# Voronoi Diagrams and Delaunay Triangulations

O'Rourke, Chapter 5

# Outline

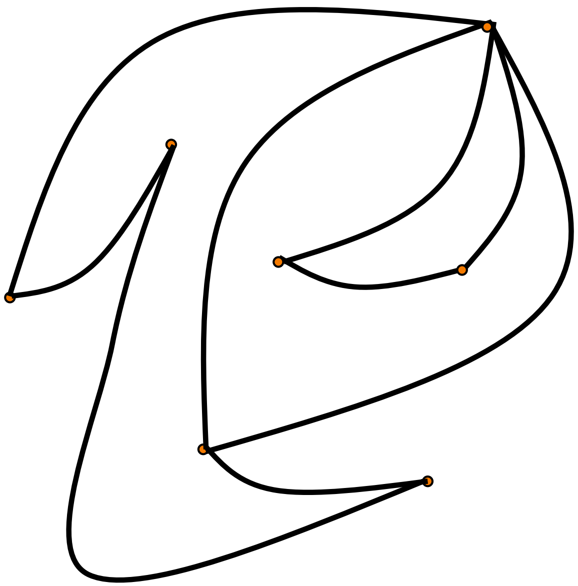
- Preliminaries
- Voronoi Diagrams / Delaunay Triangulations
- Lloyd's Algorithm

# Preliminaries

Claim:

Given a connected planar graph with $V$ vertices, $E$ edges, and $F$ faces*, the graph satisfies:

$$V - E + F = 2$$

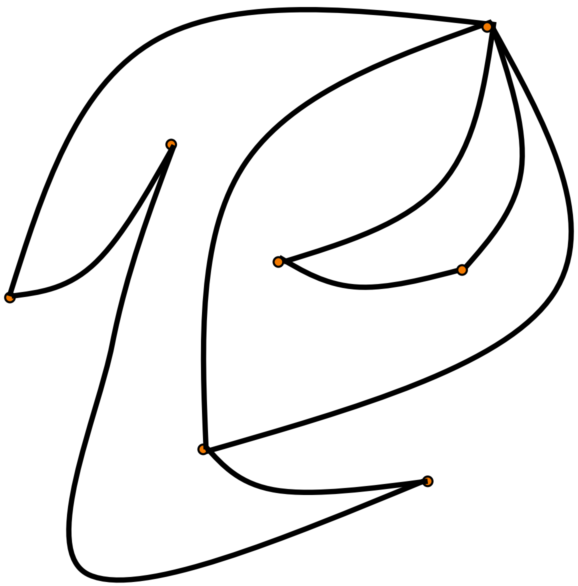


*The "external" face also counts. (Can think of this as a graph on the sphere.)

# Preliminaries

Proof:

1. Show that this is true for trees.
2. Show that this is true by induction.



# Preliminaries

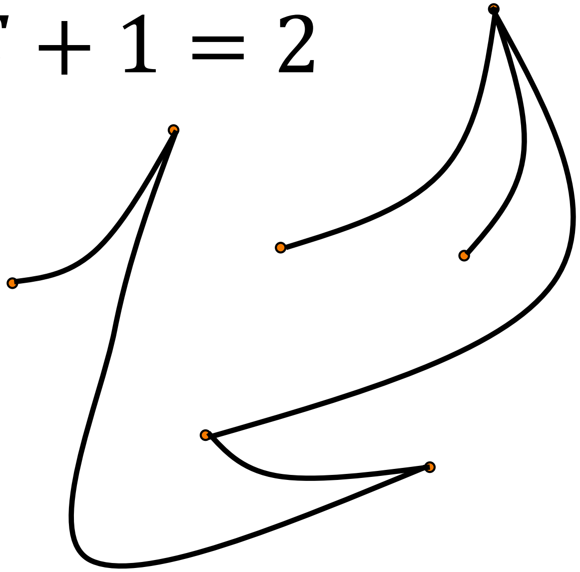
Proof (for Trees):

If a graph is a connected tree, it satisfies:

$$V = E + 1.$$

Since there is only one (external) face:

$$V - E + F = (E + 1) - E + 1 = 2$$

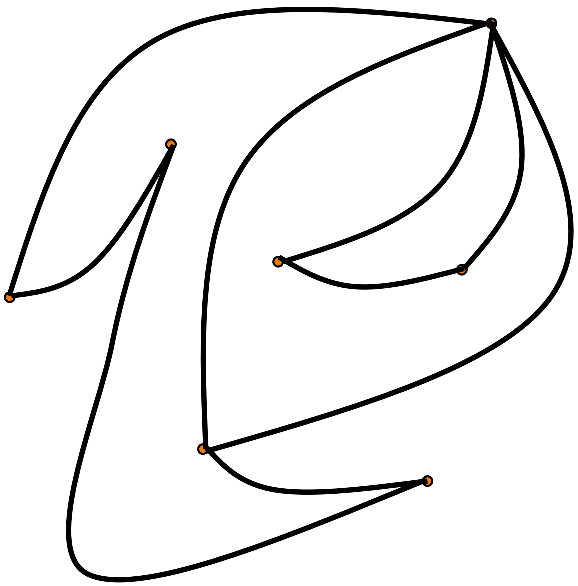


# Preliminaries

Proof (by Induction):

Suppose that we are given a graph $G$.

- If it's a tree, we are done.
- Otherwise, it has a cycle.



# Preliminaries

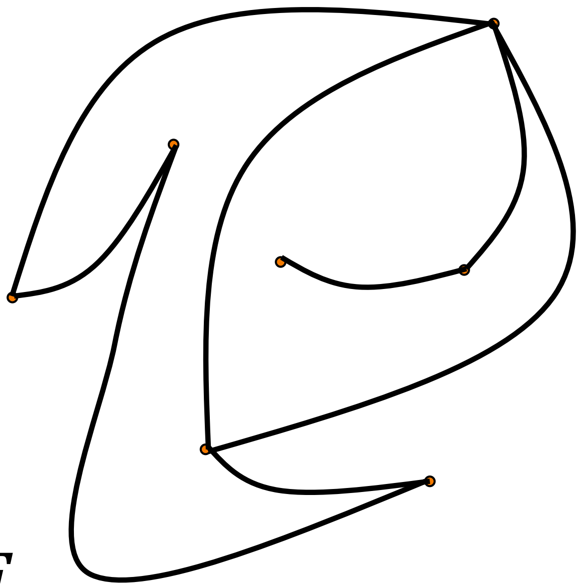
<u>Proof (by Induction)</u>:

Suppose that we are given a graph $G$.

- If it's a tree, we are done.
- Otherwise, it has a cycle.

Removing an edge on the cycle gives a graph $G'$ with:

- The same vertex set ($V' = V$)
- One less edge ($E' = E - 1$)
- One less face ($F' = F - 1$)



By induction:

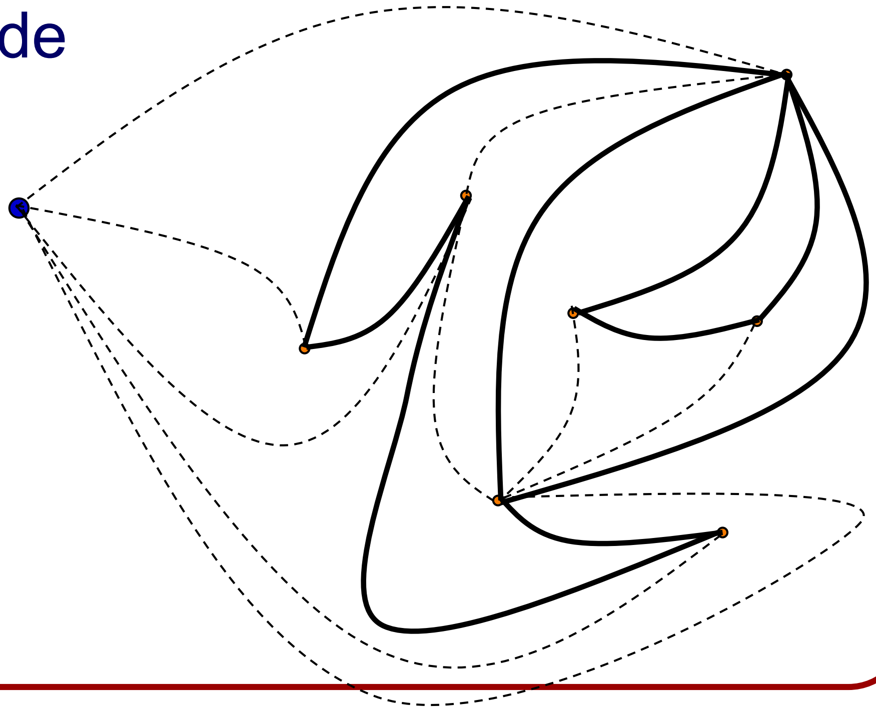
$$2 = V' - E' + F' = V - E + F$$

# Preliminaries

Note:

Given a planar graph $G$, we can get a planar graph $G'$ with triangle faces:

- Triangulate the interior polygons
- Add a "virtual point" outside and triangulate the exterior polygon.

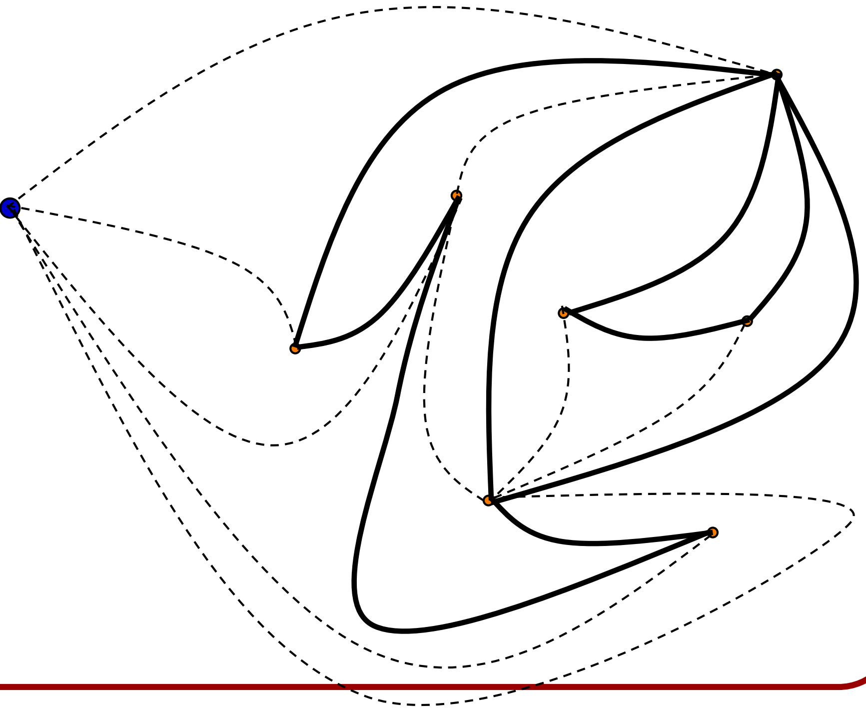


# Preliminaries

Note:

The new graph has:

- $V' = V + 1$, $E' \geq E$, $F' \geq F$
- $V' - E' + F' = 2$
- $3E' = 2F'$



# Preliminaries

Note:

The new graph has:

- $V' = V + 1$, $E' \geq E$, $F' \geq F$
- $V' - E' + F' = 2$
- $3E' = 2F'$

This gives:

$$E' = 3V' - 6 \quad\quad F' = 2V' - 4$$
$$\Downarrow \quad\quad \Downarrow$$
$$E \leq 3V - 3 \quad\quad F \leq 2V - 2$$

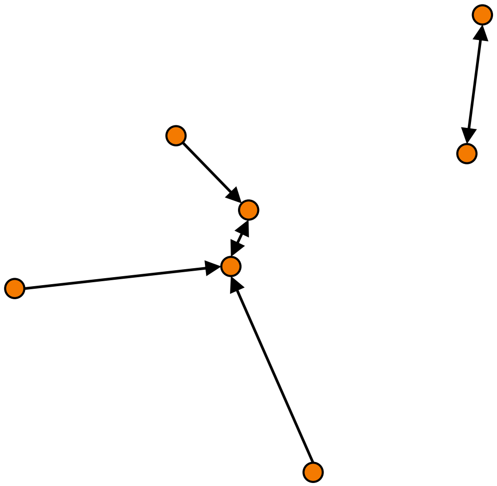
The number of edges/faces of a planar graph is linear in the number of vertices.

# Preliminaries

<u>Definition</u>:

Given a set of points $\{p_1, \ldots, p_n\} \subset \mathbb{R}^d$, the *nearest-neighbor graph* is the directed graph with an edge from $p_i$ to $p_j$, whenever:

$$\|p_k - p_i\| \geq \|p_j - p_i\| \quad \forall 1 \leq k \leq n.$$

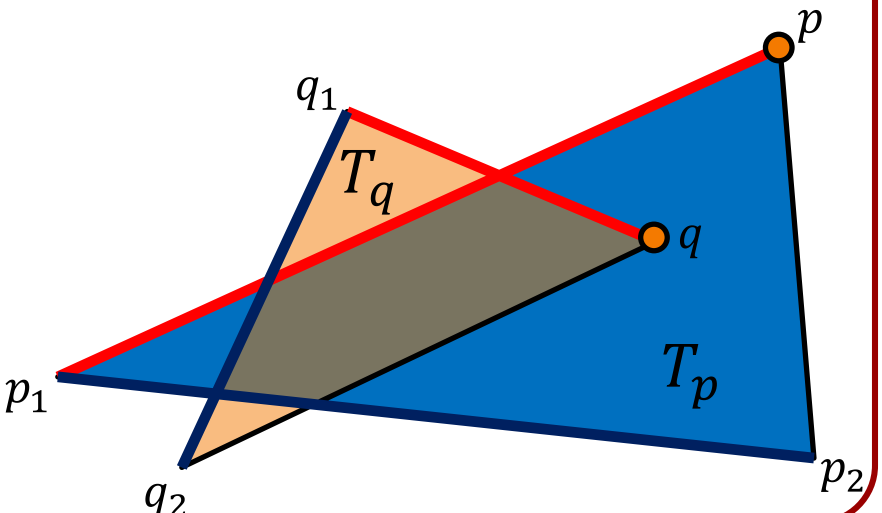


Naively, the nearest-neighbor can be computed in $O(n^2)$ time by testing all possible neighbors.

# Preliminaries

Claim:

Given triangles $T_p = \Delta p p_1 p_2$ and $T_q = \Delta q q_1 q_2$ such that the $p_i$ are not in $T_q$ and the $q_i$ are not in $T_p$, if the segments $\overline{p_1 p_2}$ and $\overline{q_1 q_2}$ intersect, there exist indices $i, j \in \{1,2\}$ such that $\overline{p p_i}$ and $\overline{q q_j}$ intersect.



# Preliminaries

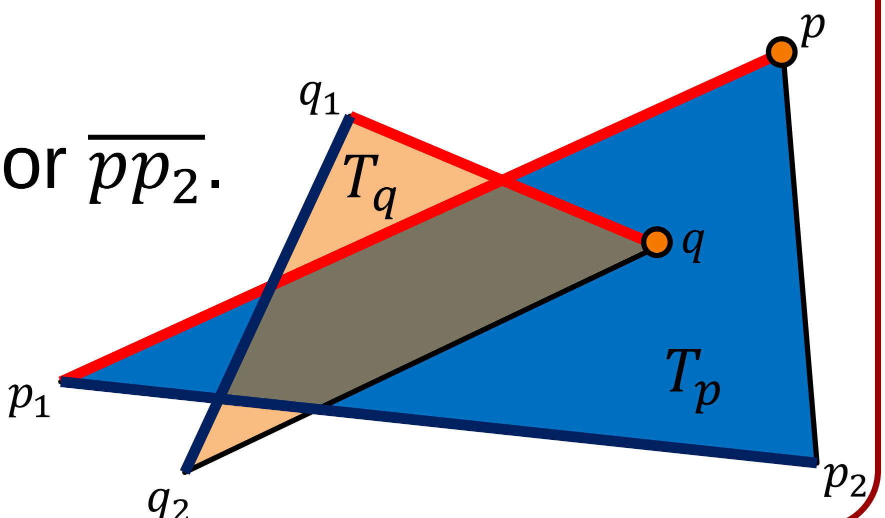
Proof: $(q \in T_p)$

Since $\overline{p_1 p_2} \cap \overline{q_1 q_2} \neq \emptyset$, one of the $q_i$ must be left of $\overline{p_1 p_2}$ and the other must be to the right.

Without loss of generality assume $q_1$ is left.

Then the edge $\overline{q q_1}$ is left of $\overline{p_1 p_2}$ and passes from inside $T_p$ to outside.

So it must cross either $\overline{p p_1}$ or $\overline{p p_2}$.

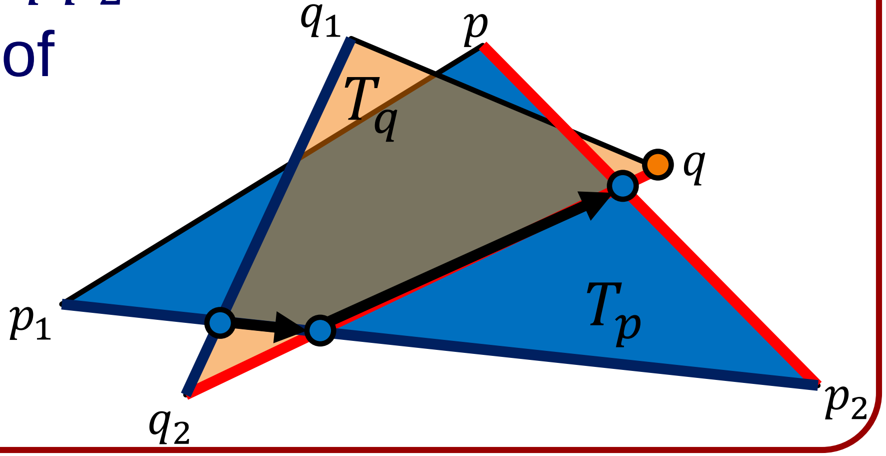


# Preliminaries

Proof: ($q \notin T_p$)

- Start at the intersection of $\overline{p_1 p_2}$ and $\overline{q_1 q_2}$.
- Advance along edge $\overline{p_1 p_2}$ while staying inside $T_q$.
- We hit edge of $\overline{q q_i} \subset T_q$, otherwise $p_1 \in T_q$ or $p_2 \in T_q$.
  Assume the edge we hit is $\overline{q q_2}$.
- Advance along edge $\overline{q q_2}$ while staying inside $T_p$.
- We hit edge of $\overline{p p_j} \subset T_p$, otherwise $q \in T_p$ or $q_2 \in T_p$.
  Assume the edge we hit is $\overline{p p_2}$.

$\Rightarrow$ We found an intersection of $\overline{p p_2}$ and $\overline{q q_2}$.



# Outline

- Preliminaries
- Voronoi Diagrams / Delaunay Triangulations
- Lloyd's Algorithm

# Voronoi Diagrams

Definition:

Given points $P = \{p_1, \ldots, p_n\}$, the *Voronoi region* of point $p_i$, $V(p_i)$, is the set of points at least as close to $p_i$ as to any other point in $P$:

$$V(p_i) = \left\{x \middle| |p_i - x| \leq \left|p_j - x\right| \forall 1 \leq j \leq n\right\}$$

# Voronoi Diagrams

<u>Definition</u>:

The set of points with more than one nearest neighbor in $P$ is the *Voronoi Diagram* of $P$:

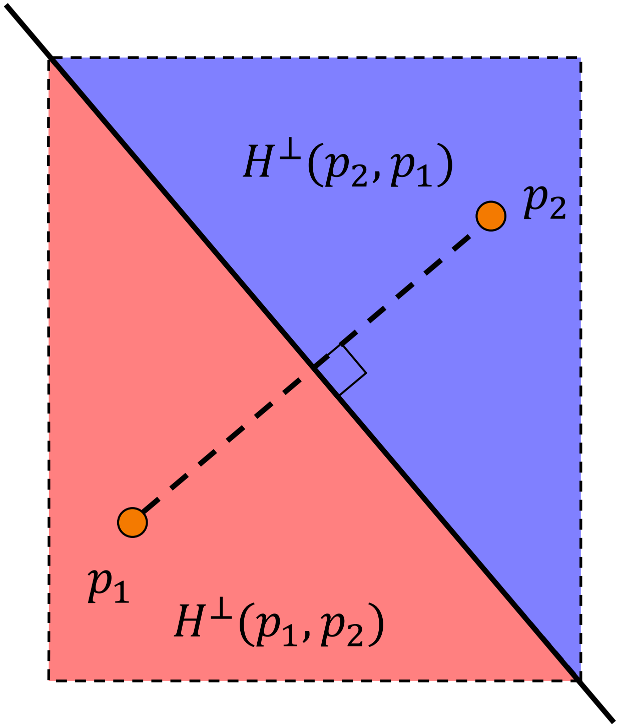
- The set with two nearest neighbors make up the *edges* of the diagram.
- The set with three or more nearest neighbors make up the *vertices* of the diagram.

The points $P$ are called the *sites* of the Voronoi diagram.

