The Ball-Pivot Algorithm for Surface Reconstruction

James Doverspike

Contributions

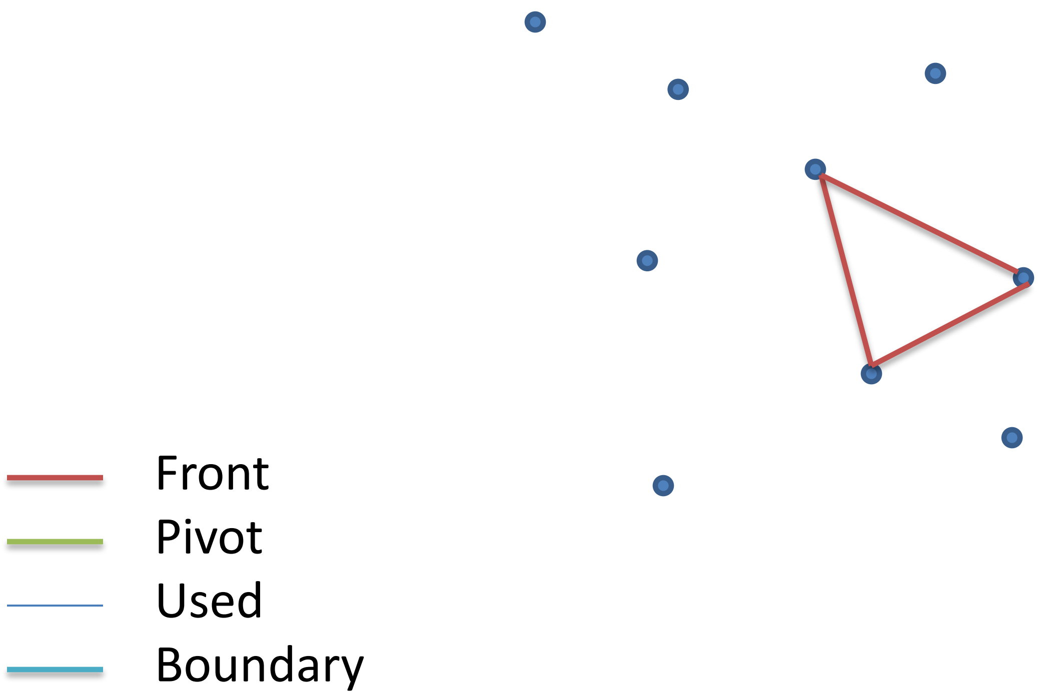
- Simple algorithm
- Manifold subset of an alpha shape
- Linear-time and space complexity
- Out-of-core
- Handles noise

Contributions

The main contribution of the paper is a geometric linear-time algorithm for surface reconstruction from noisy data.

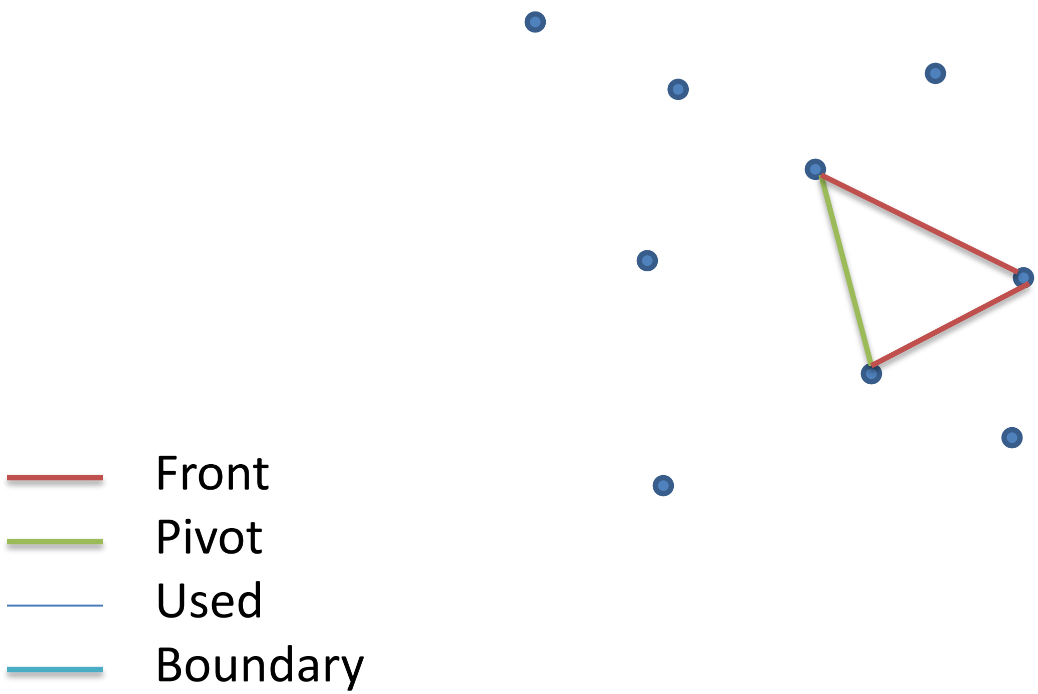
Overview

The algorithm takes in a set of oriented points and produces a mesh in linear time.



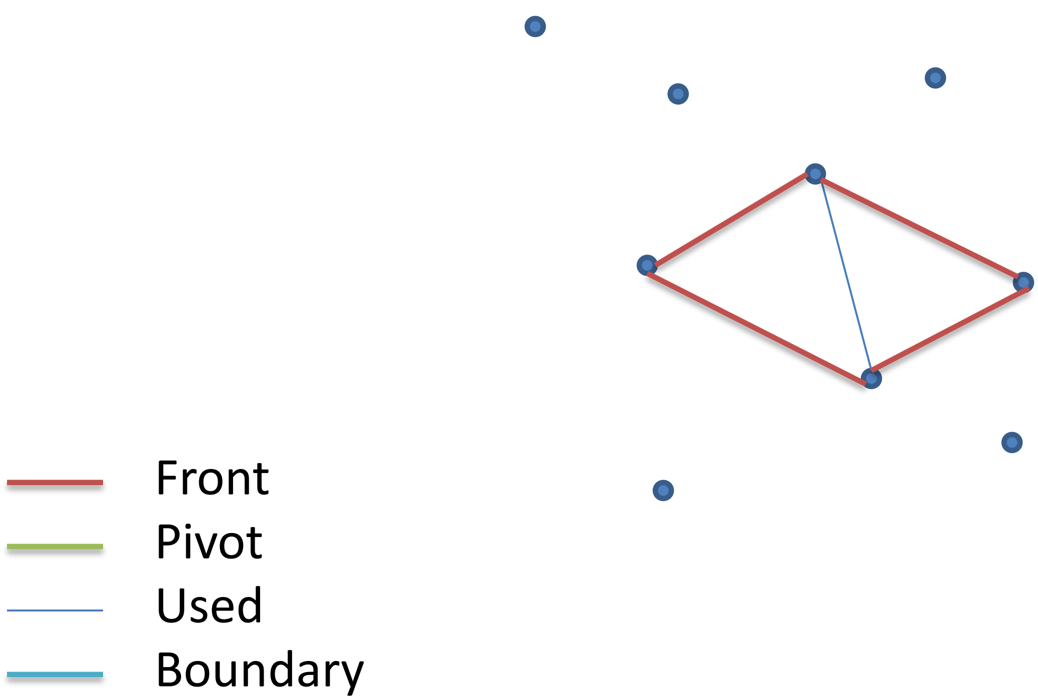
Overview

The algorithm takes in a set of oriented points and produces a mesh in linear time.



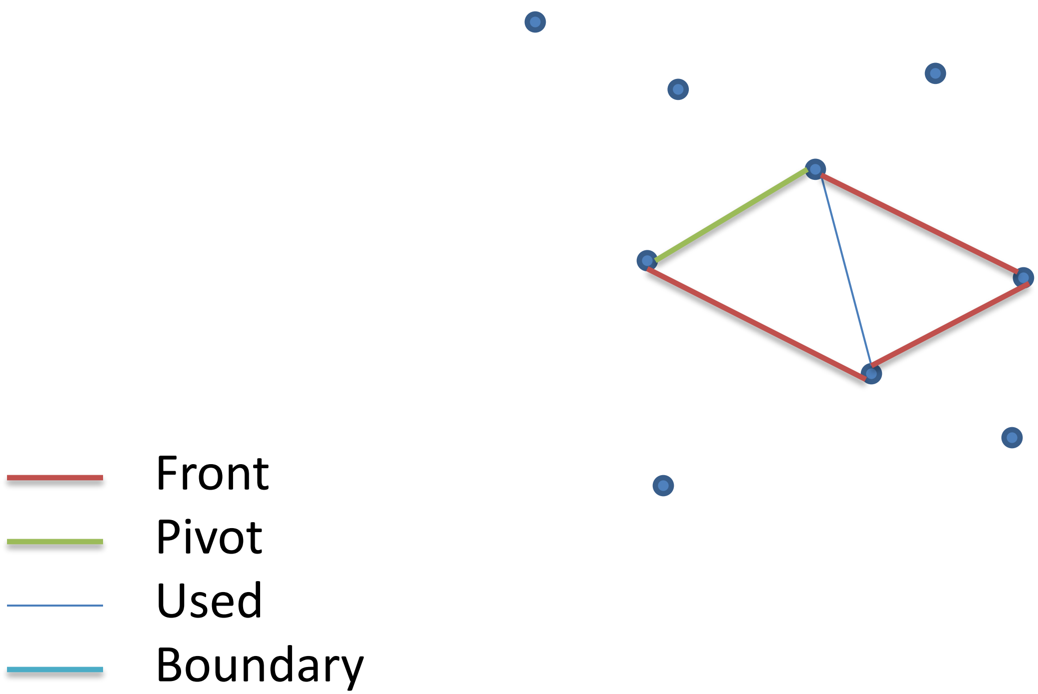
Overview

The algorithm takes in a set of oriented points and produces a mesh in linear time.



