Triangulation and Voronoi Regions

CS535

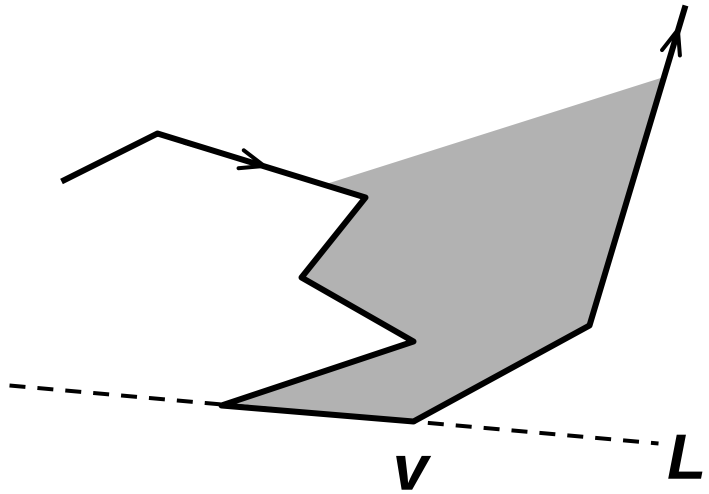
Daniel G. Aliaga
Department of Computer Science
Purdue University

[some slides based on Profs. Shmuel Wimer and Andy Mirzaian

Triangulation Theory

Lemma: *Every polygon must have at least one strictly convex vertex.*

Proof: Let the vertices be counterclockwise ordered. Traversing the boundary, a convex vertex corresponds to a left turn.



Pick the lowest vertex (pick the rightmost if there are a few).

L is a line passing through *v*.

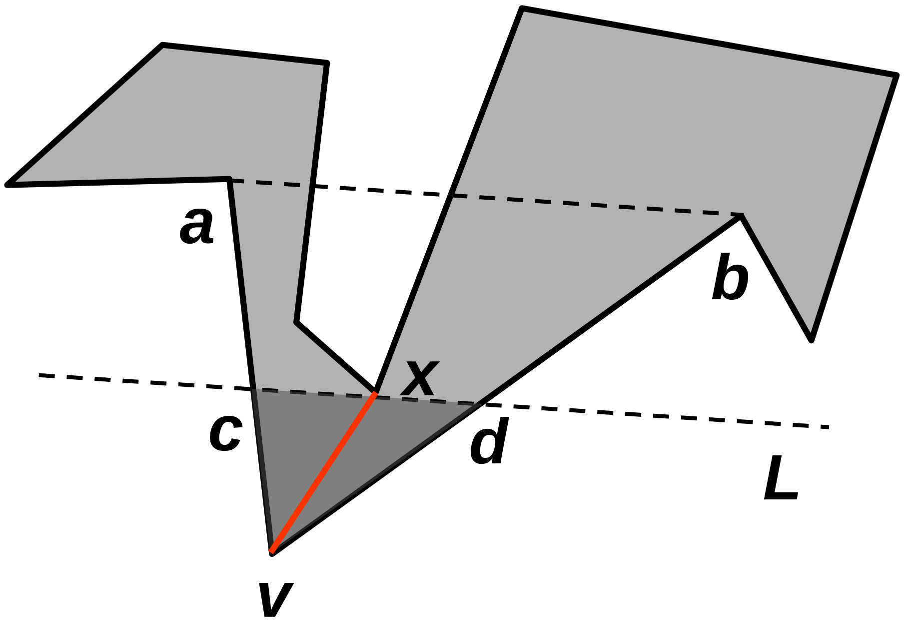
The edge following *v* must lie above *L*. ■

Triangulation Theory

Lemma: *Every polygon of $n \geq 4$ vertices has a diagonal.*

Proof: There exists a strictly convex vertex *v*. Let *a* and *b* be vertices adjacent to *v*. If *[a,b]* is a diagonal we are done.

Else...



Δavb must contain at least one vertex of P.

Let x be the closest vertex to v, measured orthogonal to the line passing through ab.

The interior of Δcvd cannot contain any point of ∂P.

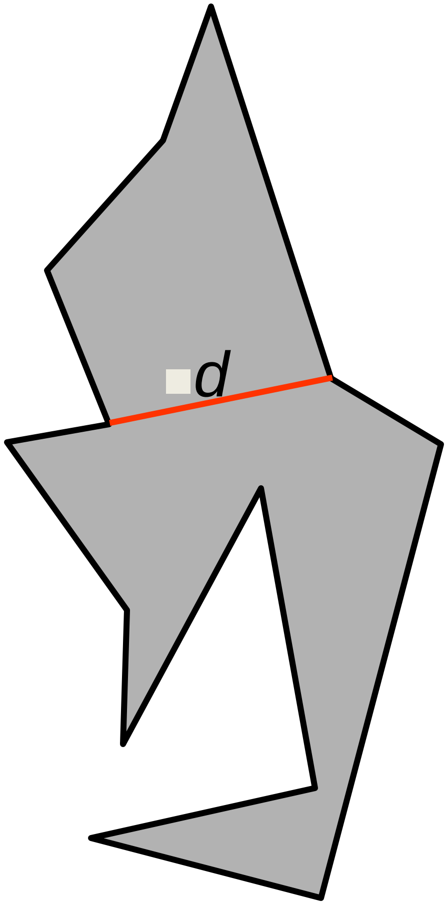
Therefore $[x, v] \cap P = \{x, v\}$, hence a diagonal. ■

Theorem: *Every n-vertex polygon P can be triangulated.*

Proof: By induction on n.

If $n=3$ P is a triangle.

Let $n \geq 4$. By lemma, P has a diagonal d which divides P into two polygons P_1 and P_2, having $n_1 < n$ and $n_2 < n$ vertices, respectively. P_1 and P_2 can be triangulated by induction hypothesis. ■



Triangulation Theory

Interesting Question: Do all triangulations of a given polygon have the same number of diagonals and triangles?

Answer: *Every triangulation of an n-vertex polygon P has n-3 diagonals and n-2 triangles.*

Triangulation Complexity

Implementing triangulation as in the existence theorem requires $O\left(n^4\right)$ time.

There are $n(n-3)/2$ diagonal candidates.

Cheking validity of a diagonal requires intersection test against all edges and previously defined diagonals, which takes $O(n)$ time.

This is repeated $n-3$ times, yielding total $O\left(n^4\right)$ time.

Triangulation Complexity

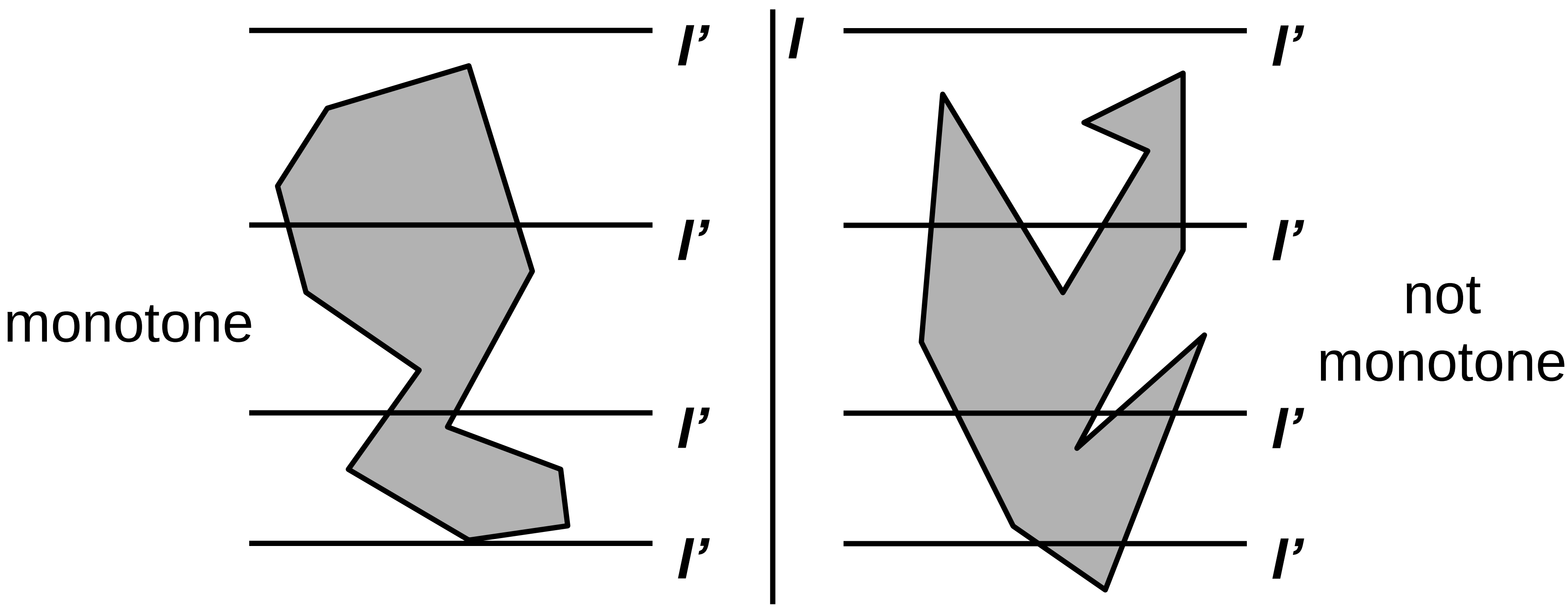
- Naïve: $O(n^4)$
  - There are $n(n-3)/2$ diagonal candidates
  - Checking the validity of a diagonal against all previous edges and diagonals takes $O(n)$
  - This is repeated $n-3$ times
  - Total time $O(n^4)$

Triangulation Complexity

- Lennes, 1911: $O(n^2)$
  - Pick the leftmost vertex v of P and connect its two neighbors u and w. Checking whether uw is a diagonal takes O(n). If it is, the rest is a (n-1)-vertex polygon.
  - If uw is not a diagonal, get x, the farthest vertex from uw inside Δuvw. This takes O(n) time. vx is a diagonal dividing P into P1 and P2, having n total number of vertices.
  - Recursive application of the above procedure consumes total $O(n^2)$ time.