# Voronoi Diagrams

2 Points:

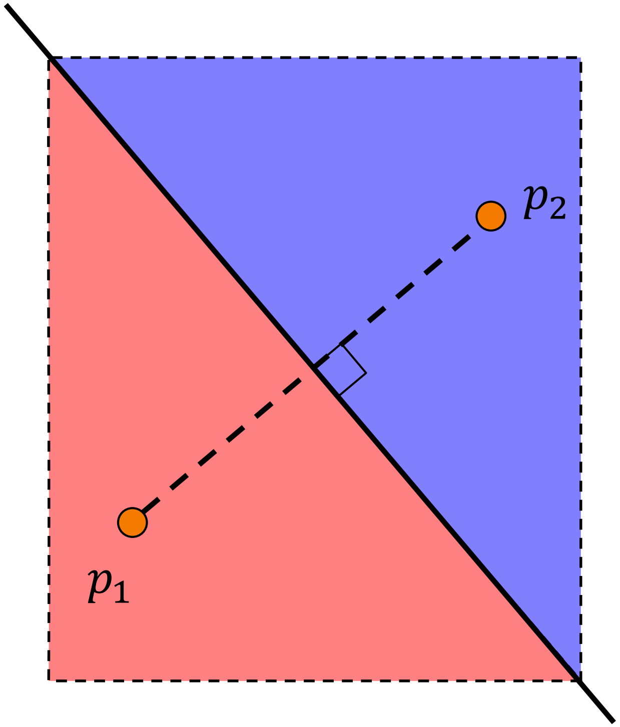
When $P = \{p_1, p_2\}$, the regions are defined by the perpendicular bisector:

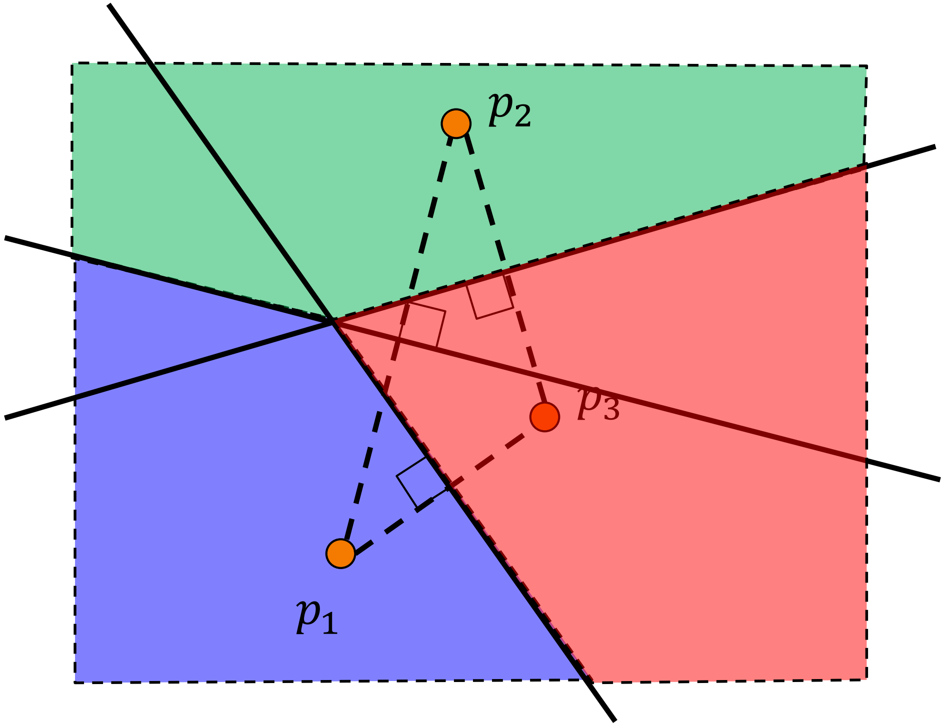


# Voronoi Diagrams

3 Points:

When $P = \{p_1, p_2, p_3\}$, the regions are defined by the three perpendicular bisectors:



# Voronoi Diagrams

3 Points:

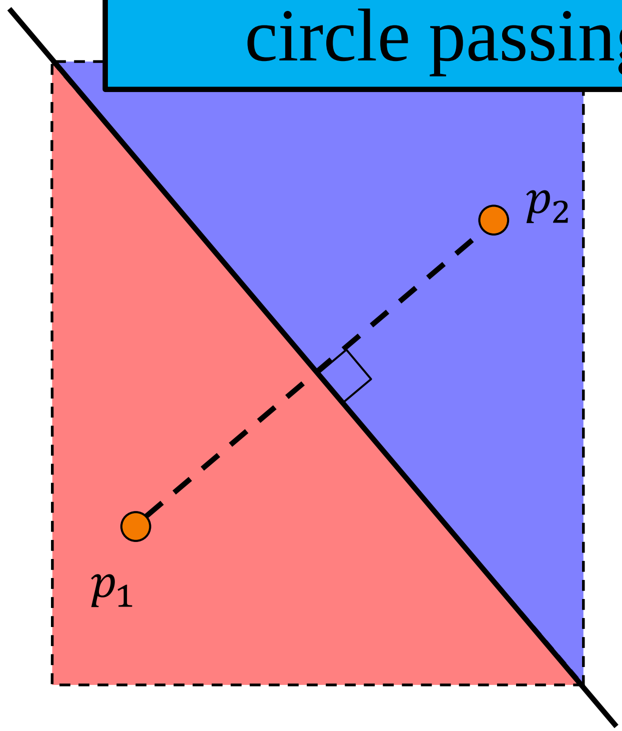
The three bisectors intersect at a point

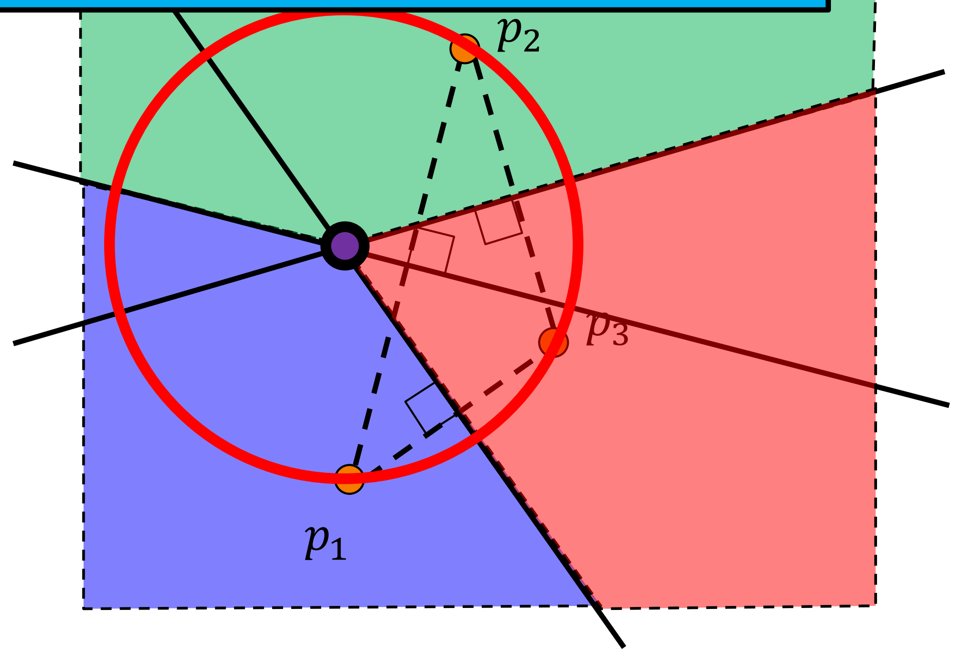
Whe … efined

The intersection can be outside the triangle.

by …

The point of intersection is the center of the circle passing through the three points.



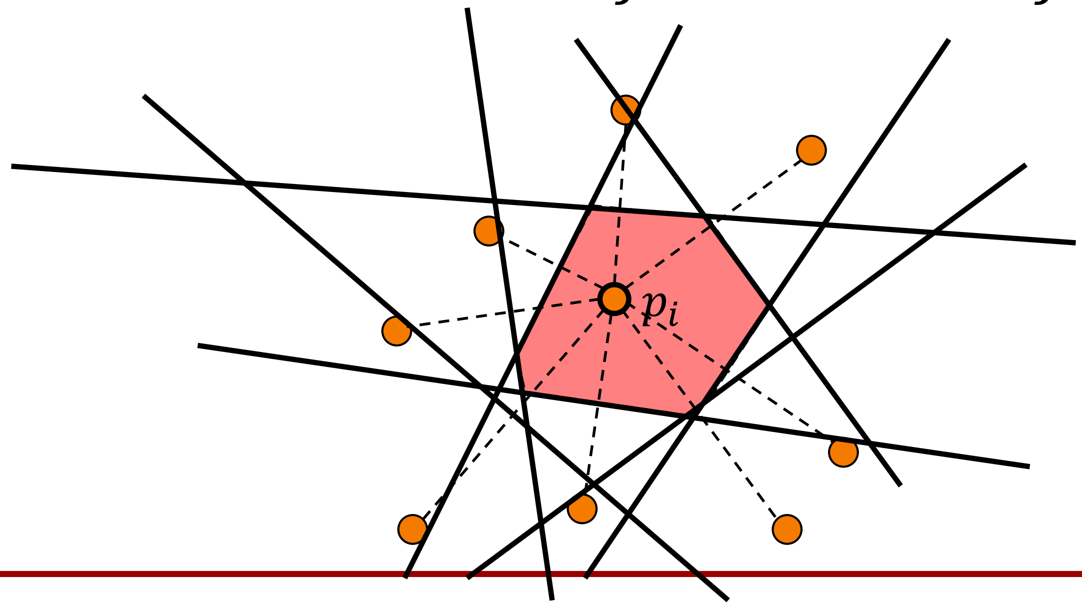



# Voronoi Diagrams

More Generally:

The Voronoi region associated to point $p_i$ is the intersection of the half-spaces defined by the perpendicular bisectors:

$$V(p_i) = \cap_{j \neq i} H^{\perp}(p_i, p_j)$$

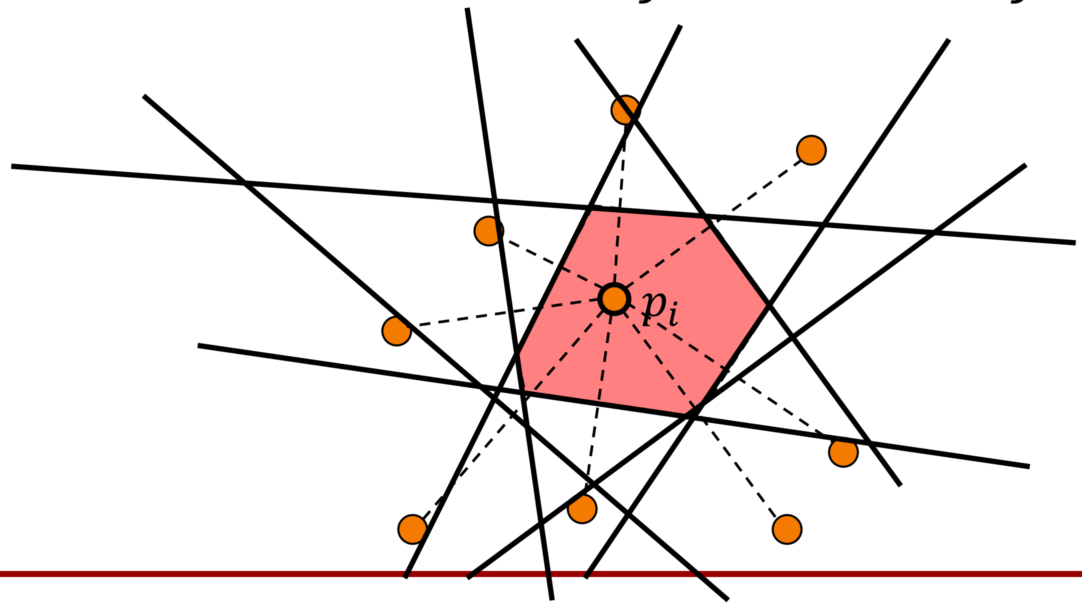


# Voronoi Diagrams

More Generally:

The Voronoi region associated to point $p_i$ is the intersection of the half-spaces defined by the perpendicular bisectors:

⇒ Voronoi regions are convex polygons.

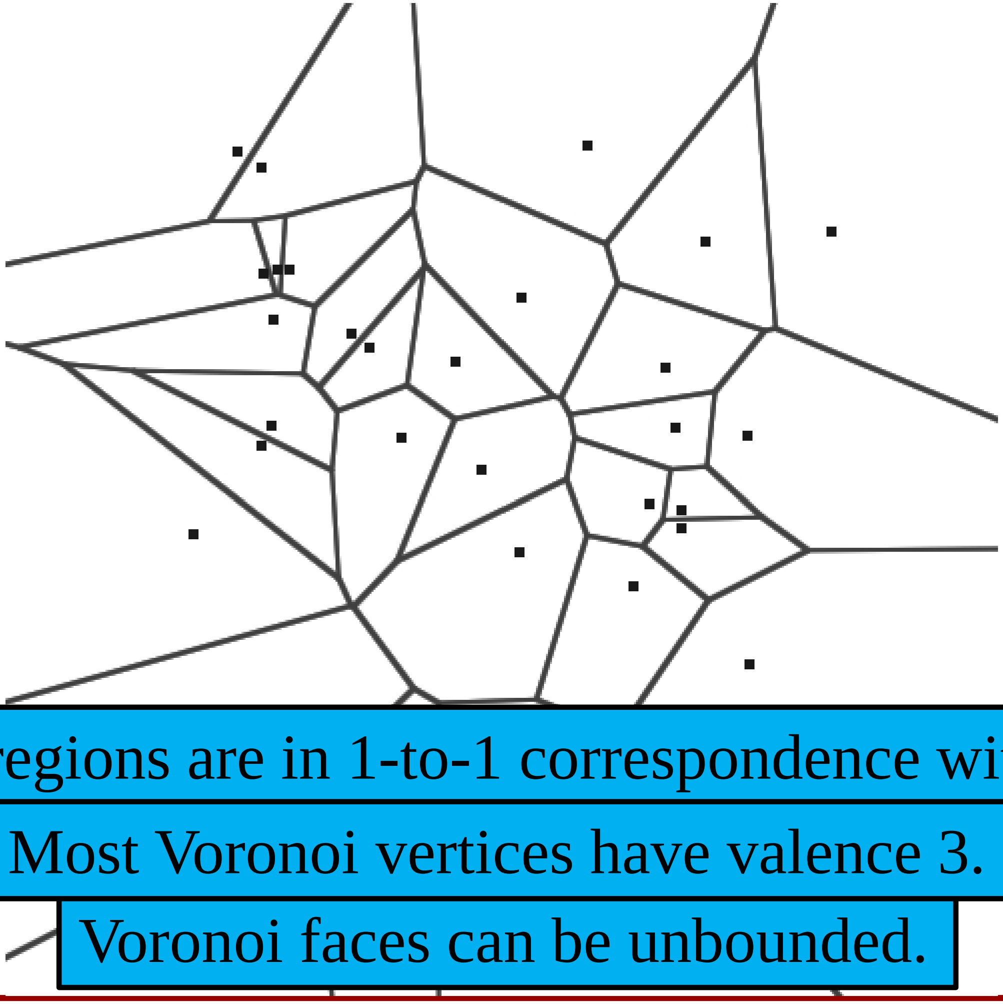


# Voronoi Diagrams

More Generally:

# Voronoi Diagrams

More Generally:



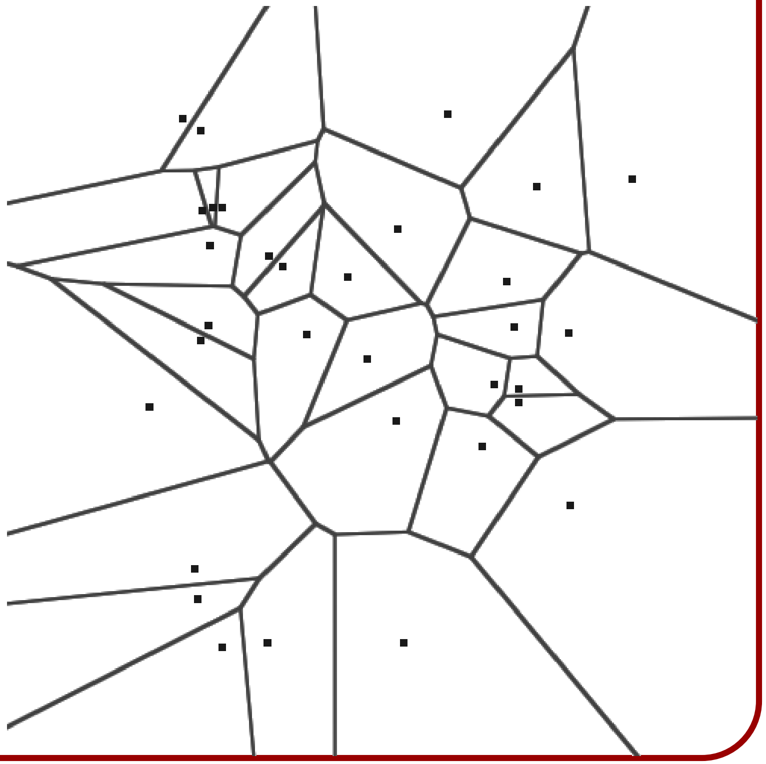
Voronoi regions are in 1-to-1 correspondence with points.

Most Voronoi vertices have valence 3.

Voronoi faces can be unbounded.

# Voronoi Diagrams

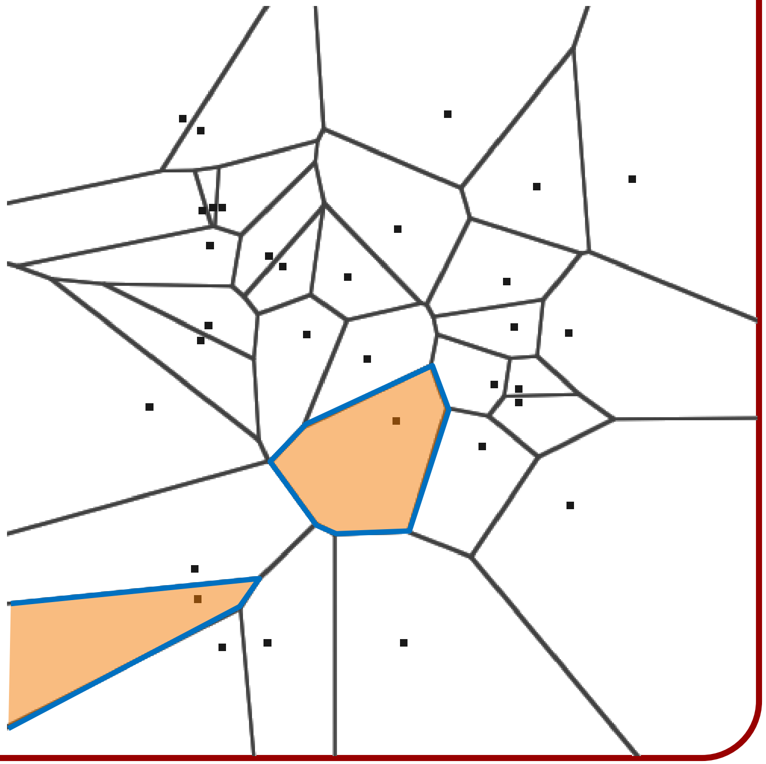
Properties:



# Voronoi Diagrams

Properties:

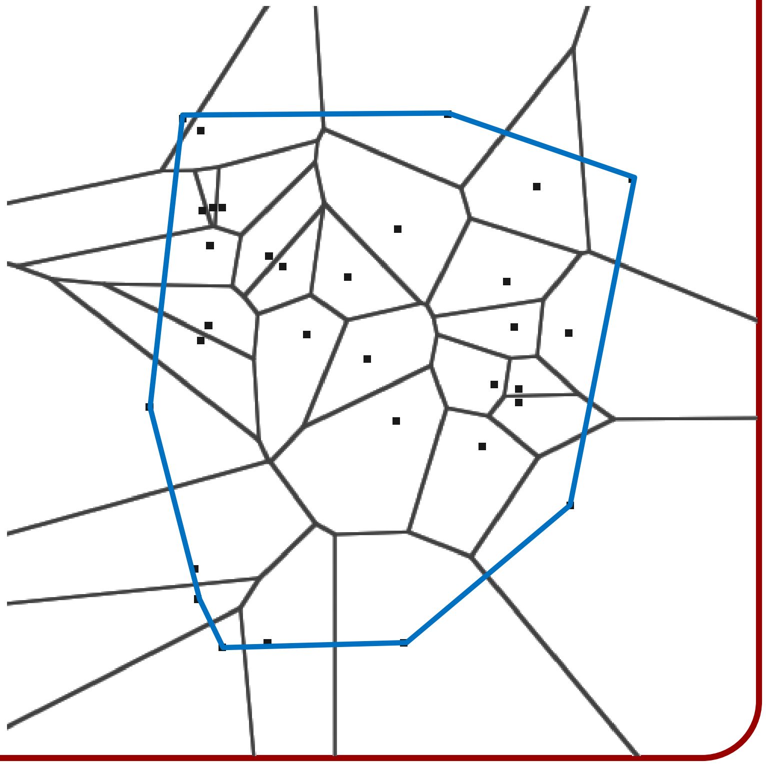
- Each Voronoi region is convex.



# Voronoi Diagrams

Properties:

- Each Voronoi region is convex.
- $V(p_i)$ is unbounded $\Leftrightarrow$ $p_i$ is on the convex hull of $P$.



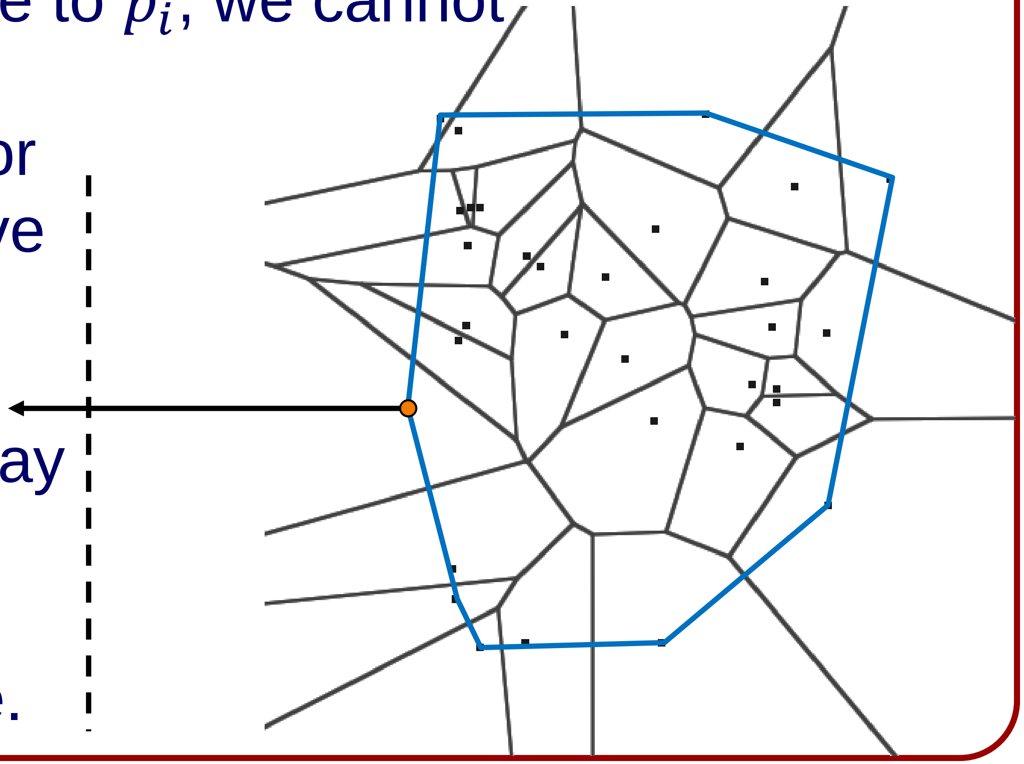
# Voronoi Diagrams

Proof (⇒):

Draw a ray from $p_i$ to infinity in the interior of $V(p_i)$ and consider the perpendicular hyperplanes:

- At infinity, the points of $P$ are on one side.
- Sliding the hyperplane to $p_i$, we cannot hit a point $q \neq p_i$. Otherwise the bisector of $p_i$ and $q$ would have to intersect the ray, contradicting the assumption that the ray is interior to $V(p_i)$.

⇒ At $p_i$ the hyperplane has $P$ all on one side.



