Algorithm Investigation

Michael Hlastala, Tysen Radovich, Andrew Dixon, Michael Hewitt

October 30, 2019

1 Algorithms for Computing Geometric Measures of Melodic Similarity

The focus of this algorithm is to find the similarities between two provided melodies. Rather than an applications such as *Shazam*, which focuses on finding an exact match to a given audio input, the proposed algorithm is designed to take two melodies and compare their similarity. An example use of this algorithm is for detecting copyright infringement. With an algorithm like this, two artists in a legal dispute of originality of a melody could use an algorithm such as this to determine the accuracy of two melodies in two different songs to compare how similar the structure of the song is.

This algorithm can also be used to develop Query by Humming software. Query by Humming is a concept used for song identification by having a user recreate a melody by a method such as humming or singing. A Query by Humming software has major applications in a commercial market. With an accurate melody identification algorithm, useful applications could be made to allow users to discover new music by recreating songs they have heard, or discover music with comparable melodies.

The algorithm uses differences of pitches to form the melodies of a given input, and then overlays each melody over each other to determine the difference between the two. There are two ways the melodies are compared: the first is in the z direction, so by comparing the distance between the pitch of the melodies, and the second is by comparing the melodies in the θ direction, so by comparing the length of a given pitch. Combining these two factors develops the difference between the two pitches

Aloupis, Greg, et al. "Algorithms for Computing Geometric Measures of Melodic Similarity." *Computer Music Journal*, vol. 30, no. 3, Fall 2006, pp. 67–76. *EBSCOhost*, doi:10.1162/comj.2006.30.3.67.

2 Proposal of Algorithms for Navigation and Obstacles Avoidance of Autonomous Mobile Robot

Autonomous vehicles navigate their world by gathering as much information of their surroundings as possible while processing that information into a format beneficial to quick, precise, and intelligent decision making. While autonomous road vehicles have much higher stakes and far more information to take in, autonomous indoor vehicles such as the classic Roomba tend to have larger constraints. These can range from the size of the vehicle to the price at which they are sold at, and due to these constraints new issues arise. The algorithms in the paper "Proposal of Algorithms for Navigation and Obstacles Avoidance of Autonomous Mobile Robot" seek to solve some of these navigation and object detection problems given the resources available on small scale - indoor vehicles. The paper looks at the possible detection methods that are available and what each of their limitations would be, and creates an algorithm for finding a best path given the processing limitations on these small vehicles.

For example, sonar works well for close distance object avoidance but isn't a reliable source for navigating a path or determining which path will be optimal. For this, a laser range finder is used which produces a tremendous number of points and digitally map each to a x,y,z point in space. The issue here is that processing and graphing this data, which is constantly changing as the vehicle moves, is very time and space intensive. While have a 3D model would be beneficial to a self-driving car, it isn't necessary for an indoor vehicle needing to make far fewer complex decisions. To reduce the complexity of navigating a path the algorithm proposed in this paper named 3D-to-2D image compression takes the 3D point cloud and reduces it to a 2D image that is far less intensive to process but remains sufficient for navigation. With the assistance of sonar and a camera for vertex detection the algorithm allows for quick processing and optimal path selection.

Hoang, T. T., et al. "Proposal of Algorithms for Navigation and Obstacles Avoidance of Autonomous Mobile Robot." *2013 IEEE 8th Conference on Industrial Electronics and Applications (ICIEA)*, 2013, doi:10.1109/iciea.2013.6566569.

3 Parametric Search Made Practical

I am using the parametric search made practical. In this algorithm it makes the Fréchet-distance problem possible using the parametric search made practical framework. The best way to describe the Fréchet-distance problem is that if there is a person who is walking a dog on a leash, if the two are both walking along two different curves how do we minimize the maximum distance between the person and the dog. Basically you want the smallest size of the leash that can get so that it covers the maximum distance between the two. Using the parametric search made practical you take this leash value for every point compare using Quicksort-based parametric search. According to Rene and Veltkamp, ”Using the framework to do sorting-based parametric search requires the user to implement the following classes or functions:(1) A class or function that computes the items that are to be sorted from the input. The input may consist of these items themselves; (2) A class derived from Comparisonbase that computes the roots associated with the comparison of two items, and that can determine the outcome of the comparison once the solution of the decision problem P is known for all these roots;(3) A class or function that solves the decision problem.” This is talking about how we actually use a parametric search for this algorithm. Once this is determined we can take that compare the minimum distance then sort that by using Quicksort.

Van Oostrum, René and Remco C Veltkamp. “Parametric Search Made Practical.” Computational Geometry: Theory and Applications, vol. 28, no. 2-3, 2004, pp. 75–88. https://www.sciencedirect.com/science/article/pii/S0925772104000203