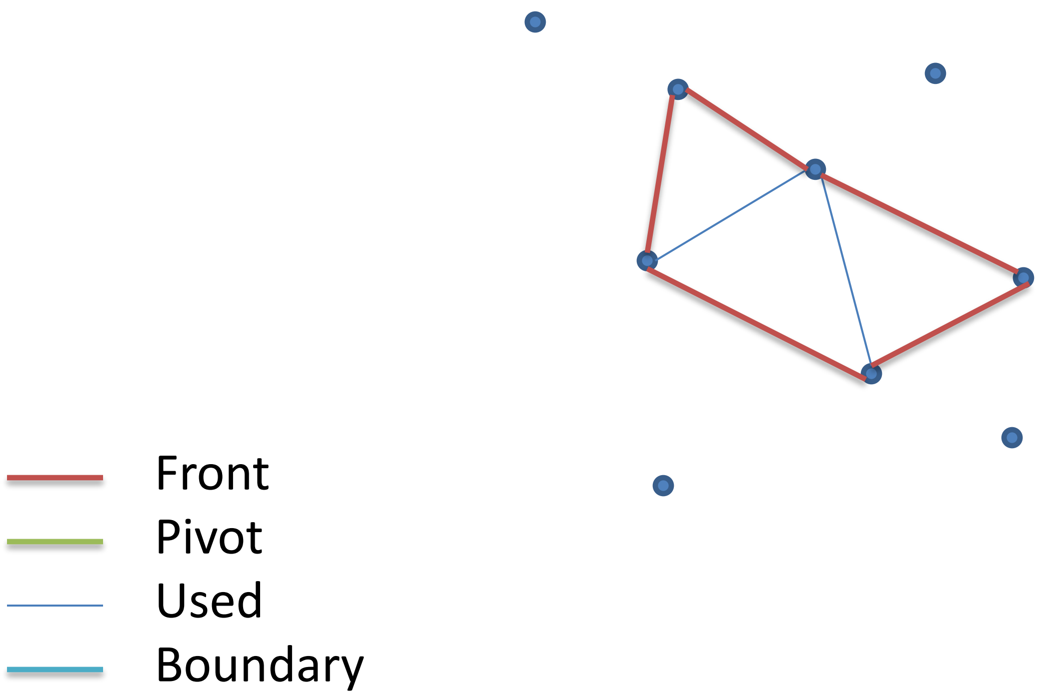
Overview

The algorithm takes in a set of oriented points and produces a mesh in linear time.



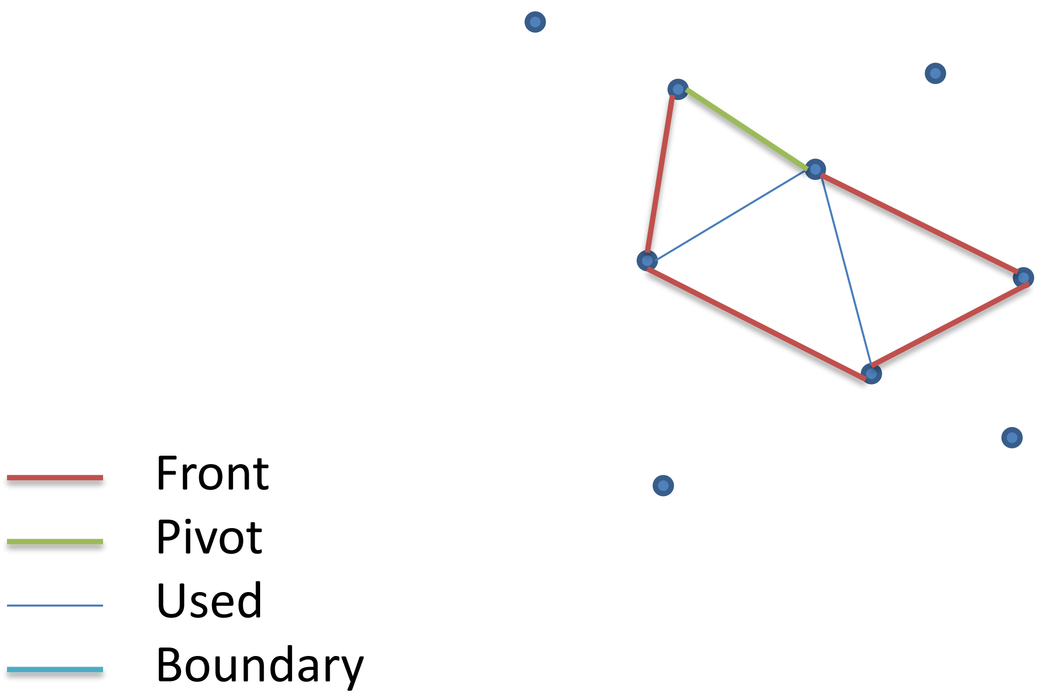
Overview

The algorithm takes in a set of oriented points and produces a mesh in linear time.



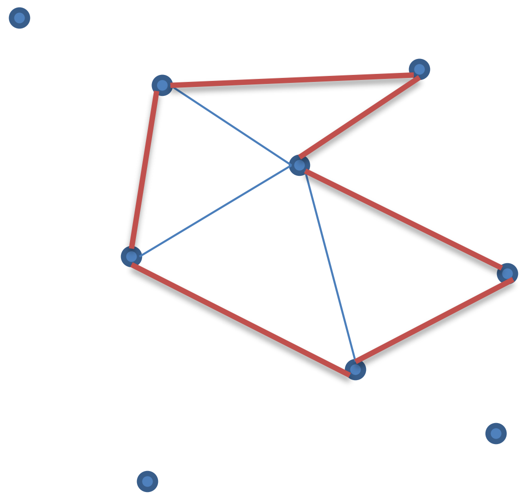
Overview

The algorithm takes in a set of oriented points and produces a mesh in linear time.



Overview

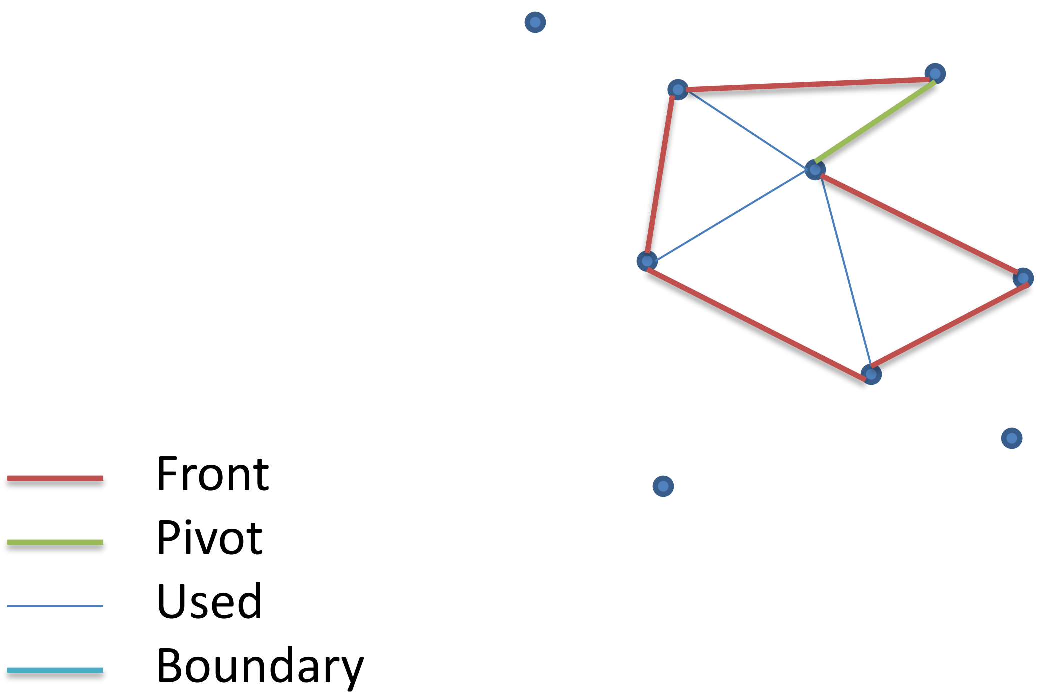
The algorithm takes in a set of oriented points and produces a mesh in linear time.



Front
Pivot
Used
Boundary

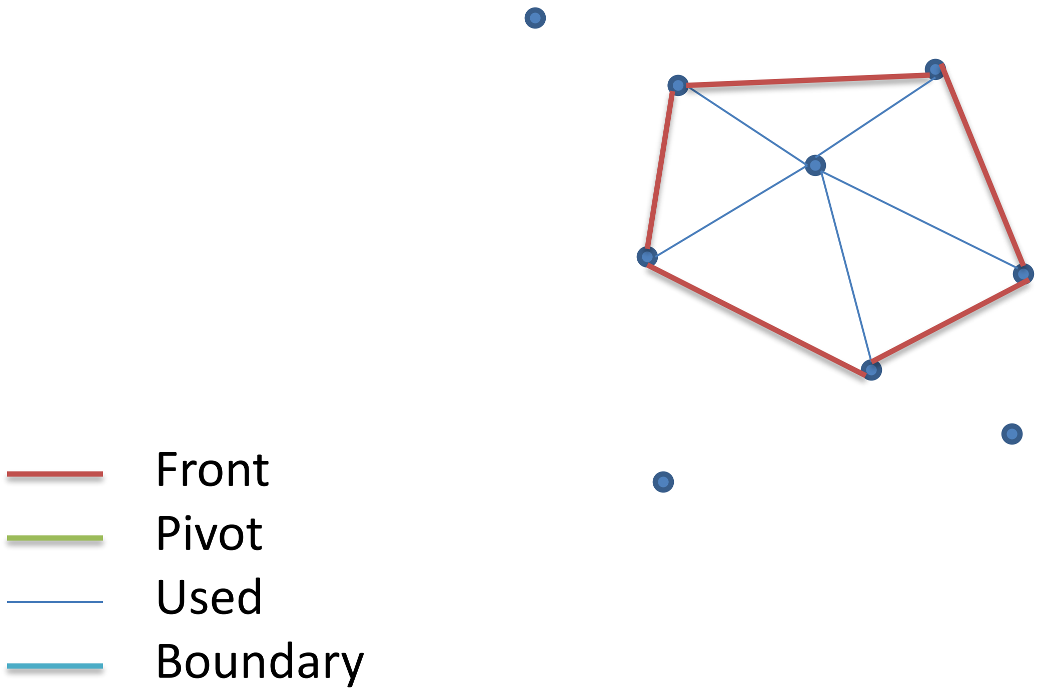
Overview

The algorithm takes in a set of oriented points and produces a mesh in linear time.