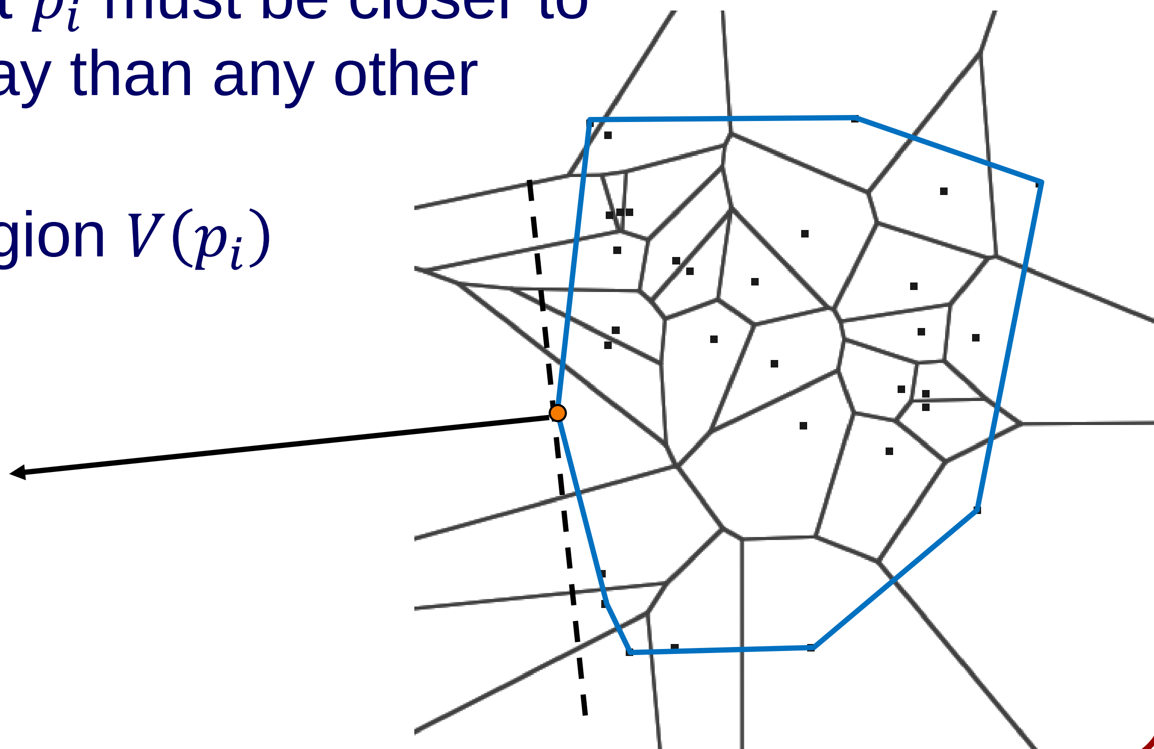
# Voronoi Diagrams

Proof (⇐):

If $p_i$ is on the convex hull, we can find a hyperplane through $p_i$ with $P$ all on one side.

Drawing the perpendicular ray through $p_i$ in the opposite direction, the point $p_i$ must be closer to any point on the ray than any other point in $P$.

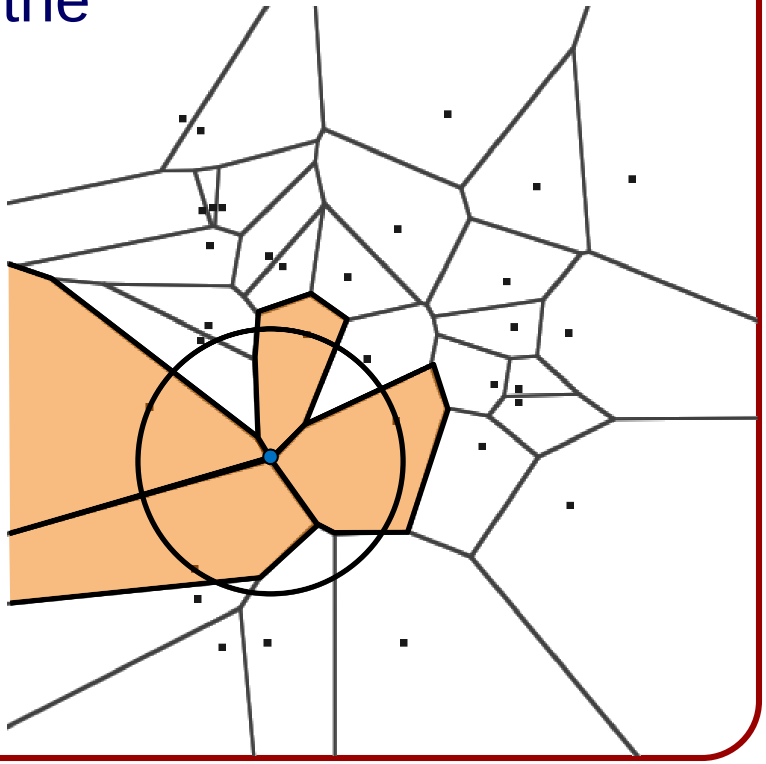
⇒ The Voronoi region $V(p_i)$ is unbounded.



# Voronoi Diagrams

Properties:

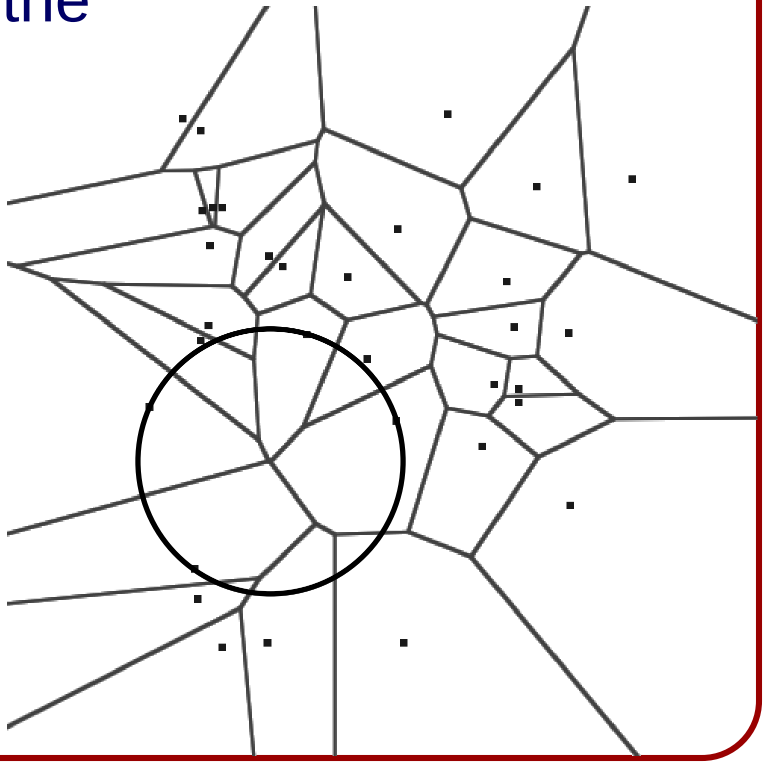
- Each Voronoi region is convex.
- $V(p_i)$ is unbounded $\Leftrightarrow$ $p_i$ is on the convex hull of $P$.
- If $v \in V(p_1) \cap \cdots \cap V(p_k)$ then $v$ is the center of a circle, $C(v)$, with $p_1, \ldots, p_k$ on the boundary.



# Voronoi Diagrams

Properties:

- Each Voronoi region is convex.
- $V(p_i)$ is unbounded $\Leftrightarrow$ $p_i$ is on the convex hull of $P$.
- If $v \in V(p_1) \cap \cdots \cap V(p_k)$ then $v$ is the center of some circle, $C(v)$, with $p_1, \ldots, p_k$ on the boundary.
- The interior of $C(v)$ contains no points.



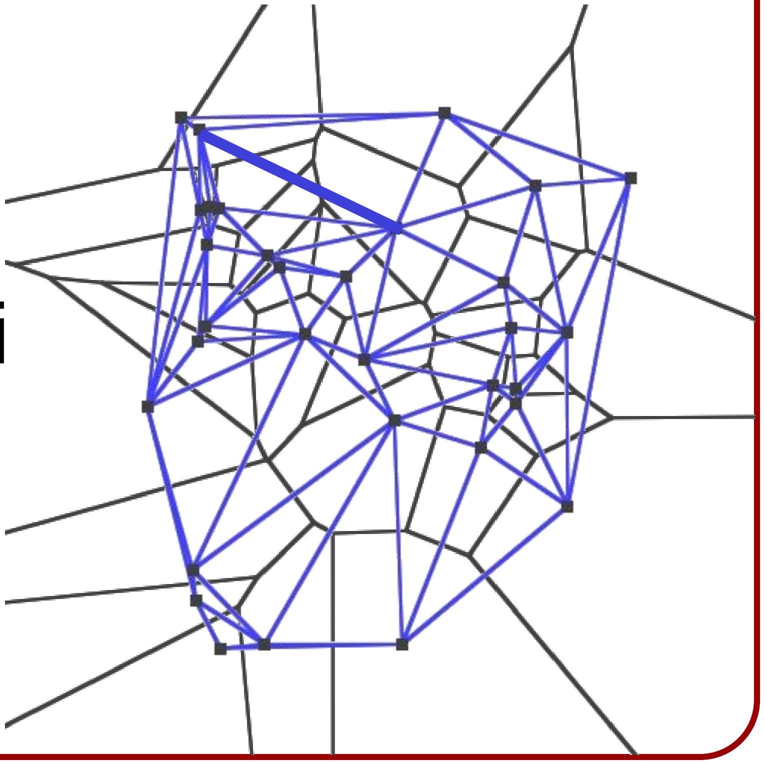
# Delaunay Triangulation

Definition:

The *Delaunay triangulation* is the straight-line dual of the Voronoi Diagram.

Note:

The Delaunay edges don't have to cross their Voronoi duals.



# Delaunay Triangulation

Properties:

- There is a circle through $p_i$ and $p_j$ that does not contain any other points $\Leftrightarrow$ $\overline{p_i p_j}$ is a Delaunay edge.
- The circumcircle of $p_i$, $p_j$, and $p_k$ does not contain any other points $\Leftrightarrow$ $\Delta p_i p_j p_k$ is a Delaunay triangle.
- The edges of the convex hull are in $D(P)$.
- If $p_j$ is the nearest neighbor of $p_i$ then $\overline{p_i p_j}$ is a Delaunay edge.
- The edges of $D(P)$ don't intersect.
- $D(P)$ is a triangulation if no 4 points are co-circular.

# Delaunay Triangulation

Properties:

- There is a circle through $p_i$ and $p_j$ that does not contain any other points $\Leftrightarrow$ $\overline{p_i p_j}$ is a Delaunay edge.
- The circumcircle of $p_i$, $p_j$, and $p_k$ does not contain any other points $\Leftrightarrow$ $\Delta p_i p_j p_k$ is a Delaunay triangle.
- T
- I … a D
- The edges of $D(P)$ don't intersect.
- $D(P)$ is a triangulation if no 4 points are co-circular.

Also:

- Maximizes minimum angle
- Defines the smoothest interpolant

# Delaunay Triangulation

Note:

Assuming that the edges of $D(P)$ do not cross, we get a planar graph.

$\Rightarrow$ The number of edges/faces in a Delaunay Triangulation is linear in the number of vertices.

$\Rightarrow$ The number of edges/vertices in a Voronoi Diagram is linear in the number of faces.

$\Rightarrow$ The number of vertices/edges/faces in a Voronoi Diagram is linear in the number of sites.

# Delaunay Triangulation

Properties:

- There is a circle through $p_i$ and $p_j$ that does not contain any other points $\Leftrightarrow$ $\overline{p_i p_j}$ is a Delaunay edge.

Proof ($\Leftarrow$):

If $\overline{p_i p_j}$ is a Delaunay edge, then the Voronoi regions $V(p_i)$ and $V(p_j)$ intersect at an edge.

Set $v$ to be some point on the interior of the edge.

$|v - p_i| = \left|v - p_j\right| = r$ and $|v - p_k| > r \ \forall k \neq i, j$.

The circle at $v$ with radius $r$ is empty of other points.

# Delaunay Triangulation

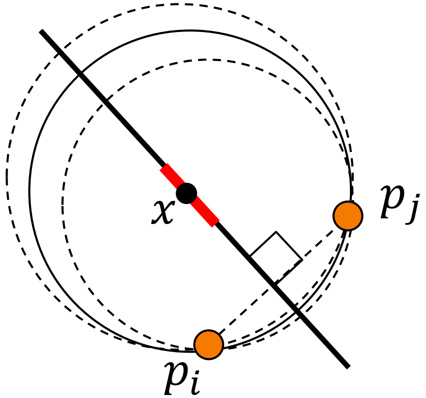
Properties:

- There is a circle through $p_i$ and $p_j$ that does not contain any other points $\Leftrightarrow$ $\overline{p_i p_j}$ is a Delaunay edge.

Proof ($\Rightarrow$):

If there is a circle through $p_i$ and $p_j$, empty of other points, with center $x$, then $x \in V(p_i) \cap V(p_j)$.

Since no other point is in or on the circle there is a neighborhood of centers around $x$ on the bisector with circles through $p_i$ and $p_j$ empty of other points.




$x$ is on a Voronoi edge.

# Delaunay Triangulation

Properties:

- The circumcircle of $p_i$, $p_j$, and $p_k$ does not contain any other points $\Leftrightarrow$ $\Delta p_i p_j p_k$ is a Delaunay triangle.

Proof:

There is a circle through $p_i$, $p_j$, and $p_k$ empty of points if and only if the intersections of the associated Voronoi regions, $V(p_i) \cap V(p_j) \cap V(p_k)$ is non-empty.

$\Leftrightarrow$ There is a Voronoi vertex with valence three

$\Leftrightarrow$ There is a dual Delaunay face with three sides

# Delaunay Triangulation

Properties:

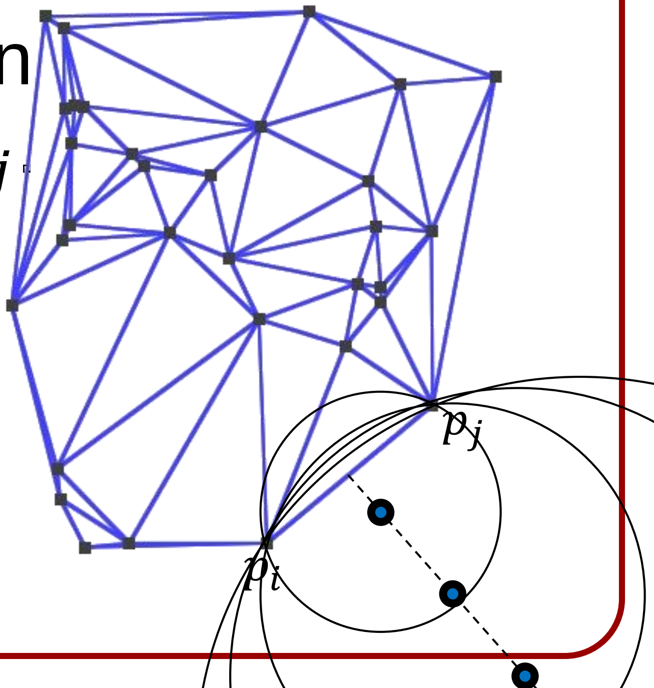
- The edges of the convex hull are in $D(P)$.

Proof:

Suppose that $\overrightarrow{p_i p_j}$ is an edge of the hull of $P$.

Consider empty circles centered on the bisector that intersect $p_i$ and $p_j$.

As we move out along the bisector the circle converges to the half-space to the right of $\overrightarrow{p_i p_j}$.



# Delaunay Triangulation

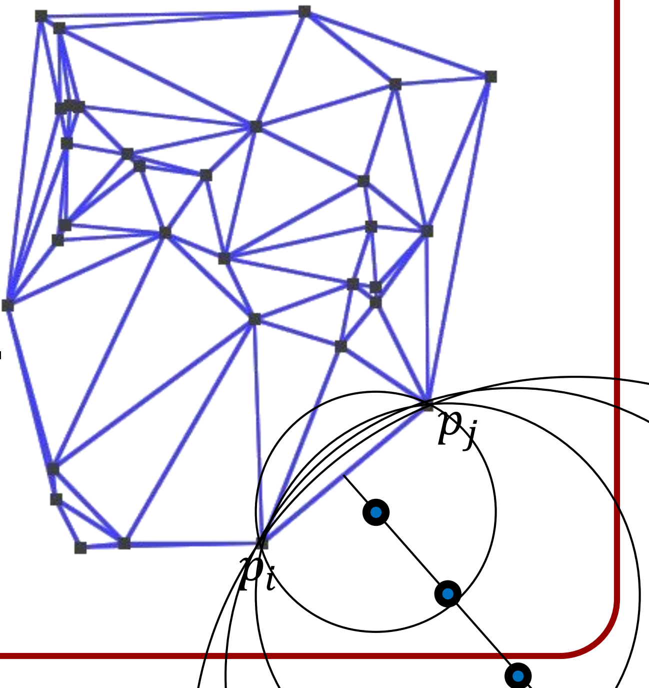
Properties:

- The edges of the convex hull are in $D(P)$.

Proof:

Suppose that $\overrightarrow{p_i p_j}$ is an edge of the hull of $P$.

$\Rightarrow$ Since the half space to right of $\overrightarrow{p_i p_j}$ is empty of points, in the limit we get a circle through $p_i$ and $p_j$ that is empty of points.




$\Rightarrow$ The edge $\overrightarrow{p_i p_j}$ is in $D(P)$.

# Delaunay Triangulation

Properties:

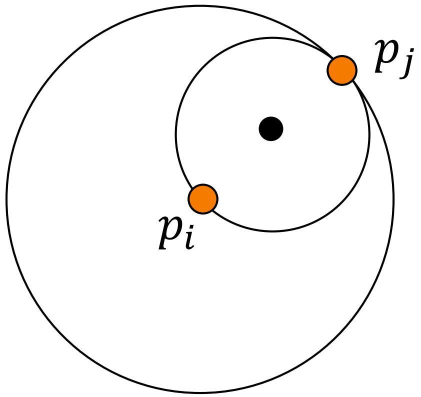
- If $p_j$ is the nearest neighbor of $p_i$ then $\overline{p_i p_j}$ is a Delaunay edge.

Proof:

$p_j$ is the nearest neighbor of $p_i$ iff. the circle around $p_i$ with radius $|p_i - p_j|$ is empty of other points.

$\Rightarrow$ The circle through $(p_i + p_j)/2$ with radius $|p_i - p_j|/2$ is empty of other points.

$\Rightarrow$ $\overline{p_i p_j}$ is a Delaunay edge.



# Delaunay Triangulation

Properties:

- If $p_j$ is the nearest neighbor of $p_i$ then $\overline{p_i p_j}$ is a Delaunay edge.

Implications:

The nearest neighbor graph is a subset of the Delaunay triangulation.

We will show that the Delaunay triangulation can be computed in $O(n \log n)$ time.

$\Rightarrow$ We can compute the nearest-neighbor graph in $O(n \log n)$.

# Delaunay Triangulation

Properties:

- The edges of $D(P)$ don't intersect.

Proof:

Given an edge $\overline{p_1p_2}$ in $D(P)$, there is a circle with $p_1$ and $p_2$ on its boundary and empty of other points.

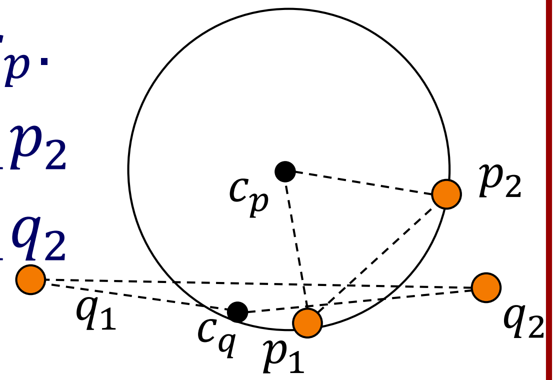
Let be $\overline{q_1q_2}$ be an edge in $D(p)$ that intersect $\overline{p_1p_2}$:

$q_1$ and $q_2$ cannot be in the circle about $c_p$.

$\Rightarrow$ $q_1$ and $q_2$ are not in the triangle $\Delta c_p p_1 p_2$

$\Rightarrow$ $p_1$ and $p_2$ are not in the triangle $\Delta c_q q_1 q_2$

$\Rightarrow$ One of $\overline{c_p p_1}$ or $\overline{c_p p_2}$ intersects one of $\overline{c_q q_1}$ or $\overline{c_q q_2}$.



# Delaunay Triangulation

Properties:

- The edges of $D(P)$ don't intersect.

Proof:

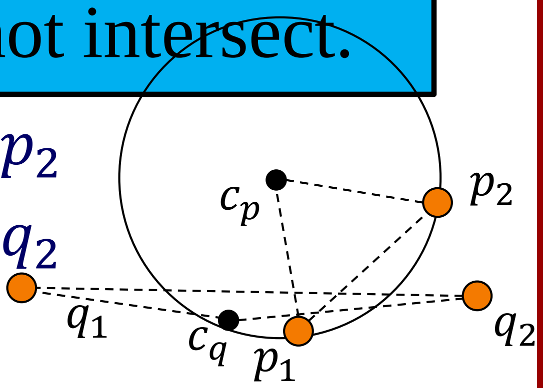
Given an edge $\overline{p_1 p_2}$ in $D(P)$, there is a circle with $p_1$ and $p_2$ on its boundary and empty of other points.

But $\overline{c_p p_i}$ is in the Voronoi region of $p_i$ and $\overline{c_q q_j}$ is in the Voronoi region of $q_j$, so they cannot intersect.

$\Rightarrow$ $q_1$ and $q_2$ are not in the triangle $\Delta c_p p_1 p_2$

$\Rightarrow$ $p_1$ and $p_2$ are not in the triangle $\Delta c_q q_1 q_2$

$\Rightarrow$ One of $\overline{c_p p_1}$ or $\overline{c_p p_2}$ intersects one of $\overline{c_q q_1}$ or $\overline{c_q q_2}$.



# Delaunay Triangulation

Properties:

- $D(P)$ is a triangulation if no 4 points are co-circular.

Proof:

If no four points are circular, every Voronoi vertex has valence three.

$\Leftrightarrow$ Every dual face in $D(P)$ has three sides.

# Outline

- Preliminaries
- Voronoi Diagrams / Delaunay Triangulations
  - Naive Algorithm
  - Fortune's Algorithm
- Lloyd's Algorithm

# Naive Algorithm

Delaunay( $\{p_1, \ldots, p_n\}$ )
    for $i \in [1, n]$
        for $j \in [1, i)$
            for $k \in [1, j)$
                $(c, r) \leftarrow$ Circumcircle( $p_i$ , $p_j$ , $p_k$ )
                isTriangle $\leftarrow$ true
                for $l \in [1, k)$
                    if( $\|p_l - c\| < r$ ) isTriangle $\leftarrow$ false
                if( isTriangle ) Output( $p_i$ , $p_j$ , $p_k$ )

Complexity: $O(n^4)$

# Voronoi Diagrams and Cones

Key Idea:

We can think of generating Voronoi regions by expanding circles centered at points of $P$.

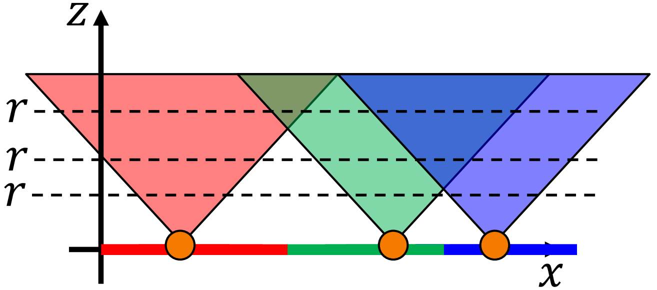
When multiple circles overlap a point, track the one whose center is closer.

# Voronoi Diagrams and Cones

Key Idea:

We can visualize the Voronoi regions by drawing right cones over the points, with axes along the positive $z$-axis.

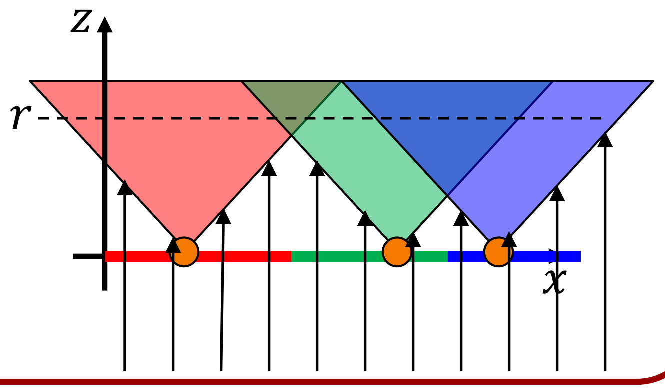
Circles with radius $r$ are the projections of the intersections of the plane $z = r$ plane with the cones, onto the $xy$-plane.



