Voronoi Diagrams and Nearest-Neighbor Search

Gabriel Ewing
Department of Electrical Engineering
and Computer Science
Case Western Reserve University
Cleveland, OH
gre5@case.edu

***Abstract*—The abstract goes here.**

I. Introduction

Robotic motion planning is the problem of finding a path from the starting configuration of a robot to some goal configuration that does not collide with any obstacles. In the case where the positions of the obstacles are known *a priori* but the start and goal configurations are not, it can be useful to create a *roadmap* of the configuration space. Roadmaps demonstrate connected edges through the free space such that a robot can easily move to some edge from a starting configuration, and easily move to the goal configuration from some edge. If there is a path between the two edges in free space, the roadmap should have a path between them.

A. Voronoi Diagrams

Voronoi diagrams are a class of roadmap. The basic definition of a Voronoi diagram is that its edges represent the points that are equidistant from the closest obstacles to that point. This means that each edge represents the path that maintains the maximal separation from two obstacles while traveling between them. Following the Voronoi edge between two obstacles minimizes the chance of a collision with an obstacle, which is a desirable property when the transition dynamics for the environment are non-deterministic and the cost of a collision is high. Visually, a Voronoi diagram divides the free space into *Voronoi regions* of points that share a common closest obstacle.

Voronoi diagrams are theoretically defined in spaces of arbitrary dimension and for obstacles of any shape. However, the complexity of creating the diagram grows quickly with the dimensionality of the configuration space. Additionally, more complex obstacle shapes bring more complex edges on the diagram. In this paper, we focus on two-dimensional environments with obstacles that are points.

B. Nearest-Neighbors Classification

Voronoi diagrams have other uses beyond robot motion planning. One application, which we explore in this paper, is to use a Voronoi diagram to build a *k-Nearest-Neighbors (KNN) classifier*. KNN classifiers come from the field of machine learning. The classification problem is as follows: given a *training set* of feature vectors with labels of either 0 or 1, produce a machine that, given a new feature vector *T*, will identify the *k* members of the training set with the closest feature vectors to *T*. The output label for *T* will be the class that the majority of the *k* nearest neighbors belong to. The metric used to define the "closest" feature vectors can vary based on the characterization of the available features, but the simplest way to measure it is to take the distance between the two points plotted in Euclidean space.

A complete Voronoi diagram can be used trivially as a KNN classifier with $k = 1$; finding the Voronoi region that a test sample belongs to is sufficient, as the nearest neighbor is the training sample associated with that region. For higher values of k, the classification problem is non-trivial. We explore how to implement such a system in this paper.

II. Methods

A. Creating Voronoi Diagrams

The problem of an efficient algorithm for creating Voronoi diagrams has been thoroughly explored in the literature. [1].

$$x = \frac{a_y b_x \pm \sqrt{a_y b_y((a_y - b_y)^2 + b_x^2)}}{a_y - b_y} \tag{1}$$

III. Conclusion

The conclusion goes here.

References

[1] S. Fortune, "A sweepline algorithm for voronoi diagrams," *Algorithmica*, vol. 2, no. 1-4, pp. 153–174, 1987.