Minkowski Sums

Outline:

I. Definition

II. C-obstacles

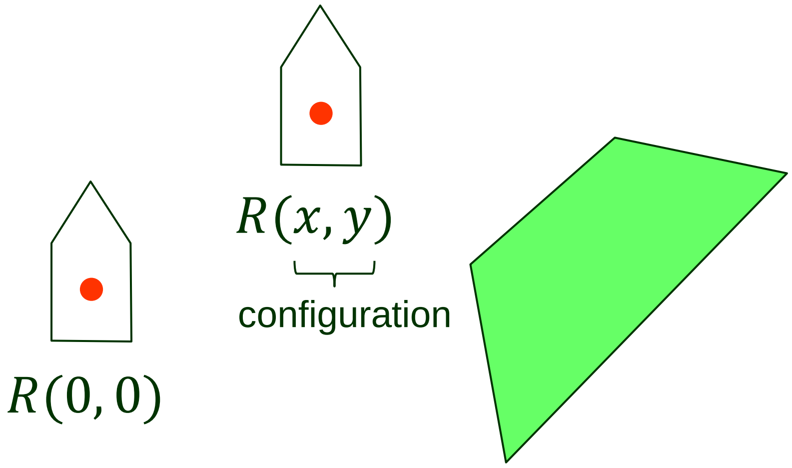
III. Complexity of the sum of two convex polygons

IV. Computation

V. Translational motion planning

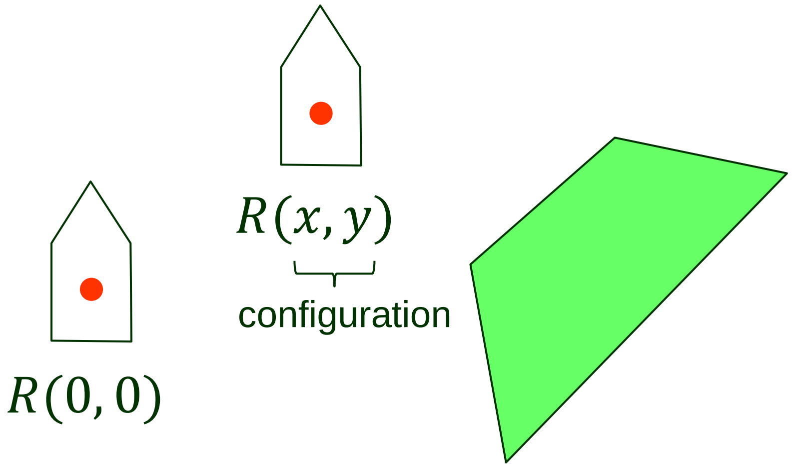
C-obstacle for a Translational Robot

Robot R Obstacle P



C-obstacle for a Translational Robot

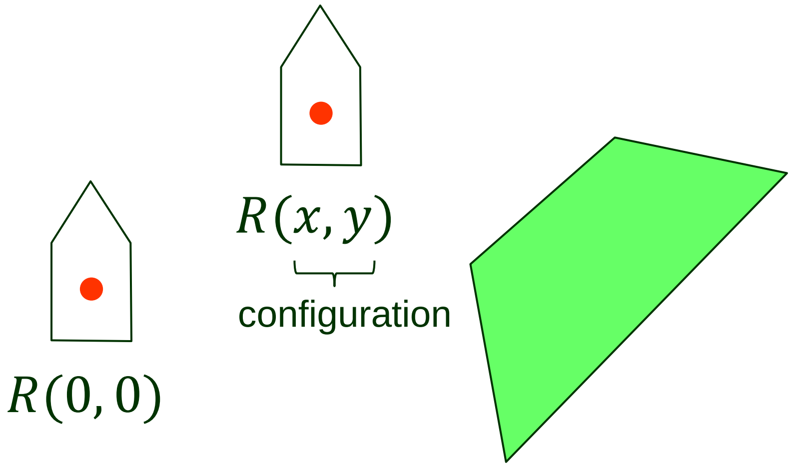
Robot R Obstacle P C-obstacle CP



$$CP = \{(x,y) \mid R(x,y) \cap P \neq \emptyset\}$$

C-obstacle for a Translational Robot

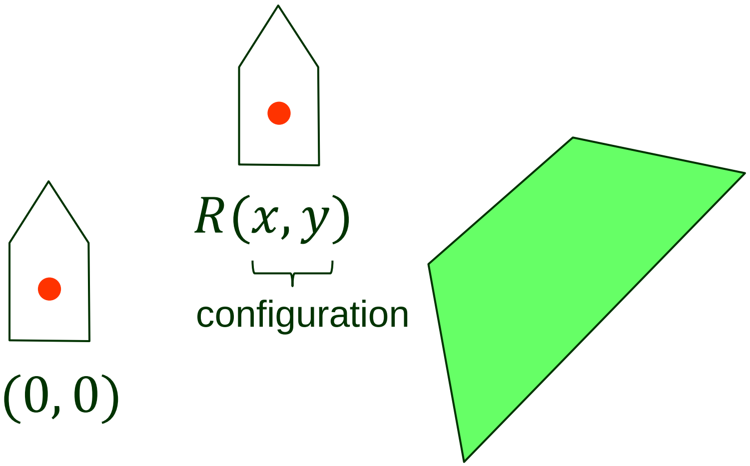
Robot R Obstacle P C-obstacle CP

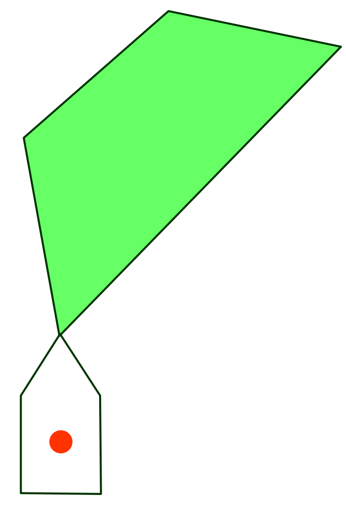


$$CP = \{(x,y) \mid \overbrace{R(x,y) \cap P \neq \emptyset}^{\text{collision}}\}$$

C-obstacle for a Translational Robot

Robot R Obstacle P C-obstacle CP

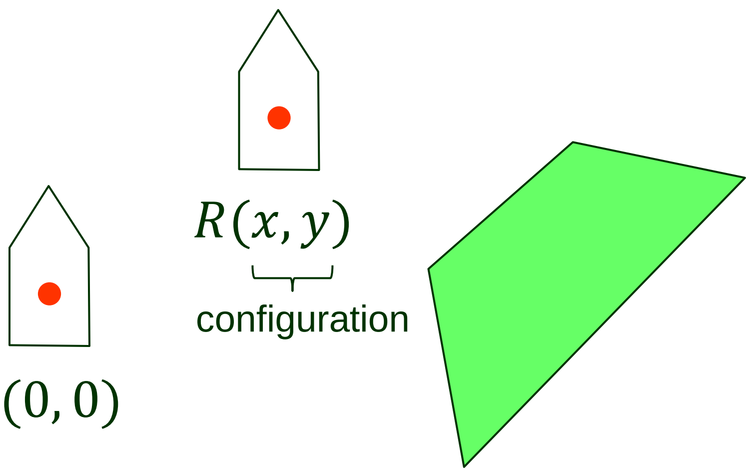


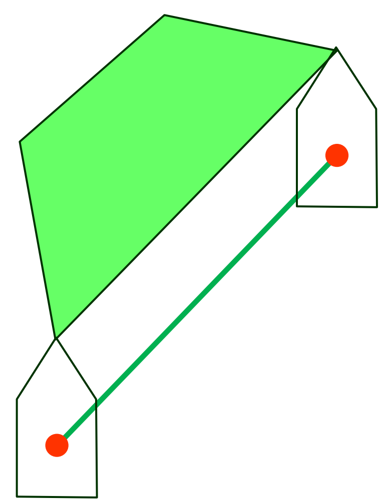


$$CP = \{(x,y) \mid \overbrace{R(x,y) \cap P \neq \emptyset}^{\text{collision}}\}$$

C-obstacle for a Translational Robot

Robot R Obstacle P C-obstacle CP

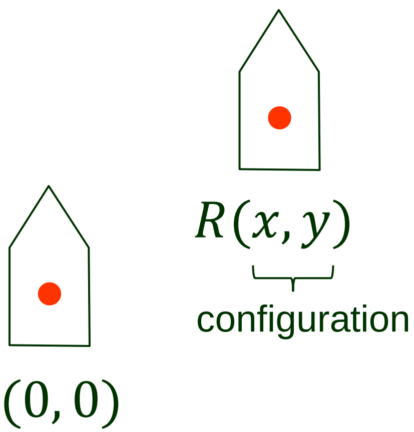


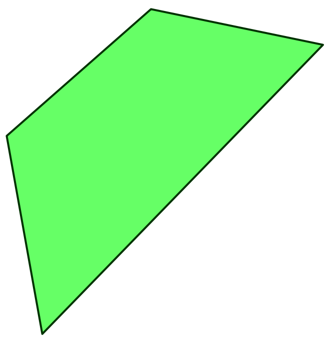


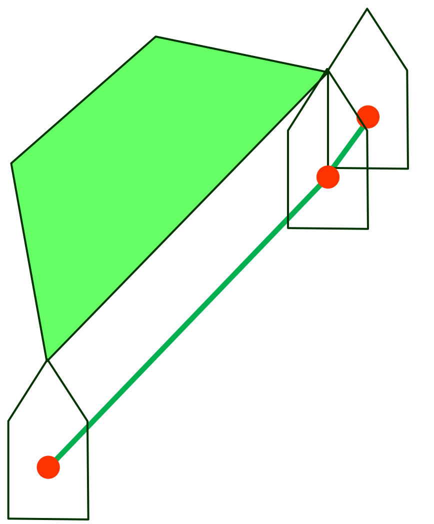
$$CP = \{(x,y) \mid \overbrace{R(x,y) \cap P \neq \emptyset}^{\text{collision}}\}$$

C-obstacle for a Translational Robot

Robot R | Obstacle P | C-obstacle CP



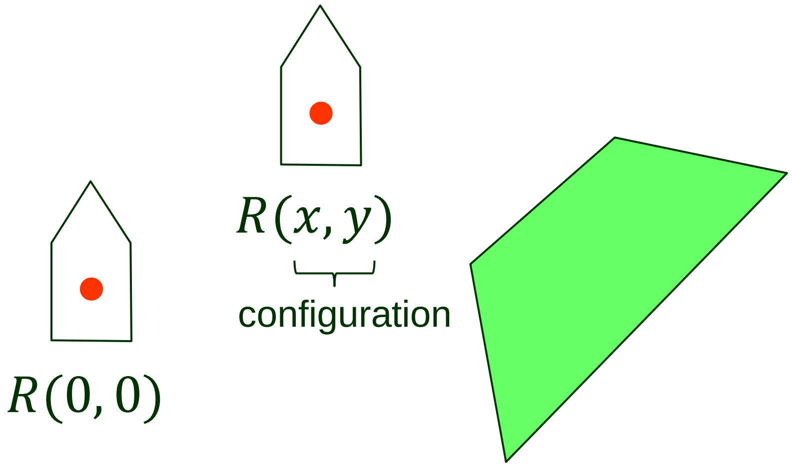


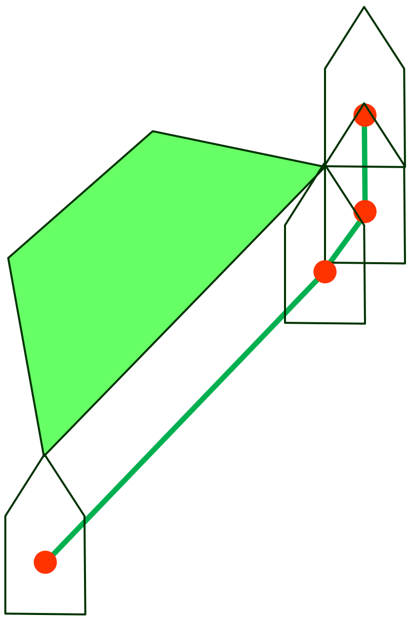


$$CP = \{(x,y) \mid \overbrace{R(x,y) \cap P \neq \emptyset}^{\text{collision}}\}$$

C-obstacle for a Translational Robot

Robot R Obstacle P C-obstacle CP

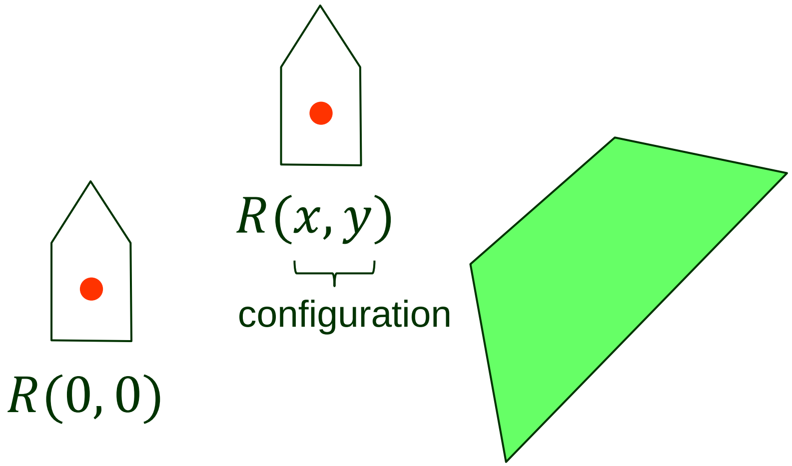


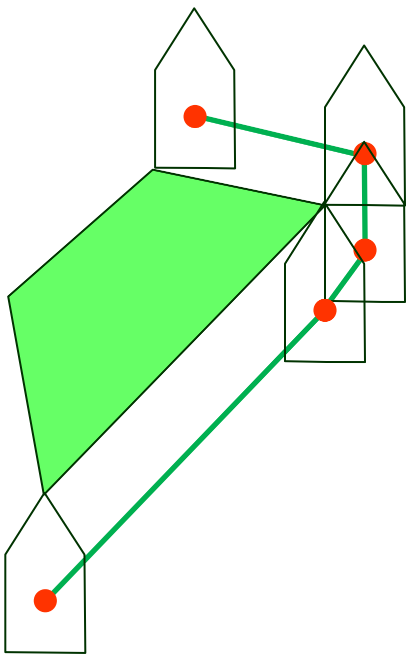


$$CP = \{(x,y) \mid \overbrace{R(x,y) \cap P \neq \emptyset}^{\text{collision}}\}$$

C-obstacle for a Translational Robot

Robot R Obstacle P C-obstacle CP

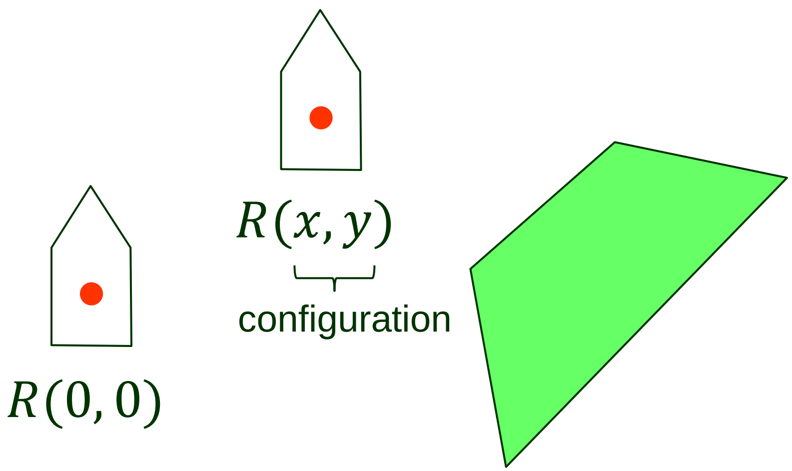


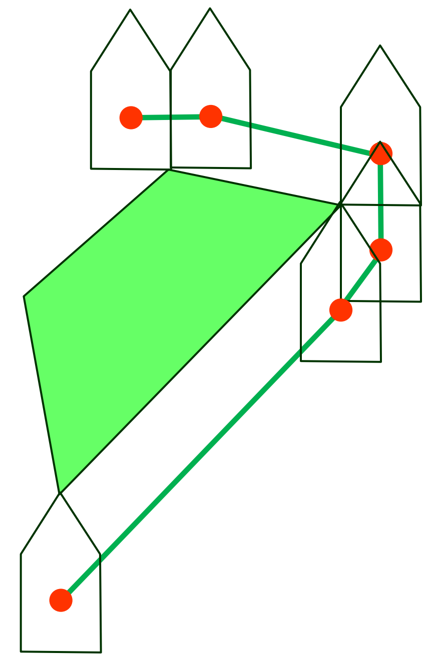


$$CP = \{(x,y) \mid \overbrace{R(x,y) \cap P \neq \emptyset}^{\text{collision}}\}$$

C-obstacle for a Translational Robot

Robot R Obstacle P C-obstacle CP

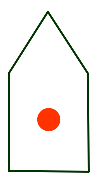


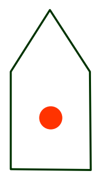


$$CP = \{(x, y) \mid \overbrace{R(x, y) \cap P \neq \emptyset}^{\text{collision}}\}$$

C-obstacle for a Translational Robot

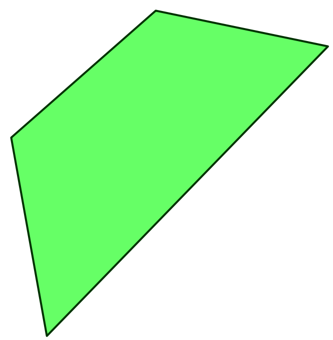
Robot R | Obstacle P | C-obstacle CP

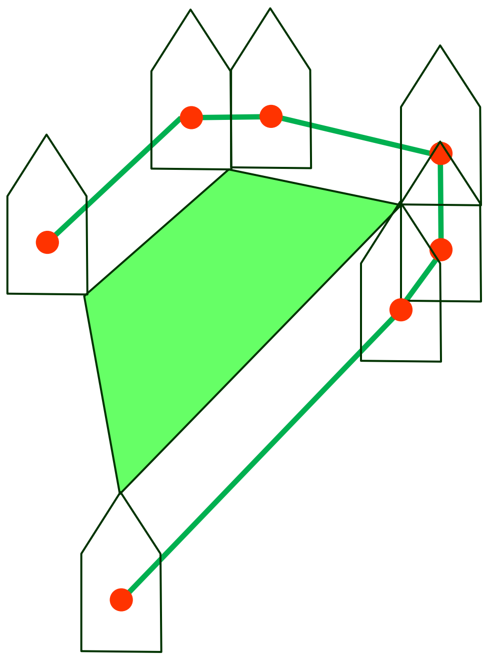




$R\underbrace{(x,y)}_{\text{configuration}}$

$R(0,0)$

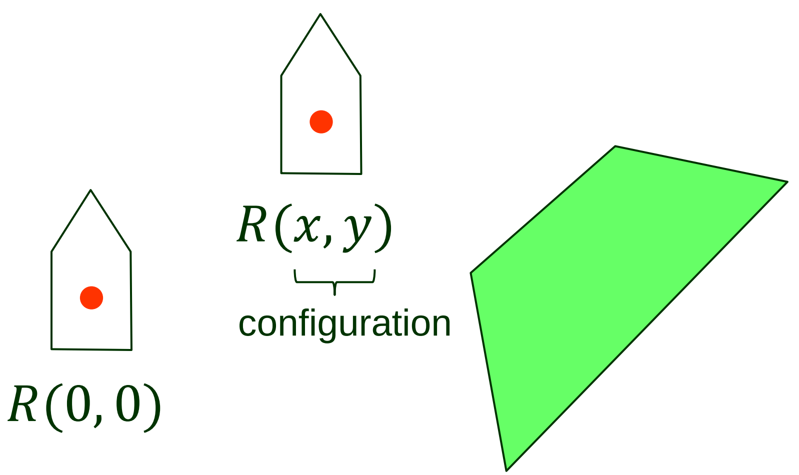


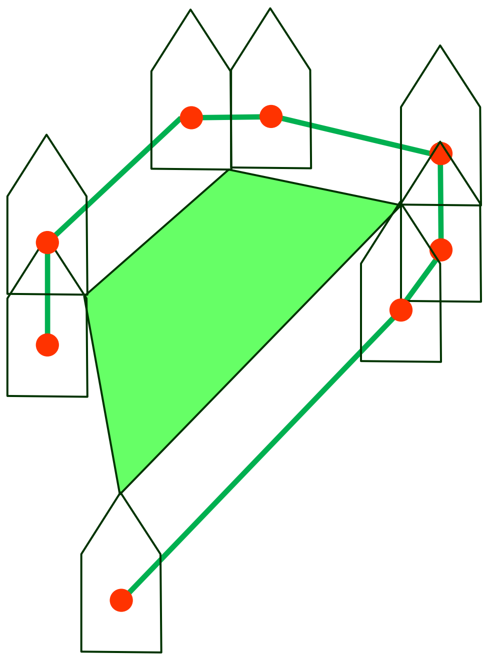


$$CP = \{(x,y) \mid \overbrace{R(x,y) \cap P \neq \emptyset}^{\text{collision}}\}$$

