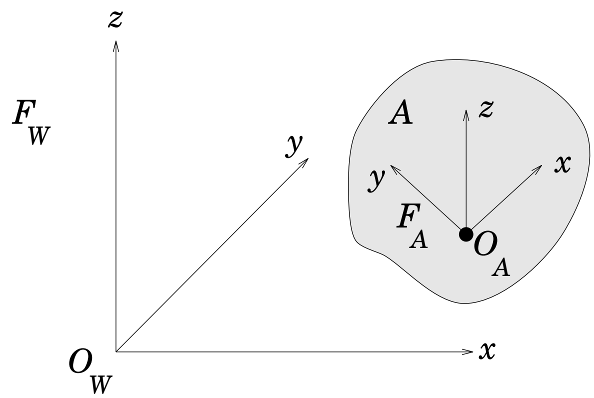
Configuration Space

Acknowledgement: Parts of these course notes are based on notes from courses given by Jean-Claude Latombe at Stanford University (and Chapters 2 and 3 in his text *Robot Motion Planning*, Kluwer, 1991), O. Burçhan Bayazıt at Washington University in St. Louis. Seth Hutchinson at the University of Illinois at Urbana-Champaign, and Leo Joskowicz at Hebrew University.

C-space of Rigid Object

Main Idea: Represent the robot as a point, called a *configuration*, in a parameter space, the *configuration space* (or *C-space*).

Note: For now, assume robot is a rigid object not constrained by any kinematic or dynamic constraints.



Workspace $\mathcal{W}$: physical workspace

- represented as N-dimensional Euclidean Space $\mathbb{R}^N$, where $N = 2, 3$
- $\mathcal{F}_\mathcal{W}$: *fixed* Cartesian coordinate system (frame) of $\mathcal{W}$
- $\mathcal{O}_\mathcal{W}$: *fixed* origin of $\mathcal{F}_\mathcal{W}$

Robot $\mathcal{A}$: moving rigid object/robot

- represented as compact subset of $\mathbb{R}^N$ (at reference position and orientation)
- $\mathcal{F}_\mathcal{A}$: frame of $\mathcal{A}$ (aka 'local' frame of $\mathcal{A}$)
  – *fixed* wrt $\mathcal{A}$ (i.e., each point in $\mathcal{A}$ has fixed coordinates in $\mathcal{F}_\mathcal{A}$)
  – *moving* wrt $\mathcal{F}_\mathcal{W}$
- $\mathcal{O}_\mathcal{A}$: origin of $\mathcal{F}_\mathcal{A}$ (aka the **reference point** of $\mathcal{A}$)

C-space of Rigid Object (cont'd)

Definitions:

- A **configuration q** of $\mathcal{A}$ is a specification of the position and orientation of $\mathcal{F}_{\mathcal{A}}$ wrt $\mathcal{F}_{\mathcal{W}}$
- The **configuration space** of $\mathcal{A}$ is the space $\mathcal{C}$ of all the possible configurations of $\mathcal{A}$
- The **reference configuration** of $\mathcal{A}$, denoted by 0, is some (arbitrary) configuration of $\mathcal{A}$

Note: $\mathcal{C}$ is *independent* of choice of $\mathcal{F}_{\mathcal{A}}$ and $\mathcal{F}_{\mathcal{W}}$ (but its *representation* is not)

Notation:

- $\mathcal{A}(\mathbf{q})$: subset of $\mathcal{W}$ occupied by $\mathcal{A}$ at configuration $\mathbf{q}$
- $a(\mathbf{q})$: position of point $a \in \mathcal{A}$ in $\mathcal{W}$ when $\mathcal{A}$ at configuration $\mathbf{q}$
- $x(\mathbf{q})$: position of feature $x \in A$ in $\mathcal{W}$ when $\mathcal{A}$ at configuration $\mathbf{q}$

Configurations as transformations

Sometimes useful to think of configurations as rigid body transformations...

transformation $TR_{\mathbf{q}}$ rotates and translates $\mathcal{A}(0)$ to $\mathcal{A}(\mathbf{q})$

- denoted $TR_{\mathbf{q}}(\mathcal{A}(0)) = \mathcal{A}(\mathbf{q})$
- preserves distance and orientation

Fact 1: for every $\mathbf{q} \in \mathcal{C}$ there exists a unique rotation r and translation t, such that $TR_{\mathbf{q}} = t \circ r$ (composition with r applied first), i.e., $TR_{\mathbf{q}}(\mathcal{A}(0)) = \mathcal{A}(\mathbf{q})$

Fact 2: the space of rigid body transformations has the structure of a non-commutative group; composition of two transformations is the binary operator

Embedding $\mathcal{C}$ in Euclidean Space

Idea: represent $\mathcal{C}$ (for rigid $\mathcal{A}$) as a subset of $\mathbb{R}^M$

- M depends on N, the dimension of $\mathcal{W}$ (e.g, $M = N + N^2$)
- $\mathbb{R}^M$ is **ambient space** of $\mathcal{C}$
- representation of $\mathcal{C}$ is **embedding** of $\mathcal{C}$ in $\mathbb{R}^M$

Configuration $\mathbf{q}$ *represented as pair* $(\mathcal{T}, \Theta)$ (translation and rotation)

- depends on choice of $\mathcal{F}_{\mathcal{A}}$ and $\mathcal{F}_{\mathcal{W}}$
- $\mathcal{T}$ determines position of $\mathcal{O}_{\mathcal{A}}$ in $\mathcal{F}_{\mathcal{W}}$
  - N-vector of coordinates of $\mathcal{O}_{\mathcal{A}}$ in $\mathcal{F}_{\mathcal{W}}$
- Θ determines orientation of $\mathcal{F}_{\mathcal{A}}$'s axes with respect to $\mathcal{F}_{\mathcal{W}}$
  - $N \times N$ matrix whose columns are components of the unit vectors along $\mathcal{F}_{\mathcal{A}}$'s axes in $\mathcal{F}_{\mathcal{W}}$
  - Θ in Special Othogonal Group $SO(N)$ of $N \times N$ matrices in $\mathbb{R}^{N^2}$ with orthonormal columns and rows and determinant $+1$
  - called 'rotation matrices'

Embedding $\mathcal{C}$ in Euclidean Space (cont'd)

Suppose **reference configuration** 0 is where $\mathcal{F}_\mathcal{A}$ and $\mathcal{F}_\mathcal{W}$ coincide:

- $\mathcal{T} = \vec{0}$ (zero N-vector)
- $\Theta = I$ ($N \times N$ identity matrix)

Then, for every point $a \in \mathcal{A}$,

$$\begin{cases} \mathbf{a} & = & a(0) \\ a(\mathbf{q}) & = & \Theta a(0) + \mathcal{T} \end{cases}$$

- $\mathbf{a}$ is N-vector of coordinates of point a in $\mathcal{F}_\mathcal{A}$
- $a(\mathbf{q})$ is N-vector of coordinates of point a in $\mathcal{F}_\mathcal{W}$ when $\mathcal{A}$ is in configuration $\mathbf{q}$

Note: Given $\mathcal{T}$ and Θ, the position of any point of $\mathcal{A}$ in $\mathcal{F}_\mathcal{W}$ can be computed.

Fortunately, the form of the rotation matrices Θ is known for both two and three-dimensional $\mathcal{W}$.

Constructing the Rotation Matrix Θ

Ex: Two-Dimensional Workspaces

The following matrix Θ represents the orientation of $\mathcal{F}_{\mathcal{A}}$ after a rotation of angle θ around $\mathcal{O}_{\mathcal{W}}$, starting from the reference configuration 0:

$$\Theta = \begin{pmatrix} r_{11} & r_{12} \\ r_{21} & r_{22} \end{pmatrix} = \begin{pmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{pmatrix}$$

Let $\mathcal{M}_{\mathcal{C}}$ equal the subset $\mathbb{R}^N \times SO(N) \subset \mathbb{R}^M$

- where $M = N + N^2$ ($M = 6$ for $\mathcal{W} = \mathbb{R}^2$ and $M = 12$ for $\mathcal{W} = \mathbb{R}^3$)
- $\mathcal{M}_{\mathcal{C}}$ represents $\mathcal{C}$ in $\mathbb{R}^M$

For $\mathcal{W} = \mathbb{R}^2$, each $\mathbf{q} \in \mathcal{C}$ is given by six numbers $(x, y, r_{11}, r_{12}, r_{21}, r_{22}) \in \mathcal{M}_{\mathcal{C}}$. The constraints (boundaries) of $\mathcal{M}_{\mathcal{C}}$ are given by:

$$\begin{aligned} h_1(x, y, r_{11}, r_{12}, r_{21}, r_{22}) &= r_{11}^2 + r_{21}^2 - 1 = 0 && (1) \\ h_2(x, y, r_{11}, r_{12}, r_{21}, r_{22}) &= r_{12}^2 + r_{22}^2 - 1 = 0 && (2) \\ h_3(x, y, r_{11}, r_{12}, r_{21}, r_{22}) &= r_{11}r_{12} + r_{21}r_{22} = 0 && (3) \end{aligned}$$

Similarly, for $\mathcal{W} = \mathbb{R}^3$, the rotation matrix Θ is known, and $\mathcal{M}_{\mathcal{C}}$ is a subset of $\mathbb{R}^{12}$ (see Latombe, Ch. 2)

Properties of $\mathcal{C}$

Fact: $\mathcal{C}$ is a smooth m-dimensional manifold, where fixed m depends on M

- *local* topological and differential structures are identical to those of $\mathbb{R}^m$ (not global though)
- key point: $\mathcal{C}$ is locally 'like' $\mathbb{R}^m$ so we can parameterize it, e.g.,
  - *stereographic projection* (only for $SO(1)$),
  - *Euler angles* (3 angles for $S0(3)$),
  - *quaternions* (4 values for $SO(3)$).