Triangulation Complexity

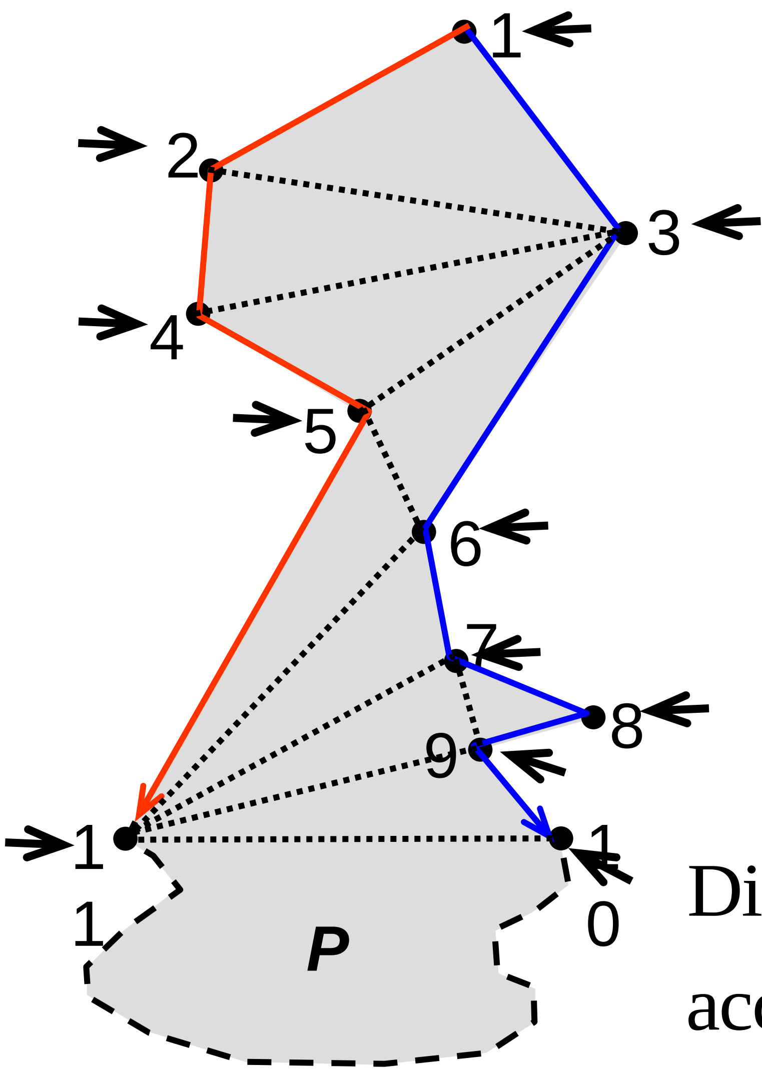
Definition: *P is monotone w.r.t to a line l if P intersects with any line l' perpendicular to l in a single segment, a point or it doesn't intersect.*



Triangulation Complexity

- An n-vertex simple polygon can be partitioned into y-monotone polygons in O(nlogn) time and O(n) storage
- Monotone polygon can be triangulated in O(n)

Triangulating y-Monotone Polygon (Sketch)



P's vertices are sorted in descending *y*.

P's boundary is traversed with one leg on left and one leg on right, by poping and pushing vertices from a stack.

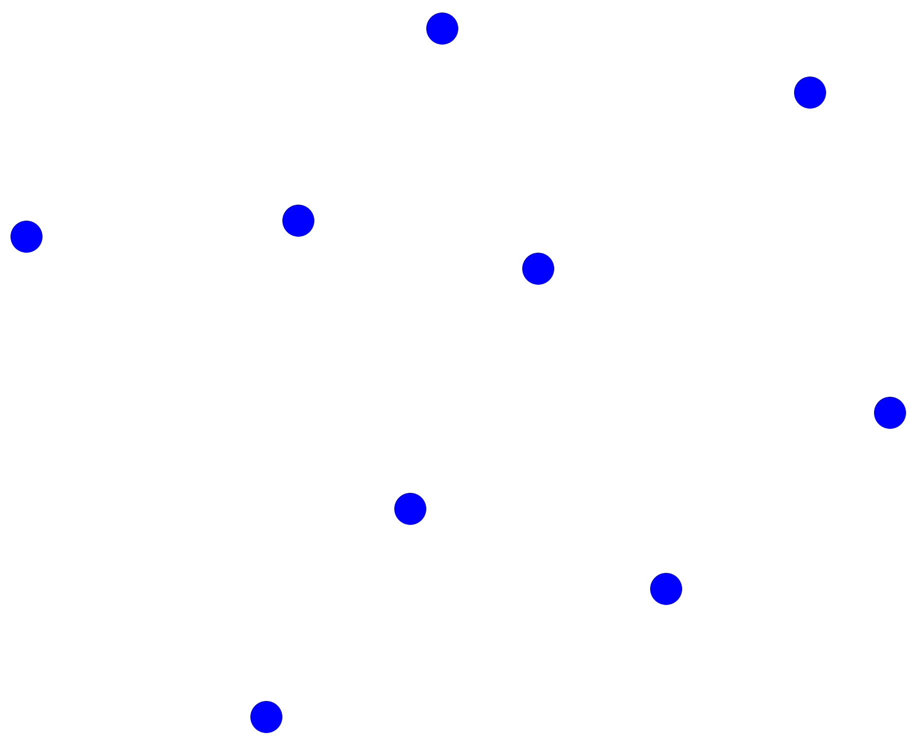
Diagonals insertions are decided according to stack's top status.

Triangulation Complexity

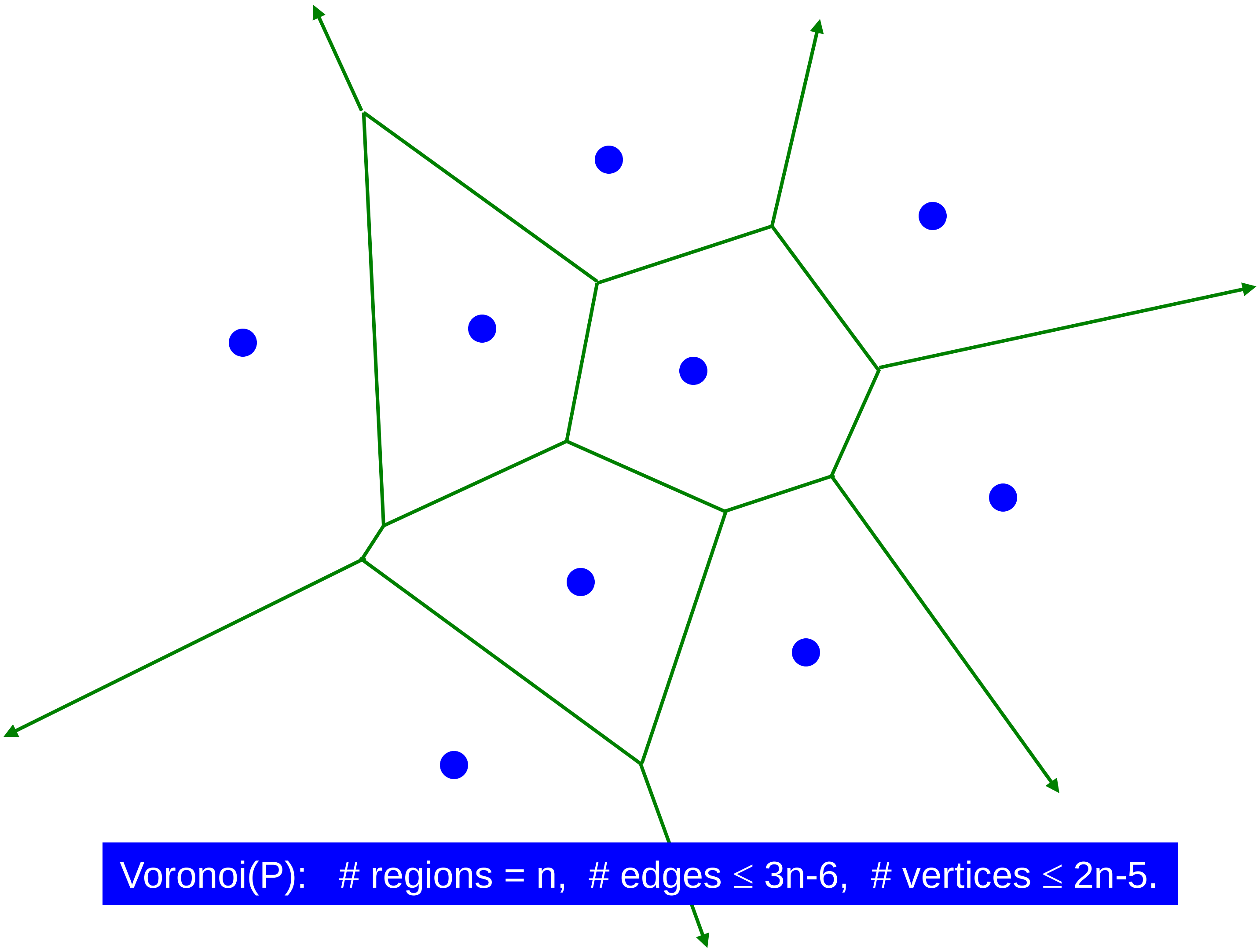
- Theorem: (Gary et. al. 1978) A simple n-vertex polygon can be triangulated in O(nlogn) time and O(n) storage
- The problem has been studied extensively between 1978 and 1991, when in 1991 Chazelle presented an <u>O(n) time complexity algorithm.</u>

Voronoi Diagram

- $P = \{ p_1, p_2, \ldots , p_n \}$ a set of n points in the plane.

