Computational Geometry

Tutorial #3 — Farthest Pairs

Institut für Betriebssysteme
und Rechnerverbund
Algorithmik

Peter Kramer | *November 14th, 2024*

Farthest point pairs
What we know

Exercise 3 (Farthest Point Pairs). **(5+10 points)**

Given a set $\mathcal{P}$ of n points in the Euclidean plane, two points $p, q \in \mathcal{P}$ are a *farthest pair* in $\mathcal{P}$ if

$$\forall\, u, v \in \mathcal{P}: \quad |p - q| \;\geq\; |u - v|.$$

The Euclidean distance between p and q is then also called the *diameter* of $\mathcal{P}$.

a) Prove that all farthest pairs in $\mathcal{P}$ are vertices of the convex hull $\mathrm{conv}(\mathcal{P})$.

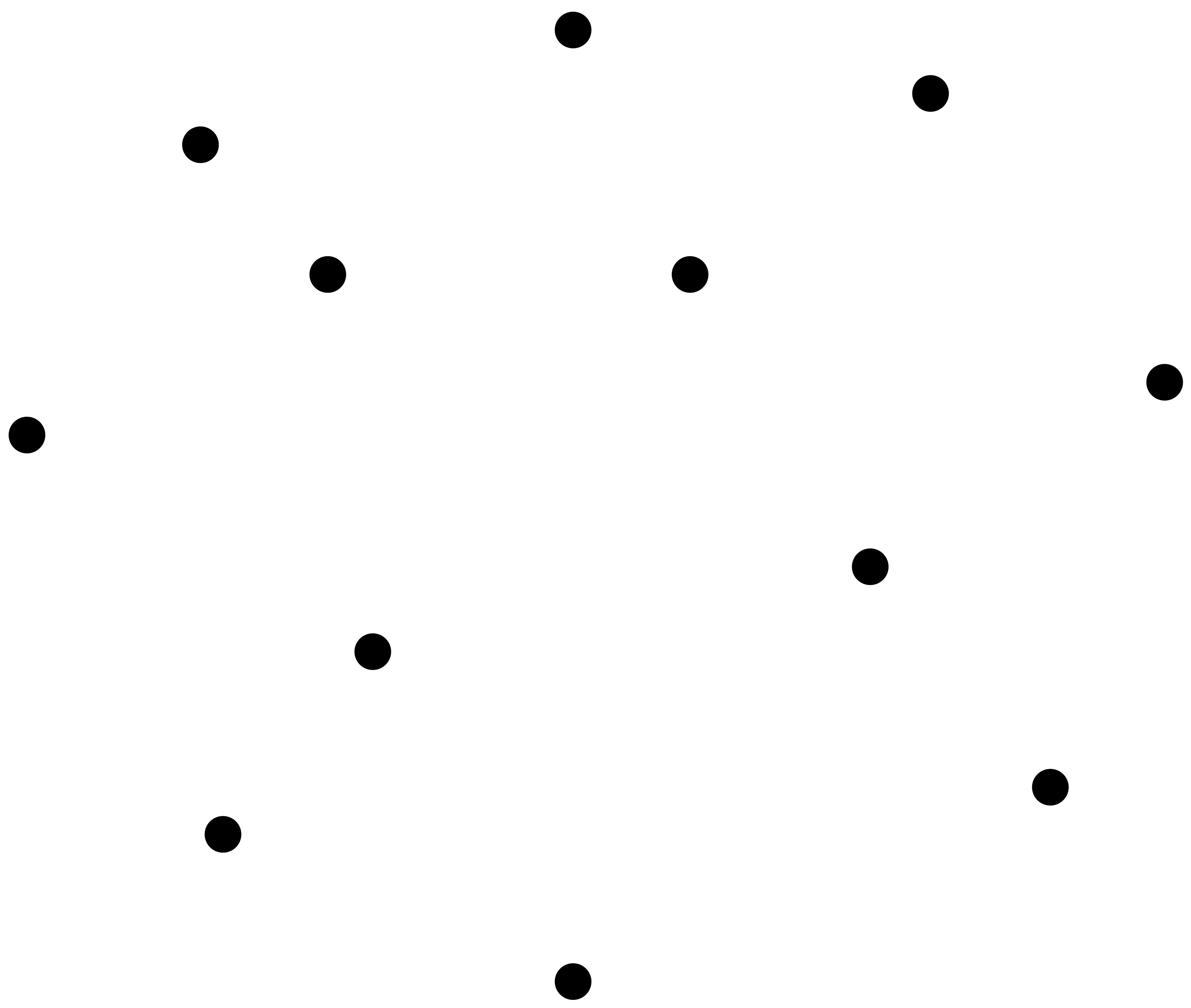
b) Design an $\mathcal{O}(n)$ algorithm that approximates the diameter of $\mathcal{P}$ up to a constant factor. Argue its correctness, approximation factor, and runtime!

- We will assume that **a)** is true — Discussion next week :)
- An exact $\mathcal{O}(n^2)$ time algorithm is trivial — can we do better?

Farthest point pairs
Convex hull

Let $\mathscr{P}$ be set of n points in the Euclidean plane $\mathbb{R}^2$, in general position*.

Lemma E3.1 All farthest pairs of $\mathscr{P}$ consist of two vertices of the convex hull $\mathrm{conv}(\mathscr{P})$.

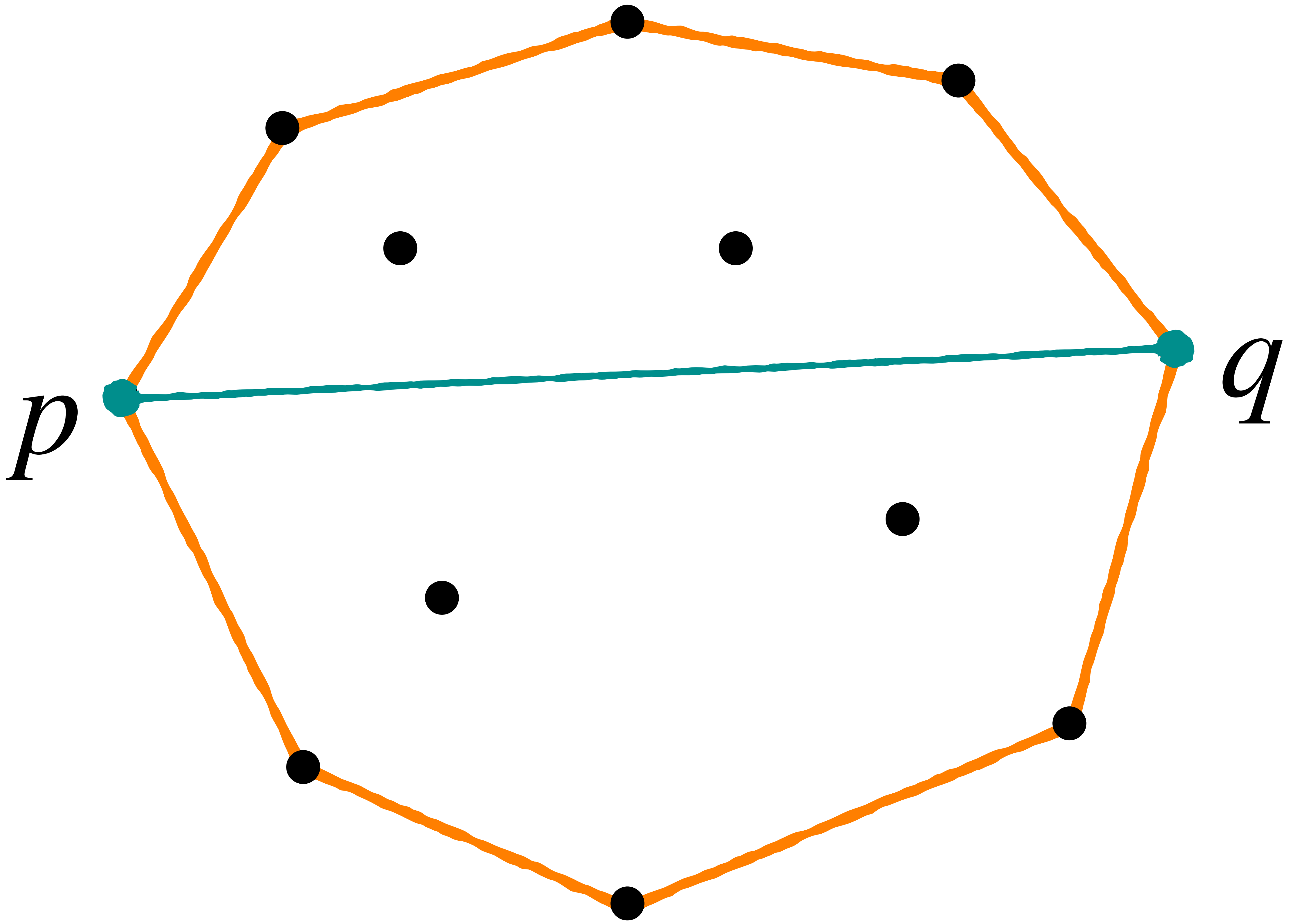


** no three points in $\mathscr{P}$ are collinear.*

Farthest point pairs
Convex hull

Let $\mathscr{P}$ be set of n points in the Euclidean plane $\mathbb{R}^2$, in general position*.

Lemma E3.1 All farthest pairs of $\mathscr{P}$ consist of two vertices of the convex hull $\mathrm{conv}(\mathscr{P})$.



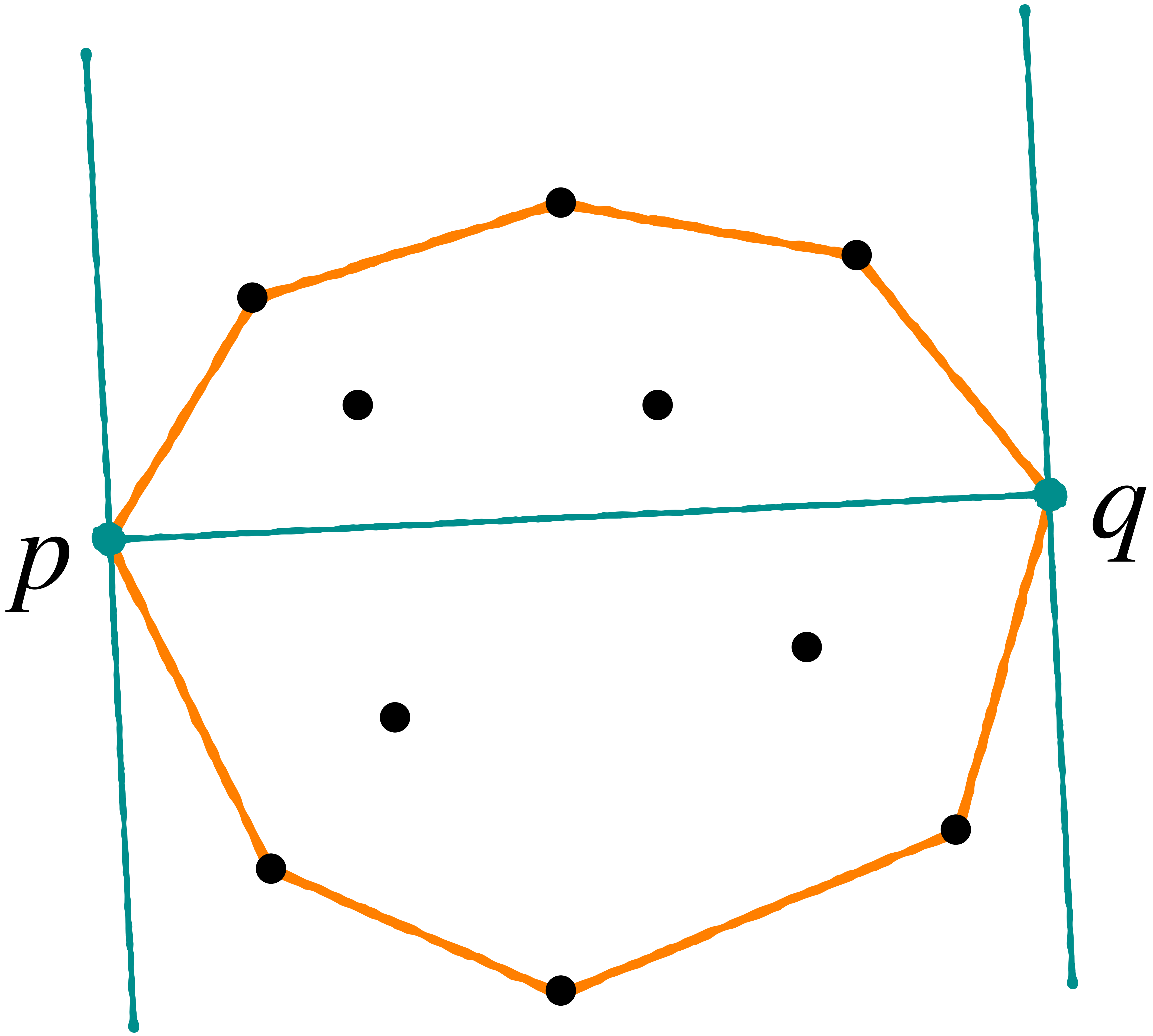
** no three points in $\mathscr{P}$ are collinear.*

Farthest point pairs
Antipodal pairs

Let $\mathscr{P}$ be set of n points in the Euclidean plane $\mathbb{R}^2$, in general position*.

Lemma E3.1 All farthest pairs of $\mathscr{P}$ consist of two vertices of the convex hull $\text{conv}(\mathscr{P})$.

Definition. Two points $p, q \in \mathscr{P}$ are **antipodal** if there exist parallel **supporting lines** through them which touch, but do not cut the convex hull.



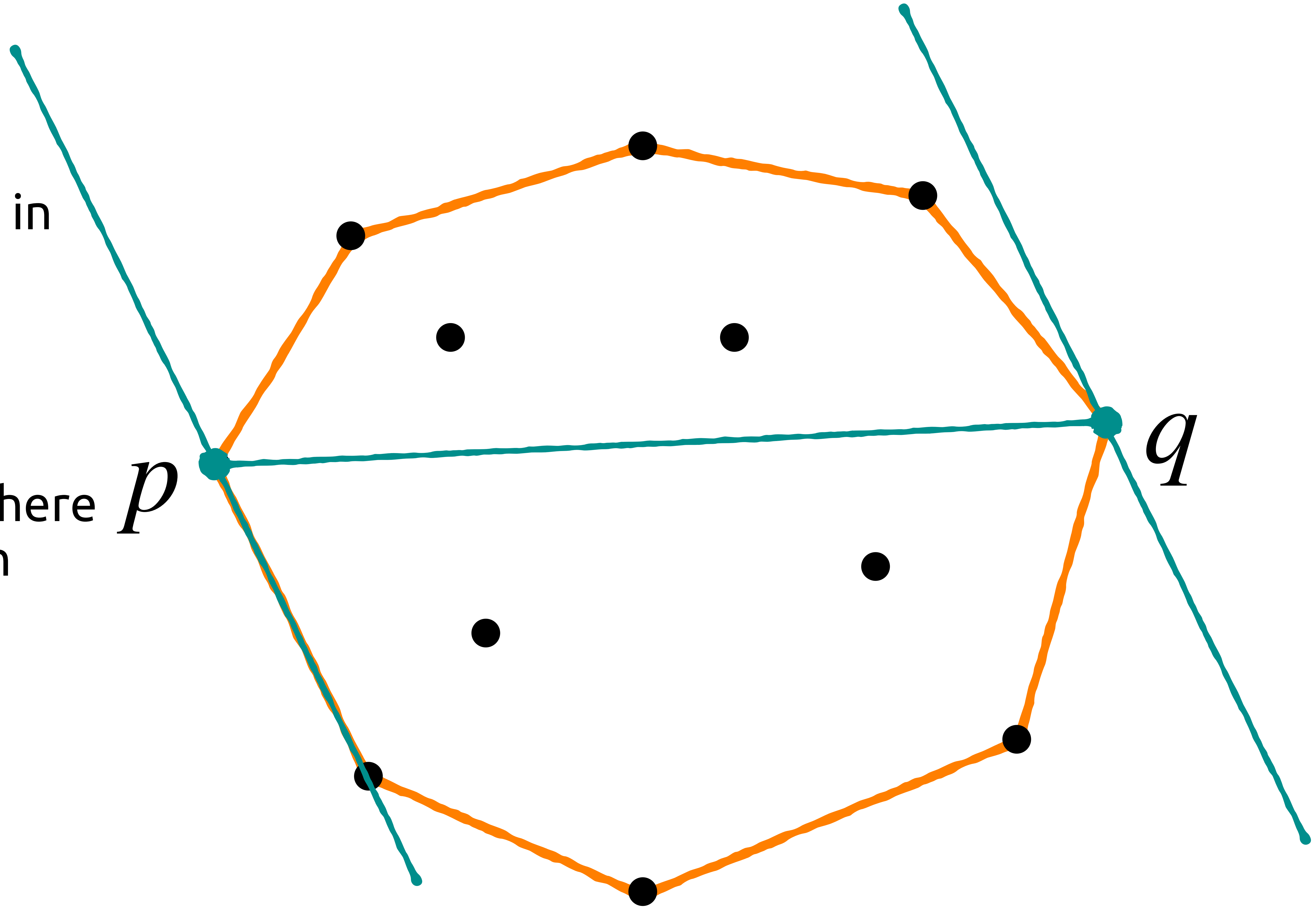
* *no three points in $\mathscr{P}$ are collinear.*

Farthest point pairs
Antipodal pairs

Let $\mathscr{P}$ be set of n points in the Euclidean plane $\mathbb{R}^2$, in general position*.

Lemma E3.1 All farthest pairs of $\mathscr{P}$ consist of two vertices of the convex hull $\mathrm{conv}(\mathscr{P})$.

Definition. Two points $p, q \in \mathscr{P}$ are **antipodal** if there exist parallel **supporting lines** through them which touch, but do not cut the convex hull.



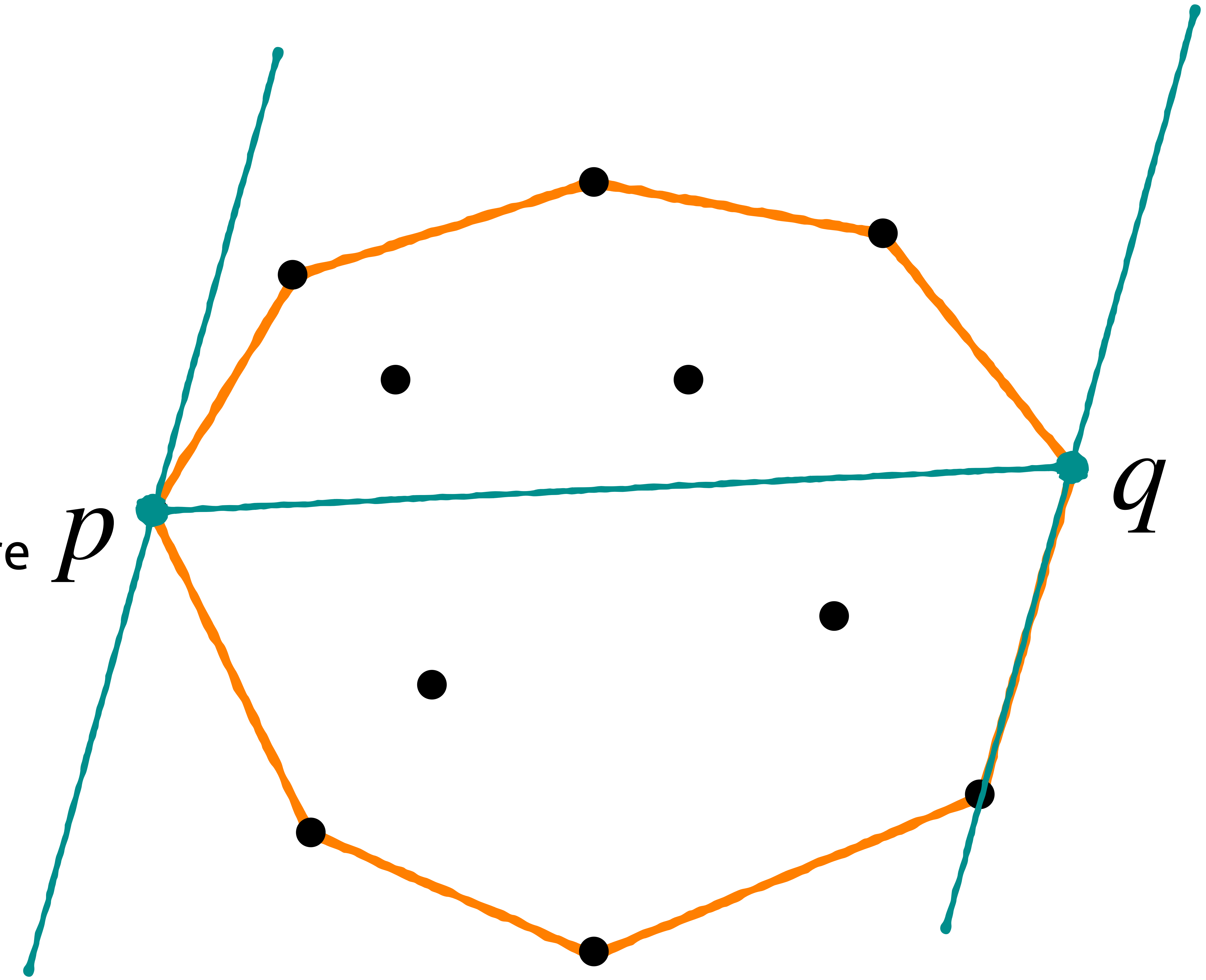
** no three points in $\mathscr{P}$ are collinear.*

Farthest point pairs
Antipodal pairs

Let $\mathscr{P}$ be set of n points in the Euclidean plane $\mathbb{R}^2$, in general position*.

Lemma E3.1 All farthest pairs of $\mathscr{P}$ consist of two vertices of the convex hull $\operatorname{conv}(\mathscr{P})$.

Definition. Two points $p, q \in \mathscr{P}$ are **antipodal** if there exist parallel **supporting lines** through them which touch, but do not cut the convex hull.