Fact: Sometimes can get by with simpler $\mathcal{C}$ (use symmetry)

- circle, sphere: $\mathcal{C} = \mathbb{R}^N$ if choose origin at center
- cylinder in $\mathbb{R}^3$: $\mathcal{C} = \mathbb{R}^3 \times S^2$ (unit sphere in $\mathbb{R}^3$) if choose $\mathcal{O}_\mathcal{A}$ on cylinder's axis and one axis of $\mathcal{F}_\mathcal{A}$ on cylinder's axis
- no rotation allowed: $\mathcal{C} = \mathbb{R}^N$

Obstacles in C-space

Let $\mathcal{W}$ contain fixed obstacles $\mathcal{B}_1, \mathcal{B}_2, \ldots, \mathcal{B}_q$ which are closed, but not necessarily bounded, regions of $\mathbb{R}^N$.

Definition: The obstacle $\mathcal{B}_i$ in $\mathcal{W}$ maps in $\mathcal{C}$ to the region

$$\mathcal{CB}_i = \{\mathbf{q} \in \mathcal{C} | \mathcal{A}(\mathbf{q}) \cap \mathcal{B}_i \neq \emptyset\}$$

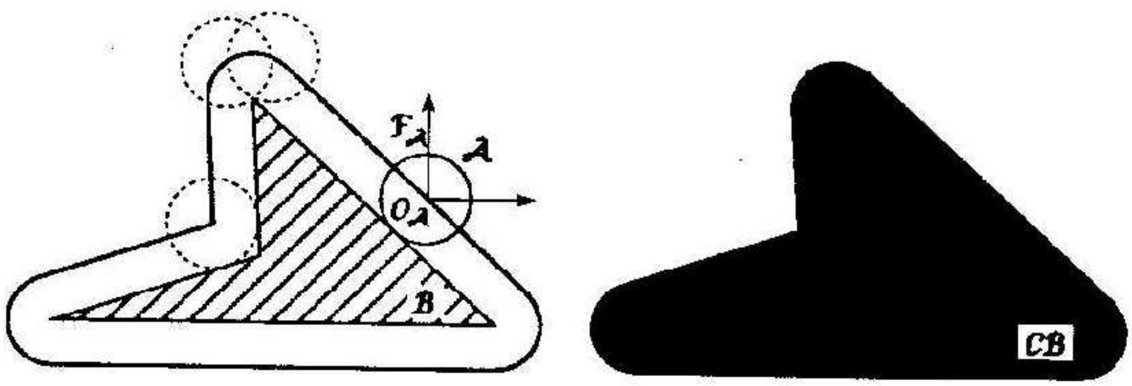
$\mathcal{CB}_i$ is called a *C-obstacle*. The union of all C-obstacles is called the *C-obstacle region*.

Properties of C-obstacles

If $\mathcal{B}_i$ and $\mathcal{A}$ are:

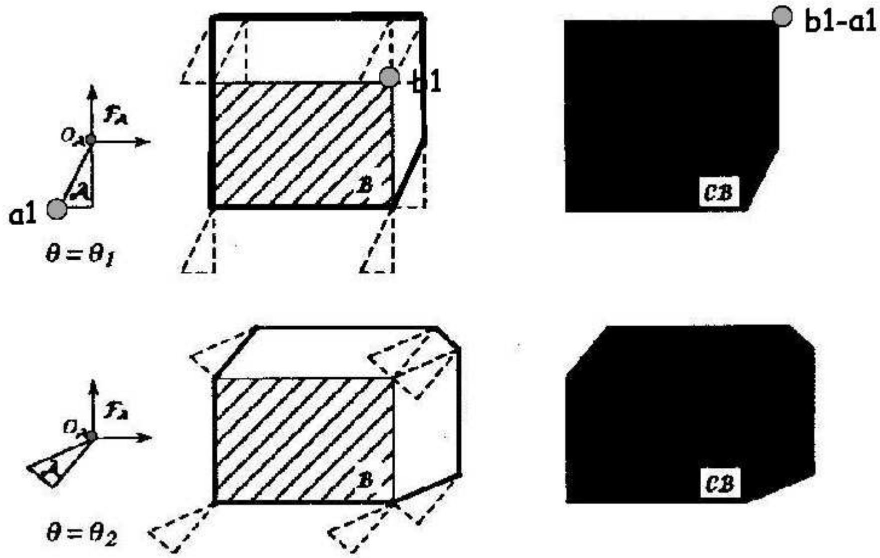
- *compact sets*, then $\mathcal{CB}_i$ is a compact set
- *algebraic*, then $\mathcal{CB}_i$ is algebraic
- *closed*, then $\mathcal{CB}_i$ is closed
- *connected*, then $\mathcal{CB}_i$ is connected

C-obstacle Example: $\mathcal{A}$ is a disc and $\mathcal{W} = \mathbb{R}^2$



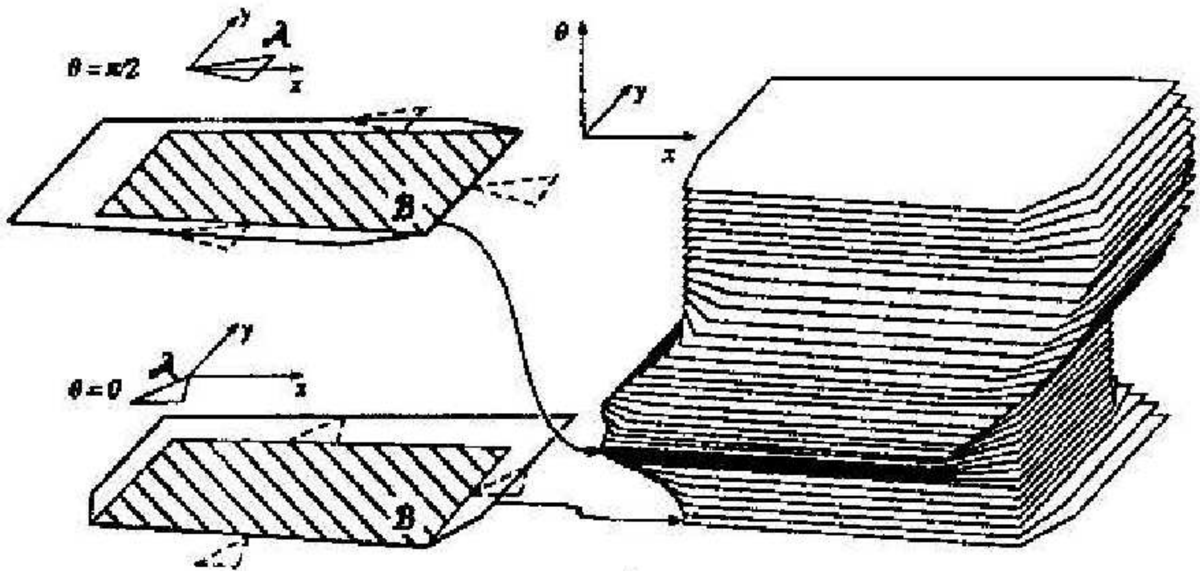
- select $\mathcal{O}_\mathcal{A}$ at center of disc (and thus $\mathcal{F}_\mathcal{A}$ also)
- $\mathcal{CB}$ of polygon $\mathcal{B}$ obtained by growing $\mathcal{B}$ by radius of $\mathcal{A}$

C-obstacle Example: $\mathcal{A}$ and $\mathcal{B}$ are convex polygons, $\mathcal{W} = \mathbb{R}^2$



- $\mathcal{A}$ can translate freely, but cannot rotate (for two orientations of $\mathcal{A}$)
- boundary of $\mathcal{CB}$ is curve followed by $\mathcal{O}_{\mathcal{A}}$ when $\mathcal{A}$ slides in contact with $\mathcal{B}$'s boundary
- $\mathcal{CB}$ is also convex

C-obstacle Example: $\mathcal{A}$ and $\mathcal{B}$ are convex polygons, $\mathcal{W} = \mathbb{R}^2$



- $\mathcal{A}$ can translate and rotate
- parameterize $\mathbf{q}$ by $(x, y, \theta) \in \mathbb{R}^2 \times [0, 2\pi]$
- cross-section of $\mathcal{CB}$ is 2D C-obstacle at orientation of $\mathcal{A}$ on θ axis

Paths in C-Space

A *path in C-space* is a 'piece' of a continuous curve connecting two configurations q_{init} and q_{goal}. Or, more formally, a path is continuous map:

$$\tau : s \in [0,1] \mapsto \tau(s) \in \mathcal{C}$$

where $\tau(0) = \mathbf{q}_{init}$ is the initial configuration and $\tau(1) = \mathbf{q}_{goal}$ is the goal configuration of the path.