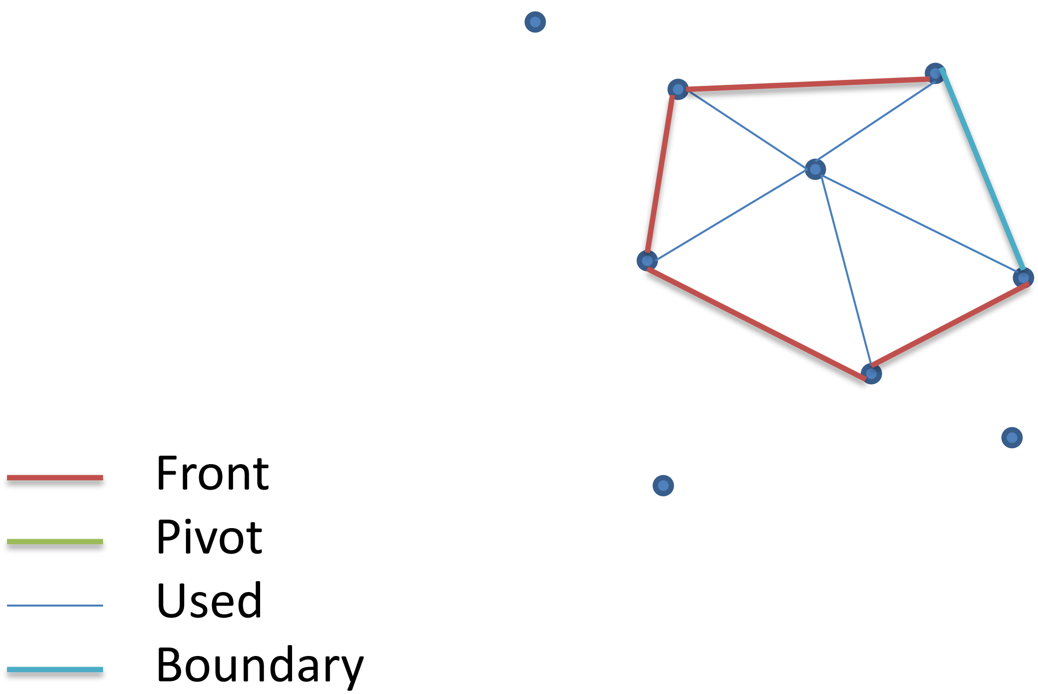
Overview

The algorithm takes in a set of oriented points and produces a mesh in linear time.



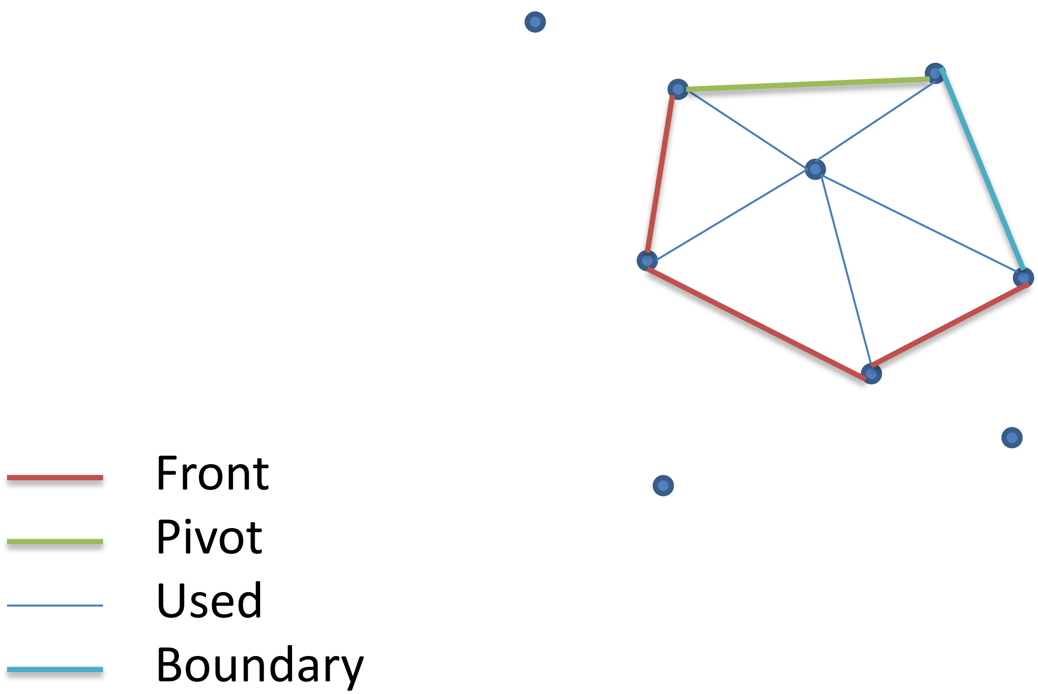
Overview

The algorithm takes in a set of oriented points and produces a mesh in linear time.



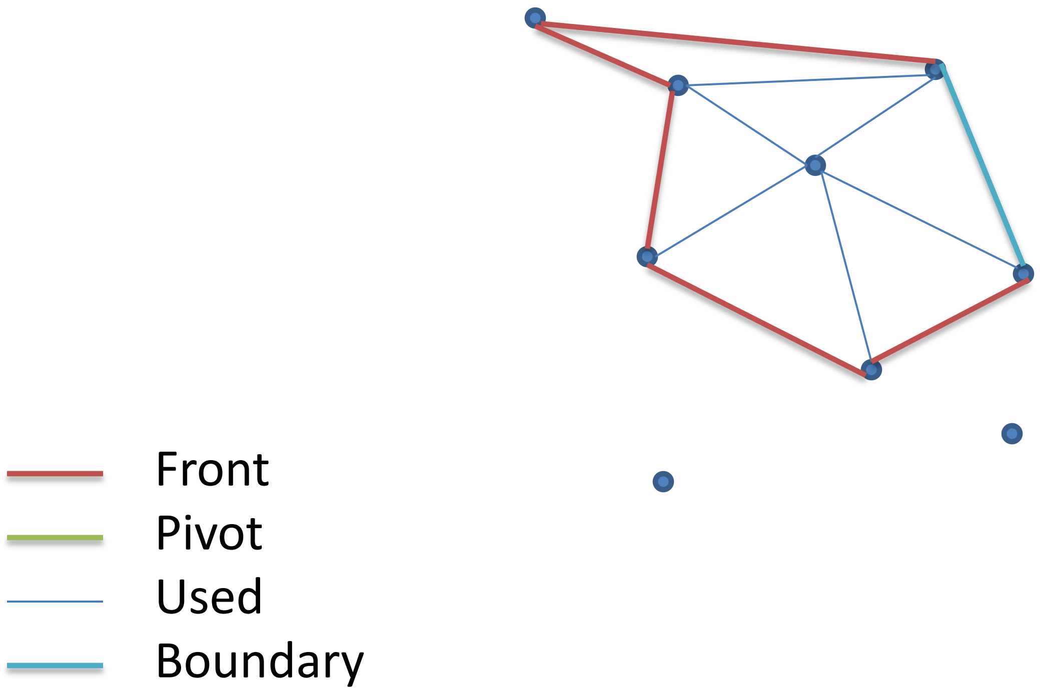
Overview

The algorithm takes in a set of oriented points and produces a mesh in linear time.



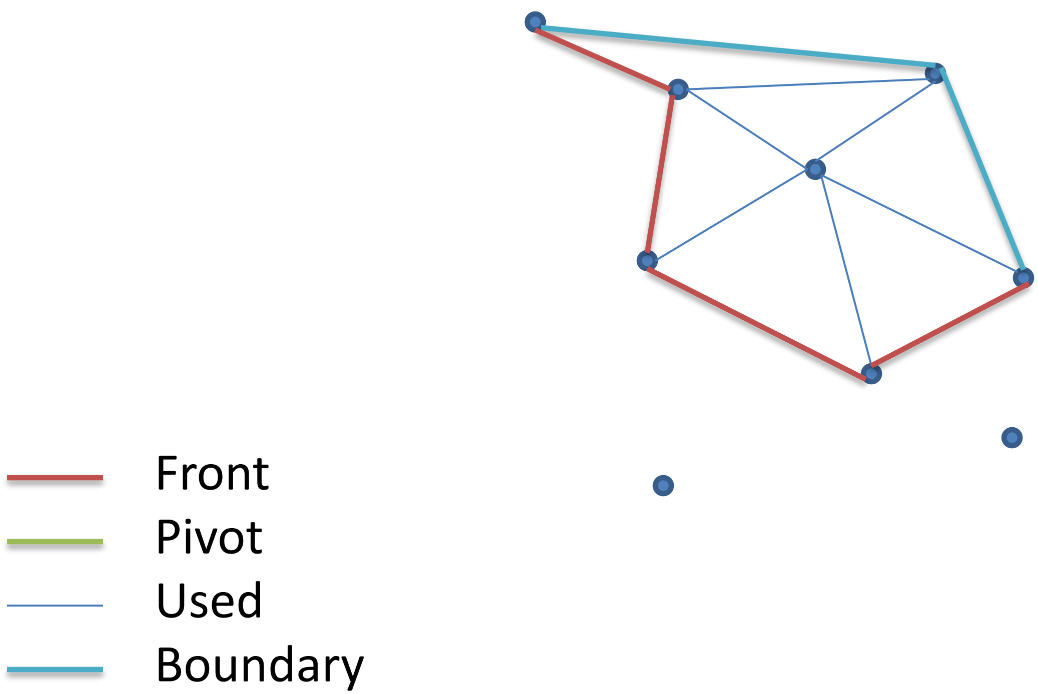
Overview

The algorithm takes in a set of oriented points and produces a mesh in linear time.



