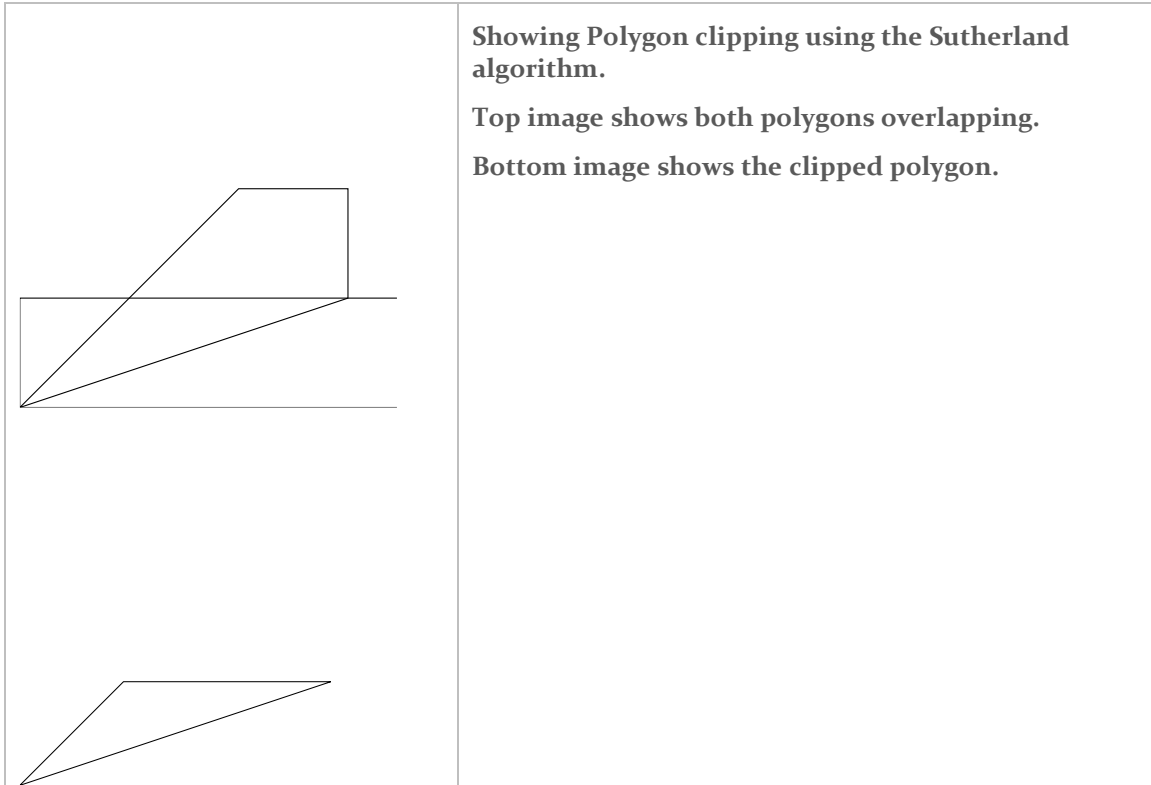


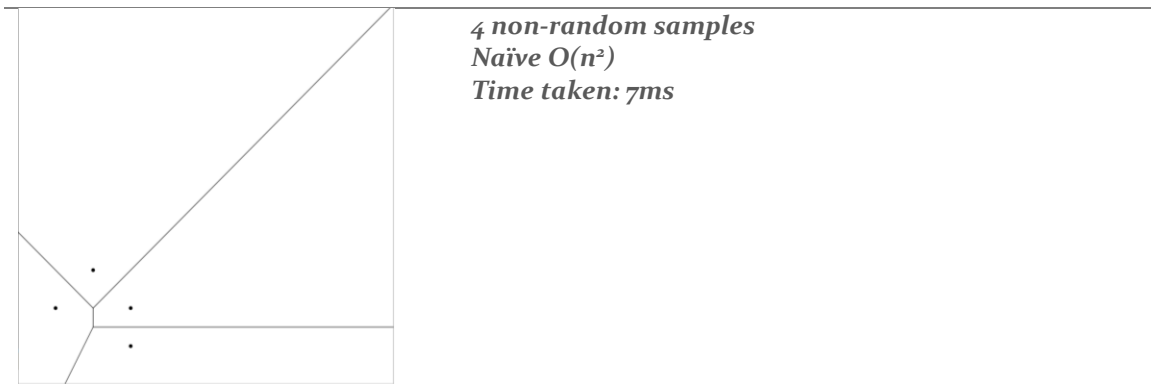
Geometry Processing

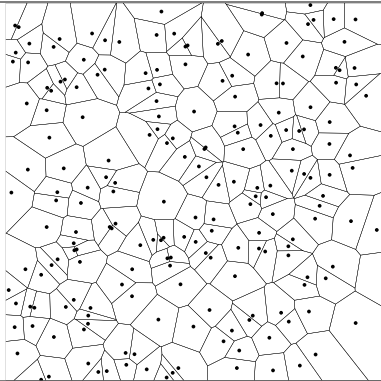
CSE306

Polygon Clipping – Sutherland Algorithm

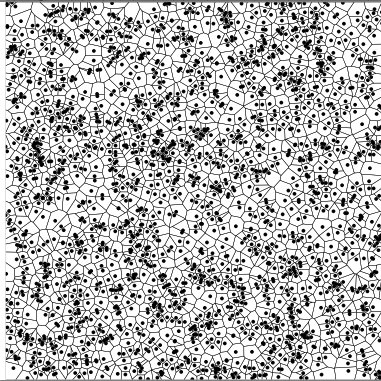


Voronoi diagram



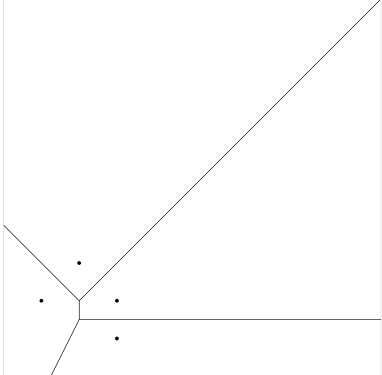
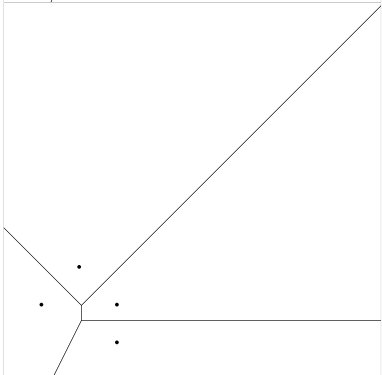
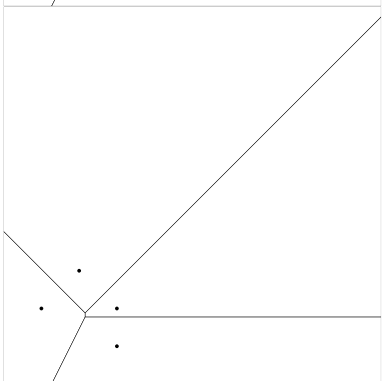


200 random samples
Naïve $O(n^2)$
Time taken: 37ms



2000 random samples
Naïve $O(n^2)$
Time taken: 2207ms

Power Diagram

	<p><i>4 non-random samples</i></p> <p>Vector 1 = (0.1,0.2,0) Weight = 0.03 Vector 2 = (0.2,0.3,0) Weight = 0.03 Vector 3 = (0.3,0.2,0) Weight = 0.03 Vector 1 = (0.3,0.1,0) Weight = 0.03</p> <p>Vector 1 = (0.1,0.2,0) Weight = 0.01 Vector 2 = (0.2,0.3,0) Weight = 0.03 Vector 3 = (0.3,0.2,0) Weight = 0.05 Vector 1 = (0.3,0.1,0) Weight = 0.03</p>
	<p>Vector 1 = (0.1,0.2,0) Weight = 0.01 Vector 2 = (0.2,0.3,0) Weight = 0.03 Vector 3 = (0.3,0.2,0) Weight = 0.08 Vector 1 = (0.3,0.1,0) Weight = 0.03</p> <p>Time taken: 3ms (each)</p>
	

LBFGS – SEMI DISCRETE OPTIMAL TRANSPORT

The code for this optimisation is almost complete. The maths was confusing for me, so I am not sure that the formulas I have written are correct. I also looked at Sebastian's code for aid and have got some of his code copied and pasted and commented out in my file. I take no ownership for this code, I was using it for help.

Please refer to the following functions:

File Polygon.hpp

Area()

Integral()

FLUIDS VIA SEMI-DISCRETE OPTIMAL TRANSPORT

I have implemented the majority of the Gallouet Merigot algorithm, but since I couldn't finish the semi optimal transport function I was not able to continue further.