"Continuous map" means that:

$$\forall s_1, s_2 \in [0,1] : \lim_{s_2 \to s_1} d(\tau(s_1), \tau(s_2)) = 0$$

where $d : \mathcal{C} \times \mathcal{C} \to \mathbf{R}^+ \cup \{0\}$ is the chosen metric over $\mathcal{C}$.

<u>*Example of a metric in $\mathcal{C}$:*</u>

$$d(\mathbf{q}_1, \mathbf{q}_2) = max_{a \in \mathcal{A}} \|a(\mathbf{q}_1) - a(\mathbf{q}_2)\|$$

where $\|x - y\|$ is the Euclidean metric in $\mathcal{W}$.

<u>Other Potential Constraints on the path</u>

- minimal length, smoothness, number of turns, limits on curvature, etc.
- trajectory can be parameterized by time (effectively adding another dimention to configuration space)

Paths

Types of Paths:

- *Free path*: $\tau : [0, 1] \rightarrow \mathcal{C}_{free} = \mathcal{C} \backslash \mathcal{CB}$
  (entirely in $\mathcal{C}_{\text{free}}$, no contact)
- *Semi-free path*: $\tau : [0, 1] \rightarrow cl(\mathcal{C}_{free})$ (may have contact, but not invalid)

Homotopic Paths

- Two paths with the same start and goal configurations are *homotopic* if one can be continuously deformed into the other

Connectedness of C-Space

- C is connected if all pairs of configurations in C can be connected by a path
- C is simply connected if any two paths connecting the same endpoints are homotopic.
  Examples: $\mathbb{R}^2$ or $\mathbb{R}^3$
- Otherwise, C is multiply connected.
  Examples: S^1 (infinite number of homotopy classes) and $SO(3)$ (two homotopy classes)

Representation

Several Possible Representations: (also holds for robot $\mathcal{A}$)

- *set representation*: a compact set of points describing the object
- *functional representation*: the intersection of halfspaces
  - $\mathcal{B} = \{h_i(\mathbf{x})\}$, $h_i(\mathbf{x}) \leq 0$
  - $\mathbf{p} \in \mathcal{B} \Longleftrightarrow h_i(\mathbf{p}) \leq 0, \forall i$
  - notation:
    $\overline{h}_i^+(\mathbf{x}) = \{\mathbf{p} \mid h_i(\mathbf{p}) \geq 0\}$
    $\overline{h}_i^-(\mathbf{x}) = \{\mathbf{p} \mid h_i(\mathbf{p}) \leq 0\}$
- *boundary representation*: an ordered sequence of primitive features
  - $\mathcal{B} = \{b_i\}$
  - if $\mathcal{W} = \mathbb{R}^2$, then, the b_i are vertices, line segments, lines, arcs, etc.
  - if $\mathcal{W} = \mathbb{R}^3$, then, we also have facets, surfaces, etc.

Constructing C-obstacles in a Polygonal Workspace

Note: points on boundary of C-obstacle $\mathcal{CB}$ correspond to configurations in which robot $\mathcal{A}$ is in contact with Workspace obstacle $\mathcal{B}$

Basic Algorithm to Construct $\mathcal{CB}$

input: $\mathcal{A}$ and $\mathcal{B}$ (boundary representation)

output: $\mathcal{CB}$ (boundary representation)

1. For each pair of features (a_i, b_i), $a_i \in \mathcal{A}$ and $b_i \in \mathcal{B}$
   compute *contact constraint* for a_i and b_i
   (when a_i is in contact with b_i but $\mathcal{A}$ does not properly intersect $\mathcal{B}$)
2. Trace the boundary of $\mathcal{CB}$ by 'walking' along surfaces defined by contact constraints

---

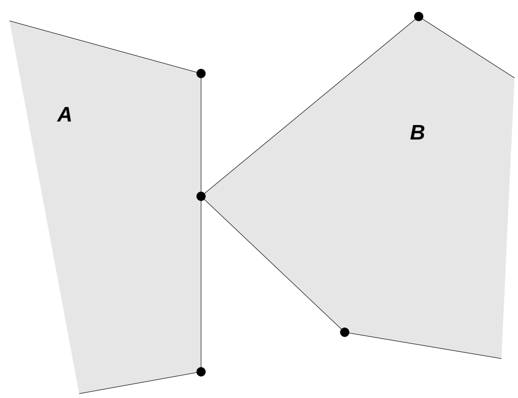
Contact Constaints

Determined by:

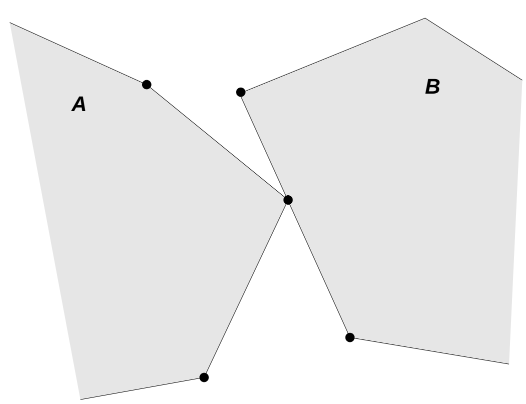
- feature type (vertex, edge, surface)
- degrees of freedom (e.g., translation, rotation)

Example: $\mathcal{A}$ and $\mathcal{B}$ are Convex Polygons ($\mathcal{W} = \mathbb{R}^2$)

Type A Contact: edge of $\mathcal{A}$ in contact with vertex of $\mathcal{B}$



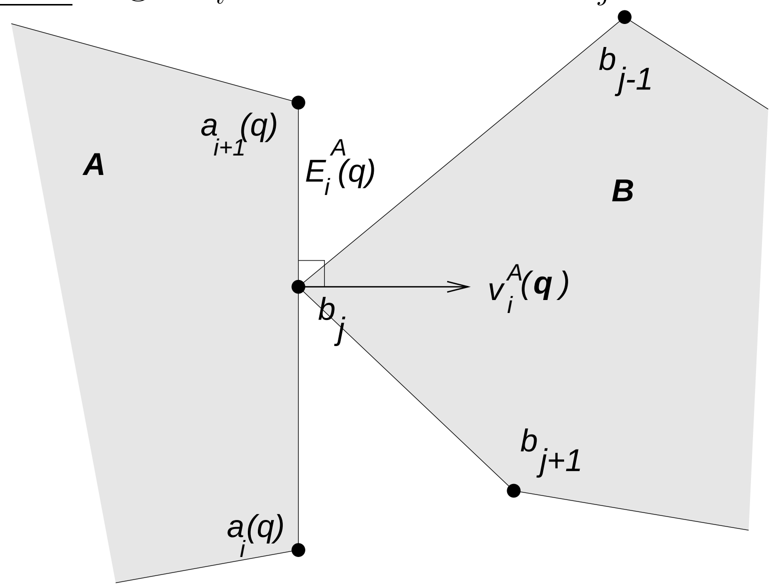
Type B Contact: vertex of $\mathcal{A}$ in contact with edge of $\mathcal{B}$



Note: If vertex of $\mathcal{A}$ in contact with vertex of $\mathcal{B}$, then we say it is both a type A and type B contact.

Deriving Contact Constraints (convex polygons)

Type A Contacts: edge $E_i^{\mathcal{A}}$ of $\mathcal{A}$ with vertex b_j of $\mathcal{B}$



Feasibility (aka Applicability)

- Type A contact between $E_i^{\mathcal{A}}$ and b_j *feasible* (i.e., contact with no overlap) only for a subrange of orientations of $\mathcal{A}$
- feasible subrange is given by two conditions (angles between vectors $\leq 90°$)

$$\vec{\nu}_i^{\mathcal{A}}(\mathbf{q}) \cdot (b_{j-1} - b_j) \geq 0$$

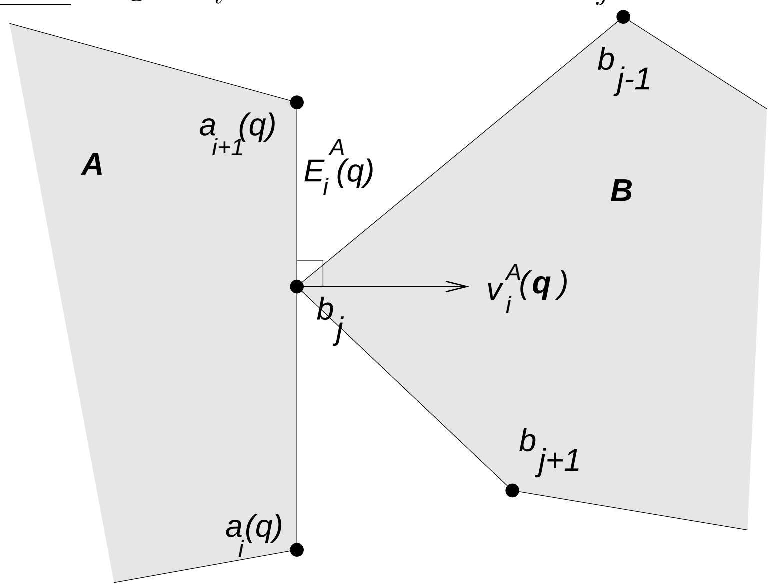
$$\vec{\nu}_i^{\mathcal{A}}(\mathbf{q}) \cdot (b_{j+1} - b_j) \geq 0$$

