Distance and Collision Detection

Efi Fogel

efif@post.tau.ac.il

School of computer science, Tel Aviv University

The Papers

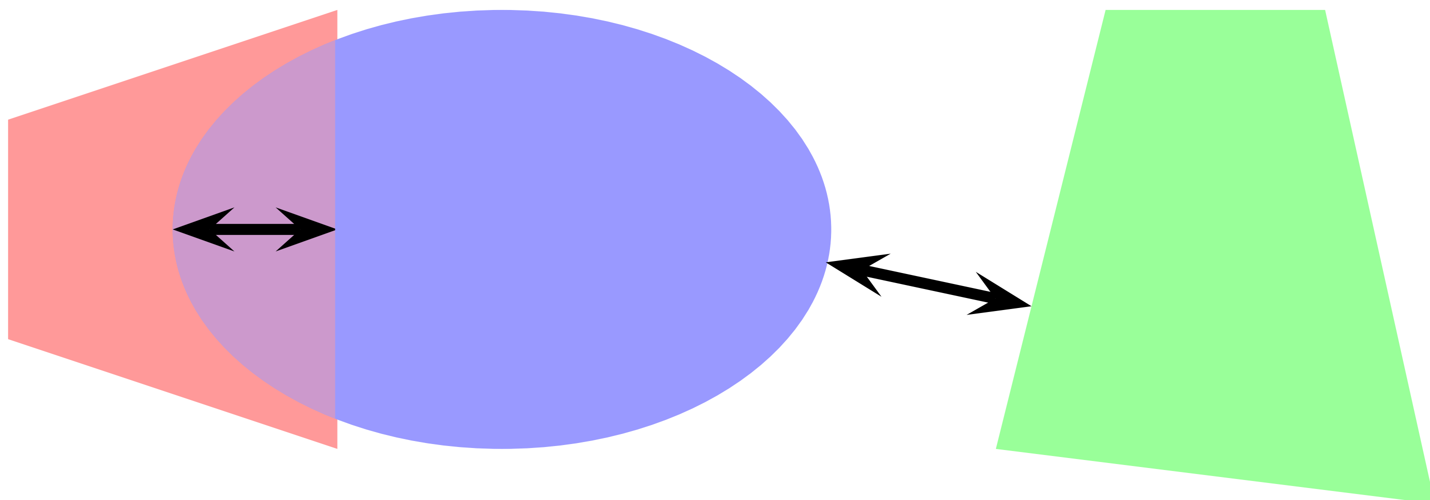
- **A Fast Procedure for Computing the Distance Between Complex Objects in Three-Dimensional Space**
- **Computing Minimum and Penetration Distances between Convex Polyhedra**

About the Algorithms

- First came Barr, Gilbert, and Wolfe. For example, “Finding the nearest point in a polytope” by Wolfe, 1976.
- In 1988 appeared “A Fast procedure for computing the distance between ...” known as GJK.
- S. Cameron enhanced GJK in “Computing Minimum and Penetration Distances between ...”.
  - and also described modifications to compute penetration distances
- Lin & Canny, “A fast Algorithm for incremental distance calculation”, 1991

Introduction

In many fields (e.g., Robotics, CAD, Graphics, etc) it is important to know whether two objects in 3D intersect or are in close proximity.

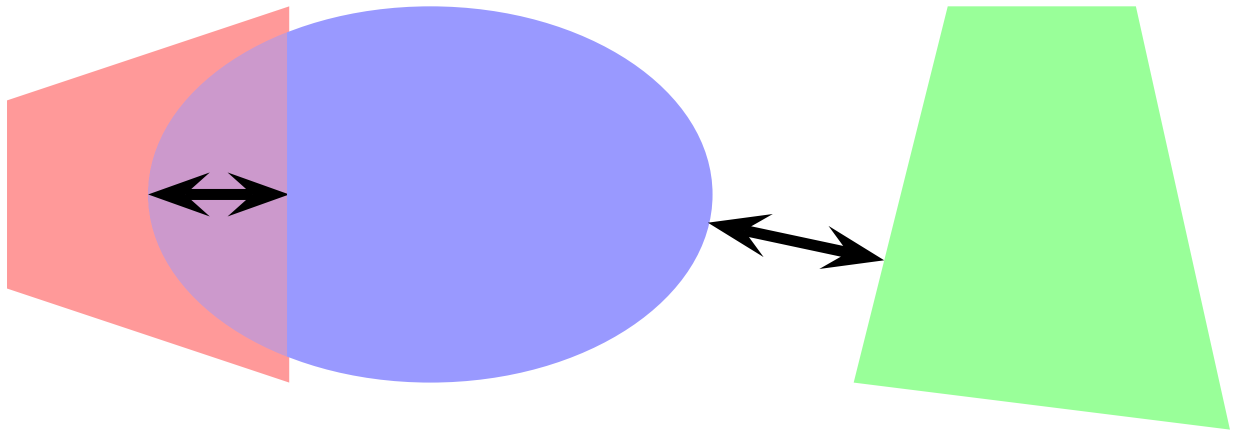


Minimum Translation Distance (Cameron)

When two simulated objects interpenetrate, we may need to know how to extricate the system from this condition.

$$\mathrm{MTD}^+(A,B) = \inf_t \{|t| : A + t \text{ is in contact with } B\}$$

$$\mathrm{MTD}(A,B) = \begin{cases} -\mathrm{MTD}^+(A,B) & \text{objects overlap} \\ \mathrm{MTD}^+(A,B) & \text{objects do not overlap} \end{cases}$$

