

A*를 이용한 장애물 회피

정현룡*(전남대 대학원 기계공학과), 김영배(전남대 대학원 기계공학과)

Avoidance obstacles using A* algorithm in the Eyebot

Jung, HyunRyong(Mech. Eng. Dept, CNU), Young-Bae Kim(Mech. Eng. Dept, CNU)

ABSTRACT

The A* algorithm is usually used in game programming, mainly because it is fast in finding a optimal path to goal. In this paper, This algorithm was utilized for path finding.

HMM(Histogramic In-Motion Mapping) method is used in map-building. Map is updated continuously with range data sampled by PSD sensors. From the map, A* algorithm finds a optimal path and sends subsequently the most suitable point to the Eyebot.

A* algorithm has been tested on the Eyebot in various unknown maps of unknown and proved to work well. It could escape the local minimum, also.

Key Words : A*, Mobile robot, Eyebot, OpenGL, Cell grid

1. 서론

장애물 회피(Obstacle Avoidance)는 mobile robot 시스템의 성공 여부에 있어 매우 중요한 부분이다. 모든 mobile robot은 장애물을 인식하고 정지하는 간단한 알고리즘부터, 아주 정교한 알고리즘을 탑재하여 장애물을 회피하는 것까지 아주 다양하다. 후자의 경우는 장애물의 탐색뿐만 아니라, 여러 가지 계산으로 인해서 전자보다 더욱 복잡하게 된다. A* 알고리즘은 주로 게임에서 많이 사용되는 알고리즘으로 최단거리를 찾기 위해서 사용된다. 그러나 주로 미리 알려진 맵에서 사용되기 때문에 mobile robot에 응용하기 위해서, Map-Building을 실시간으로 하는 동시에 Map에 따른 새로운 경로를 생성하는 방법을 사용하였다. 기존의 알고리즘의 경우는 local minimum에서 빠지 나오지 못하였으나 A* 알고리즘은 이러한 문제점을 해결할 수 있다. Map-Building에는 HMM(Histogramic In-Motion Mapping)을 사용하였다.

2. Other Obstacle Avoidance Methods

2.1 The Certainty Grid for Obstacle Representation

이 방법은 Carnegie-Mellon University에서 개발된 방법으로 격자 방식(grid-type world model)이다. 여기서 사용되는 world model은 certainty grid라고 하며, 초음파 센서와 같은 거리 측정에 있어서 부정확한 센서에 알맞은 방법이다. Certainty grid에서 로봇의 활동 영역은 사각 2차원 배열로 표시되며, cell이라고 하며, 각각의 cell들은 certainty value(CV)^[1]를 가지게 된다. 이 CV 값은 cell에 장애물이 존재할 가능성을 나타내게 된다. CV값은 probability function에 의해서 갱신된다. 예를 들어, 부채꼴 형태로 퍼지는 초음파 센서의 경우 거리 데이터 값이 추정되었다고 했을 때, 거리는 알 수 있으나, 물체가 정확히 어디에 있는지는 알 수 없게 된다. CMU에서는 이러한 이유로, 다음과 같은 방법을 사용하였다. Probabilistic function을 사용하여, 음과가 진행

한 축에 있는 cell들의 CV값만을 갱신하였다. 그리고 off-line을 통해서 robot의 path를 결정하였다.

2.2 Potential Field Methods

Potential Field Methods는 Khatib⁽¹⁾가 제안한 방법으로, 로봇이 움직이는 동안 장애물은 척력(repulsive force)을, 목표는 인력(attractive force)을 로봇에게 띄게 된다. 그리고 resultant force R은 장애물의 척력과 목표의 인력의 합으로 나타내게 된다.

Krogh⁽²⁾는 이 algorithm에 로봇의 인접한 부분에 장애물이 있을 때의 로봇의 속도에 대해서 추가하였다. Krogh와 Thorpe⁽³⁾는 global path planning과 local planning을 합친 방법을 제안하였다.

이러한 방법들의 공통점은 미리 알고 있고, 이미 규정된 map이라는 것이다. Brook⁽⁷⁾은 조음과 센서를 이용하였으며, 읽어 들인 값들을 repulsive force로 간주하여, 그 값들의 합이 특정 threshold 값을 넘어 가면, 로봇이 경지하고 resultant force 방향으로 로봇의 방향을 돌리고 진행을 계속하는 방법을 사용하였다. Potential field method의 단점은 로봇이 좁은 복도를 통과할 때, 경로가 진동하는 현상을 보인다.

