고리즘을 개선하는 연구를 수행해야 할 것이다.

참고문헌

[1] 한승봉, "초음파 센서를 이용한 이동로봇의 맵 빌딩 알고리즘 및 주행 알고리즘에 관한 연구", 한국해양대학교 석사논문, 2007.

[2] 이세일, "타일 맵에서 A* 알고리즘을 이용한 유닛들의 길 찾기 방법 제안", 한국컴퓨터정보학회 논문지, 제9권 제3호, pp. 71-77, 2004.

[3] 곽상필, 최병재, 류석환, "자율이동로봇의 경로계획과 주행에 관한 연구", 한국퍼지 및 지능시스템학회 논문지, 제15권 제2호 pp. 427-430, 2005.

[4] 김범재, "자율이동로봇의 충돌회피 알고리즘 구현과 CAN을 이용한 통합화", 부산대학교 석사논문, 1999.

[5] 배성혁, 최동렵, "이동로봇의 장애물 회피를 위한 경로계획", 한국자동제어학술회의 논문집, 제1권, pp. 766-771, 1994.

[6] A. Howard and L. Kitchen, "Generating Sonar Maps in Highly Specular Environments" Proc. of the 4th International Conference on Control, Automation, Robotics and Vision, pp.637-642, 1996.

저 자 소 개

김종화(金鍾和)

1981년, 1985년, 1989년 부산대학교 학사, 석사, 박사 졸업(제어공학). 1996년-1997년, University of Wales, Cardiff 연구교수. 1990년-현재 한국해양대학교 컴퓨터제어전자통신공학부 교수.

김진규(金珍圭)

1999년, 2002년, 2005년 한국해양대 제어계측공학과 학사, 석사, 박사수료. 2003년-2004년 LG전자(주) 영상제품연구소 주임연구원. 2006년-현재 대전·충남지방중소기업청.

임재권(林在權)

2002년, 2004년 한국해양대 제어계측공학과 학사, 석사졸업.
2004-현재 한국해양대 제어계측공학과 박사과정.

한승봉(韓承奉)

2005년, 2007년 한국해양대 제어계측공학과 학사, 석사졸업(공학석사). 2007년-현재 삼성전자(주) 디지털미디어총괄 디지털미디어 연구소 연구원.