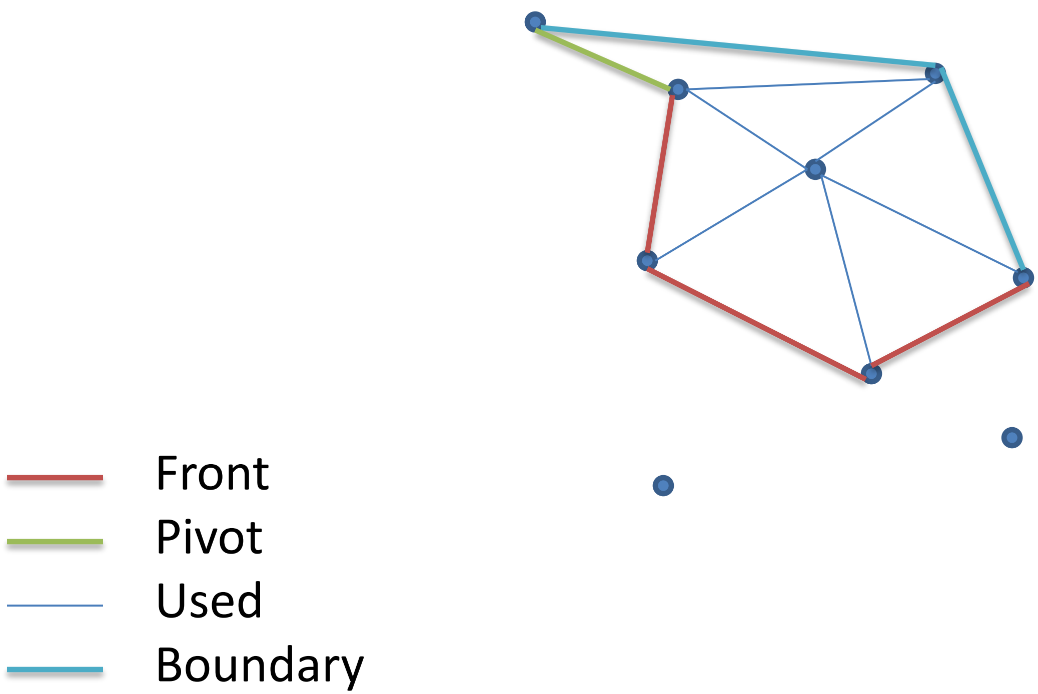
Overview

The algorithm takes in a set of oriented points and produces a mesh in linear time.



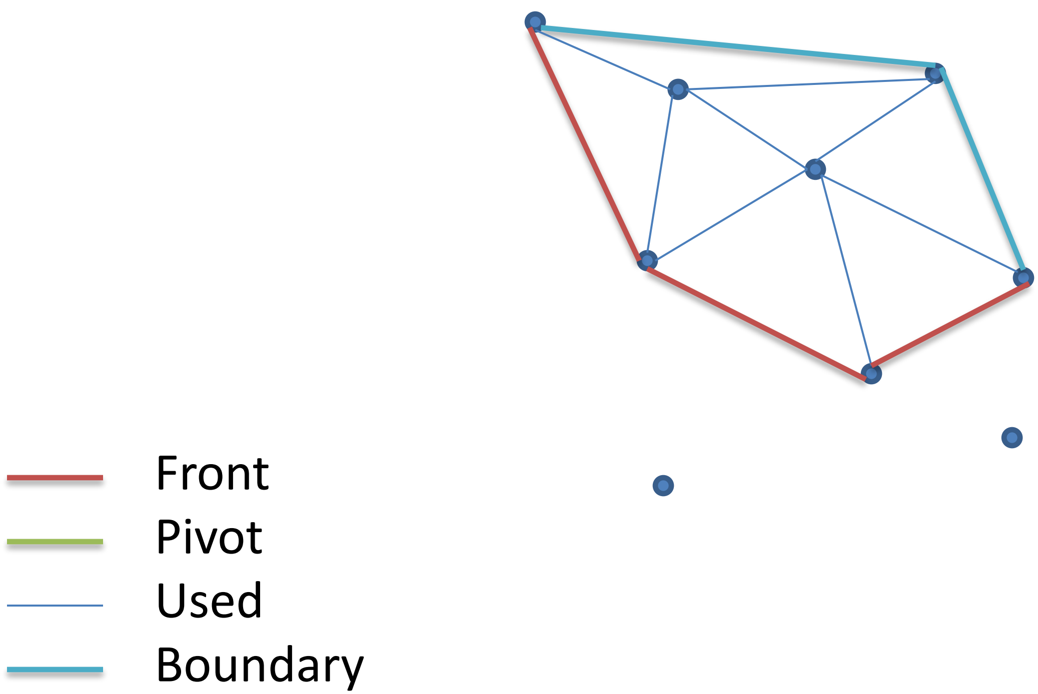
Overview

The algorithm takes in a set of oriented points and produces a mesh in linear time.



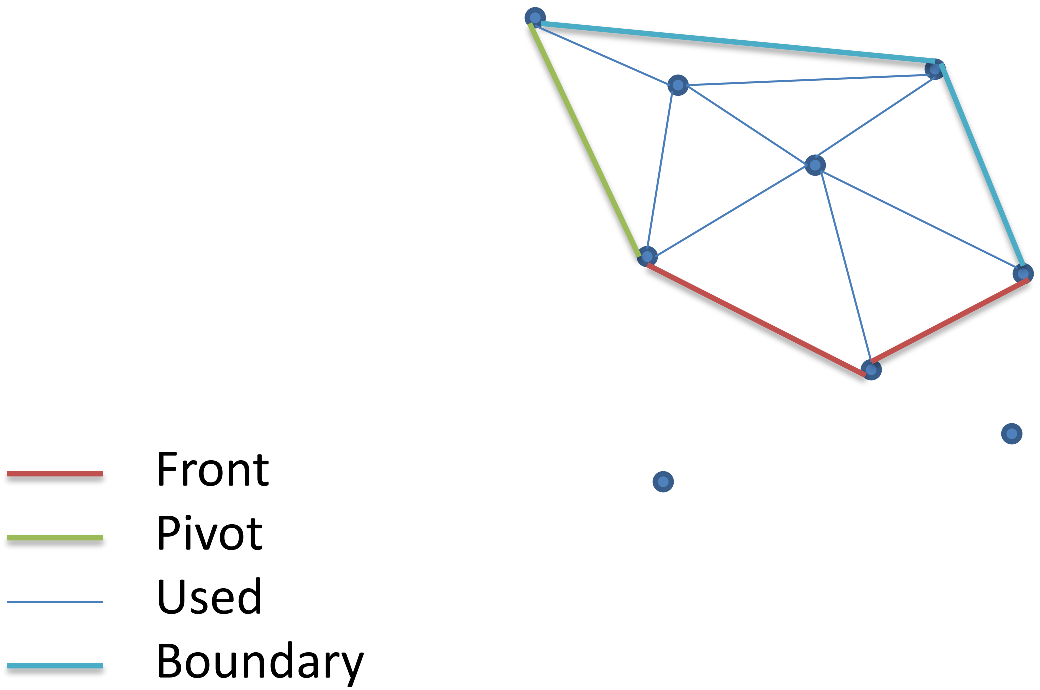
Overview

The algorithm takes in a set of oriented points and produces a mesh in linear time.



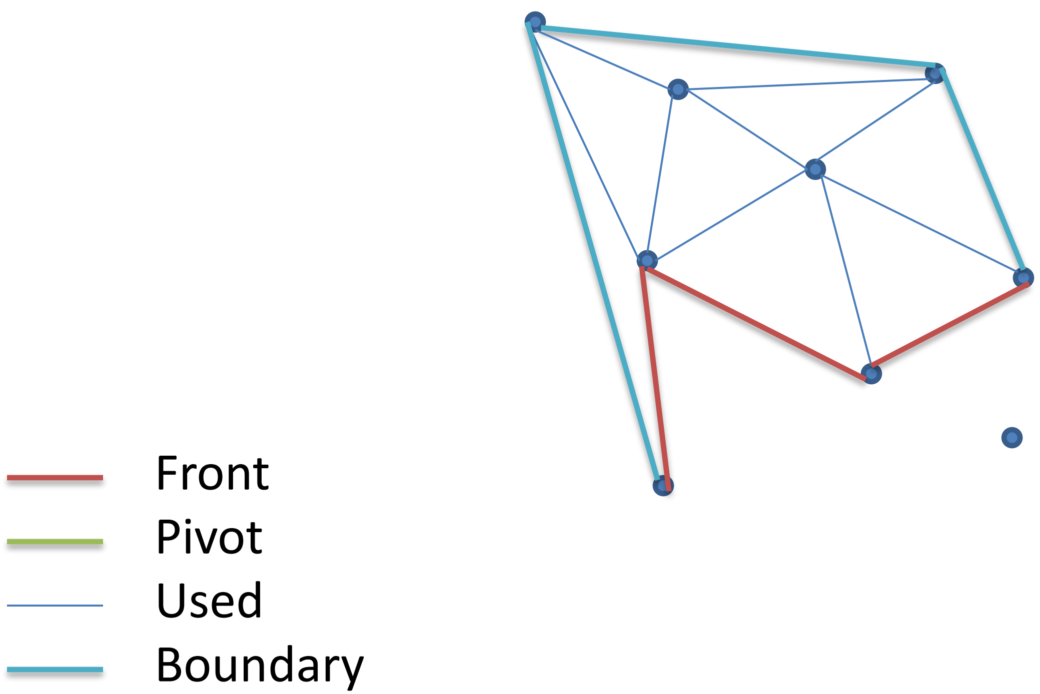
Overview

The algorithm takes in a set of oriented points and produces a mesh in linear time.



