일반화 가시화 그래프의 구현[1)]

안진영[0] 유견아
덕성여자대학교 전산학과
{jyahn[0], kyeonah}@namhae.duksung.ac.kr

The Implementation of Generalized Visibility Graph

Jinyoung Ahn[0] Kyeonah Yu
Dept. of computer science, Duksung Women's University

요 약

가시화 그래프(visibility graph, Vgraph)는 로봇의 경로를 계획할 때, 최적의 경로를 구하기 위해 널리 이용되는 지도 접근 방식중의 하나이다. 원래 Vgraph는 다각형으로 모델링된 로봇이 다각형 환경의 평면상에서 움직일 때 움직이는 로봇을 점으로 환산한 환경인 형상공간(configuration space, C-공간)에서 정의되었는데 이를 원형 로봇 혹은 일반화 다각형(generalized polygon) 환경으로 확장한 것이 일반화 가시화 그래프(GVgraph)이다. 본 논문에서는 기존의 다각형 환경에서 정의된 Vgraph를 형성하는 알고리즘과 동일한 시간복잡도로 GVgraph를 구현하는 알고리즘을 소개하고 미세 운동계획(fine motion planning)에 응용하는 예를 보여준다.

1. 서론

로봇 경로계획 알고리즘의 대부분은 선분이나 다각형을 다루도록 개발되어 있어 로봇이 작업하는 환경의 다양한 물체들을 선분만으로 모델링한다. 로봇 작업환경에 있는 곡선을 포함한 장애물들은 여러 작은 선분으로 근사 치환(approximate)하여 기존의 알고리즘을 적용하는 것이 일반적이다. 하지만 근사 치환은 정확한 답을 보장하지 못하며 무수히 많은 선분으로 대치함으로써 계산량을 증가시키는 등의 단점이 있다[1]. 그러므로 기존의 알고리즘들을 원의 호까지 포함하는 일반화 다각형을 다룰 수 있도록 확대한다면 로봇 작업환경을 정확하게 모델링할 수 있어 보다 정확한 경로 계획을 가능하게 할 것이다. 가시화 그래프(Vgraph)는 2차원 C-공간에서 다각형 장애물을 대상으로 만들어지는 그래프이며 최적 경로를 찾기 위하여 가장 널리 사용되는 방법중의 하나이다[2]. Vgraph는 최단 경로를 계획하는 문제이 주목적이긴 하지만 자유 공간에서 로봇의 운동방향을 이산화할 때, 동일한 토폴로지를 갖는 구간으로 나누기 위해 임계방향(critical orientation)을 찾는 문제에도 이용되는 등의 다른 목적으로 이용되기도 한다[3].

일반화 가시화 그래프(GVgraph)는 논문 [4]에서 정의되었으며 이 논문에서는 볼록 일반화 다각형인 경우에 호를 처리하는 아이디어를 제시하였다. [2]에서는 지역적 비볼록(locally non-convex) 일반화 다각형으로 확장하고 [4]의 방법을 이용할 수 있다고 언급하였으나 모두 실제 구현한 예는 없다. 본 논문에서는 기존의 Vgraph를 다각형뿐만 아니라 원의 호까지 포함하는 일반화 다각형 환경의 GVgraph로 확장하는 방법을 소개하고 구현한다. GVgraph를 생성하는 알고리즘은 기존의 Vgraph 알고리즘에 비해 시간복잡도(time complexity)가 나빠지지 않음을 보이며 구현된 알고리즘을 앞에서 응용 분야에 적용한 예를 보인다.

2. 일반화 가시화 그래프

이 장에서는 기존의 Vgraph를 사용하는 방법에 대해 살펴보고 원의 호를 처리하기 위해 어떻게 수정되는지를 설명한다. 또한 수정된 알고리즘의 시간 복잡도에 대하여 살펴본다.

1) 이 연구는 한국과학재단의 목적기초사업(과제번호 R04-2001-000-00048-0)의 연구비지원에 의해 수행되었음.