# Voronoi Diagrams and Cones

Key Idea:

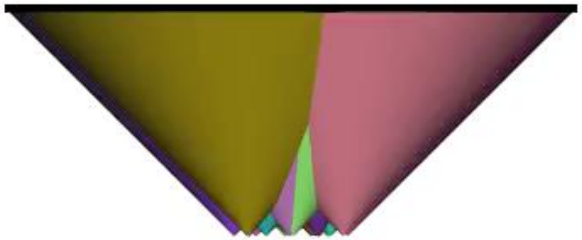
To track the circle with the closer center, we can render the cones with an orthographic camera looking up the $z$-axis.



# Voronoi Diagrams and Cones

Key Idea:

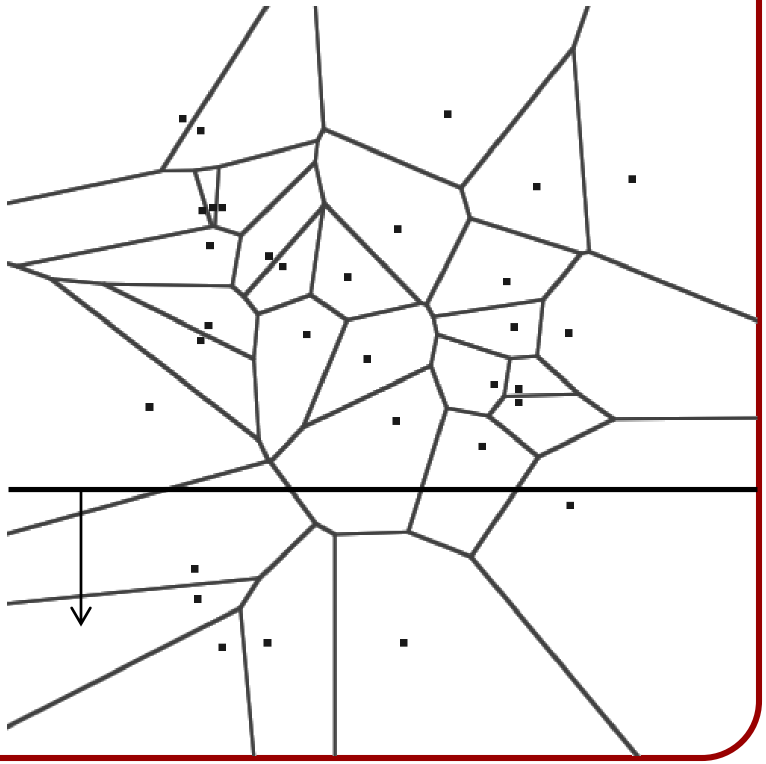
To track the circle with the closer center, we can render the cones with an orthographic camera looking up the $z$-axis.



# Fortune's Algorithm

Approach:

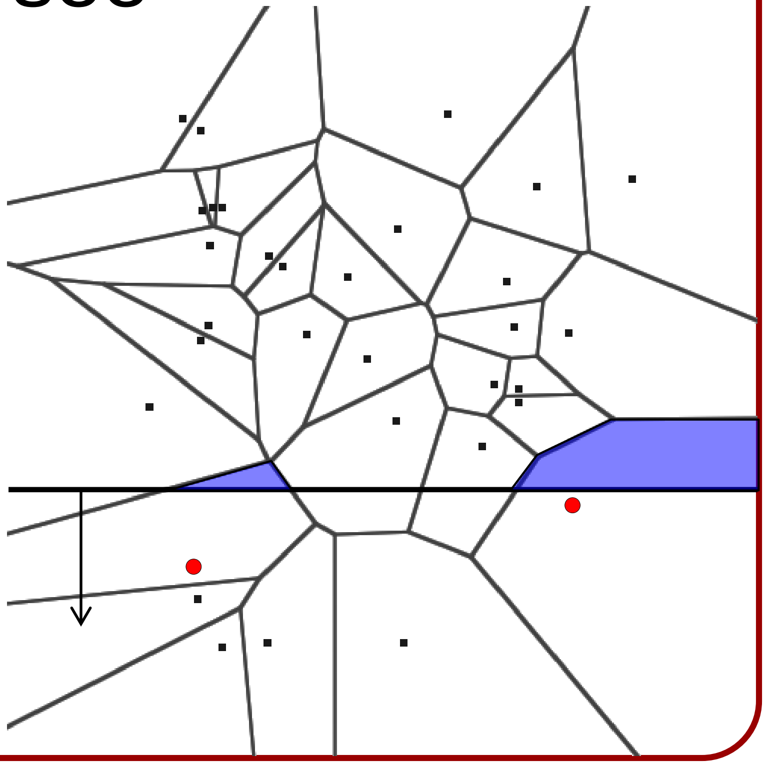
Sweep a line and maintain the solution for all points behind the line.



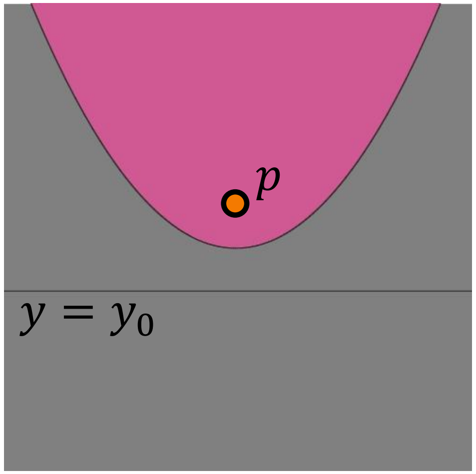
# Fortune's Algorithm

<u>Why This Shouldn't Work</u>:
The Voronoi region behind the line can depend on points that are in front of the line!
(Looking up the $z$-axis, we see the cone before the apex.)

<u>Key Idea</u>:
We can finalize points behind the line that are closer to a site than to the line.



# Fortune's Algorithm




Given a site $p \in P$ and the line with height $y_0$, we can finalize the points satisfying:

$$\{(x,y) \mid (y-y_0)^2 > \|p-(x,y)\|^2\}$$

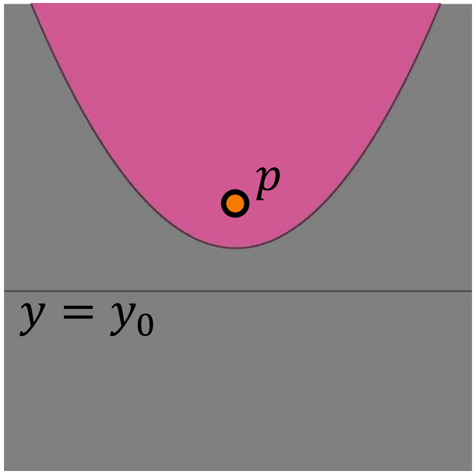
Points on the boundary satisfy:

$$(y-y_0)^2 = \|p-(x,y)\|^2$$

Setting $z = \|p-(x,y)\|$, points on the boundary satisfy:

$$z = y - y_0$$

# Fortune's Algorithm




Formally:

$\Rightarrow$ We can describe points on the boundary as the $xy$-coordinates of the points in 3D with:

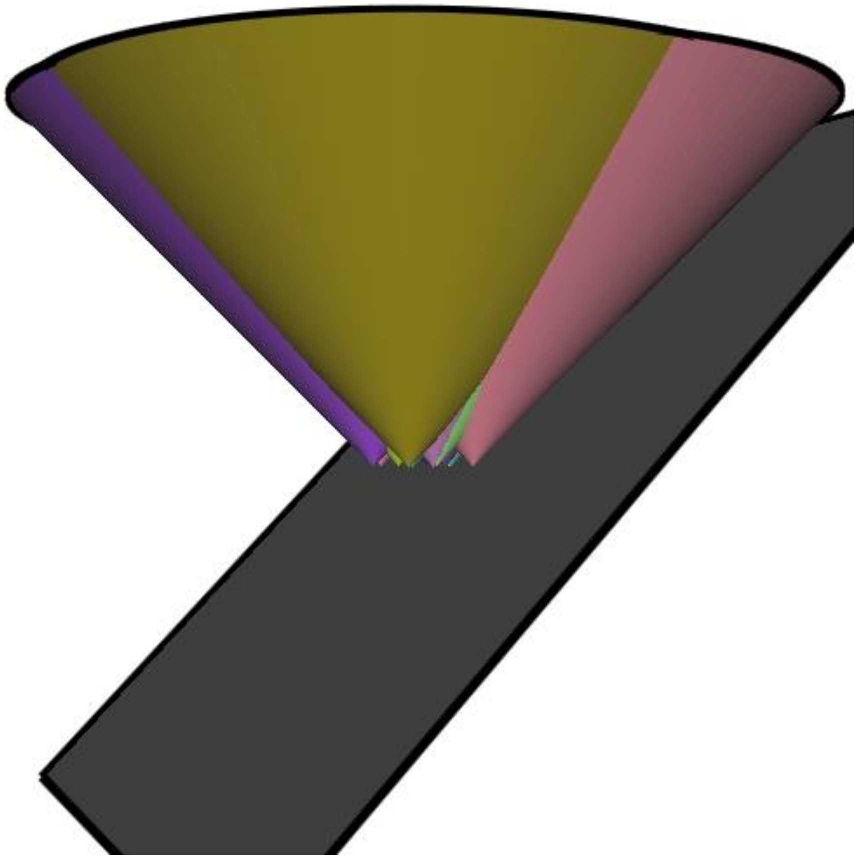
1. $z(x,y) = \|p - (x,y)\|$
2. $z(x,y) = y - y_0$

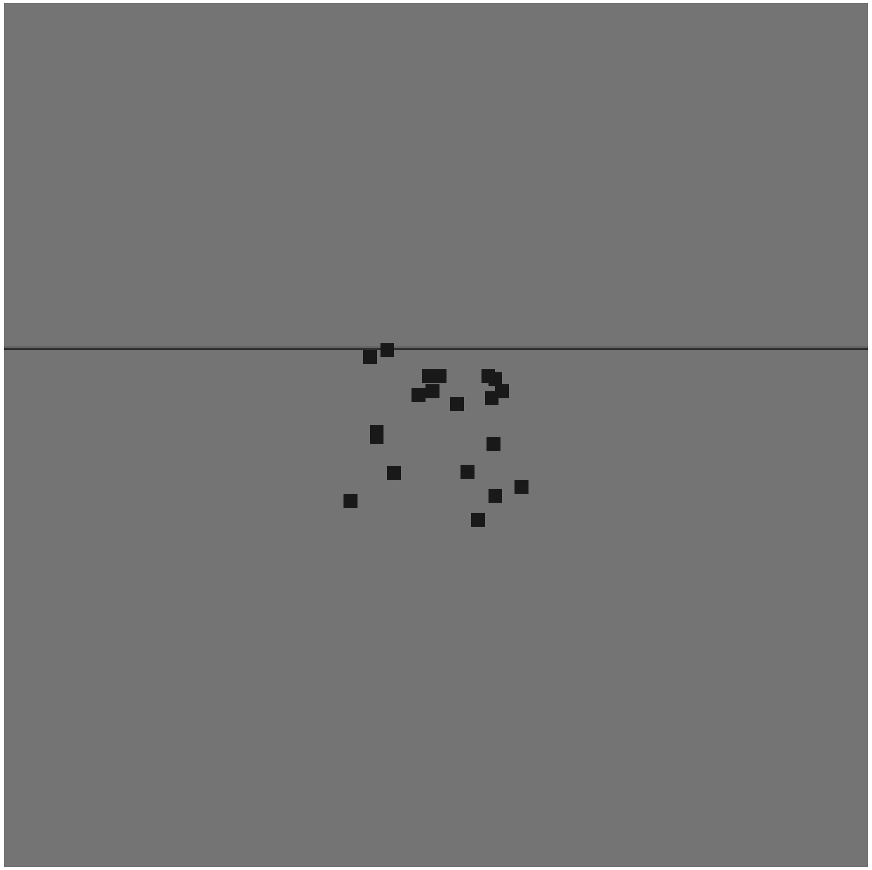
Points on the right cone, centered at $p$, centered around the positive $z$-axis

Points on the plane, making a 45° angle with the $xy$-plane, passing through the line $y = y_0$ and $z = 0$

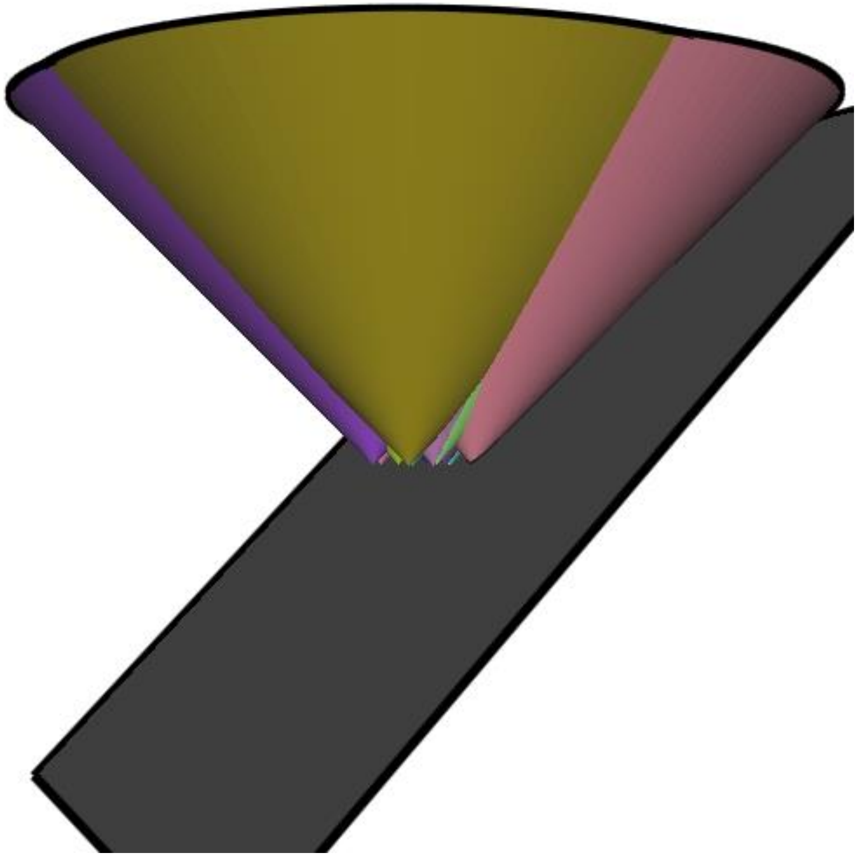
Sweep the cones with a plane parallel to the $x$-axis making a 45° angle with the $xy$-plane.