2.3 The VFF(Virtual Field Force) Concept

VFF⁽⁸⁾는 장애물을 histogram grid를 이용해서 표시한다. CMU의 CV와 같이 histogram grid의 (i, j) cell은 certainty value, $c_{i,j}$ 를 갖는다. Histogram grid는 빈들이지고, 갱신되는 방법에 있어서 certainty grid와는 다르다. CMU의 방법은 조음과에 의해 읽어 들인 값의 영역에 있는 모든 cell에 영향을 주기 때문에, 계산 양이 많아져, on-board computer에서는 실시간 실행이 어려워진다. 하지만, 이 방법은 센서에 의해서 거리 데이터가 갱신될 때 센서가 진행하는 방향의 축에 있는 오직 하나의 cell만 증가시키게 된다. 그리고 potential field 개념을 histogram grid에 적용한다. 방법은 다음과 같다. 현재 로봇의 위치에서 로봇을 중심으로 33×33 cell들을 active window라고 지정하고, 그 안에 위치한 cell들이 가지고 있는 certainty value에 비례한 값의 합(F_r)이, robot에 repulsive force로 정의된다. 그리고 target으로의 attractive force(F_a)와의 합 vector(R)를 구하게 된다.

VFF는 대부분의 환경에서 잘 동작하지만, 다음과 같은 경우에는 그렇지 못하다.

첫 번째는, 문을 통과하지 못한다는 것이다. 문의 양쪽 밑에서 repulsive force로 인해서 로봇이 문으로부터 멀어지게 된다.

두 번째는, histogram grid의 특성으로 인해서 발생하는 것으로, 로봇이 움직이는 동안 repulsive force를 계속해서 갱신하게 되는데, 로봇이 다음 cell로 이동하므로 R이 크게 변하는 현상이 발생한다. 예를 들어, cell(m, n)에 의해서 repulsive force가 발생하고, 로봇의 VCP(Vehicle Center Point)이 (m, n+6)이라고 하자. 하나의 cell 크기는 $10\text{cm} \times 10\text{cm}$ 이다. cell(m, n)에 의해서 로봇에 영향을 미치는 repulsive force의 크기는

$|F_{m,n}| = k/0.6^2 = 2.8k$ 이다. 로봇이 (m, n+5)로 이동한 후의 repulsive force의 크기는

$|F_{m,n}| = k/0.5^2 = 4k$ 이다. 변화율이 42%나 되어,

로봇의 steering control에 있어서 fluctuation이 일어나게 된다. 이러한 현상은 로봇이 좁은 통로를 이동할 때 발생하게 된다. 로봇이 통로의 중심선을 따라가는 경우는 안정적이다. 중심으로부터 한쪽으로 기울었을 때는 한쪽 벽으로부터 강한 repulsive force를 받게 되어 로봇의 경로가 불안정하게 된다.

3. 이론

3.1 3D-Map Building

로봇이 목표점까지 도달하기 위한 경로가 나오기 전에 선행되어야 하는 것이 map-building이다. 이번 논문에서는 map-building을 위해서 HMM(Histogramic In-Motion Grid)을 사용하였다. HMM은 조음과 센서를 사용한다는 가정으로 시작한 algorithm이다. 하지만, 본 논문에서는 PSD 센서를 사용하기 때문에 기본 가정은 수정하도록 하였다. 즉, 조음과의 경우는 장애물이 진행한 영역의 오른쪽에 있거나, 왼쪽에 있거나 측정된 거리는 간격된다. 그래서 정확히 장애물이 어디에 있는지는 알 수 없다. 그러나 PSD의 경우는 조음과처럼 방향성에 있어서 부정확하지 않다는 것이다. 하지만, 노이즈나 기타 읽기 오류 등으로 발생하는 error를 줄이기 위해서 HMM에서 사용되는 GRO Mask를 사용하도록 하였다(Fig.1).

3.2 A* algorithm

A* algorithm⁽⁹⁾은 목표지까지 최적인 path를 찾도록 고안된 알고리즘으로 움직이는 동안 cost를 계

산해서 다음 목적지를 결정한다. 그리고 2차원 grid-based world 에 적합하다.

A* algorithm이 수행되는 순서는 다음과 같다.

- ① Create OPEN and CLOSED list
- ② If OPEN is empty, exit with failure
- ③ Select the first node on OPEN, remove it from OPEN, and put it on CLOSED.
Call this node n
- ④ Investigate the near 8 nodes
- ⑤ Select the node of the lowest cost
- ⑥ If n is goal node, exit successfully.

초기의 OPEN 리스트에 들어가는 셀 들은 맵의 모든 셀 들이 들어갈게 된다. 그리고 인접한 8개의 셀 들을 검색하여 가장 작은 cost를 가지는 셀을 선택하고, OPEN 리스트에서는 제거한다. 그리고 선택된 셀을 CLOSED 리스트로 옮긴다. 그리고 선택된 셀들 중에 가장 작은 cost를 가지는 값들을 Successor로 저장한다. 그리고 이러한 방법을 계속해서 반복한다.