2.1 가시화 그래프

Vgraph는 장애물의 꼭지점들을 노드로 하고 꼭지점들을 연결한 선분중에 장애물과 겹치지 않는 선분을 링크로 하는 무방향(non-directed) 그래프로 정의된다. Vgraph는 형상공간(configuration space, C-공간)에서 정의된다. C-공간이란 움직이는 로봇을 점으로 환산한 공간을 말하며[5], 앞으로 본 논문에서 언급하는 장애물은 환산된 C-공간 장애물이다. C-공간에서 로봇의 경로를 계획하는 방법은 여러 가지가 있으나 Vgraph는 최단거리를 찾는데 효과적으로 이용된다[2]. Vgraph를 이용해 최단경로를 찾는 과정은 크게 두 단계로 나누어 볼 수 있다. 첫 번째는 로봇의 시작형상과 목표형상, 그리고 장애물로부터 Vgraph를 생성하는 단계이고 두 번째는 생성된 Vgraph를 이용하여 최단거리를 탐색하는 단계이다.

Vgraph를 생성하는 첫 번째 단계에서는 노드와 링크를 생성한다. 이를 위해 먼저 C-공간으로 형상화된 여러 장애물과 시작형상, 그리고 목적형상이 배치되어 있어야 하고 이때 Vgraph의 노드는 시작형상과 목표형상 그리고 장애물의 꼭지점으로 이루어진다. 이렇게 생성된 노드 중 중 임의의 두 개의 노드를 연결한 선분과 장애물이 교차하지 않는다면, 즉 연결한 선분이 Cfree상에 존재한다면 이는 Vgraph의 링크가 된다. 그림 1은 이렇게 생성된 Vgraph이다.

최단경로를 찾기위한 두 번째 단계에서는 여러 가지 탐색 알고리즘을 사용할 수 있다. 특히, 휴리스틱(heuristic)값으로 유클리드 거리를 취하는 A* 알고리즘의 경우 시간 복잡도가 장애물 꼭지점의 개수에 대해 선형(linear)이기 때문에 주로 이용된다. 한편 필요에 따라 노드와 링크의 개수가 줄어든 축소(Reduced) Vgraph를 생성할 수도 있다. 하나의 꼭지점에서 연결된 경로중, 긴 경로는 최단경로를 탐색하는데 고려되지 않는다. 이러한 전제 하에 긴 경로를 제거하므로 축소 Vgraph의 생성이 가능하다[2]. Vgraph의 크기가 작을수록 유리한 응용에 이용될 수 있다.

2.2 일반화 가시화 그래프

일반화 다각형은 원의 호와 다각형으로 이루어진 도형군이다. GVgraph는 C-공간상에 시작점, 끝점과 함께 일반화 다각형으로 이루어진 장애물들이 배치되어 있는 경우에 생성된다. 이때

장애물을 이루고 있는 선분의 끝점뿐만 아니라 호의 끝점 또한 꼭지점(vertex)이라 한다[4].

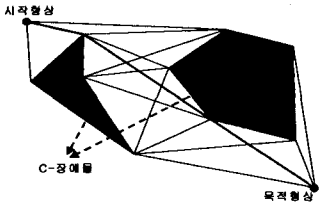
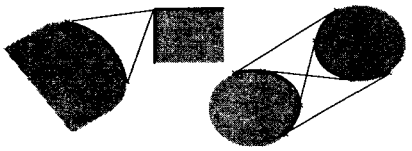


그림 1. Visibility Graph의 예

GVgraph를 생성하기 위해 노드와 링크를 정해주어야 한다. 이를 위해 다루어주어야 하는 경우는 다음과 같다.
①꼭지점과 꼭지점
②호와 꼭지점
③호와 호