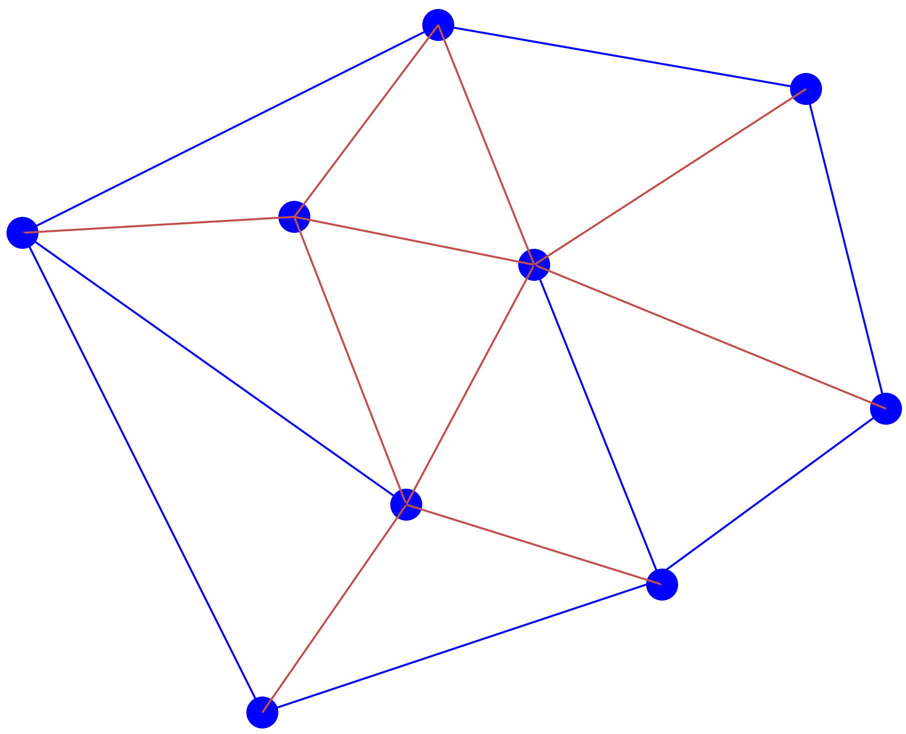


Voronoi Diagram:



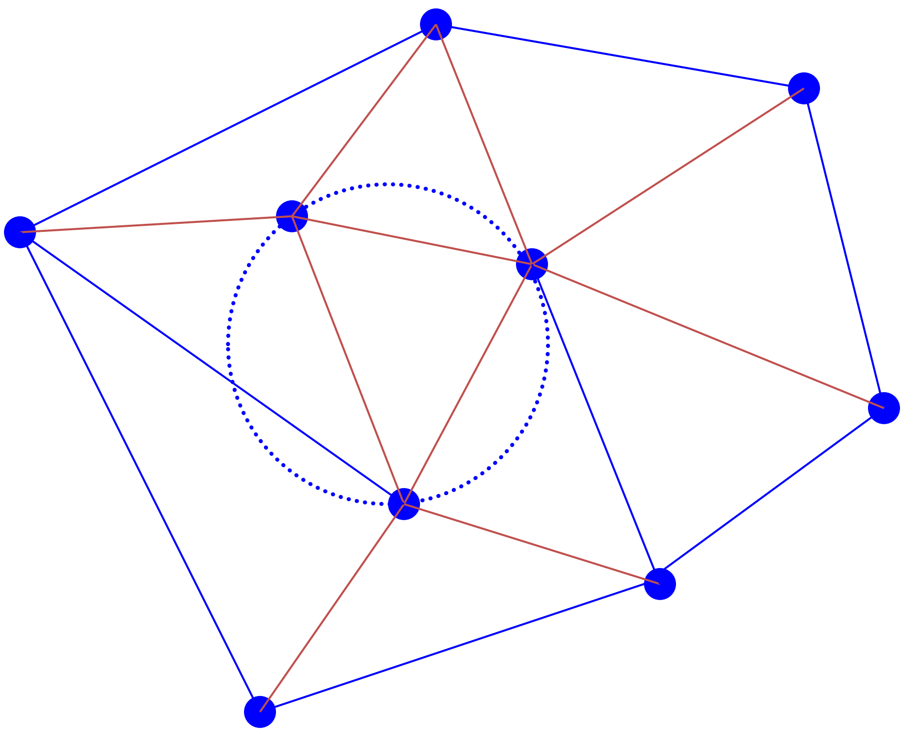
Voronoi(P): # regions = n, # edges ≤ 3n-6, # vertices ≤ 2n-5.

Delaunay Triangulation = Dual of the Voronoi Diagram



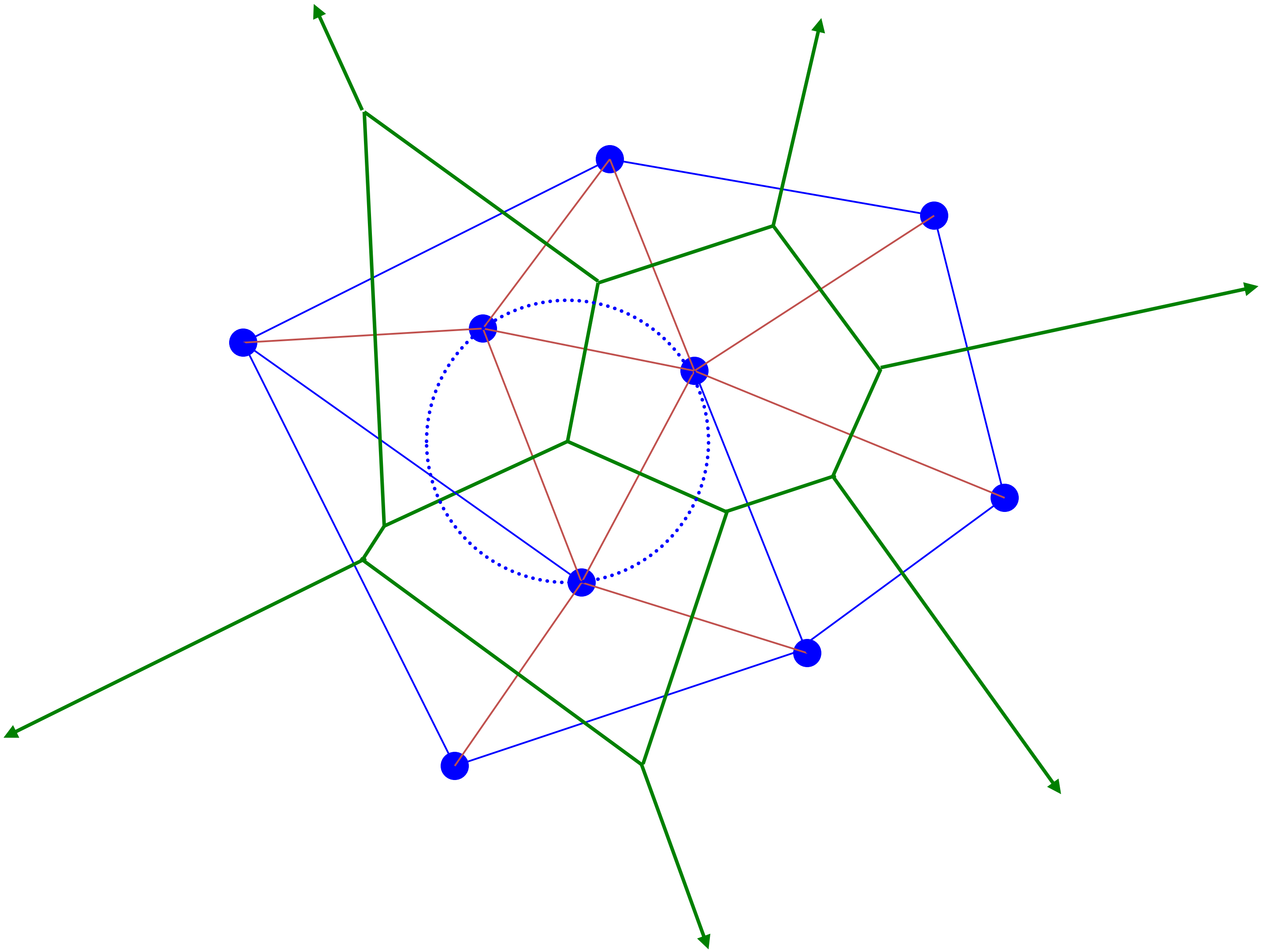
DT(P): # vertices = n, # edges ≤ 3n-6, # triangles ≤ 2n-5.