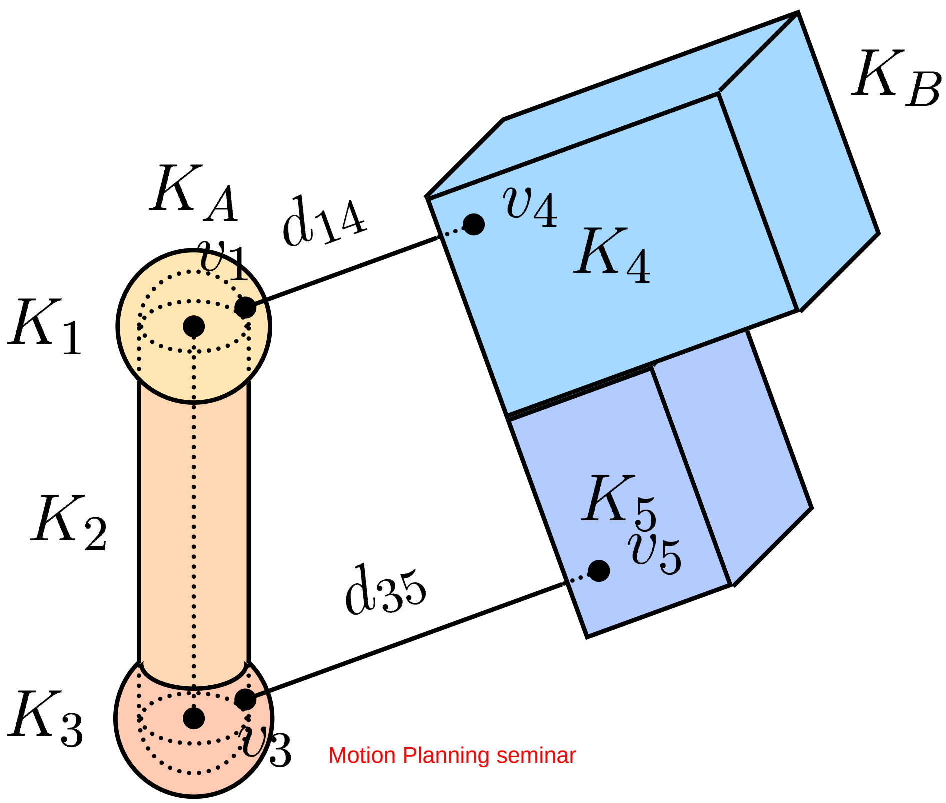


The Approach

- Compute the distance between convex sets in d-dimensional space
  - Efficient when $d = 3$
- Terminate after a finite number of iterations
  - Linear in the total number of vertices $m = m_1 + m_2$
- Practical

Handled Object Shapes and Representations

- Objects that are the union of convex polytopes and their spherical extensions
- Spherical extensions are valuable
  - May be used to cover an object with a safety shell
  - Economical representations



Preliminaries

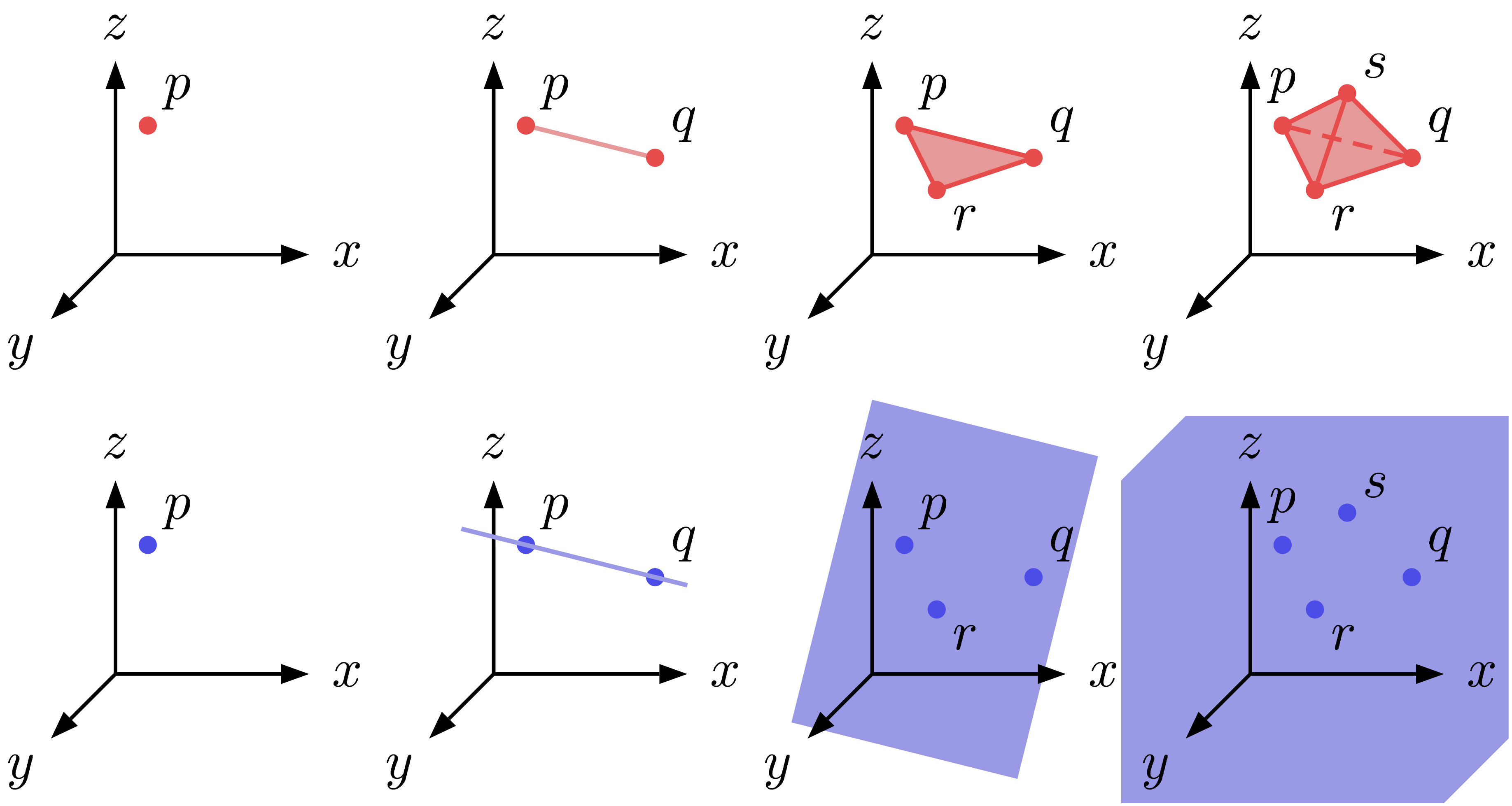
- The *affine hull* of a set $X \subseteq \mathbb{R}^d$, denoted by $\operatorname{aff}(X)$, is the intersection of all affine subspaces of $\mathbb{R}^d$ containing X.

$$\operatorname{aff}(X) = \{\sum_{i=1}^{l} \lambda^i x_i : x_i \in X, \sum_{i=1}^{l} \lambda^i = 1\}$$

- The *convex hull* of a set $X \subseteq \mathbb{R}^d$, denoted by $\operatorname{con}(X)$, is the intersection of all convex sets in $\mathbb{R}^d$ containing X.

$$\operatorname{con}(X) = \{\sum_{i=1}^{l} \lambda^i x_i : x_i \in X, \lambda^i \geq 0, \sum_{i=1}^{l} \lambda^i = 1\}$$