일반화 다각형에서는 호를 포함하기 때문에 Vgraph에 비해 특수한 경우를 포함한다. 먼저 ①의 경우일 때 Vgarph와 마찬가지로 노드에는 시작형상, 목적형상, 장애물의 꼭지점을 포함한다. 그리고 꼭지점을 연결한 선분 중에 링크가 장애물과 교차하지 않는 선분이 링크가 된다. 호의 경우, 선분과는 다른 복잡한 성질을 갖기 때문에[6] 호의 끝점들에 대해 다른 꼭지점이나 호의 가시(visible)여부를 결정하는 것의 의미가 없다. 그러므로 ②,③의 경우에는 호의 가시성(visibility)을 따져주기 위해 가상점(fictitious point)을 생성한다. ②의 경우 노드로 정해진 꼭지점과 호가 접하는 점이 가상점이다. 그림 2(a)에서처럼 꼭지점에서 하나의 호에 최대 2개의 가상점이 생성된다. ③의 경우에는 하나의 호에 대해 외접점과 내접점이 최대 2개씩 그러므로 총 4개의 가상점이 생길 수 있음을 이는 그림 2(b)에서 볼 수 있다. 즉, GVgraph의 노드에는 가상점이 추가되고 링크에는 가상점을 생성한 선분과 호 위에 추가된 두 개의 이웃한 가상점을 연결한 부분이 추가된다. 이렇게 생성된 GVgraph의 예는 그림 3과 같다.



(a) 꼭지점과 호 (b) 호와 호
그림 2. 가상점이 생기는 경우

GVgraph와 Vgraph는 생성과정에 차이가 있을 뿐 이후의 알고리즘 과정은 동일하다. 즉, 생성된 GVgraph도 탐색 알고리즘을 이용해 최단경로를 탐색할 수 있다.

2.3 수정된 방법의 시간 복잡도

Vgraph를 형성하기 위한 장애물들의 꼭지점의 수를 n이라 할 때, 임의의 두 개의 꼭지점을 연결한 선분이 교차하는지의 여부를 판별해야 한다. n개의 꼭지점 중에서 두 개의 꼭지점을 선택하는 경우의 수는 ${}_nC_2$ 이다. 장애물을 구성하는 선분의 수는 장애물 꼭지점의 수와 같으므로 선택된 꼭지점을 연결한 선분이 장애물을 구성하는 선분과 교차하는지를 판별하기 위해서는 $O(n^3)$의 시간 복잡도가 요구된다.
GVgraph는 각 호에 대해 생기는 가상점으로 인해 노드의 수는 $O(n^2)$으로 늘어난다. 하지만 임의의 호에 늘어난 가상점은 호와 선분 호와 호사이의 링크를 구하는데 영향을 주지 않는다. 그러므로 GVgraph를 생성하는 시간복잡도는 Vgraph의 시간복잡도에 비해 변함이 없다. 단, 생성된 그래프의 노드수는 늘어나므로 최단경로를 찾는 탐색 알고리즘에 따라 탐색시간은 달라질 수 있다.

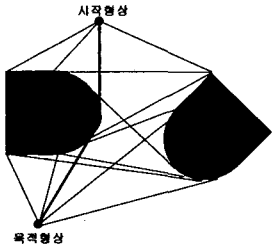


그림 3. Generalized Visibility Graph의 예

3. 구현 및 결과

Vgraph를 생성하여 최소 경로를 탐색하는 알고리즘은 Pentium Ⅳ플랫폼에서 Visual C++를 이용하여 구현하였다. 이 장에서는 구현된 알고리즘이 일반화 다각형으로 모델링된 환경에서 다각형 혹은 원형 로봇을 위한 최단 경로를 계획하는 일 및 부품을 고정하는 고정쇠 적재 문제에서 부품의 운동 방향을 이산화하는 문제에 응용할 수 있음을 보인다.

3.1 알고리즘의 구현

본 논문에서는 일반화 다각형의 동질적인 데이터 표현(homogeneous data representation)을 제공하기 위해 호와 꼭지점을 도형의 모서리라고 정의하고 일반화 다각형을 꼭지점 리스트로 표현하는 것과 마찬가지로 일반화 다각형을 모서리 리스트로 표현한다. 모서리를 시작점, 중점, 끝점의 세 점으로 구성한 순서집합을 택하여 표현하면 모서리가 꼭지점인 경우는 시작점과 끝점이 같은 특수한 경우로 표현된다. 그림 4는 5개의 모서리로 구성된 일반화 다각형의 표현 예를 보여준다.