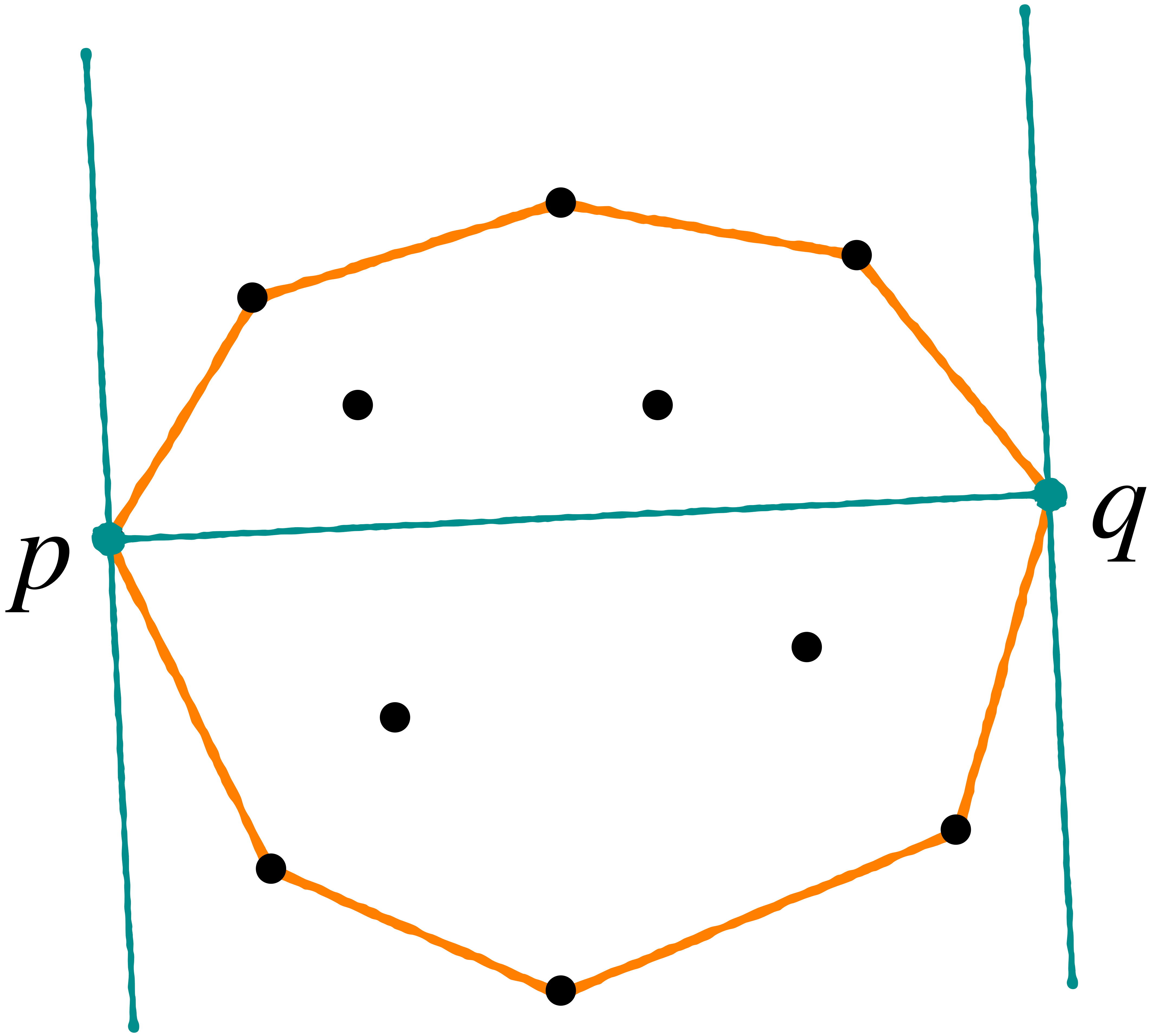
** no three points in $\mathscr{P}$ are collinear.*

Farthest point pairs
Antipodal pairs

Let $\mathscr{P}$ be set of n points in the Euclidean plane $\mathbb{R}^2$, in general position*.

Lemma E3.1 All farthest pairs of $\mathscr{P}$ consist of two vertices of the convex hull $\mathrm{conv}(\mathscr{P})$.

Definition. Two points $p, q \in \mathscr{P}$ are **antipodal** if there exist parallel **supporting lines** through them which touch, but do not cut the convex hull.



** no three points in $\mathscr{P}$ are collinear.*

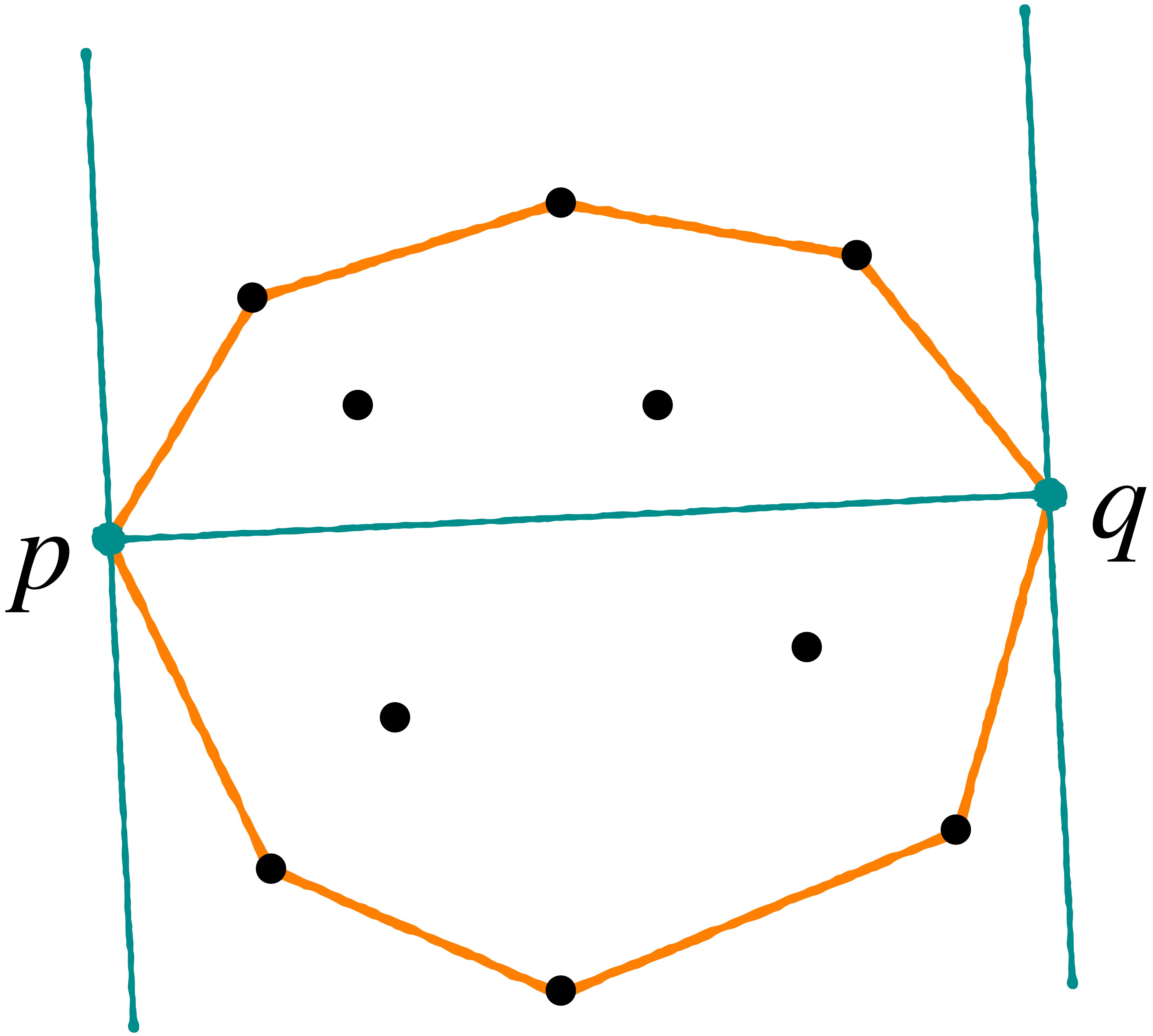
Farthest point pairs
Antipodal pairs

Let $\mathscr{P}$ be set of n points in the Euclidean plane $\mathbb{R}^2$, in general position*.

Lemma E3.1 All farthest pairs of $\mathscr{P}$ consist of two vertices of the convex hull $\text{conv}(\mathscr{P})$.

Definition. Two points $p, q \in \mathscr{P}$ are **antipodal** if there exist parallel **supporting lines** through them which touch, but do not cut the convex hull.

Lemma E3.2 All farthest pairs of $\mathscr{P}$ are also antipodal.



** no three points in $\mathscr{P}$ are collinear.*

Farthest point pairs
Antipodal pairs

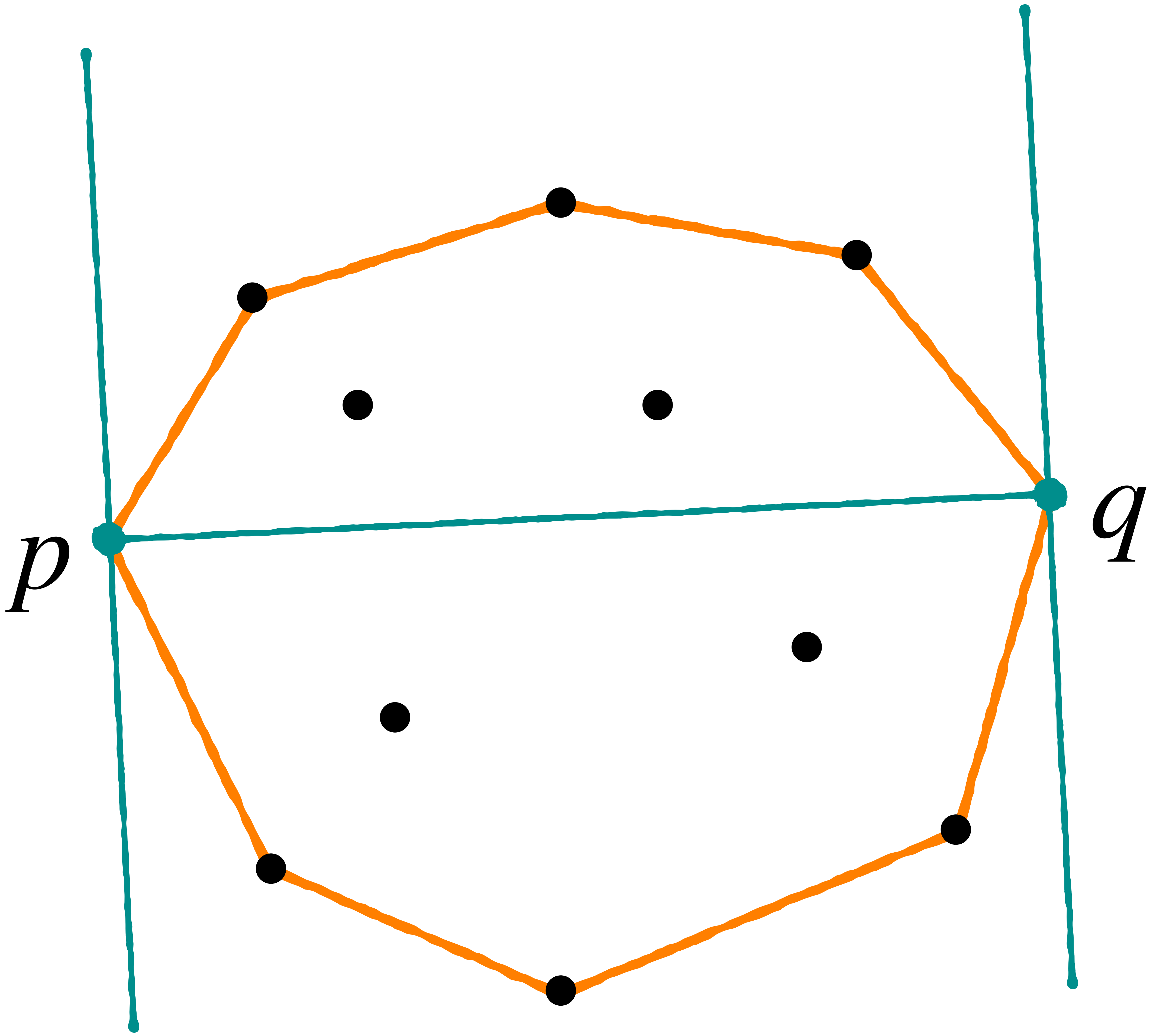
Let $\mathscr{P}$ be set of n points in the Euclidean plane $\mathbb{R}^2$, in general position*.

Lemma E3.1 All farthest pairs of $\mathscr{P}$ consist of two vertices of the convex hull $\text{conv}(\mathscr{P})$.

Definition. Two points $p, q \in \mathscr{P}$ are **antipodal** if there exist parallel **supporting lines** through them which touch, but do not cut the convex hull.

Lemma E3.2 All farthest pairs of $\mathscr{P}$ are also antipodal.

Take 10 minutes to think about this and discuss :)



** no three points in $\mathscr{P}$ are collinear.*

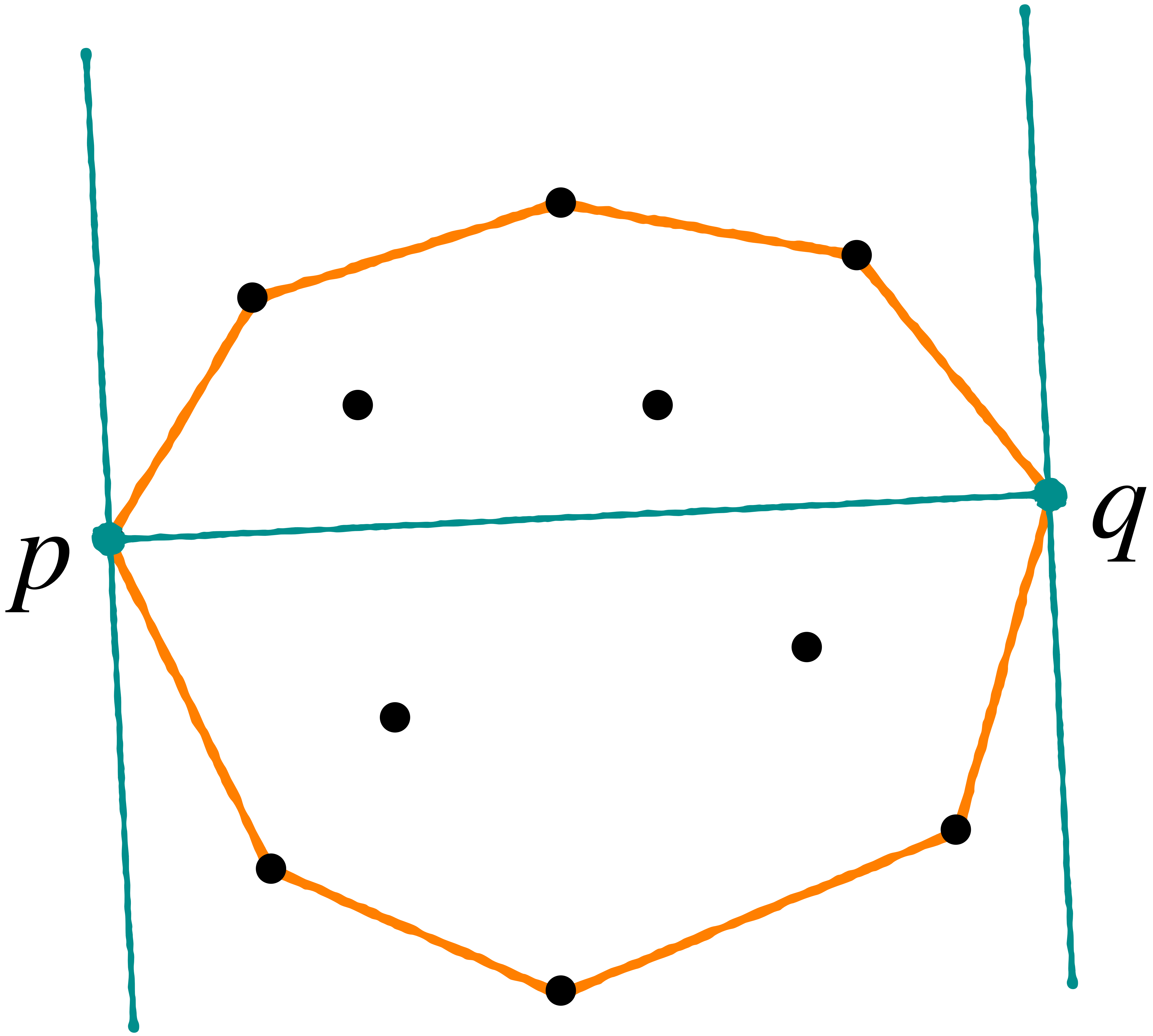
Farthest point pairs
Antipodal pairs

Let $\mathscr{P}$ be set of n points in the Euclidean plane $\mathbb{R}^2$, in general position*.

Lemma E3.1 All farthest pairs of $\mathscr{P}$ consist of two vertices of the convex hull $\text{conv}(\mathscr{P})$.

Definition. Two points $p, q \in \mathscr{P}$ are **antipodal** if there exist parallel **supporting lines** through them which touch, but do not cut the convex hull.

Lemma E3.2 All farthest pairs of $\mathscr{P}$ are also antipodal.



** no three points in $\mathscr{P}$ are collinear.*

Farthest point pairs

Antipodal pairs

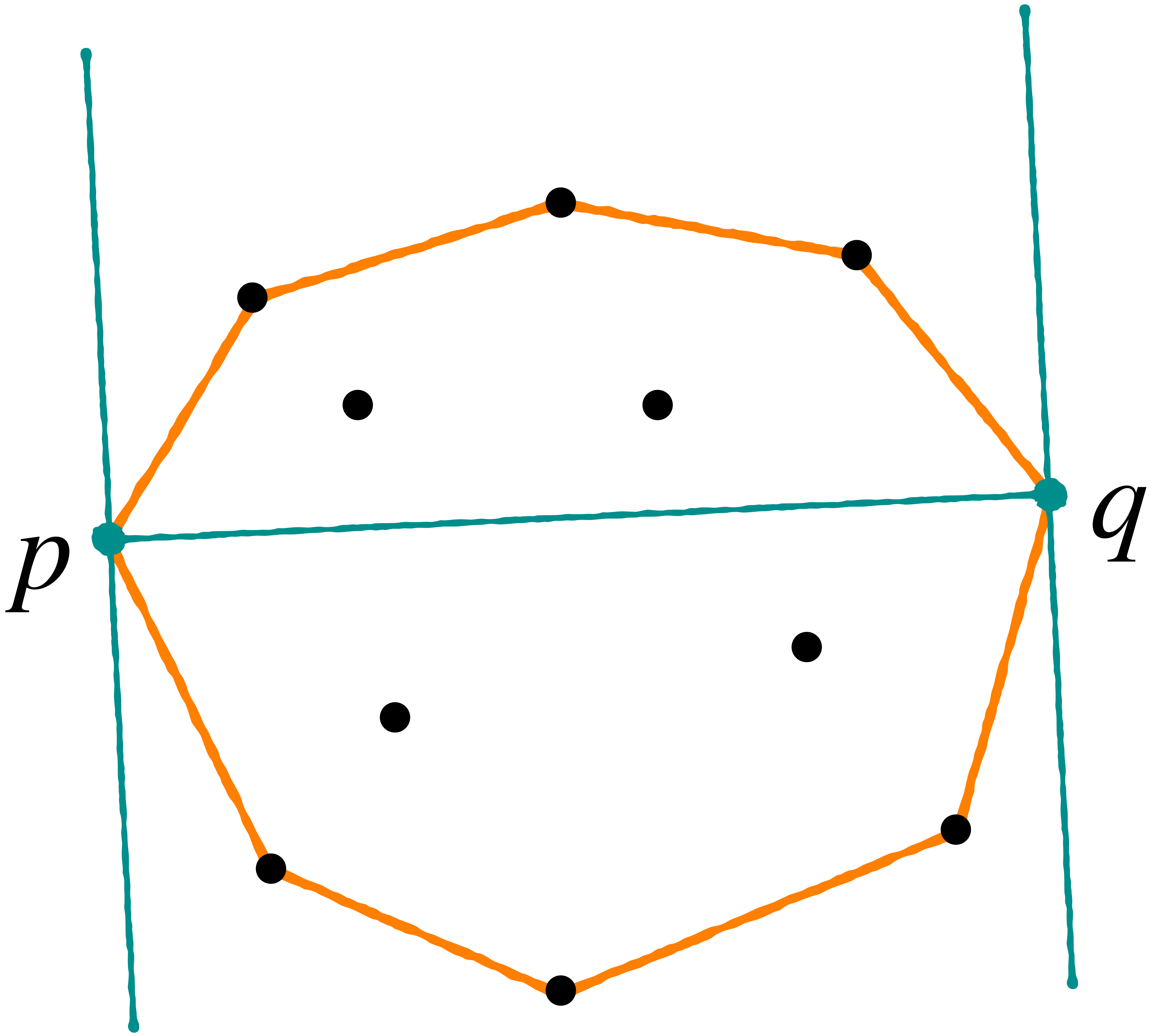
Let $\mathscr{P}$ be set of n points in the Euclidean plane $\mathbb{R}^2$, in general position*.

Lemma E3.1 All farthest pairs of $\mathscr{P}$ consist of two vertices of the convex hull $\mathrm{conv}(\mathscr{P})$.

Definition. Two points $p, q \in \mathscr{P}$ are **antipodal** if there exist parallel **supporting lines** through them which touch, but do not cut the convex hull.

Lemma E3.2 All farthest pairs of $\mathscr{P}$ are also antipodal.

Lemma E3.3 There are $\mathcal{O}(n)$ antipodal pairs in $\mathscr{P}$.



** no three points in $\mathscr{P}$ are collinear.*

Rotating Calipers Algorithm
Michael Shamos, 1978

Let $\mathscr{P}$ be set of n points in the Euclidean plane $\mathbb{R}^2$, in general position*.

Theorem E3.4 All farthest pairs and the diameter of $\mathscr{P}$ can be computed in $\mathcal{O}(n \log n)$.