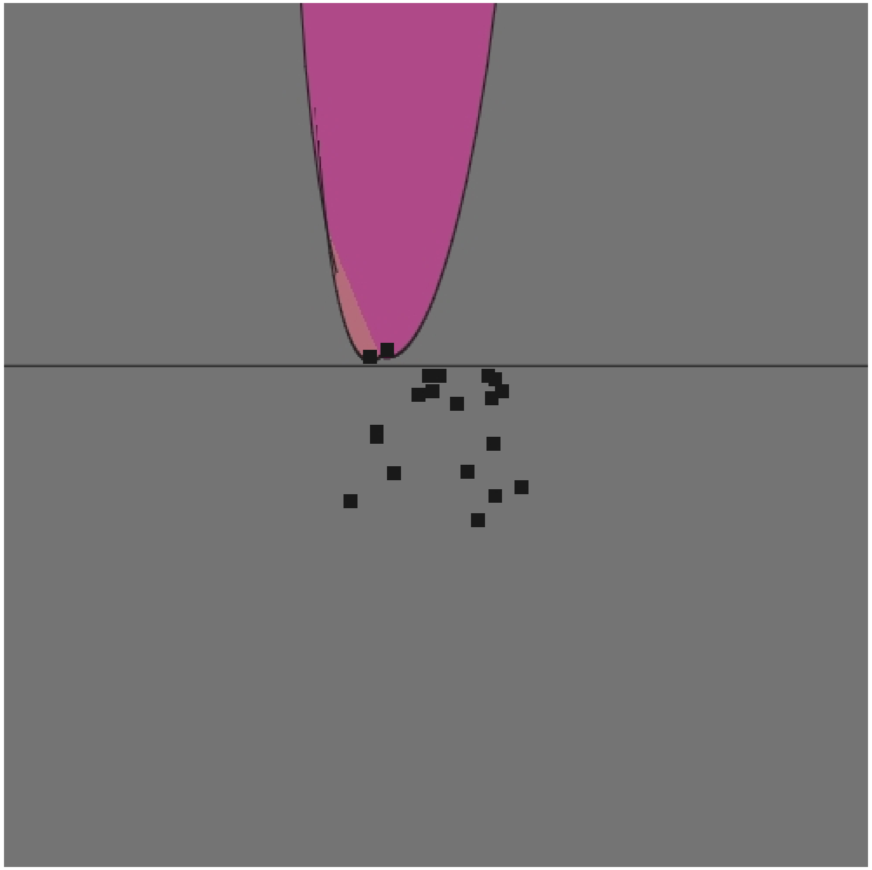
# Fortune’s Algorithm



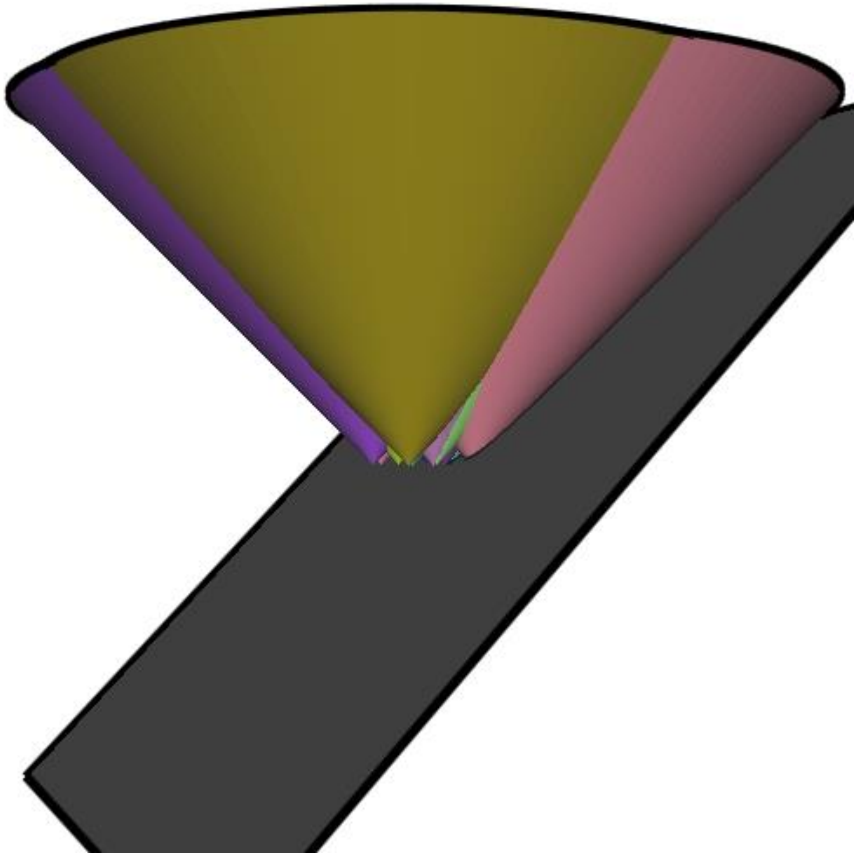


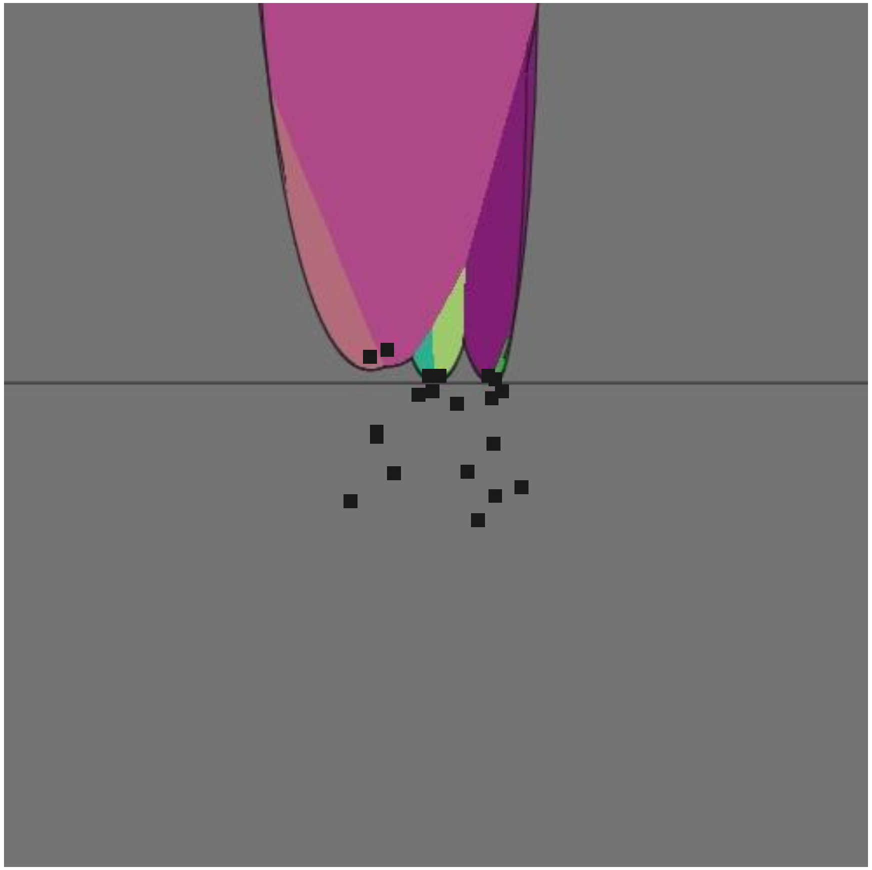
# Fortune's Algorithm



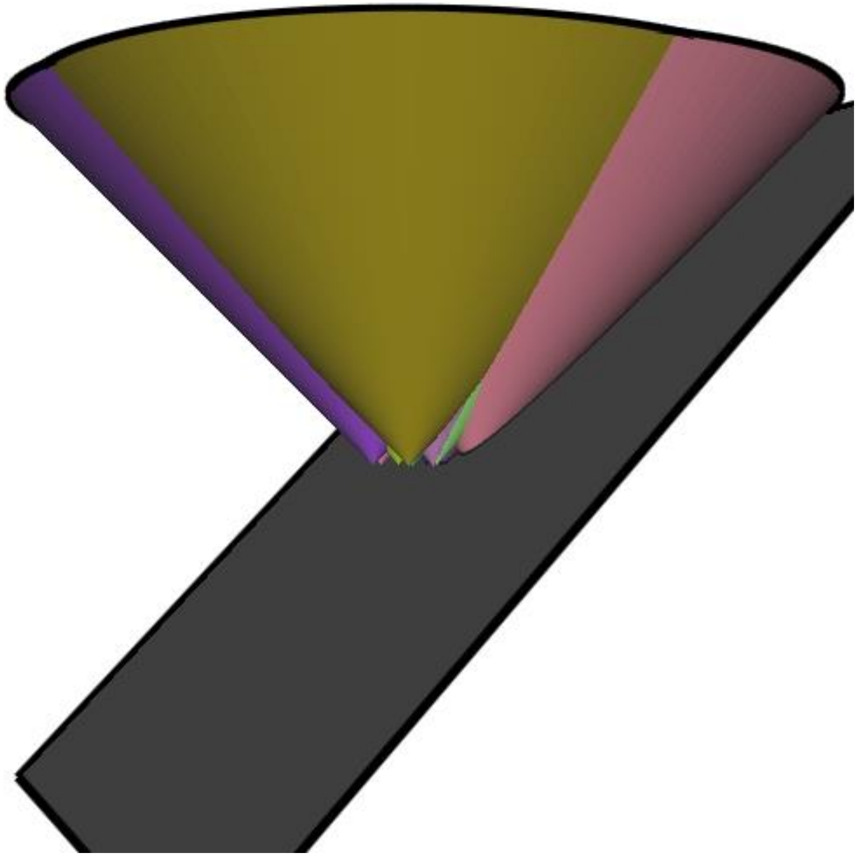


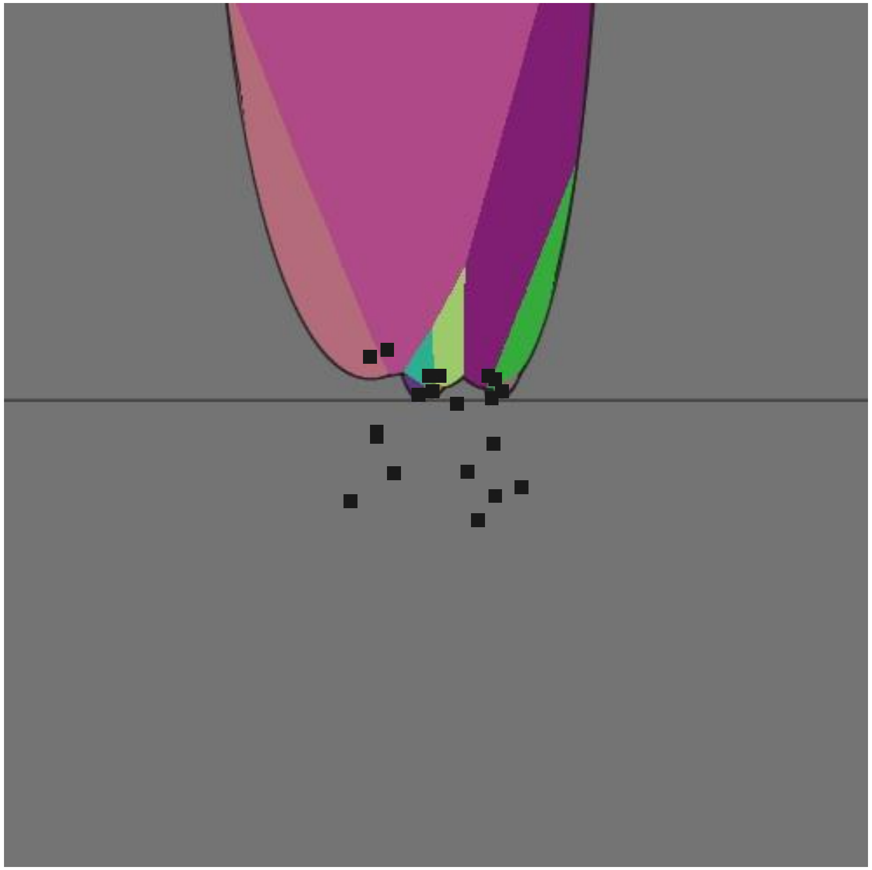
# Fortune's Algorithm



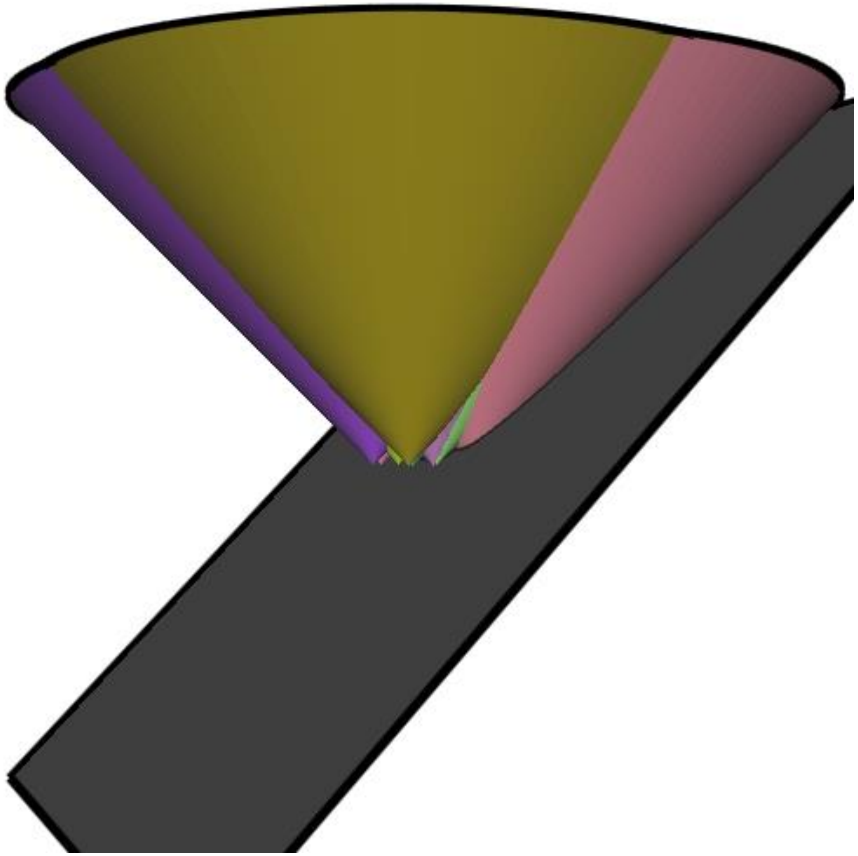


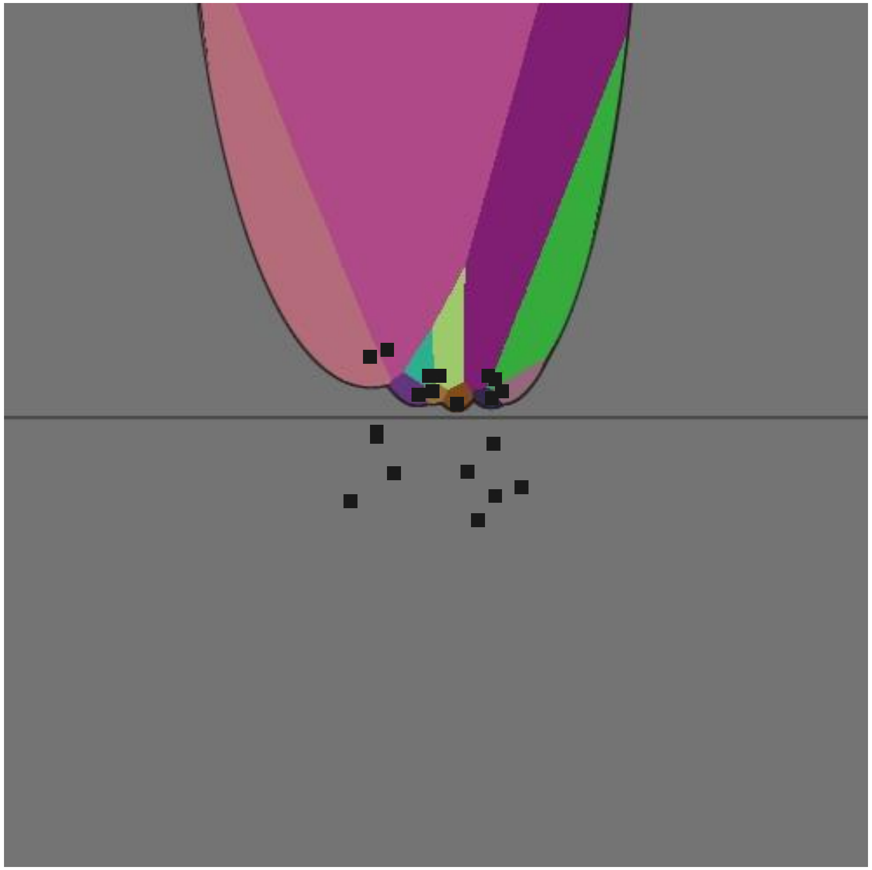
# Fortune's Algorithm



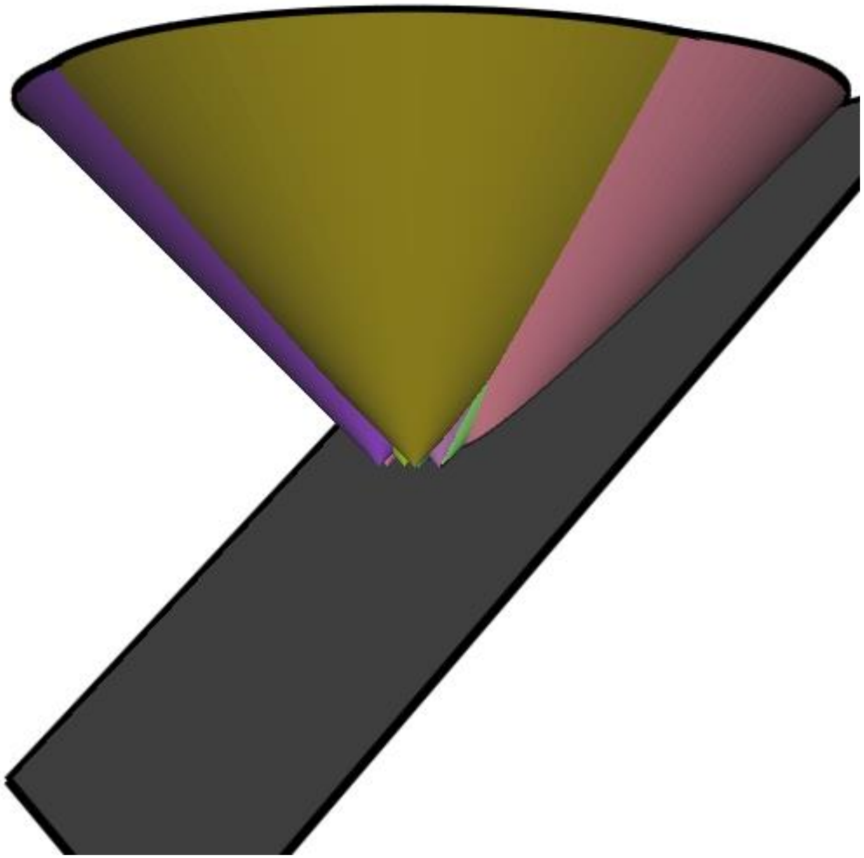


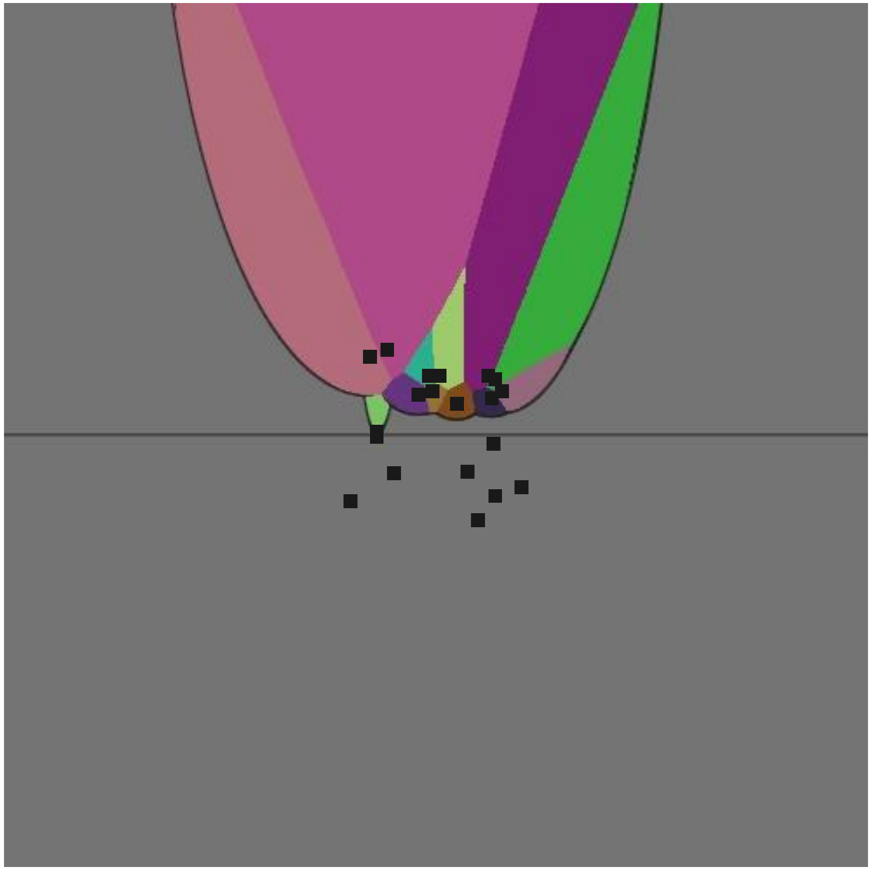
# Fortune's Algorithm



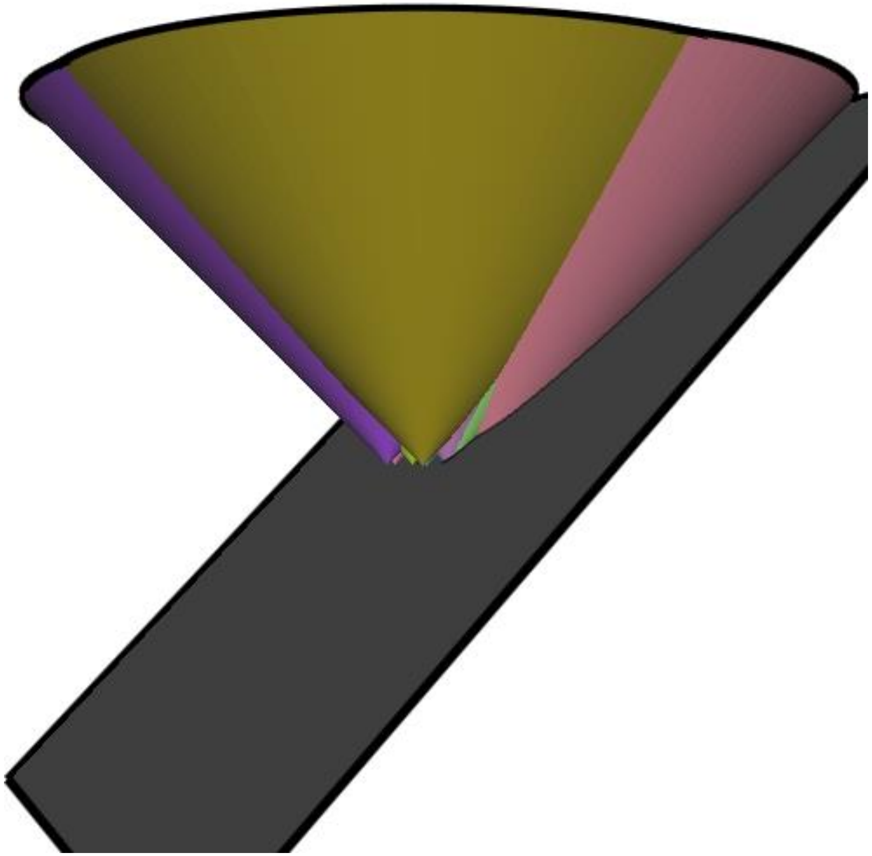


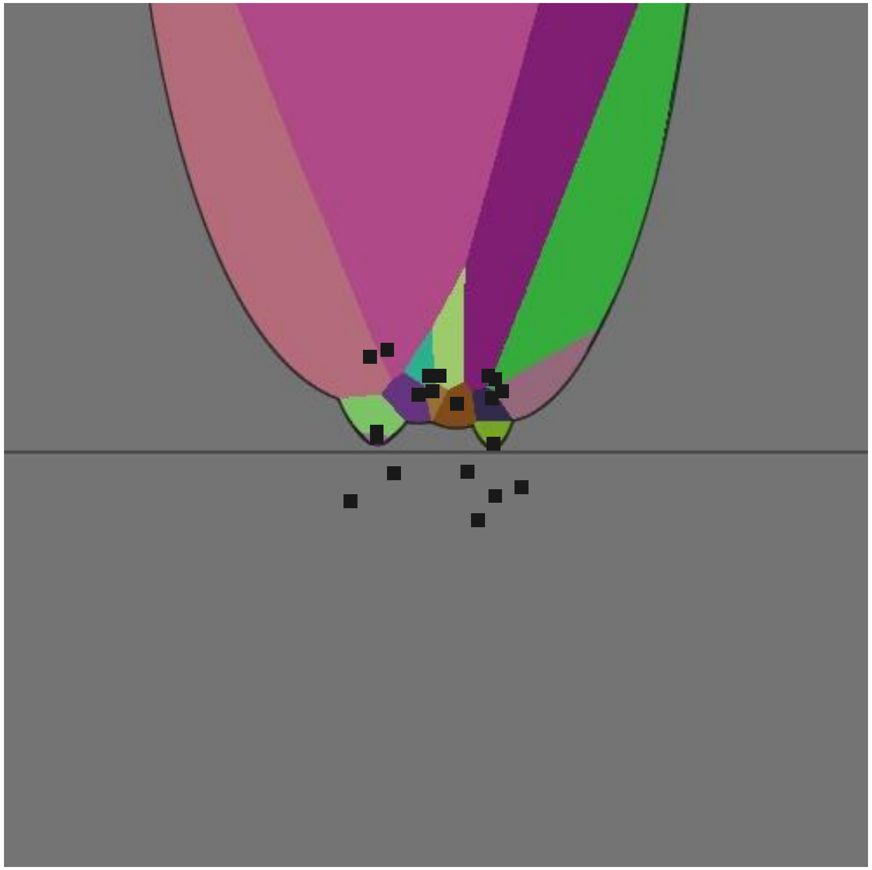
# Fortune's Algorithm



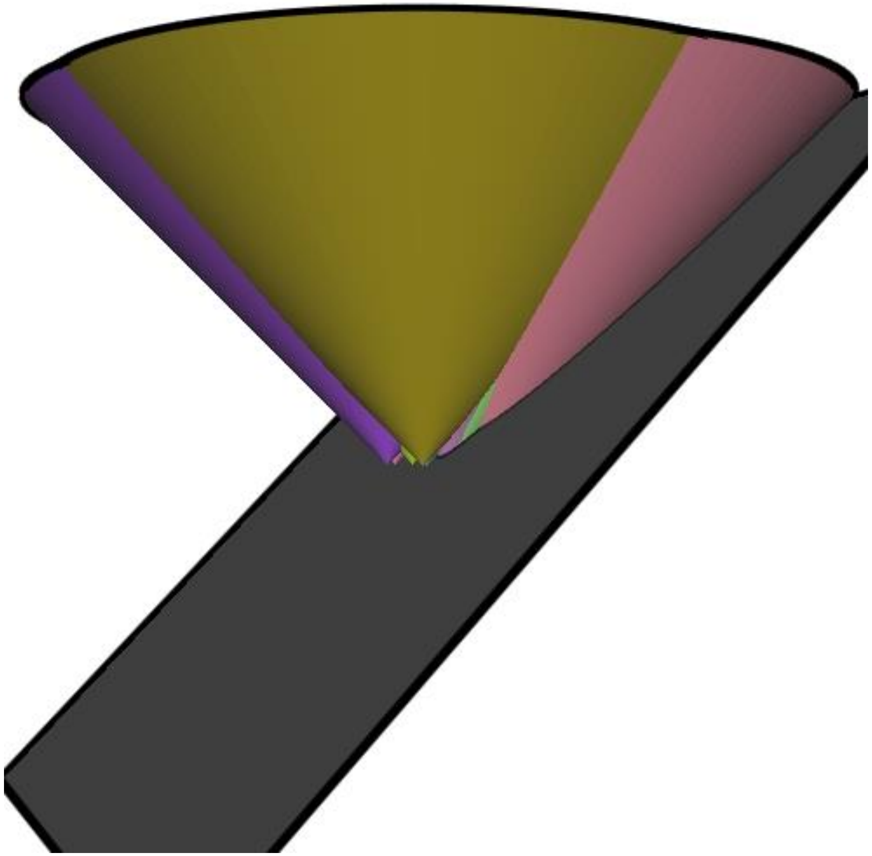


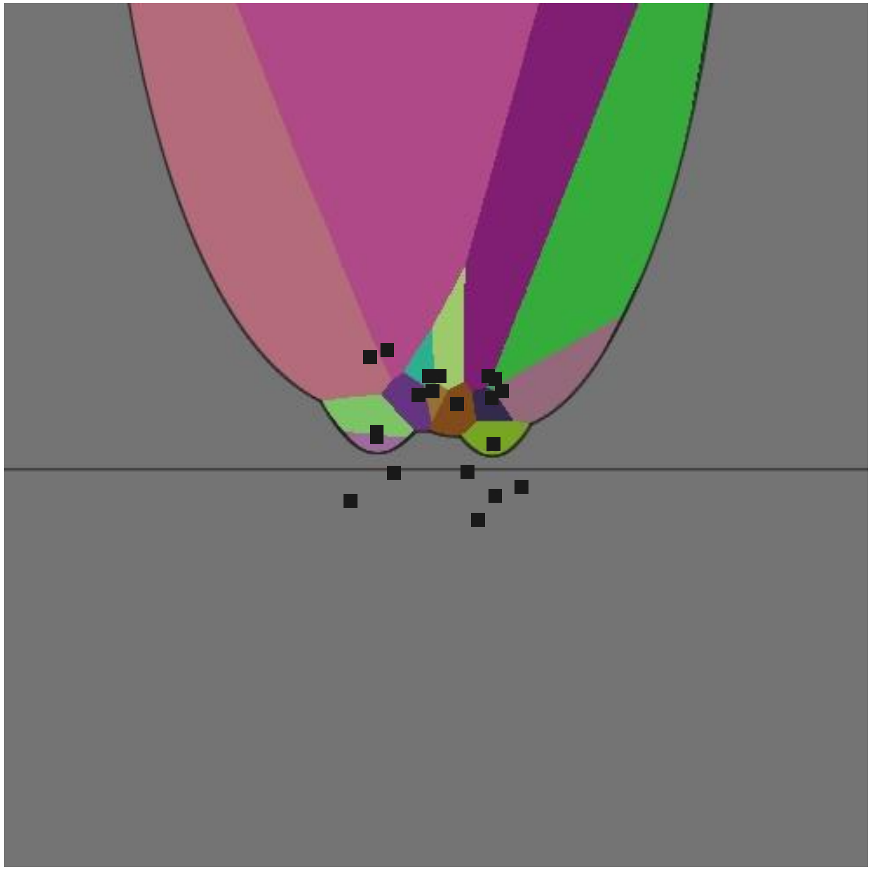
# Fortune's Algorithm



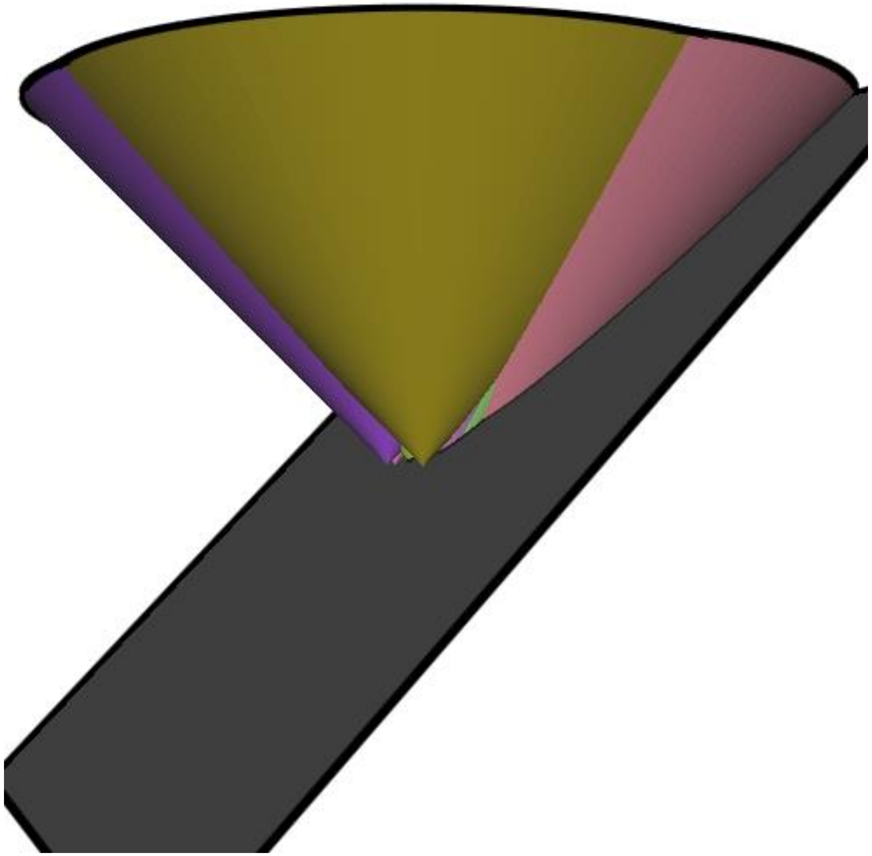


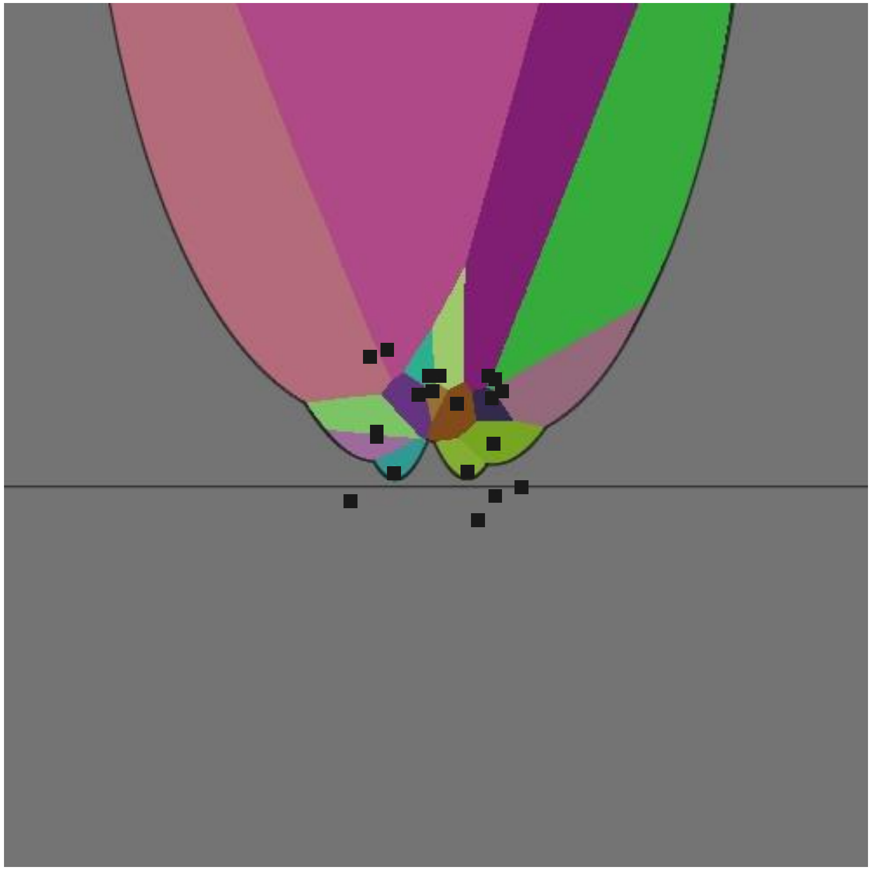
# Fortune's Algorithm



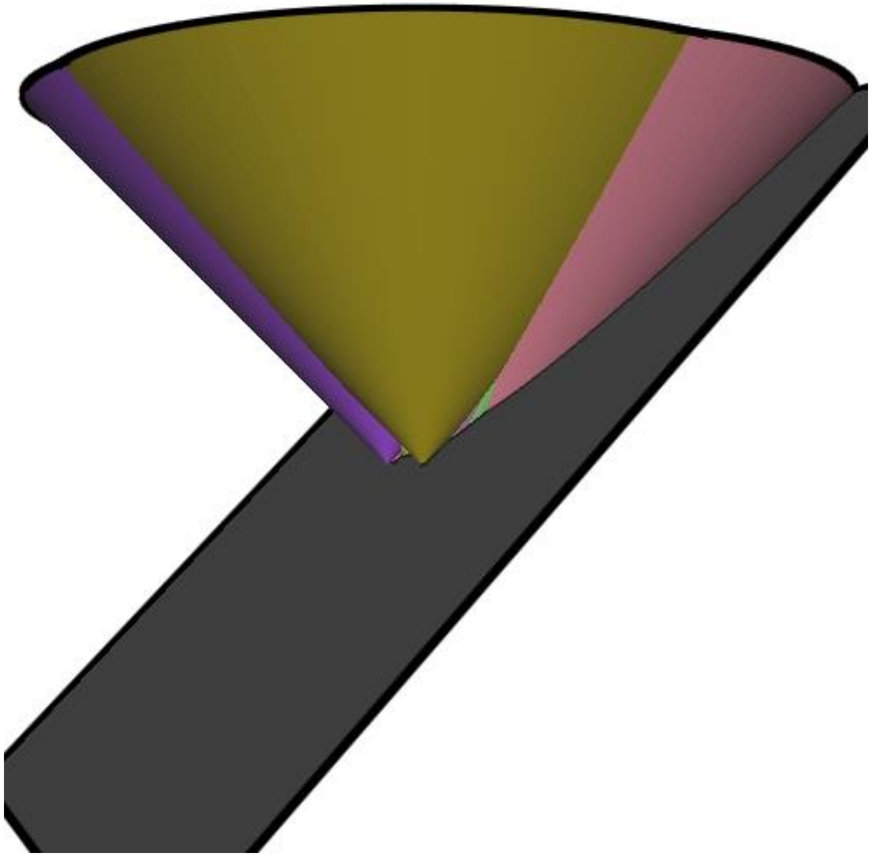


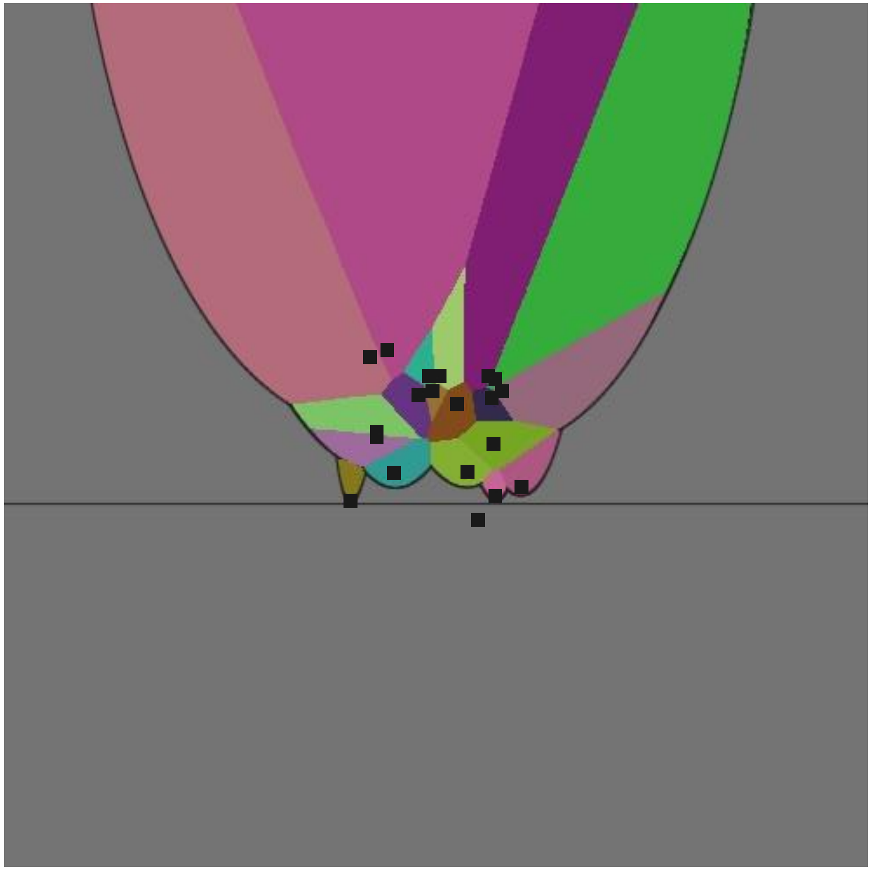
# Fortune's Algorithm



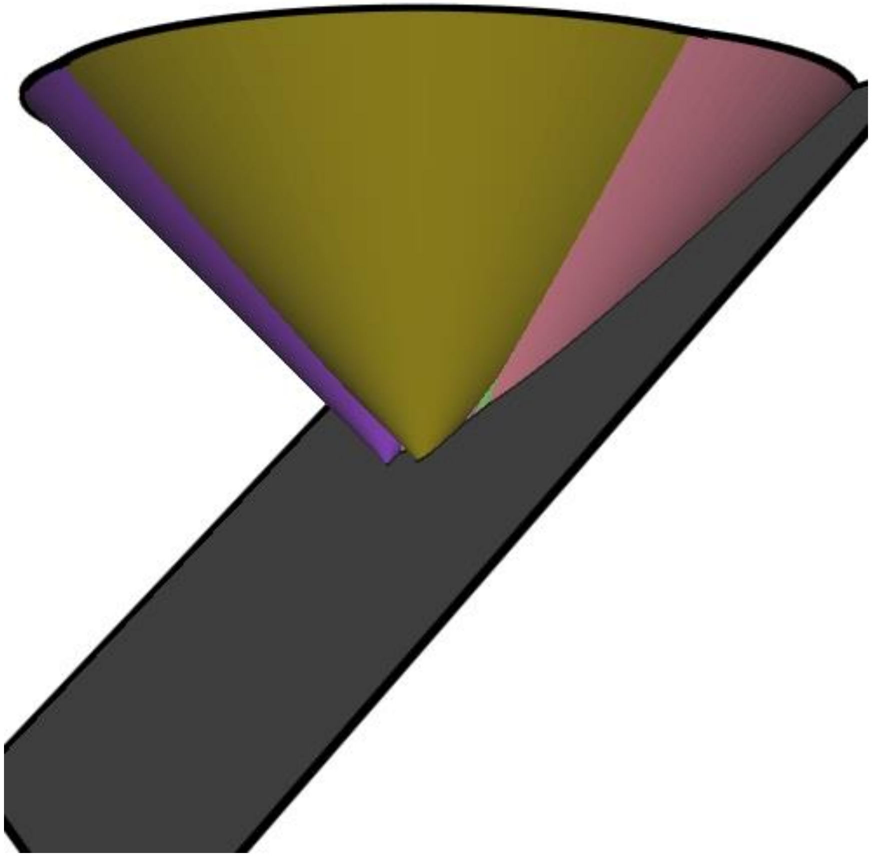


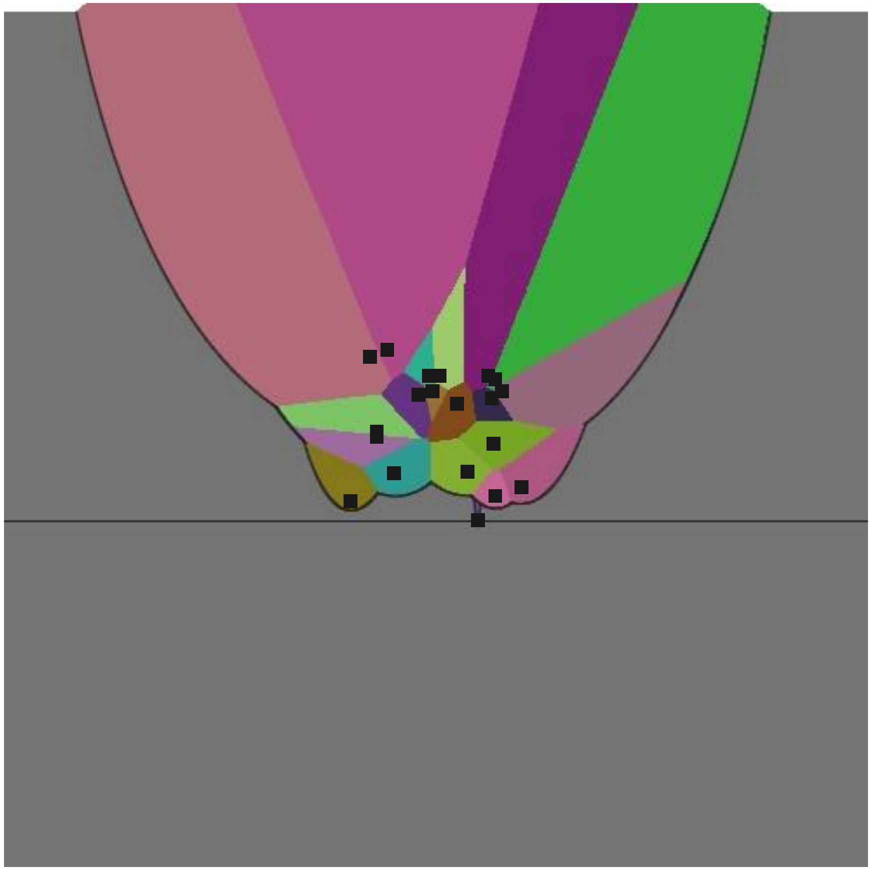
# Fortune’s Algorithm



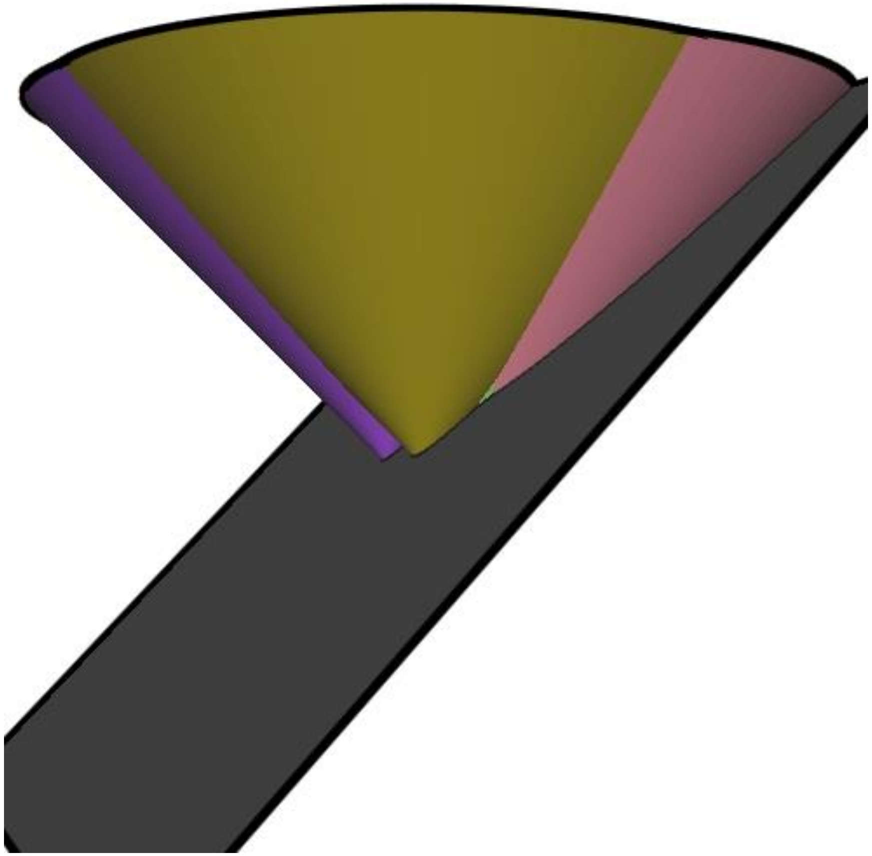


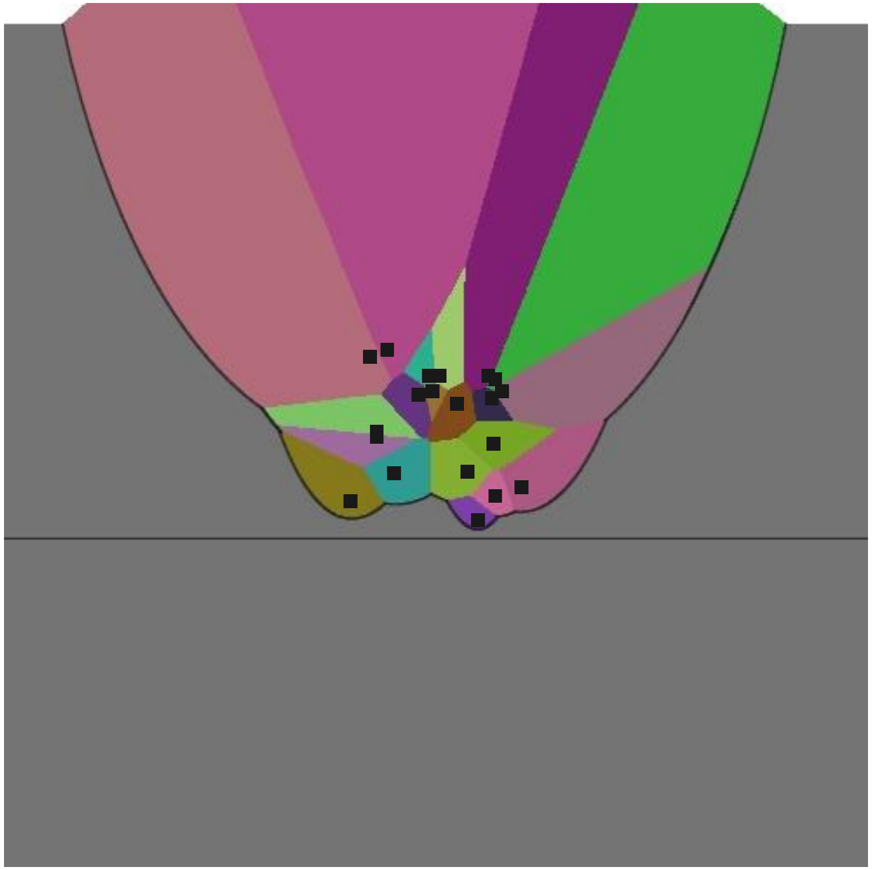
# Fortune's Algorithm



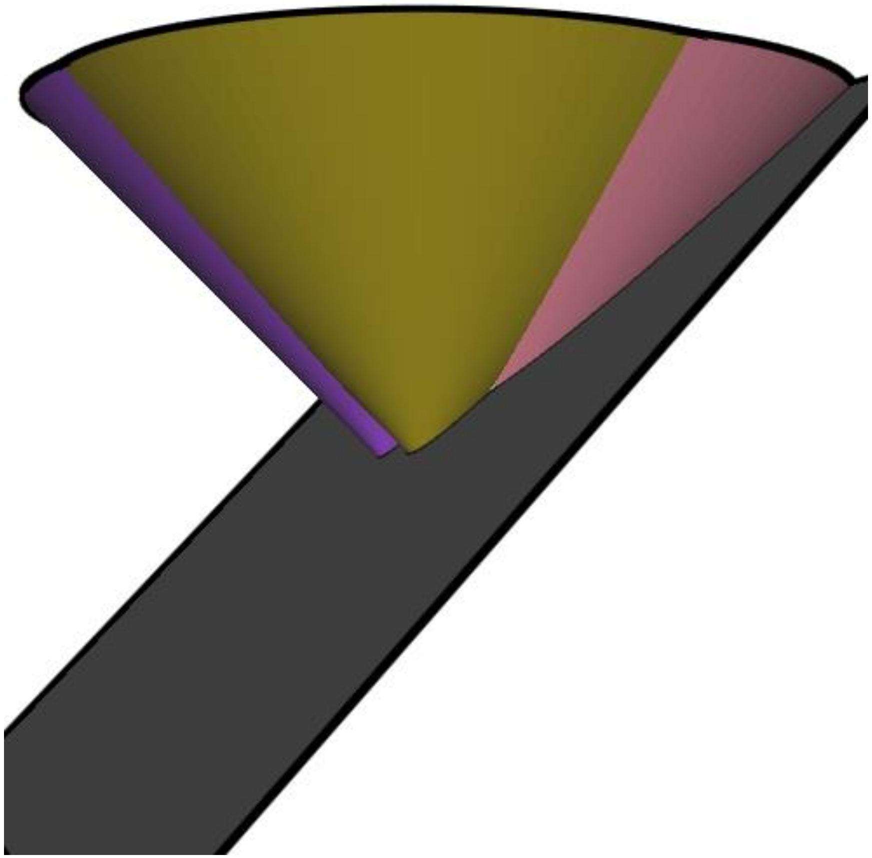