Delaunay Triangulation



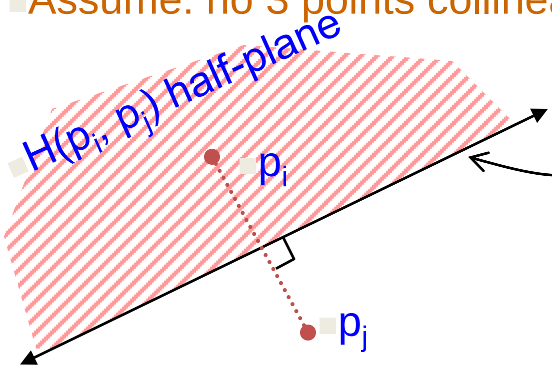
Delaunay triangles have the "empty circle" property.

Voronoi Diagram & Delaunay Triangulation

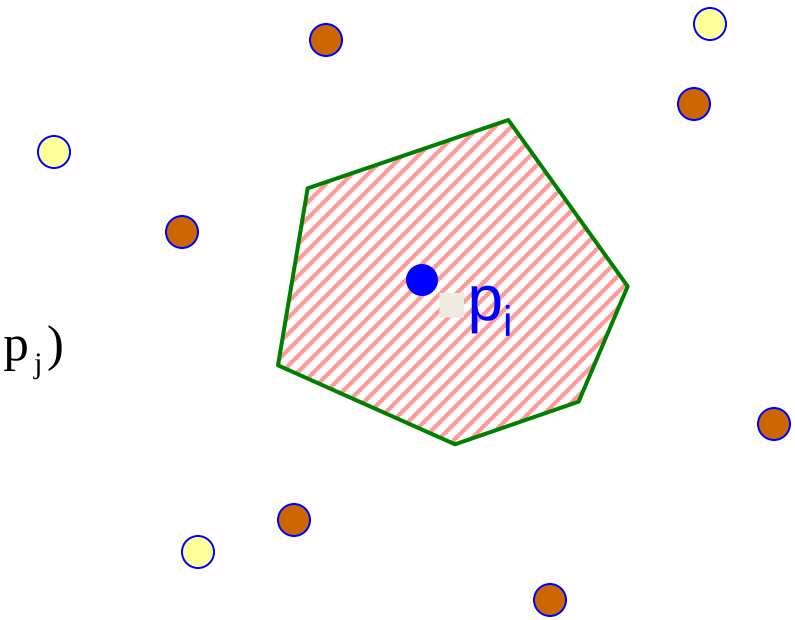


Voronoi Diagram

- $P = \{ p_1, p_2, \ldots , p_n\}$ a set of n points in the plane.
- Assume: no 3 points collinear, no 4 points cocircular.



- $PB(p_i, p_j)$ perpendicular bisector of $\overline{p_i p_j}$.



- Voronoi Region of p_i:

$$V(p_i) = \bigcap_{\substack{j=1 \\ j \neq i}}^{n} H(p_i, p_j)$$

- Voronoi Diagram of P:

$$VD(P) = \bigcup_{i=1}^{n} \{ V(p_i) \}$$

VD Properties

- Each Voronoi region $V(p_i)$ is a convex polygon (possibly unbounded).
- $V(p_i)$ is unbounded $\Leftrightarrow$ p_i is on the boundary of $CH(P)$.
- Consider a Voronoi vertex $v = V(p_i) \cap V(p_j) \cap V(p_k)$.
  Let $C(v)$ = the circle centered at v passing through p_i, p_j, p_k.
- $C(v)$ is circumcircle of Delaunay Triangle (p_i, p_j, p_k).
- $C(v)$ is an empty circle, i.e., its interior contains no other sites of P.
- p_j = a nearest neighbor of p_i $\Rightarrow$ $V(p_i) \cap V(p_j)$ is a Voronoi edge
  $\Rightarrow$ (p_i, p_j) is a Delaunay edge.

DT Properties

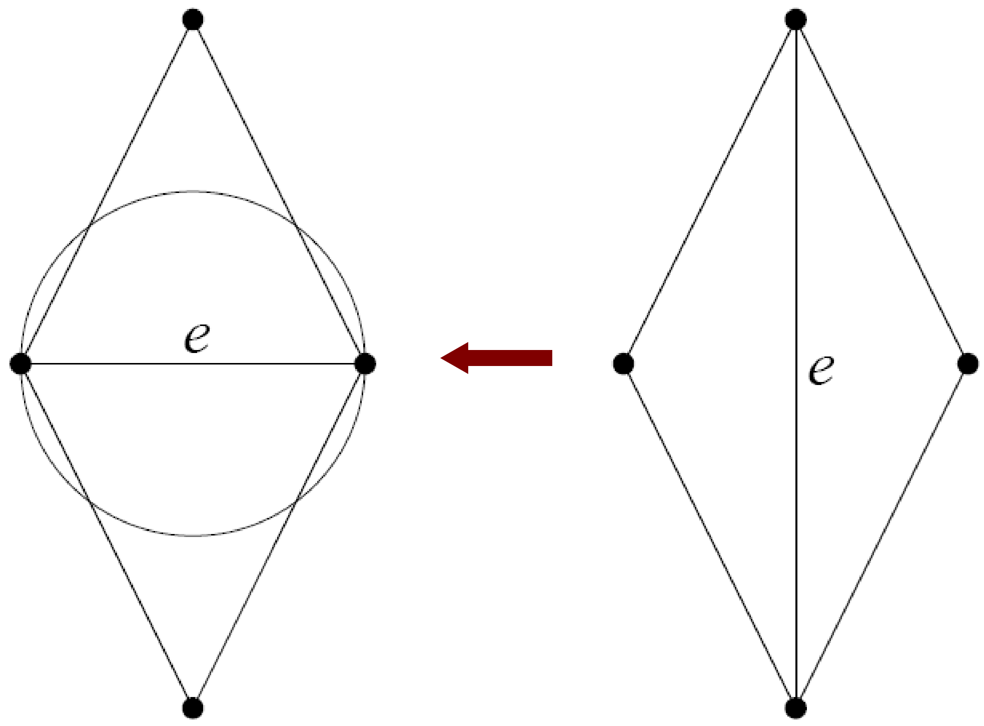
- DT(P) is straight-line dual of VD(P).
- DT(P) is a triangulation of P, i.e., each bounded face is a triangle (if P is in general position).
- (p_i, p_j) is a Delaunay edge $\Leftrightarrow \exists$ an empty circle passing through p_i and p_j.
- Each triangular face of DT(P) is dual of a Voronoi vertex of VD(P).
- Each edge of DT(P) corresponds to an edge of VD(P).
- Each node of DT(P), a site, corresponds to a Voronoi region of VD(P).
- Boundary of DT(P) is CH(P).
- Interior of each triangle in DT(P) is empty, i.e., contains no point of P.

Computing Delaunay Triangulation

- Many algorithms: O(nlogn)

- Lets use flipping:
  - Recall: A *Delaunay Triangulation* is a set of triangles T in which each edge of T possesses at least one empty circumcircle.
  - Empty: A circumcircle is said to be empty if it contains no nodes of the set V

What is a flip?



A non-Delaunay edge flipped

Flip Algorithm

- ??

Flip Algorithm

1. Let V be the set of input vertices.
2. T = Any Triangulation of V.
3. Repeat until all edges of T are Delaunay edges.
    a. Find a non-delaunay edge that is flippable
    b. Flip

Naïve Complexity: $O(n^2)$

Locally Delaunay → Globally Delaunay

- If T is a triangulation with all its edges locally Delaunay , then T is the Delaunay triangulation.
- Proof by contradiction:
  - Let all edges of T be locally Delaunay but an edge of T is not Delaunay, so flip it…

Flipping

- Other flipping ideas?