Feasibility (Applicability) Condition: $\text{APPL}_{i,j}^{\mathcal{A}}(\mathbf{q})$

- conjunction of the two conditions above
- when $\mathbf{q}$ expressed as $(\mathcal{T}, \Theta)$, depends only on Θ (orientation)

Deriving Contact Constraints (cont'd)

Type A Contacts: edge $E_i^{\mathcal{A}}$ of $\mathcal{A}$ with vertex b_j of $\mathcal{B}$



Contact Condition

- when Type A contact occurs, the vectors $\vec{\nu}_i^{\mathcal{A}}(\mathbf{q})$ (normal to edge) and $(b_j - a_i(\mathbf{q}))$ (along edge) are perpendicular

C-surfaces of Type A (where Type A contact occurs)

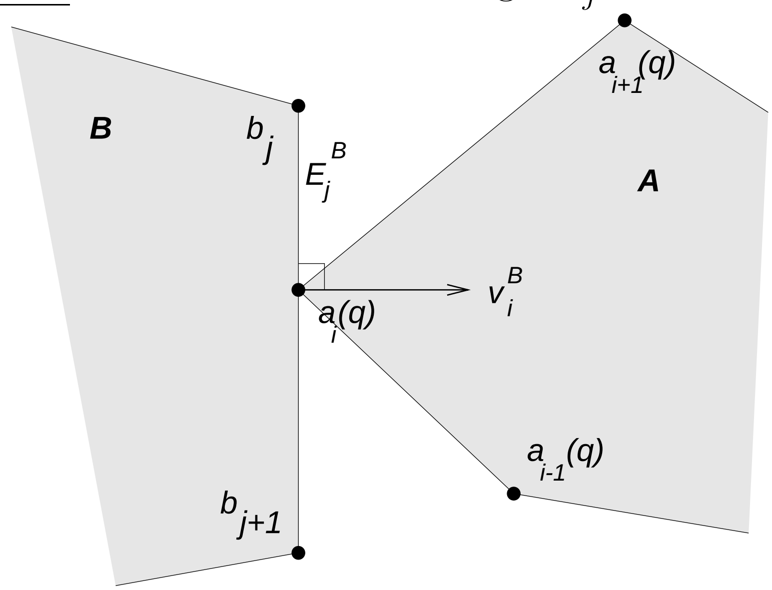
- as $\mathcal{A}$ is moved, but Type A contact is maintained, the configuration moves along a surface in $\mathcal{C}$, called a **C-surface** of type A (a 2D submanifold of $\mathcal{C}$)
- This C-surface is expressed by: $f_{i,j}^{\mathcal{A}}(\mathbf{q}) = 0$ with $f_{i,j}^{\mathcal{A}}(\mathbf{q}) = \vec{\nu}_i^{\mathcal{A}}(\mathbf{q}) \cdot (b_j - a_i(\mathbf{q}))$
- This C-surface partitions $\mathcal{C}$ into two halfspaces, and the C-obstacle $\mathcal{CB}$ lies completely within the closed halfspace determined by $f_{i,j}^{\mathcal{A}}(\mathbf{q}) \leq 0$.

C-constraint of Type A: $\mathrm{CONST}_{i,j}^{\mathcal{A}}(\mathbf{q}) \equiv \mathrm{APPL}_{i,j}^{\mathcal{A}}(\mathbf{q}) \Longrightarrow (f_{i,j}^{\mathcal{A}}(\mathbf{q}) \leq 0)$

- if $\mathrm{APPL}_{i,j}^{\mathcal{A}}(\mathbf{q})$ is FALSE, then $\mathrm{CONST}_{i,j}^{\mathcal{A}}(\mathbf{q})$ is TRUE
  i.e., ignore C-surface $f_{i,j}^{\mathcal{A}}(\mathbf{q}) = 0$

Deriving Contact Constraints (cont'd)

Type B Contacts: vertex a_i of $\mathcal{A}$ with edge $E_j^{\mathcal{B}}$ of $\mathcal{B}$



Feasibility (aka Applicability)

- Type B contact between a_i and $E_j^{\mathcal{B}}$ *feasible* (i.e., contact with no overlap) only for a subrange of orientations of $\mathcal{A}$
- feasible subrange is given by two conditions (angles between vectors $\leq 90°$)

$$(a_{i-1} - a_i) \cdot \vec{\nu}_j^{\mathcal{B}} \geq 0$$

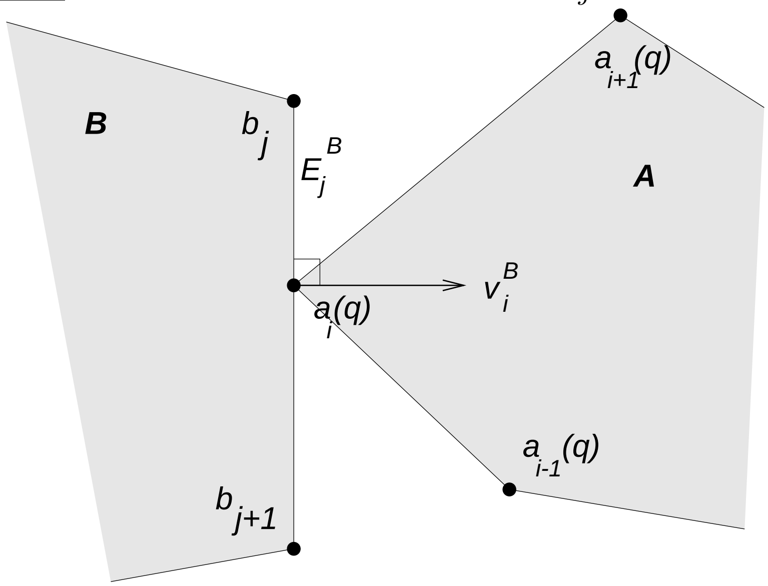
$$(a_{i+1} - a_i) \cdot \vec{\nu}_j^{\mathcal{B}} \geq 0$$

Feasibility (Applicability) Condition: $\mathrm{APPL}_{i,j}^{\mathcal{B}}(\mathbf{q})$

- conjunction of the two conditions above
- when $\mathbf{q}$ expressed as $(\mathcal{T}, \Theta)$, depends only on Θ (orientation)

Deriving Contact Constraints (cont'd)

Type B Contacts: vertex a_i of $\mathcal{A}$ with edge $E_j^{\mathcal{B}}$ of $\mathcal{B}$



Contact Condition

- when Type B contact occurs, the vectors $(a_i(\mathbf{q}) - b_j)$ (along edge) and $\vec{\nu}_j^{\mathcal{B}}(\mathbf{q})$ (normal to edge) are perpendicular

C-surfaces of Type B (where Type B contact occurs)

- as $\mathcal{A}$ is moved, but Type B contact is maintained, the configuration moves along a surface in $\mathcal{C}$, called a **C-surface** of type B (a 2D submanifold of $\mathcal{C}$)
- This C-surface is expressed by: $f_{i,j}^{\mathcal{B}}(\mathbf{q}) = 0$ with $f_{i,j}^{\mathcal{B}}(\mathbf{q}) = (a_i(\mathbf{q}) - b_j) \cdot \vec{\nu}_j^{\mathcal{B}}(\mathbf{q})$
- This C-surface partitions $\mathcal{C}$ into two halfspaces, and the C-obstacle $\mathcal{CB}$ lies completely within the closed halfspace determined by $f_{i,j}^{\mathcal{B}}(\mathbf{q}) \leq 0$.

C-constraint of Type B: $\mathrm{CONST}_{i,j}^{\mathcal{B}}(\mathbf{q}) \equiv \mathrm{APPL}_{i,j}^{\mathcal{B}}(\mathbf{q}) \Longrightarrow (f_{i,j}^{\mathcal{B}}(\mathbf{q}) \leq 0)$

- if $\mathrm{APPL}_{i,j}^{\mathcal{B}}(\mathbf{q})$ is FALSE, then $\mathrm{CONST}_{i,j}^{\mathcal{B}}(\mathbf{q})$ is TRUE
  i.e., ignore C-surface $f_{i,j}^{\mathcal{B}}(\mathbf{q}) = 0$

A C-obstacle predicate CB($\mathbf{q}$)

Definition:

$$\mathrm{CB}(\mathbf{q}) = \begin{cases} \textsc{true} & \text{if } \mathcal{A}(\mathbf{q}) \cap \mathcal{B} \neq \emptyset \text{, or equivalently, if } \mathbf{q} \in \mathcal{CB} \\ & \text{(i.e., overlap or contact occurs)} \\ \textsc{false} & \text{otherwise} \end{cases}$$

We can prove that

$$\mathrm{CB}(\mathbf{q}) \equiv \left[\wedge_{i,j}\mathrm{CONST}^{\mathcal{A}}_{i,j}(\mathbf{q})\right] \wedge \left[\wedge_{i,j}\mathrm{CONST}^{\mathcal{B}}_{i,j}(\mathbf{q})\right]$$

i.e., CB($\mathbf{q}$) = TRUE if and only if *every* CONST expression is TRUE.

