수정된 Generalized Voronoi Graph 를 사용한 차량형 로봇의 센서기반 주행 알고리즘 개발

Development of Sensor-based Navigation Algorithm for Car-like Robot Using Generalized Voronoi Graph

*원천, 이지영, #한창수

*Quan Yuan, Ji Yeong Lee, #Changsoo Han(cshan@hanyang.ac.kr)

한양대학교 기계공학과

Key words : Sensor-based, Non-holonomic, Car-like mobile robot, Generalized Voronoi Graph

1. Introduction

Sensor based planning is necessary for the realistic deployment of the robots into unknown spaces. In the past, the sensor based planners [1] were limited to robots modeled as point robots operating in the plane. This paper proposed a sensor-based approach for car-like mobile robot to make it access and trace the GVG.

In this paper, we present a sensor based algorithm for car-like mobile robot based on GVG theory. But the car-like mobile robot is limited by the minimum turning radius, so the GVG algorithm cannot be implemented to the car-like mobile robot directly. We proposed an algorithm enables the car-like mobile robot can follow the GVG as much as possible.

2. Car-like Mobile Robot Model

The configuration of a car-like robot, sketched in Fig.1, at the instant t is completely defined in $C=R^2\times S^1$ expressed by $\boldsymbol{q}=[x_r, yr, \theta_r]^T$, by the position (x_r, yr) of the reference point and the heading direction θ_r of the robot.

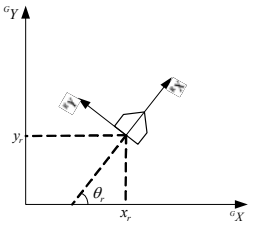


Fig.1 Kinematic model of the mobile robot

The model of the car-like robot which we will refer in this paper is described by the control system.

$$\dot{q}=\begin{bmatrix}\cos\theta_r\\ \sin\theta_r\\ 0\end{bmatrix}v_{ref}+\begin{bmatrix}0\\ 0\\ 1\end{bmatrix}\omega \qquad (1)$$

where v_{ref} is the linear velocity, ω is the angular velocity. We express the control input vector as $u=[v_{ref}\ \ \omega]$ T.

3. Sensor-based Algorithm to Generate GVG

The concept of the navigation which is proposed in this paper is based on the GVG, and combined with the non-holonomic [3] constraints. GVG is made up of curves and vertexes, the robot can't follow exactly. In this approach, GVG is incrementally constructed. When the robot is at the meet point, we present an algorithm enables the car-like robot go through the meet point tracing along the GVG as much as possible. Backward motion is considered to avoid collision.

Accessibility: Finding a path onto the GVG. We define two vectors: one for car-like robot to access the GVG and the other for car-like robot to turn to the tangent direction of GVG. As shown in eq.(2), $\overrightarrow{p_ig_i}$ is the vector combined with two vectors:

$$\overrightarrow{p_ig_i}=sign\frac{\nabla d_1+\nabla d_2}{\left\|\nabla d_1+\nabla d_2\right\|}+\nabla d_1(d_2-d_1) \qquad (2)$$

After we get the $\overrightarrow{p_ig_i}$ vector, we implement the "follow the carrot" [2] algorithm. We can calculate the error angle e_θ between the robot and the $\overrightarrow{p_ig_i}$ vector, and then it will calculate the velocity and

turning radius. Actually, the instant goal g_i is calculated instantly. Therefore we can access the GVG incrementally.

Car-like robot is limited by the minimum turning radius. When it accessed the GVG it may collide with obstacles if it goes forward. So here we mention the backward motion to avoid collision.

Traceability: traversing the roadmap to vicinity of the goal. The car-like mobile robot moves along the tangent direction of GVG edge. Here we proposed a method to decide the tangent direction of the GVG edge.

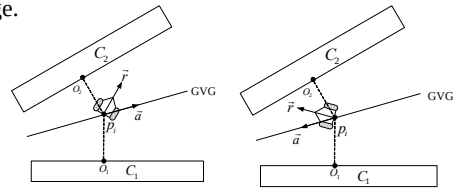


Fig.2 tangent direction of the GVG

$\vec{r}$ is the unit vector which represents the robot orientation. As shown in the Fig.2, we choose the $\vec{a}$ as the tangent direction of GVG, which is minimum difference with robot orientation.

Meet point motion: Meet point is multi-equidistant, and the distance to the obstacle is the minimum distance. We can get the information of the distances to obstacles: d_1, d_2, and d_3. And the tangent direction of GVG edge at the meet point θ_{eij}, then we can get the difference between robot orientation and GVG edge, which denoted as $\Delta\theta_{ij}$, we pick up the minimum denoted as $\Delta\theta$. The meet point motion algorithm is shown below:

Car-like robot motion algorithm at Meet point

i=1

1. **for ;**
2. **if** $L=R_{min}tan(\Delta\theta/2i)<d_1$
3. **for k=1:i**
4. **if** $\Delta\theta>0$

go backward for $R_{min}tan(\Delta\theta/2i)$, and turn right with minimum turning radius for $\Delta\theta/i$, and then go backward for $R_{min}tan(\Delta\theta/2i)$;

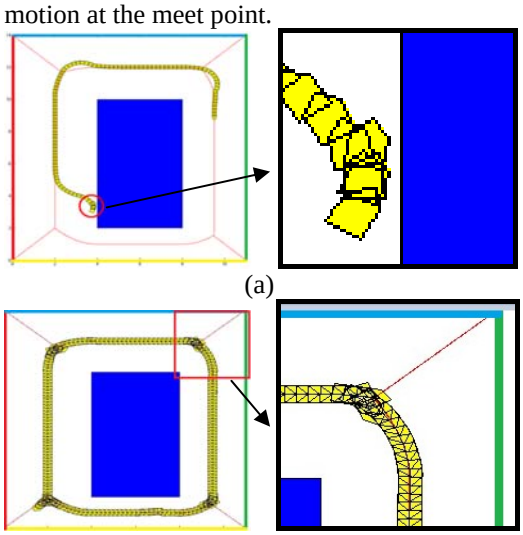
5. **else**

go backward for $R_{min}tan(\Delta\theta/2i)$, and turn left with minimum turning radius for$\Delta\theta/i$, and then go backward for $R_{min}tan(\Delta\theta/2i)$;

6. **endif**
7. **endfor**
8. **else**
9. i++;
10. **endif**
11. **endfor**

Simulation results:

As shown in Fig.3(a), we implement the accessibility and traceability algorithm to the robot, we can see it goes backward to avoid collision and then access trace the GVG edge successfully. From Fig.3(b),we can see the robot motion at the meet point.



(a)

(b)

Fig.3 Simulation results

4. Conclusion

From the simulation result, we can see the algorithm proposed in this paper make the car-like robot access and trace the GVG in unknown environment without collision successfully. Our future work will mention the departability.

Reference

1. H.Choset and J.Burdick,"sensor-based motion Planning II: Incremental Construction of Generalized Voronoi Graph.etc
2. Thomas Hellström, Ola Ringdahl. Autonomous Path Tracking Using Recorded Orientation and Steering Commands.etc
3. Huifang Wang, Yangzhou Chen, Soueres P. An Efficient Geometric Algorithm to Compute Time-optimal trajectories for a Car-like Robot. 2007 46th IEEE conference on Decision and control.