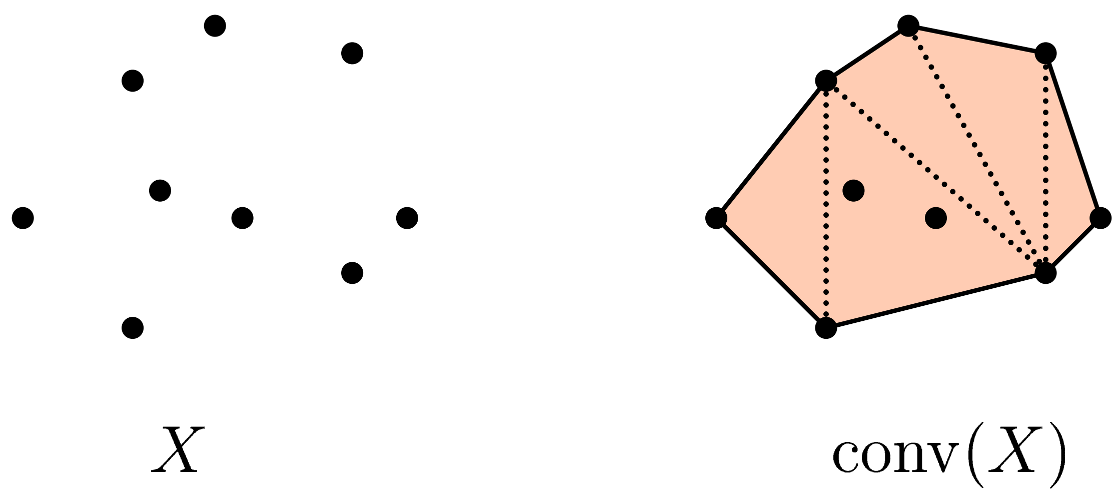
Convex and Affine hulls in $\mathbb{R}^3$



Caratheodory's theorem

Theorem 1 *Let $X \subseteq \mathbb{R}^d$. Then each point of $\mathrm{conv}(X)$ is a convex combination of at most $d+1$ points of X.*

For example, in the plane, $\mathrm{conv}(X)$ is the union of all triangles with vertices at points of X.



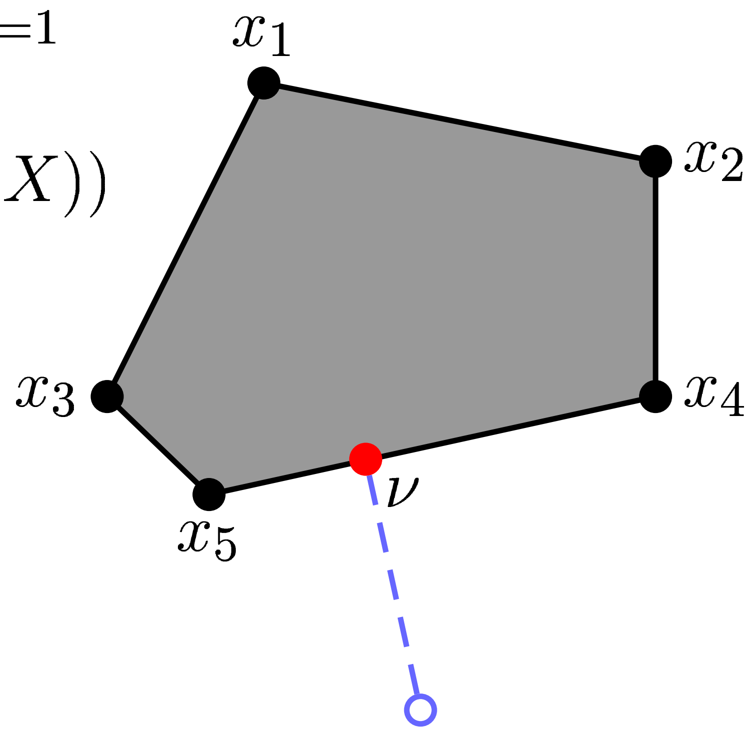
The nearest point to the origin

$\nu(X) \in X$ – nearest point in X to origin O,

$$|\nu(X)| = \min\{|x| : x \in X\}$$

$$\nu(\mathrm{con}(X)) = \sum_{i=1}^{l} \lambda^i x_i,\ x_i \in X,\ \lambda^i \geq 0,\ \sum_{i=1}^{l} \lambda^i = 1$$

$$l \leq \begin{cases} d+1 & \nu(\mathrm{con}(X)) = O,\ (O \in \mathrm{con}(X)) \\ d & \nu(\mathrm{con}(X)) \neq O \end{cases}$$

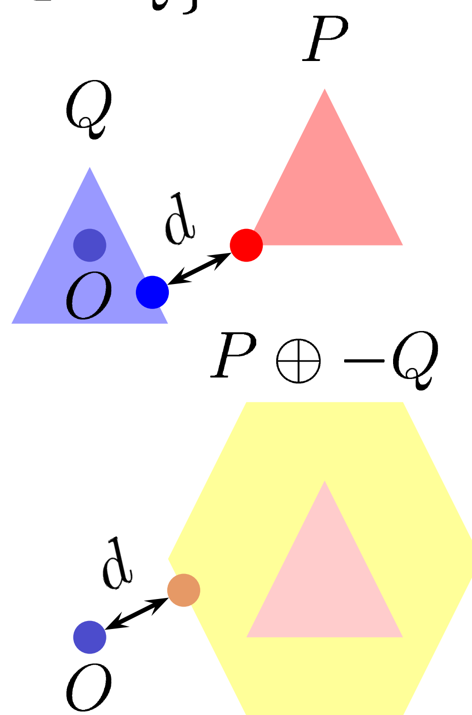


Translational C-space Obstacle (Cameron)

$$\mathrm{TCSO}(P,Q) = \{p - q : p \in P,\ q \in Q\} = K$$

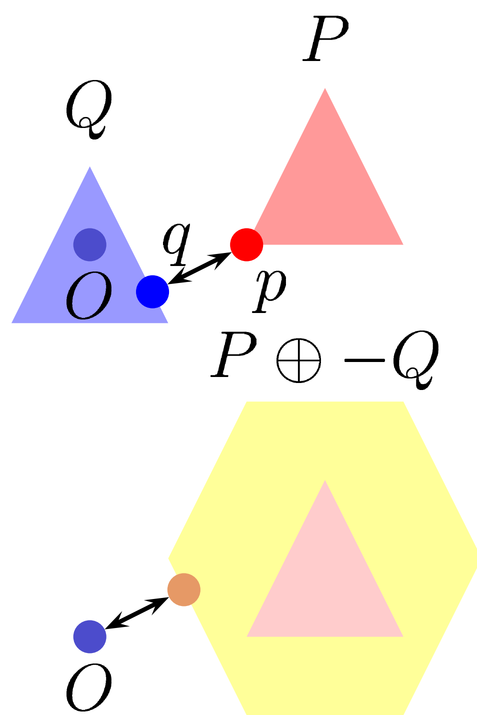
Recognized as Minkowski Sum

$$\begin{aligned}
\mathrm{TCSO}(P,Q) &= P \oplus -Q = \{p + \bar{q} : p \in P,\ \bar{q} \in -Q\} \\
\mathrm{MTD}(P,Q) &= \mathrm{MTD}(O,K) = d \\
&= \min\{|x| : x \in K\} = |\nu(K)| \\
\nu(K) &= \sum_{i \in I_K} \lambda^i x_i \\
&= \sum_{i \in I_P} \lambda^i p_i - \sum_{i \in I_Q} \lambda^i q_i
\end{aligned}$$