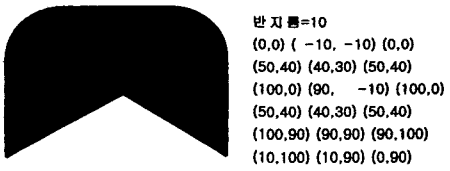


그림 4. 일반화 다각형의 모서리 리스트 표현

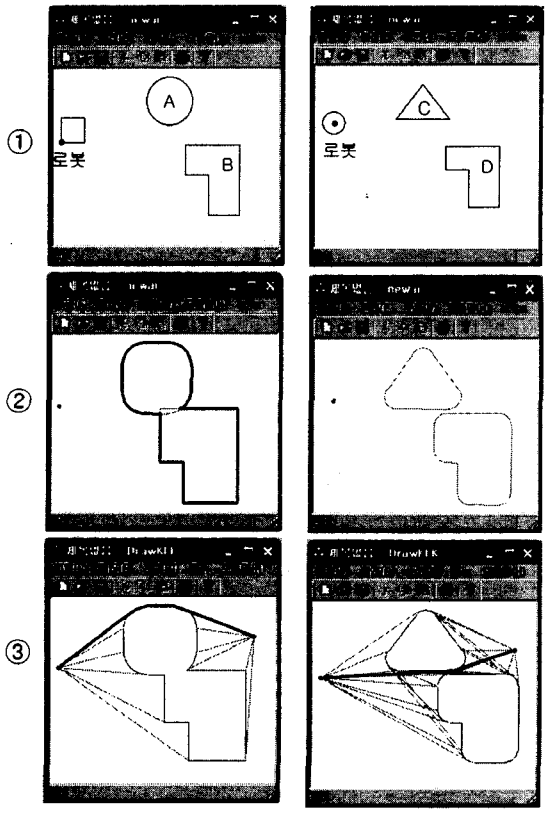
이러한 자료구조로 표현되면 장애물이 n개의 모서리로 이루어져 있는 경우, ${}_nC_2$모서리 쌍에 대해 2.2절에서 설명한 방법으로 구현된다.
① 꼭지점과 꼭지점 : 선택된 모서리가 둘다 꼭지점인 경우 리스트의 첫 번째 좌표를 연결하여 다른 선분과의 교차여부를 판별해 링크를 구한다. 이때 Vgraph에서는 고려하지 않았던 직선과 호의 교차여부를 판별하여야 한다. 이를 위해 [6]에서 소개된 일반화 다각형을 위해 수정된 plane-sweep 알고리즘을 이용하여 판별할 수 있다.
② 호와 꼭지점 : 꼭지점과 호의 접선을 이용해 가상점을 구할 수 있는데, 이를 위의 리스트의 마지막 모서리에 대하여 구현한다면 원의 중점이 $(10, 90)$이므로 원의 방정식 $(x-10)^2+(y-90)^2=10^2$이 된다. 그러면 꼭지점을 지나고 중심에서 반지름인 10만큼의 거리를 갖는 직선의 식을 구하여 가상점의

후보를 구할 수 있다. 후보 가상점 중 호 위의 점은 노드에 추가된다.
③ 호와 호 : 호와 호의 접선을 구하기 위해서는 호에 대한 원의 방정식을 ②경우와 같이 구하고 두 원의 식에 모두 접하는 직선의 식을 구한다. 구해진 식을 이용해 접선을 구한 후 가상점의 후보를 구할 수 있다. ②의 경우에서 처럼 후보 가상점 중 호 위의 점이 노드에 추가된다.

②,③의 경우 가상점을 만든 연결선이 링크가 되므로 호에 대한 연결선과 다른 선분들과의 유클리드 거리를 휴리스틱값으로 갖는 A* 알고리즘을 이용해 최단경로를 탐색한다.

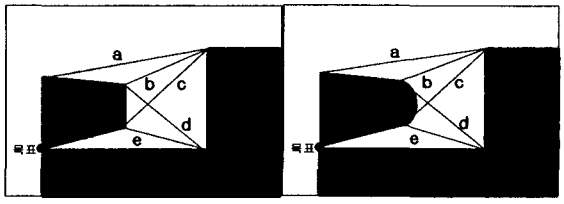
3.2 실행 결과

그림 5는 원형 및 다각형 장애물이 섞여 있는 작업 환경에서 다각형과 원형 로봇이 각각 움직이는 경우를 C-공간 장애물로 변환한 뒤, GVgraph를 생성하고 최단경로를 찾은 예를 보여준다. C-공간 표현의 의미를 강조하기 위해 (a)의 다각형 로봇은 장애물 A와 B 사이를 통과하지 못하는 크기로 (b)의 C와 D사이를 통과할 수 있는 크기로 가정하였다. 작업 환경에서 구분