C-obstacle for a Translational Robot

Robot R Obstacle P C-obstacle CP

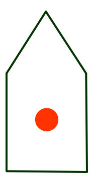




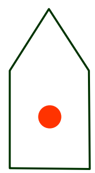
$$CP = \{(x,y) \mid \overbrace{R(x,y) \cap P \neq \emptyset}^{\text{collision}}\}$$

C-obstacle for a Translational Robot

Robot R

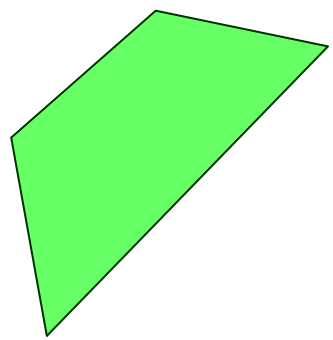


$R(0,0)$

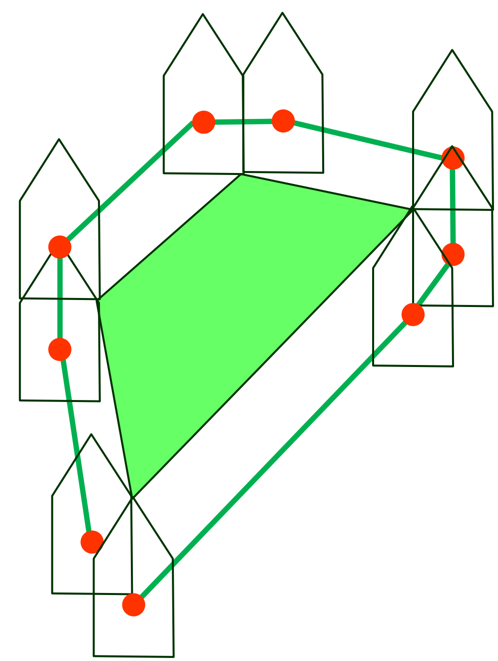


$\underbrace{R(x,y)}_{\text{configuration}}$

Obstacle P



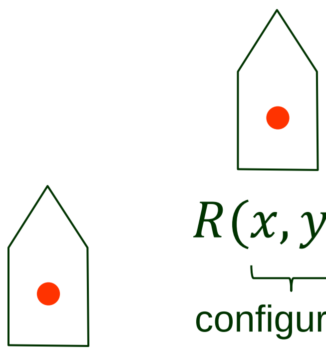
C-obstacle CP



$$CP = \{(x,y) \mid \overbrace{R(x,y) \cap P \neq \emptyset}^{\text{collision}}\}$$

C-obstacle for a Translational Robot

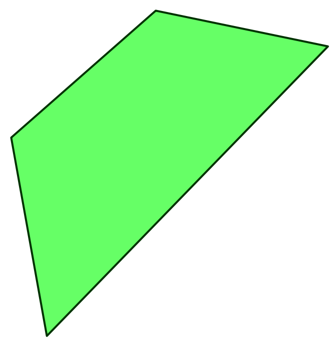
Robot R



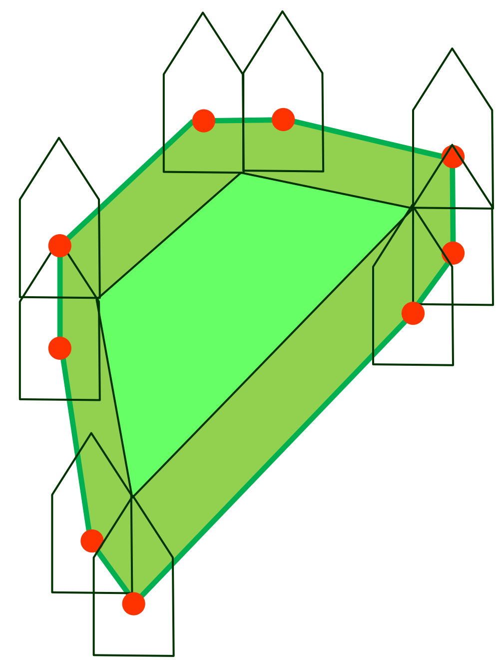
$\underbrace{R(x,y)}_{\text{configuration}}$

$R(0,0)$

Obstacle P



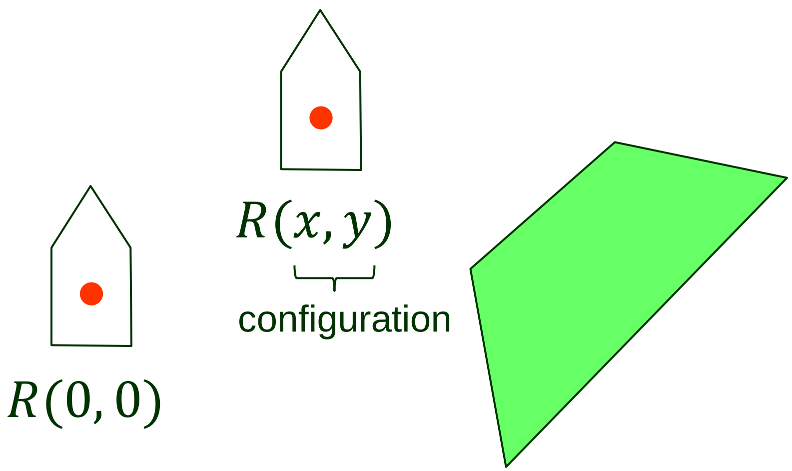
C-obstacle CP

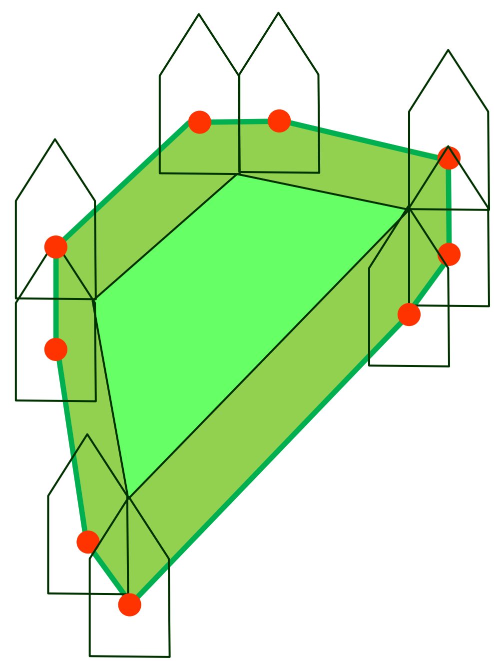


$$CP = \{(x,y) \mid \overbrace{R(x,y) \cap P \neq \emptyset}^{\text{collision}}\}$$

C-obstacle for a Translational Robot

Robot R Obstacle P C-obstacle CP





$$CP = \{(x,y) \mid \overbrace{R(x,y) \cap P \neq \emptyset}^{\text{collision}}\}$$

The boundary of CP is traced out by the reference point of R as it slides along the boundary of P.

I. Minkowski Sum

Hermann Minkowski (1864-1909) – Albert Einstein was his former student.

Two sets $S_1, S_2 \in \mathbb{R}^2$

$$S_1 \oplus S_2 = \{p + q \mid p \in S_1, q \in S_2\}$$

I. Minkowski Sum

Hermann Minkowski (1864-1909) – Albert Einstein was his former student.

Two sets $S_1, S_2 \in \mathbb{R}^2$

$$S_1 \oplus S_2 = \{p + q \mid p \in S_1, q \in S_2\}$$

$S_1 = \{1, 2\}, S_2 = \{-3, 0\}$

I. Minkowski Sum

Hermann Minkowski (1864-1909) – Albert Einstein was his former student.

Two sets $S_1, S_2 \in \mathbb{R}^2$

$$S_1 \oplus S_2 = \{p + q \mid p \in S_1, q \in S_2\}$$

$S_1 = \{1, 2\}, S_2 = \{-3, 0\}$

$1 + (-3) = -2$ $\quad$ $2 + (-3) = -1$

$1 + 0 = 1$ $\quad$ $2 + 0 = 2$

I. Minkowski Sum

Hermann Minkowski (1864-1909) – Albert Einstein was his former student.