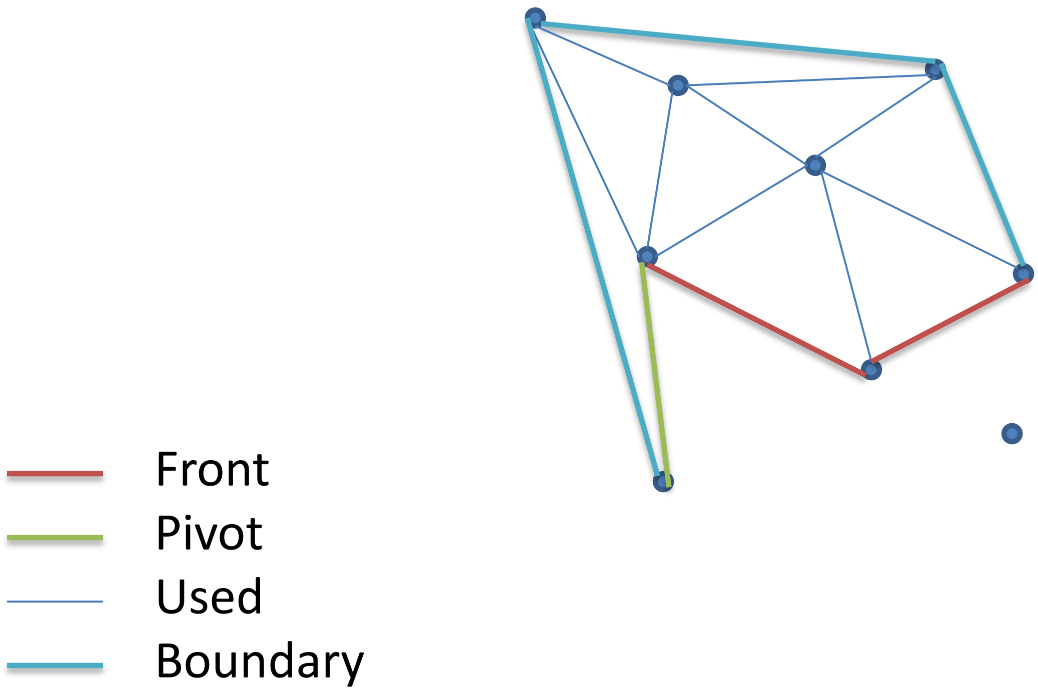
Overview

The algorithm takes in a set of oriented points and produces a mesh in linear time.



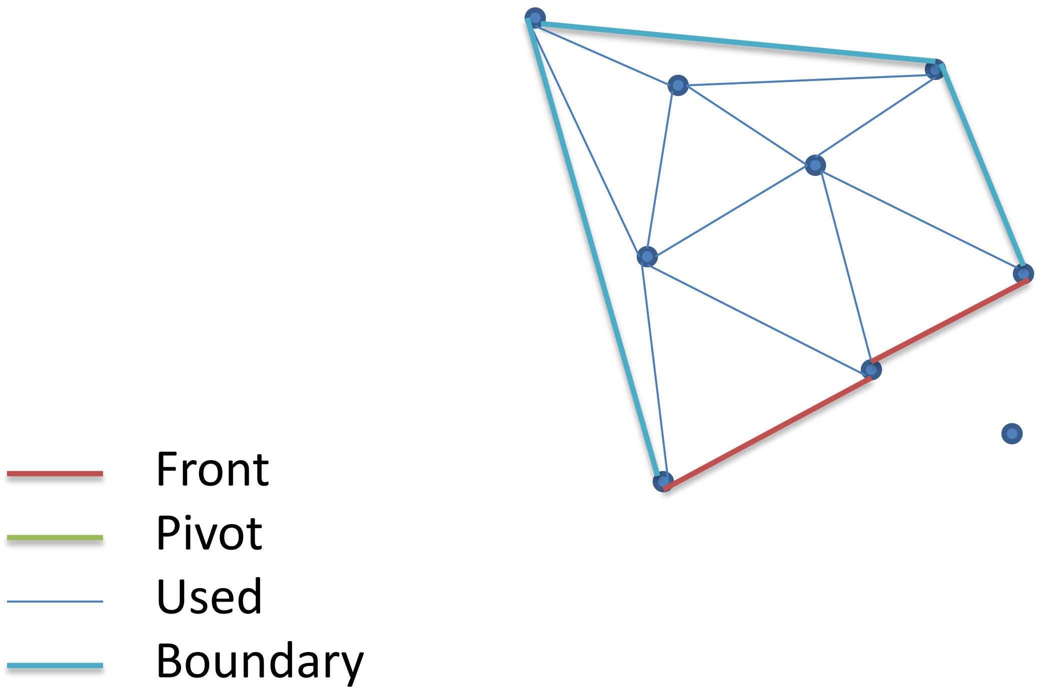
Overview

The algorithm takes in a set of oriented points and produces a mesh in linear time.



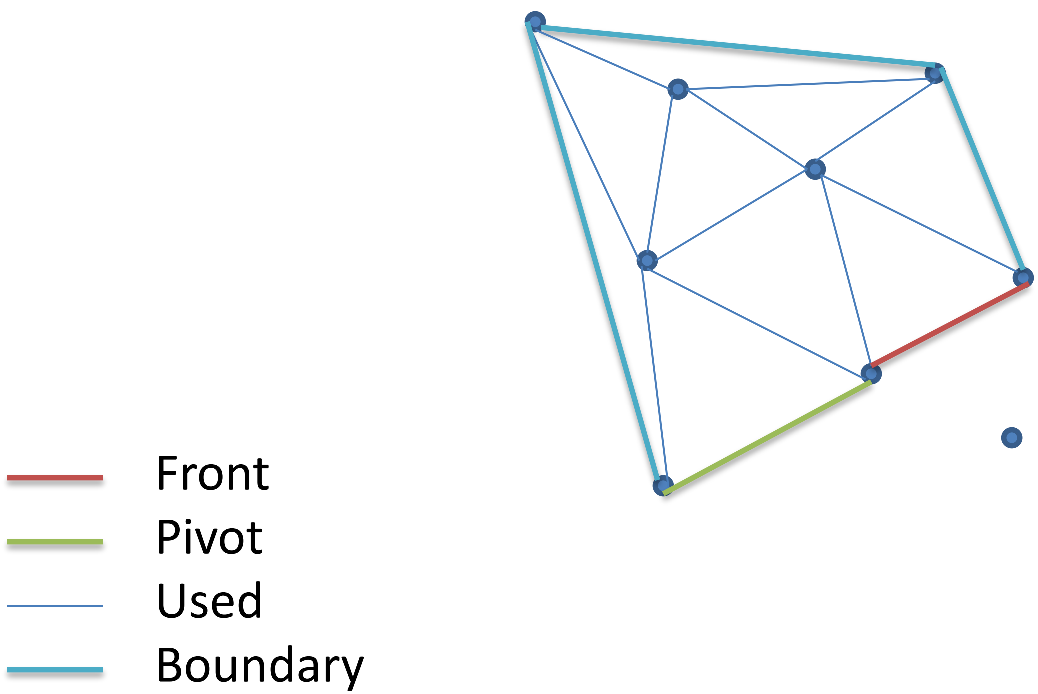
Overview

The algorithm takes in a set of oriented points and produces a mesh in linear time.



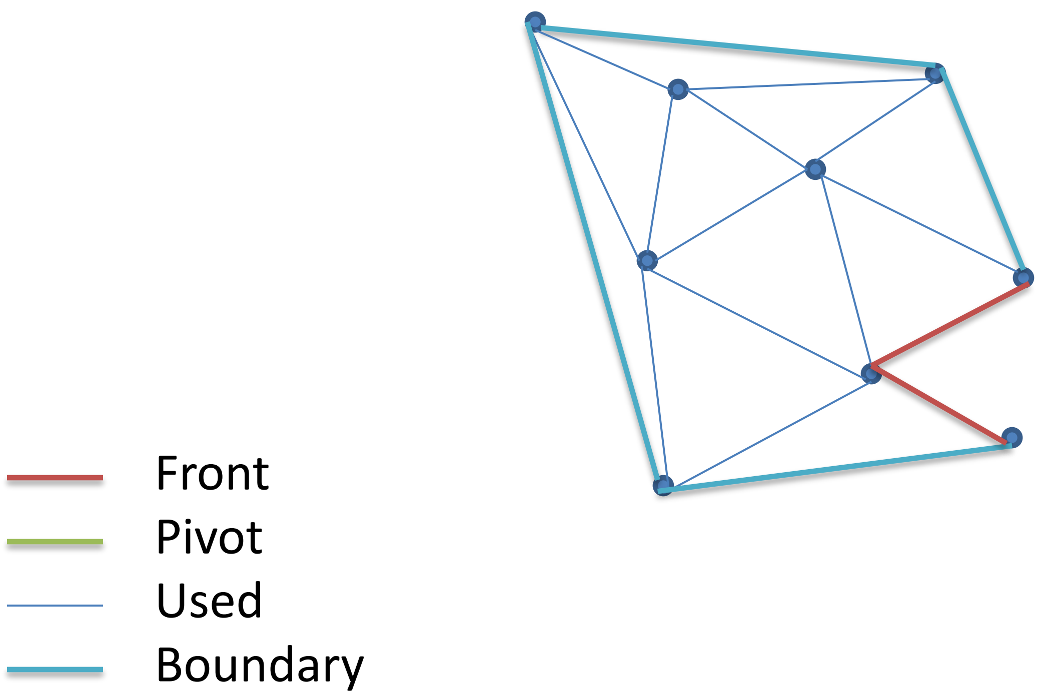
Overview

The algorithm takes in a set of oriented points and produces a mesh in linear time.