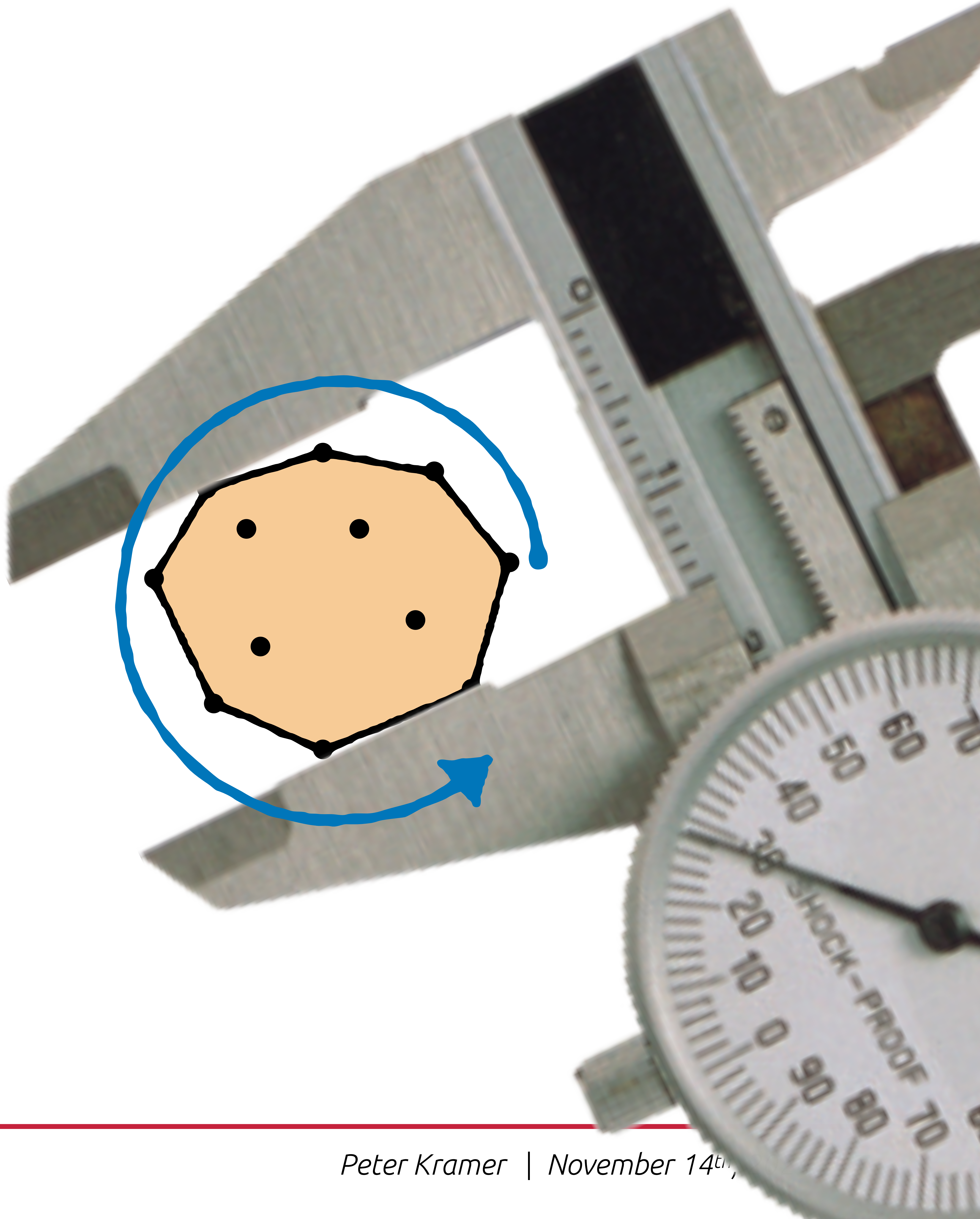
Idea: Compute the convex hull of $\mathscr{P}$, then enumerate **all** antipodal pairs and track the farthest by “rotating” parallel supporting lines around the hull, like calipers.

Rotating Calipers Algorithm
Michael Shamos, 1978

Let $\mathscr{P}$ be set of n points in the Euclidean plane $\mathbb{R}^2$, in general position*.

Theorem E3.4 All farthest pairs and the diameter of $\mathscr{P}$ can be computed in $\mathcal{O}(n \log n)$.

Idea: Compute the convex hull of $\mathscr{P}$, then enumerate **all** antipodal pairs and track the farthest by "rotating" parallel supporting lines around the hull, like calipers.



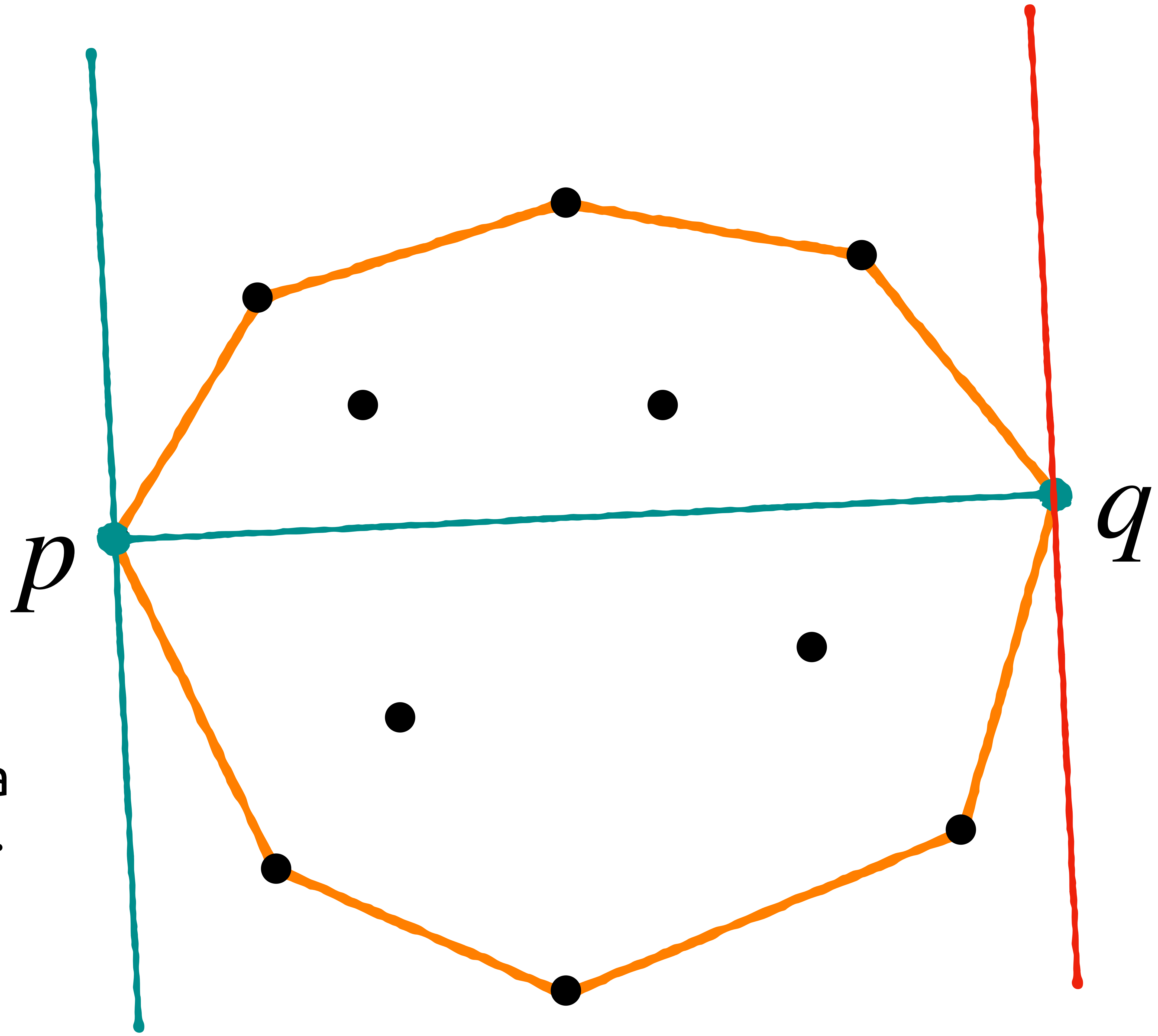
Rotating Calipers Algorithm
Michael Shamos, 1978

Let $\mathscr{P}$ be set of n points in the Euclidean plane $\mathbb{R}^2$, in general position*.

Theorem E3.4 All farthest pairs and the diameter of $\mathscr{P}$ can be computed in $\mathcal{O}(n \log n)$.

Idea: Compute the convex hull of $\mathscr{P}$, then enumerate **all** antipodal pairs and track the farthest by "rotating" parallel supporting lines around the hull, like calipers.

Whenever one of the lines "hits" a vertex, we've found a pair. We just go all the way around and output the pairs.



** no three points in $\mathscr{P}$ are collinear.*

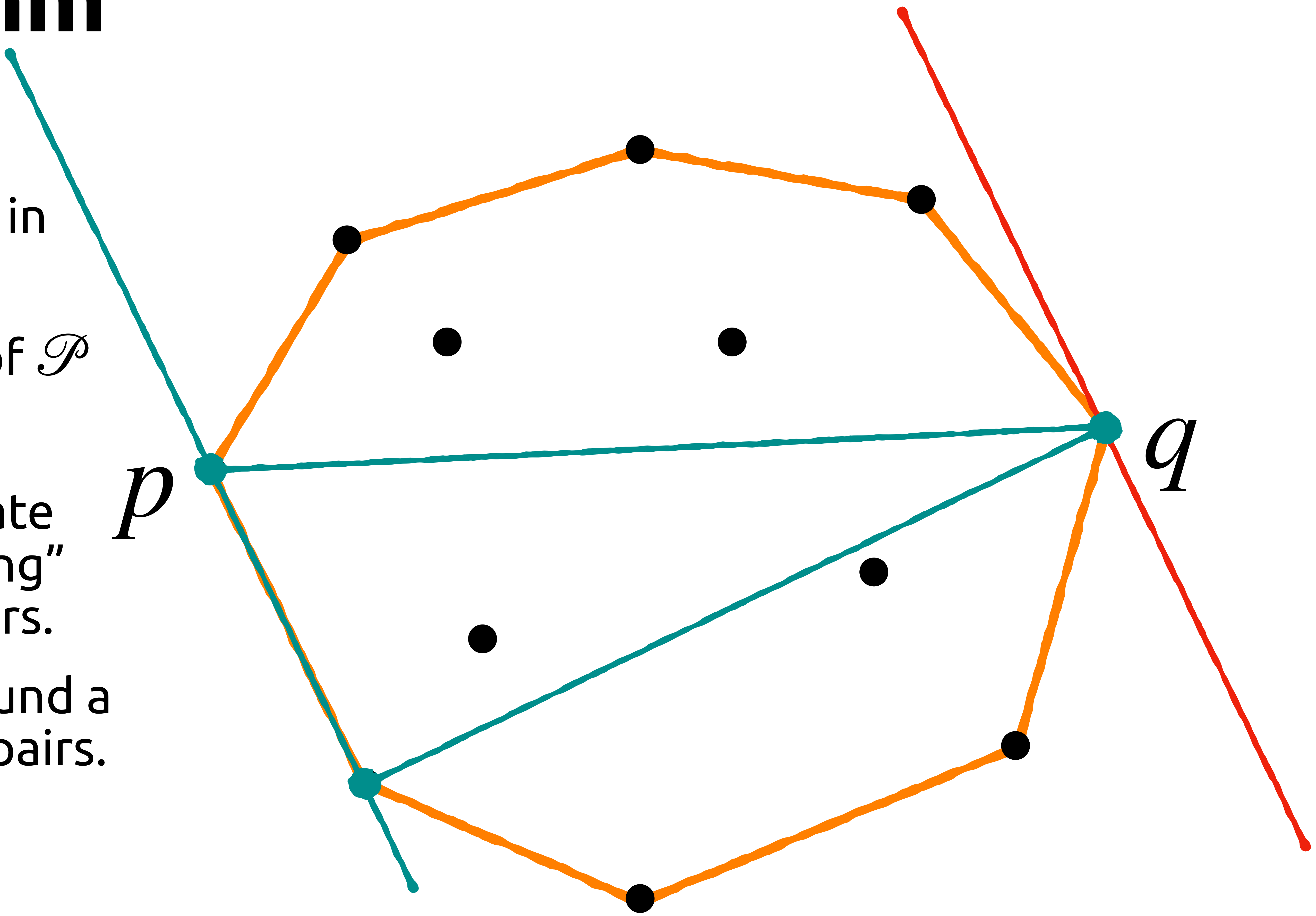
Rotating Calipers Algorithm
Michael Shamos, 1978

Let $\mathscr{P}$ be set of n points in the Euclidean plane $\mathbb{R}^2$, in general position*.

Theorem E3.4 All farthest pairs and the diameter of $\mathscr{P}$ can be computed in $\mathcal{O}(n \log n)$.

Idea: Compute the convex hull of $\mathscr{P}$, then enumerate **all** antipodal pairs and track the farthest by "rotating" parallel supporting lines around the hull, like calipers.

Whenever one of the lines "hits" a vertex, we've found a pair. We just go all the way around and output the pairs.



** no three points in $\mathscr{P}$ are collinear.*

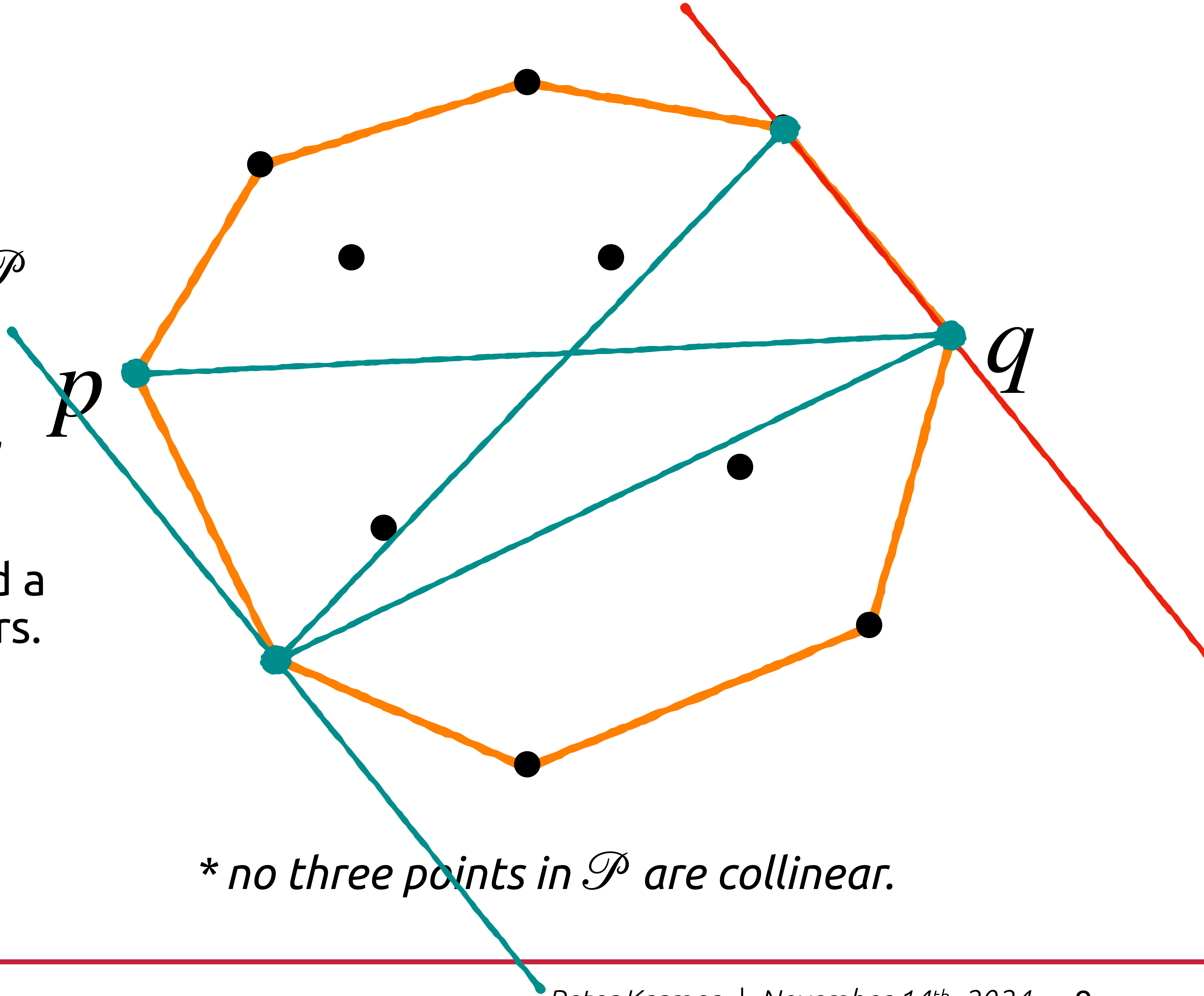
Rotating Calipers Algorithm
Michael Shamos, 1978

Let $\mathscr{P}$ be set of n points in the Euclidean plane $\mathbb{R}^2$, in general position*.

Theorem E3.4 All farthest pairs and the diameter of $\mathscr{P}$ can be computed in $\mathcal{O}(n \log n)$.

Idea: Compute the convex hull of $\mathscr{P}$, then enumerate **all** antipodal pairs and track the farthest by "rotating" parallel supporting lines around the hull, like calipers.

Whenever one of the lines "hits" a vertex, we've found a pair. We just go all the way around and output the pairs.



** no three points in $\mathscr{P}$ are collinear.*

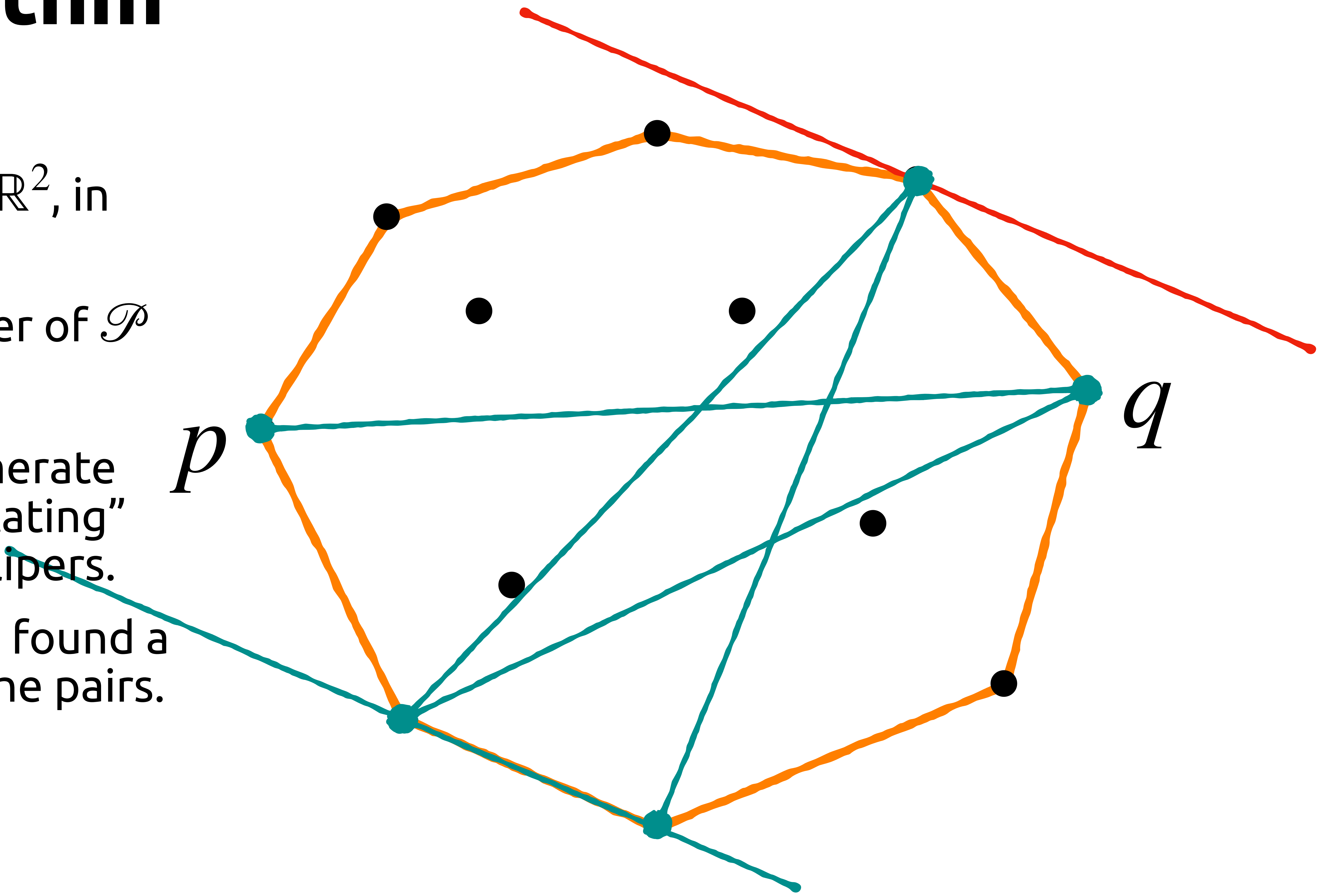
Rotating Calipers Algorithm
Michael Shamos, 1978

Let $\mathscr{P}$ be set of n points in the Euclidean plane $\mathbb{R}^2$, in general position*.

Theorem E3.4 All farthest pairs and the diameter of $\mathscr{P}$ can be computed in $\mathcal{O}(n \log n)$.

Idea: Compute the convex hull of $\mathscr{P}$, then enumerate **all** antipodal pairs and track the farthest by "rotating" parallel supporting lines around the hull, like calipers.

Whenever one of the lines "hits" a vertex, we've found a pair. We just go all the way around and output the pairs.



** no three points in $\mathscr{P}$ are collinear.*

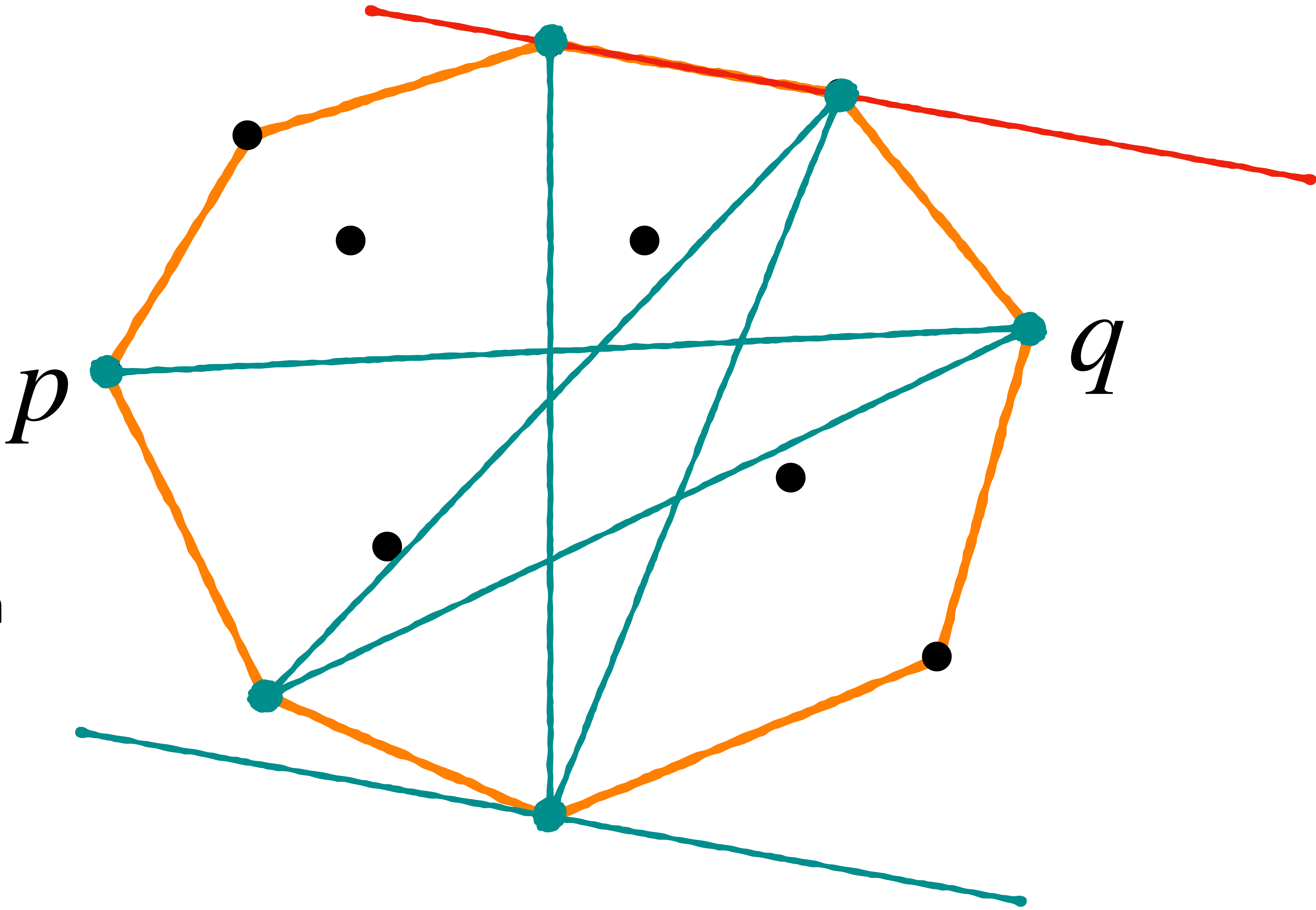
Rotating Calipers Algorithm
Michael Shamos, 1978

Let $\mathscr{P}$ be set of n points in the Euclidean plane $\mathbb{R}^2$, in general position*.

