Detect Sparse Observation Zones
Senior Design Group 14-31

Project Plan

Team Members:
John Harding
Nicholas McLaren
Michael Ore
Andrew Upah
Bryce Wilson

Advisor:
Dr. Ruchi Chaudhary, Department of Biology, University of Florida

Client:
Prof. Gordon Burleigh, Department of Biology, University of Florida

Contents

Problem Statement

There are a number of big databases that share biodiversity data for biological research and collaboration. For example, the Global Biodiversity Information Facility (http://www.gbif.org) has millions of species records such as latitude and longitude coordinates. In addition, the Avian Knowledge network (http://www.avianknowledge.net) has over 100 million bird observation records. These databases provide an enormous amount of information that is useful in understanding the patterns and dynamics of various species across a region. Another interesting, yet not well explored, direction is finding out the biggest regions where there are few or no species records. This may represent different features in the landscape or areas where there has been little or no sampling. This project is focused on developing a software tool that inputs a collection of millions of points and outputs the largest areas where there are few or no observations. More formally, this problem is written in the following way:

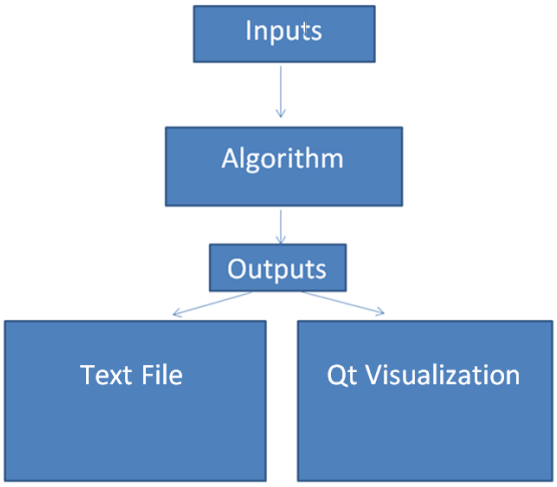
Problem (Detect Sparse Observation Zones):

Input: a collection of latitude and longitude points: *k*, *n*.

Output: *n* latitude and longitude points with corresponding radiuses such that there are no more than *k* points in the respective circle of each point.

System Block Diagram

The system block diagram shown below is represented showing the basic outlay of how our overall system will be broken down. The inputs and outputs were presented above in the problem statement. In order to achieve this, we need to create some sort of algorithm that locates the sparse regions. The output of this algorithm will be will be written to a text file and displayed via Qt, a cross-platform GUI application framework.



Operating Environment

This application will be able to execute on Windows, Linux, and MacOS operating systems through any computer in a lab, office, or home environment. Testing will be done on all machines to ensure our program will work according to the above statement.

Functional Requirements

1. Algorithm must take as input a list of lat-long coordinates and a number k describing the number of observation points included in the output areas (in order to account for outliers)
2. User interface must output a graphical representation of the largest areas that each encompass k observations

Non-Functional Requirements

1. Program must be able to run on Windows/Linux/MacOS
2. System must have an intuitive and aesthetically pleasing UI
3. Algorithm must run in a reasonable time for large inputs (at least for millions of data points)
4. The display should be responsive even while displaying a large number of geometric shapes
5. Algorithm should be novel, publishable

