CS51 – Project Report

Convex Hull and Sweep Line Algorithm

-Vinay Mishra ([vmishra@g.harvard.edu](mailto:vmishra@g.harvard.edu))

HUID: 10886759

Submission date: 30-Apr-2015

# Overview

In this project, I implemented two algorithms related to computational geometry: ‘finding the convex hull’ of given points and ‘determining whether any pair of line segments intersects’ in two dimension.

I used C++ language to implement these algorithms and OpenGL libraries for visualization. I used ‘MS visual studio 2013’ as IDE to organize and compile my code. OpenGL libraries opengl32.lib, glu32.lib and glut32.lib was included as additional dependencies in the visual studio project. The program reads the input points from file ‘2dpointsdata.txt’, where each line corresponds to a point in 2D plane. The first line of this input file specifies the type of data. When the code is compiled and run, a graphics window is opened with the input points drawn in red color.

The first algorithm of convex hull is accessed by pressing character ‘c’ or ‘C’ in the keyboard and a line polygon is drawn in green color which represents the convex hull for the input points.

The second algorithm of determining segment intersection, is accessed by keyboard character ‘s’ or ‘S’. The blue lines represent the input segments and any intersecting segments are drawn in red color. There are two limitations of this implementation. First, no input segment is vertical and second, no three input segments intersect at a single point.

While we are running the program the input data file can be modified to visualize the outcome of algorithms for different data set. I have used the float data type in this implementation, but since point, line and event types are generic C++ templates, it is easy to alter the code to accommodate a different data type e.g. integer or double.

# PLANNING

Here is my [draft specification](https://github.com/vmishra22/FinalCS51/blob/master/Draft_Specification.pdf) and [final specification](https://github.com/vmishra22/FinalCS51/blob/master/Final_Specification.pdf) of the project. The planning of the project started with finding the computational geometry algorithms which can be implemented in the given time frame. In the draft specification I mentioned that I might use OpenGL for the visualization, if time permits. In the end, I was able to integrate graphics code with algorithms implementation. Implementation of the segment intersection algorithm involved implementing an additional algorithm which I had not taken into account initially. SEGMENTS-INTERSECT algorithm of the reference book (Introduction to Algorithms, 3rd edition) is implemented to check segments intersection and used in ordering the segments when the sweep line event occurs.

Initially, I had planned on providing a Boolean output for segment intersection algorithm, if any of the input segments intersect, but in the end I was able to collect all the intersecting segments which are redrawn in different color to provide a better visualization.

# Design and Implementation

Since both the algorithms have some common data structures involved, I started with defining Point and Line classes using C++ templates, so that these objects could be used with any meaningful input data type (e.g. int, float etc.) and with the possibility of extending implementation to three dimensional space. The input point’s data is read from the file using STL stream functions. The points are sorted based on the y-coordinates and using lesser x-coordinate in case of tie. Next, the points are sorted by polar angles in counterclockwise order using first point as reference. I used STL containers vector and stack for the convex hull algorithm with sort function from STL algorithm library.

I also introduced another class for an event which was not mentioned in the final specification document. After reading the input points, a vector of event objects are created. Using this vector container, the event objects are inserted in the priority queue in the order of the x-coordinates and in case of tie, the points with lower y-coordinate are put first. While iterating over the event queue, a segment label set is maintained. The ordering of segment in this set is decided based on whether vertical sweep line intersects the segments and whichever segment has higher y-coordinate at intersection point, is placed before the other. I learned using different type of STL containers e.g. Set, Vector, Priority Queue and Stack, while implementing both of the algorithms.

# Reflection

I have tried to put some sanity checks for the input, but I wish I would have put some more tests for input data. I would have also liked to improve user interface of the application using OpenGL window graphics, which would include the labeling of segments and points. OpenGL animation could also be integrated with current implementation, which would present the incremental outcome of the algorithm application.

# Advice for future students

In my opinion, one should select the project topic which he/she really wishes to learn about. This will give the impetus to put an extra effort in finishing the project. The scope of the project should also fit well with the deadline.