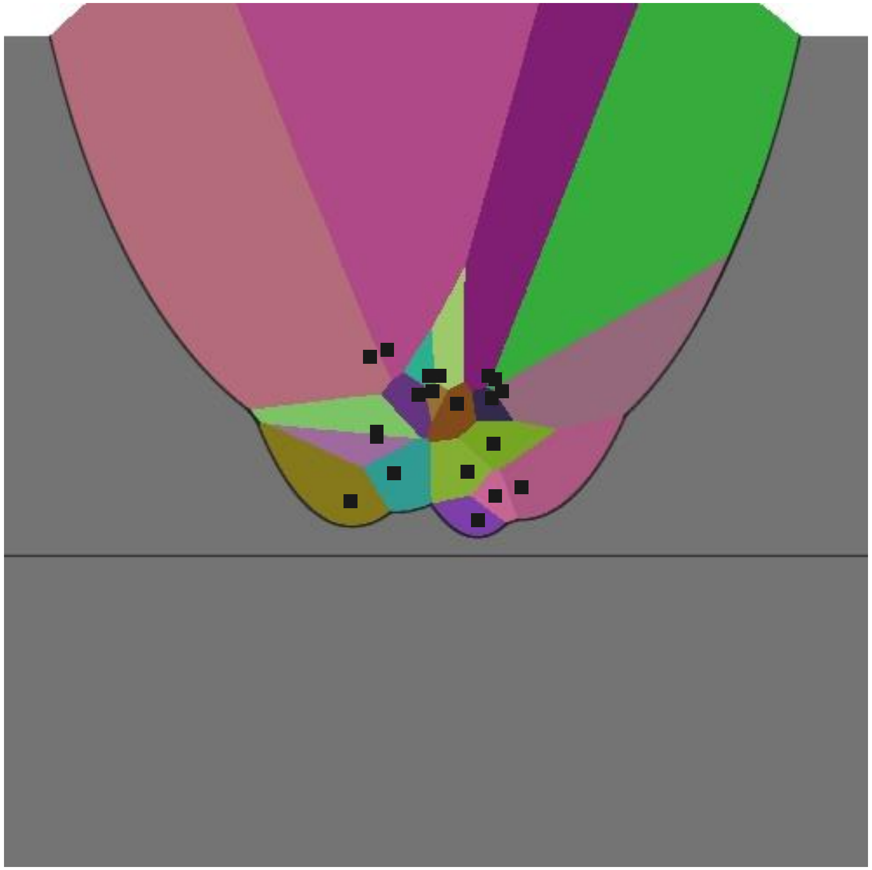
# Fortune's Algorithm



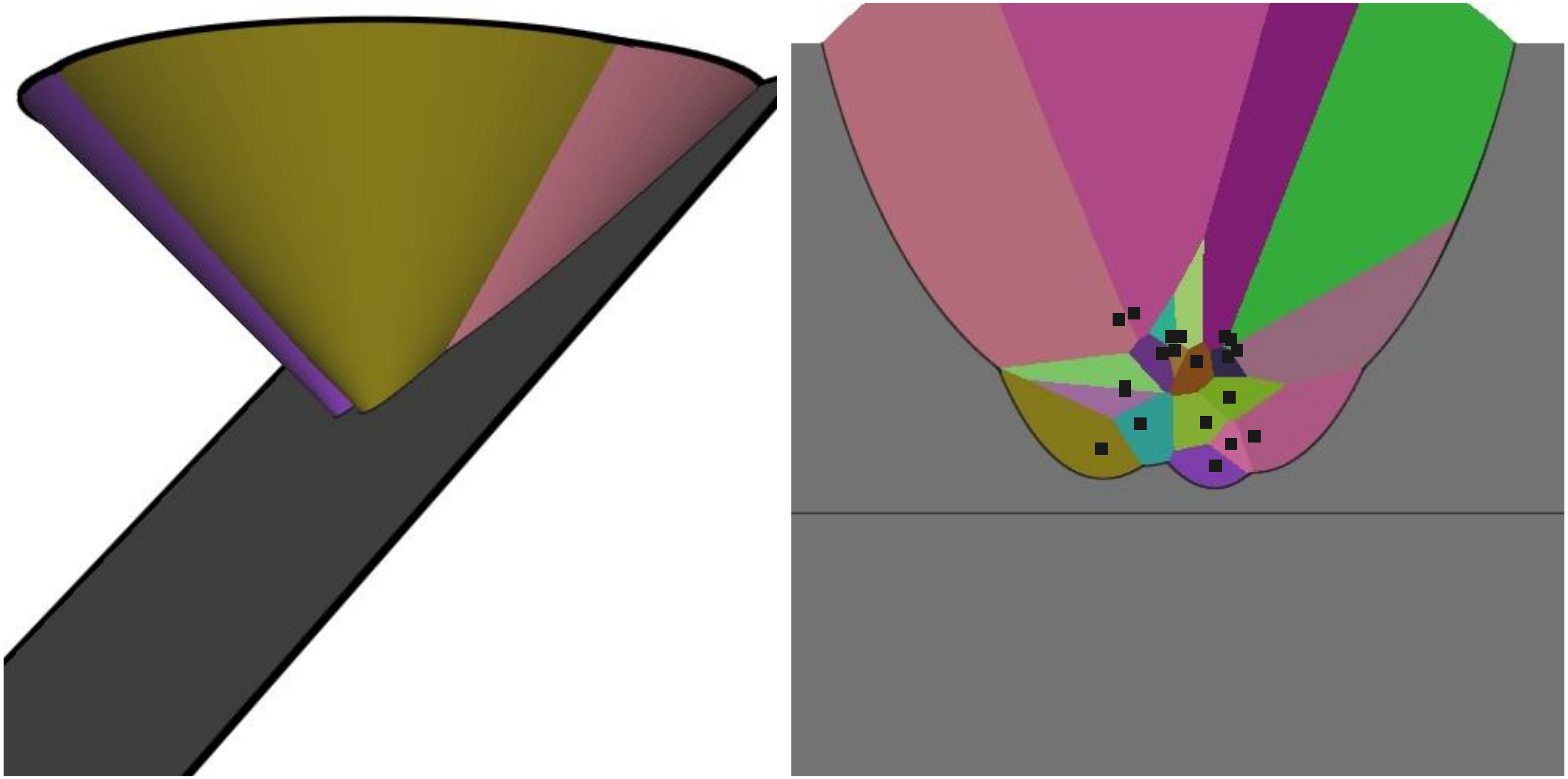


# Fortune's Algorithm

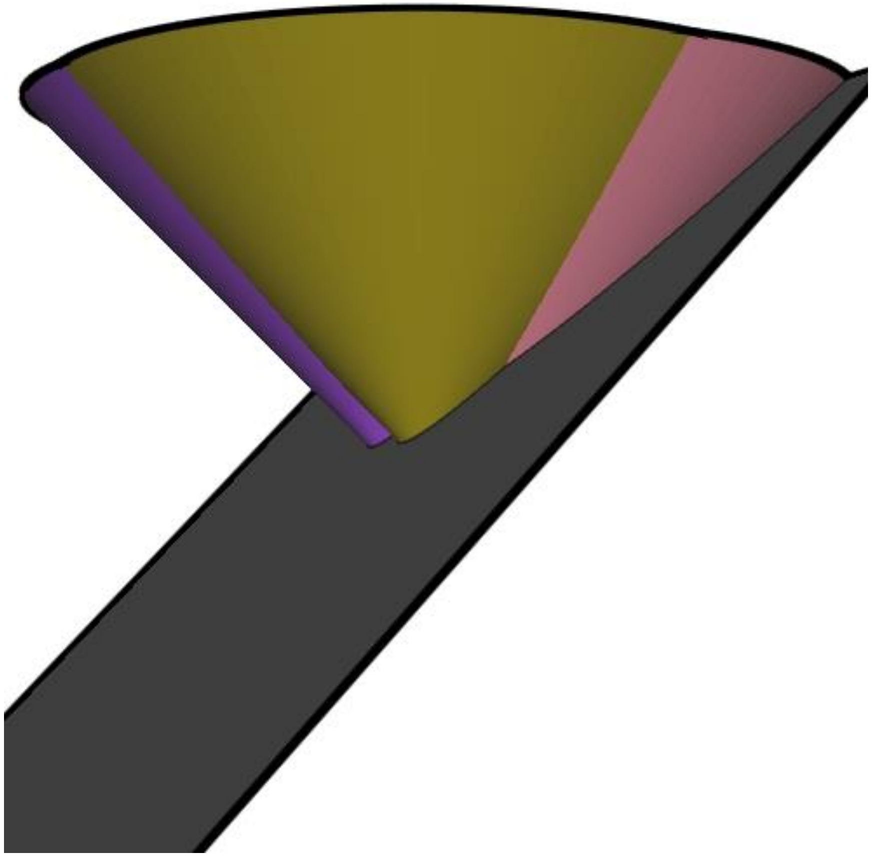


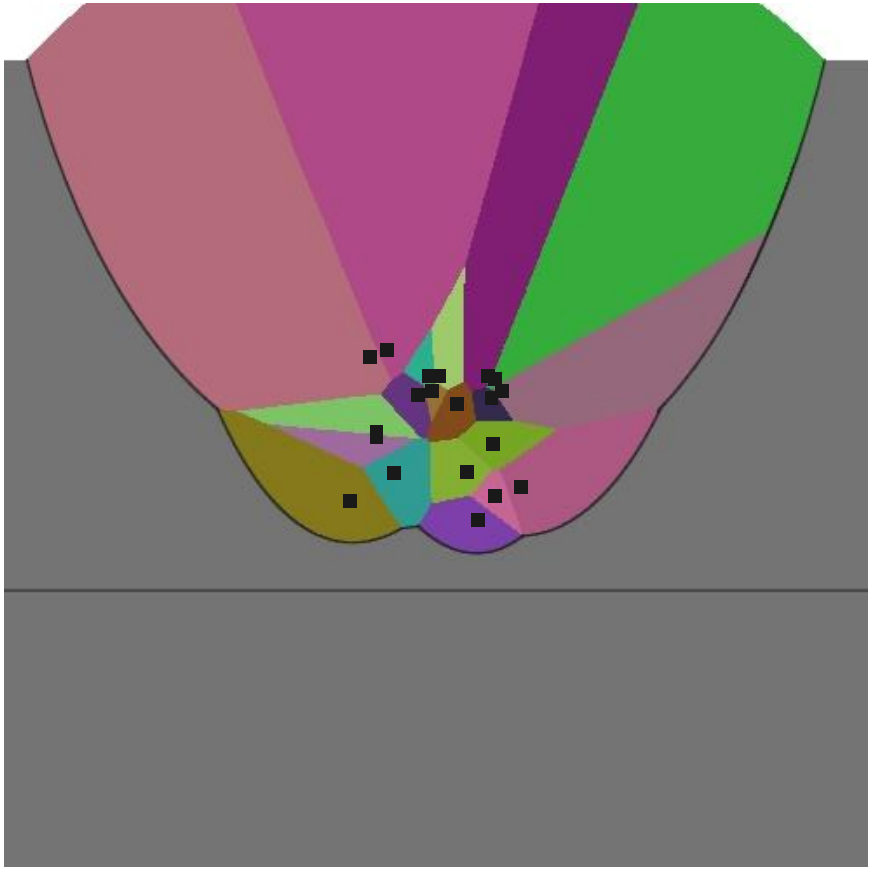


# Fortune’s Algorithm

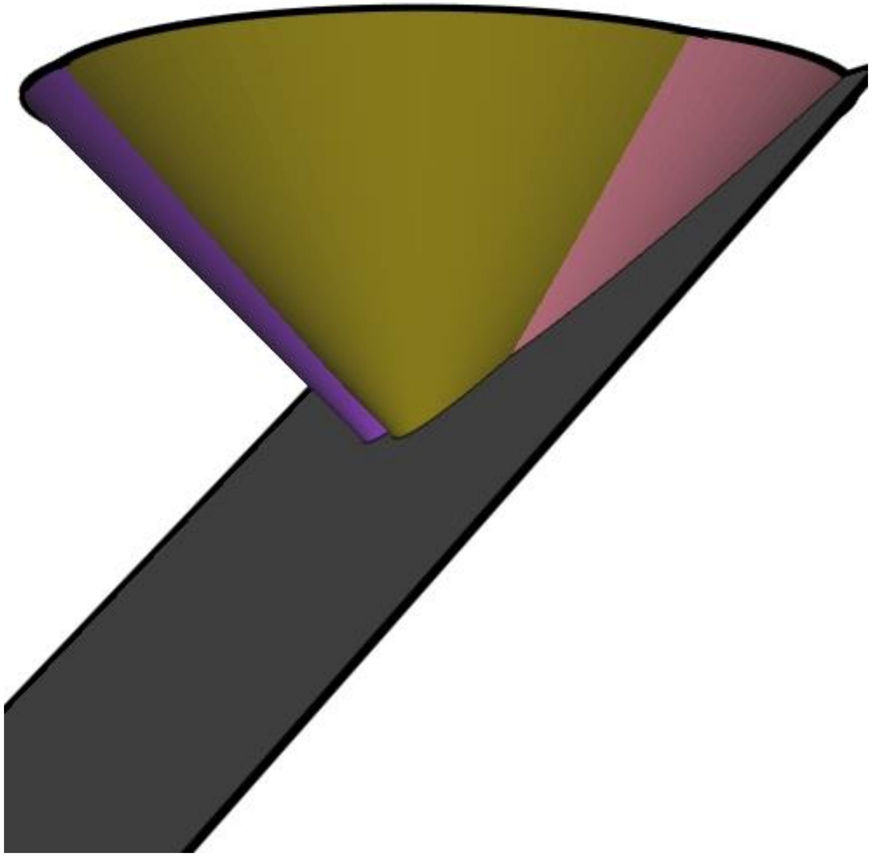


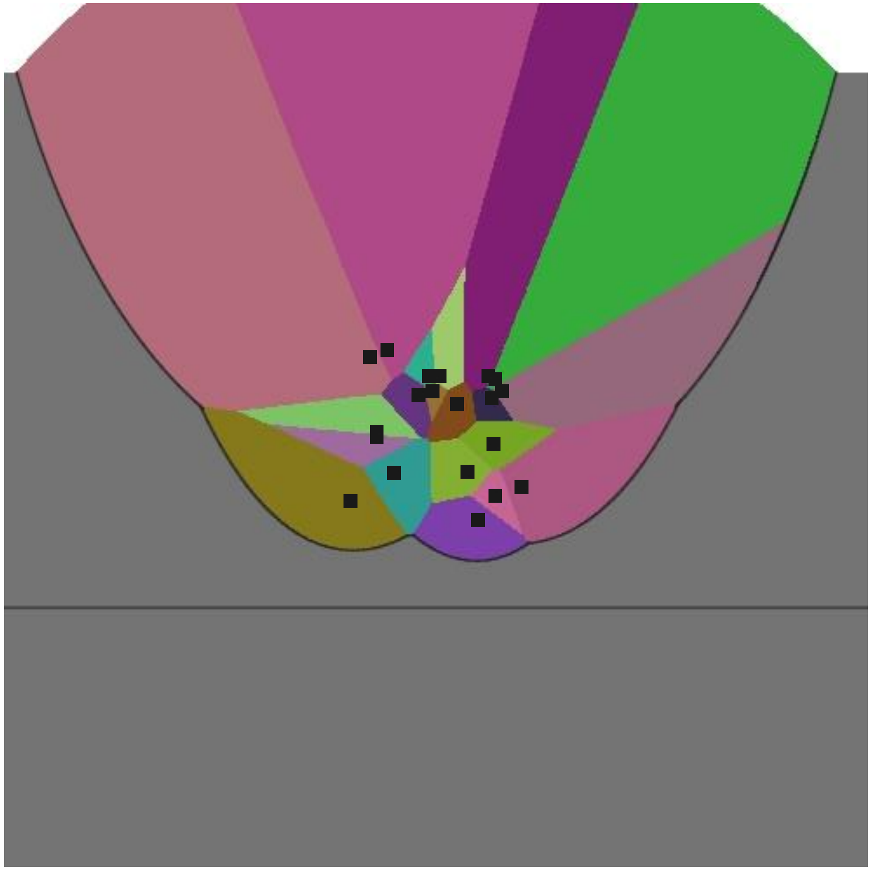
# Fortune's Algorithm



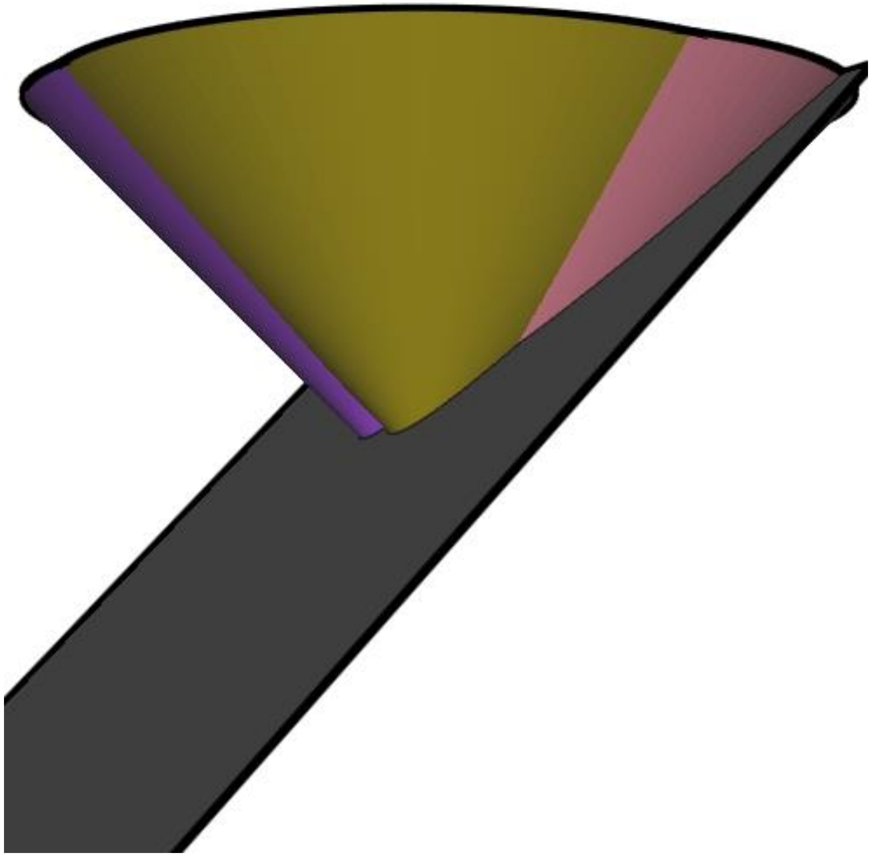


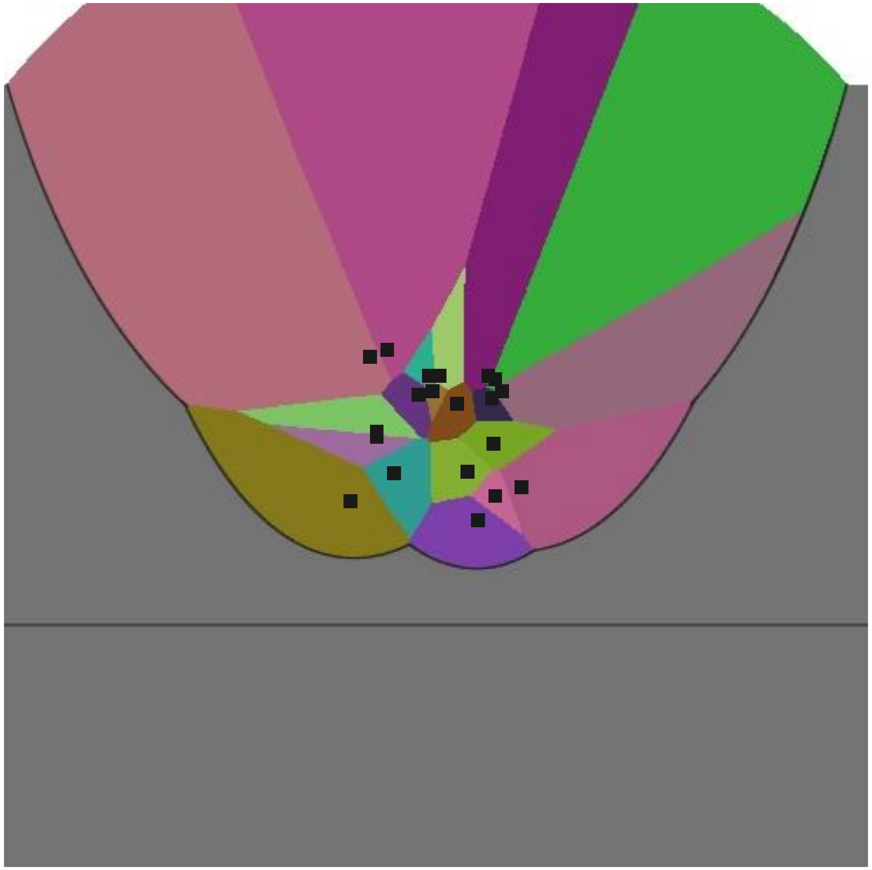
# Fortune’s Algorithm



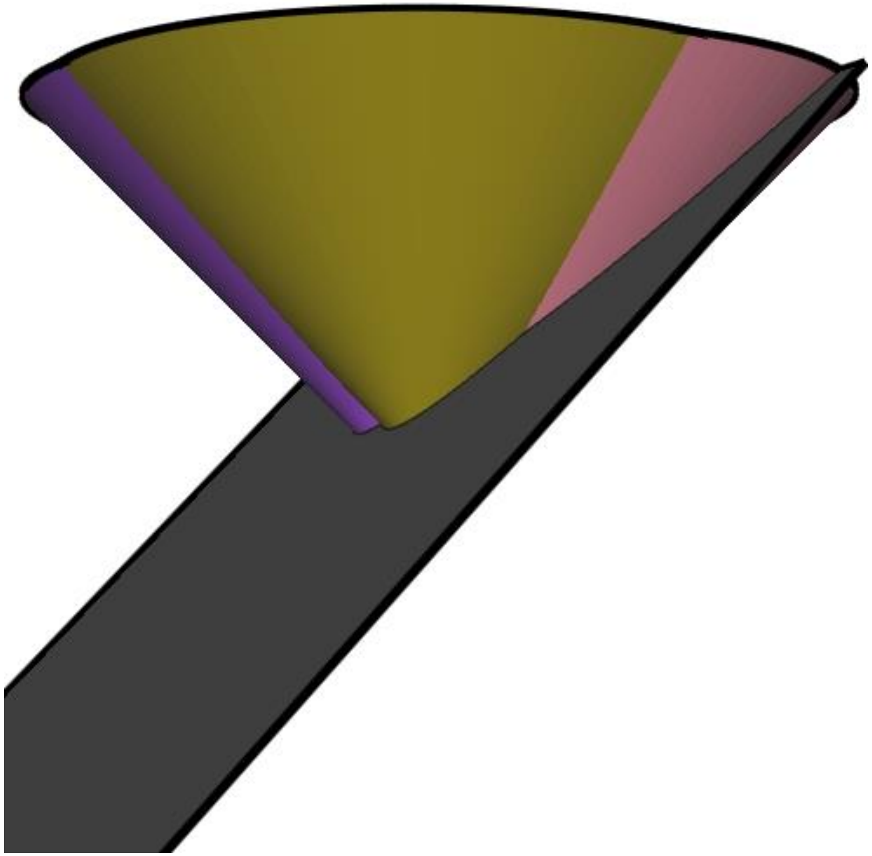


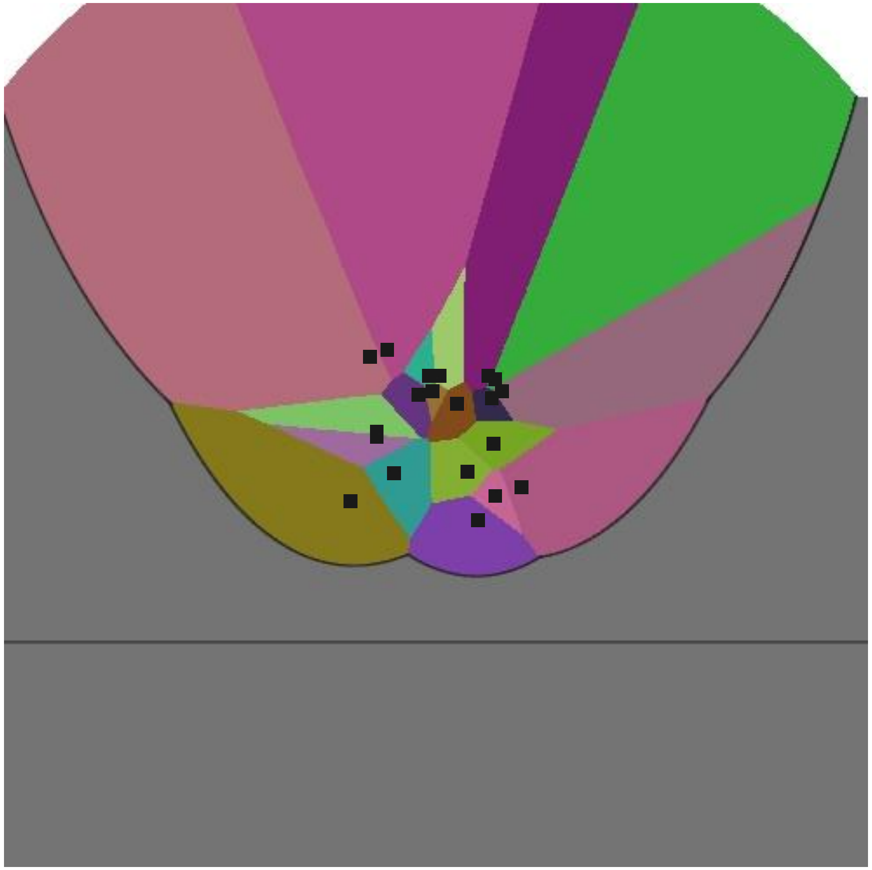
# Fortune's Algorithm



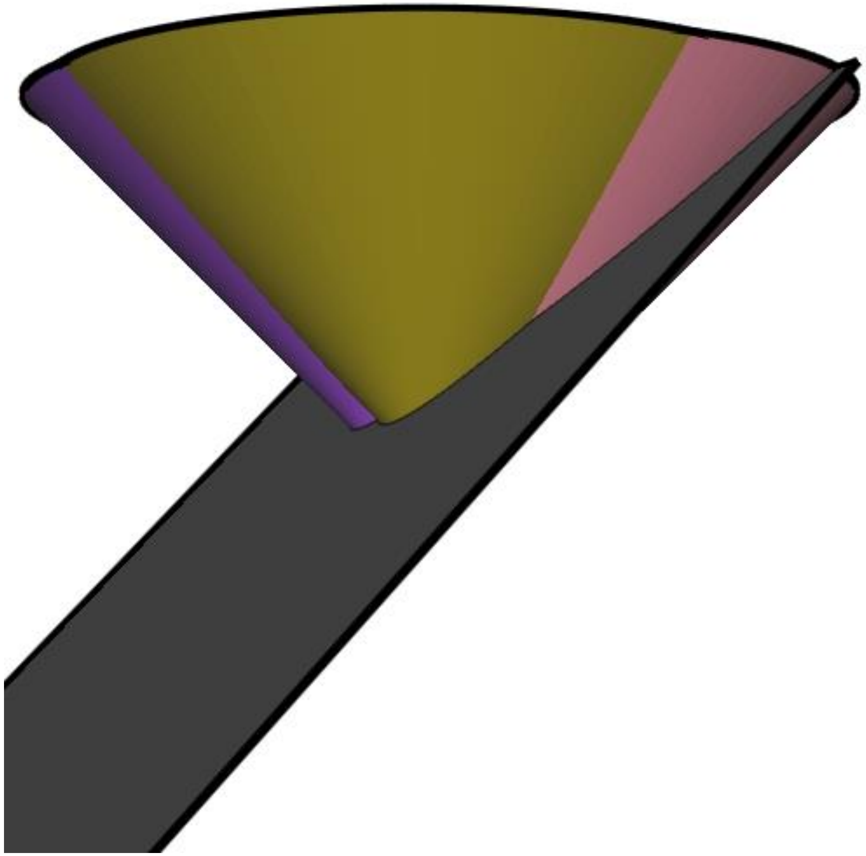


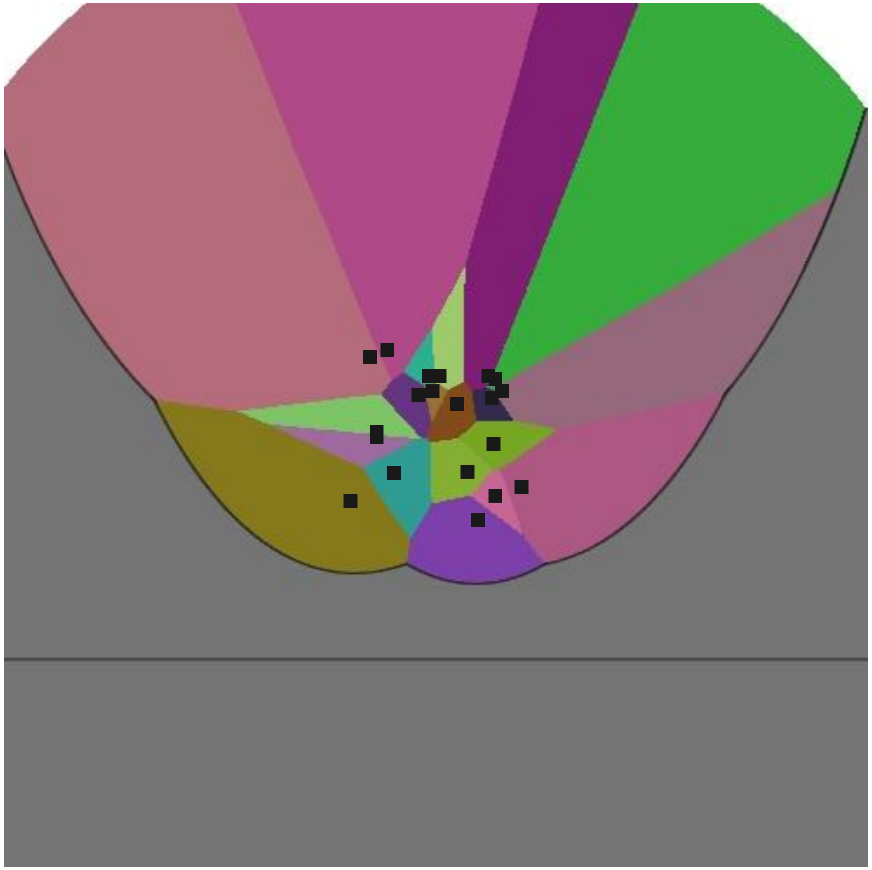
# Fortune's Algorithm



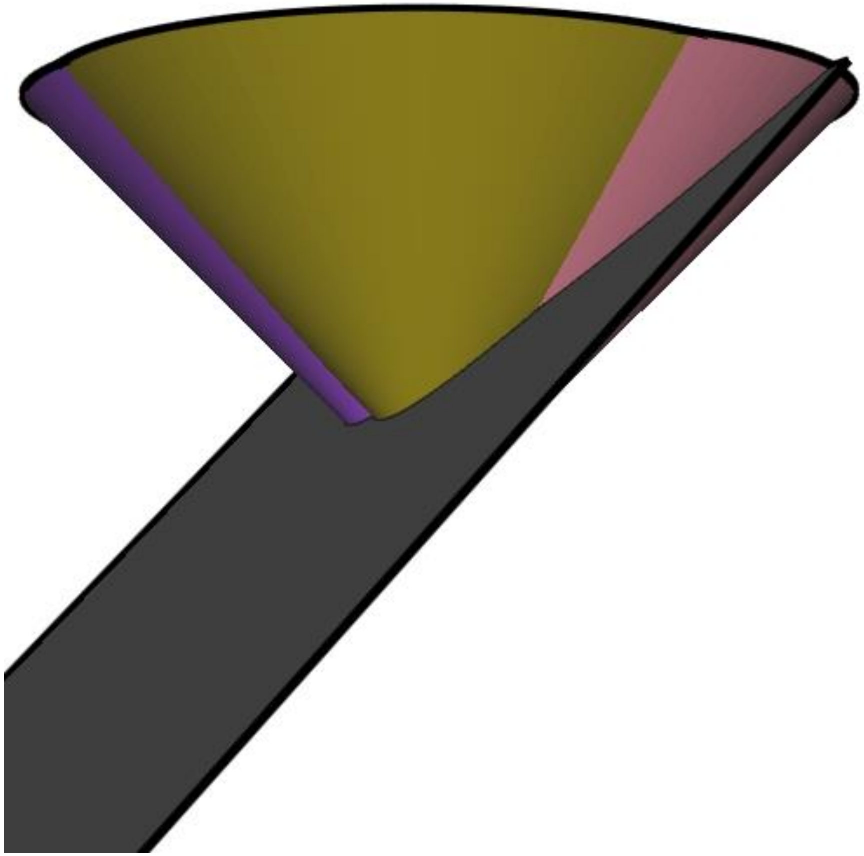


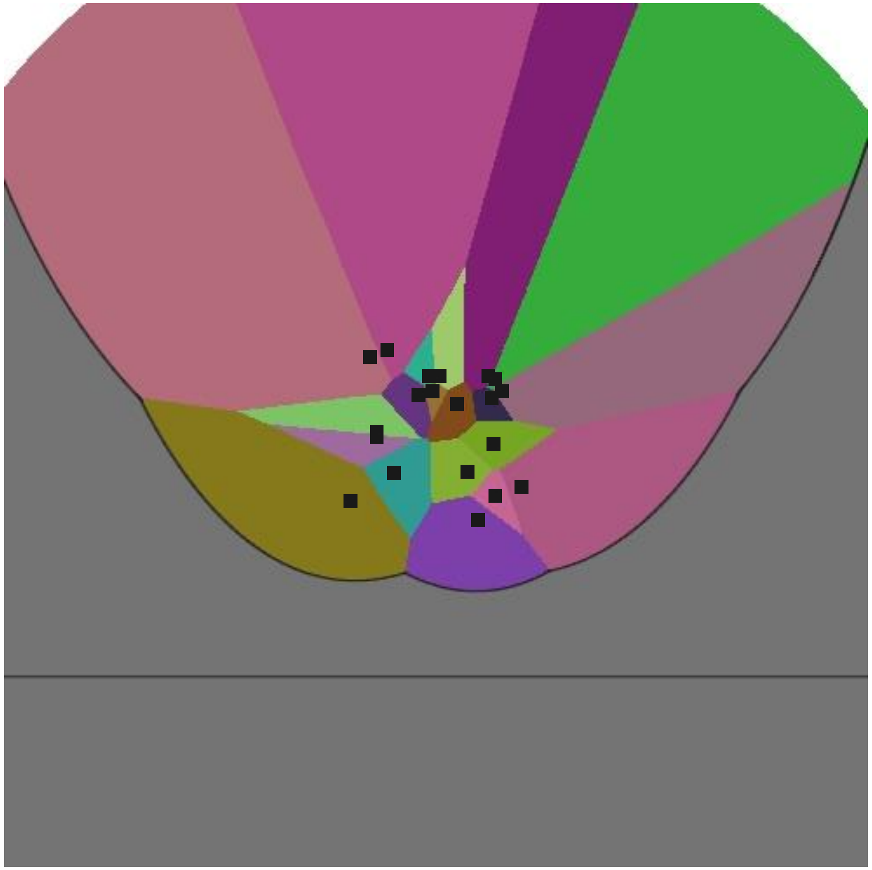
# Fortune's Algorithm





# Fortune's Algorithm





# Fortune's Algorithm

Sweep with a plane $\pi_y$, parallel to the $x$-axis, making a $45°$ angle with the $xy$-plane.

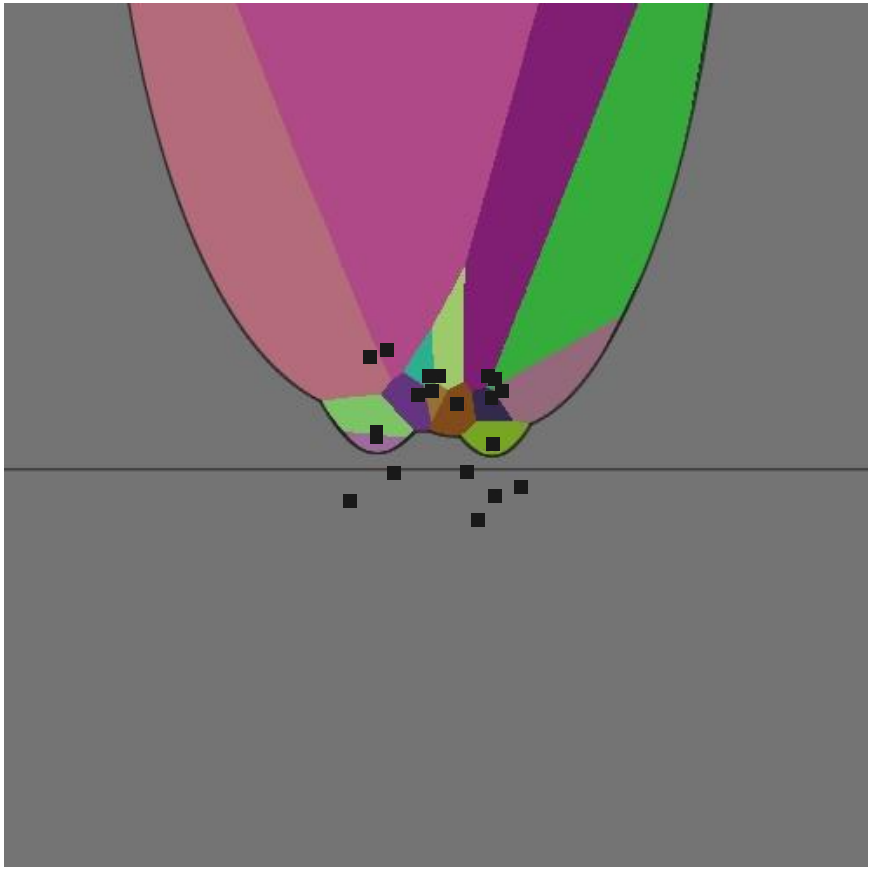
"Render" the cones and the plane with an orthographic camera looking up the $z$-axis.

At each point, we see:

- The part of $\pi_y$ that is in front of the line (since it is below the $xy$-plane and hence below the cones).