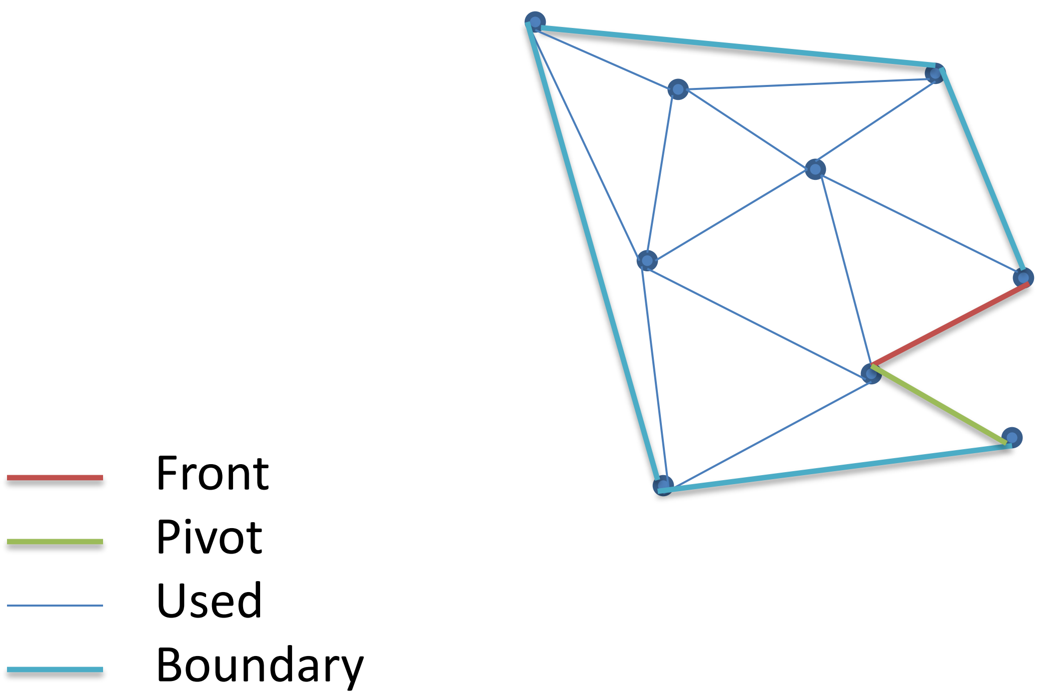
Overview

The algorithm takes in a set of oriented points and produces a mesh in linear time.



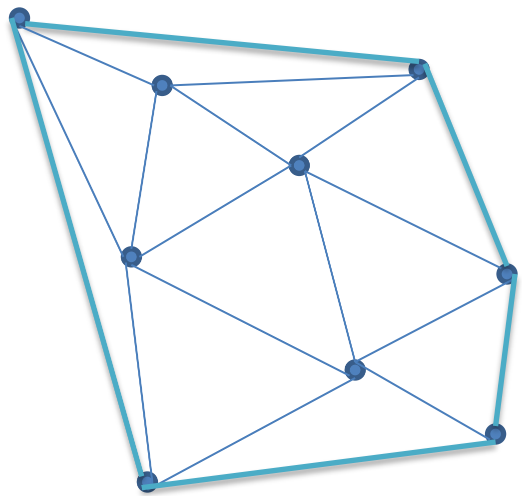
Overview

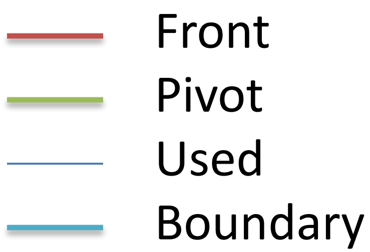
The algorithm takes in a set of oriented points and produces a mesh in linear time.



Overview

The algorithm takes in a set of oriented points and produces a mesh in linear time.





Data structures

- List of edges on front F
- List of computed triangles
- Voxel grid of unused vertices
- List of frozen edges (for out-of-core)

Definitions

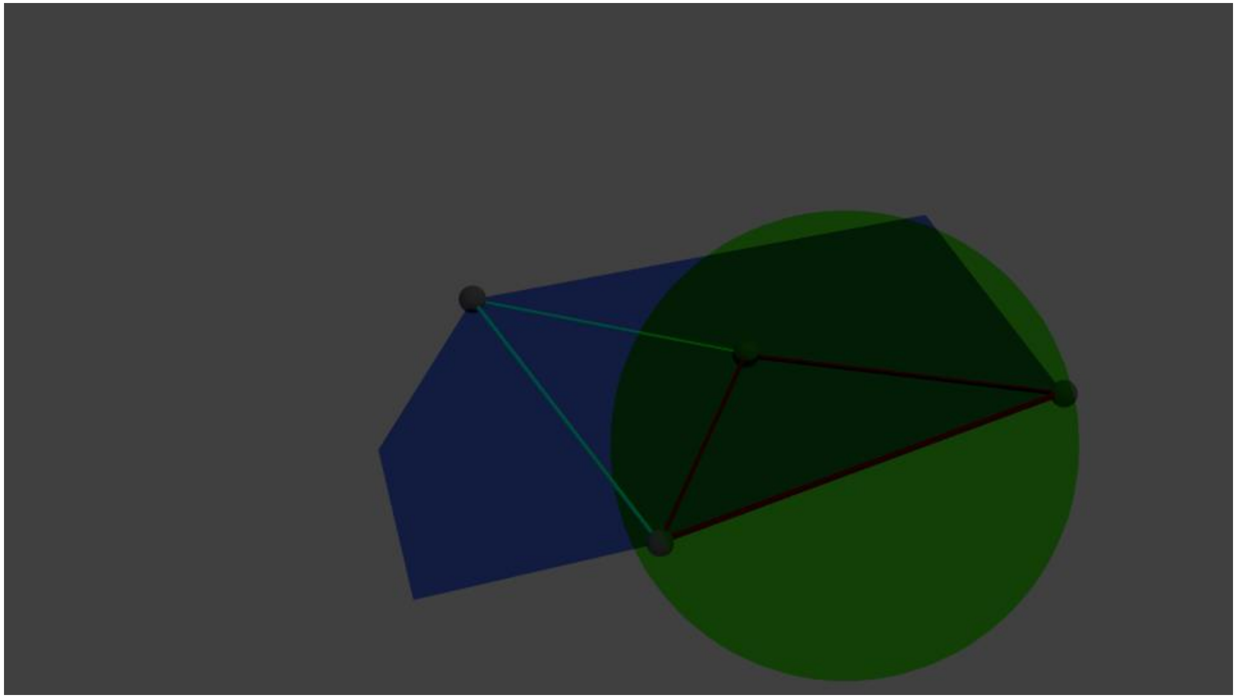
- ρ the radius of the ball
- γ the circle about which the center of the ball moves
- σ_i, σ_j, σ_k the three vertices of triangle τ
- c_{ijo} the center of the ball touching σ_i, σ_j, σ_k
- n the normal of triangle τ
- e_{ij} the edge between vertices σ_i and σ_j
- m the midpoint of e_{ij}

Finding seed triangles

1. Pick an unused vertex σ
2. Consider all pairs of points σ_a, σ_b in a ρ-neighborhood in order of distance from σ
3. Build potential seed triangles $\tau(\sigma, \sigma_a, \sigma_b)$
4. Verify that the triangle normal matches the vertex normals
5. Test that a ρ-ball with center in the outward halfspace touches all three vertices and contains no other points