Theorem E3.4 All farthest pairs and the diameter of $\mathscr{P}$ can be computed in $\mathcal{O}(n \log n)$.

Idea: Compute the convex hull of $\mathscr{P}$, then enumerate **all** antipodal pairs and track the farthest by “rotating” parallel supporting lines around the hull, like calipers.

Whenever one of the lines “hits” a vertex, we’ve found a pair. We just go all the way around and output the pairs.



* *no three points in $\mathscr{P}$ are collinear.*

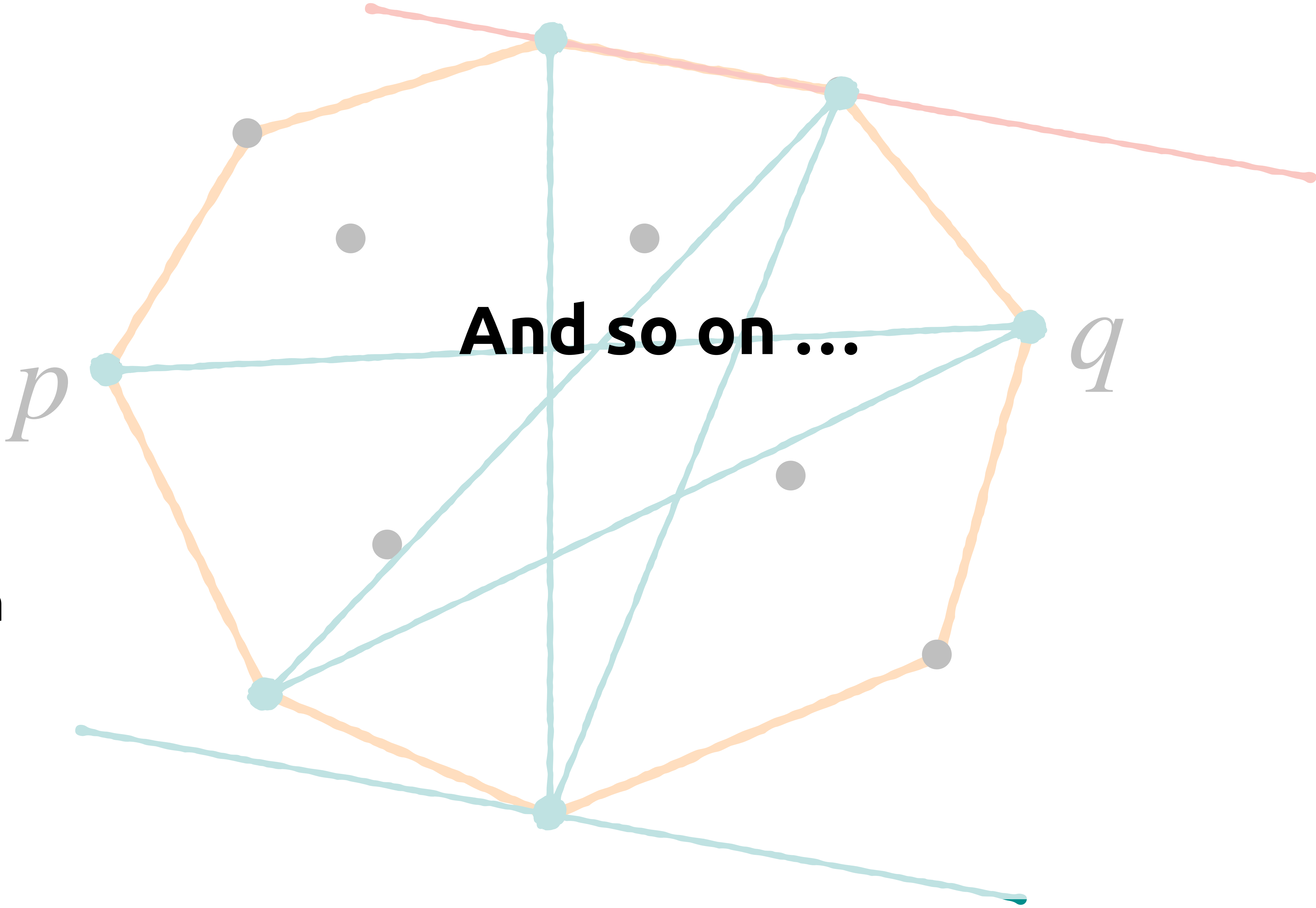
Rotating Calipers Algorithm
Michael Shamos, 1978

Let $\mathscr{P}$ be set of n points in the Euclidean plane $\mathbb{R}^2$, in general position*.

Theorem E3.4 All farthest pairs and the diameter of $\mathscr{P}$ can be computed in $\mathcal{O}(n \log n)$.

Idea: Compute the convex hull of $\mathscr{P}$, then enumerate **all** antipodal pairs and track the farthest by "rotating" parallel supporting lines around the hull, like calipers.

Whenever one of the lines "hits" a vertex, we've found a pair. We just go all the way around and output the pairs.



* *no three points in $\mathscr{P}$ are collinear.*

Rotating Calipers Algorithm
Michael Shamos, 1978

Let $\mathscr{P}$ be set of n points in the Euclidean plane $\mathbb{R}^2$, in general position*.

Theorem E3.4 All farthest pairs and the diameter of $\mathscr{P}$ can be computed in $\mathcal{O}(n \log n)$.

Idea: Compute the convex hull of $\mathscr{P}$, then enumerate **all** antipodal pairs and track the farthest by "rotating" parallel supporting lines around the hull, like calipers.

Whenever one of the lines "hits" a vertex, we've found a pair. We just go all the way around and output the pairs.

* *no three points in $\mathscr{P}$ are collinear.*

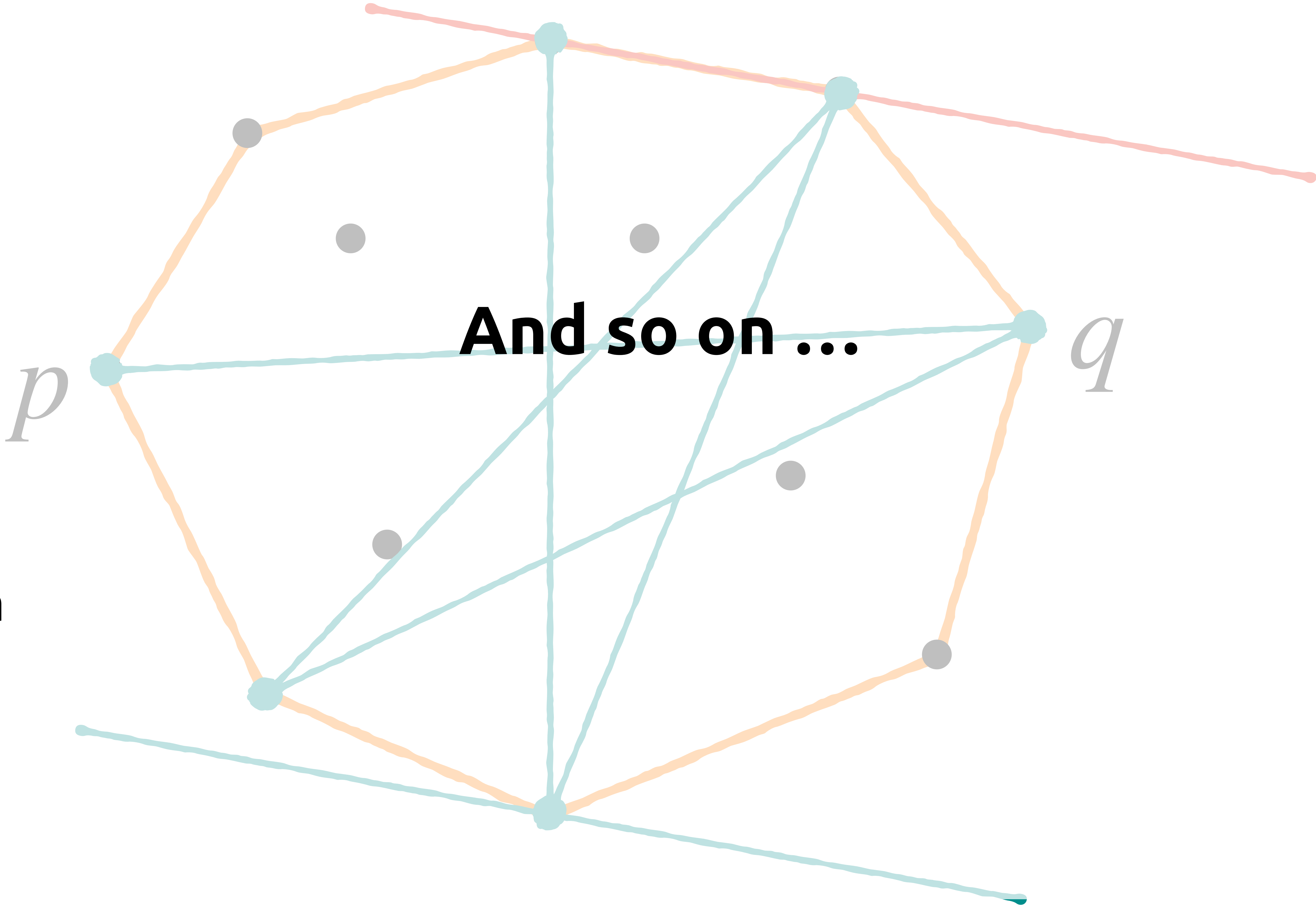
Rotating Calipers Algorithm
Michael Shamos, 1978

Let $\mathscr{P}$ be set of n points in the Euclidean plane $\mathbb{R}^2$, in general position*.

Theorem E3.4 All farthest pairs and the diameter of $\mathscr{P}$ can be computed in $\mathcal{O}(n \log n)$.

Idea: Compute the convex hull of $\mathscr{P}$, then enumerate **all** antipodal pairs and track the farthest by "rotating" parallel supporting lines around the hull, like calipers.

Whenever one of the lines "hits" a vertex, we've found a pair. We just go all the way around and output the pairs.

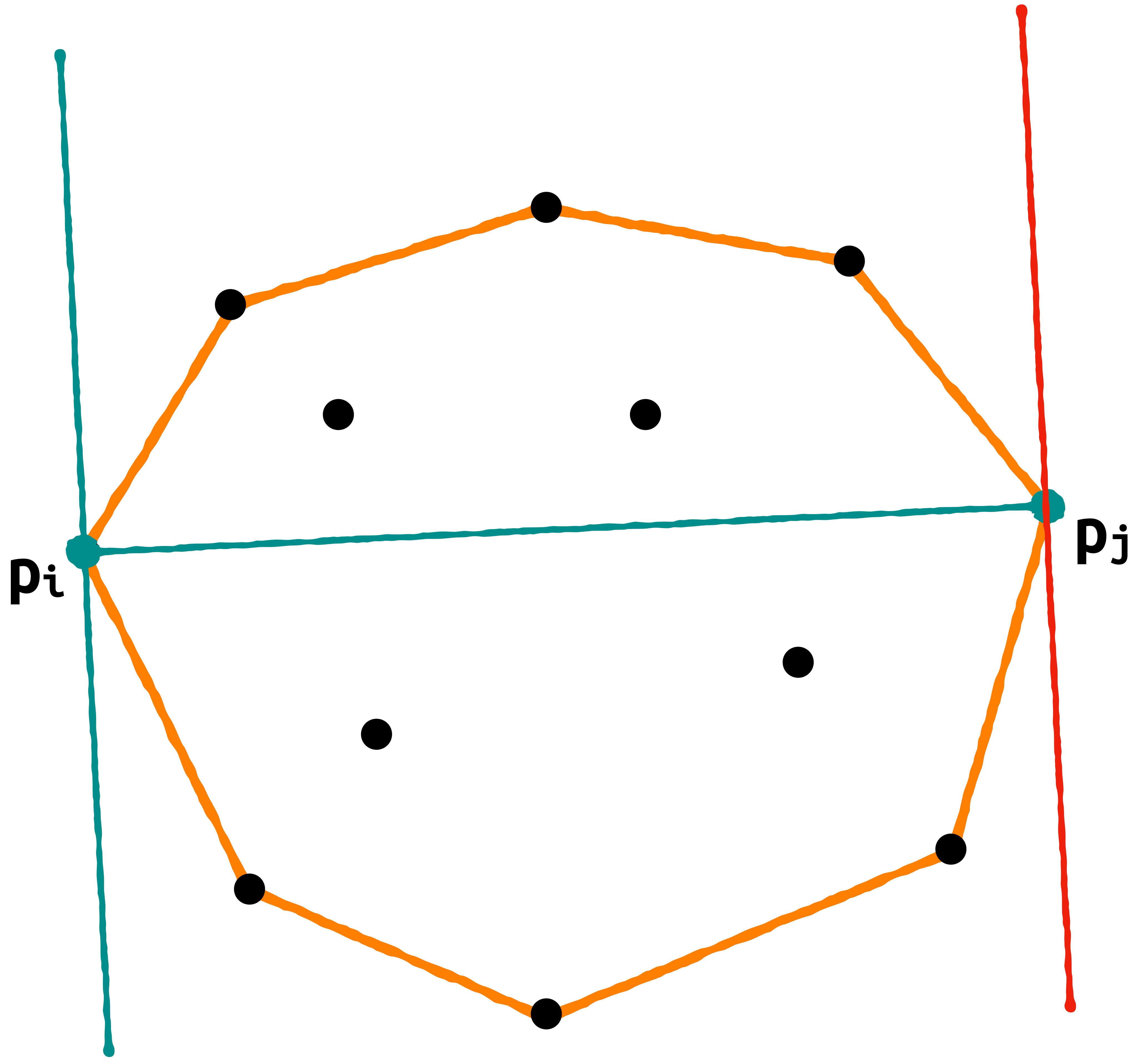


** no three points in $\mathscr{P}$ are collinear.*

Rotating Calipers Algorithm
Michael Shamos, 1978

Theorem E3.4 All farthest pairs and the diameter of of n points $\mathscr{P}$ in general position in the Euclidean plane $\mathbb{R}^2$ can be computed in $\mathcal{O}(n \log n)$.

```
Diameter(n: number, (p₁, …, pₙ): points) : number {
   // Linear probing / brute force – implicitly, i = 1
   find first (i,j) such that (pᵢ,pⱼ) is antipodal
   let diameter = 0
   while (j != n) {
      // Which edge do we hit?
      if A(Δ(pᵢ, pᵢ₊₁, pⱼ₊₁)) > A(Δ(pᵢ, pᵢ₊₁, pⱼ)) {
         ++j
      } else {
         ++i
      }
      // pᵢ,pⱼ is a farthest pair!
      diameter = max(diameter, d(pᵢ,pⱼ))
      // [… edge case handling for parallel lines: Up to 3 more pairs]
   }
   return diameter
}
```



* *no three points in $\mathscr{P}$ are collinear.*

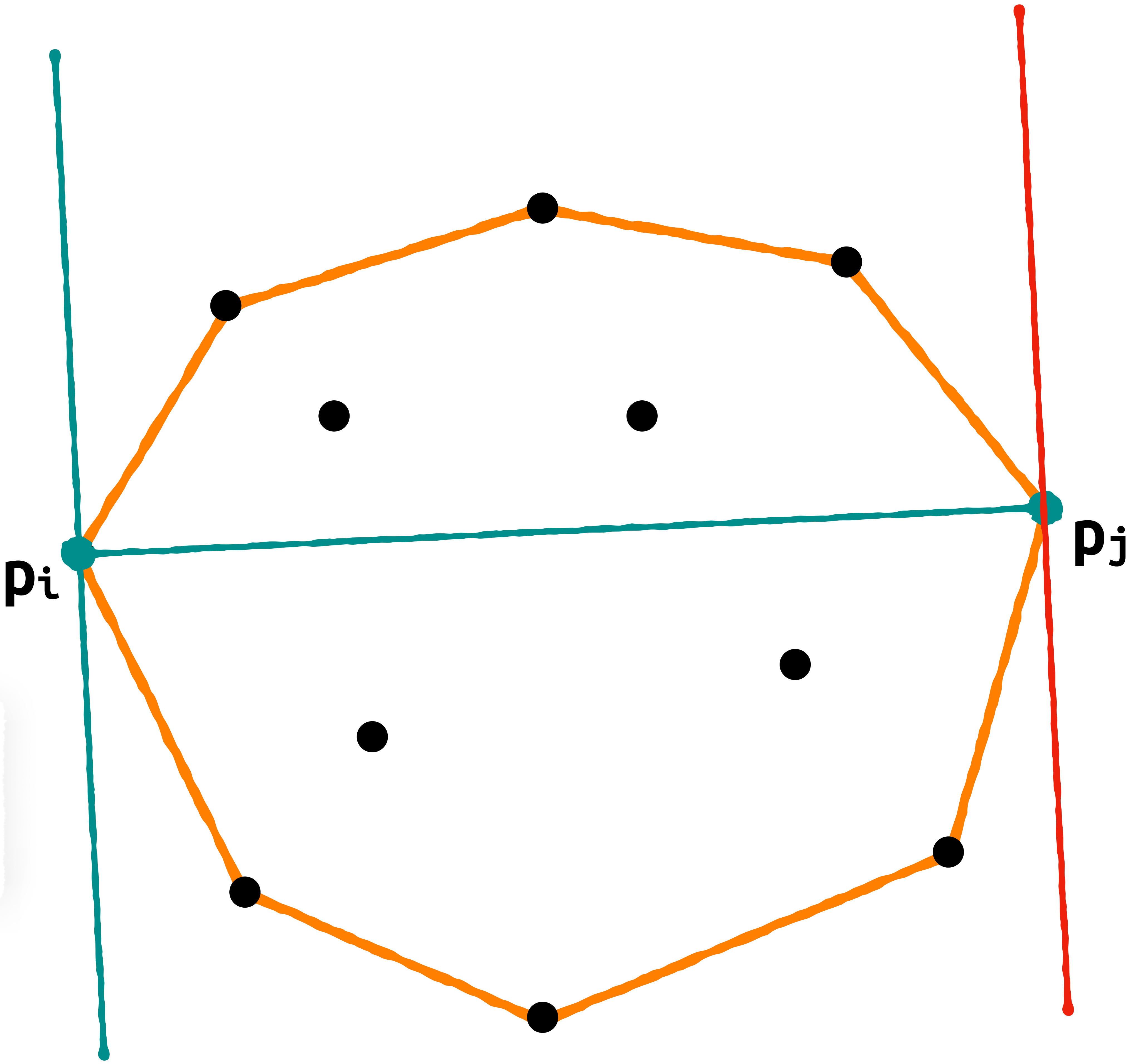
Rotating Calipers Algorithm
Michael Shamos, 1978

Theorem E3.4 All farthest pairs and the diameter of of n points $\mathscr{P}$ in general position in the Euclidean plane $\mathbb{R}^2$ can be computed in $\mathcal{O}(n \log n)$.

```
Diameter(n: number, (p_1, …, p_n): points) : number {
   // Linear probing / brute force – implicitly, i = 1
   find first (i,j) such that (p_i,p_j) is antipodal
   let diameter = 0
   while (j != n) {
      // Which edge do we hit?
      if A(Δ(p_i, p_i+1, p_j+1)) > A(Δ(p_i, p_i+1, p_j)) {
         ++j
      } else {
         ++i
      }
      // p_i,p_j is a farthest pair!
      diameter = max(diameter, d(p_i,p_j))
      // [… edge case handling for parallel lines: Up to 3 more pairs]
   }
   return diameter
}
```

$A(\triangle(p,q,r)) > 0 \Leftrightarrow p,q,r$ oriented in counterclockwise (CCW) order

- $A(\triangle(p,q,r)) \begin{Bmatrix} > 0 \\ = 0 \\ < 0 \end{Bmatrix} \Leftrightarrow p,q,r \begin{Bmatrix} \text{Left turn} \\ \text{collinear} \\ \text{Right turn} \end{Bmatrix}$

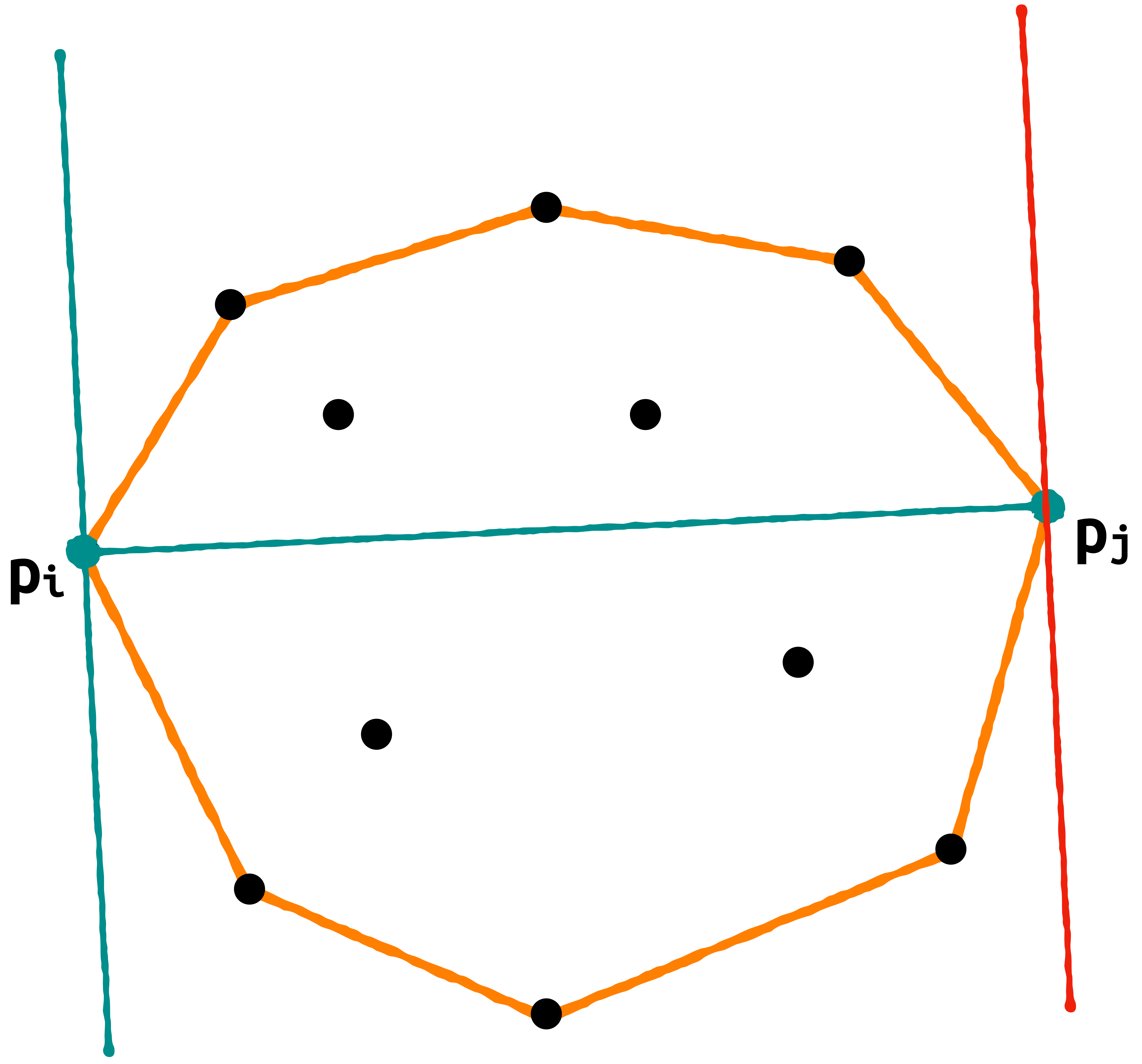


* no three points in $\mathscr{P}$ are collinear.