- The part of the cones that are behind the line and below $\pi_y$.

# Fortune's Algorithm

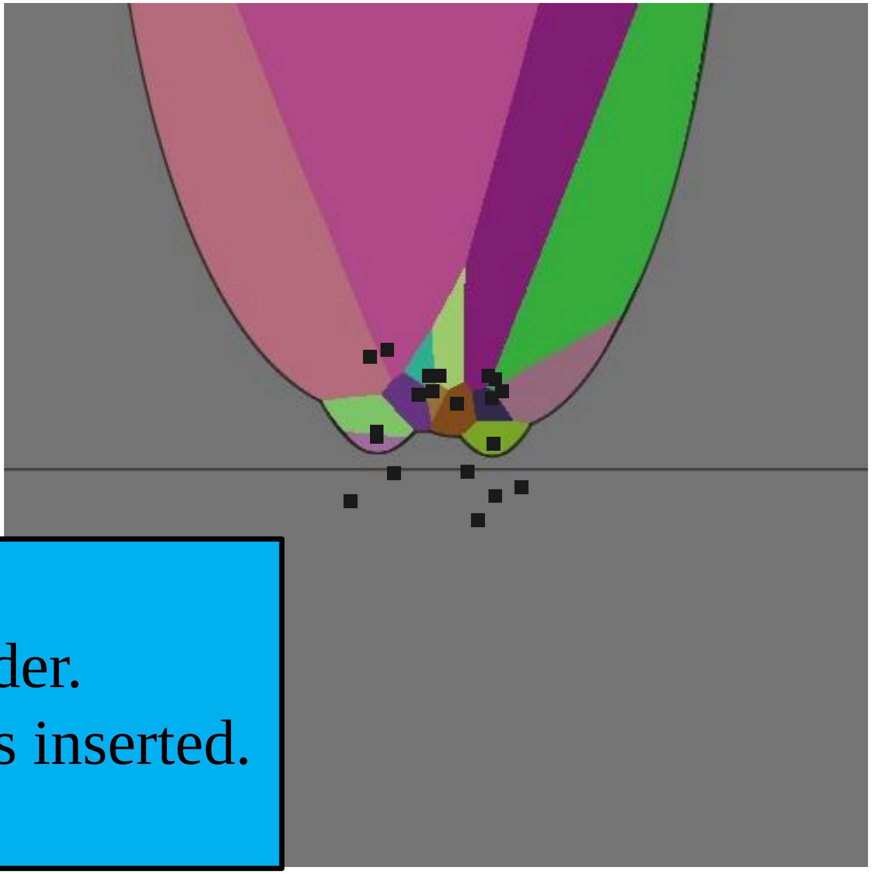
As $y$ advances, the algorithm maintains a set of parabolic fronts (the projection of the intersections of $\pi_y$ with the cones).



At any point, the Voronoi diagram is finalized behind the parabolic fronts.

# Fortune’s Algorithm

As $y$ advances, the algorithm maintains a set of parabolic fronts (the projection of the intersections of $\pi_y$ with the cones).



At any point, the Voronoi diagram is finalized behind the

<u>Implementation</u>:

- The fronts are maintained in order.
- As $y$ intersects a site, its front is inserted.
- Complexity $O(n \log n)$.

# Outline

- Preliminaries
- Voronoi Diagrams / Delaunay Triangulations
- Lloyd’s Algorithm

# Lloyd's Algorithm

Challenge:

Solve for the position of points $P = \{p_1, \dots, p_n\}$ inside a region (e.g. the unit square) minimizing:

$$E(P) = \int_{[0,1]^2} d^2(q, P)\, dq$$

where $d(q, P) = \min_i |p_i - q|$.

# Lloyd's Algorithm

Approach:

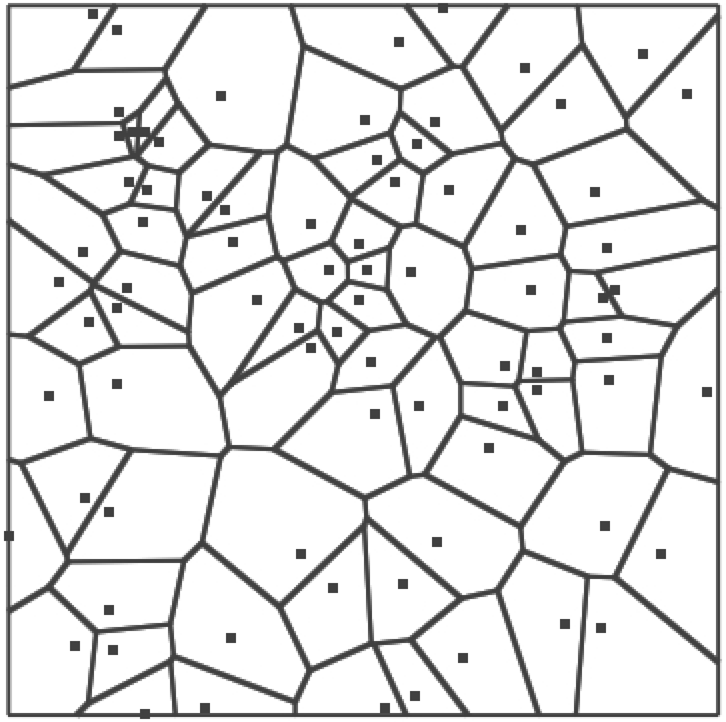
1. Initialize the points to random positions.
2. Compute the Voronoi Diagram of the points, clipped to the unit square.