Two sets $S_1, S_2 \in \mathbb{R}^2$

$$S_1 \oplus S_2 = \{p + q \mid p \in S_1, q \in S_2\}$$

$S_1 = \{1, 2\}, S_2 = \{-3, 0\}$

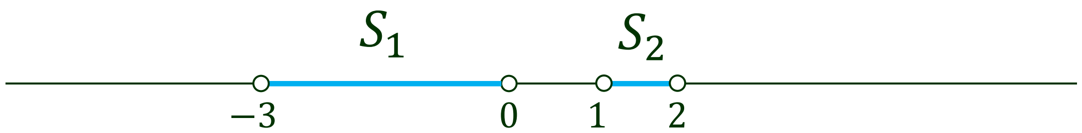
$1 + (-3) = -2$ $\quad$ $2 + (-3) = -1$

$1 + 0 = 1$ $\quad$ $2 + 0 = 2$

$$S_1 \oplus S_2 = \{-2, -1, 1, 2\}$$

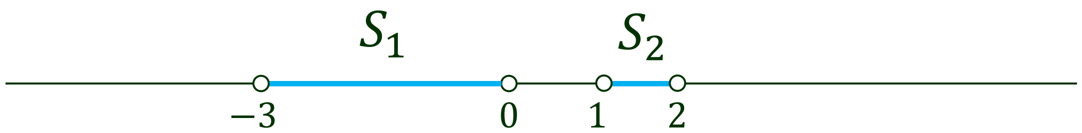
Minkowski Sum of 1D Sets

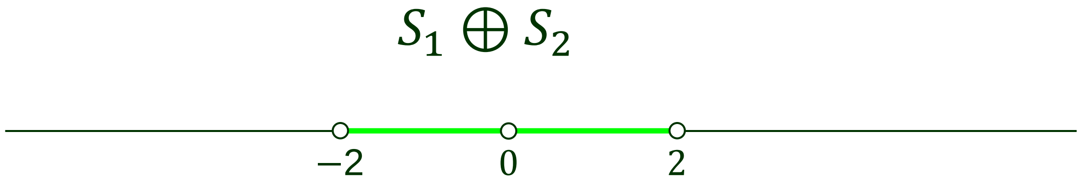
$S_1 = [-3, 0], S_2 = [1, 2]$



Minkowski Sum of 1D Sets

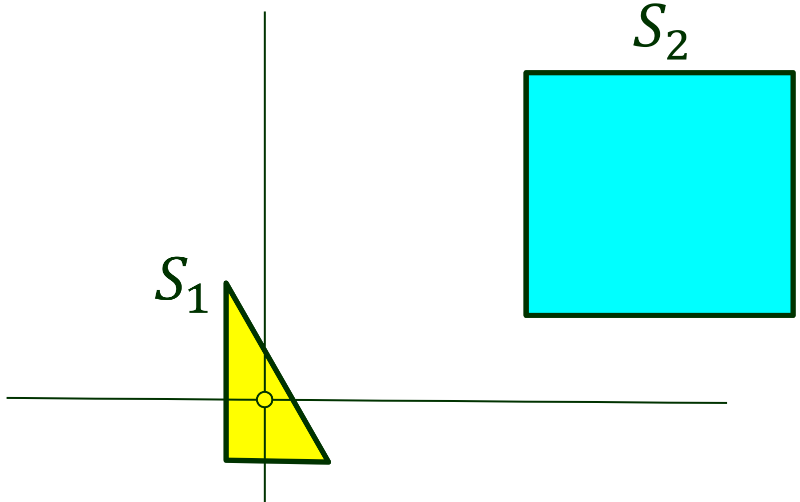
$S_1 = [-3, 0], S_2 = [1, 2]$





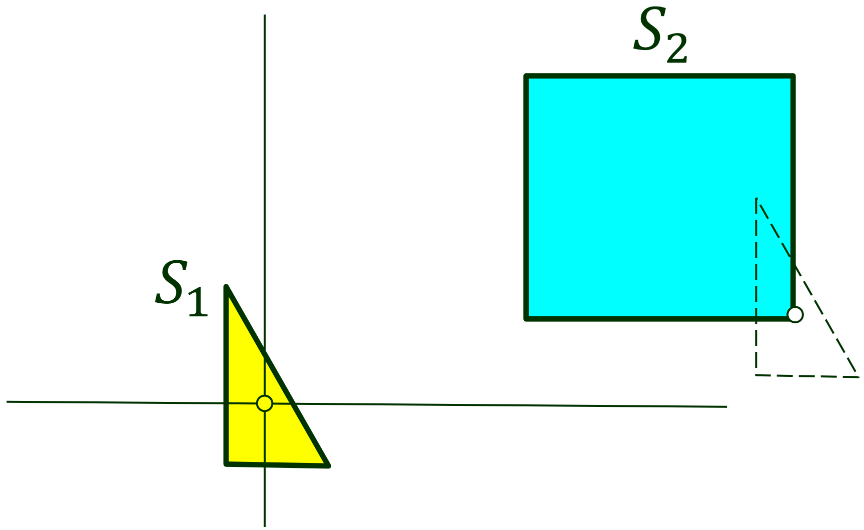
Minkowski Sum of 2D Sets

$$S_1 \oplus S_2 = \{p + q \mid p \in S_1, q \in S_2\}$$



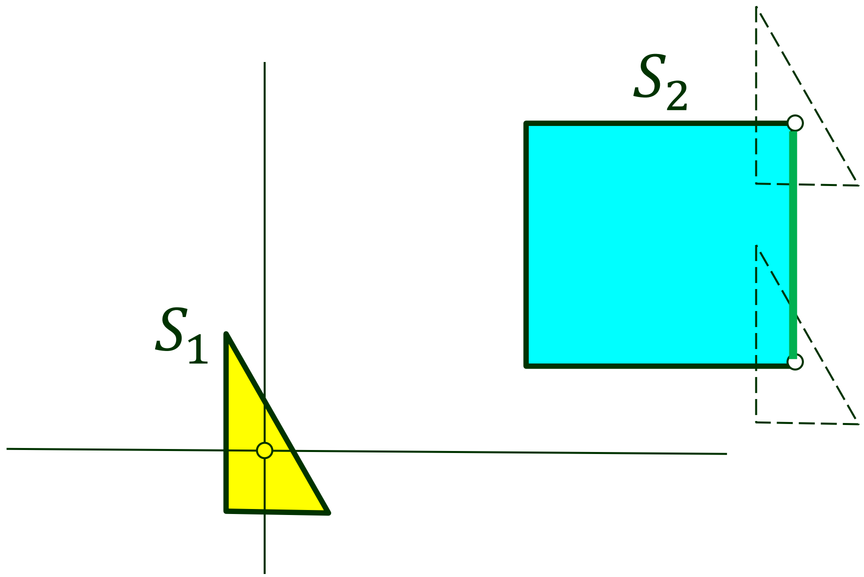
Minkowski Sum of 2D Sets

$$S_1 \oplus S_2 = \{p + q \mid p \in S_1, q \in S_2\}$$



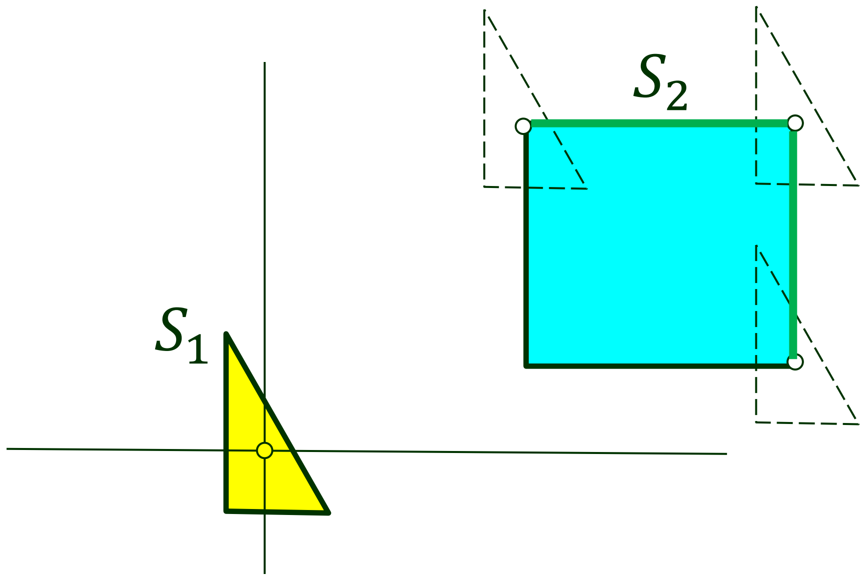
Minkowski Sum of 2D Sets

$$S_1 \oplus S_2 = \{p + q \mid p \in S_1, q \in S_2\}$$



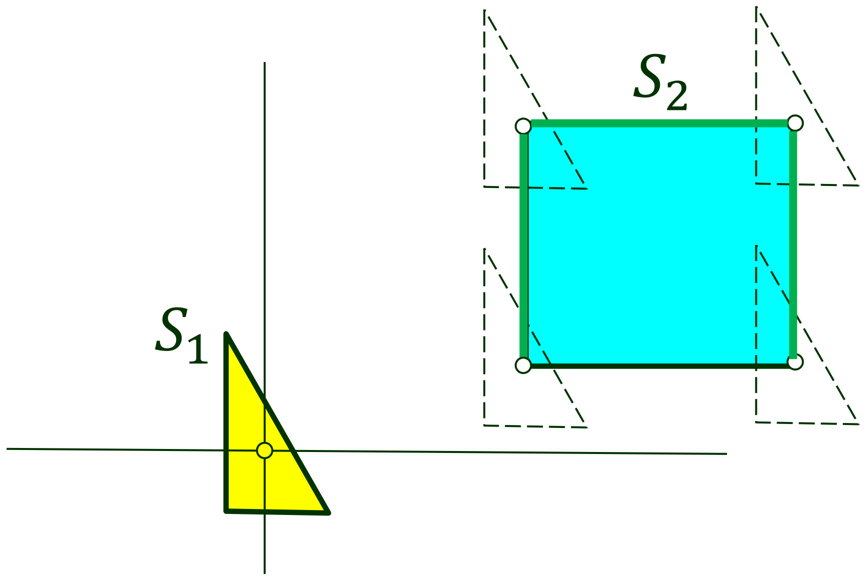
Minkowski Sum of 2D Sets

$$S_1 \oplus S_2 = \{p + q \mid p \in S_1, q \in S_2\}$$



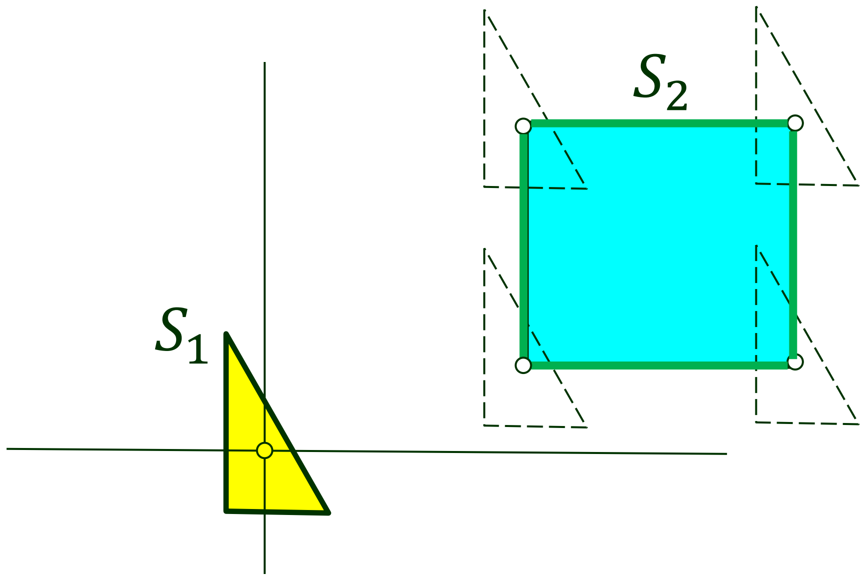
Minkowski Sum of 2D Sets

$$S_1 \oplus S_2 = \{p + q \mid p \in S_1, q \in S_2\}$$



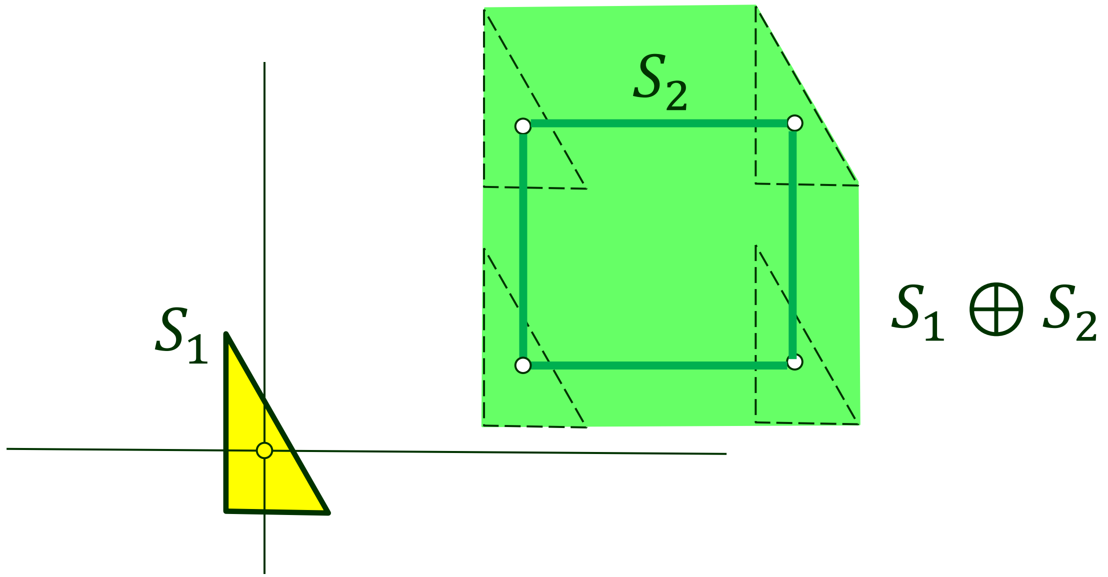
Minkowski Sum of 2D Sets

$$S_1 \oplus S_2 = \{p + q \mid p \in S_1, q \in S_2\}$$



Minkowski Sum of 2D Sets

$$S_1 \oplus S_2 = \{p + q \mid p \in S_1, q \in S_2\}$$



Negation of a Set

$$p = (p_x, p_y) \mapsto -p = (-p_x, -p_y)$$

Negation of a Set

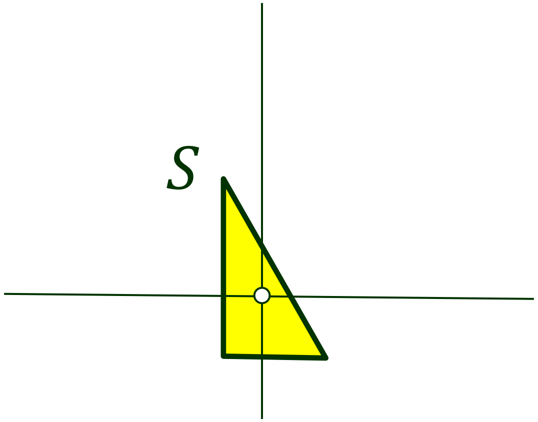
$$p = (p_x, p_y) \mapsto -p = (-p_x, -p_y)$$

A set S:

Negation of a Set

$$p = \left(p_x, p_y\right) \mapsto -p = (-p_x, -p_y)$$

A set S:

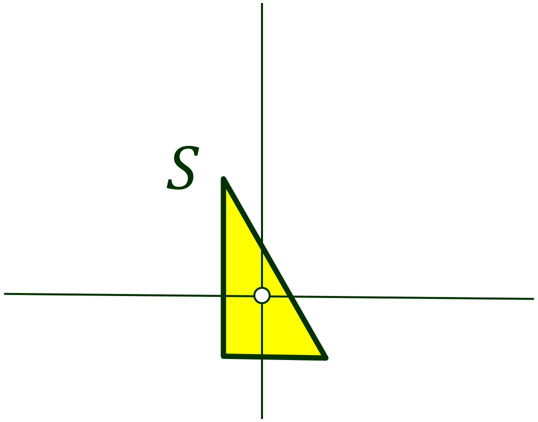


Negation of a Set

$$p = (p_x, p_y) \mapsto -p = (-p_x, -p_y)$$

A set S:

$$-S = \{-p \mid p \in S\}$$

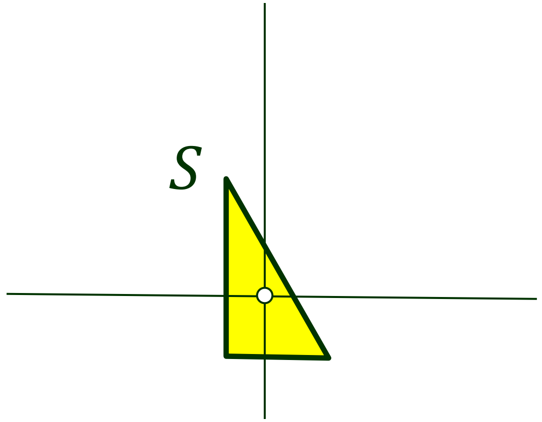


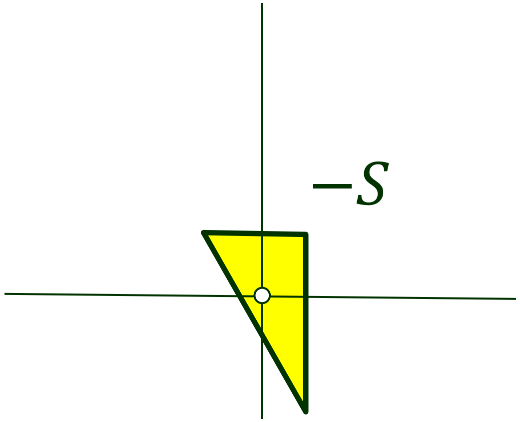
Negation of a Set

$$p = (p_x, p_y) \mapsto -p = (-p_x, -p_y)$$

A set S:

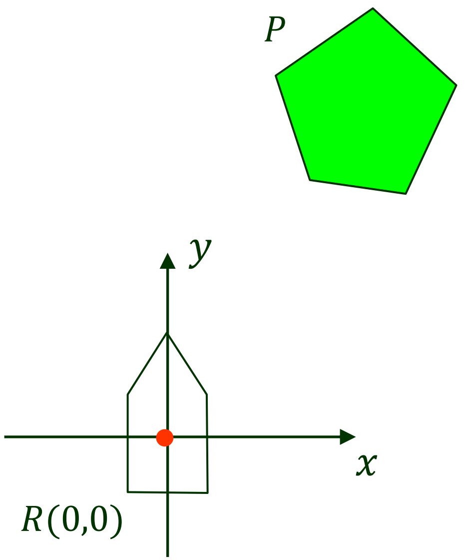
$$-S = \{-p \mid p \in S\}$$





II. Formula for C-obstacle

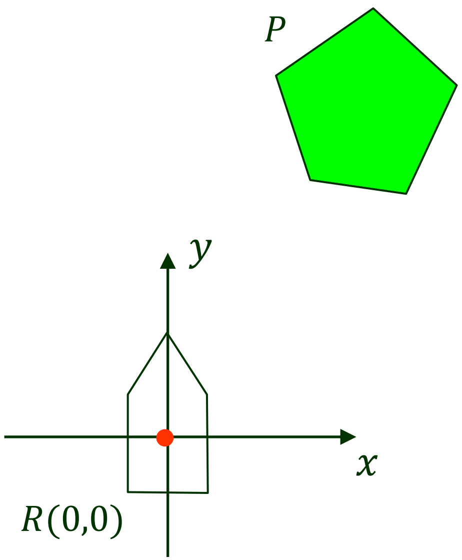
Theorem 1 The C-obstacle CP of P is $P \oplus \left(-R(0,0)\right)$.



II. Formula for C-obstacle

Theorem 1 The C-obstacle CP of P is $P \oplus \left(\underline{-R(0,0)}\right)$.

$$= \{(-x, -y) \mid (x, y) \in R(0,0)\}$$

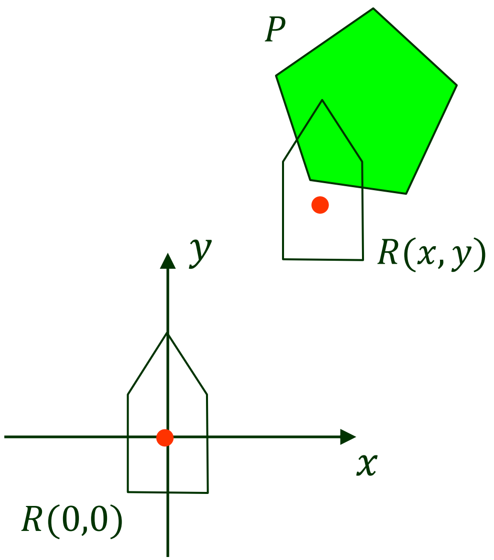


II. Formula for C-obstacle

Theorem 1 The C-obstacle CP of P is $P \oplus \left(\underline{-R(0,0)}\right)$.

$= \{(-x, -y) | \, (x, y) \in R(0, 0)\}$

Proof $(\Rightarrow)$ Suppose $(x, y) \in CP$, i.e., $R(x, y)$ intersects P.



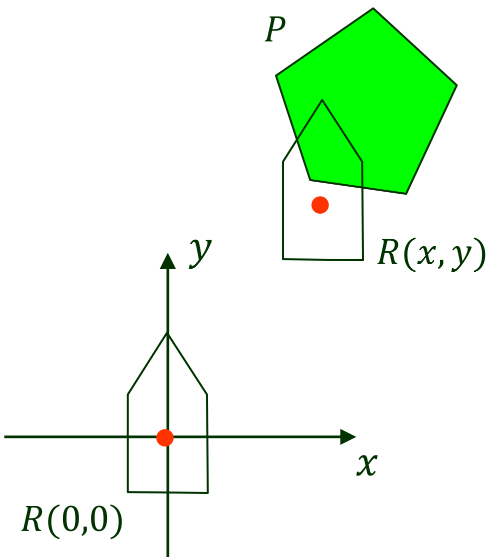
II. Formula for C-obstacle

Theorem 1 The C-obstacle CP of P is $P \oplus \left(\underline{-R(0,0)}\right)$.

$= \{(-x, -y) \mid (x, y) \in R(0,0)\}$

Proof ($\Rightarrow$) Suppose $(x, y) \in CP$, i.e., $R(x, y)$ intersects P.

We need to show $(x, y) \in P \oplus (-R(0,0))$.



II. Formula for C-obstacle

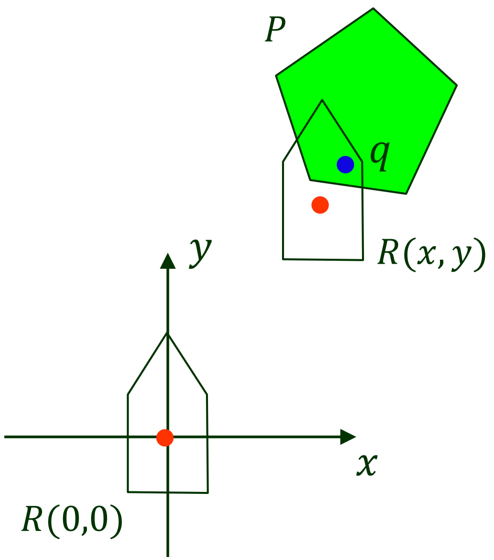
Theorem 1 The C-obstacle CP of P is $P \oplus \left(\underline{-R(0,0)}\right)$.

$= \{(-x, -y) \mid (x, y) \in R(0, 0)\}$

Proof $(\Rightarrow)$ Suppose $(x, y) \in CP$, i.e., $R(x, y)$ intersects P.

We need to show $(x, y) \in P \oplus (-R(0,0))$.

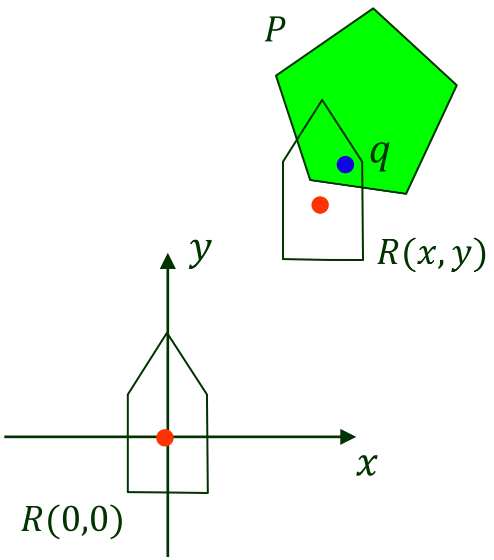
Let $q = (q_x, q_y)$ be a point in the intersection.



II. Formula for C-obstacle

Theorem 1 The C-obstacle CP of P is $P \oplus \left(\underline{-R(0,0)}\right)$.

$$= \{(-x,-y) \mid (x,y) \in R(0,0)\}$$



Proof $(\Rightarrow)$ Suppose $(x,y) \in CP$, i.e., $R(x,y)$ intersects P.

We need to show $(x,y) \in P \oplus (-R(0,0))$.

Let $q = (q_x, q_y)$ be a point in the intersection.

$q \in R(x,y)$

II. Formula for C-obstacle

Theorem 1 The C-obstacle CP of P is $P \oplus \left(\underline{-R(0,0)}\right)$.

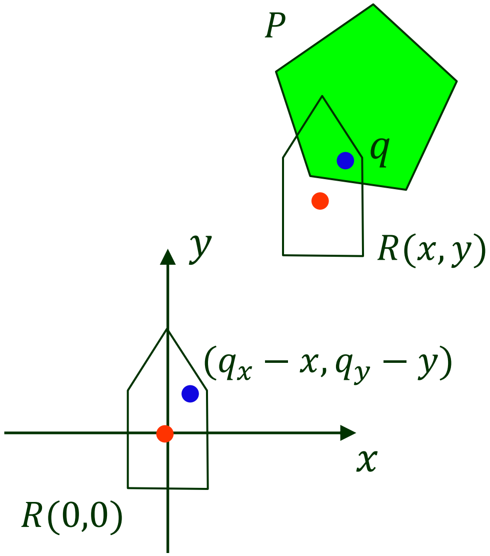
$$= \{(-x, -y) \mid (x, y) \in R(0, 0)\}$$

Proof ($\Rightarrow$) Suppose $(x, y) \in CP$, i.e., $R(x, y)$ intersects P.

We need to show $(x, y) \in P \oplus (-R(0,0))$.

Let $q = (q_x, q_y)$ be a point in the intersection.

$q \in R(x, y) \implies (q_x - x, q_y - y) \in R(0,0)$



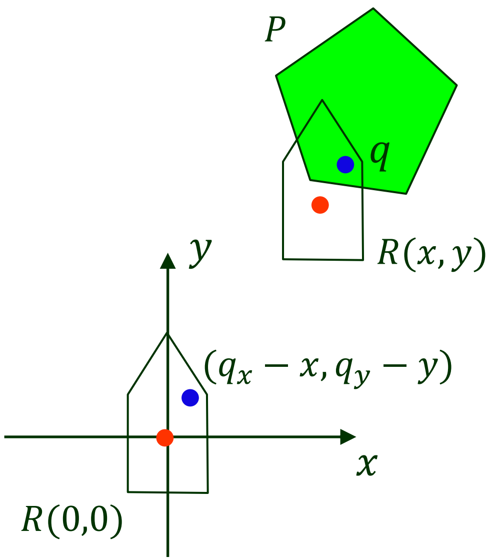
II. Formula for C-obstacle

Theorem 1 The C-obstacle CP of P is $P \oplus \left(\underline{-R(0,0)}\right)$.

$= \{(-x, -y) \mid (x, y) \in R(0, 0)\}$

Proof $(\Rightarrow)$ Suppose $(x, y) \in CP$, i.e., $R(x, y)$ intersects P.

We need to show $(x, y) \in P \oplus (-R(0,0))$.



Let $q = (q_x, q_y)$ be a point in the intersection.

$q \in R(x, y) \implies (q_x - x, q_y - y) \in R(0,0)$

$\implies \left(-q_x + x, -q_y + y\right) \in -R(0,0)$

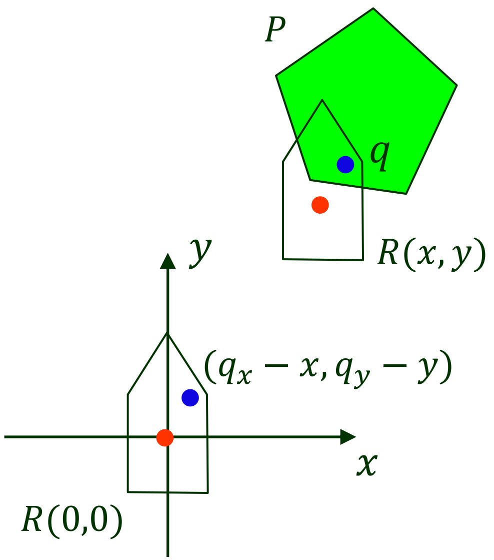
II. Formula for C-obstacle

Theorem 1 The C-obstacle CP of P is $P \oplus \left(\underline{-R(0,0)}\right)$.

$$= \{(-x, -y) \mid (x, y) \in R(0, 0)\}$$

Proof ($\Rightarrow$) Suppose $(x, y) \in CP$, i.e., $R(x, y)$ intersects P.