Rotating Calipers Algorithm
Michael Shamos, 1978

Theorem E3.4 All farthest pairs and the diameter of of n points $\mathscr{P}$ in general position in the Euclidean plane $\mathbb{R}^2$ can be computed in $\mathcal{O}(n \log n)$.

```
Diameter(n: number, (p₁, …, pₙ): points) : number {
   // Linear probing / brute force – implicitly, i = 1
   find first (i,j) such that (pᵢ,pⱼ) is antipodal
   let diameter = 0
   while (j != n) {
      // Which edge do we hit?
      if A(Δ(pᵢ, pᵢ₊₁, pⱼ₊₁)) > A(Δ(pᵢ, pᵢ₊₁, pⱼ)) {
         ++j
      } else {
         ++i
      }
      // pᵢ,pⱼ is a farthest pair!
      diameter = max(diameter, d(pᵢ,pⱼ))
      // [… edge case handling for parallel lines: Up to 3 more pairs]
   }
   return diameter
}
```

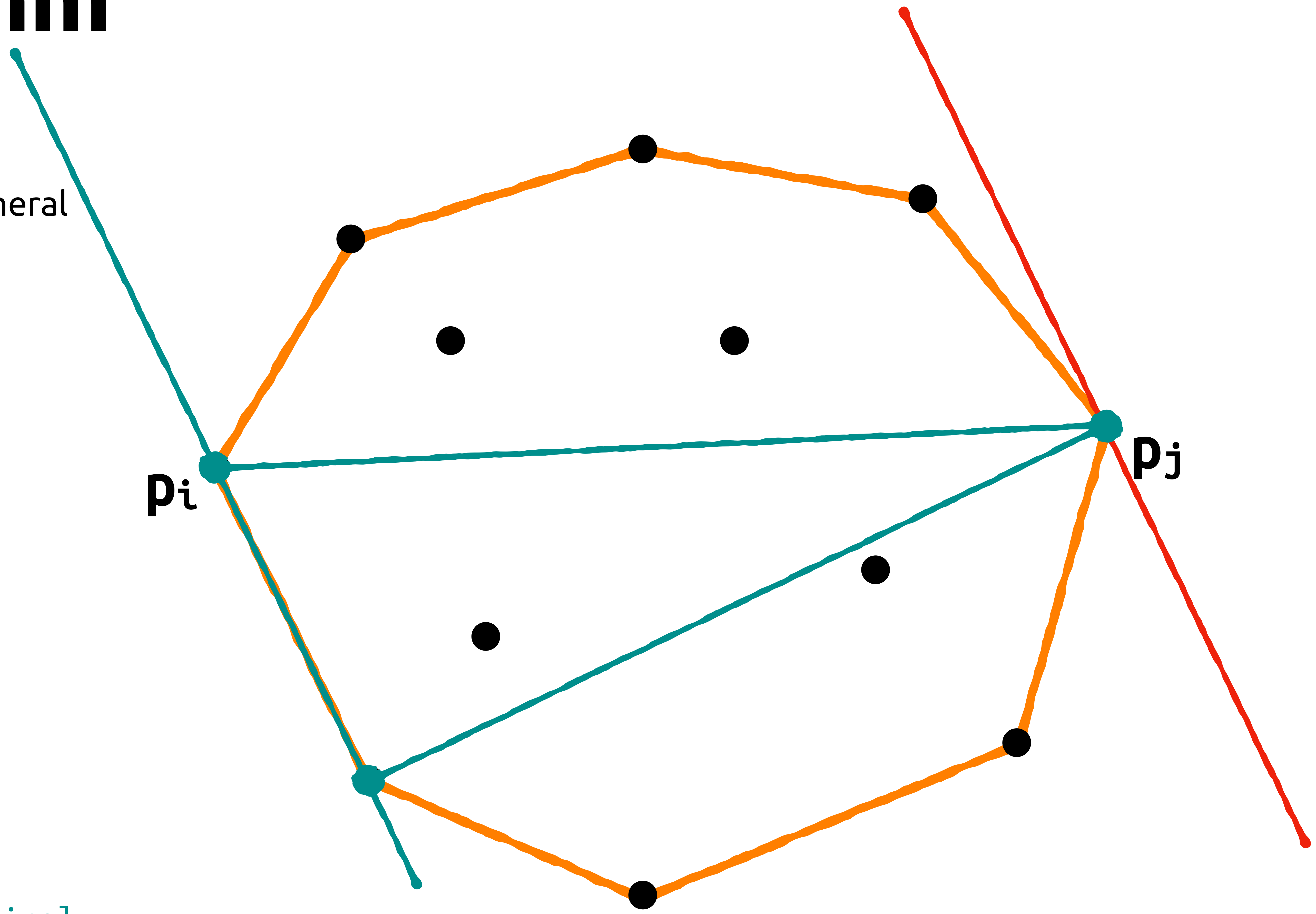


* *no three points in $\mathscr{P}$ are collinear.*

Rotating Calipers Algorithm
Michael Shamos, 1978

Theorem E3.4 All farthest pairs and the diameter of of n points $\mathscr{P}$ in general position in the Euclidean plane $\mathbb{R}^2$ can be computed in $\mathcal{O}(n \log n)$.

```
Diameter(n: number, (p₁, …, pₙ): points) : number {
   // Linear probing / brute force – implicitly, i = 1
   find first (i,j) such that (pᵢ,pⱼ) is antipodal
   let diameter = 0
   while (j != n) {
      // Which edge do we hit?
      if A(Δ(pᵢ, pᵢ₊₁, pⱼ₊₁)) > A(Δ(pᵢ, pᵢ₊₁, pⱼ)) {
         ++j
      } else {
         ++i
      }
      // pᵢ,pⱼ is a farthest pair!
      diameter = max(diameter, d(pᵢ,pⱼ))
      // [… edge case handling for parallel lines: Up to 3 more pairs]
   }
   return diameter
}
```

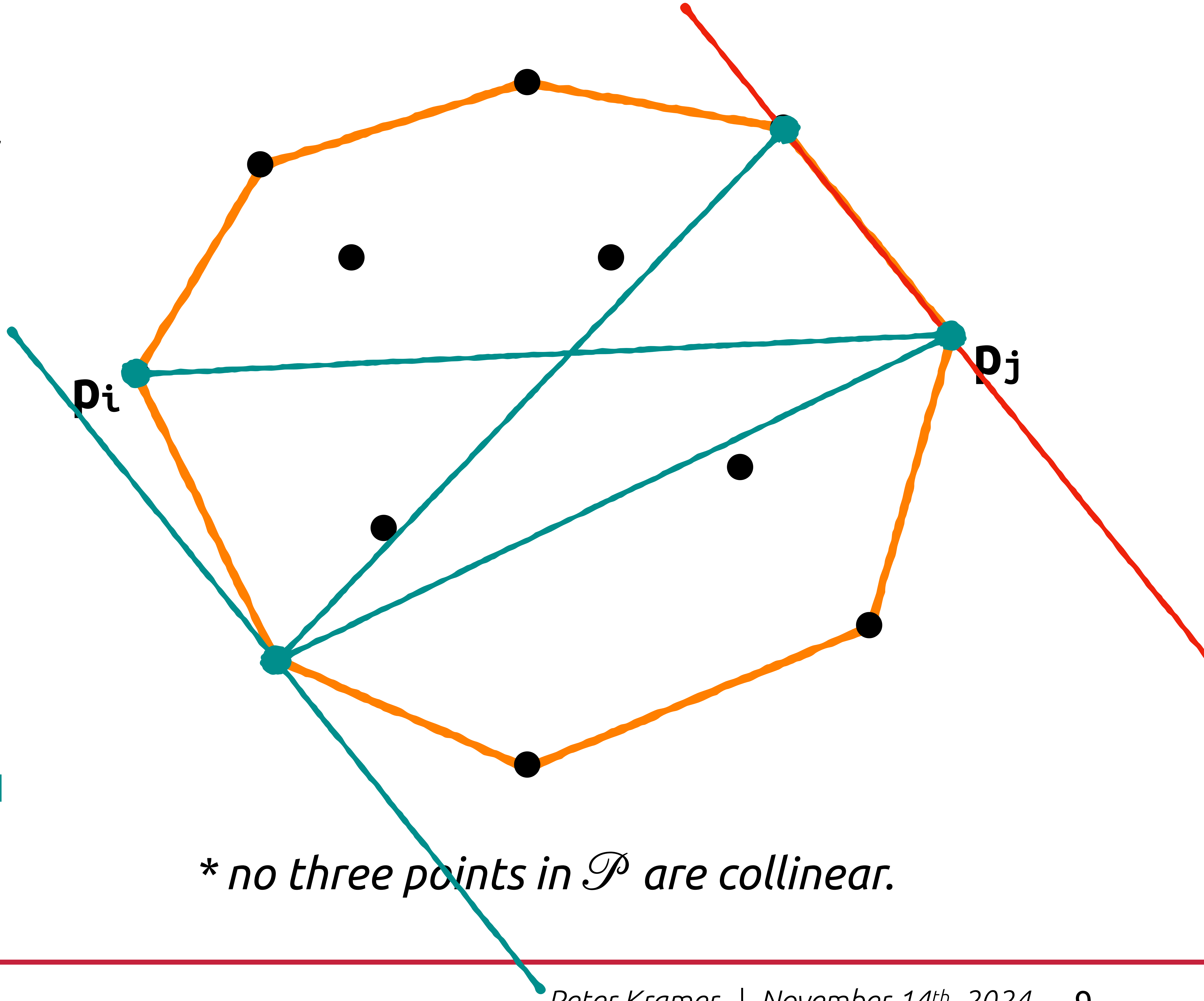


* *no three points in $\mathscr{P}$ are collinear.*

Rotating Calipers Algorithm
Michael Shamos, 1978

Theorem E3.4 All farthest pairs and the diameter of of n points $\mathscr{P}$ in general position in the Euclidean plane $\mathbb{R}^2$ can be computed in $\mathcal{O}(n \log n)$.

```
Diameter(n: number, (p₁, …, pₙ): points) : number {
   // Linear probing / brute force – implicitly, i = 1
   find first (i,j) such that (pᵢ,pⱼ) is antipodal
   let diameter = 0
   while (j != n) {
      // Which edge do we hit?
      if A(Δ(pᵢ, pᵢ₊₁, pⱼ₊₁)) > A(Δ(pᵢ, pᵢ₊₁, pⱼ)) {
         ++j
      } else {
         ++i
      }
      // pᵢ,pⱼ is a farthest pair!
      diameter = max(diameter, d(pᵢ,pⱼ))
      // [… edge case handling for parallel lines: Up to 3 more pairs]
   }
   return diameter
}
```

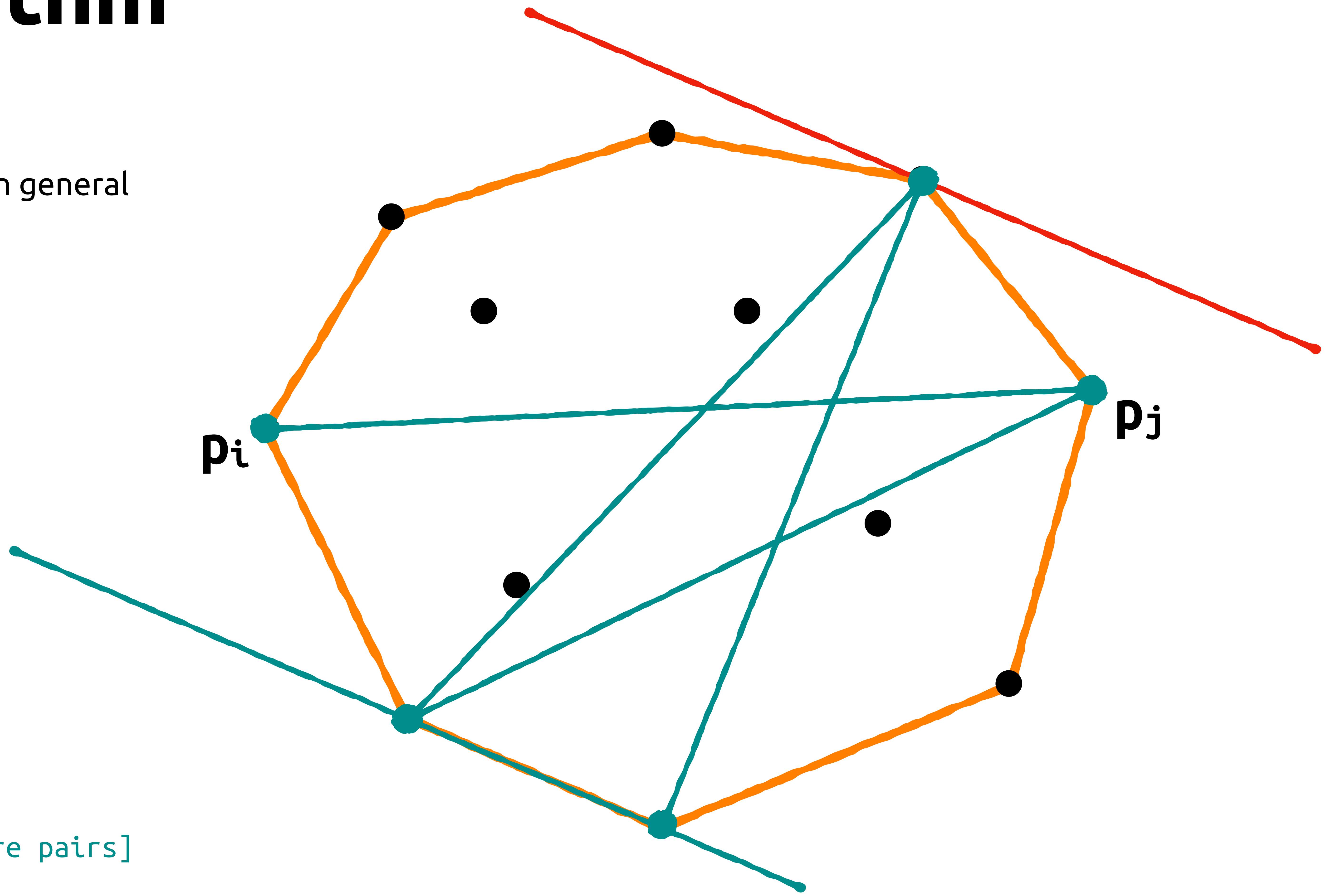


* *no three points in $\mathscr{P}$ are collinear.*

Rotating Calipers Algorithm
Michael Shamos, 1978

Theorem E3.4 All farthest pairs and the diameter of of n points $\mathscr{P}$ in general position in the Euclidean plane $\mathbb{R}^2$ can be computed in $\mathcal{O}(n \log n)$.

```
Diameter(n: number, (p₁, …, pₙ): points) : number {
   // Linear probing / brute force – implicitly, i = 1
   find first (i,j) such that (pᵢ,pⱼ) is antipodal
   let diameter = 0
   while (j != n) {
      // Which edge do we hit?
      if A(Δ(pᵢ, pᵢ₊₁, pⱼ₊₁)) > A(Δ(pᵢ, pᵢ₊₁, pⱼ)) {
         ++j
      } else {
         ++i
      }
      // pᵢ,pⱼ is a farthest pair!
      diameter = max(diameter, d(pᵢ,pⱼ))
      // [… edge case handling for parallel lines: Up to 3 more pairs]
   }
   return diameter
}
```

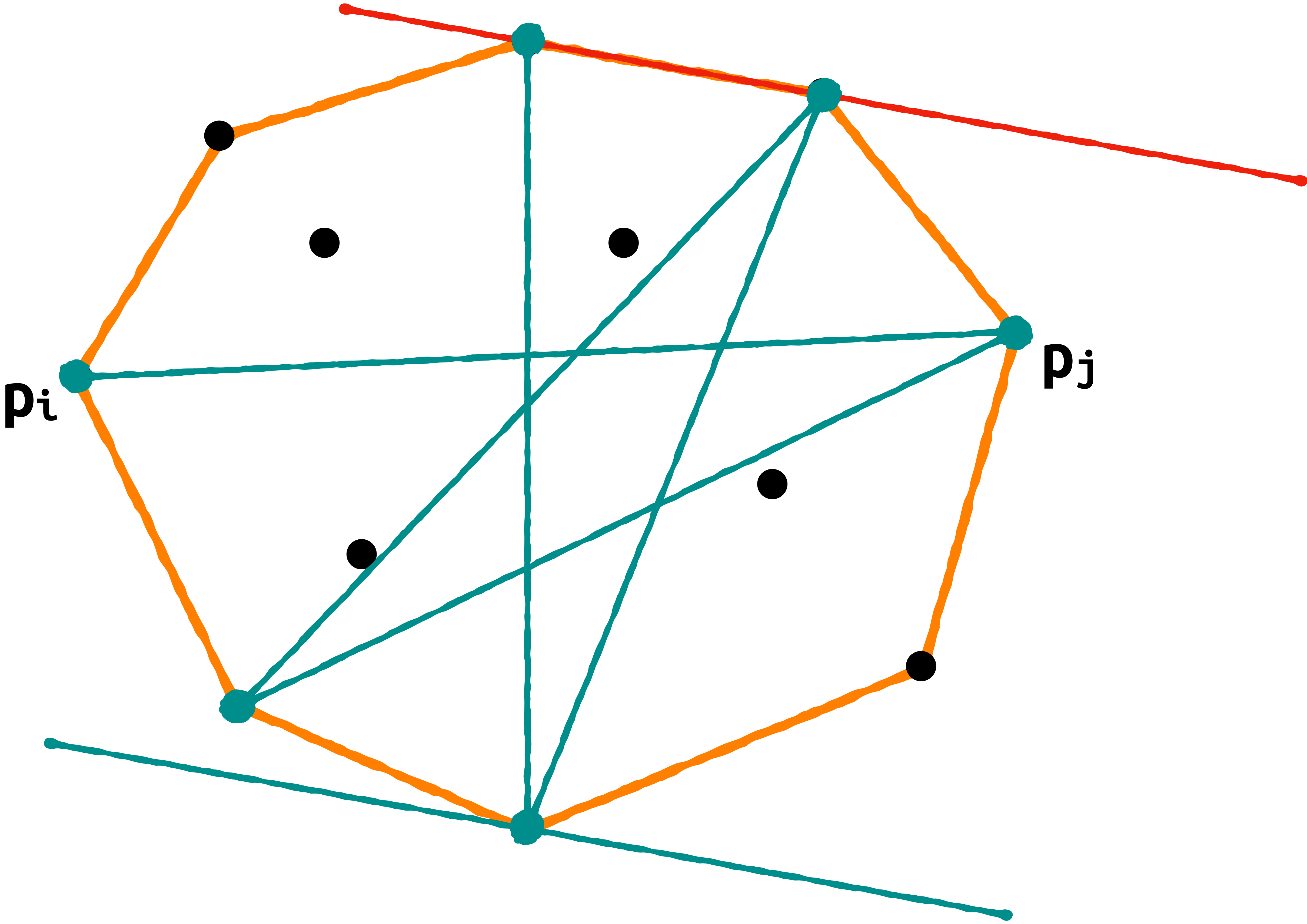


* *no three points in $\mathscr{P}$ are collinear.*

Rotating Calipers Algorithm
Michael Shamos, 1978

Theorem E3.4 All farthest pairs and the diameter of of n points $\mathscr{P}$ in general position in the Euclidean plane $\mathbb{R}^2$ can be computed in $\mathcal{O}(n \log n)$.

```
Diameter(n: number, (p₁, …, pₙ): points) : number {
   // Linear probing / brute force – implicitly, i = 1
   find first (i,j) such that (pᵢ,pⱼ) is antipodal
   let diameter = 0
   while (j != n) {
      // Which edge do we hit?
      if A(Δ(pᵢ, pᵢ₊₁, pⱼ₊₁)) > A(Δ(pᵢ, pᵢ₊₁, pⱼ)) {
         ++j
      } else {
         ++i
      }
      // pᵢ,pⱼ is a farthest pair!
      diameter = max(diameter, d(pᵢ,pⱼ))
      // [… edge case handling for parallel lines: Up to 3 more pairs]
   }
   return diameter
}
```