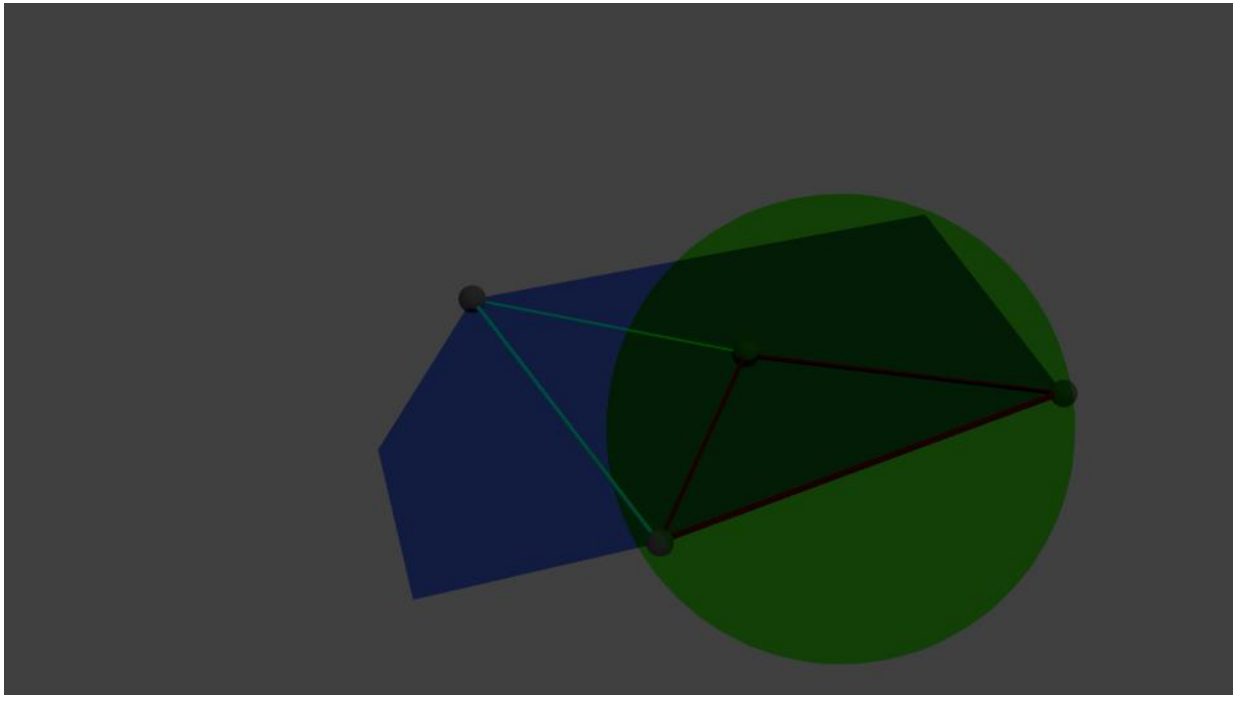
Rotation

- The point c_{ijo} rotates about the axis e_{ij} to create a circle γ



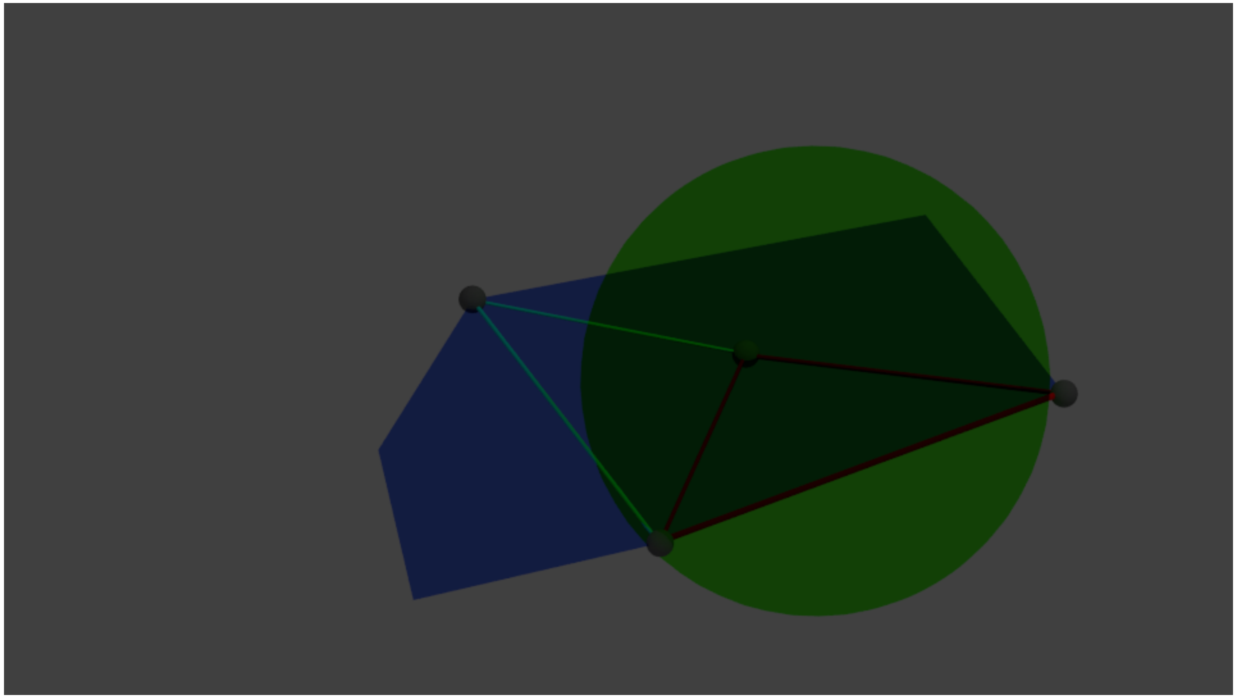
Rotation

- The point c_{ijo} rotates about the axis e_{ij} to create a circle γ



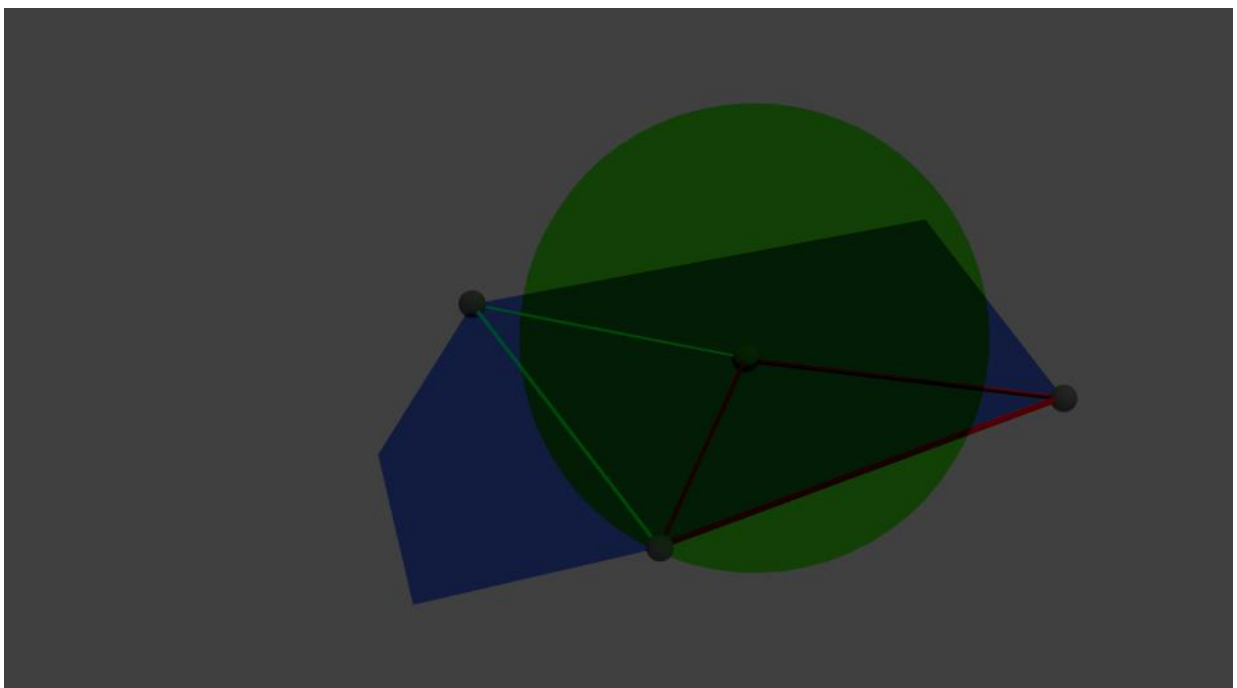
Rotation

- The point c_{ijo} rotates about the axis e_{ij} to create a circle γ



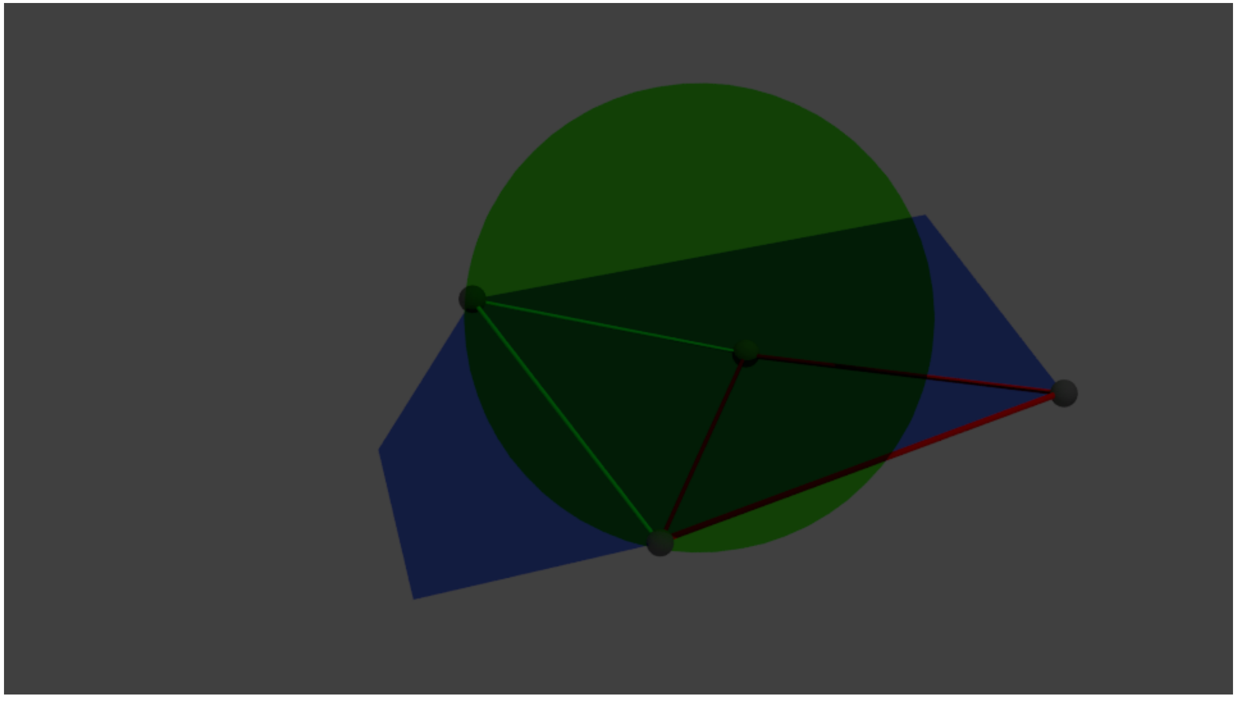
Rotation

- The point c_{ijo} rotates about the axis e_{ij} to create a circle γ



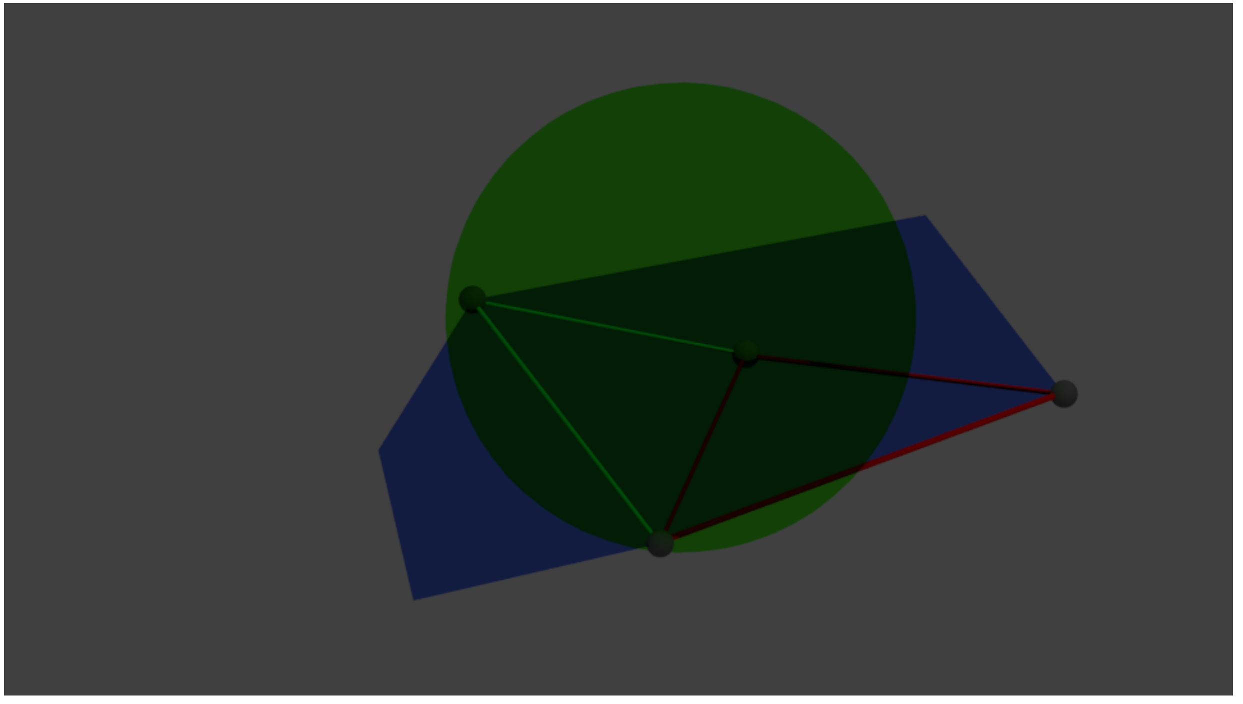
Rotation

- The point c_{ijo} rotates about the axis e_{ij} to create a circle γ



Rotation

- The point c_{ijo} rotates about the axis e_{ij} to create a circle γ



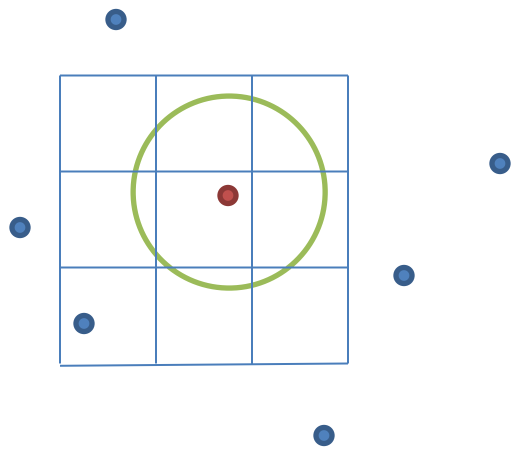
Intersection

Finding σ_k :

- For each vertex in a 2ρ-neighborhood of m compute the center c_x of a ball touching σ_i, σ_j, and σ_x
- Each c_x lies on γ and can be computed by intersecting a ρ-sphere centered at σ_x with γ
- Select the first point c_x on the trajectory γ