We need to show $(x, y) \in P \oplus (-R(0,0))$.



Let $q = (q_x, q_y)$ be a point in the intersection.

$$q \in R(x, y) \implies (q_x - x, q_y - y) \in R(0,0)$$

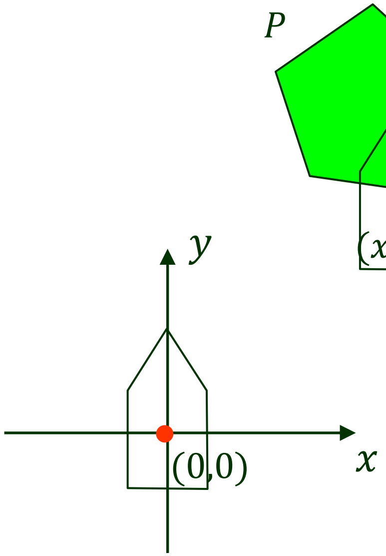
$$\implies \left(-q_x + x, -q_y + y\right) \in -R(0,0)$$

$$q \in P \quad \Downarrow$$

$$q + \left(-q_x + x, -q_y + y\right) = (x, y) \in P \oplus (-R(0,0))$$

Proof (cont'd)

(⇐) Let $(x, y) \in P \oplus (-R(0,0))$.

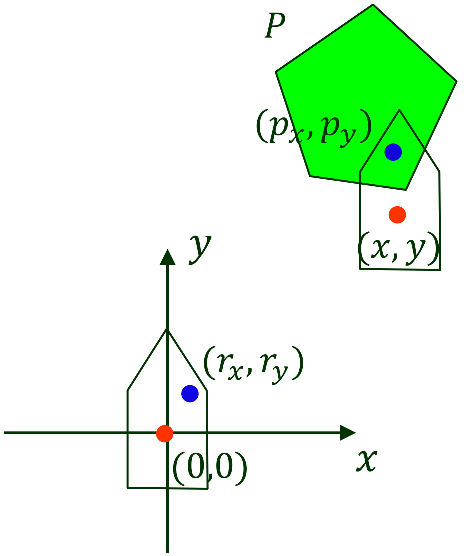


Proof (cont'd)

($\Leftarrow$) Let $(x, y) \in P \oplus (-R(0,0))$.

There exists $(r_x, r_y) \in R(0,0)$ and $(p_x, p_y) \in P$ such that

$$(x, y) = (p_x - r_x, p_y - r_y)$$



Proof (cont'd)

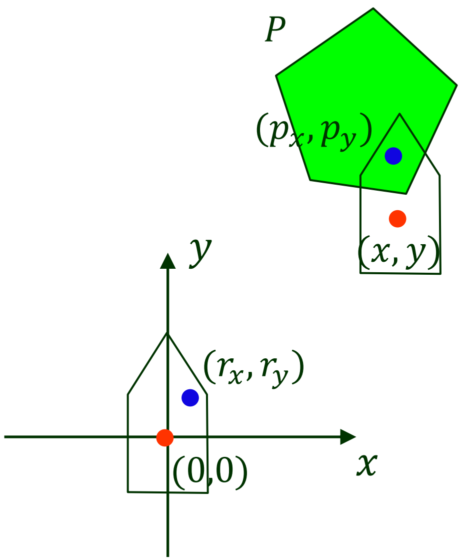
($\Leftarrow$) Let $(x, y) \in P \oplus (-R(0,0))$.

There exists $(r_x, r_y) \in R(0,0)$ and $(p_x, p_y) \in P$ such that

$$(x, y) = (p_x - r_x, p_y - r_y)$$

$$\Downarrow$$

$p_x = r_x + x$ and $p_y = r_y + y$



Proof (cont'd)

$(\Leftarrow)$ Let $(x, y) \in P \oplus (-R(0,0))$.

There exists $(r_x, r_y) \in R(0,0)$ and $(p_x, p_y) \in P$ such that

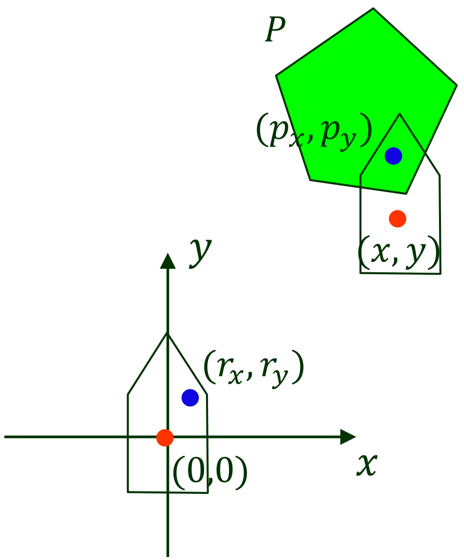
$$(x, y) = (p_x - r_x, p_y - r_y)$$

$$\Downarrow$$

$$p_x = r_x + x \text{ and } p_y = r_y + y$$

$$\Downarrow \; (r_x, r_y) \in R(0,0)$$

$$(p_x, p_y) \in R(x, y)$$



Proof (cont'd)

$(\Leftarrow)$ Let $(x,y) \in P \oplus (-R(0,0))$.

There exists $(r_x, r_y) \in R(0,0)$ and $(p_x, p_y) \in P$ such that

$$(x,y) = (p_x - r_x, p_y - r_y)$$

$$\Downarrow$$

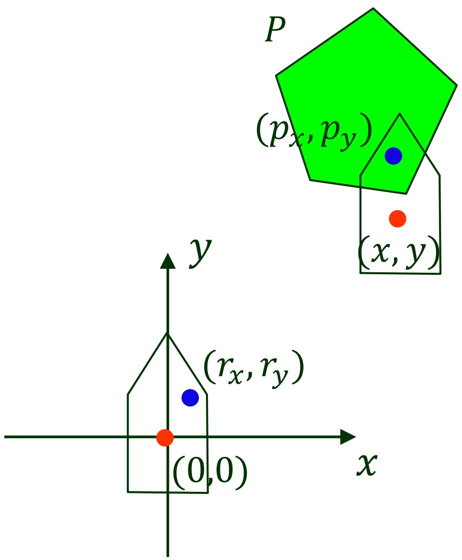
$$p_x = r_x + x \text{ and } p_y = r_y + y$$

$$\Downarrow \; (r_x, r_y) \in R(0,0)$$

$$(p_x, p_y) \in R(x,y)$$

$$\Downarrow \; (p_x, p_y) \in P$$

$$(p_x, p_y) \in P \cap R(x,y)$$



Proof (cont'd)

($\Leftarrow$) Let $(x, y) \in P \oplus (-R(0,0))$.

There exists $(r_x, r_y) \in R(0,0)$ and $(p_x, p_y) \in P$ such that

$$(x, y) = (p_x - r_x, p_y - r_y)$$

$$\Downarrow$$

$$p_x = r_x + x \text{ and } p_y = r_y + y$$

$$\Downarrow \quad (r_x, r_y) \in R(0,0)$$

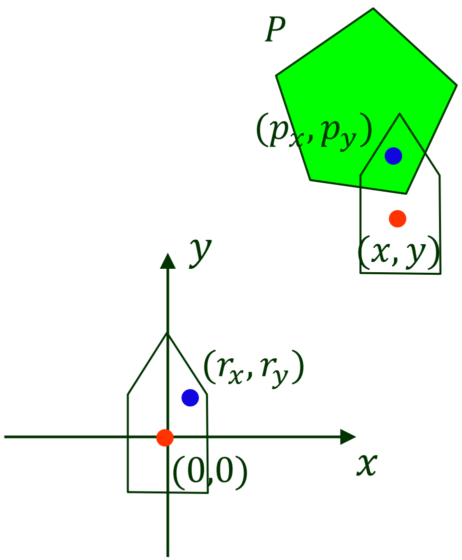
$$(p_x, p_y) \in R(x, y)$$

$$\Downarrow \quad (p_x, p_y) \in P$$

$$(p_x, p_y) \in P \cap R(x, y)$$

$$\Downarrow$$

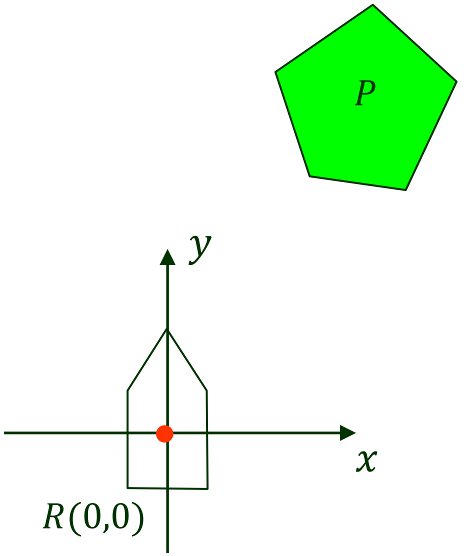
$R(x, y)$ intersects P, i.e., $(x, y) \in CP$.



Verification via an Example

Two equivalent ways of C-obstacle construction:

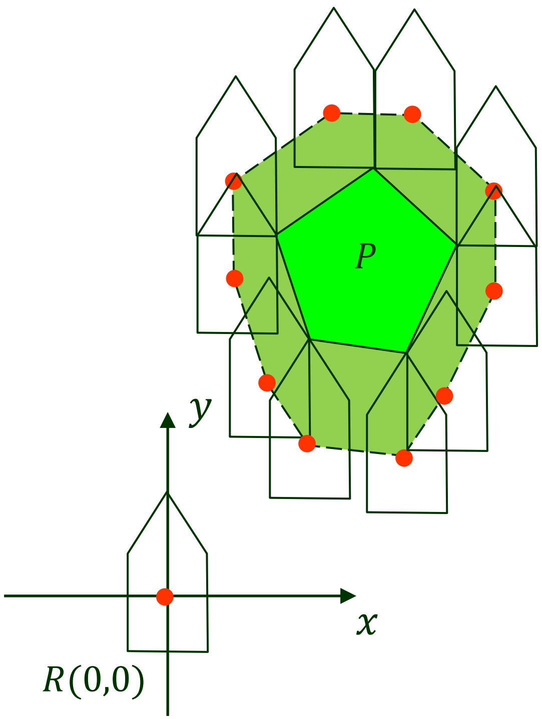
Straightforward



Verification via an Example

Two equivalent ways of C-obstacle construction:

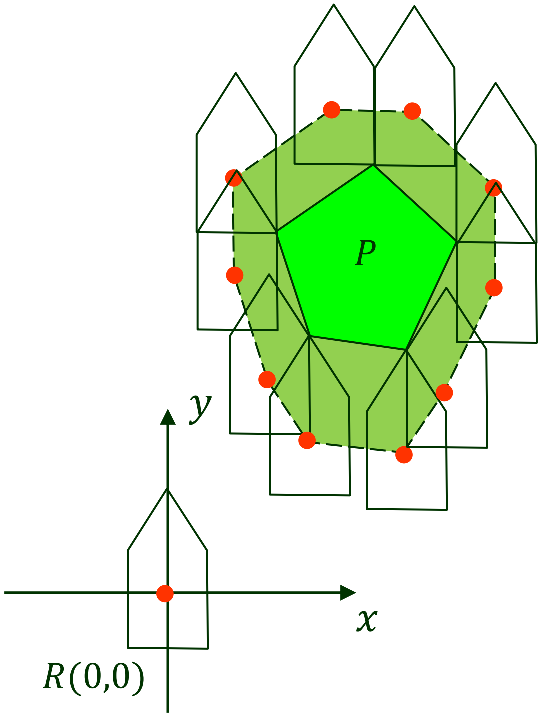
Straightforward



Verification via an Example

Two equivalent ways of C-obstacle construction:

Straightforward



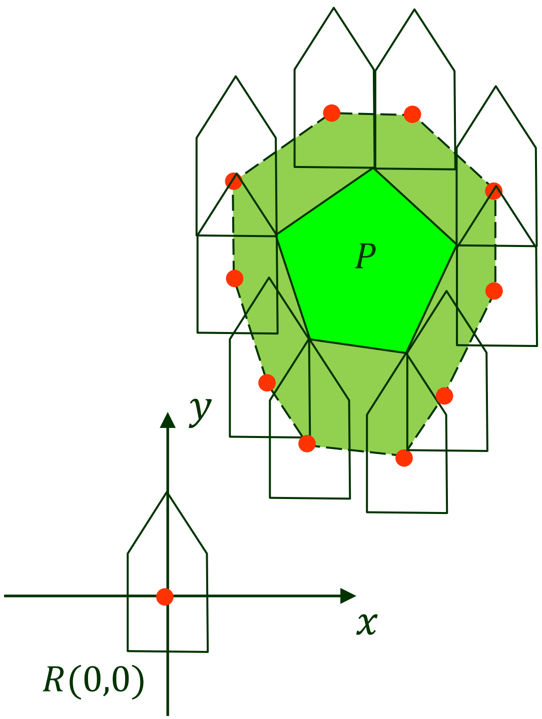
via Minkowski sum

$$P \oplus (-R(0,0))$$

Verification via an Example

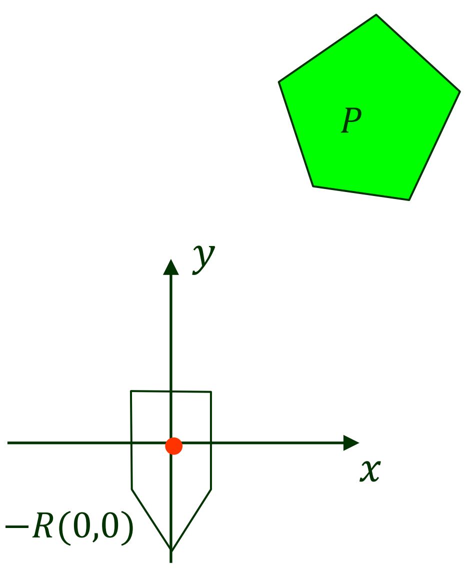
Two equivalent ways of C-obstacle construction:

Straightforward



via Minkowski sum

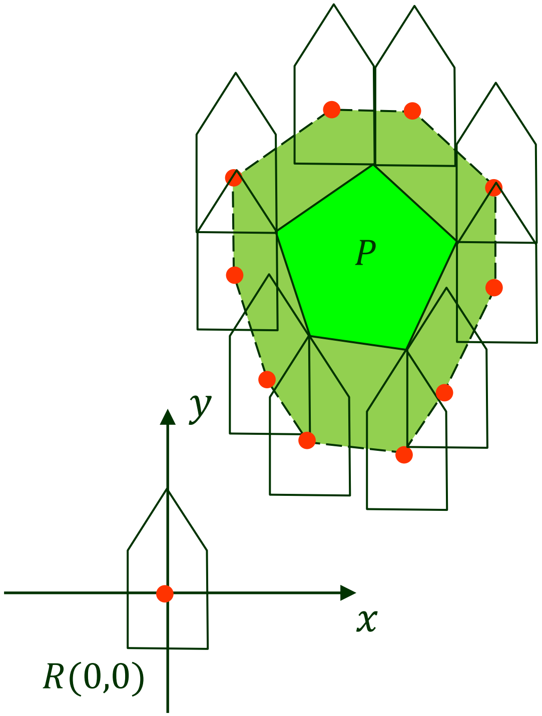
$$P \oplus (-R(0,0))$$



Verification via an Example

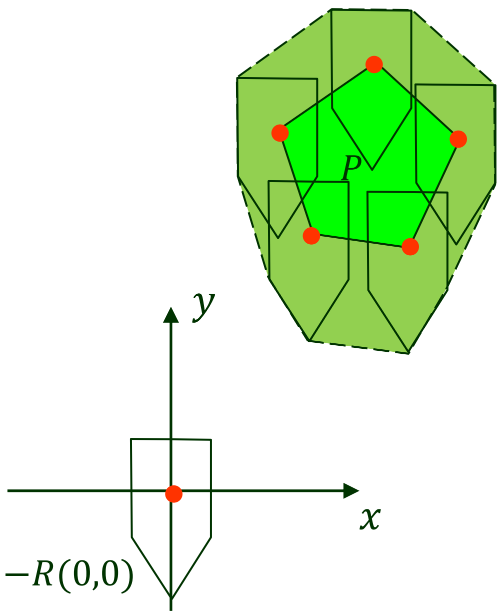
Two equivalent ways of C-obstacle construction:

Straightforward



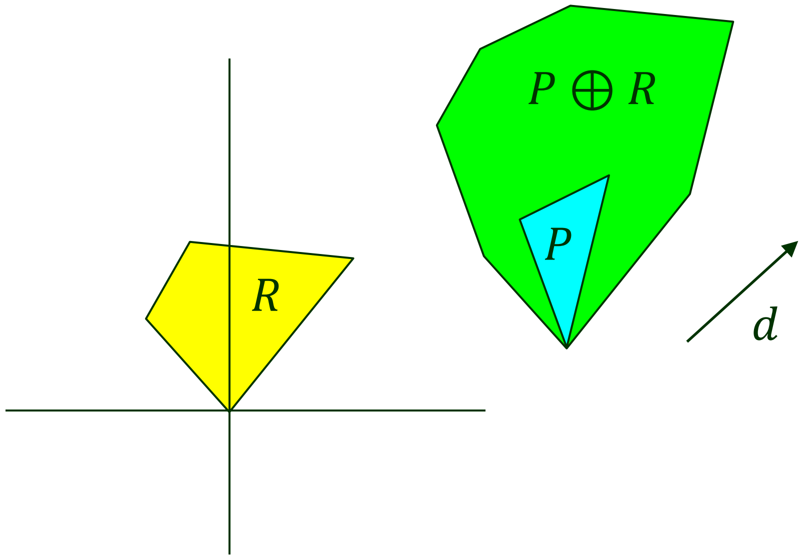
via Minkowski sum

$$P \oplus (-R(0,0))$$



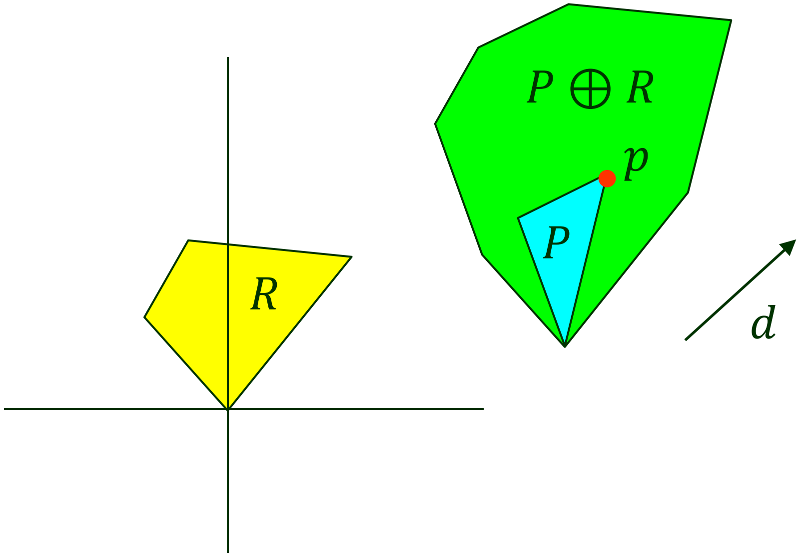
III. Extreme Points

Two polygons P and R.