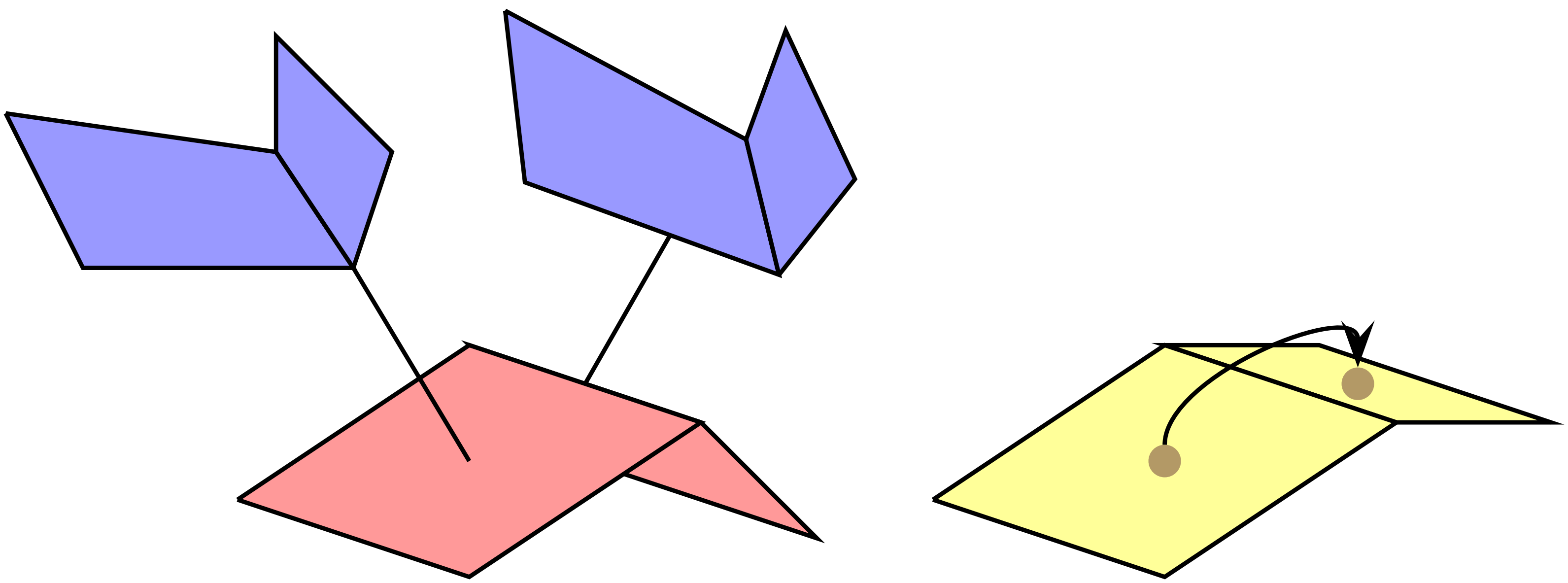
Witness Points

- p and q are the witness points - realize the minimum distance
  - Each is a surface point on P and Q resp.
  - Witness points are not necessary unique
- $p - q$ is the TC-witness point (Cameron)
  - A surface point on $\mathrm{TCSO}(P, Q)$



Tracking

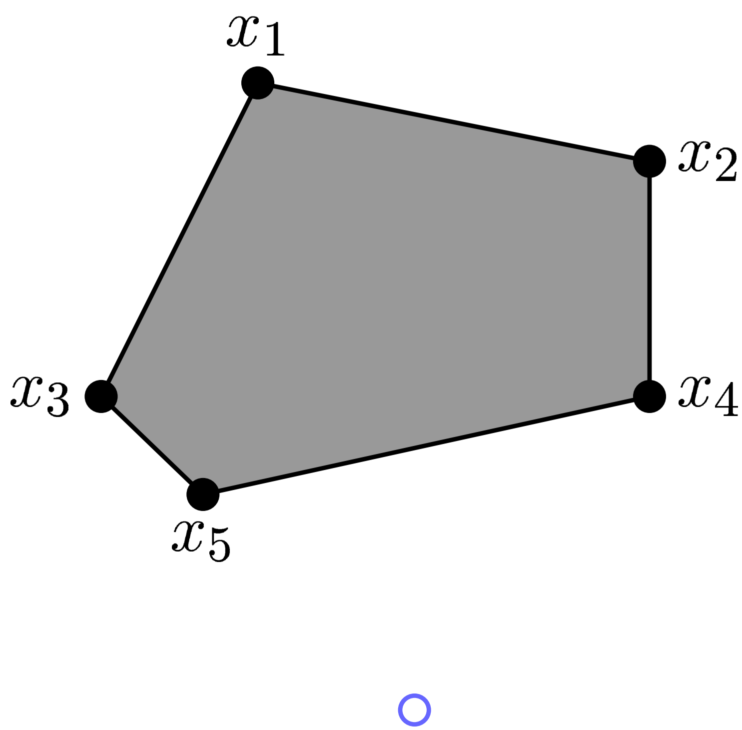
- The distance algorithm is called many times in time steps
- Make sense to use the witness points found at the last step