ALGORITHM: Compute CB($\mathbf{q}$)

input: $\mathbf{q}$, convex polygons $\mathcal{A}$ and $\mathcal{B}$ (boundary representation)

output: TRUE if $\mathbf{q} \in \mathcal{CB}$, FALSE otherwise

1. For each edge E_i of $\mathcal{A}$ and vertex b_j of $\mathcal{B}$, evaluate $\mathrm{CONST}^{\mathcal{A}}_{i,j}(\mathbf{q})$.
   - if FALSE, then **return** FALSE ($\mathbf{q} \notin \mathcal{CB}$)
2. For each vertex a_i of $\mathcal{A}$ and edge e_j of $\mathcal{B}$, evaluate $\mathrm{CONST}^{\mathcal{B}}_{i,j}(\mathbf{q})$.
   - if FALSE, then **return** FALSE ($\mathbf{q} \notin \mathcal{CB}$)
3. **return** TRUE ($\mathbf{q} \in \mathcal{CB}$)

Complexity: $O(n_{\mathcal{A}} n_{\mathcal{B}})$

- number of C-constaints = $2n_{\mathcal{A}}n_{\mathcal{B}}$
- in worst-case evaluate them all

Proving the predicate CB($\mathbf{q}$) is correct

Assumption: $\mathcal{A}$ and $\mathcal{B}$ are convex polygons

We'll now prove that $\mathrm{CB}(\mathbf{q}) \equiv \left[\wedge_{i,j}\mathrm{CONST}^{\mathcal{A}}_{i,j}(\mathbf{q})\right] \wedge \left[\wedge_{i,j}\mathrm{CONST}^{\mathcal{B}}_{i,j}(\mathbf{q})\right]$

The following fact will be useful (Lemma 1, p. 112 in Latombe)

Lemma: *Convex polygons P and Q do not intersect if and only if one of them contains an edge e such that the other polygon lies entirely in the open outer halfspace (halfplane) of e.*

Theorem: *Let $\mathcal{A}$ and $\mathcal{B}$ be convex polygons, where the C-obstacle corresponding to $\mathcal{B}$ is $\mathcal{CB} = \{\mathbf{q} \in \mathcal{C} | \mathcal{A}(\mathbf{q}) \cap \mathcal{B} \neq \emptyset$. Then $\mathbf{q} \in \mathcal{CB} \Longleftrightarrow \mathrm{CB}(\mathbf{q})$, where $\mathrm{CB}(\mathbf{q})$ is as defined above.*

Proof:
Note that

$$\mathbf{q} \in \mathcal{CB} \Longleftrightarrow \mathrm{CB}(\mathbf{q})$$

is equivalent to

$$\mathbf{q} \notin \mathcal{CB} \Longleftrightarrow \neg\mathrm{CB}(\mathbf{q})$$

with

$$\neg\mathrm{CB}(\mathbf{q}) \equiv \left[\vee_{i,j}\neg\mathrm{CONST}^{\mathcal{A}}_{i,j}(\mathbf{q})\right] \vee \left[\vee_{i,j}\neg\mathrm{CONST}^{\mathcal{B}}_{i,j}(\mathbf{q})\right]$$

i.e., if any CONST is FALSE then $\neg\mathrm{CB}(\mathbf{q})$ is TRUE (so $\mathbf{q} \notin \mathcal{CB}$).

We'll prove the second expression.

Proof of CB($\mathbf{q}$) – case 1 (if)

Case 1 (if): $\mathbf{q} \notin \mathcal{CB} \Longrightarrow \neg$CB($\mathbf{q}$)

Let $\mathbf{q}$ be some configuration such that $\mathbf{q} \notin \mathcal{CB}$, i.e., $\mathcal{A}(\mathbf{q}) \cap \mathcal{B} = \emptyset$. We will show that some CONST is FALSE, which will imply that CB($\mathbf{q}$) is also FALSE.

• since $\mathbf{q} \notin \mathcal{CB}$, we have $\mathcal{A}(\mathbf{q}) \cap \mathcal{B} = \emptyset$.

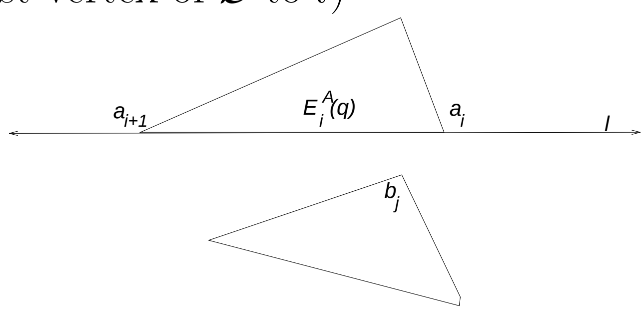
• since $\mathcal{A}$ and $\mathcal{B}$ are convex, and $\mathcal{A}(\mathbf{q}) \cap \mathcal{B} = \emptyset$, by the lemma we know that there exists a line l that contains an edge of one of the polygons and separates $\mathcal{A}(\mathbf{q})$ and $\mathcal{B}$, i.e., either

$$(i) \quad E_i^{\mathcal{A}}(\mathbf{q}) \subset l \quad \text{and} \quad \mathcal{B} \subset l^+$$

or

$$(ii) \quad E_i^{\mathcal{B}}(\mathbf{q}) \subset l \quad \text{and} \quad \mathcal{A}(\mathbf{q}) \subset l^+$$

case (i): there must exist some $b_j \in \mathcal{B}$ such that APPL$_{i,j}^{\mathcal{A}}(\mathbf{q})$ is TRUE (choose b_j to be closest vertex of $\mathcal{B}$ to l)



$$f_{i,j}^{\mathcal{A}}(\mathbf{q}) = \nu_i^{\mathcal{A}}(\mathbf{q}) \cdot [b_j - a_i(\mathbf{q})] > 0$$

so CONST$_{i,j}^{\mathcal{A}}(\mathbf{q}) \equiv$ APPL$_{i,j}^{\mathcal{A}}(\mathbf{q}) \Longrightarrow [f_{i,j}^{\mathcal{A}}(\mathbf{q}) \leq 0]$ is FALSE
and $\neg$CONST$_{i,j}^{\mathcal{A}}(\mathbf{q})$ is TRUE
and $\neg$CB($\mathbf{q}$) is TRUE

case (ii): similar

Q.E.D. (of Case 1)

Proof of $\mathrm{CB}(\mathbf{q})$ – case 2 (only if)

Case 2 (only if): $\neg\mathrm{CB}(\mathbf{q}) \Longrightarrow \mathbf{q} \notin \mathcal{CB}$

Suppose $\neg\mathrm{CB}(\mathbf{q})$ is TRUE.

- then some $\mathrm{CONST}_{i,j}^{\mathcal{A}}(\mathbf{q})$ is FALSE, i.e., for some i, j we have either

$$
\begin{array}{rl}
(i) & \mathrm{APPL}_{i,j}^{\mathcal{A}}(\mathbf{q}) = \textsc{true} \quad \text{and} \quad f_{i,j}^{\mathcal{A}}(\mathbf{q}) > 0 \\
\text{or} & \\
(ii) & \mathrm{APPL}_{i,j}^{\mathcal{B}}(\mathbf{q}) = \textsc{true} \quad \text{and} \quad f_{i,j}^{\mathcal{B}}(\mathbf{q}) > 0
\end{array}
$$

In either case, $f_{i,j}(\mathbf{q}) = 0$ is a line that separates the two polygons (and contains an edge of one of them).

so, by the lemma, $\mathcal{A}(\mathbf{q}) \cap \mathcal{B} = \emptyset$

and $\mathbf{q} \notin \mathcal{CB}$

Q.E.D. (of Case 2, and Theorem)

Non-Convex Polygons

Suppose we decompose $\mathcal{A}$ and $\mathcal{B}$ into convex polygons

$$\mathcal{A} = \cup_k \mathcal{A}_k \qquad \text{and} \qquad \mathcal{B} = \cup_l \mathcal{B}_l$$

and define C-obstacle $\mathcal{CB}_{k,l}$ as

$$\mathcal{CB}_{k,l} = \{\mathbf{q} \in \mathcal{C} | A_k(\mathbf{q}) \cap \mathcal{B}_l \neq \emptyset\}$$

and

$$\mathcal{CB} = \cup_{k,l} \mathcal{CB}_{k,l}$$

and thus

$$\mathbf{q} \in \mathcal{CB} \Longleftrightarrow \vee_{k,l} \mathbf{q} \in \mathcal{CB}_{k,l}$$

i.e., $\mathbf{q}$ is in at last one of the C-obstacles $\mathcal{CB}_{k,l}$

Then, using the predicate $\mathrm{CB}_{k,l}(\mathbf{q})$ for each pair of polygons in the convex decomposition of $\mathcal{A}$ and $\mathcal{B}$, we get the following:

$$\mathrm{CB}(\mathbf{q}) \equiv \vee_{k,l} \mathrm{CB}_{k,l}(\mathbf{q})$$

i.e., $\mathrm{CB}(\mathbf{q})$ is TRUE if at least one of the $\mathrm{CB}_{k,l}(\mathbf{q})$ is TRUE

decomposing a non-convex polygon into convex pieces:

- standard problem in computational geometry
- covering (allows overlap among convex pieces)
- partitioning (no overlap among convex pieces)
- can be done in $O(n \log n)$ time
- *optimal* means minimal number of pieces
  – optimal decomposition NP-Hard if polygon has holes