III. Extreme Points

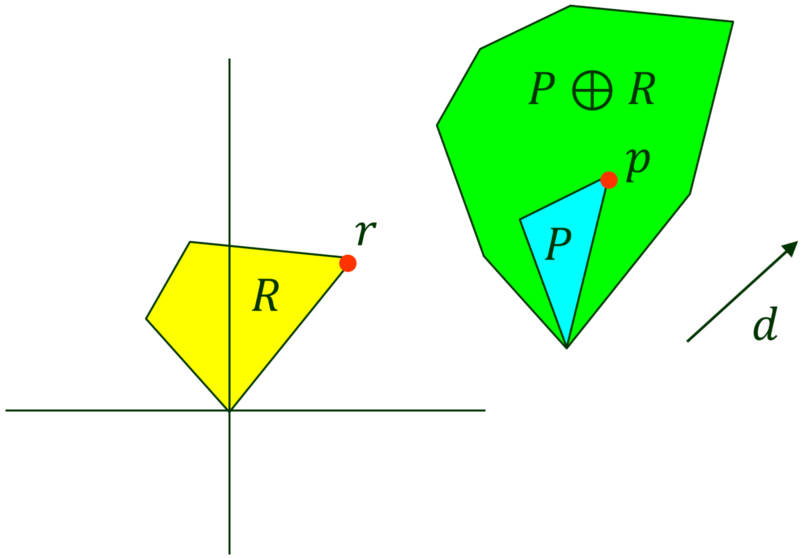
Two polygons P and R.



p: an *extreme point* on P
in the direction d, that is,
$p \cdot d \geq q \cdot d$ for any $q \in P$.

III. Extreme Points

Two polygons P and R.

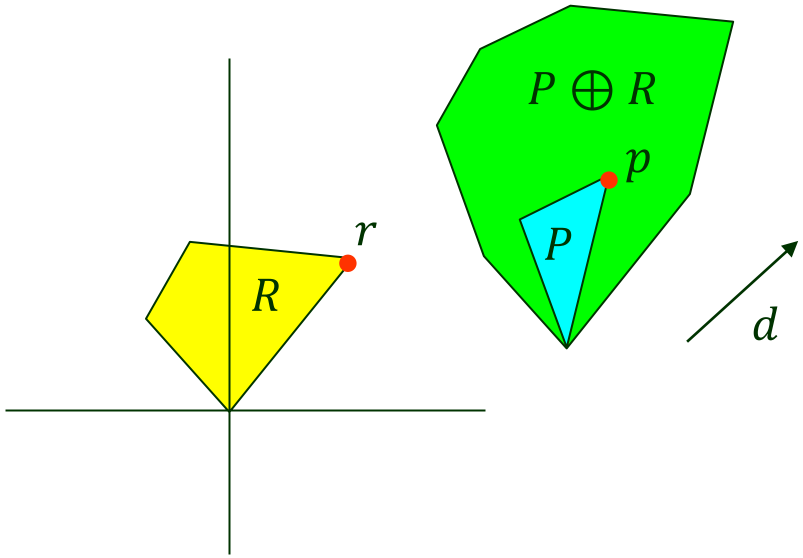


p: an *extreme point* on P
in the direction d, that is,
$p \cdot d \geq q \cdot d$ for any $q \in P$.

r: an extreme point on R
in the direction d, that is,
$r \cdot d \geq t \cdot d$ for any $t \in R$.

III. Extreme Points

Two polygons P and R.



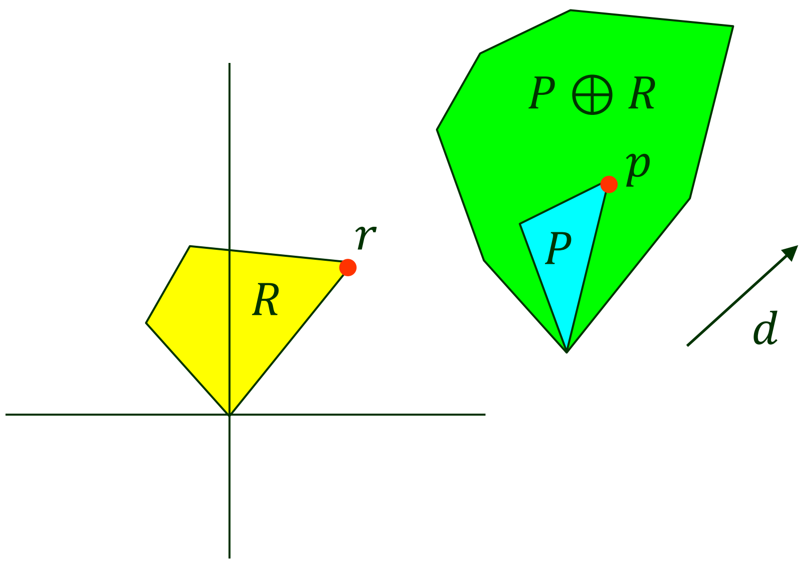
p: an *extreme point* on P
in the direction d, that is,
$p \cdot d \geq q \cdot d$ for any $q \in P$.

r: an extreme point on R
in the direction d, that is,
$r \cdot d \geq t \cdot d$ for any $t \in R$.

Any point $s \in P \oplus R$ has $s = q + t$ for some $q \in P$ and $t \in R$.

III. Extreme Points

Two polygons P and R.



p: an *extreme point* on P
in the direction d, that is,
$p \cdot d \geq q \cdot d$ for any $q \in P$.

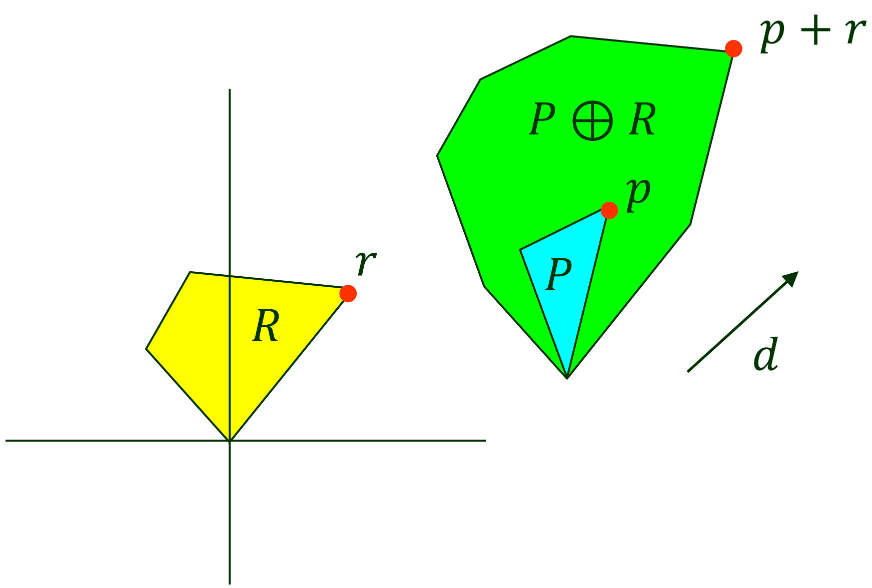
r: an extreme point on R
in the direction d, that is,
$r \cdot d \geq t \cdot d$ for any $t \in R$.

Any point $s \in P \oplus R$ has $s = q + t$ for some $q \in P$ and $t \in R$.

$$s \cdot d = (q + t) \cdot d \leq (p + r) \cdot d$$

III. Extreme Points

Two polygons P and R.



p: an *extreme point* on P
in the direction d, that is,
$p \cdot d \geq q \cdot d$ for any $q \in P$.

r: an extreme point on R
in the direction d, that is,
$r \cdot d \geq t \cdot d$ for any $t \in R$.

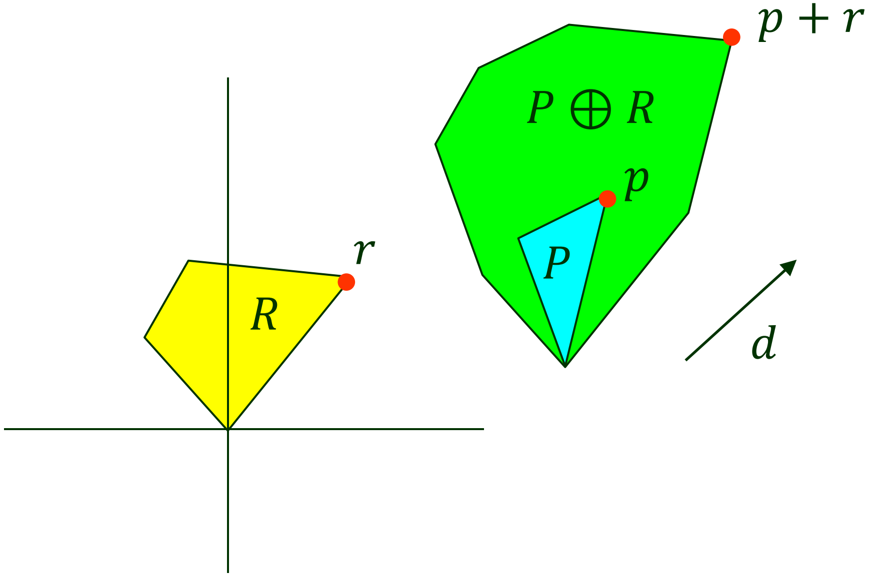
Any point $s \in P \oplus R$ has $s = q + t$ for some $q \in P$ and $t \in R$.

$$s \cdot d = (q + t) \cdot d \leq (p + r) \cdot d$$

$p + r$ is an extreme point in the direction d on $P \oplus R$.

III. Extreme Points

Two polygons P and R.



p: an *extreme point* on P in the direction d, that is, $p \cdot d \geq q \cdot d$ for any $q \in P$.

r: an extreme point on R in the direction d, that is, $r \cdot d \geq t \cdot d$ for any $t \in R$.

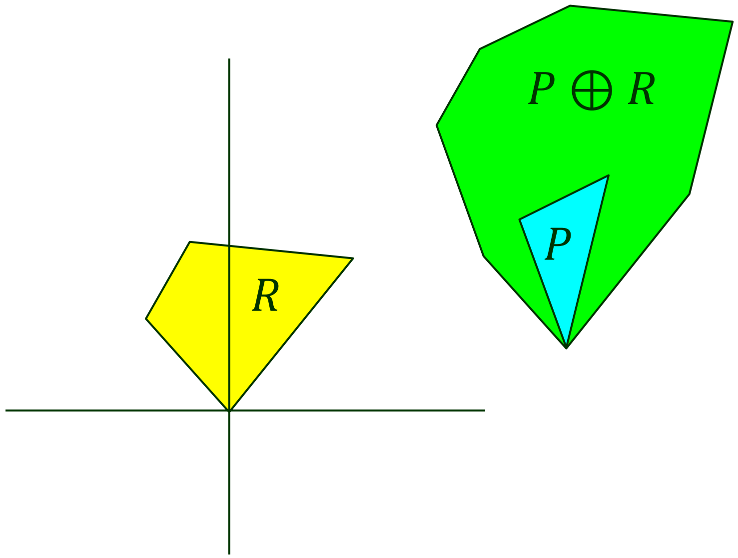
Any point $s \in P \oplus R$ has $s = q + t$ for some $q \in P$ and $t \in R$.

$$s \cdot d = (q+t) \cdot d \leq (p+r) \cdot d$$

$p + r$ is an extreme point in the direction d on $P \oplus R$.

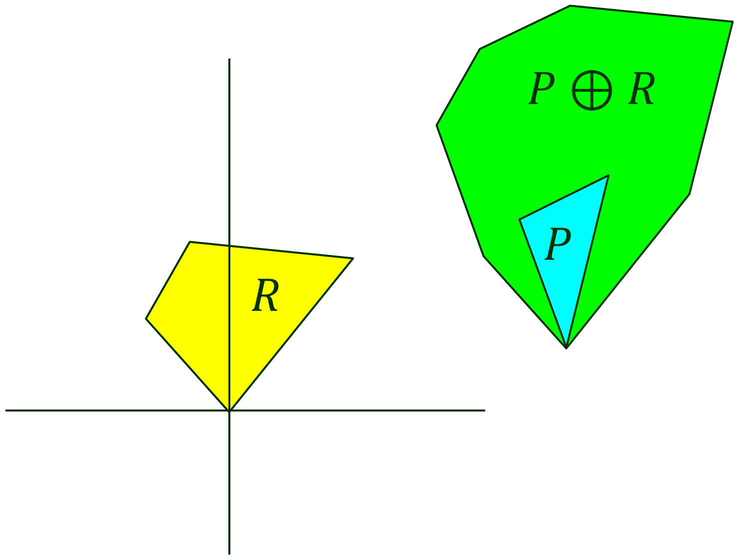
An extreme point in the direction d on $P \oplus R$ is the sum of two extreme points in d on P and R, respectively.

Minkowski Sum of Convex Polygons



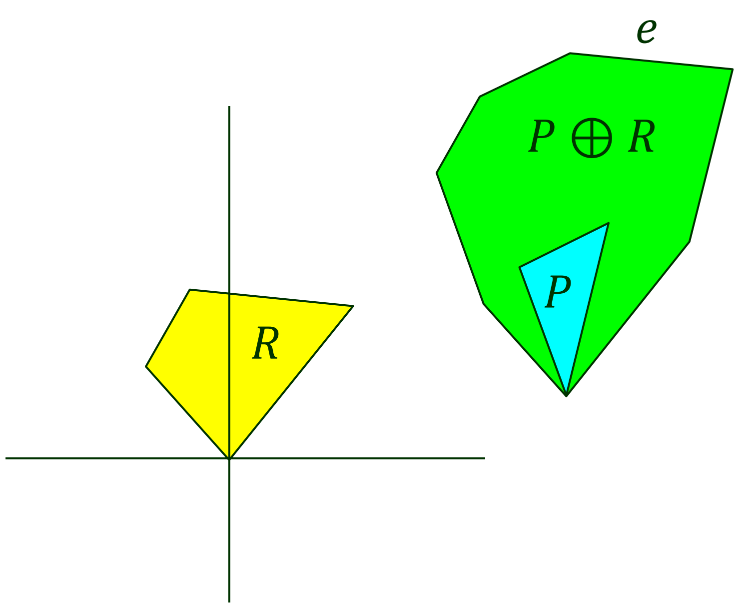
Theorem 2 Let P and R be two convex polygons with n and m edges, respectively. Then $P \oplus R$ is a convex polygon with $\leq n + m$ edges.

Complexity of Minkowski Sum



Proof Convexity of the Minkowski sum of two convex sets follows from the definition.

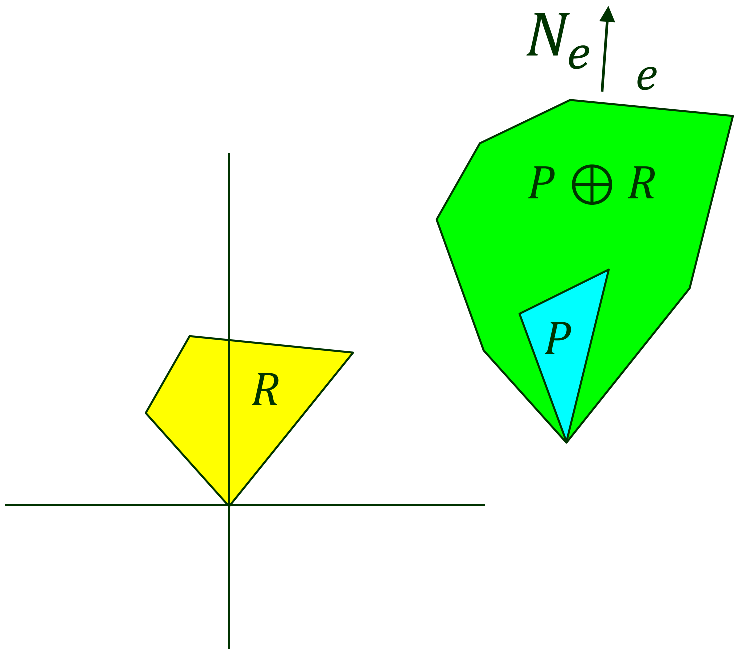
Complexity of Minkowski Sum



Proof Convexity of the Minkowski sum of two convex sets follows from the definition.

To bound the complexity, consider an edge e of $P \oplus R$.

Complexity of Minkowski Sum

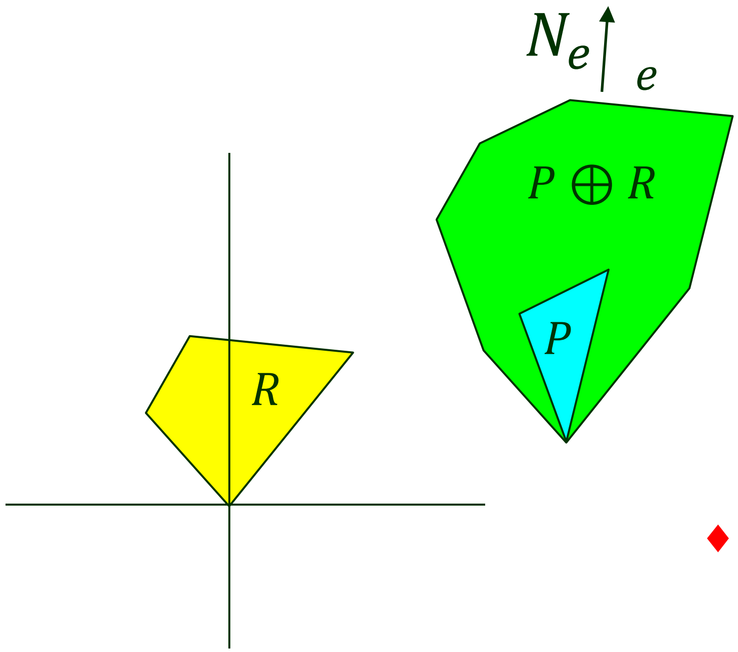


Proof Convexity of the Minkowski sum of two convex sets follows from the definition.

To bound the complexity, consider an edge e of $P \oplus R$.

N_e: outward normal of e.

Complexity of Minkowski Sum



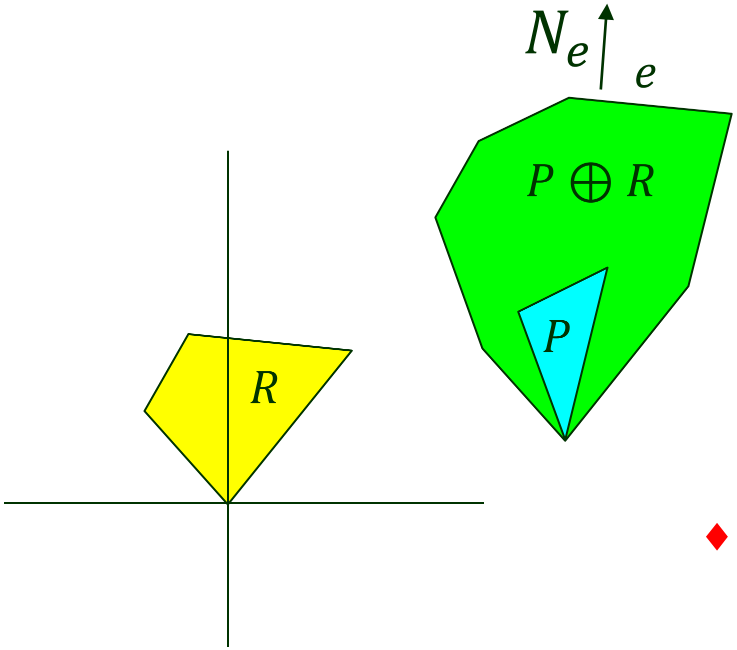
Proof Convexity of the Minkowski sum of two convex sets follows from the definition.

To bound the complexity, consider an edge e of $P \oplus R$.

N_e: outward normal of e.

- e must be generated by points on P and R that are extreme in N_e.