Intersection

- The vertices are stored in a voxel grid
  - Each voxel covers ρ^d ($\rho x \rho x \rho$ for $\mathbb{R}^3$)
- Only 9 voxels need to be checked



Overview

1. Find seed triangle
2. Pivot to closest point
3. Add triangle to mesh
4. Add edges to front F
5. Pivot around both edges

Limitations

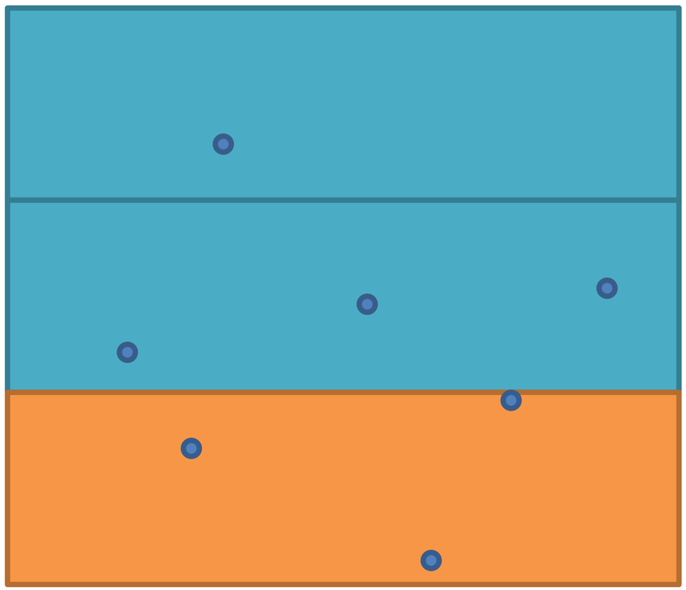
- The size of ρ changes the surface
  - Small ρ can pass through the surface
  - Large ρ can miss finer triangles

Alpha shapes

- The algorithm is similar to alpha shapes except that no edges or vertices are produced
- The size of ρ needs to be chosen carefully

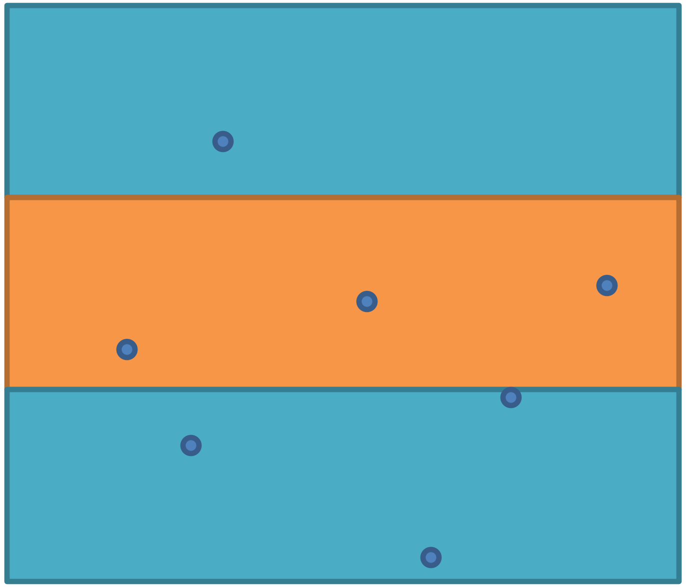
Out-of-core

- Algorithm easily amenable to low memory
- Break into sections



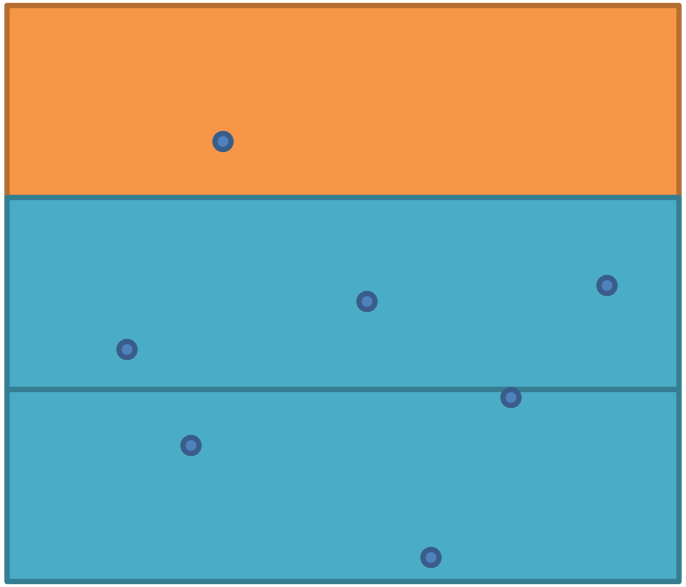
Out-of-core

- Algorithm easily amenable to low memory
- Break into sections



Out-of-core

- Algorithm easily amenable to low memory
- Break into sections



Out-of-core

- Edges that cross boundary are marked as frozen and used in the next section