Boundary Representation of $\mathcal{CB}$ in $\mathbb{R}^2$ (translation only)

Definitions:

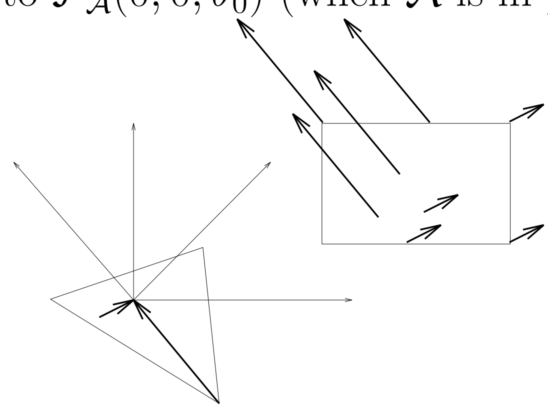
- $\mathcal{A}(x, y, \theta)$ denotes the set of points in $\mathcal{A}$ when $\mathcal{A}$ is at position (x, y, θ)
- $X \ominus Y = \{z | \exists x \in X, \exists y \in Y : z = x - y\}$ ($\ominus$ is Minkowski operator for set difference)

C-obstacles: for a fixed value θ_0 of θ, we can show

$$\begin{aligned} \mathcal{CB}_{\theta_0} &= \mathcal{B} \ominus \mathcal{A}(0, 0, \theta_0) \\ &= \{x \in \mathcal{W} | \exists b \in B, \exists a \in \mathcal{A}(0, 0, \theta_0) : x = b - a\} \end{aligned}$$

<u>*What is* $-\mathbf{a}$*?*</u>

The vector from $a \in \mathcal{A}$ to $\mathcal{F}_{\mathcal{A}}(0, 0, \theta_0)$ (when $\mathcal{A}$ is in position $(0, 0, \theta_0)$)



So, one way to get $\mathcal{CB}$ is to look at *all pairs* of points $b \in \mathcal{B}$ and $a \in \mathcal{A}(0, 0, \theta_0)$ – not very efficient...

Useful Fact

Let θ_0 be a *general orientation* of convex $\mathcal{A}$ relative to convex $\mathcal{B}$, i.e., no edge of $\mathcal{A}(x, y, \theta_0)$ is parallel to an edge of $\mathcal{B}$.

Then, the *vertices* of $\mathcal{CB}_{\theta_0}$ can be obtained as follows:

- if $\mathrm{APPL}^{\mathcal{A}}_{i,j}(0, 0, \theta_0)$ is TRUE: (recall APPL only depends on θ_0, not x, y)

$$b_j - a_i(0, 0, \theta_0)$$
$$b_j - a_{i+1}(0, 0, \theta_0)$$

  i.e., look at both endpoints of edge $E^{\mathcal{A}}_i$

- if $\mathrm{APPL}^{\mathcal{B}}_{i,j}(0, 0, \theta_0)$ is TRUE:

$$b_j - a_i(0, 0, \theta_0)$$
$$b_{j+1} - a_i(0, 0, \theta_0)$$

  i.e., look at both endpoints of edge $E^{\mathcal{B}}_i$

An Algorithm to compute $\mathcal{CB}_{\theta_0}$

Note: since $\mathcal{A}$ is convex, the counter-clockwise angle between two successive vectors $-\nu_i^{\mathcal{A}}(\theta_0)$ and $-\nu_{i+1}^{\mathcal{A}}(\theta_0)$ is smaller than π (vertices numbered counter-clockwise) – also true for $\mathcal{B}$ since it is convex

Thus, if θ_0 is a general orientation of $\mathcal{A}$ relative to $\mathcal{B}$:

- $\text{APPL}_{i,j}^{\mathcal{A}}(\theta_0)$ is TRUE **iff** $-\nu_i^{\mathcal{A}}(\theta_0)$ points between $\nu_{j-1}^{\mathcal{B}}(\theta_0)$ and $\nu_j^{\mathcal{B}}(\theta_0)$
- $\text{APPL}_{i,j}^{\mathcal{B}}(\theta_0)$ is TRUE **iff** $\nu_j^{\mathcal{B}}(\theta_0)$ points between $-\nu_{i-1}^{\mathcal{A}}(\theta_0)$ and $-\nu_i^{\mathcal{A}}(\theta_0)$

ALGORITHM: COMPUTE $\mathcal{CB}_{\theta_0}$

1. sort the vectors $\nu_1^{\mathcal{B}}(\theta_0), \ldots, \nu_{n_{\mathcal{B}}}^{\mathcal{B}}(\theta_0), -\nu_1^{\mathcal{A}}(\theta_0), \ldots, -\nu_{n_{\mathcal{A}}}^{\mathcal{A}}(\theta_0)$

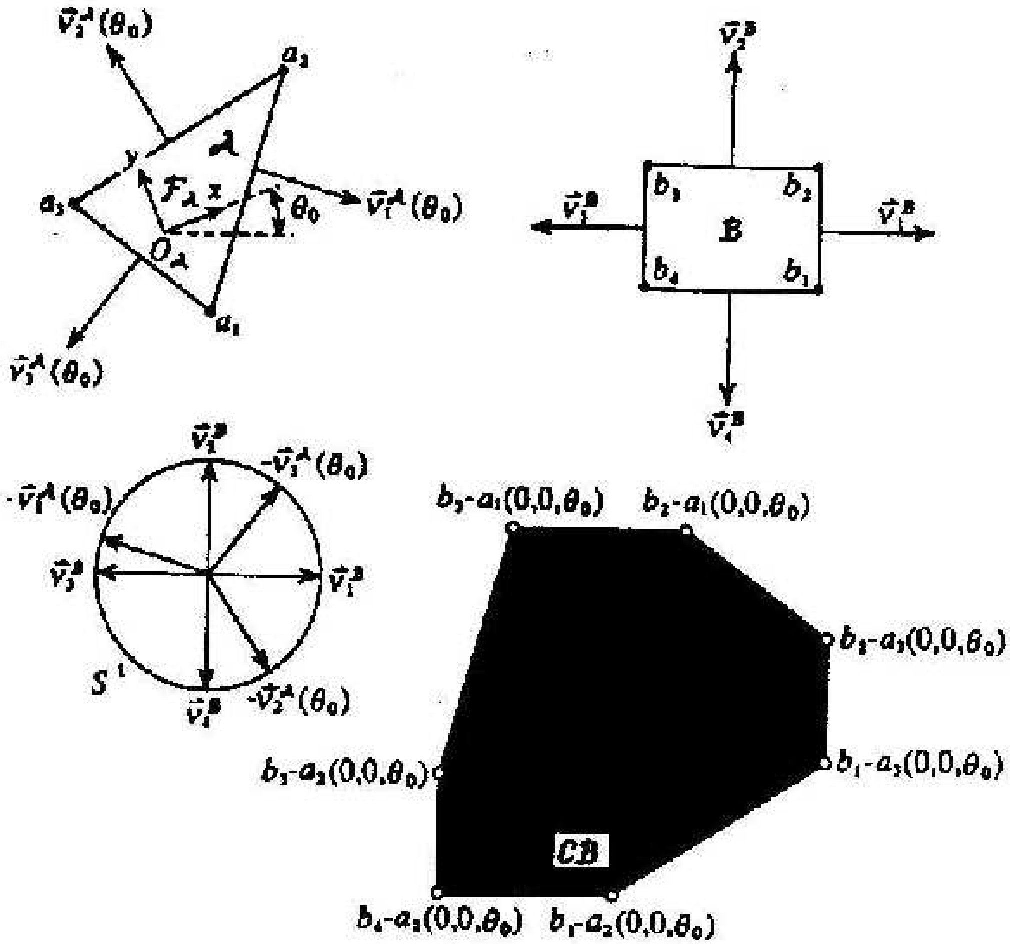
   note: takes $O(n_{\mathcal{A}} + n_{\mathcal{B}})$ time (merge two sorted lists)

2. scan the list and construct the vertices of $\mathcal{CB}_{\theta_0}$ (there are at most $n_{\mathcal{B}} + n_{\mathcal{A}}$)

   e.g., if we're at $-\nu_i^{\mathcal{A}}(\theta_0)$ and it points between $\nu_{j-1}^{\mathcal{B}}(\theta_0)$ and $\nu_j^{\mathcal{B}}(\theta_0)$, then construct the two vertices: $b_j - a_i(0, 0, \theta_0)$ and $b_j - a_{i+1}(0, 0, \theta_0)$.

   note: takes $O(n_{\mathcal{A}} + n_{\mathcal{B}})$ time

Example of Algorithm



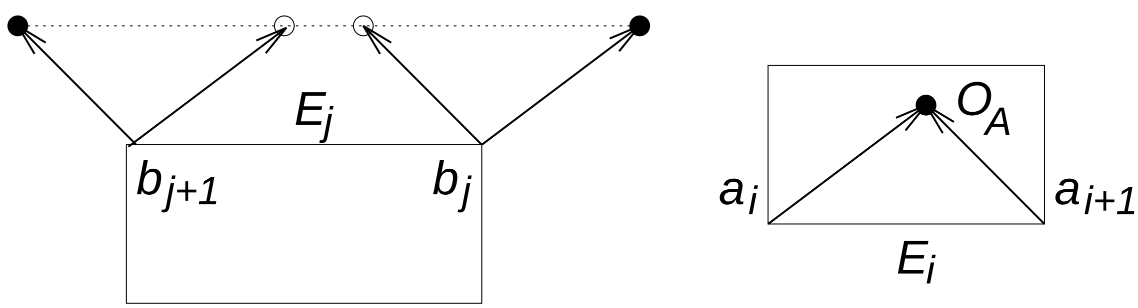
Critcal Orientations of $\mathcal{A}$ (not general position)

What if an edge of $\mathcal{A}(x, y, \theta_0)$ is parallel to an edge of $\mathcal{B}$?