Complexity of Minkowski Sum



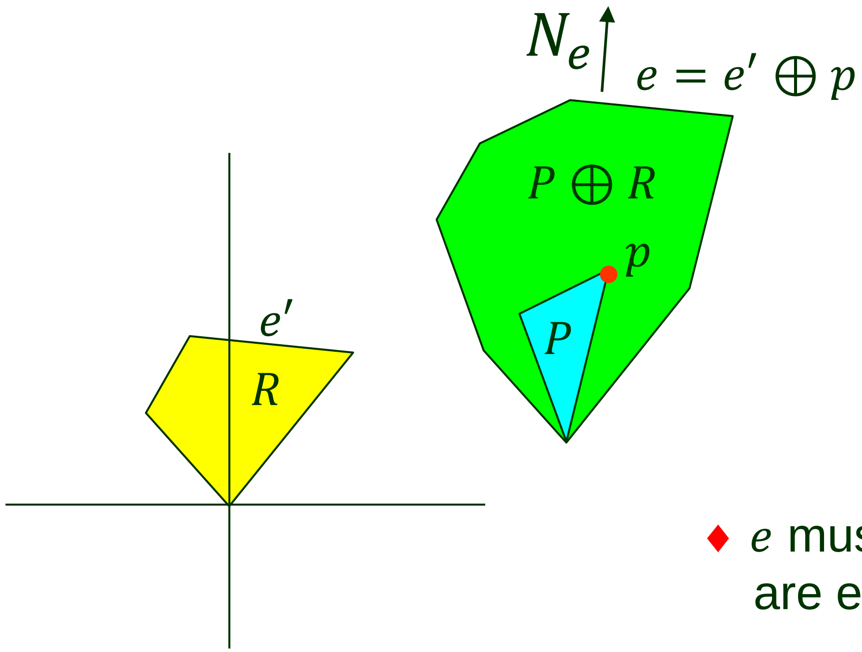
Proof Convexity of the Minkowski sum of two convex sets follows from the definition.

To bound the complexity, consider an edge e of $P \oplus R$.

N_e: outward normal of e.

- e must be generated by points on P and R that are extreme in N_e.
- At least one of P and R must have an edge extreme in N_e.

Complexity of Minkowski Sum



Proof Convexity of the Minkowski sum of two convex sets follows from the definition.

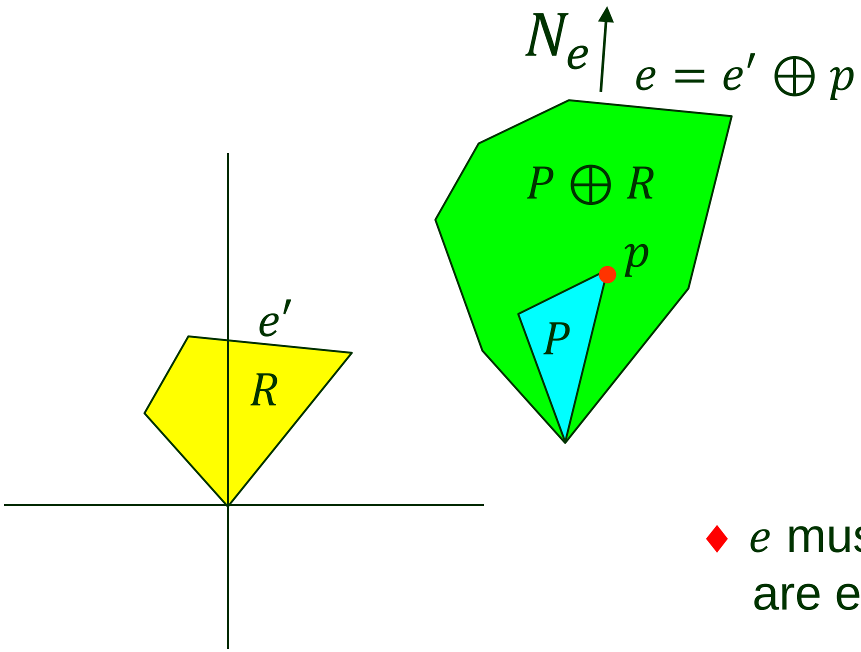
To bound the complexity, consider an edge e of $P \oplus R$.

N_e: outward normal of e.

- e must be generated by points on P and R that are extreme in N_e.

- At least one of P and R must have an edge extreme in N_e.
- Without loss of generality, this edge is, say, e' on R.

Complexity of Minkowski Sum



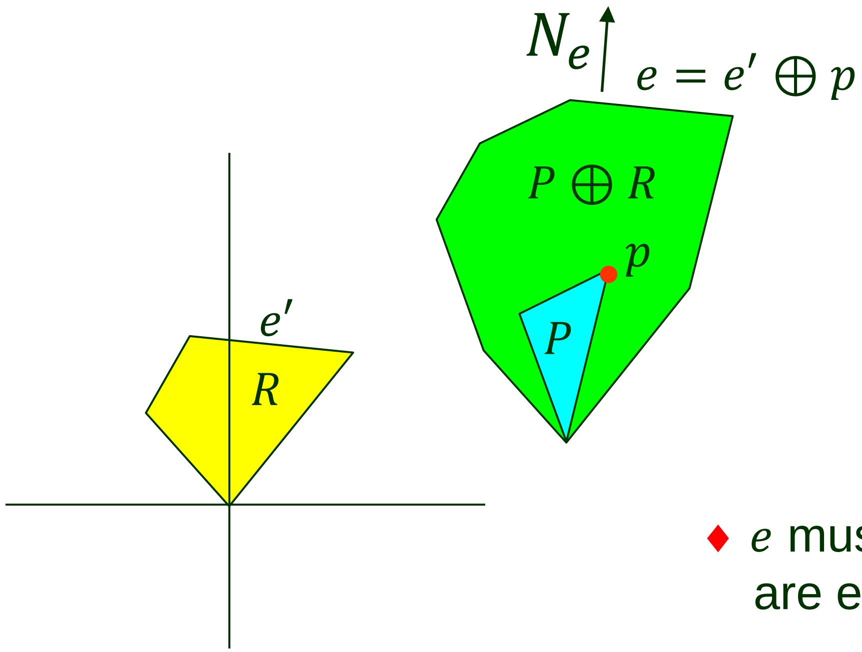
Proof Convexity of the Minkowski sum of two convex sets follows from the definition.

To bound the complexity, consider an edge e of $P \oplus R$.

N_e: outward normal of e.

- e must be generated by points on P and R that are extreme in N_e.
- At least one of P and R must have an edge extreme in N_e.
- Without loss of generality, this edge is, say, e' on R.
- Charge e to e'.

Complexity of Minkowski Sum



Proof Convexity of the Minkowski sum of two convex sets follows from the definition.

To bound the complexity, consider an edge e of $P \oplus R$.

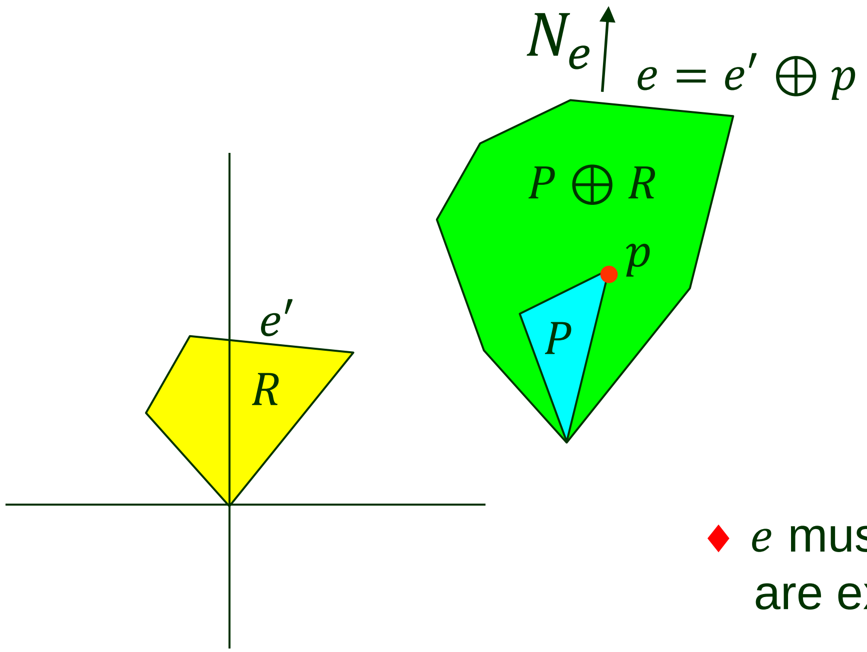
N_e: outward normal of e.

- e must be generated by points on P and R that are extreme in N_e.

- At least one of P and R must have an edge extreme in N_e.
- Without loss of generality, this edge is, say, e' on R.
- Charge e to e'.

Every edge of P and R is charged at most once.

Complexity of Minkowski Sum



Proof Convexity of the Minkowski sum of two convex sets follows from the definition.

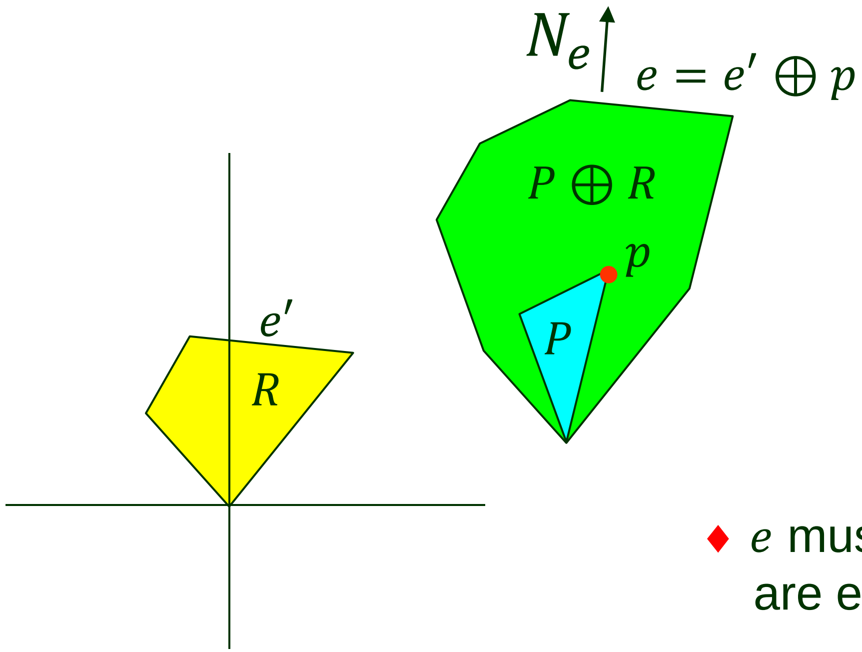
To bound the complexity, consider an edge e of $P \oplus R$.

N_e: outward normal of e.

- e must be generated by points on P and R that are extreme in N_e.
- At least one of P and R must have an edge extreme in N_e.
- Without loss of generality, this edge is, say, e' on R.
- Charge e to e'.

Every edge of P and R is charged at most once. $\Longrightarrow$ # edges of $P \oplus R \leq n + m$.

Complexity of Minkowski Sum



Proof Convexity of the Minkowski sum of two convex sets follows from the definition.

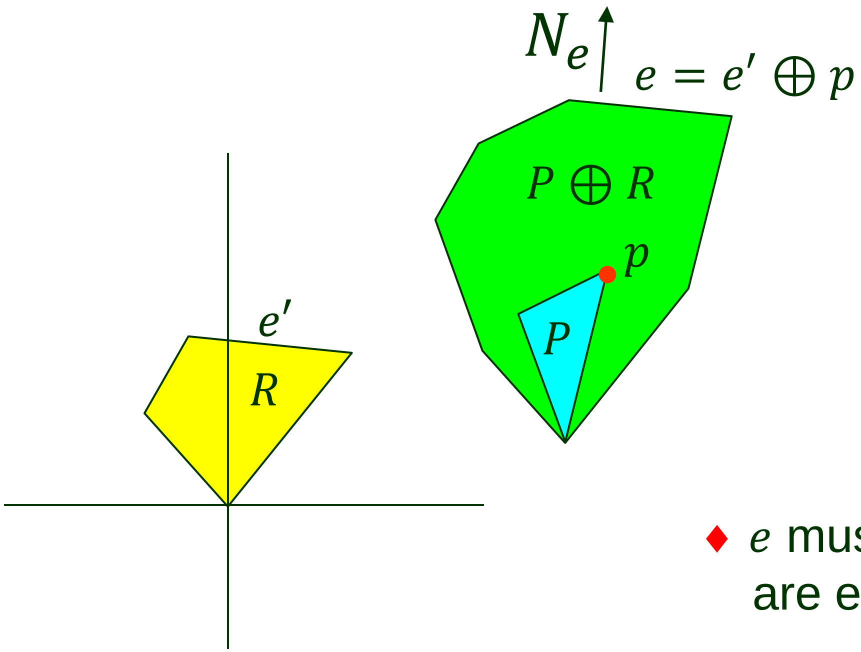
To bound the complexity, consider an edge e of $P \oplus R$.

N_e: outward normal of e.

- e must be generated by points on P and R that are extreme in N_e.
- At least one of P and R must have an edge extreme in N_e.
- Without loss of generality, this edge is, say, e' on R.
- Charge e to e'.

Every edge of P and R is charged at most once. $\Longrightarrow$ # edges of $P \oplus R \le n + m$.

Complexity of Minkowski Sum



Proof Convexity of the Minkowski sum of two convex sets follows from the definition.

To bound the complexity, consider an edge e of $P \oplus R$.

N_e: outward normal of e.

- e must be generated by points on P and R that are extreme in N_e.
- At least one of P and R must have an edge extreme in N_e.
- Without loss of generality, this edge is, say, e' on R.
- Charge e to e'.

Every edge of P and R is charged at most once. $\Longrightarrow$ # edges of $P \oplus R \leq n + m$.

The upper bound $n + m$ is achieved if P and R have no parallel edges.

IV. Computation of the Minkowski Sum

Compute $P \oplus R$ when P and R are convex.

IV. Computation of the Minkowski Sum

Compute $P \oplus R$ when P and R are convex.

Brute-force algorithm

IV. Computation of the Minkowski Sum

Compute $P \oplus R$ when P and R are convex.

Brute-force algorithm

- Compute $v + w$ for each pair (v, w) of vertices with $v \in P$ and $w \in R$.

IV. Computation of the Minkowski Sum

Compute $P \oplus R$ when P and R are convex.

Brute-force algorithm

- Compute $v + w$ for each pair (v, w) of vertices with $v \in P$ and $w \in R$.

- Construct the convex hull of all the sum vertices.

IV. Computation of the Minkowski Sum

Compute $P \oplus R$ when P and R are convex.

Brute-force algorithm

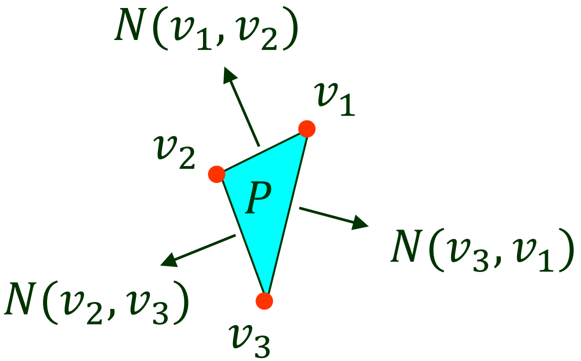
- Compute $v + w$ for each pair (v, w) of vertices with $v \in P$ and $w \in R$.

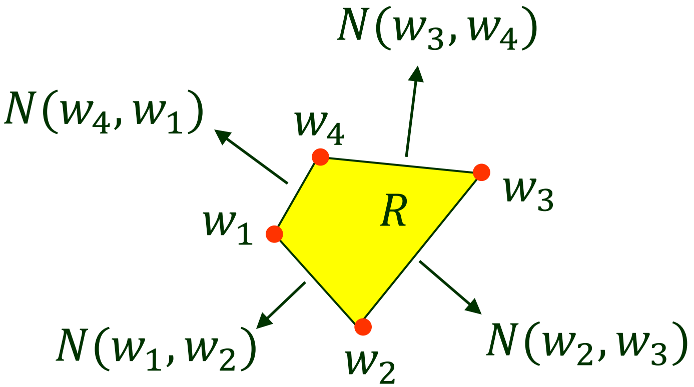
- Construct the convex hull of all the sum vertices.

$$O(nm(\log n + \log m))$$

Faster Computation

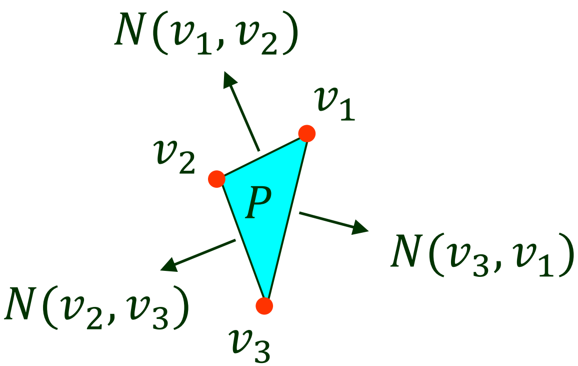
Idea: Look at a pair of vertices that are extreme in the same direction.

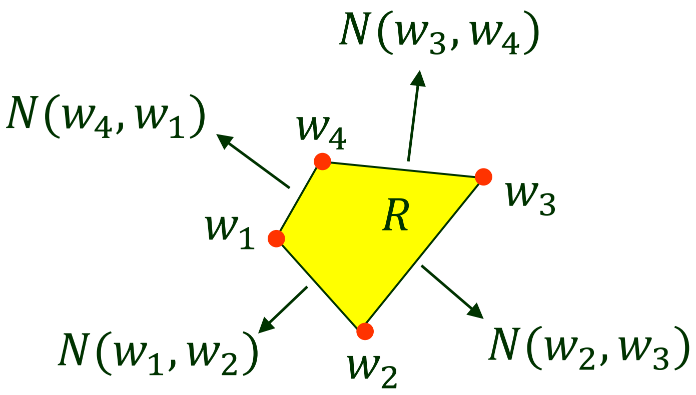




Faster Computation

Idea: Look at a pair of vertices that are extreme in the same direction.

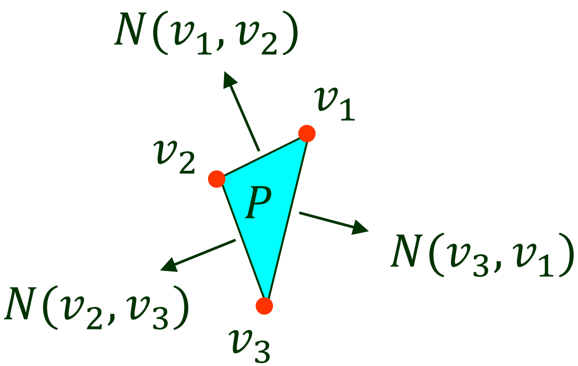


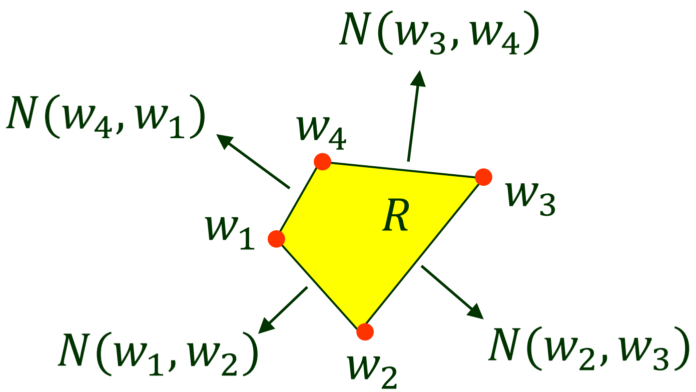


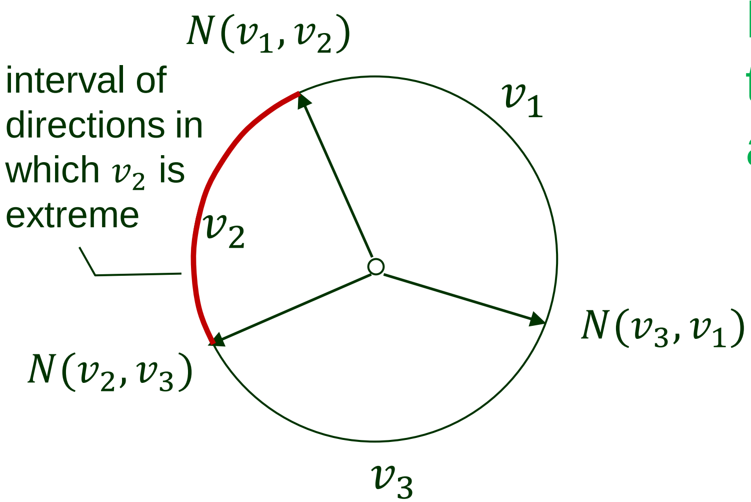
Represent all the directions by a unit circle.