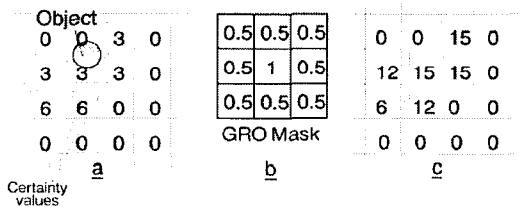


Fig. 1 (a). 로봇이 움직이는 동안 하나의 장애물에 대한 CV값
(b). 3×3 mask for growth rate operator(GRO)
(c). GRO를 적용한 상태

Cost를 계산하는 방법은 다음과 같다.

$$f = g + h$$

여기서,

- f = total cost
- g = The sum of all the costs it took to get here
- h = Heuristic Function

4. 실험

실험에서 사용된 로봇은 UWA(University of Western Australia)의 CHPS(Centre for

Intelligent Information Processing System) Lab에서 개발된 Eyebot을 사용하였다. Eyebot은 two-wheel이며, 3개의 PSD sensor, CMOS camera, digital compass, LCD 그리고 RF module로 이루어져 있다. 실험은 로봇에서 3개의 PSD센서로부터 읽어들이는 거리데이터와 현재의 위치 데이터를 computer로 보내게 되며, 실시간으로 HMM을 이용해서 map-building을 하고, A*에 의해서 나온 새로운 경로를 로봇으로 전송하게 된다.

Computer에서는 로봇으로부터 들어오는 거리데이터와 위치데이터를 바탕으로 OpenGL을 이용해서 3차원으로 map-building을 하게된다. 그리고 CV값이 9이상이면, 그 cell이 속한 A*의 grid에 장애물이 존재하는 것으로 인식하도록 하였다.

로봇에서는 computer에서 받은 다음 위치의 grid 데이터로부터 현재 로봇의 위치에서 목표점까지 이루는 각을 계산하고 회전을 한 후 목표로 이동하게 된다.

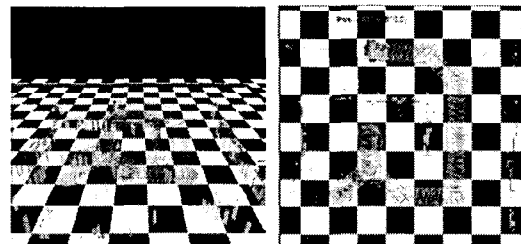


Fig. 2 Result of Experiment(Local Minimum)

Fig.2는 local minimum에 대한 테스트이다. 기존의 path finding 이론의 경우는 local minimum에서 빠지 나오지 못하였다. 그래서 local minimum에 빠졌을 경우가 발생하면, 벽을 따라서 빠지 나오는 등의 추가적인 방법을 사용하였다. 하지만, A* algorithm의 경우는 local minimum을 빠지 나오는 것을 알 수 있다.

5. 결론

실험에서 보여지듯이 A* algorithm은 로봇의 Path Finding에 있어서 아주 잘 동작함을 알 수 있었다. 하지만, A* algorithm에서 항상 존재되어야 하는 것은 먼저 맵이 정확해야 한다는 것이다. 만약 맵이 잘못 그려졌을 경우는 잘못된 경로를 생성하게 된다. 그러므로 먼저 정확한 Map Building을 하는 것이 중요하다.

참고문헌

1. Elfes, A, "Using Occupancy Grids for Mobile Robot Perception and Navigation." Computer Magazine, June, pp. 46-57 1989
2. Moravec, H. P. and Elfes, A., "High Resolution Maps from Wide Angle Sonar." Proceedings of the IEEE Conference on Robotics and Automation, pp.116-121, 1985
3. Moravec, H. P., "Sensor Fusion in Certainty Grids for Mobile Robots." AI Magazine, Summer, pp.61-74, 1988.
4. Khatib, O., "Real-Time Obstacle Avoidance for Manipulators and Mobile Robots." IEEE International Conference on Robotics and Automation, pp.500-505, 1985
5. Krogh, B. H., "A Generalized Potential Field Approach to Obstacle Avoidance Control." International Robotics Research Conference, Bethlehem, Pennsylvania, August, 1984.
6. Krogh, B.H. and Thorpe, C. E., "Integrated Path Planning and Dynamic Steering Control for Autonomous Vehicles."
7. Brooks, R. A., "A Robust Layered Control System for a Mobile Robot." IEEE Journal of Robotics and Automation, Vol. RA-2, No. 1, 1986, pp. 14-23.
8. Borenstein, J. and Koren, Y., "Real-time Obstacle Avoidance for Fast Mobile Robots." IEEE Transactions on Systems, Man, and Cybernetics, Vol. 19, No. 5, sept/Oct 1989, pp. 1179-1187.
9. Steve Rabin, "AI Game Programming Wisdom". Charles River Media, Inc, pp. 103-146.