Algorithm Sketch

- Finding the nearest point to the origin

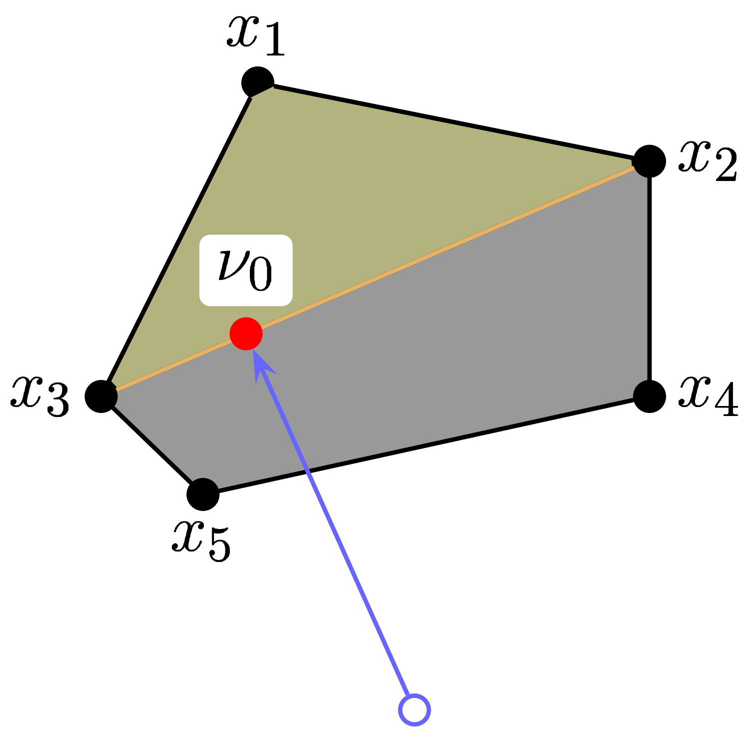
An example in $\mathbb{R}^2$



Algorithm Sketch

- Finding the nearest point to the origin

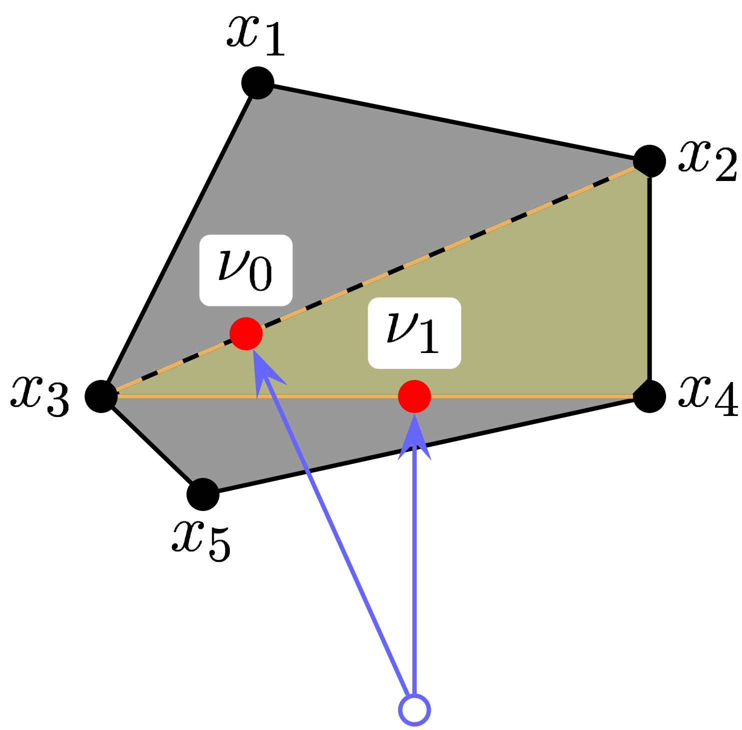
An example in $\mathbb{R}^2$



Algorithm Sketch

- Finding the nearest point to the origin

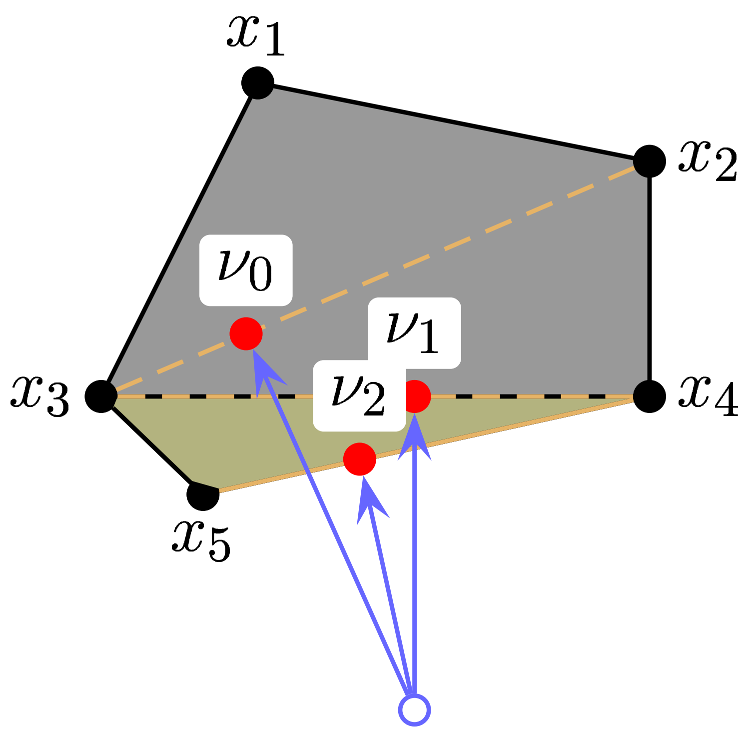
An example in $\mathbb{R}^2$



Algorithm Sketch

- Finding the nearest point to the origin

An example in $\mathbb{R}^2$



The algorithm (Cameron)

Require: X is a compact convex set in $\mathbb{R}^d$

$S \leftarrow \mathrm{init\_simplex}(X)$

while $!\mathrm{best\_simplex}(S)$ **do**

$S \leftarrow \mathrm{refine\_simplex}(S, X)$

end while

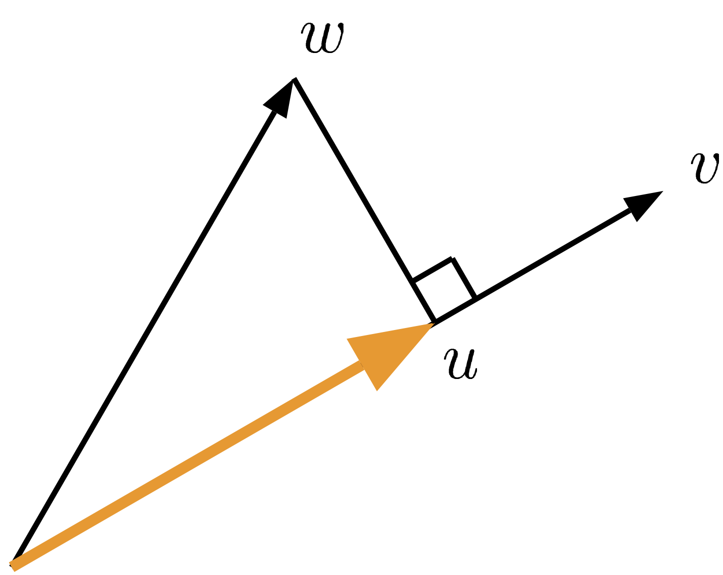
- $\mathrm{init\_simplex}(X)$ – computes the initial points $x_1, \ldots, x_v,\ 1 \leq v \leq d+1$
- $\mathrm{best\_simplex}(X)$ – returns true if the simplex contains the witness point, and false otherwise.
- $\mathrm{refine\_simplex}(S, X)$ – computes a neighboring simplex