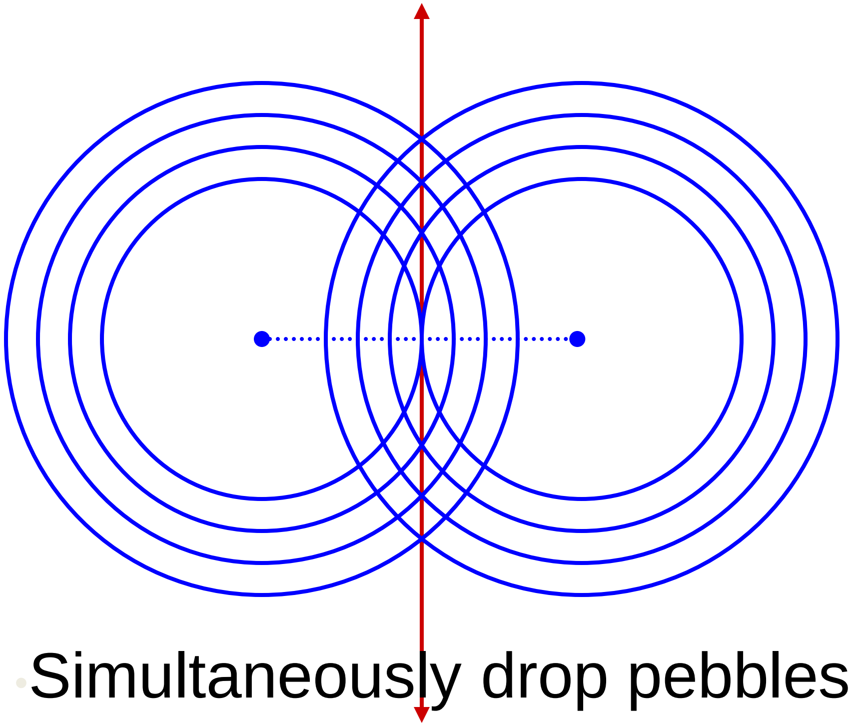
Randomized Incremental Flipping

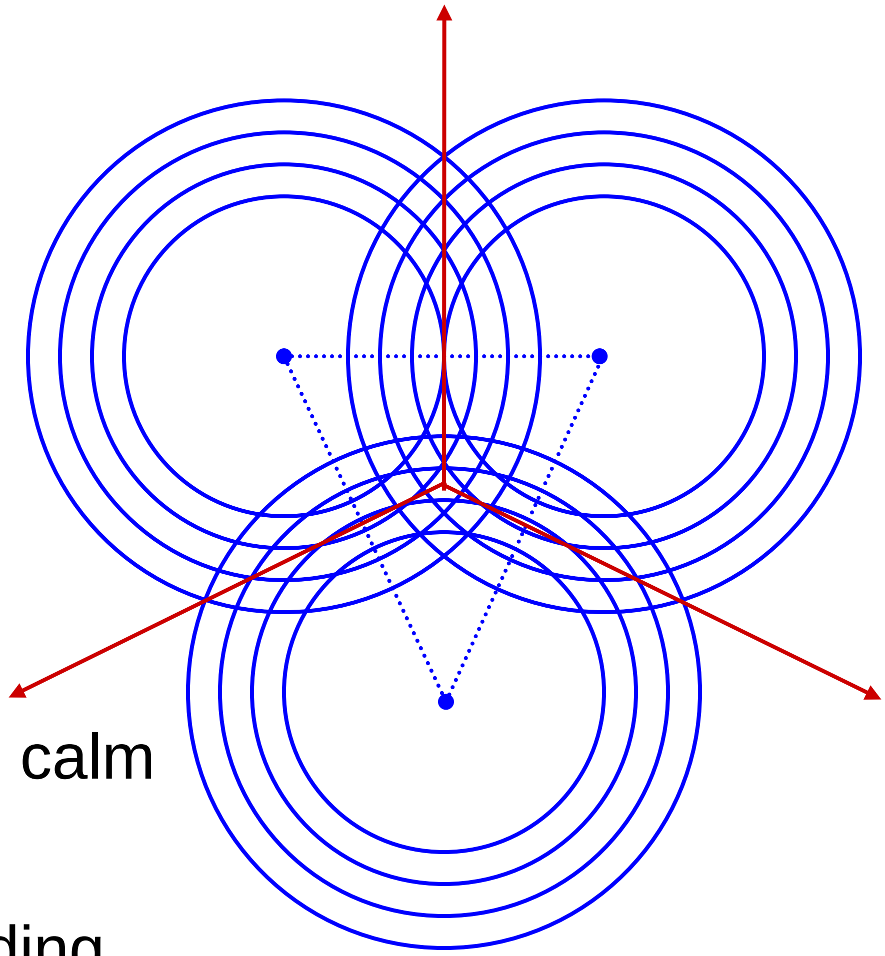
- Complexity can be O(nlogn)

Popular Method

- Fortune’s Algorithm
  - “A sweepline algorithm for Voronoi “Algorithms”, 1987, O(nlogn)

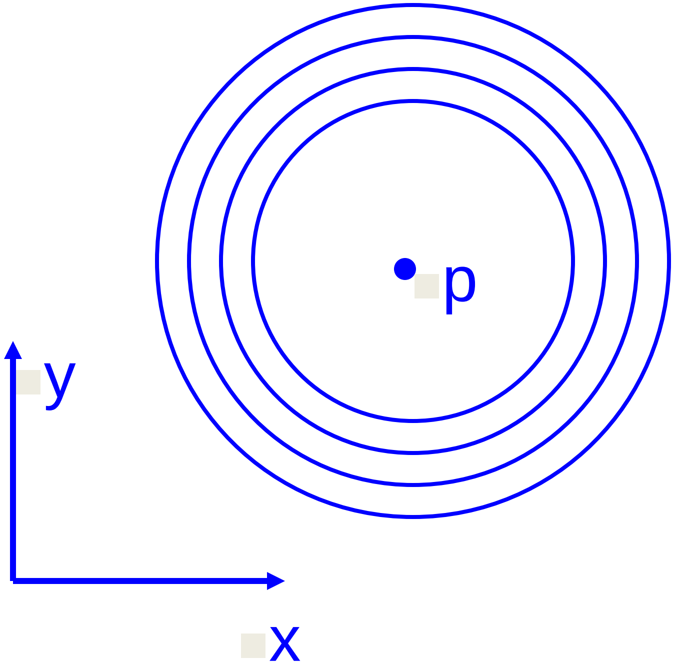
The Wave Propagation View

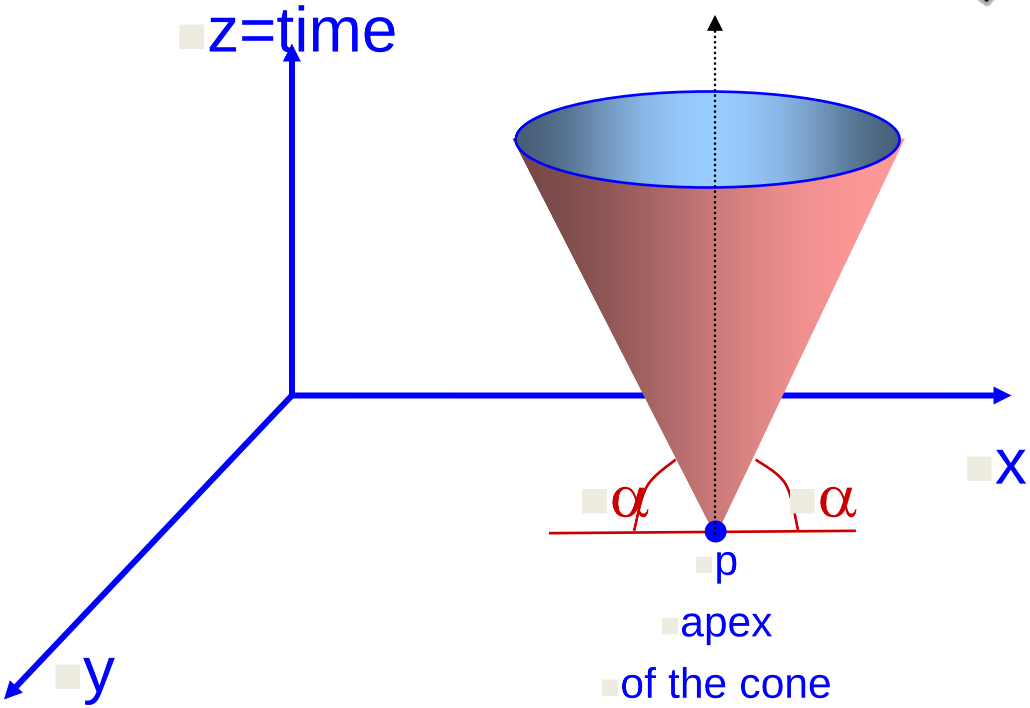




- Simultaneously drop pebbles on calm lake at n sites.
- Watch the intersection of expanding waves.