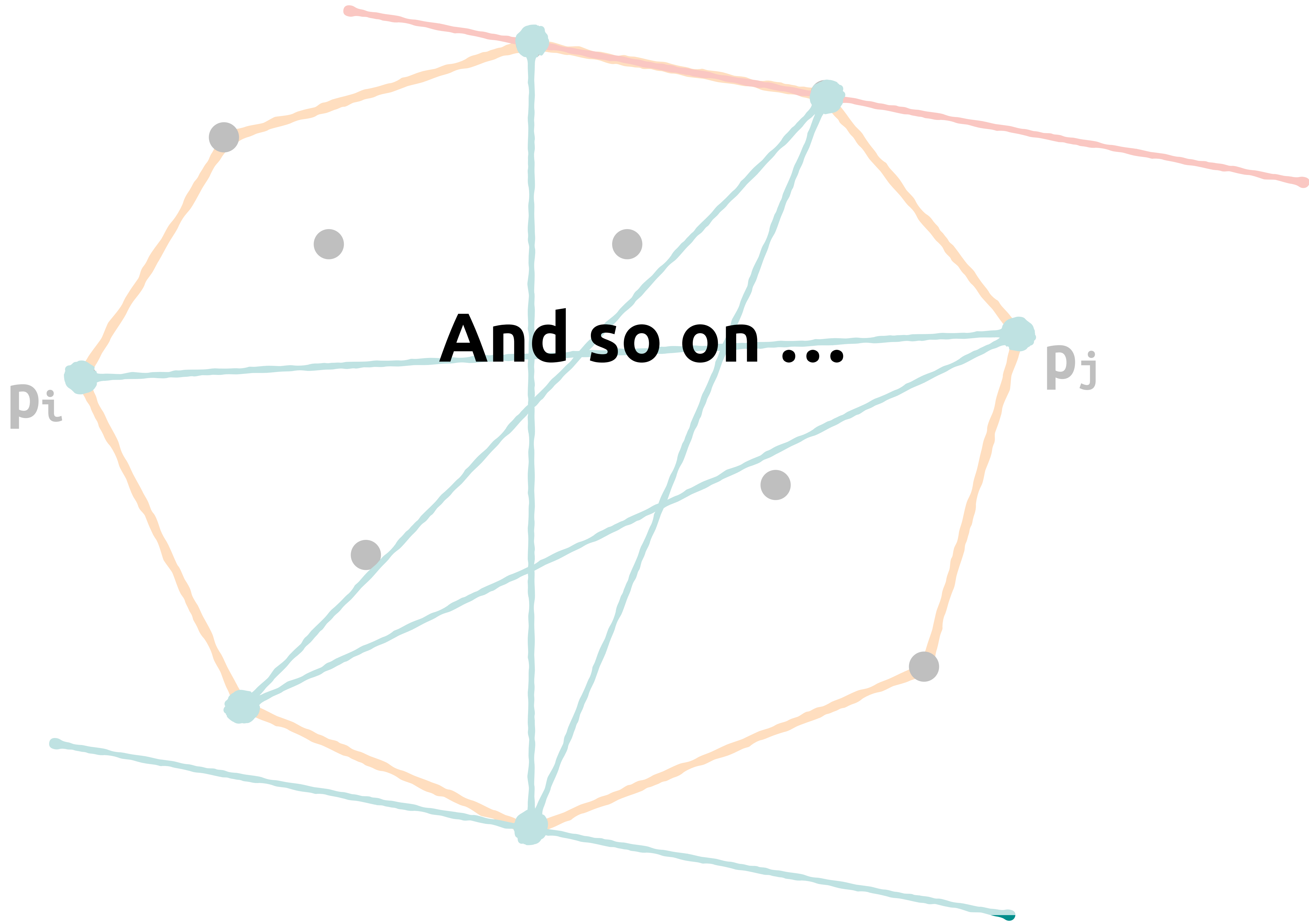


** no three points in $\mathscr{P}$ are collinear.*

Rotating Calipers Algorithm
Michael Shamos, 1978

Theorem E3.4 All farthest pairs and the diameter of of n points $\mathscr{P}$ in general position in the Euclidean plane $\mathbb{R}^2$ can be computed in $\mathcal{O}(n \log n)$.

```
Diameter(n: number, (p1, …, pn): points) : number {
   // Linear probing / brute force – implicitly, i = 1
   find first (i,j) such that (pi,pj) is antipodal
   let diameter = 0
   while (j != n) {
      // Which edge do we hit?
      if A(Δ(pi, pi+1, pj+1)) > A(Δ(pi, pi+1, pj)) {
         ++j
      } else {
         ++i
      }
      // pi,pj is a farthest pair!
      diameter = max(diameter, d(pi,pj))
      // [… edge case handling for parallel lines: Up to 3 more pairs]
   }
   return diameter
}
```

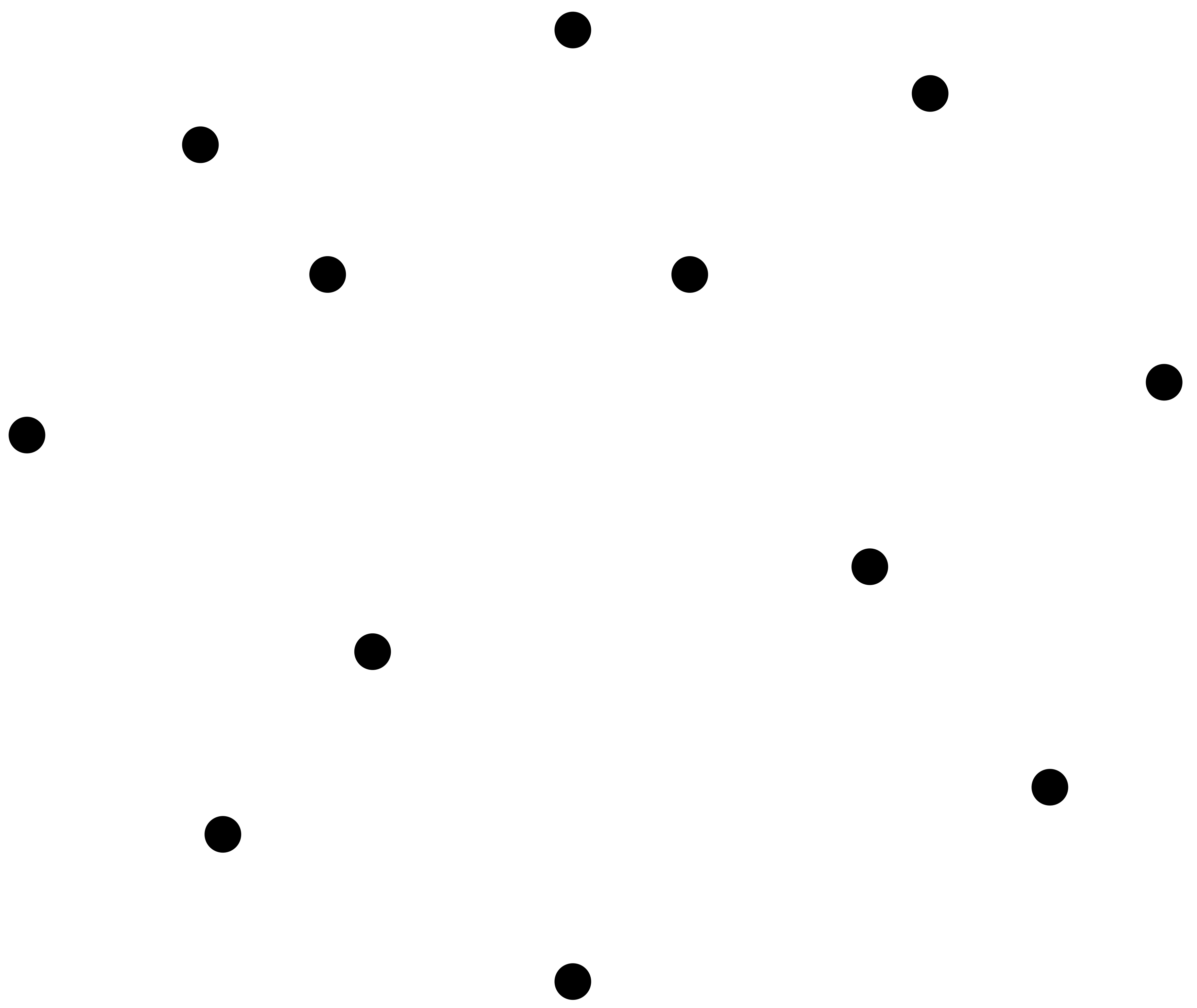


* *no three points in $\mathscr{P}$ are collinear.*

Farthest point pairs
Convex hull

Let $\mathscr{P}$ be set of n points in the Euclidean plane $\mathbb{R}^2$, in general position*.

Theorem E3.4 All farthest pairs and the diameter of $\mathscr{P}$ can be computed in $\mathcal{O}(n \log n)$.

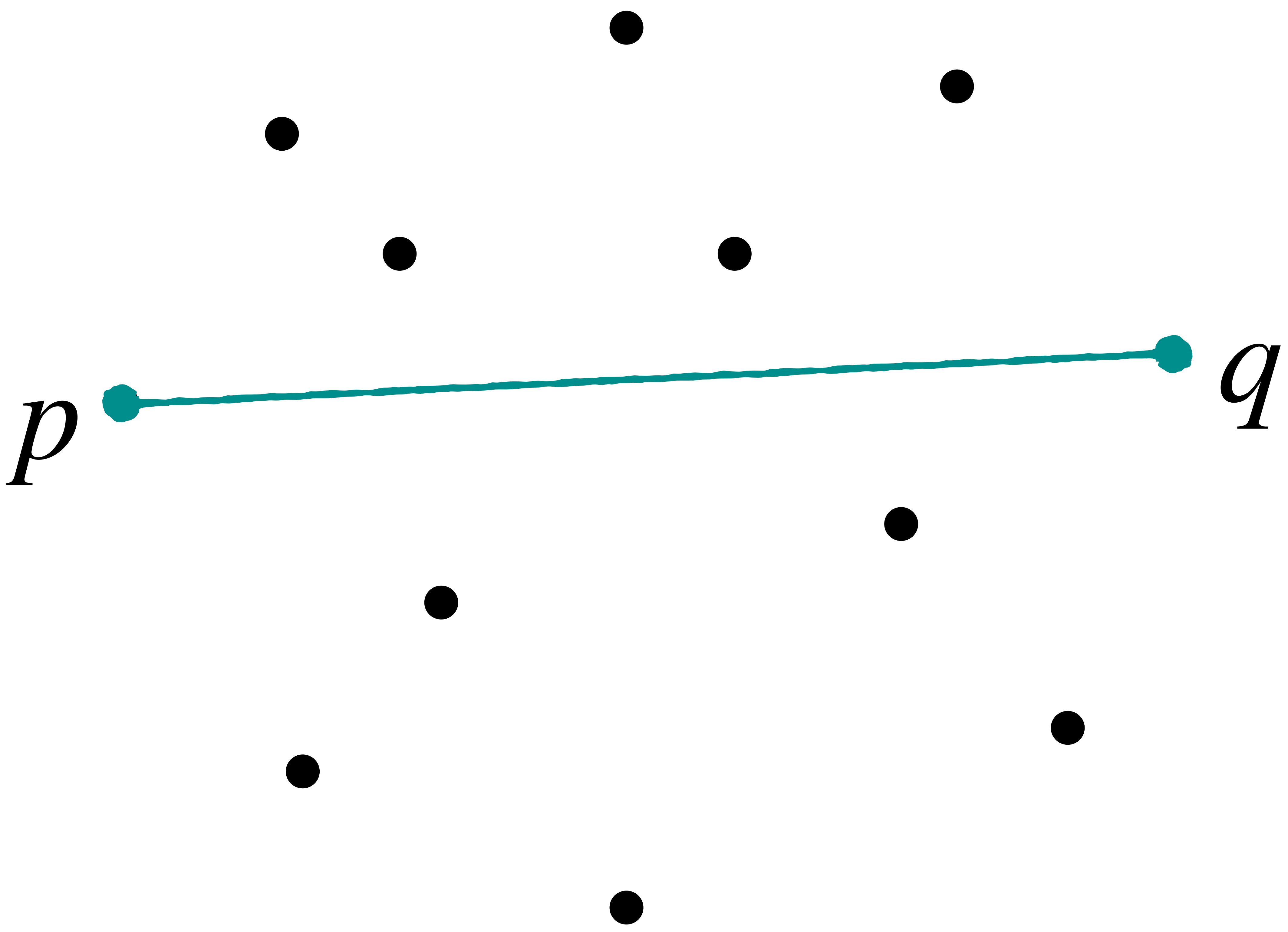


** no three points in $\mathscr{P}$ are collinear.*

Farthest point pairs
Convex hull

Let $\mathscr{P}$ be set of n points in the Euclidean plane $\mathbb{R}^2$, in general position*.

Theorem E3.4 All farthest pairs and the diameter of $\mathscr{P}$ can be computed in $\mathcal{O}(n \log n)$.



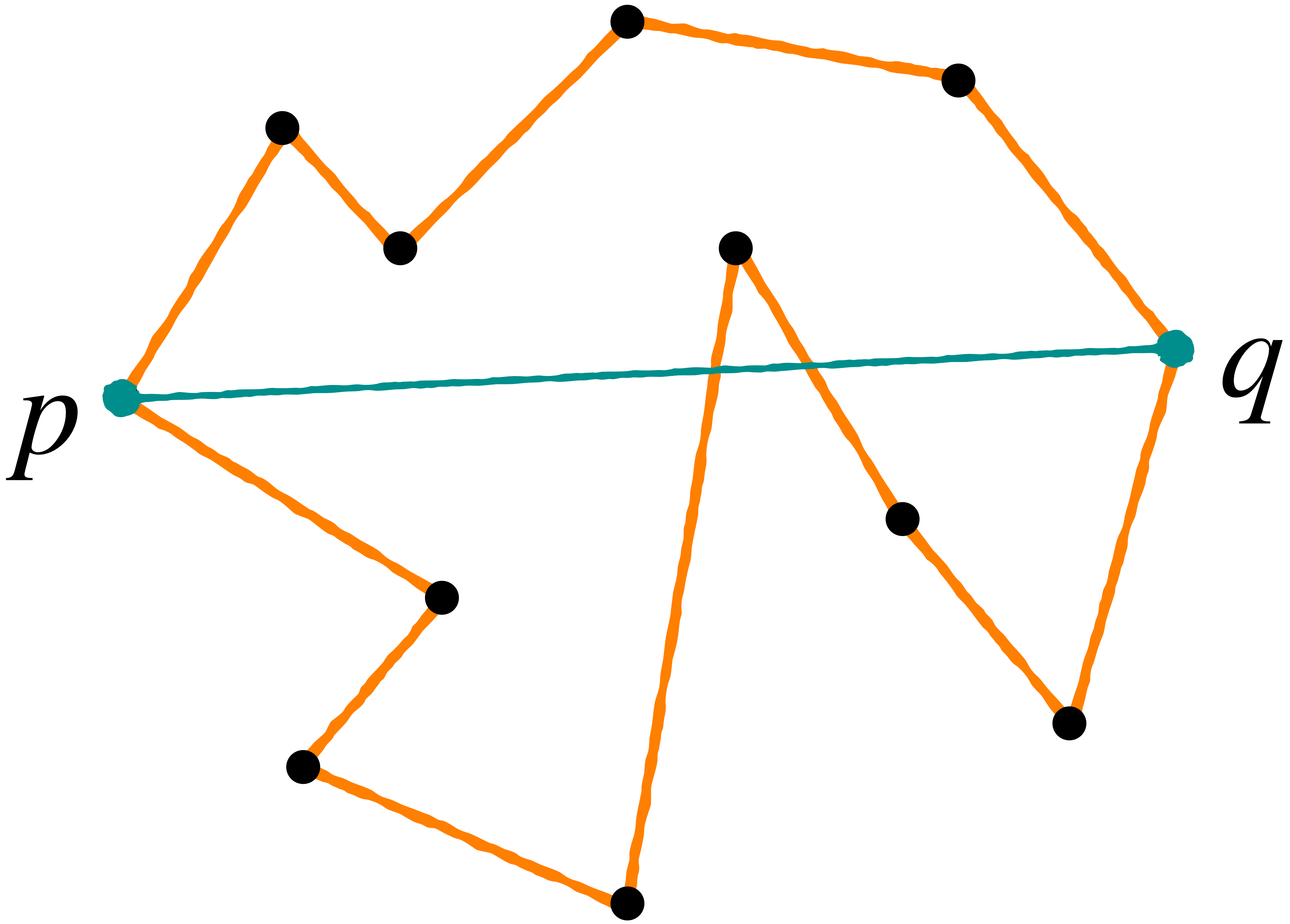
** no three points in $\mathscr{P}$ are collinear.*

Farthest point pairs
Convex hull

Let $\mathscr{P}$ be set of n points in the Euclidean plane $\mathbb{R}^2$, in general position*.

Theorem E3.4 All farthest pairs and the diameter of $\mathscr{P}$ can be computed in $\mathcal{O}(n \log n)$.

Theorem E3.5 All farthest pairs and the diameter of an n-vertex simple polygon P can be computed in …?



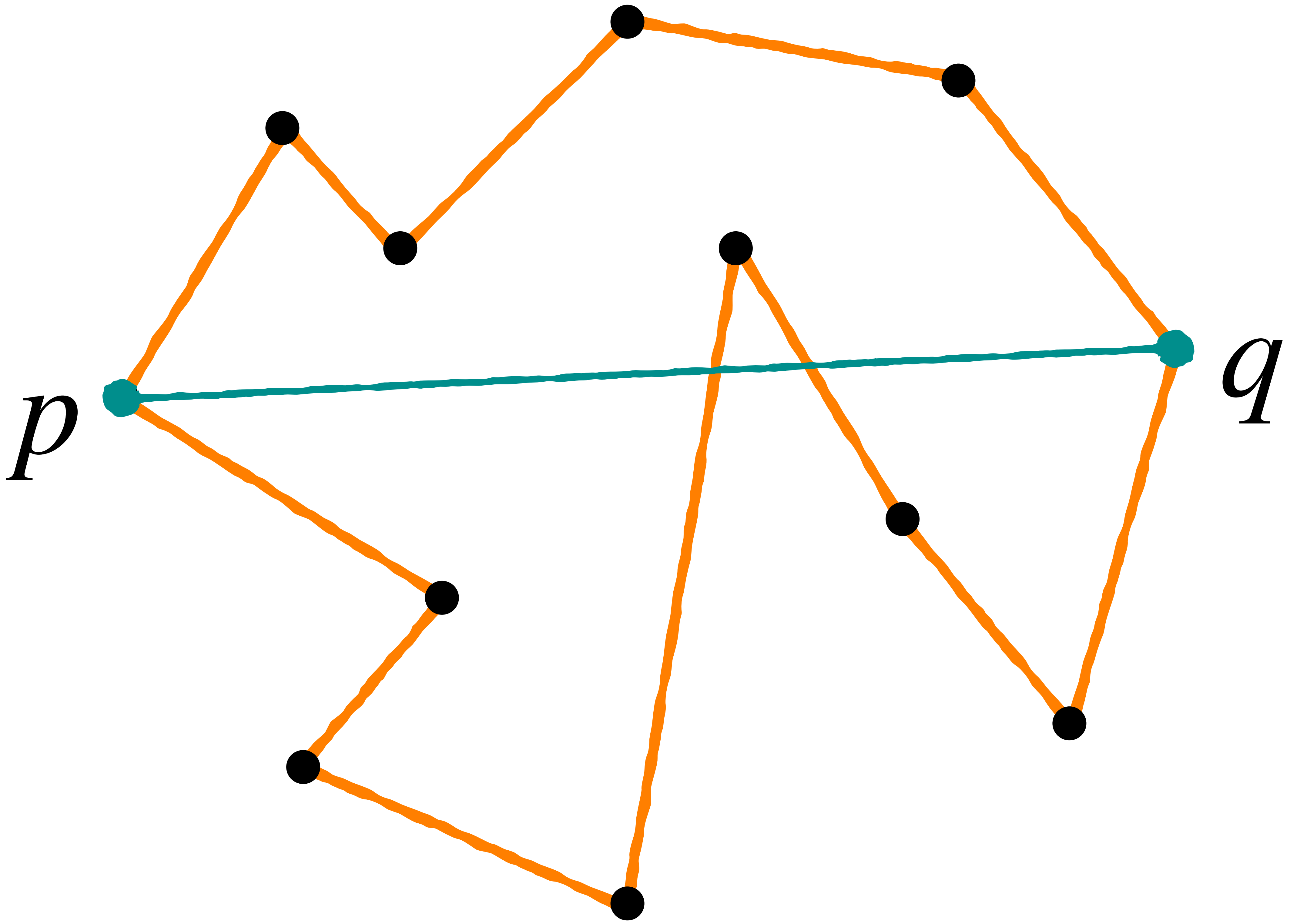
* *no three points in $\mathscr{P}$ are collinear.*

Farthest point pairs
Convex hull

Let $\mathscr{P}$ be set of n points in the Euclidean plane $\mathbb{R}^2$, in general position*.

Theorem E3.4 All farthest pairs and the diameter of $\mathscr{P}$ can be computed in $\mathcal{O}(n \log n)$.

Theorem E3.5 All farthest pairs and the diameter of an n-vertex simple polygon P can be computed in $\mathcal{O}(n)$.



* *no three points in $\mathscr{P}$ are collinear.*

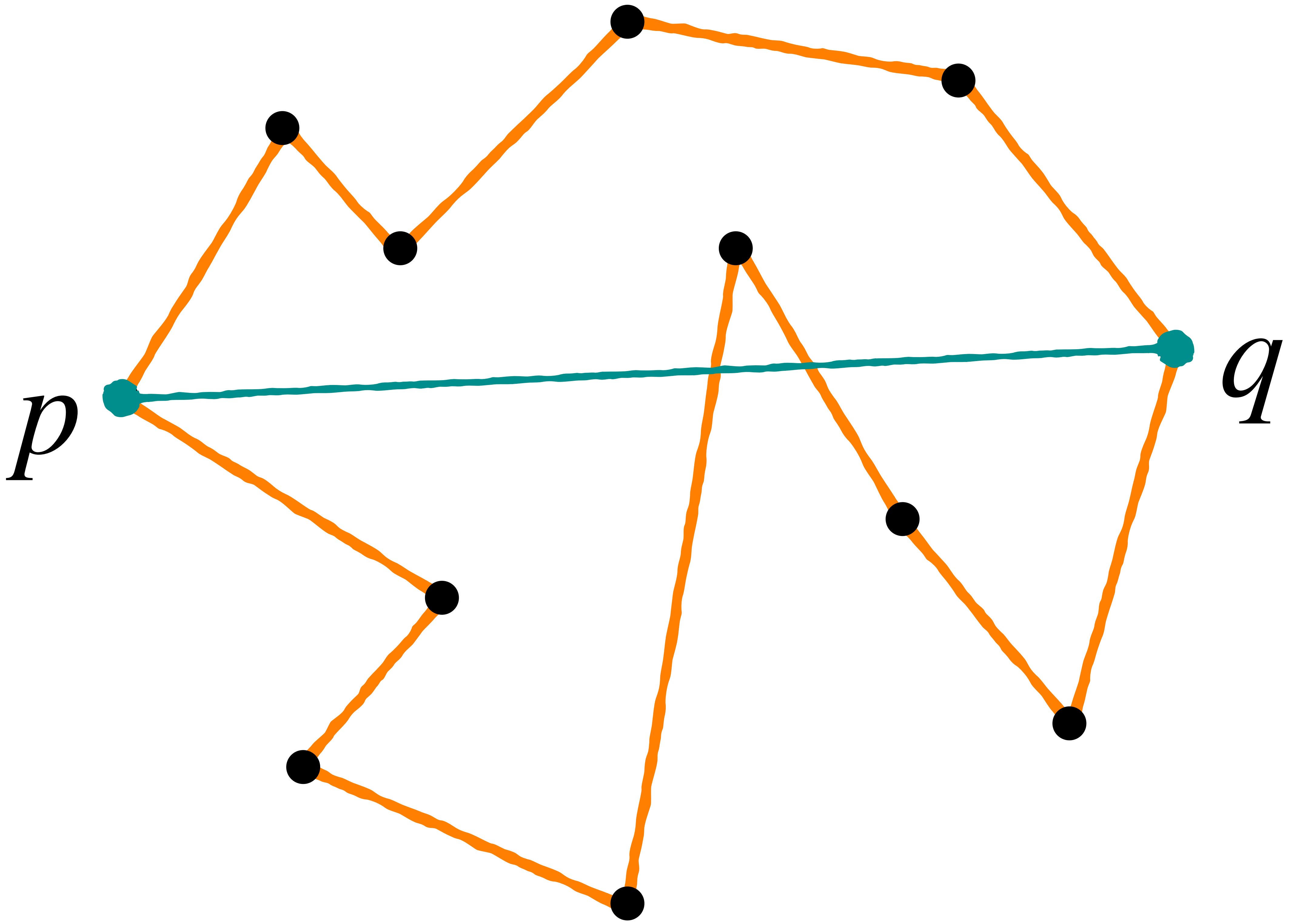
Farthest point pairs
Convex hull

Let $\mathscr{P}$ be set of n points in the Euclidean plane $\mathbb{R}^2$, in general position*.