3. Replace the points' positions with the centers of mass of their Voronoi cells.
4. Go to step 2.

# Lloyd’s Algorithm

1. Initialize the points to random positions.

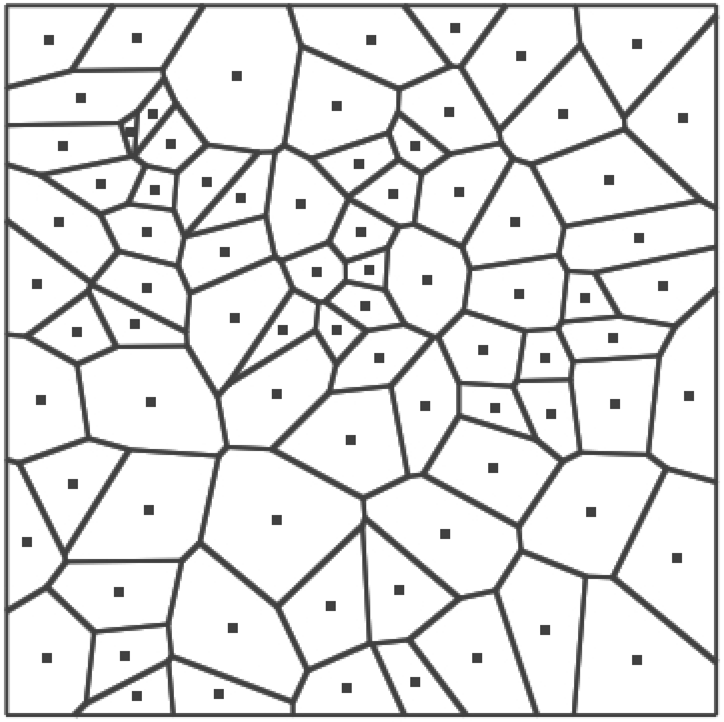
# Lloyd’s Algorithm

2. Compute the Voronoi Diagram of the points, clipped to the unit square.



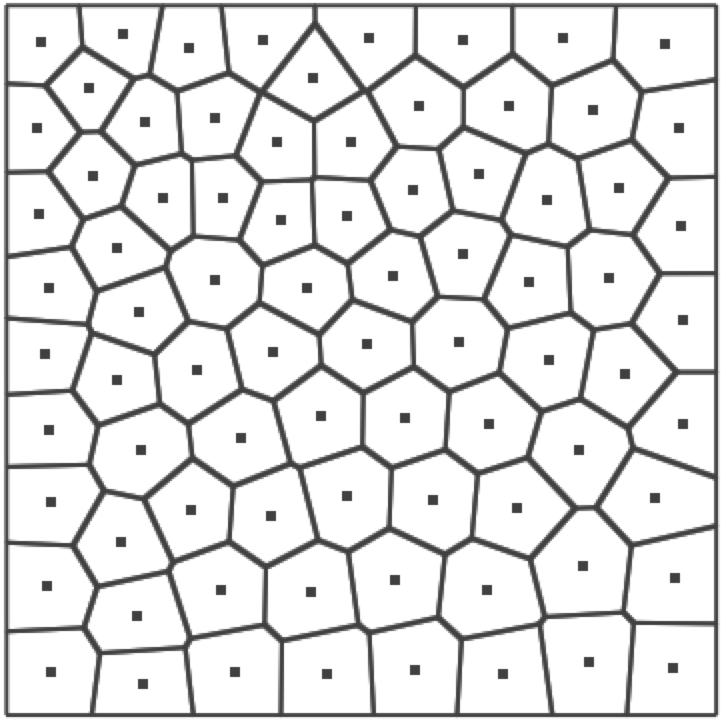
# Lloyd’s Algorithm

3. Replace the points’ positions with the centers of mass of their Voronoi cells.



# Lloyd’s Algorithm

4. Go to step 2.



# Lloyd’s Algorithm

2. Compute the Voronoi Diagram of the points, clipped to the unit square.

Holding the points in $P$ fixed:

$$\int_{[0,1]^2} d^2(q,P)\,dq = \sum_{F_i \in V(P)} \int_{F_i} \|p_i - q\|^2 dq$$

⇒ Using the Voronoi cells gives a partition of unit square that minimizes the energy, relative to the points’ positions.

# Lloyd's Algorithm

3. Replace the points' positions with the centers of mass of their Voronoi cells.

Holding the partition into Voronoi cells fixed and setting $C(F_i)$ be the center of mass of the $i$-th Voronoi face:

$$\underset{p\in[0,1]^2}{\arg\min}\int_{F_i}\|p-q\|^2 dq = C(F_i).$$

⇒ Using the centers of mass gives positions that minimize the energy, relative to the partition of the unit square.