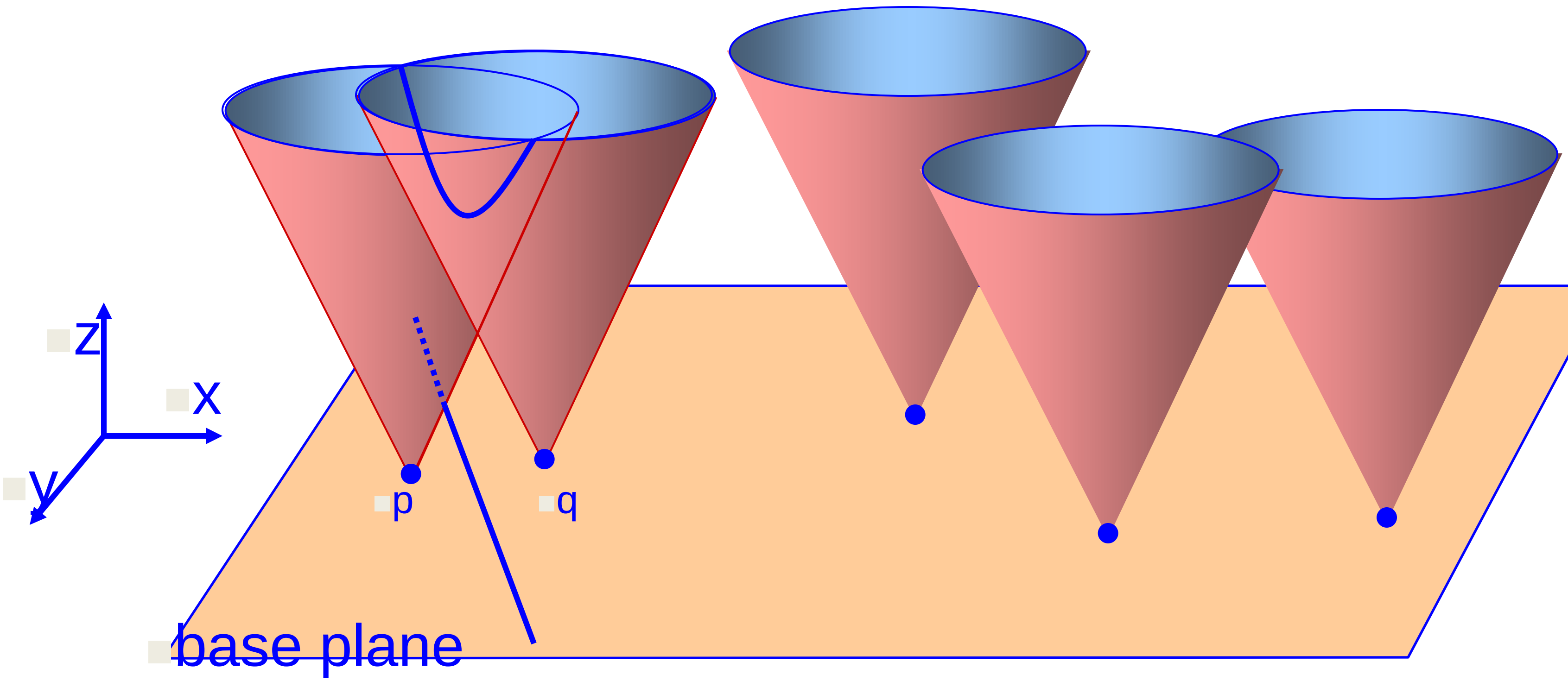
Time as 3rd dimension





- All sites have identical opaque cones

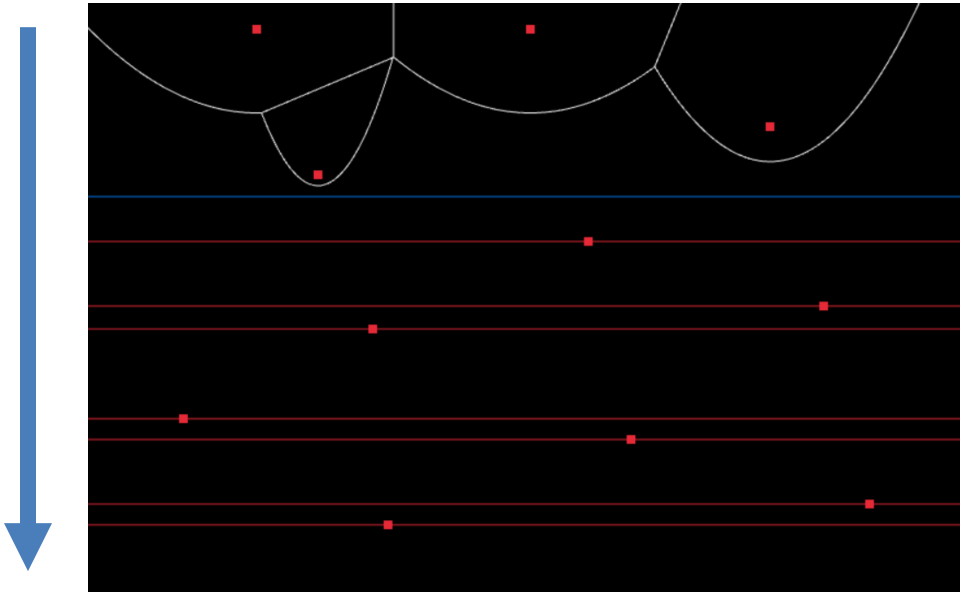
Time as 3rd dimension



- All sites have identical opaque cones.
- cone(p) $\cap$ cone(q) = vertical hyperbola h(p,q).
- Vertical projection of h(p,q) on the xy base plane is PB(p,q).

Fortune’s Algorithm

- A sweep-line approach:
  - Visit sites in order and grow the cells as we sweep
  - Maintains a beach-line
  - Is event driven

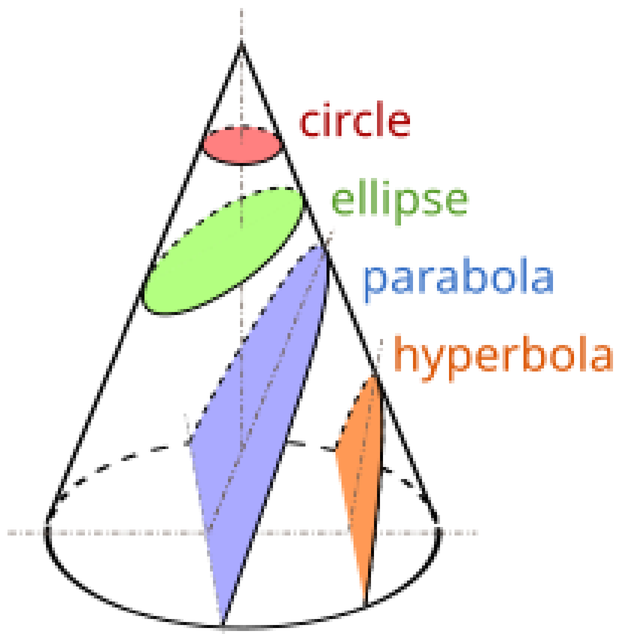


https://jacquesheunis.com/post/fortunes-algorithm/

But first a sidetrack to parabolas…

- What is a parabola?
  - One of the conic sections:

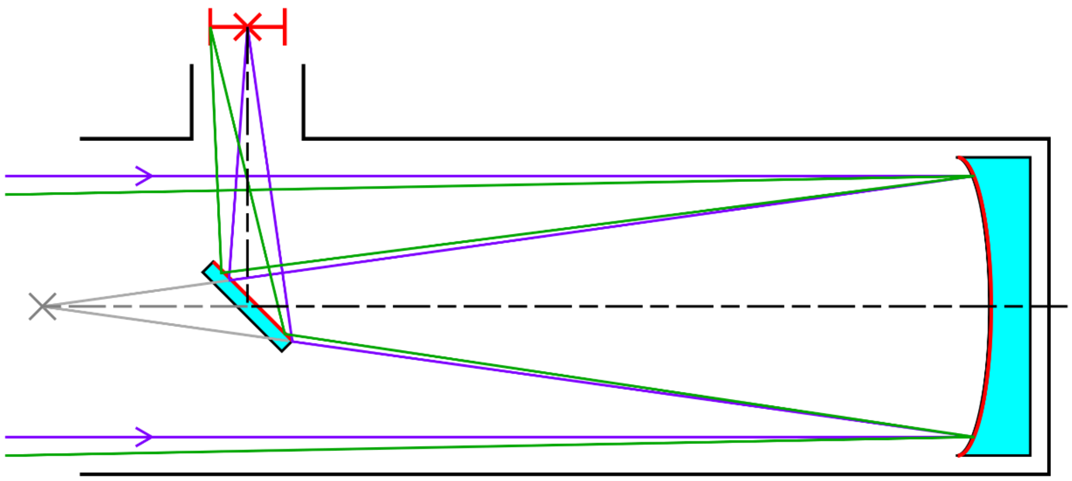
$y = ax^2 + bx + c$ with $a \neq 0$, or simply $y = ax^2$



https://en.wikipedia.org/wiki/Parabola

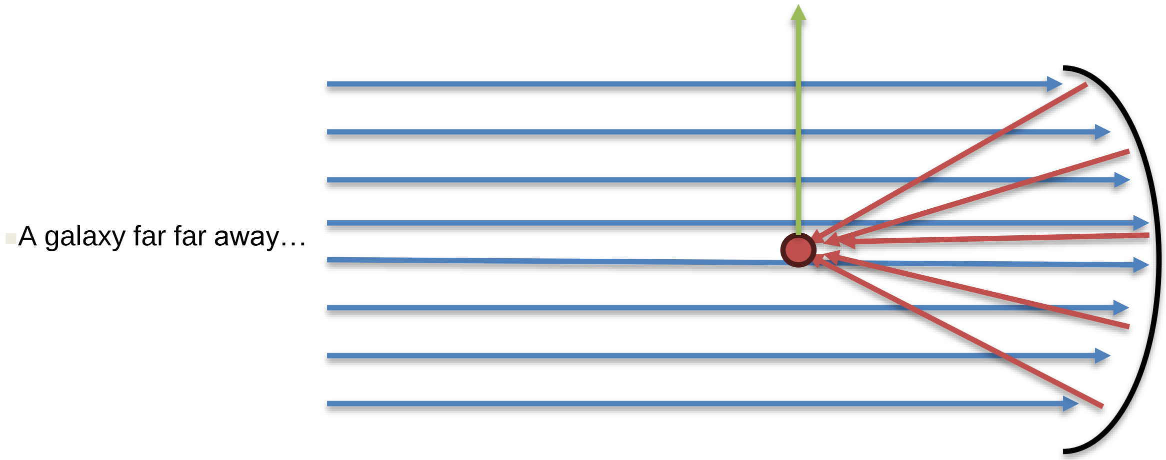
Properties (or other definitions)?