Theorem E3.4 All farthest pairs and the diameter of $\mathscr{P}$ can be computed in $\mathcal{O}(n \log n)$.

Theorem E3.5 All farthest pairs and the diameter of an n-vertex simple polygon P can be computed in $\mathcal{O}(n)$.

Reason: Convex hull of P in $\mathcal{O}(n)$, not $\Omega(n \log n)$.



** no three points in $\mathscr{P}$ are collinear.*

Rotating Calipers Algorithm
Michael Shamos, 1978

Distances [edit]

- Diameter (maximum width) of a convex polygon[6][7]
- Width (minimum width) of a convex polygon[8]
- Maximum distance between two convex polygons[9][10]
- Minimum distance between two convex polygons[11][12]
- Widest empty (or separating) strip between two convex polygons (a simplified low-dimensional variant of a problem arising in support vector machine based machine learning)
- Grenander distance between two convex polygons[13]
- Optimal strip separation (used in medical imaging and solid modeling)[14]

Bounding boxes [edit]

- Minimum area oriented bounding box
- Minimum perimeter oriented bounding box

Triangulations [edit]

- Onion triangulations
- Spiral triangulations
- Quadrangulation
- Nice triangulation
- Art gallery problem
- Wedge placement optimization problem[15]

Multi-polygon operations [edit]

- Union of two convex polygons
- Common tangents to two convex polygons
- Intersection of two convex polygons[16]
- Critical support lines of two convex polygons
- Vector sums (or Minkowski sum) of two convex polygons[17]
- Convex hull of two convex polygons

Traversals [edit]

- Shortest transversals[18][19]
- Thinnest-strip transversals[20]

Others [edit]

- Non parametric decision rules for machine learned classification[21]
- Aperture angle optimizations for visibility problems in computer vision[22]
- Finding longest cells in millions of biological cells[23]
- Comparing precision of two people at firing range
- Classify sections of brain from scan images

Rotating Calipers Algorithm
Michael Shamos, 1978

Distances [edit]

- Diameter (maximum width) of a convex polygon[6][7]
- Width (minimum width) of a convex polygon[8]

- Diameter (maximum width) of a convex polygon[6][7]
- Maximum distance between two convex polygons[9][10]
- Minimum distance between two convex polygons[11][12]

Triangulations [edit]

- Onion triangulations
- Spiral triangulations
- Quadrangulation
- Nice triangulation
- Art gallery problem
- Wedge placement optimization problem[15]

Multi-polygon operations [edit]

- Union of two convex polygons
- Common tangents to two convex polygons
- Intersection of two convex polygons[16]
- Critical support lines of two convex polygons
- Vector sums (or Minkowski sum) of two convex polygons[17]
- Convex hull of two convex polygons

Traversals [edit]

- Shortest transversals[18][19]
- Thinnest-strip transversals[20]

Others [edit]

- Non parametric decision rules for machine learned classification[21]
- Aperture angle optimizations for visibility problems in computer vision[22]
- Finding longest cells in millions of biological cells[23]
- Comparing precision of two people at firing range
- Classify sections of brain from scan images

Homework Sheet #2

Peter Kramer | November 14th, 2024

Computational Geometry – Sheet 2 — **Winter 2024/2025**

Prof. Dr. Sándor P. Fekete — **Due** 28.11.2024

Peter Kramer — **Discussion** 05.12.2024

Please submit your handwritten answers in pairs, *using the box in front of* `IZ338` before *the exercise timeslot on the due date above. Make sure to include your* full names, matriculation numbers, *and the* programmes *that you are enrolled in. In accordance with the* `guidelines` *of the TU Braunschweig, using AI tools to solve any part of the exercises is* ***not permitted***.

Exercise 1 (Geometric Predicates). **(5 points)**

Using only the leftTurn and rightTurn predicates from Lecture 1, design a geometric predicate that decides whether a given line segment $\overline{pq}$ intersects a counterclockwise triangle $\triangle(u, v, w)$:

$$\operatorname{conv}(p, q) \cap \operatorname{conv}(u, v, w) = \varnothing\ ?$$

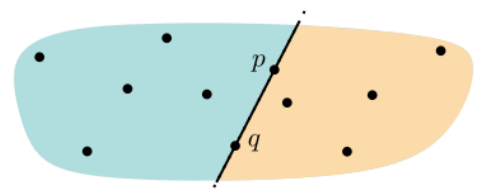
You may assume that each of the five points is unique and that no three points are collinear. Please explain your solution and briefly argue its correctness.

Exercise 2 (Partitioning Points). **(15 points)**

Consider a set $\mathcal{P}$ in the Euclidean plane $\mathbb{R}^2$ in general position according to *Definition E1*.

a) Prove that there exist points $p, q \in \mathcal{P}$ that divide $\mathcal{P}$ evenly based on left-/rightTurn:

$$|\{\, r \in \mathcal{P} \mid \operatorname{leftTurn}(p, q, r) = \texttt{true} \,\}| = |\mathcal{P}|/2 \pm 1.$$



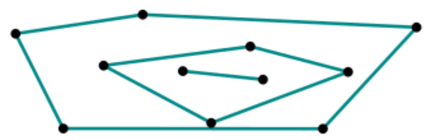
b) Design an algorithm that finds p and q in $\mathcal{O}(n)$ time for $n = |\mathcal{P}|$.

(Hint: Start with **b)**, a good correctness proof can also give you a constructive proof of existence.)

Exercise 3 (Convex layers). **(10 points)**

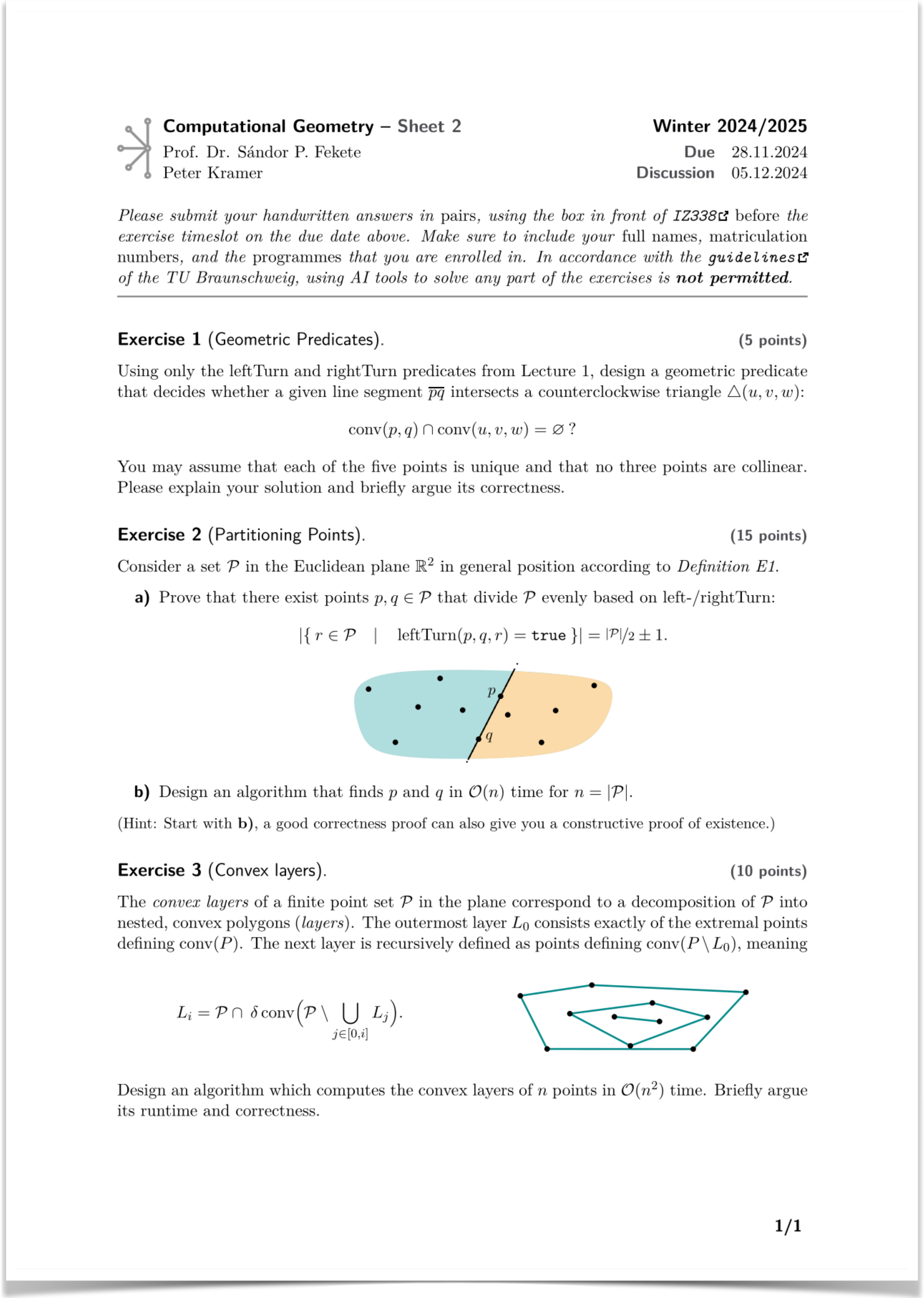
The *convex layers* of a finite point set $\mathcal{P}$ in the plane correspond to a decomposition of $\mathcal{P}$ into nested, convex polygons (*layers*). The outermost layer L_0 consists exactly of the extremal points defining $\operatorname{conv}(P)$. The next layer is recursively defined as points defining $\operatorname{conv}(P \setminus L_0)$, meaning

$$L_i = \mathcal{P} \cap \delta \operatorname{conv}\Big(\mathcal{P} \setminus \bigcup_{j \in [0,i]} L_j\Big).$$



Design an algorithm which computes the convex layers of n points in $\mathcal{O}(n^2)$ time. Briefly argue its runtime and correctness.

1/1



Computational Geometry – Sheet 2 **Winter 2024/2025**

Prof. Dr. Sándor P. Fekete **Due** 28.11.2024

Peter Kramer **Discussion** 05.12.2024

Please submit your handwritten answers in pairs, using the box in front of IZ338 *before the exercise timeslot on the due date above. Make sure to include your full names, matriculation numbers, and the programmes that you are enrolled in. In accordance with the* `guidelines` *of the TU Braunschweig, using AI tools to solve any part of the exercises is* ***not permitted****.*

Exercise 1 (Geometric Predicates). **(5 points)**

Using only the leftTurn and rightTurn predicates from Lecture 1, design a geometric predicate that decides whether a given line segment $\overline{pq}$ intersects a counterclockwise triangle $\triangle(u, v, w)$:

$$\operatorname{conv}(p, q) \cap \operatorname{conv}(u, v, w) = \varnothing\ ?$$

You may assume that each of the five points is unique and that no three points are collinear. Please explain your solution and briefly argue its correctness.

Exercise 2 (Partitioning Points). **(15 points)**

Consider a set $\mathcal{P}$ in the Euclidean plane $\mathbb{R}^2$ in general position according to *Definition E1*.

a) Prove that there exist points $p, q \in \mathcal{P}$ that divide $\mathcal{P}$ evenly based on left-/rightTurn:

$$|\{\, r \in \mathcal{P} \;\mid\; \text{leftTurn}(p, q, r) = \texttt{true} \,\}| = |\mathcal{P}|/2 \pm 1.$$

b) Design an algorithm that finds p and q in $\mathcal{O}(n)$ time for $n = |\mathcal{P}|$.

(Hint: Start with **b)**, a good correctness proof can also give you a constructive proof of existence.)

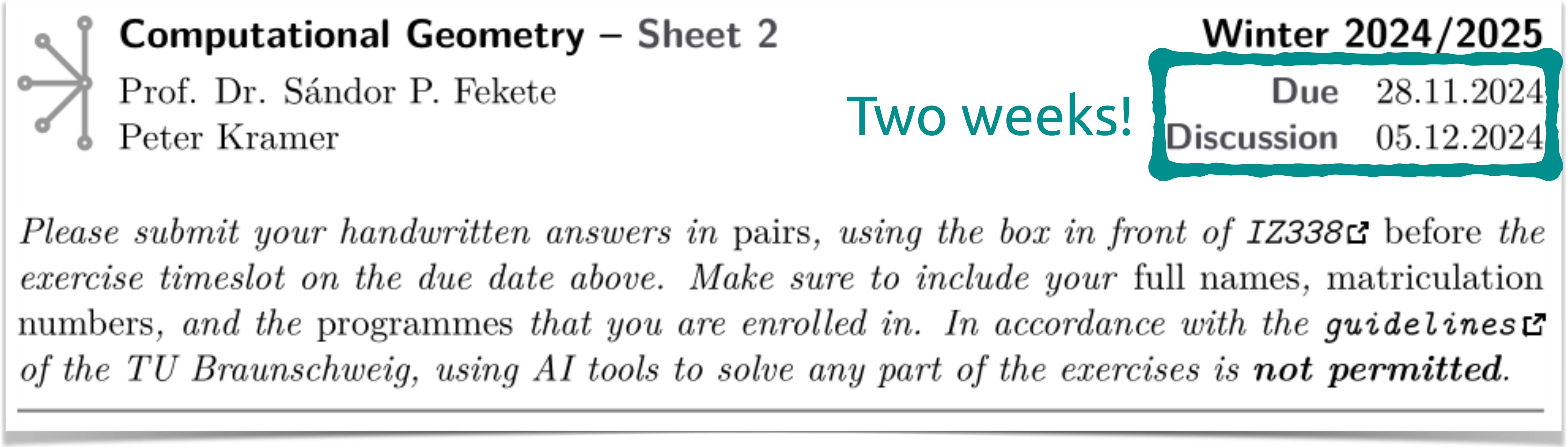
Exercise 3 (Convex layers). **(10 points)**

The *convex layers* of a finite point set $\mathcal{P}$ in the plane correspond to a decomposition of $\mathcal{P}$ into nested, convex polygons (*layers*). The outermost layer L_0 consists exactly of the extremal points defining $\operatorname{conv}(P)$. The next layer is recursively defined as points defining $\operatorname{conv}(P \setminus L_0)$, meaning

$$L_i = \mathcal{P} \cap \delta \operatorname{conv}\Big(\mathcal{P} \setminus \bigcup_{j \in [0,i]} L_j\Big).$$

Design an algorithm which computes the convex layers of n points in $\mathcal{O}(n^2)$ time. Briefly argue its runtime and correctness.

1/1



Computational Geometry – Sheet 2 **Winter 2024/2025**

Prof. Dr. Sándor P. Fekete **Due** 28.11.2024

Peter Kramer **Discussion** 05.12.2024

Please submit your handwritten answers in pairs, using the box in front of IZ338 *before the exercise timeslot on the due date above. Make sure to include your full names, matriculation numbers, and the programmes that you are enrolled in. In accordance with the* `guidelines` *of the TU Braunschweig, using AI tools to solve any part of the exercises is* ***not permitted****.*

- You may change homework partners at any time. Grading is tracked individually, not by group.
- A total of **70 points across all sheets** is sufficient for the coursework / Studienleistung.
- **So far:** 35 points, this sheet: 30 points

Computational Geometry – Sheet 2 **Winter 2024/2025**

Prof. Dr. Sándor P. Fekete **Due** 28.11.2024

Peter Kramer **Discussion** 05.12.2024

Please submit your handwritten answers in pairs, using the box in front of `IZ338` *before the exercise timeslot on the due date above. Make sure to include your full names, matriculation numbers, and the programmes that you are enrolled in. In accordance with the* `guidelines` *of the TU Braunschweig, using AI tools to solve any part of the exercises is* ***not permitted****.*

Exercise 1 (Geometric Predicates). **(5 points)**

Using only the leftTurn and rightTurn predicates from Lecture 1, design a geometric predicate that decides whether a given line segment $\overline{pq}$ intersects a counterclockwise triangle $\triangle(u, v, w)$:

$$\mathrm{conv}(p, q) \cap \mathrm{conv}(u, v, w) = \varnothing\ ?$$

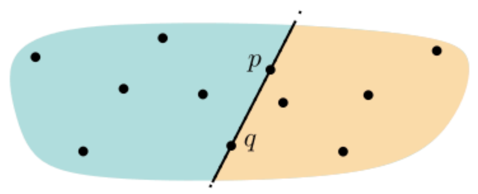
You may assume that each of the five points is unique and that no three points are collinear. Please explain your solution and briefly argue its correctness.

Exercise 2 (Partitioning Points). **(15 points)**

Consider a set $\mathcal{P}$ in the Euclidean plane $\mathbb{R}^2$ in general position according to *Definition E1*.

a) Prove that there exist points $p, q \in \mathcal{P}$ that divide $\mathcal{P}$ evenly based on left-/rightTurn:

$$|\{\, r \in \mathcal{P} \mid \mathrm{leftTurn}(p, q, r) = \texttt{true} \,\}| = |\mathcal{P}|/2 \pm 1.$$



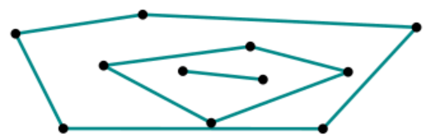
b) Design an algorithm that finds p and q in $\mathcal{O}(n)$ time for $n = |\mathcal{P}|$.

(Hint: Start with **b)**, a good correctness proof can also give you a constructive proof of existence.)

Exercise 3 (Convex layers). **(10 points)**

The *convex layers* of a finite point set $\mathcal{P}$ in the plane correspond to a decomposition of $\mathcal{P}$ into nested, convex polygons (*layers*). The outermost layer L_0 consists exactly of the extremal points defining $\mathrm{conv}(P)$. The next layer is recursively defined as points defining $\mathrm{conv}(P \setminus L_0)$, meaning

$$L_i = \mathcal{P} \cap \delta\,\mathrm{conv}\Big(\mathcal{P} \setminus \bigcup_{j \in [0,i]} L_j\Big).$$



Design an algorithm which computes the convex layers of n points in $\mathcal{O}(n^2)$ time. Briefly argue its runtime and correctness.

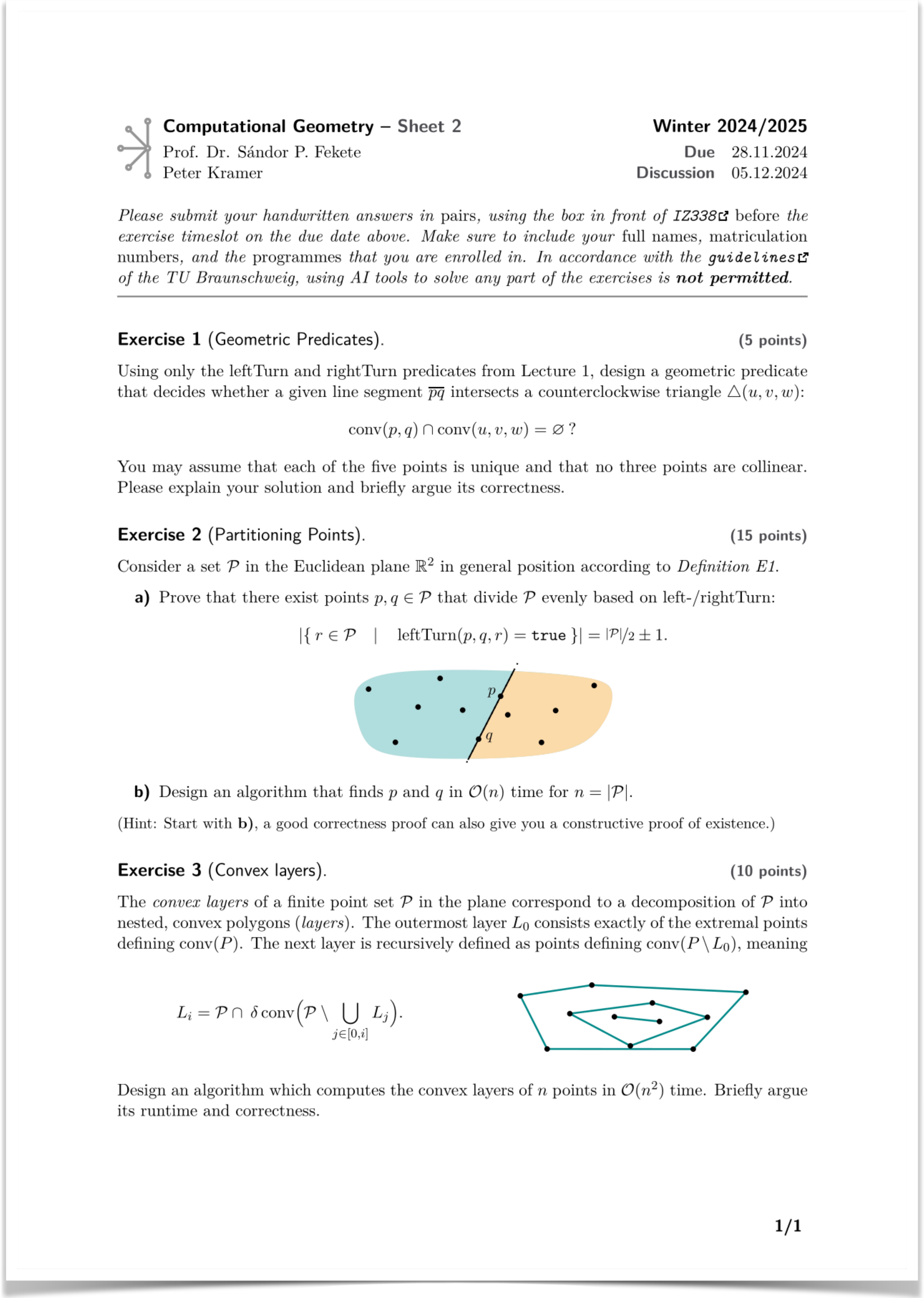
1/1

Exercise 1 (Geometric Predicates). **(5 points)**

Using only the leftTurn and rightTurn predicates from Lecture 1, design a geometric predicate that decides whether a given line segment $\overline{pq}$ intersects a counterclockwise triangle $\triangle(u, v, w)$:

$$\mathrm{conv}(p, q) \cap \mathrm{conv}(u, v, w) = \varnothing\ ?$$

You may assume that each of the five points is unique and that no three points are collinear. Please explain your solution and briefly argue its correctness.