Inner (dot) Product

The projection of w onto the unit vector v, is the vector u, whose length is $||w||$ times the cosine of the angle between v and w.

$$||u|| = \frac{v \cdot w}{||v||}$$

$$||u||^2 = u \cdot w$$



Notations

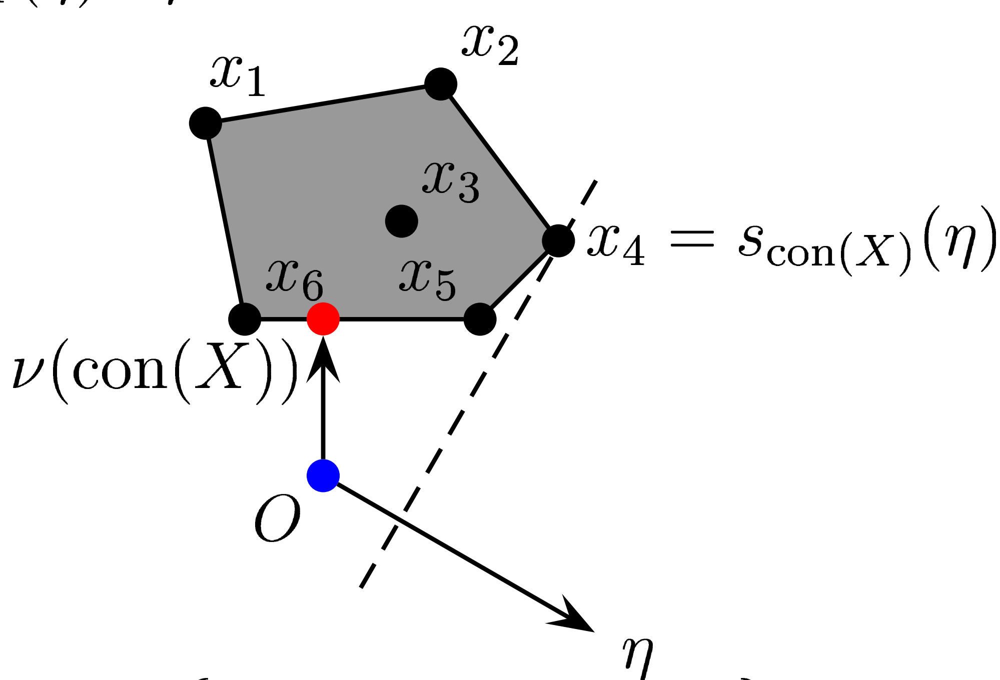
- $h_X(\eta)$ – the support function of X, $h_X : \mathbb{R}^d \to \mathbb{R}$,

$$h_X(\eta) = \max\{x \cdot \eta : x \in X\}$$

- $s_X(\eta)$ – the support vertex, any witness of $h_X(\eta)$,

$$h_X(\eta) = s_X(\eta) \cdot \eta$$

$$h_{\mathrm{con}(X)}(\eta) = h_X(\eta)$$
$$s_{\mathrm{con}(X)}(\eta) = s_X(\eta)$$



$X = \{x_1, x_2, x_3, x_4, x_5, x_6\}$

Minkowski Sum

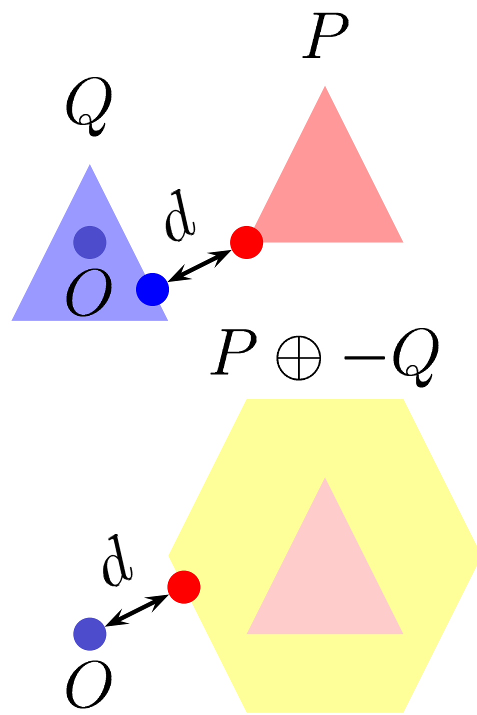
$$K = P \oplus -Q = \{p - q : p \in P,\ q \in Q\}$$

$$d = \min\{|x| : x \in K\} = |\nu(K)|$$

$$\nu(K) = \sum_{i \in I_K} \lambda^i x_i = \sum_{i \in I_P} \lambda^i p_i - \sum_{i \in I_Q} \lambda^i q_i$$

$$h_K(\eta) = h_P(\eta) + h_Q(-\eta)$$

$$s_K(\eta) = s_P(\eta) - s_Q(-\eta)$$



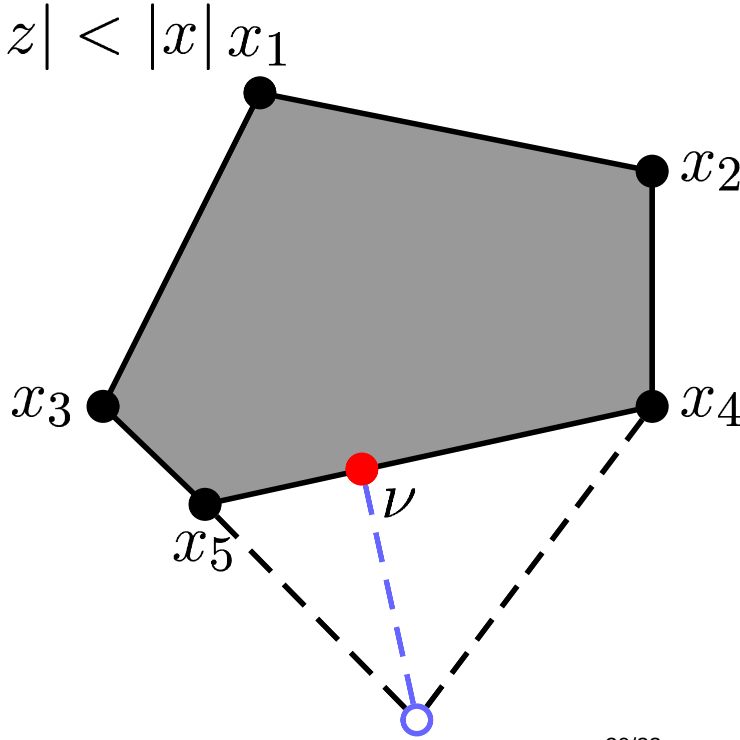
Theorem

Theorem 1 *Let $K \subseteq \mathbb{R}^d$ be compact and convex, and define $g_K : \mathbb{R}^d \to \mathbb{R}$ by:*

$$g_K(x) = |x|^2 + h_K(-x)$$

Suppose $x \in K$. Then:

- $g_K(x) > 0 \Rightarrow \exists z \in \mathrm{con}\{x, s_K(-x)\},\ |z| < |x|$
- $x = \nu(K) \Leftrightarrow g_K(x) = 0$
- $|x - \nu(K)|^2 \leq g_K(x)$



Theoretical Algorithm

Require: $X \subset \mathbb{R}^d$ is compact and convex,
$x_1, x_2, \ldots, x_v \in X,\ 1 \leq v \leq d+1$

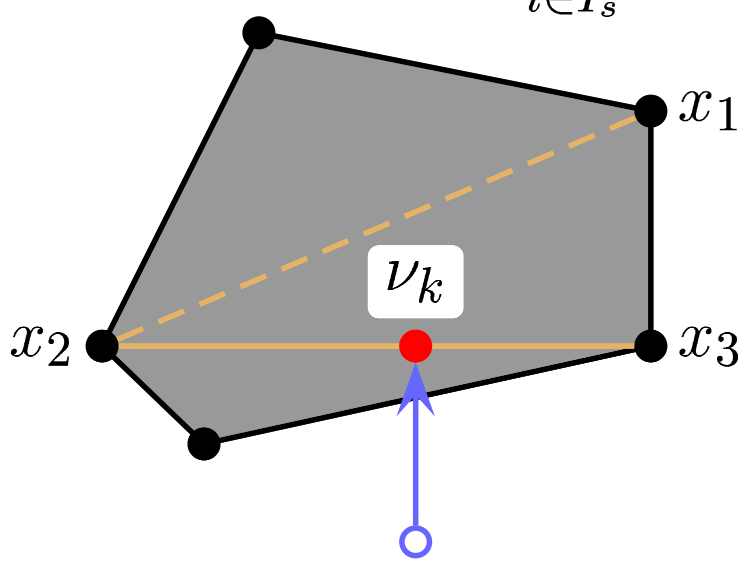
1: $k \leftarrow 0,\ S_0 \leftarrow x_1, x_2, \ldots, x_v$
2: $\nu_k = \nu(\mathrm{con}(S_k))$
3: **if** $g_K(\nu_k) == 0$ **then**
4: $\quad \nu(K) \leftarrow \nu_k$
5: $\quad$ stop
6: **end if**
7: $S_{k+1} \leftarrow \hat{S}_k \cup \{s_K(-\nu_k)\}$, where $\hat{S}_k \subset S_k$ has d elements or less and satisfies $\nu_k \in \mathrm{con}(\hat{S}_k),\ k \leftarrow k+1$
8: goto step 2.

Distance Subalgorithm

- Consider the k-th iteration, $S_k = \{x_1, x_2, \ldots, x_v\}$
- We need to compute:

$$\nu_k = \nu(\mathrm{con}(S_k)) = \sum_{i=1}^{v} \lambda^i x_i \qquad \textstyle\sum_{i=1}^{v} \lambda^i = 1,\ \lambda^i \geq 0$$

$$= \sum_{i \in I_s} \lambda^i x_i, \quad i \in I_s \subseteq \{1,2,\ldots,v\},\ \textstyle\sum_{i \in I_s} \lambda^i = 1,\ \lambda^i > 0$$



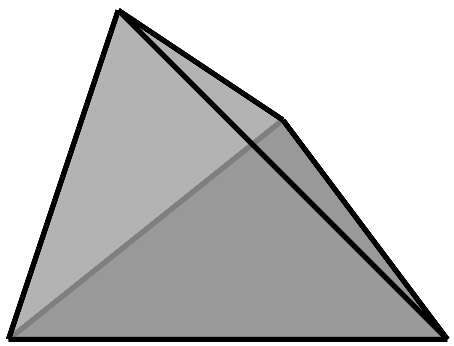
$I_s = \{2, 3\}$

Distance Subalgorithm, D.W. Johnson

- The number of all possible subsets of S_k is:

$$\sigma = \sum_{j=1}^{v} \frac{v!}{j!(v-j)!}$$

- For example, In $\mathbb{R}^3, \ v = 4, \ \sigma = 15$
  - 4 vertices, 6 open edges, 4 open faces, 1 open simplex.



Distance subalgorithm

- $S = \{x_1, x_2, \dots, x_{v=d+1}\}$ simplex $\mathbb{R}^d$. $I = \{1, 2, \dots, v\}$
- $S_s, \; s = 1, 2, \dots, \sigma$ an ordering of the subsets of S.
- Define $I_s, \; s = 1, 2, \dots, \sigma, \; S_s = \cup_{i \in I_s} \{x_i\}$
- Let I'_s be the complement of I_s in $I, \; I's = I \setminus I_s$
- Define real numbers $\Delta_i(S_s), \; i \in I_s$, and $\Delta(S_s)$:

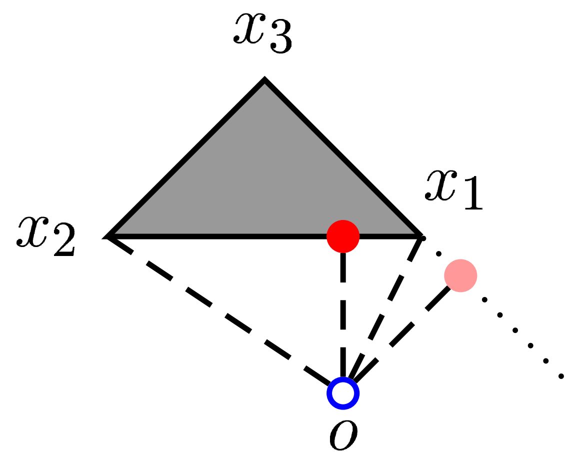
$$\Delta_i(\{x_i\}) = 1, \; i \in I$$

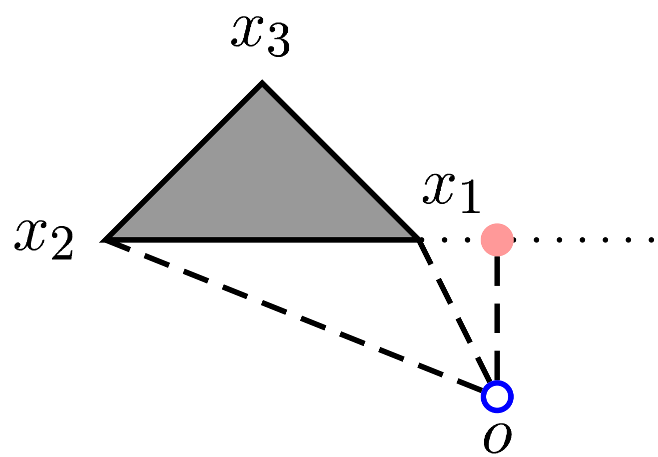
$$\Delta_j(S_s \cup \{x_j\}) = \sum_{i \in I_s} \Delta_i(S_s)(x_i \cdot x_k - x_i \cdot x_j), \; k \in I_s, \; j \in I'_s$$