- Newtonian Telescope:



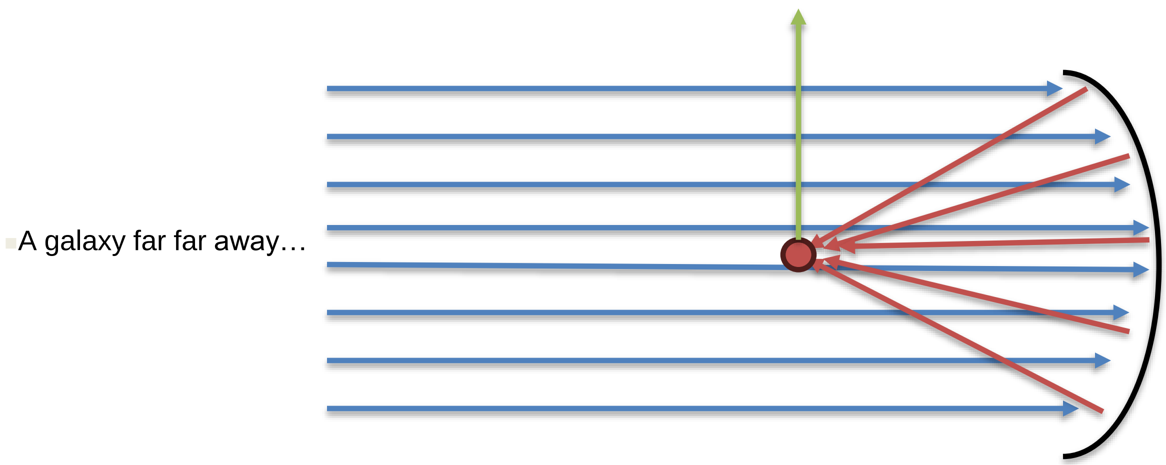
Properties (or other definitions)?

- Newtonian Telescope:



Properties (or other definitions)?

- Inverse Newtonian Telescope?



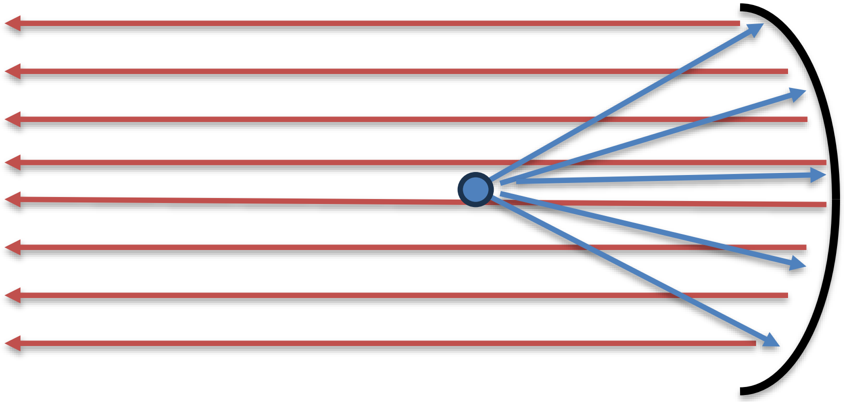
Properties (or other definitions)?

- Headlight:

- Antenna:

Properties (or other definitions)?

- Headlight:



- Antenna:
  - radio, cell tower, etc…

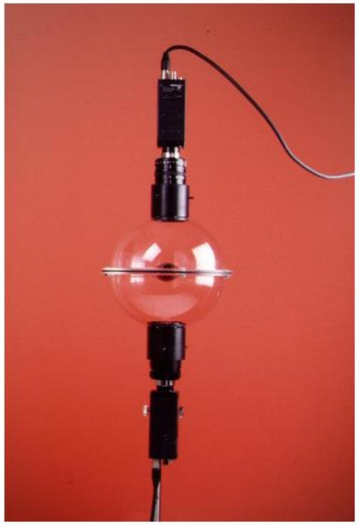
Properties (or other definitions)?

- Camera:
  - “Paraboloidal Catadioptric Camera”

[Aliaga et al.]

Properties (or other definitions)?

- Camera:
  - “Paraboloidal Catadioptric Camera”

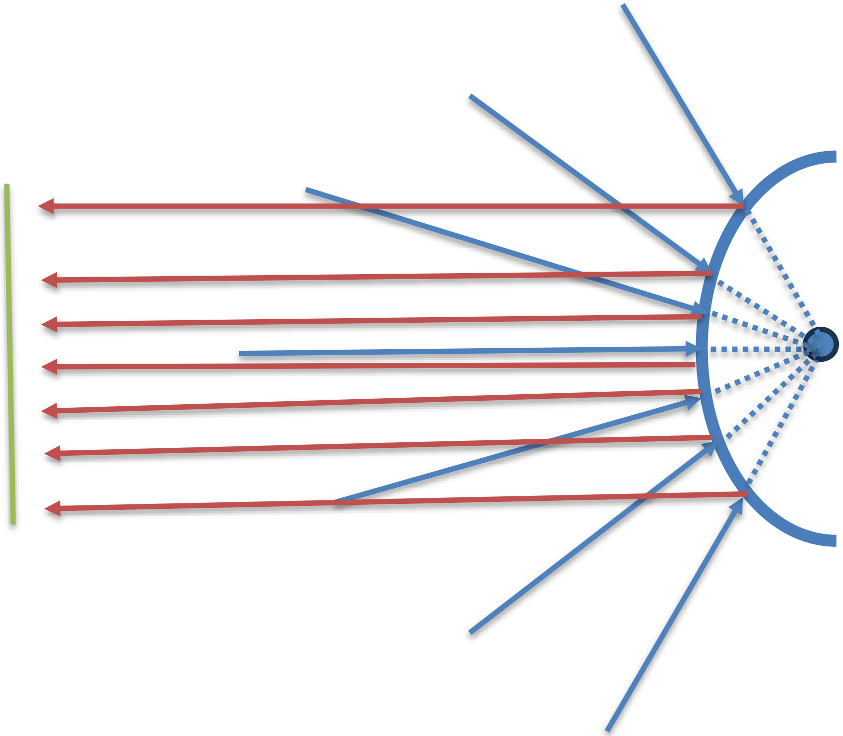
Properties (or other definitions)?

- Camera:
  - "Paraboloidal Catadioptric Camera"

Theory: 180º FOV reflects to an orthographic projection

Recall: true orthographic projection does not exist (i.e., telecentric lens is an approximation)

What to do?



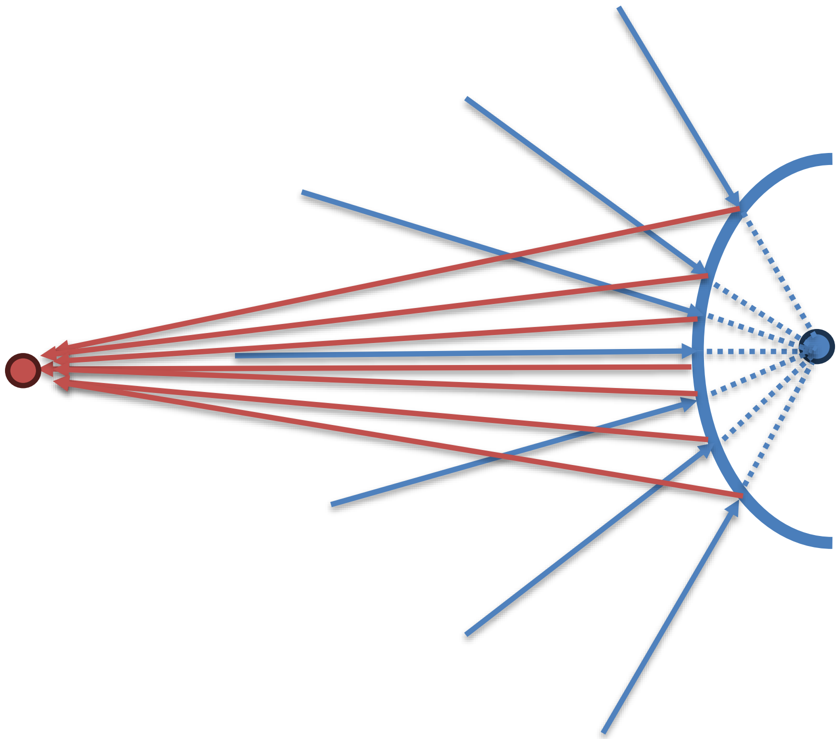
Properties (or other definitions)?

- Camera:
  - "Paraboloidal Catadioptric Camera"

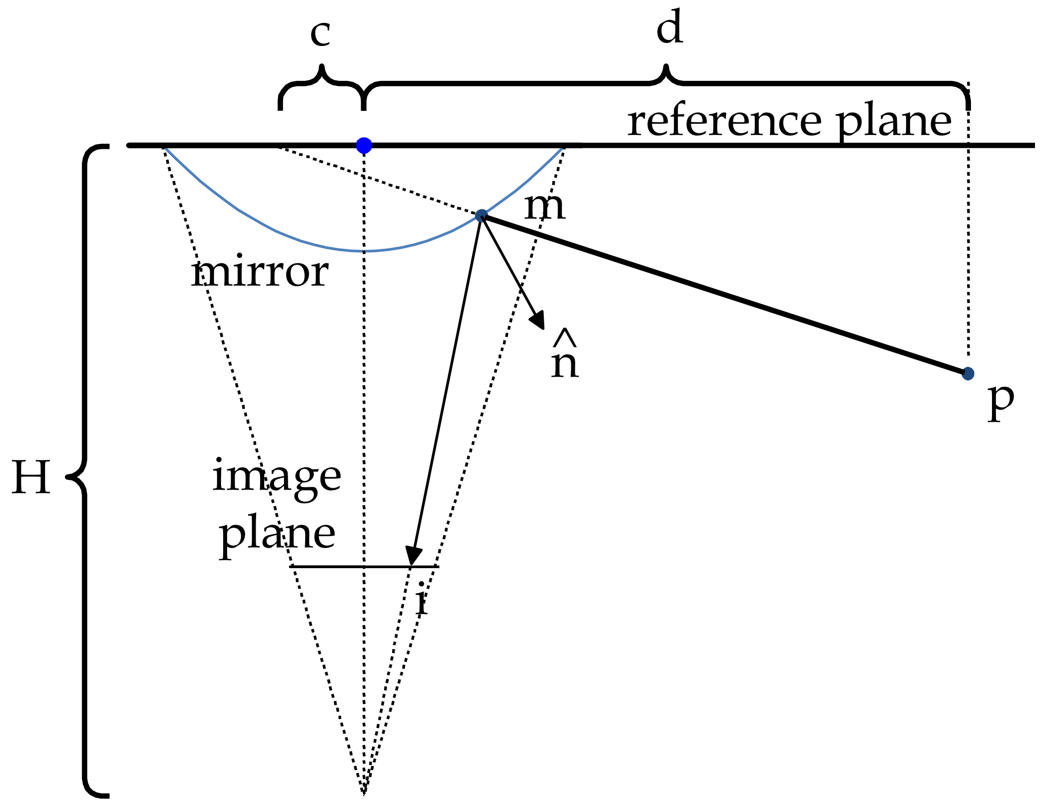
Theory: 180º FOV reflects to an orthographic projection

Recall: true orthographic projection does not exist (i.e., telecentric lens is an approximation)

What to do?