(a) 다각형 로봇 (b) 원형 로봇
그림 5. 실행결과

하기 어려운 차이가 C-공간 환경에서는 분명하게 드러나는 예이다. 그림 5의 ①은 작업환경을 묘사하며 ②는 ①에 대한 C-공간에 해당하며 ③은 앞 절에서 설명한 방법에 의해 생성된 GVgraph이다. 굵게 표시된 선이 A* 알고리즘에 의해 찾아낸 최단 경로이다.



(a) Vgraph (b) GVgraph
그림 6. 임계방향 찾기에 응용된 예

최단경로를 찾는 문제의 경우에는 2.1절에서 소개한 축소 Vgraph를 이용하면 되는데 부품의 임계 방향을 구하는 문제의 경우에는 이와 다르다. 그림 6(a)는 C-공간 장애물의 일부를 표시한 것인데 Vgraph의 링크는 최단거리를 찾는데 이용되는 것이 아니라 부품운동의 임계방향을 찾는데 사용된다. 예를 들어 로봇이 링크 a와 b 방향사이로 움직이는 것과 링크 b와 c 사이로 움직이는 것은 목표로의 경로의 토폴로지가 달라진다. 그러므로 a,b,c 등의 방향을 임계(critical) 방향이라고 하는데 이 경우는 최단경로를 구하는 경우와 달라 축소 Vgraph를 구하면 안된다. 이를 일반화다각형으로 확대한 것이 (b)인데 고정쇠 적재 문제가 이 경우의 대표적인 예라고 할 수 있다[6].

4. 결론

본 논문에서는 로봇 경로계획문제에서 최단거리를 찾는데 널리 이용되고 있는 Vgraph 알고리즘을 일반화 다각형 환경으로 확장하고 구현하였다. 이는 일반화 다각형으로 이루어진 환경에서도 로봇이 최단경로를 찾아 움직일 수 있게 한다. 좀 더 다양한 환경에서 로봇이 경로계획을 할 수 있도록 하는데 한걸음 나아간 것이다. 또한 시간복잡도 측면에서도 다각형 환경에서 보다 나빠지지 않음을 알 수 있었다.
한편 sweep-line알고리즘을 이용하면 Vgraph를 생성하기 위한 시간복잡도는 $O(n^2 logn)$으로 향상될 수 있음이 알려져 있다[2]. 하지만 GVgraph에서 호는 어떠한 축에 대한 각이 일정하지 않기 때문에 이 알고리즘의 적용이 어렵다. 이러한 경우 일반화 다각형 환경에서 요구되는 시간복잡도를 줄이기 위해서는 새로운 알고리즘을 개발하여야 한다. 즉, 일반화 다각형의 일반적인 성질들이 더 많이 연구되고 정의되어야 로보틱스 분야에 대해 효과적인 확장이 가능하다.

[참고문헌]

[1] D. Halperin, L. Kavraki, and J.-C. Latombe, "Robot algorithms", *CRC Algorithms and Theory of Computation Handbook,* M. Atallah (editor), CRC Press, 1999
[2] J.C. Latombe, *Robot Motion Planning*, Kluwer Academic Publishers, pp153-199, 1991
[3] B.R. Donald, "The Complexity of Planar Compliant Motion Planning Under Uncertainty," *Proceedings of the Fourth ACM Symposium on Computational Geometry*, pp309-318, 1988
[4] J.P. Laumond, "Obstacle growing in a nonpolygonal world", *Inform. Proc. Letter*, vol 25(1), pp41-50, 1987
[5] T. Lozano-Perz, "Spatial Planning: A Configuration Space Approach," *IEEE Transactions on Computers*, C-32(2), 108-120, 1983
[6] 유견아 안진영, 일반화 다각형을 위한 plane-sweep 알고리즘의 구현, *한국정보과학회 봄 학술발표논문집*, pp.691-693, 2002