$$\Delta(S_s) = \sum_{i \in I_s} \Delta_i(S_s)$$

Distance Subalgorithm

$$
\begin{aligned}
\Delta_1(\{x_1\}) &= 1 \\
\Delta_2(\{x_2\}) &= 1 \\
\Delta_3(\{x_3\}) &= 1 \\
\Delta_2(\{x_1, x_2\}) &= x_1 \cdot x_1 - x_1 \cdot x_2 \\
\Delta_1(\{x_1, x_2\}) &= x_2 \cdot x_2 - x_1 \cdot x_2 \\
\Delta_3(\{x_2, x_3\}) &= x_2 \cdot x_2 - x_2 \cdot x_3 \\
\Delta_2(\{x_2, x_3\}) &= x_3 \cdot x_3 - x_2 \cdot x_3 \\
\Delta_1(\{x_3, x_1\}) &= x_3 \cdot x_3 - x_3 \cdot x_1 \\
\Delta_3(\{x_3, x_1\}) &= x_1 \cdot x_1 - x_3 \cdot x_1 \\
\Delta_1(\{x_1, x_2, x_3\}) &= \ldots \\
\Delta_2(\{x_1, x_2, x_3\}) &= \ldots \\
\Delta_3(\{x_1, x_2, x_3\}) &= \ldots
\end{aligned}
$$





Theorem

Theorem 1

$$\nu(\mathrm{con}(S)) = \nu(\mathrm{con}(S_s)) = \sum_{i \in I_s} \lambda^i x_i,\ i \in I_s$$

if and only if

1. $\Delta(S_s) > 0$, *and*
2. $\Delta_i(S_s) > 0, \forall i \in I_s$, *and*
3. $\Delta_j(S_s \cup \{x_j\}) < 0,\ \forall j \in I'_s$, *and*
4. $\lambda^i = \frac{\Delta_i(S_s)}{\Delta(S_s)}$

Distance Subalgorithm

Require: $S = \{x_1, x_2, \ldots, x_v\}$, and an ordering $S_s, s = 1, 2, \ldots, \sigma$

1: $s \leftarrow 1$
2: **if** $\Delta(S_s) > 0$, and $\Delta_i(S_s) > 0, i \in I_s$, and $\Delta_j(S_s \cup \{x_j\}) \leq 0, j \in I'_s$ **then**
3:     Stop
4: **end if**
5: **if** $s < \sigma$ **then**
6:     Increment s and proceed to step 2
7: **end if**
8: Stop and report failure

Robustness Issues

How reliable is it in the presence of roundoff errors

- Errors do not accumulate!
  - Each iteration ν_k is recomputed based on S_k

$$\nu_k = \nu(\mathrm{con}(S_k))$$

Making the Main Algorithm Robust

- Translate the origin to a point on the line segment joining the centroids of P and Q

$$\rho = \frac{1}{2}(\bar{p} + \bar{q}) \qquad \bar{p} = \frac{1}{|P|}\sum_{p_i \in P} p_i,\ \bar{q} = \frac{1}{|Q|}\sum_{q_i \in Q} q_i$$

  - Helps when d is small and the ρ is large

- Replace the convergence criterion to:

$$g_K(\nu_k) \leq \epsilon(D(K))^2$$

  - $\epsilon > 0$ related to the number-type accuracy

$$\begin{aligned} D(K) &= \max\{|x| : x \in K\} \\ &\leq D(\operatorname{con}(P - \{\bar{p}\}) + D(\operatorname{con}(Q - \{\bar{q}\}) + |\bar{p} - \bar{q}| \end{aligned}$$

Making the Sub-Algorithm Robust

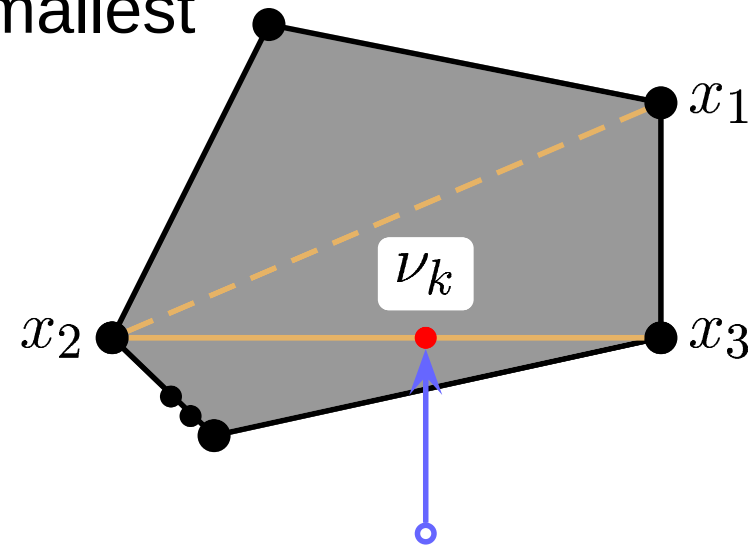
- The condition in the distance subalgorithm is not satisfied for any $s = 1, 2, \ldots, \sigma$
  - May happen when S_k is affinely dependent or nearly so
    - in $\mathbb{R}^3$ all 4 points are nearly coplanar
- Resort to a backup procedure:

Require: $S = \{s_1, s_2, \ldots, s_v\}$ a simplex
Compute the distance to all candidates $S_s \subset S$
{Compute $\nu(\text{aff}(S_s))$ for $\Delta(S_s) > 0, \Delta_i(S_s) > 0$}
Return the best

Hill Climbing (Cameron)

Expediting the computing of the support vertex

Given new support direction x, and previous support vertex v
compare $x \cdot v$ with $x \cdot v_j$ for every vertex v_j connected to v
if $x \cdot v$ is not the smallest **then**
  $v \leftarrow v_j$, such that $x \cdot v_j$ is the smallest
else
  return v
end if



Solving each Simplex

Estimating Penetration Distance

Objects overlap $\iff$ TC-space origin $\in$ TCSO

$$\mathrm{MTD}^+(O, \mathrm{con}(S)) \le \mathrm{MTD}^+(O, TCSO) \le \min_i |x_i|$$

$$\mathrm{MTD}^+(O, \mathrm{con}(S)) = \min\{|\nu(\mathrm{aff}(S_s))| : S_s \subset S,\ |S_s| = d\}$$