Computational Geometry – Sheet 2 **Winter 2024/2025**

Prof. Dr. Sándor P. Fekete **Due** 28.11.2024

Peter Kramer **Discussion** 05.12.2024

Please submit your handwritten answers in pairs, *using the box in front of* `IZ338` before *the exercise timeslot on the due date above. Make sure to include your* full names, matriculation numbers, *and the* programmes *that you are enrolled in. In accordance with the* `guidelines` *of the TU Braunschweig, using AI tools to solve any part of the exercises is* ***not permitted***.

Exercise 1 (Geometric Predicates). **(5 points)**

Using only the leftTurn and rightTurn predicates from Lecture 1, design a geometric predicate that decides whether a given line segment $\overline{pq}$ intersects a counterclockwise triangle $\triangle(u, v, w)$:

$$\mathrm{conv}(p, q) \cap \mathrm{conv}(u, v, w) = \varnothing\ ?$$

You may assume that each of the five points is unique and that no three points are collinear. Please explain your solution and briefly argue its correctness.

Exercise 2 (Partitioning Points). **(15 points)**

Consider a set $\mathcal{P}$ in the Euclidean plane $\mathbb{R}^2$ in general position according to *Definition E1*.

a) Prove that there exist points $p, q \in \mathcal{P}$ that divide $\mathcal{P}$ evenly based on left-/rightTurn:

$$|\{\, r \in \mathcal{P} \mid \mathrm{leftTurn}(p, q, r) = \texttt{true} \,\}| = {}^{|\mathcal{P}|}/_{2} \pm 1.$$

b) Design an algorithm that finds p and q in $\mathcal{O}(n)$ time for $n = |\mathcal{P}|$.

(Hint: Start with **b)**, a good correctness proof can also give you a constructive proof of existence.)

Exercise 3 (Convex layers). **(10 points)**

The *convex layers* of a finite point set $\mathcal{P}$ in the plane correspond to a decomposition of $\mathcal{P}$ into nested, convex polygons (*layers*). The outermost layer L_0 consists exactly of the extremal points defining $\mathrm{conv}(P)$. The next layer is recursively defined as points defining $\mathrm{conv}(P \setminus L_0)$, meaning

$$L_i = \mathcal{P} \cap \delta\,\mathrm{conv}\Big(\mathcal{P} \setminus \bigcup_{j \in [0,i]} L_j\Big).$$

Design an algorithm which computes the convex layers of n points in $\mathcal{O}(n^2)$ time. Briefly argue its runtime and correctness.

1/1

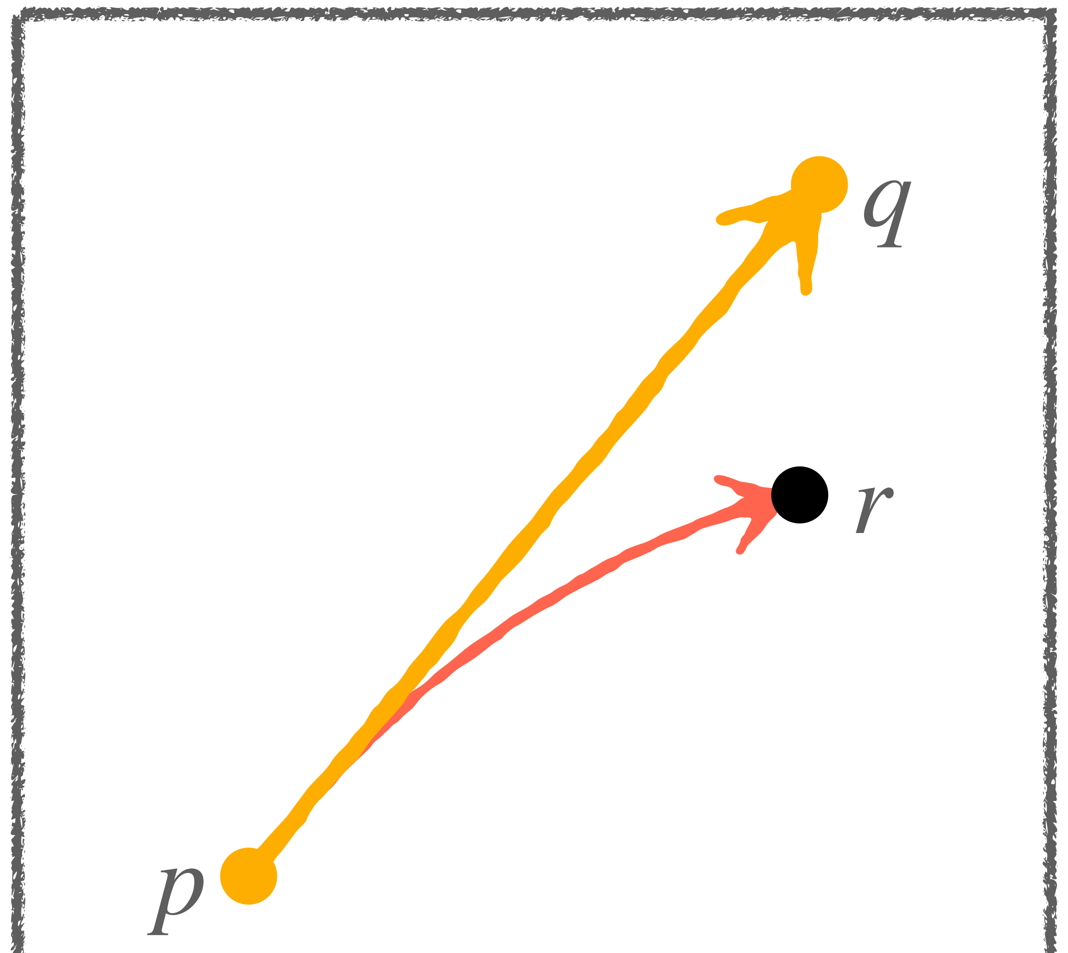
Exercise 1 (Geometric Predicates). **(5 points)**

Using only the leftTurn and rightTurn predicates from Lecture 1, design a geometric predicate that decides whether a given line segment $\overline{pq}$ intersects a counterclockwise triangle $\triangle(u, v, w)$:

$$\mathrm{conv}(p, q) \cap \mathrm{conv}(u, v, w) = \varnothing\ ?$$

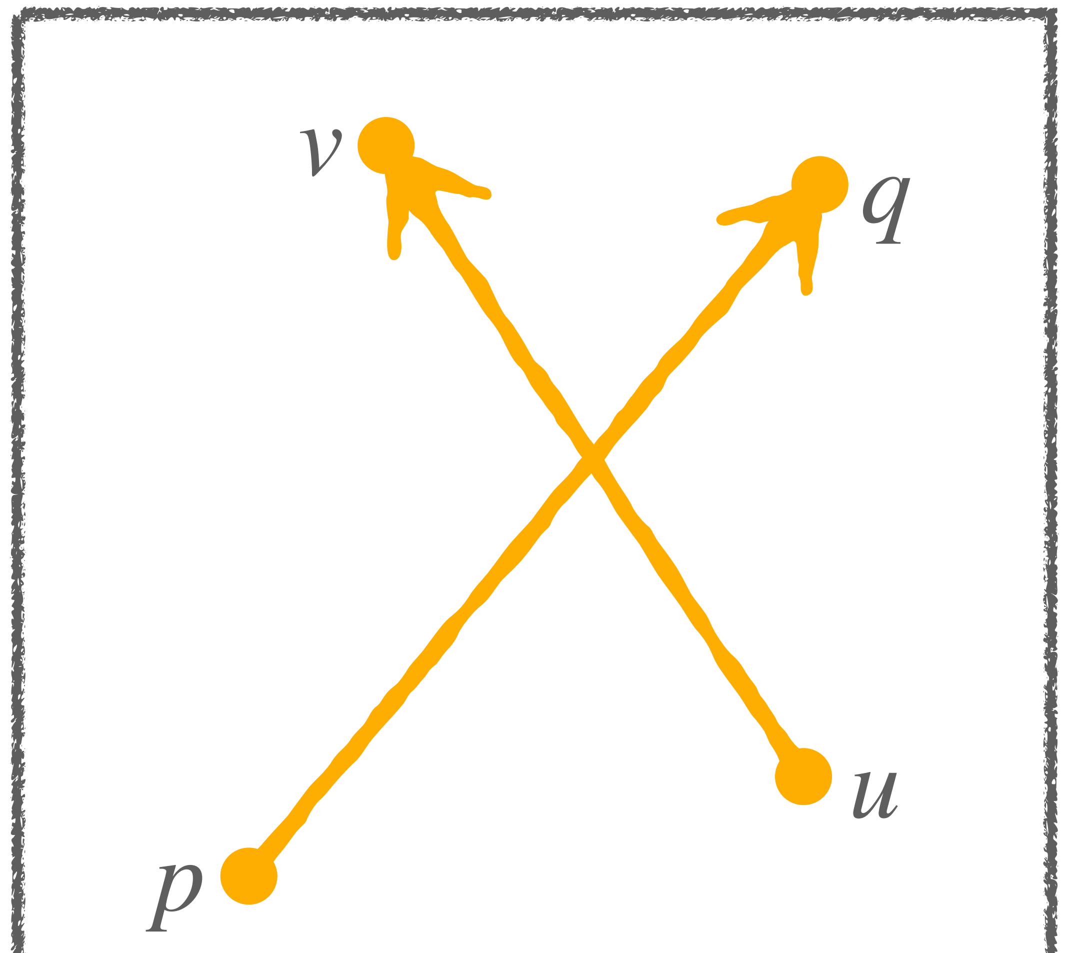
You may assume that each of the five points is unique and that no three points are collinear. Please explain your solution and briefly argue its correctness.

Point-Line Test



Is r collinear/left/right of $\overline{pq}$?

Intersection Test



(Where) do $\overline{pq}$ and $\overline{uv}$ intersect?

Computational Geometry – Sheet 2 **Winter 2024/2025**

Prof. Dr. Sándor P. Fekete **Due** 28.11.2024

Peter Kramer **Discussion** 05.12.2024

Please submit your handwritten answers in pairs, using the box in front of IZ338 before the exercise timeslot on the due date above. Make sure to include your full names, matriculation numbers, and the programmes that you are enrolled in. In accordance with the `guidelines` *of the TU Braunschweig, using AI tools to solve any part of the exercises is* ***not permitted****.*

Exercise 1 (Geometric Predicates). **(5 points)**

Using only the leftTurn and rightTurn predicates from Lecture 1, design a geometric predicate that decides whether a given line segment $\overline{pq}$ intersects a counterclockwise triangle $\triangle(u, v, w)$:

$$\mathrm{conv}(p, q) \cap \mathrm{conv}(u, v, w) = \varnothing\ ?$$

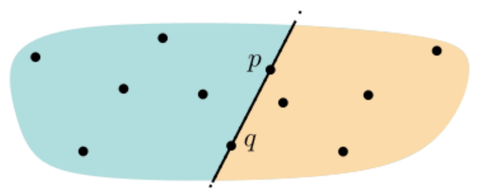
You may assume that each of the five points is unique and that no three points are collinear. Please explain your solution and briefly argue its correctness.

Exercise 2 (Partitioning Points). **(15 points)**

Consider a set $\mathcal{P}$ in the Euclidean plane $\mathbb{R}^2$ in general position according to *Definition E1.*

a) Prove that there exist points $p, q \in \mathcal{P}$ that divide $\mathcal{P}$ evenly based on left-/rightTurn:

$$|\{\ r \in \mathcal{P} \quad | \quad \mathrm{leftTurn}(p, q, r) = \texttt{true}\ \}| = {}^{|\mathcal{P}|}/_{2} \pm 1.$$



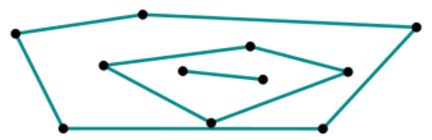
b) Design an algorithm that finds p and q in $\mathcal{O}(n)$ time for $n = |\mathcal{P}|$.

(Hint: Start with **b)**, a good correctness proof can also give you a constructive proof of existence.)

Exercise 3 (Convex layers). **(10 points)**

The *convex layers* of a finite point set $\mathcal{P}$ in the plane correspond to a decomposition of $\mathcal{P}$ into nested, convex polygons (*layers*). The outermost layer L_0 consists exactly of the extremal points defining $\mathrm{conv}(\mathcal{P})$. The next layer is recursively defined as points defining $\mathrm{conv}(\mathcal{P} \setminus L_0)$, meaning

$$L_i = \mathcal{P} \cap \delta\,\mathrm{conv}\Big(\mathcal{P} \setminus \bigcup_{j \in [0,i]} L_j\Big).$$



Design an algorithm which computes the convex layers of n points in $\mathcal{O}(n^2)$ time. Briefly argue its runtime and correctness.

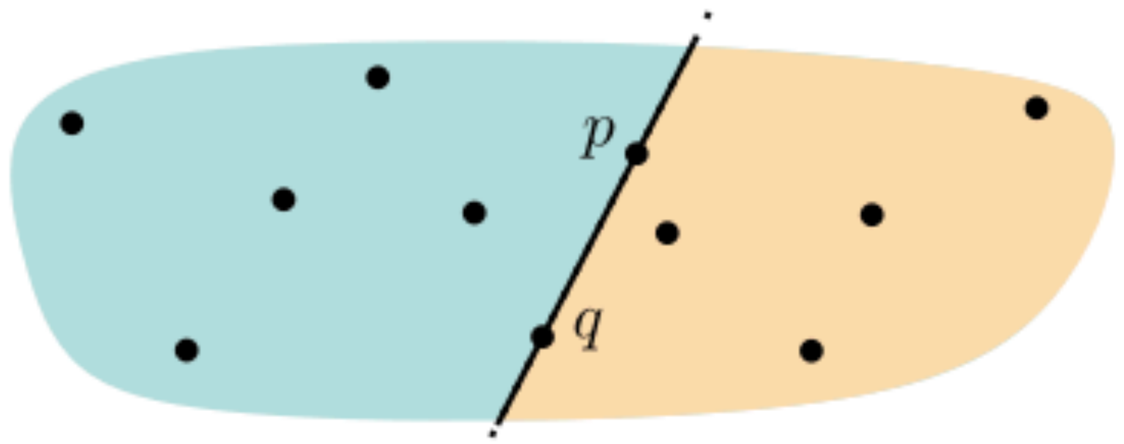
1/1

Exercise 2 (Partitioning Points). **(15 points)**

Consider a set $\mathcal{P}$ in the Euclidean plane $\mathbb{R}^2$ in general position according to *Definition E1.*

a) Prove that there exist points $p, q \in \mathcal{P}$ that divide $\mathcal{P}$ evenly based on left-/rightTurn:

$$|\{\ r \in \mathcal{P} \quad | \quad \mathrm{leftTurn}(p, q, r) = \texttt{true}\ \}| = {}^{|\mathcal{P}|}/_{2} \pm 1.$$



b) Design an algorithm that finds p and q in $\mathcal{O}(n)$ time for $n = |\mathcal{P}|$.

(Hint: Start with **b)**, a good correctness proof can also give you a constructive proof of existence.)

Computational Geometry – Sheet 2 **Winter 2024/2025**

Prof. Dr. Sándor P. Fekete **Due** 28.11.2024

Peter Kramer **Discussion** 05.12.2024

Please submit your handwritten answers in pairs, using the box in front of `IZ338` *before the exercise timeslot on the due date above. Make sure to include your full names, matriculation numbers, and the programmes that you are enrolled in. In accordance with the* `guidelines` *of the TU Braunschweig, using AI tools to solve any part of the exercises is* ***not permitted****.*

Exercise 1 (Geometric Predicates). **(5 points)**

Using only the leftTurn and rightTurn predicates from Lecture 1, design a geometric predicate that decides whether a given line segment $\overline{pq}$ intersects a counterclockwise triangle $\triangle(u, v, w)$:

$$\mathrm{conv}(p, q) \cap \mathrm{conv}(u, v, w) = \varnothing\ ?$$

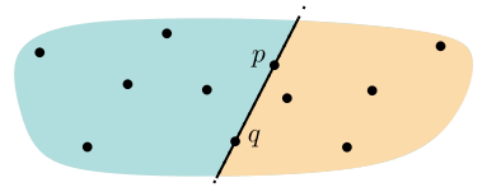
You may assume that each of the five points is unique and that no three points are collinear. Please explain your solution and briefly argue its correctness.

Exercise 2 (Partitioning Points). **(15 points)**

Consider a set $\mathcal{P}$ in the Euclidean plane $\mathbb{R}^2$ in general position according to *Definition E1.*

a) Prove that there exist points $p, q \in \mathcal{P}$ that divide $\mathcal{P}$ evenly based on left-/rightTurn:

$$|\{\, r \in \mathcal{P} \;\;|\;\; \mathrm{leftTurn}(p, q, r) = \texttt{true} \,\}| = \lfloor |\mathcal{P}|/2 \rfloor \pm 1.$$



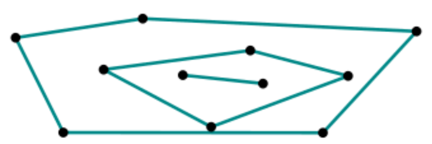
b) Design an algorithm that finds p and q in $\mathcal{O}(n)$ time for $n = |\mathcal{P}|$.

(Hint: Start with **b)**, a good correctness proof can also give you a constructive proof of existence.)

Exercise 3 (Convex layers). **(10 points)**

The *convex layers* of a finite point set $\mathcal{P}$ in the plane correspond to a decomposition of $\mathcal{P}$ into nested, convex polygons (*layers*). The outermost layer L_0 consists exactly of the extremal points defining $\mathrm{conv}(P)$. The next layer is recursively defined as points defining $\mathrm{conv}(P \setminus L_0)$, meaning

$$L_i = \mathcal{P} \cap \delta\,\mathrm{conv}\Big(\mathcal{P} \setminus \bigcup_{j \in [0,i]} L_j\Big).$$



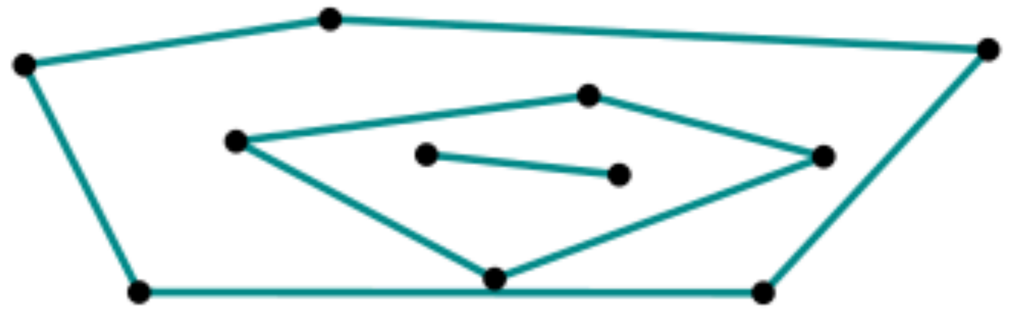
Design an algorithm which computes the convex layers of n points in $\mathcal{O}(n^2)$ time. Briefly argue its runtime and correctness.

1/1

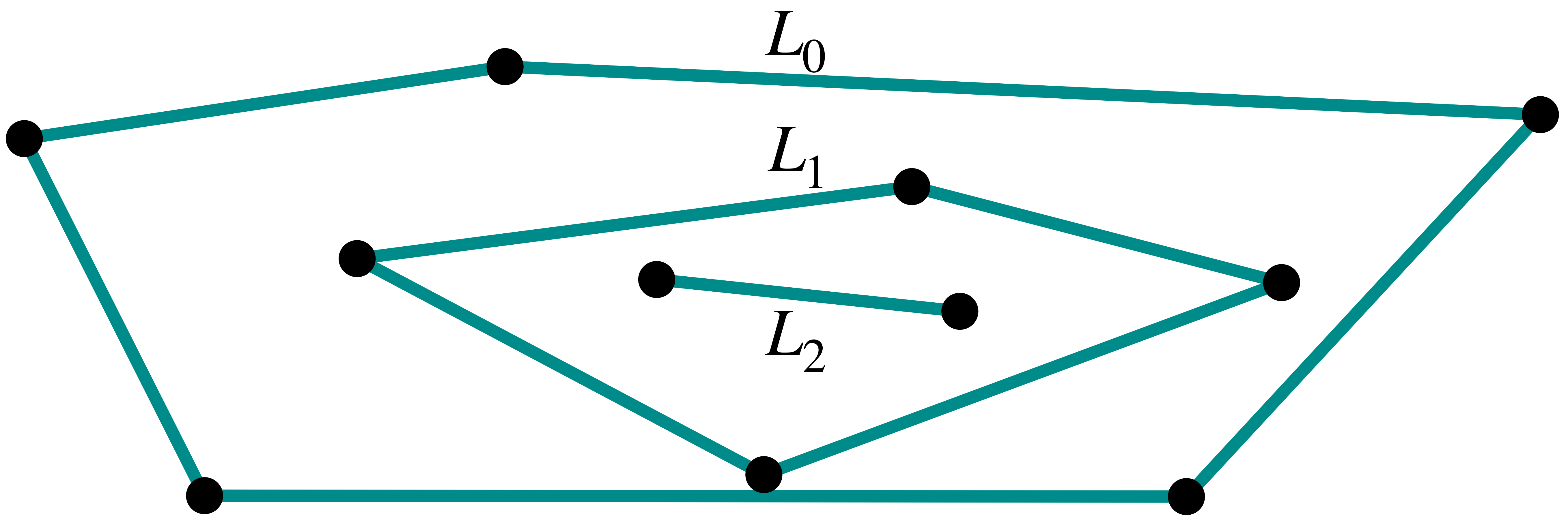
Exercise 3 (Convex layers). **(10 points)**

The *convex layers* of a finite point set $\mathcal{P}$ in the plane correspond to a decomposition of $\mathcal{P}$ into nested, convex polygons (*layers*). The outermost layer L_0 consists exactly of the extremal points defining $\mathrm{conv}(P)$. The next layer is recursively defined as points defining $\mathrm{conv}(P \setminus L_0)$, meaning

$$L_i = \mathcal{P} \cap \delta\,\mathrm{conv}\Big(\mathcal{P} \setminus \bigcup_{j \in [0,i]} L_j\Big).$$



Design an algorithm which computes the convex layers of n points in $\mathcal{O}(n^2)$ time. Briefly argue its runtime and correctness.



Thank you.