Our Camera PCC Model



A paraboloidal catadioptric setup that accounts for perspective projection occurring in a practical system

Our Camera PPC Model

- Assuming incident equals reflected angle:

$$\frac{i - m}{\| i - m \|} \cdot \frac{\hat{n}}{\| \hat{n} \|} = \frac{p - m}{\| p - m \|} \cdot \frac{\hat{n}}{\| \hat{n} \|}$$

- And given a 3D point *p*, mirror radius *r*, convergence distance *H*, we group and rewrite in terms of m_r:

$$m_r^5 - p_r m_r^4 + 2r^2 m_r^3 + (2p_r rH - 2r^2 p_r) m_r^2 + (r^4 - 4r^2 p_z H) m_r - (r^4 p_r + 2r^3 H p_r) = 0$$

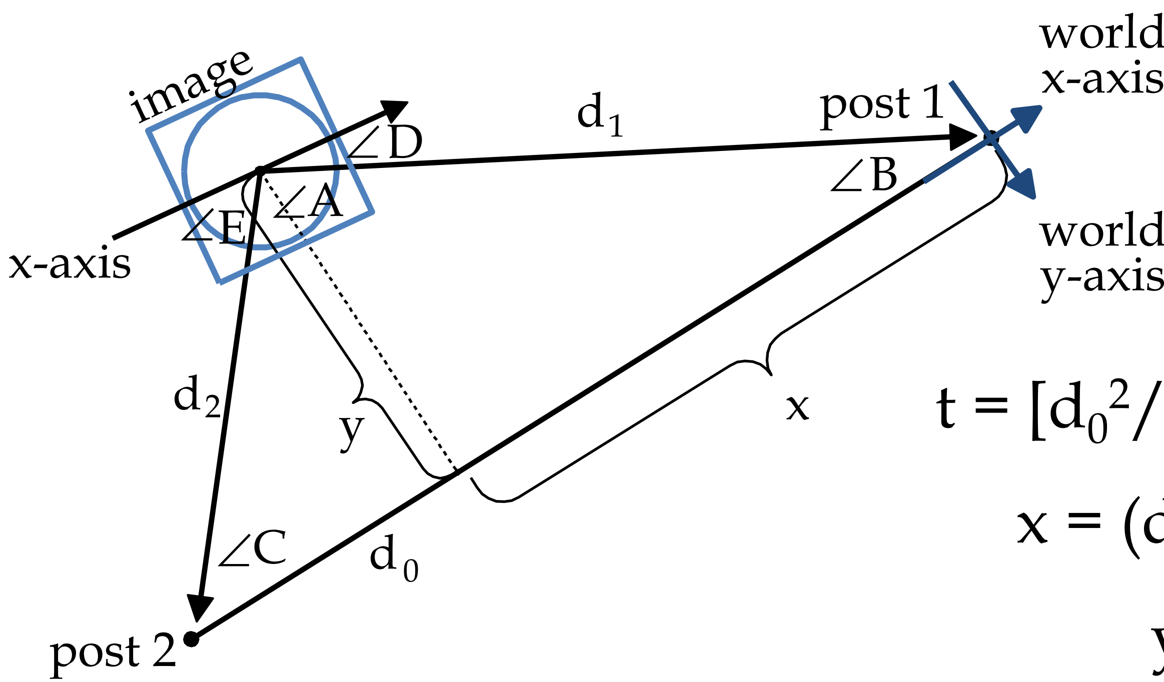
- To obtain a new expression for distance *d*:

$$\mathbf{d = (p_z m_p)/m_z - m_z/\tan(\alpha) + m_r}$$

Ex. Pose Estimation Setup

Our pose estimation algorithm uses beacons placed in the environment to triangulate position and orientation of the camera moving in a plane.

Position and Orientation



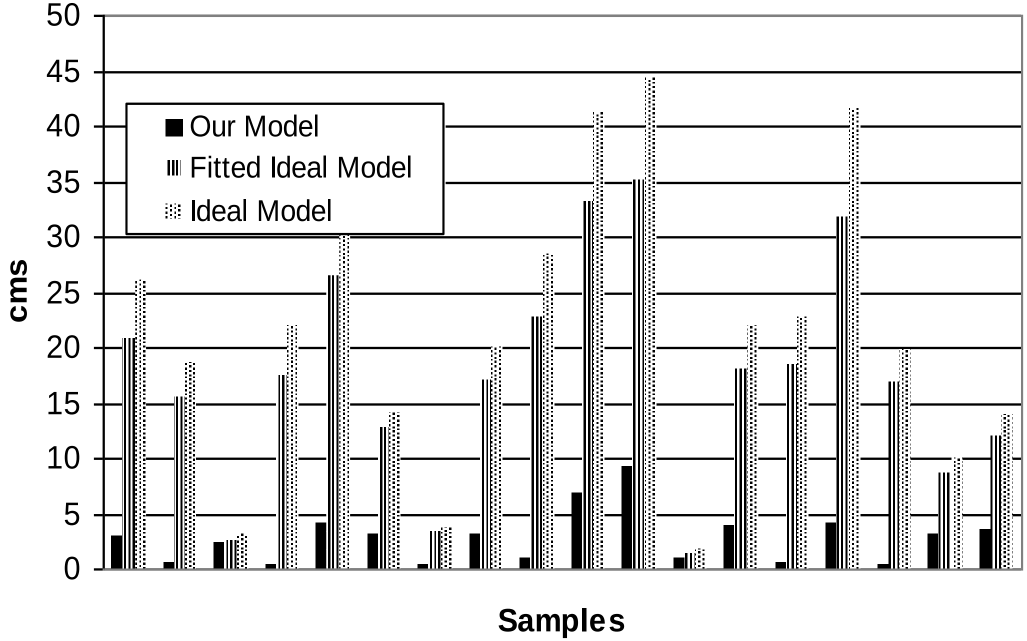
$$t = [d_0^2/(d_1^2+d_2^2-2d_1d_2\cos A)]^{1/2}$$

$$x = (d_0^2+t^2d_1^2-t^2d_2^2)/(2d0)$$

$$y = S(t^2d_1^2-x^2)^{1/2}$$

Our algorithm tracks the positions of small light bulbs and obtains camera position and orientation by solving an over-determined system.

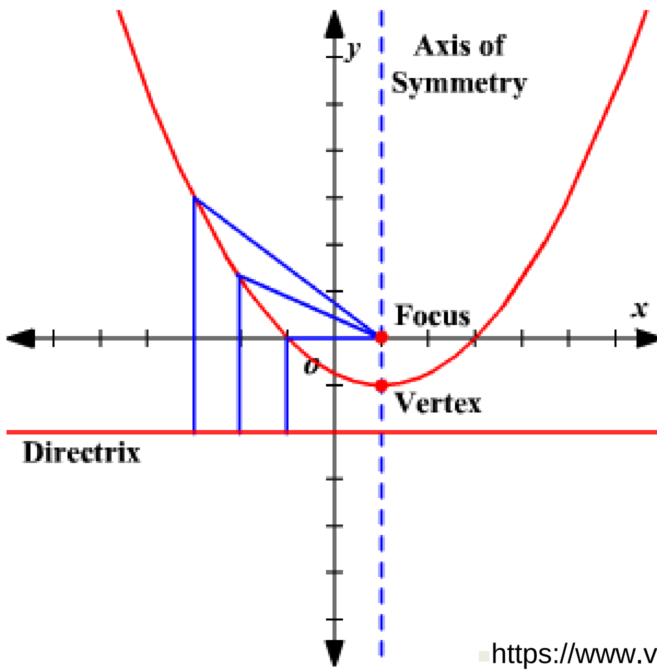
Pose Estimation Error



We achieve approximately an order of magnitude improvement over assuming an ideal catadioptric camera setup (as a percentage of the room diameter, mean error is 0.56% and σ=0.48%).

Properties (or other definitions)?

- What else?
  - Useful for triangulation! (kinda…)
  - Parabola is the line "equidistant" from the focus (i.e., a point) and a line (i.e., directrix)…



https://www.varsitytutors.com/hotmath/hotmath_help/topics/directrix

Fortune Triangulation

[https://jacquesheunis.com/post/fortunes-algorithm]