Suppose edge $E_i^{\mathcal{A}}$ of $\mathcal{A}(x, y, \theta_0)$ is parallel to edge $E_j^{\mathcal{B}}$, and contact between the two edges is feasible.

Then, the four points are potential vertices of $\mathcal{CB}_{\theta_0}$:

$$\begin{array}{ll} b_j - a_i(0,0,\theta_0) & b_{j+1} - a_i(0,0,\theta_0) \\ b_j - a_{i+1}(0,0,\theta_0) & b_{j+1} - a_{i+1}(0,0,\theta_0) \end{array}$$



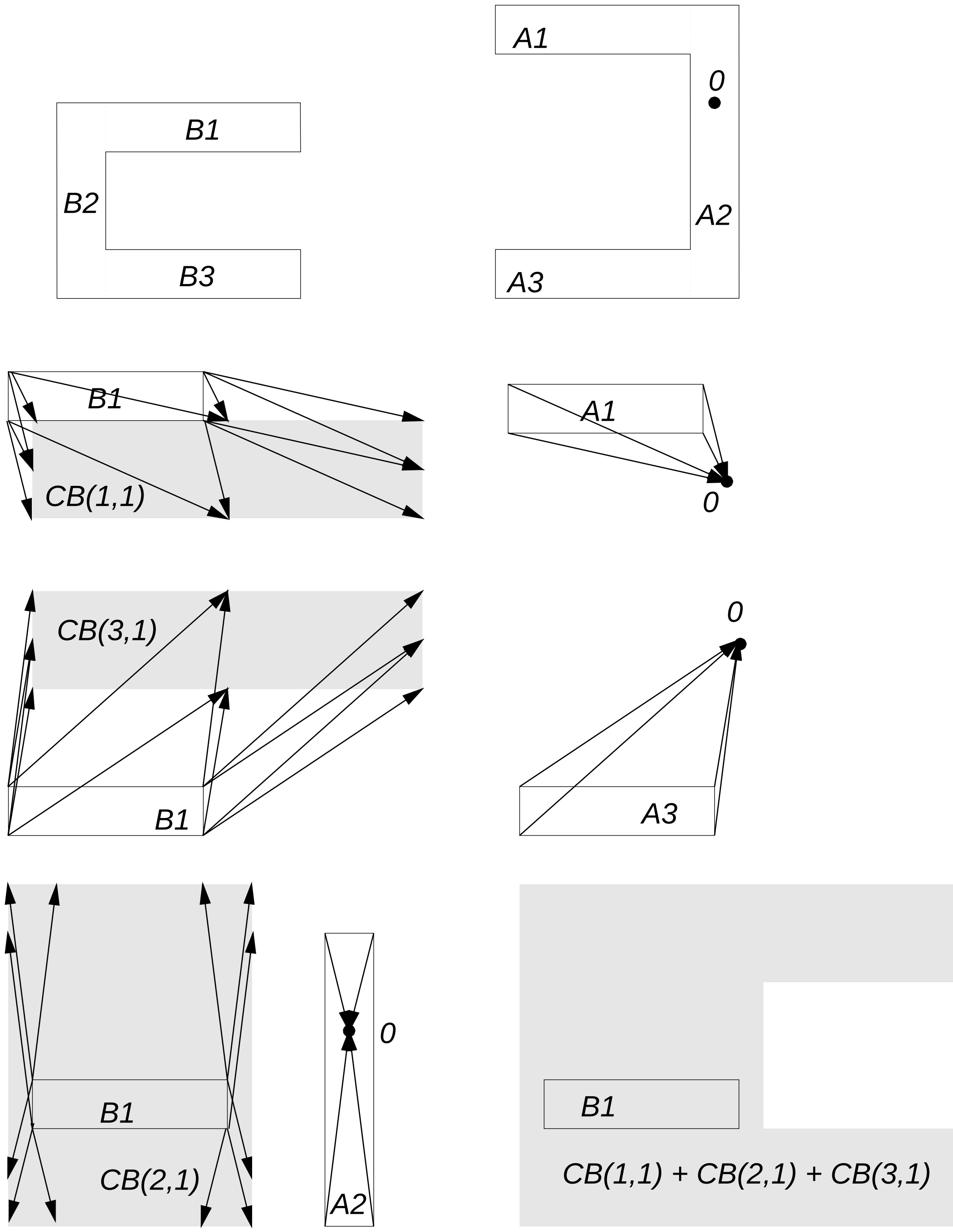
However, these points are colinear, so only the first and last are vertices of $\mathcal{CB}$.

Modifying the Algorithm for critical orientations:
This situation can easily be recognized since $-\nu_i^{\mathcal{A}}(\theta_0)$ is equal to $\nu_j^{\mathcal{B}}$ (discovered during scan in step 2).

The modified algorithm still runs in $O(n_{\mathcal{A}} + n_{\mathcal{B}})$ time.

Non-convexity: As before, decompose $\mathcal{A}$ and $\mathcal{B}$ into convex pieces and compute $\mathcal{CB}_{k,l}$ and take union

Example: $\mathcal{CB}$ union of CB_i (decompose into convex pieces)

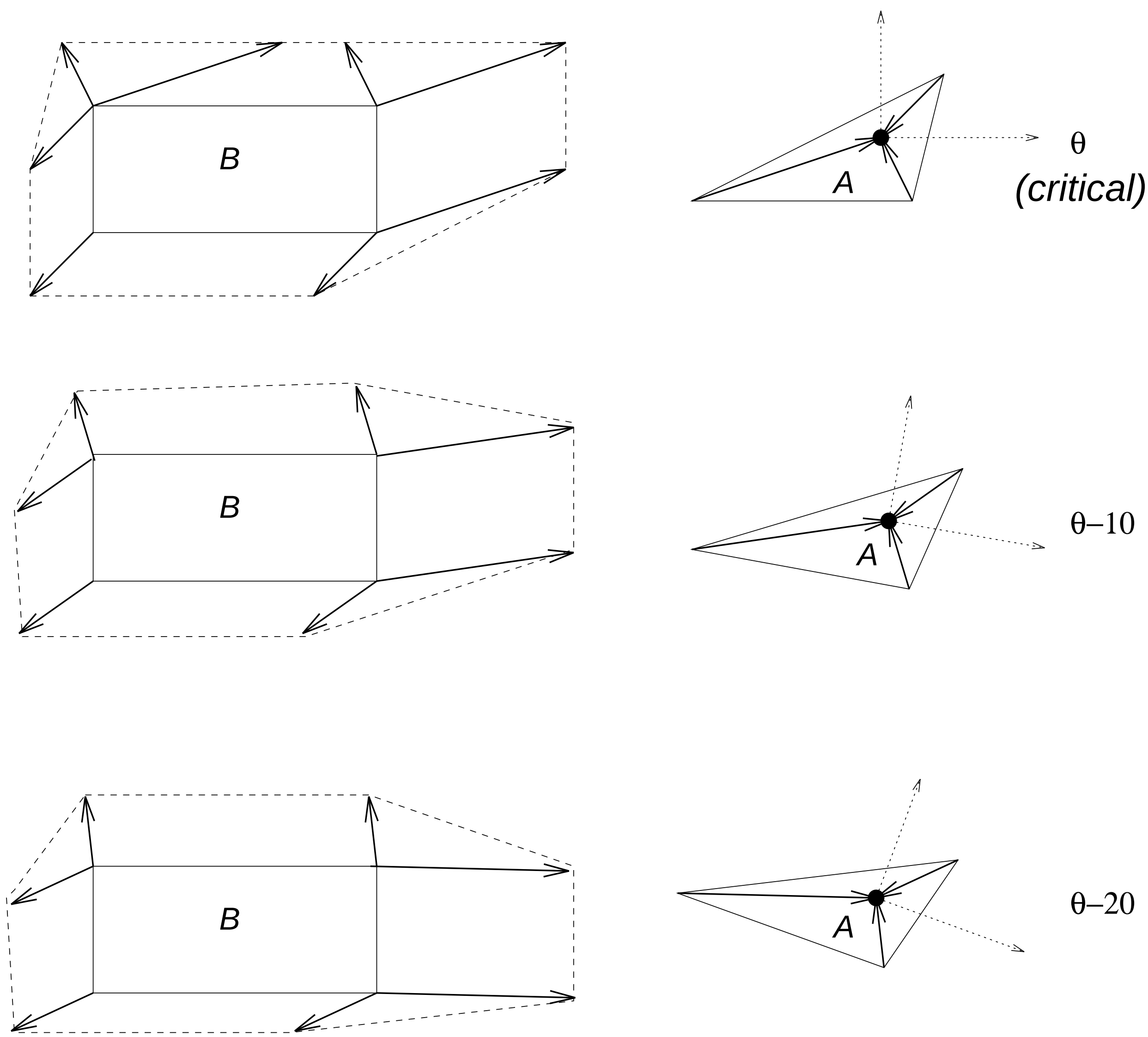


- The portion of $\mathcal{CB}$ due to $B1$ is $\mathcal{CB}_{A1,B1} \cup \mathcal{CB}_{A2,B1} \cup \mathcal{CB}_{A3,B1}$.
- The complete $\mathcal{CB} = \cup_{Ai,Bj} \mathcal{CB}(i,j)$, i.e., the above process must be carried out for $B2$ and $B3$ as well.