Faster Computation

Idea: Look at a pair of vertices that are extreme in the same direction.



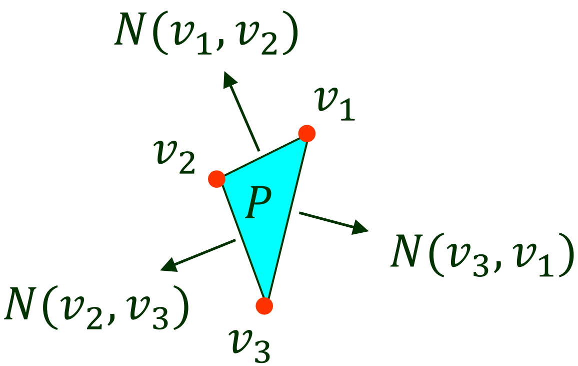


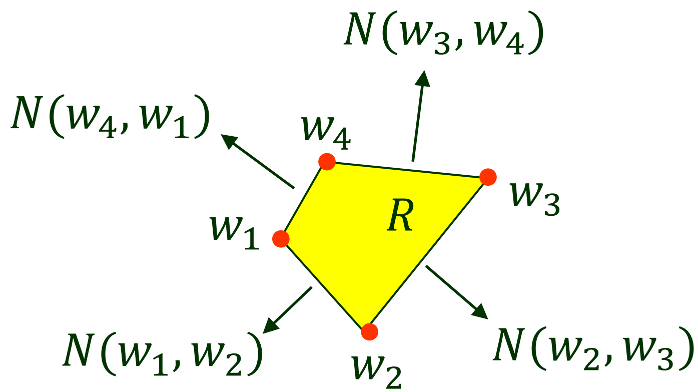


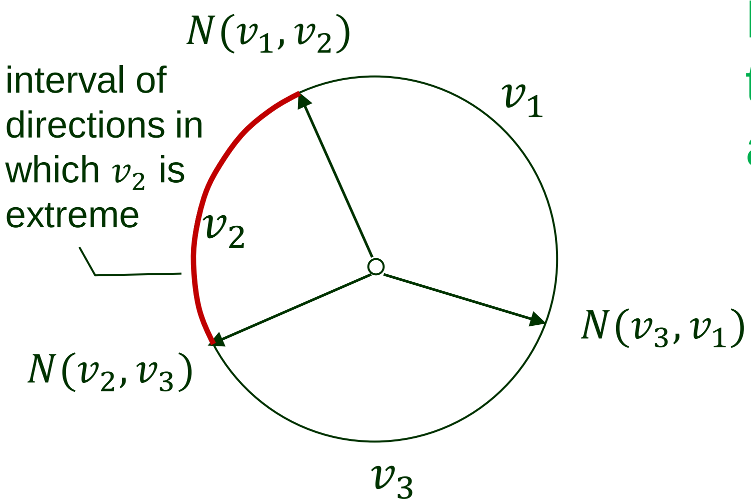
Represent all the directions by a unit circle.

Faster Computation

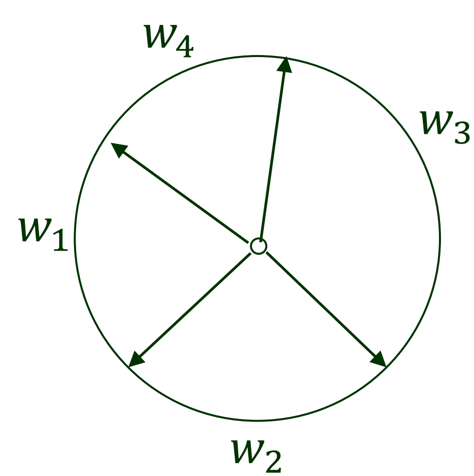
Idea: Look at a pair of vertices that are extreme in the same direction.





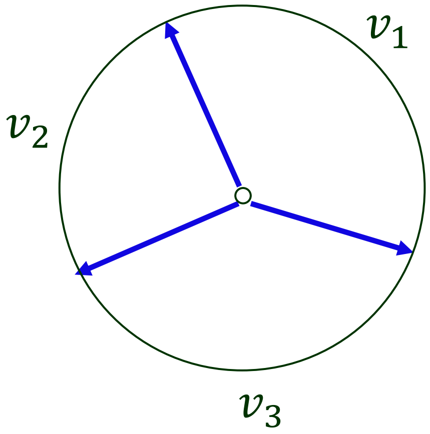


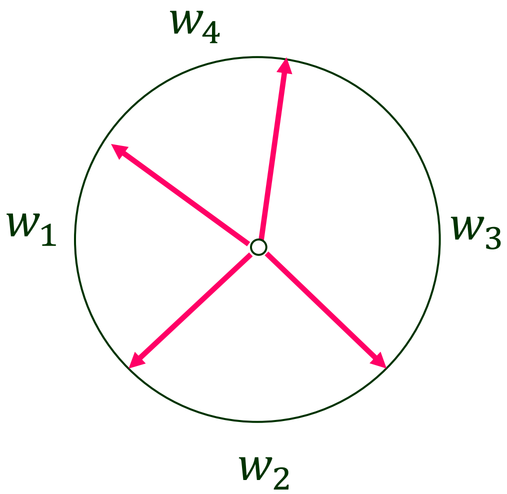
Represent all the directions by a unit circle.



Extreme Pairs

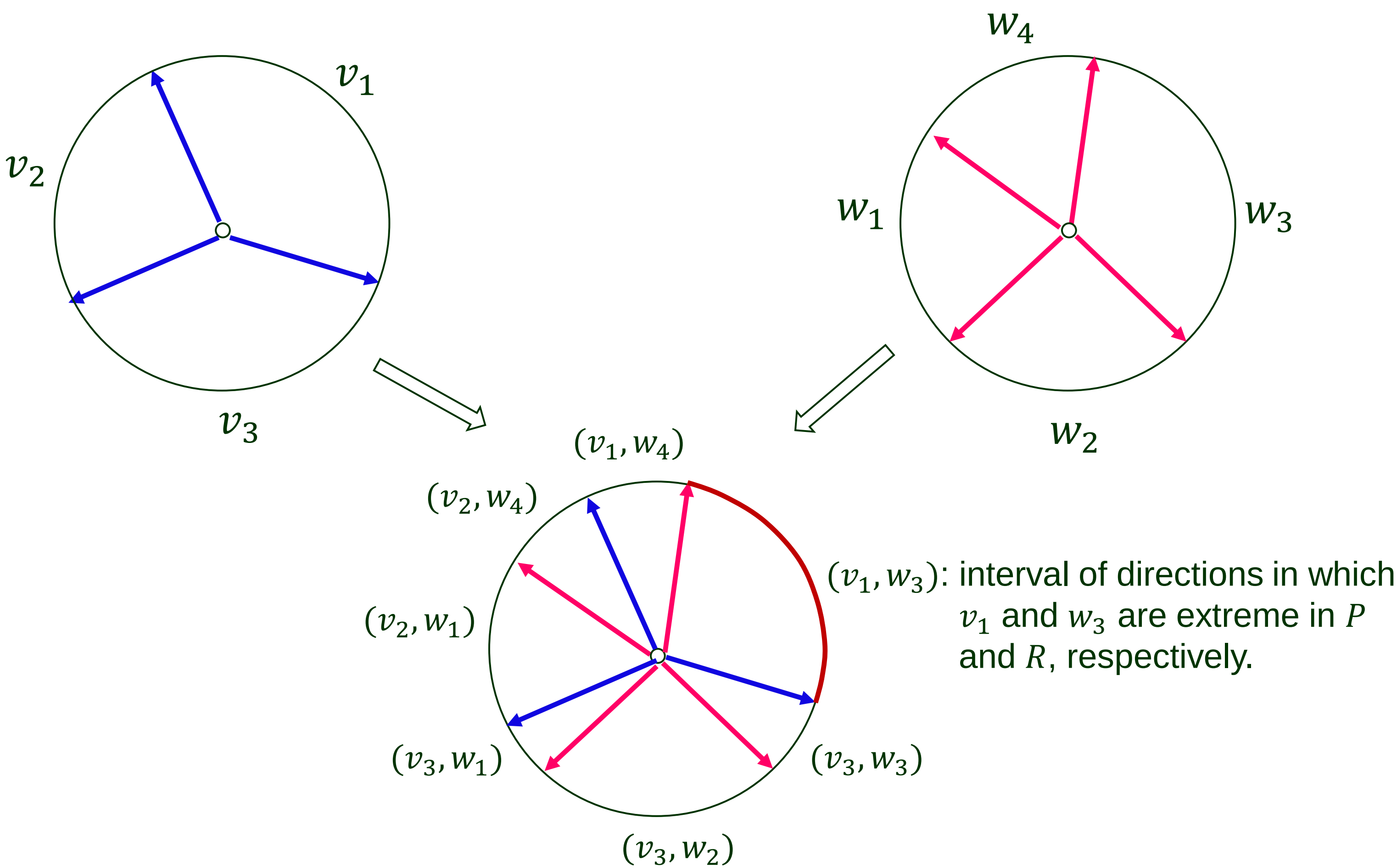
Superpose the two partitioning.





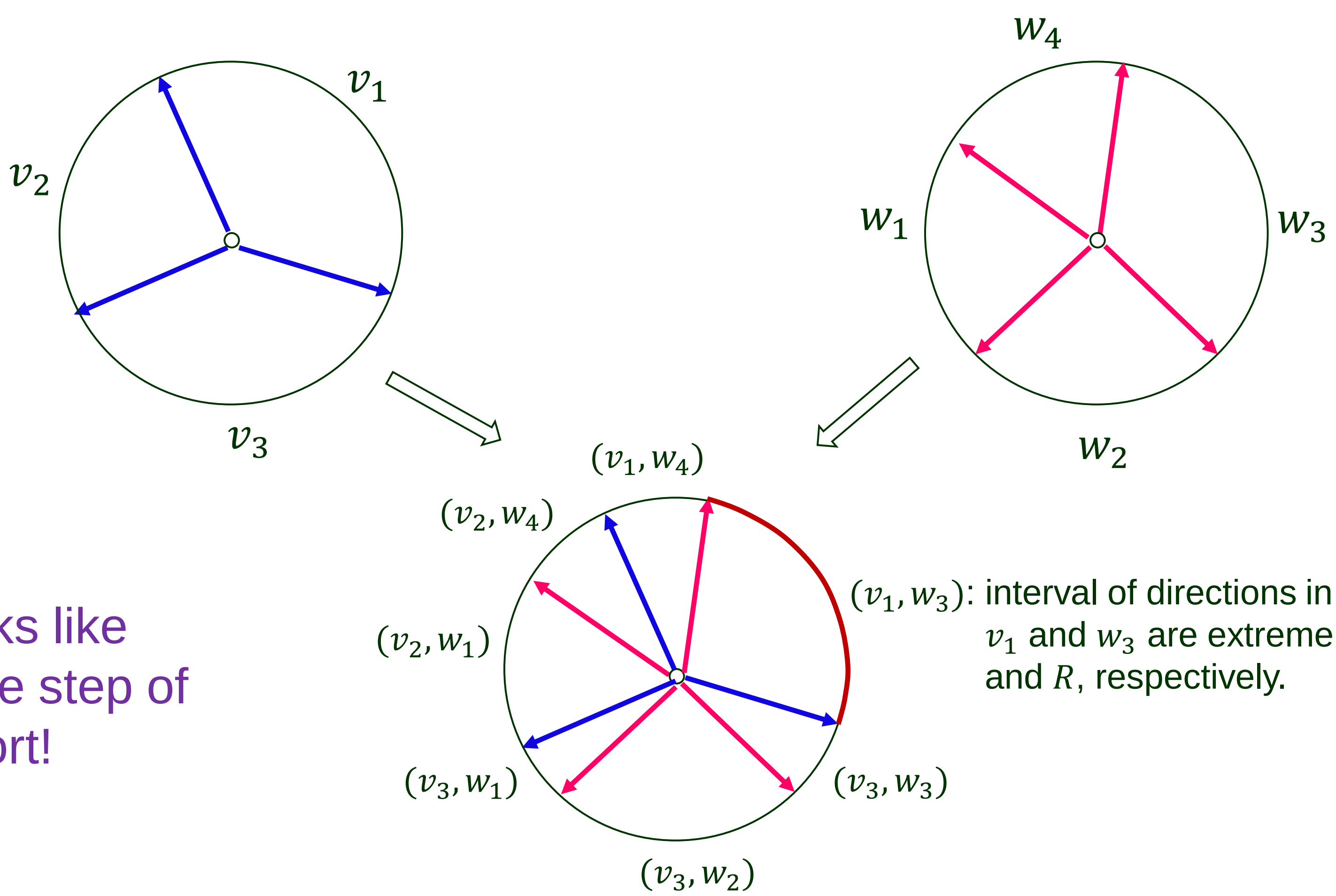
Extreme Pairs

Superpose the two partitioning.



Extreme Pairs

Superpose the two partitioning.



(v_1, w_3): interval of directions in which v_1 and w_3 are extreme in P and R, respectively.

This works like the merge step of merge sort!

The Algorithm

```
MinkowskiSum(P, R)

1.  i ← 1; j ← 1
2.  v_{n+1} ← v_1; v_{n+2} ← v_2; w_{m+1} ← w_1; w_{m+2} ← w_2
3.  repeat
4.      add v_i + w_j as a vertex to P ⊕ R
5.      if angle(v_i, v_{i+1}) < angle(w_j, w_{j+1}) // case 1
6.         then i ← i + 1
7.         ...
```

The Algorithm

```
MinkowskiSum(P, R)
```

1. $i \leftarrow 1; j \leftarrow 1$
2. $v_{n+1} \leftarrow v_1; v_{n+2} \leftarrow v_2; w_{m+1} \leftarrow w_1; w_{m+2} \leftarrow w_2$
3. repeat
4.   add $v_i + w_j$ as a vertex to $P \oplus R$
5.   if $\boxed{\text{angle}(v_i, v_{i+1})}$ < angle (w_j, w_{j+1}) // case 1
6.   then $i \leftarrow i + 1$
7.   ...

angle made by $\overrightarrow{v_i v_{i+1}}$ with the x-axis

The Algorithm

MinkowskiSum(P, R)

$i \leftarrow 1; j \leftarrow 1$

$v_{n+1} \leftarrow v_1; v_{n+2} \leftarrow v_2; w_{m+1} \leftarrow w_1; w_{m+2} \leftarrow w_2$

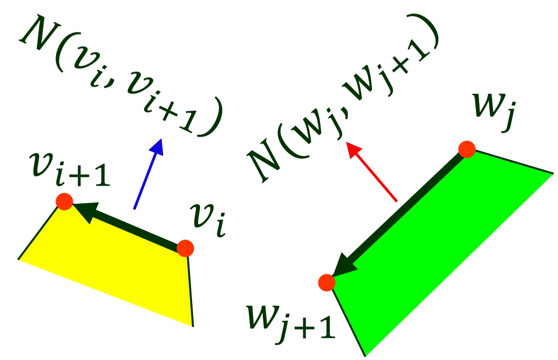
repeat

  add $v_i + w_j$ as a vertex to $P \oplus R$

  if $\boxed{\text{angle}(v_i, v_{i+1})}$ < angle (w_j, w_{j+1}) // case 1

    then $i \leftarrow i + 1$

    ...

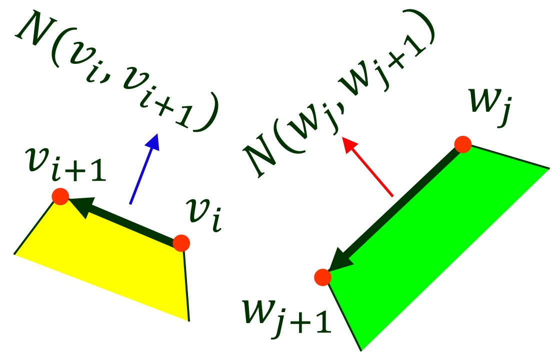


angle made by
$\overrightarrow{v_i v_{i+1}}$ with the x-axis

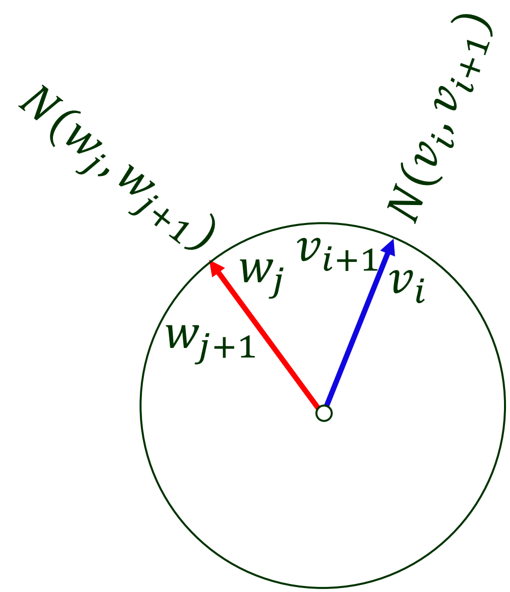
The Algorithm

MinkowskiSum(P, R)

1. $i \leftarrow 1; j \leftarrow 1$
2. $v_{n+1} \leftarrow v_1; v_{n+2} \leftarrow v_2; w_{m+1} \leftarrow w_1; w_{m+2} \leftarrow w_2$
3. repeat
4. add $v_i + w_j$ as a vertex to $P \oplus R$
5. if $\boxed{\text{angle}(v_i, v_{i+1})} < \text{angle}(w_j, w_{j+1})$ // case 1
6. then $i \leftarrow i + 1$
7. ...