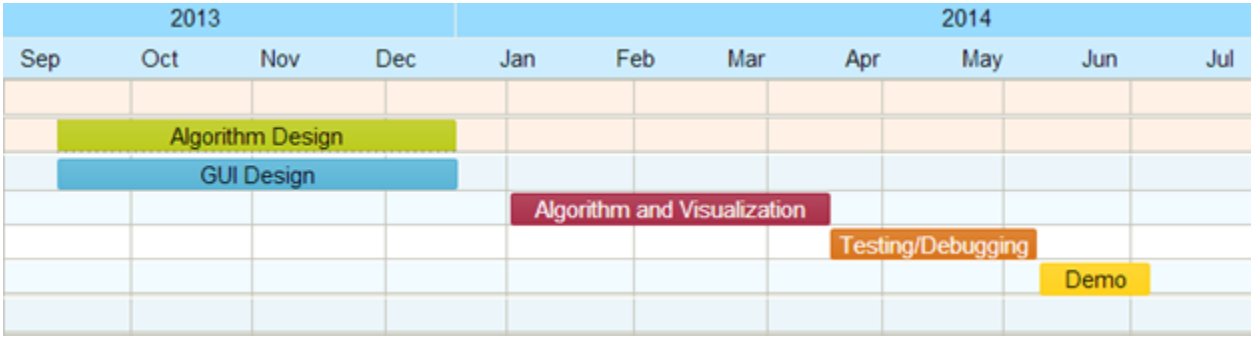
Literature Survey

We have searched through literature on geometric optimization, and couldn't find any existing algorithms that exactly matched our problem as stated. A couple of papers come close.

A paper by Sariel (2005) discusses algorithms for finding small circles that enclose k points, rather than the large circles we're looking for. As it is the reverse of what we want, we may be able to use ideas from it in our design.

One paper by Toussaint (1983) discusses a solution to another problem similar to ours, the Largest Empty Circle problem on a set of points with the circle center constrained. This is a special case of the problem we're trying to solve, so it serves as another lead.

Project Schedule



Work Breakdown Structure

We will be implementing the Waterfall methodology model for our project, so design of our entire system will be built first with integration, then testing, and finally maintenance. Individually, we will be working on two modules which is split up and shown below.

First Semester:

- Michael and Nicholas: Design the algorithm / Build a prototype to test the final algorithm
- Bryce, John, and Andrew: Design the Graphical User Interface / Make screen sketches / Develop Use Cases

Second Semester:

- Michael and Nicholas: Develop code and perform integration testing on chosen algorithm
- Bryce, John, and Andrew: Write code for user interface / test interface using example input/output data
- Everyone: Integrate both modules together / Perform final testing on program

---

Resource Requirements

As this is a software project, we aren't in need of many resources beyond our own computers. There is a small chance that we'll find a commercial software license useful. We plan on making use of several open source libraries, including CGAL (http://www.cgal.org) and Qt (http://qt-project.org). We're relying on sample data from our client for validation and to help us better understand the use cases.

Deliverables

By the end of the first semester, we plan to deliver a prototype of the core algorithmic portion of the code. We will also show mockups of our user interface design.

Towards the end of the second semester, we will have a fully integrated and polished software package implementing our algorithm and easy-to-use graphical user interface.

Risk

Solving the given problem in full involves a lot of research, and possibly the invention of novel algorithms. There is a chance that we would need to compromise on scope or accuracy in order to keep performance reasonable. We have good leads in our literature survey, so we remain optimistic that we can solve the problem satisfactorily.

Team Information

Team Members	**Role**	**Contact Information**
John Harding (CprE)	UI Designer	otangs@iastate.edu
Nicholas McLaren (CprE)	Algorithm/Implementation	nmclaren@iastate.edu
Michael Ore (CprE)	Researcher/Webmaster	orem@iastate.edu
Andrew Upah (CprE)	CGAL-UI Integration/Leader	aupah@iastate.edu
Bryce Wilson (CprE)	UI Rendering	bmwilson@iastate.edu
Ruchi Chaudhary	Advisor	ruchic@ufl.edu
Prof. Gordon Burleigh	Client	gburleigh@ufl.edu

Closing Summary

All information in this document describes our current plan for this project the overall structure we will follow to tackle this problem. If any information was not presented here, you can find more information on our design document or by contacting one of the team members. Finally, we would like to thank Prof. Gordon Burleigh for coming up with this interesting problem as well as our advisor, Dr. Ruchi Chaudhary, for guiding us along the way.

Bibliography

Har-Peled, Sariel, and Soham Mazumdar. "Fast Algorithms for Computing the Smallest K-Enclosing Circle." *Algorithmica* 41.3 (2005): 147-57. Print.

Toussaint, Godfried T. "Computing Largest Empty Circles with Location Constraints." *International Journal of Computer & Information Sciences* 12.5 (1983): 347-58. Print.