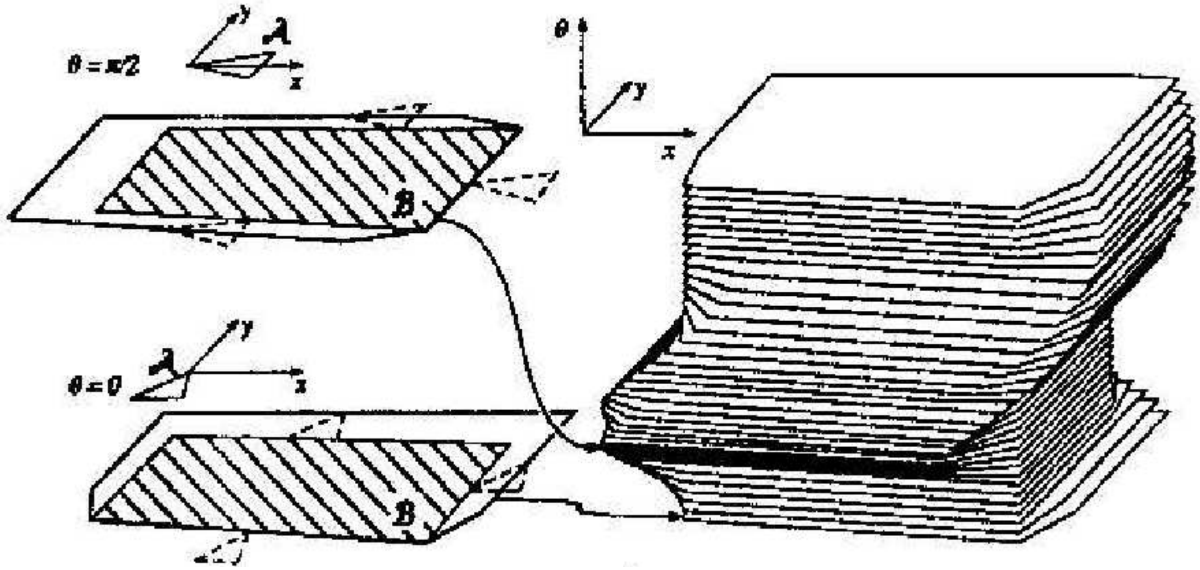
Translation and Rotation – 3-dimensional $\mathcal{CB}$

What does $\mathcal{CB}$ look like when $\mathcal{A}$ can translate and rotate?

- three-dimensional $\mathbb{R}^2 \times 2\pi$: i.e., $(x, y) \in \mathbb{R}^2$, and $\theta \in [0, 2\pi)$
- for fixed θ_0, have $\mathcal{CB}_{\theta_0}$ as before (a convex polygon)
- *vertices* (*edges*) of $\mathcal{CB}_{\theta_0}$ are contained in *edges* (*faces*) of $\mathcal{CB}$
- *vertices* of $\mathcal{CB}$ appear at *critical orientations* of $\mathcal{A}$ (e.g., when edges of $\mathcal{A}$ and $\mathcal{B}$ are parallel) – edges/faces of $\mathcal{CB}$ appear/disappear here

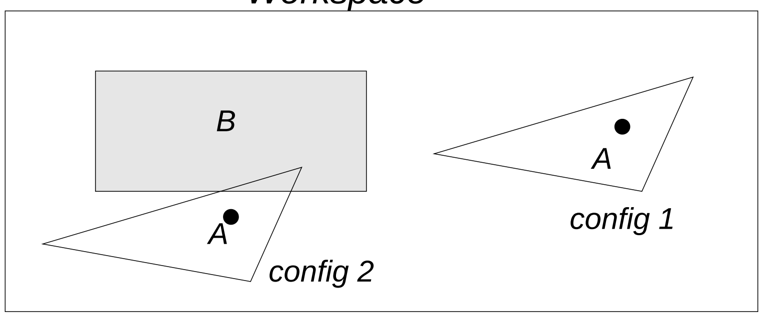


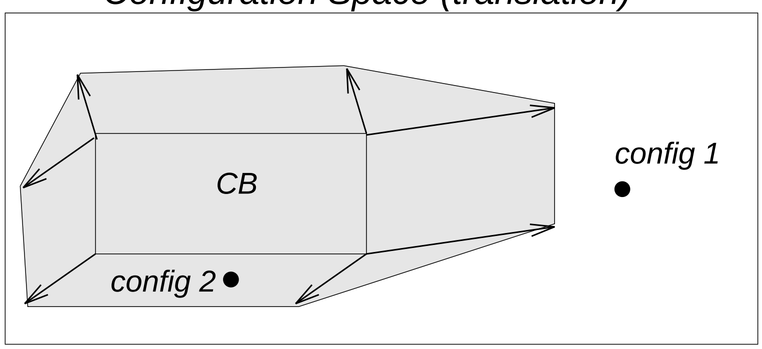
Example three-dimensional $\mathcal{CB}$



Summary – C-space and $\mathcal{CB}$

- Represent robot configuration as a point in $\mathcal{C}$ (C-space). Obstacles in $\mathcal{W}$ map to C-obstacles in $\mathcal{C}$

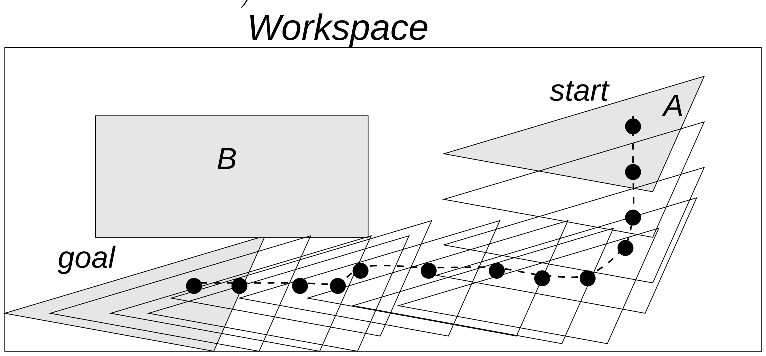


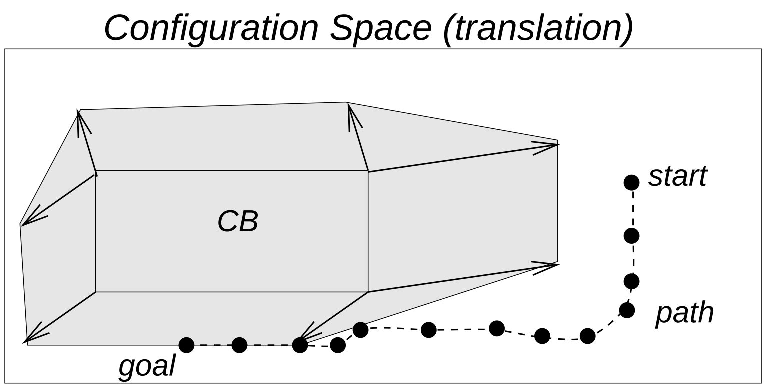


- C-constaints when $\mathcal{A}$ and $\mathcal{B}$ are (convex) polygons:
  - compute $\mathrm{APPL}_{i,j}, f_{i,j}, \mathrm{CONST}_{i,j}$ for type A and B contacts
  - $O(n_{\mathcal{A}}+n_{\mathcal{B}})$ algorithm to compute boundary of $\mathcal{CB}$ in translational case
  - C-constraints can be represented in parameterized form
  - $O(n_{\mathcal{A}}^3 n_{\mathcal{B}}^3 \log n_{\mathcal{A}} n_{\mathcal{B}})$ algorithm for translation and rotation
  - $\mathcal{CB}$ is three-dimensional volume bounded by patches of C-surfaces (which are 'ruled' surfaces, i.e., composed of lines)
- when $\mathcal{A}$ and $\mathcal{B}$ are 3D or more 'complicated' (algebraic surfaces), algorithms to explicitly compute $\mathcal{CB}$ usually not practical (eg. Collins decomposition)
  - need to use other approaches, e.g., heuristics, randomization

What's Next...

We now know how to transform motion planning problem for a dimensioned robot $\mathcal{A}$ in the workspace $\mathcal{W}$ to the problem of finding a path in $\mathcal{C}$ connecting two points (configurations of $\mathcal{A}$) that avoids C-obstacles.





Next, we will look at some basic solution methods.

Two Step Methods:

- PREPROCESSING: capture connectivity of $\mathcal{C}_{free}$ by graph or function
- QUERY PROCESSING: search the graph for a path

Three families of methods:

- CELL DECOMPOSITION (exact and approximate)
- ROADMAP (deterministic and randomized)
- POTENTIAL FIELD (deterministic and randomized)