angle made by
$\overrightarrow{v_iv_{i+1}}$ with the x-axis

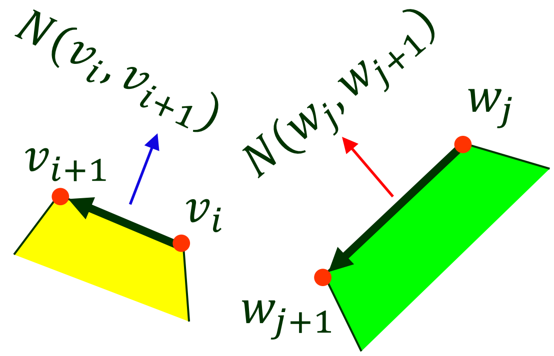


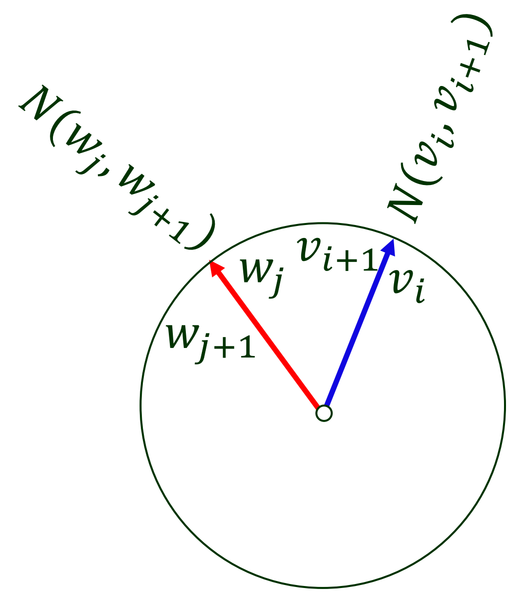
The Algorithm

```
MinkowskiSum(P, R)

i ← 1; j ← 1
v_{n+1} ← v_1; v_{n+2} ← v_2; w_{m+1} ← w_1; w_{m+2} ← w_2
repeat
    add v_i + w_j as a vertex to P ⊕ R
    if angle(v_i, v_{i+1}) < angle(w_j, w_{j+1}) // case 1
       then i ← i + 1
       ...
```

angle made by $\overrightarrow{v_i v_{i+1}}$ with the x-axis



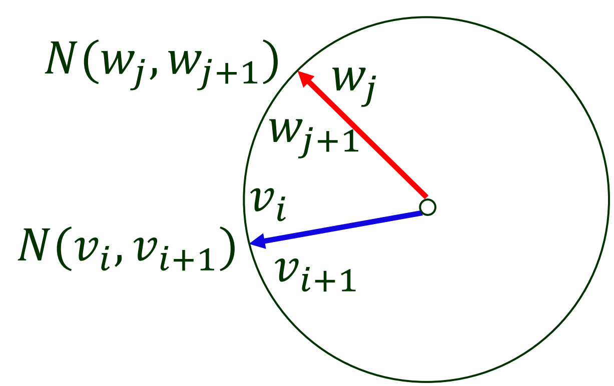


$$\cdots \to (v_i, w_j) \to (v_{i+1}, w_j) \to \cdots$$

Case 2

MinkowskiSum(P, R)

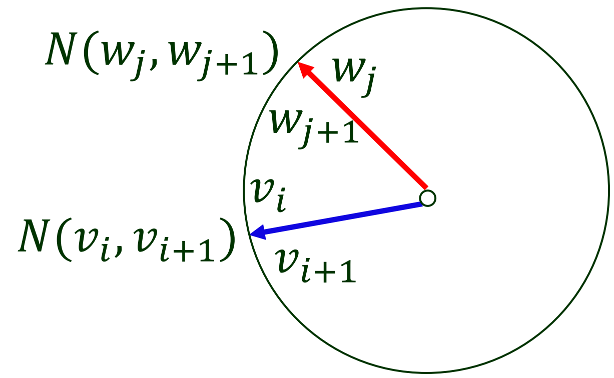
```
i ← 1; j ← 1
v_{n+1} ← v_1; v_{n+2} ← v_2; w_{m+1} ← w_1; w_{m+2} ← w_2
repeat
      add v_i + w_j as a vertex to P ⊕ R
      if angle(v_i, v_{i+1}) < angle(w_j, w_{j+1})
         then i ← i + 1
         else if angle(v_i, v_{i+1}) > angle(w_j, w_{j+1}) // case 2
                 then j ← j + 1
                 ...
```



Case 2

MinkowskiSum(P, R)

$i \leftarrow 1; j \leftarrow 1$
$v_{n+1} \leftarrow v_1; v_{n+2} \leftarrow v_2; w_{m+1} \leftarrow w_1; w_{m+2} \leftarrow w_2$
repeat
    add $v_i + w_j$ as a vertex to $P \oplus R$
    if angle$(v_i, v_{i+1}) <$ angle (w_j, w_{j+1})
        then $i \leftarrow i + 1$
        else if angle$(v_i, v_{i+1}) >$ angle (w_j, w_{j+1}) // case 2
            then $j \leftarrow j + 1$
            ...

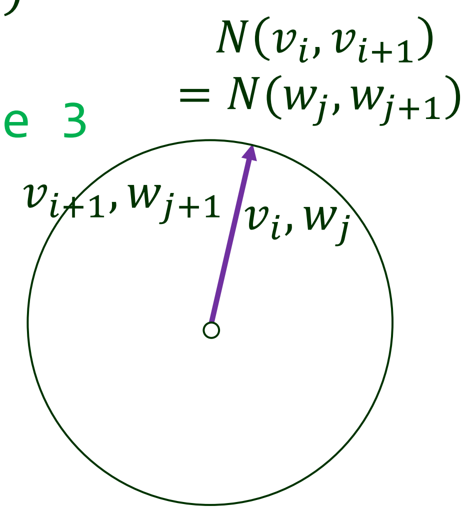


$\cdots \rightarrow (v_i, w_j) \rightarrow (v_i, w_{j+1}) \rightarrow \cdots$

Case 3

MinkowskiSum(P, R)

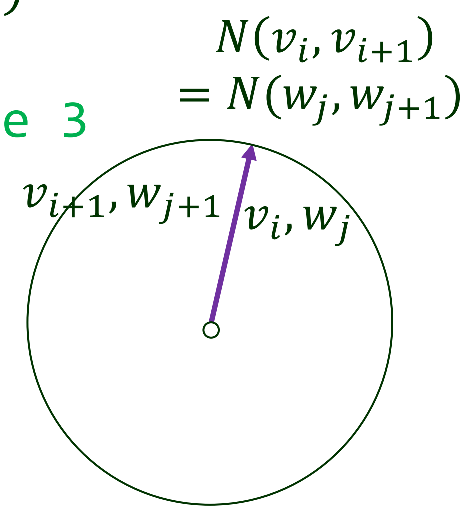
1. $i \leftarrow 1; j \leftarrow 1$
2. $v_{n+1} \leftarrow v_1; v_{n+2} \leftarrow v_2; w_{m+1} \leftarrow w_1; w_{m+2} \leftarrow w_2$
3. **repeat**
4. add $v_i + w_j$ as a vertex to $P \oplus R$
5. **if** angle$(v_i, v_{i+1}) <$ angle (w_j, w_{j+1})
6. **then** $i \leftarrow i + 1$
7. **else if** angle$(v_i, v_{i+1}) >$ angle (w_j, w_{j+1})
8. **then** $j \leftarrow j + 1$
9. **else** $i \leftarrow i + 1$; $j \leftarrow j + 1$ // case 3
10. **until** $i = n + 1$ and $j = m + 1$



Case 3

MinkowskiSum(P, R)

1. $i \leftarrow 1; j \leftarrow 1$
2. $v_{n+1} \leftarrow v_1; v_{n+2} \leftarrow v_2; w_{m+1} \leftarrow w_1; w_{m+2} \leftarrow w_2$
3. **repeat**
4.   add $v_i + w_j$ as a vertex to $P \oplus R$
5.   **if** angle$(v_i, v_{i+1}) <$ angle (w_j, w_{j+1})
6.   **then** $i \leftarrow i + 1$
7.   **else if** angle$(v_i, v_{i+1}) >$ angle (w_j, w_{j+1})
8.   **then** $j \leftarrow j + 1$
9.   **else** $i \leftarrow i + 1$; $j \leftarrow j + 1$ // case 3
10. **until** $i = n + 1$ and $j = m + 1$

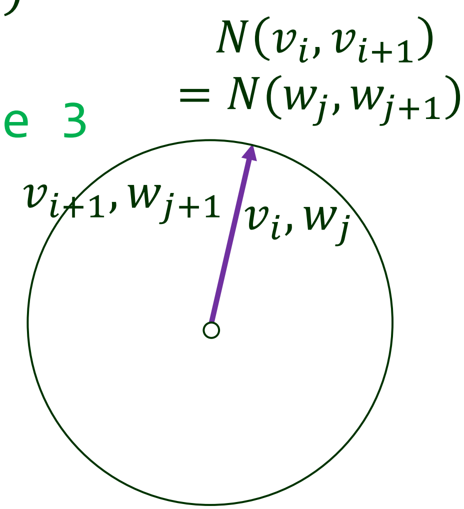


$$\cdots \rightarrow (v_i, w_j) \rightarrow (v_{i+1}, w_{j+1}) \rightarrow \cdots$$

Case 3

MinkowskiSum(P, R)

```
i ← 1; j ← 1
v_{n+1} ← v_1; v_{n+2} ← v_2; w_{m+1} ← w_1; w_{m+2} ← w_2
repeat
    add v_i + w_j as a vertex to P ⊕ R
    if angle(v_i, v_{i+1}) < angle (w_j, w_{j+1})
       then i ← i + 1
       else if angle(v_i, v_{i+1}) > angle (w_j, w_{j+1})
               then j ← j + 1
               else i ← i + 1; j ← j + 1 // case 3
until i = n + 1 and j = m + 1
```



$$\cdots \to (v_i, w_j) \to (v_{i+1}, w_{j+1}) \to \cdots$$

Running time $O(n + m)$

IV. Nonconvex Robot or Obstacle

Triangulate whichever is nonconvex.

IV. Nonconvex Robot or Obstacle

Triangulate whichever is nonconvex.

Suppose both are nonconvex and triangulated into $t_1, \ldots t_{n-2}$ and $u_1, \ldots u_{m-2}$, respectively.

IV. Nonconvex Robot or Obstacle

Triangulate whichever is nonconvex.

Suppose both are nonconvex and triangulated into $t_1, \dots t_{n-2}$ and $u_1, \dots u_{m-2}$, respectively.

$$P = \sum_{i=1}^{n-2} t_i \qquad R = \sum_{j=1}^{m-2} u_j$$

IV. Nonconvex Robot or Obstacle

Triangulate whichever is nonconvex.

Suppose both are nonconvex and triangulated into $t_1, \ldots t_{n-2}$ and $u_1, \ldots u_{m-2}$, respectively.

$$P = \sum_{i=1}^{n-2} t_i \qquad\qquad R = \sum_{j=1}^{m-2} u_j$$

Make use of the following equality for three sets S_1, S_2 and S_3:

$$S_1 \oplus (S_2 \cup S_3) = (S_1 \oplus S_2) \cup (S_1 \oplus S_3)$$

Complexity of $P \oplus R$

- Both P and R are nonconvex.

Complexity of $P \oplus R$

- Both P and R are nonconvex.

$$P \oplus R = \bigcup_{i=1}^{n-2} \bigcup_{j=1}^{m-2} t_i \oplus u_j$$

Complexity of $P \oplus R$

- Both P and R are nonconvex.

$$P \oplus R = \bigcup_{i=1}^{n-2} \bigcup_{j=1}^{m-2} t_i \oplus u_j \quad \Longrightarrow$$

Union of $O(nm)$ polygons of complexity $O(1)$

Complexity of $P \oplus R$

- Both P and R are nonconvex.

$$P \oplus R = \bigcup_{i=1}^{n-2} \bigcup_{j=1}^{m-2} t_i \oplus u_j \quad \Rightarrow$$

Union of $O(nm)$ polygons of complexity $O(1)$

$\Downarrow$

$O(n^2 m^2)$

// tight in the worst case

Complexity of $P \oplus R$

- Both P and R are nonconvex.

$$P \oplus R = \bigcup_{i=1}^{n-2} \bigcup_{j=1}^{m-2} t_i \oplus u_j$$

$\Rightarrow$ Union of $O(nm)$ polygons of complexity $O(1)$

$\Downarrow$

$O(n^2 m^2)$

// tight in the worst case

- P is convex but R is not.

Complexity of $P \oplus R$

- Both P and R are nonconvex.

$$P \oplus R = \bigcup_{i=1}^{n-2} \bigcup_{j=1}^{m-2} t_i \oplus u_j$$

Union of $O(nm)$ polygons of complexity $O(1)$

$O(n^2 m^2)$

// tight in the worst case

- P is convex but R is not.

$$P \oplus R = \bigcup_{j=1}^{m-2} P \oplus u_j$$

Complexity of $P \oplus R$

- Both P and R are nonconvex.

$$P \oplus R = \bigcup_{i=1}^{n-2} \bigcup_{j=1}^{m-2} t_i \oplus u_j$$

$\Rightarrow$ Union of $O(nm)$ polygons of complexity $O(1)$

$\Downarrow$

$O(n^2 m^2)$

// tight in the worst case

- P is convex but R is not.

$$P \oplus R = \bigcup_{j=1}^{m-2} P \oplus u_j$$

$\Rightarrow$ Union of $O(m)$ pseudodisks (any pair intersecting at ≤ 2 points)

Complexity of $P \oplus R$

- Both P and R are nonconvex.

$$P \oplus R = \bigcup_{i=1}^{n-2} \bigcup_{j=1}^{m-2} t_i \oplus u_j$$

Union of $O(nm)$ polygons of complexity $O(1)$

$O(n^2m^2)$

// tight in the worst case

- P is convex but R is not.

$$P \oplus R = \bigcup_{j=1}^{m-2} P \oplus u_j$$

Union of $O(m)$ pseudodisks (any pair intersecting at $\leq$ 2 points)

$O(nm)$

Complexity of $P \oplus R$

- Both P and R are nonconvex.

$$P \oplus R = \bigcup_{i=1}^{n-2} \bigcup_{j=1}^{m-2} t_i \oplus u_j$$

$\Rightarrow$ Union of $O(nm)$ polygons of complexity $O(1)$

$\Downarrow$

$O(n^2m^2)$

// tight in the worst case

- P is convex but R is not.

$$P \oplus R = \bigcup_{j=1}^{m-2} P \oplus u_j$$

$\Rightarrow$ Union of $O(m)$ pseudodisks (any pair intersecting at ≤ 2 points)

$\Downarrow$

$O(nm)$

- P is not convex but R is.

Complexity of $P \oplus R$

- Both P and R are nonconvex.

$$P \oplus R = \bigcup_{i=1}^{n-2} \bigcup_{j=1}^{m-2} t_i \oplus u_j \Rightarrow$$

Union of $O(nm)$ polygons of complexity $O(1)$

$\Downarrow$

$O(n^2m^2)$

// tight in the worst case

- P is convex but R is not.

$$P \oplus R = \bigcup_{j=1}^{m-2} P \oplus u_j \Rightarrow$$

Union of $O(m)$ pseudodisks (any pair intersecting at ≤ 2 points)

$\Downarrow$

$O(nm)$

- P is not convex but R is.

$$P \oplus R = \bigcup_{i=1}^{n-2} t_i \oplus R$$

$O(nm)$

V. Translational Motion Planning

We are given

- the robot R with constant complexity;
- a set of obstacles with total complexity $O(n)$.

V. Translational Motion Planning

We are given

- the robot R with constant complexity;
- a set of obstacles with total complexity $O(n)$.

♦ $T_1, T_2, \ldots, T_n$: the triangles from triangulating the obstacles.

V. Translational Motion Planning

We are given

- the robot R with constant complexity;
- a set of obstacles with total complexity $O(n)$.

♦ $T_1, T_2, \ldots, T_n$: the triangles from triangulating the obstacles.

♦ Forbidden configuration space

$$C_{forb} = \bigcup_{i=1}^{n} CP_i = \bigcup_{i=1}^{n} T_i \oplus (-R(0,0))$$

V. Translational Motion Planning

We are given

- the robot R with constant complexity;
- a set of obstacles with total complexity $O(n)$.

♦ $T_1, T_2, \ldots, T_n$: the triangles from triangulating the obstacles.

♦ Forbidden configuration space

$$C_{forb} = \bigcup_{i=1}^{n} CP_i = \bigcup_{i=1}^{n} T_i \oplus (-R(0,0))$$

Complexity $O(n)$

Computing the Forbidden C-Space

Divide-and-conquer

1. $C_{forb}^1 \leftarrow \bigcup_{i=1}^{\frac{n}{2}} CP_i$

2. $C_{forb}^2 \leftarrow \bigcup_{i=\frac{n}{2}+1}^{n} CP_i$

3. Compute $C_{forb} = C_{forb}^1 \cup C_{forb}^2$

Computing the Forbidden C-Space

Divide-and-conquer

1. $C^1_{forb} \leftarrow \bigcup_{i=1}^{\frac{n}{2}} CP_i$

2. $C^2_{forb} \leftarrow \bigcup_{i=\frac{n}{2}+1}^{n} CP_i$

3. Compute $C_{forb} = \underbrace{C^1_{forb} \cup C^2_{forb}}_{\text{overlay of planar subdivisions}}$ $O\,(n \log n)$

Computing the Forbidden C-Space

Divide-and-conquer

1. $C^1_{forb} \leftarrow \bigcup_{i=1}^{\frac{n}{2}} CP_i$

2. $C^2_{forb} \leftarrow \bigcup_{i=\frac{n}{2}+1}^{n} CP_i$

3. Compute $C_{forb} = \underbrace{C^1_{forb} \cup C^2_{forb}}$

overlay of planar subdivisions $O\,(n \log n)$

$T(n)$: time of computation

$$T(n) = 2T\left(\frac{n}{2}\right) + O(n \log n)$$

Computing the Forbidden C-Space

Divide-and-conquer

1. $C^1_{forb} \leftarrow \bigcup_{i=1}^{\frac{n}{2}} CP_i$

2. $C^2_{forb} \leftarrow \bigcup_{i=\frac{n}{2}+1}^{n} CP_i$

3. Compute $C_{forb} = \underbrace{C^1_{forb} \cup C^2_{forb}}_{\text{overlay of planar subdivisions } O(n \log n)}$

$T(n)$: time of computation

$$T(n) = 2T\left(\frac{n}{2}\right) + O(n \log n) \Longrightarrow T(n) = O(n \log^2 n)$$

Computing the Forbidden C-Space

Divide-and-conquer

1. $C_{forb}^{1} \leftarrow \bigcup_{i=1}^{\frac{n}{2}} CP_i$

2. $C_{forb}^{2} \leftarrow \bigcup_{i=\frac{n}{2}+1}^{n} CP_i$

3. Compute $C_{forb} = \underbrace{C_{forb}^{1} \cup C_{forb}^{2}}_{\text{overlay of planar subdivisions } O(n \log n)}$

$T(n)$: time of computation

$$T(n) = 2T\left(\frac{n}{2}\right) + O(n \log n) \implies T(n) = O(n \log^2 n)$$

C_{free} is the complement of C_{forb}.

Time Complexity for Path Planning

Theorem 3 $\mathcal{C}_{free}$ can be computed in time $O(n \log^2 n)$.

Time Complexity for Path Planning

Theorem 3 C_{free} can be computed in time $O(n \log^2 n)$.

Next, compute a trapezoidal map of C_{free} in $O(n \log n)$ expected time.

Time Complexity for Path Planning

Theorem 3 $\mathcal{C}_{free}$ can be computed in time $O(n \log^2 n)$.

Next, compute a trapezoidal map of $\mathcal{C}_{free}$ in $O(n \log n)$ expected time.

Total preprocessing time:

$$O(n \log^2 n + n \log n) = O(n \log^2 n)$$

Time Complexity for Path Planning

Theorem 3 C_{free} can be computed in time $O(n\log^2 n)$.

Next, compute a trapezoidal map of C_{free} in $O(n\log n)$ expected time.

Total preprocessing time:

$$O(n\log^2 n + n\log n) = O(n\log^2 n)$$

Theorem 4 Translational motion planning for a convex robot of $O(1)$ complexity among polygonal obstacles of $O(n)$ total complexity can be solved in $O(n)$ time with preprocessing in $O(n\log^2 n)$ expected time.