

Exact Cell Decomposition Methods

Acknowledgement: Parts of these course notes are based on notes from courses given by Jean-Claude Latombe at Stanford University (and Chapter 5 in his text *Robot Motion Planning*, Kluwer, 1991), O. Burçhan Bayazit at Washington University in St. Louis. Seth Hutchinson at the University of Illinois at Urbana-Champaign, and Leo Joskowicz at Hebrew University.

Cell Decomposition Methods

PREPROCESSING:

- represent \mathcal{C}_{free} as a collection of *cells* (connected regions of \mathcal{C}_{free})
⇒ planning between configurations in the same cell should be 'easy'
- build *connectivity graph* representing adjacency relations between cells
⇒ cells adjacent if can move directly between them

QUERY PROCESSING:

1. locate cells k_{init} and k_{goal} containing start and goal configurations
2. search the connectivity graph for a 'channel' or sequence of adjacent cells connecting k_{init} and k_{goal}
3. find a path that is contained in the channel of cells

Two major variants of methods:

- *exact cell decomposition:*
 - set of cells exactly covers \mathcal{C}_{free}
 - complicated cells with irregular boundaries (contact constraints)
 - harder to compute
- *approximate cell decomposition:*
 - set of cells approximately covers \mathcal{C}_{free}
 - simpler cells with more regular boundaries
 - easier to compute

Exact Cell Decomposition Methods

Idea: decompose \mathcal{C}_{free} into a collection \mathcal{K} of non-overlapping *cells* such that the union of all the cells *exactly* equals the free C-space, i.e., $\mathcal{C}_{free} = \cup_{k \in \mathcal{K}} k$

Cell Characteristics:

- geometry of cells should be simple so that it is easy to compute a path between any two configurations in a cell
- it should be pretty easy to test the adjacency of two cells, i.e., whether they share a boundary
- it should be pretty easy to find a path crossing the portion of the boundary shared by two adjacent cells

Thus, cell boundaries correspond to 'criticalities' in \mathcal{C} , i.e., something changes when a cell boundary is crossed. No such criticalities in \mathcal{C} occur within a cell.

The main drawback of exact cell decomposition methods is the difficulty of computing the decomposition ...

- a simple, efficient sweep-line approach works when $\mathcal{C} = \mathbb{R}^2$ (no rotation) and \mathcal{CB} is a polygonal region, i.e., each C-obstacle \mathcal{CB}_i is a polygon
 - a bit harder, but still tractable, when $\mathcal{C} = \mathbb{R}^2 \times SO(2)$ (rotation)
- a method is known for the general case – impractical, but important as existence proof for a general path planning method

Exact Cell Decomposition when $\mathcal{C} = \mathbb{R}^2$

Assumptions: the robot \mathcal{A} and all obstacles \mathcal{B}_i are polygons, and \mathcal{C}_{free} is bounded (not really necessary, just easier). Note, this implies that \mathcal{CB} is a polygonal region.

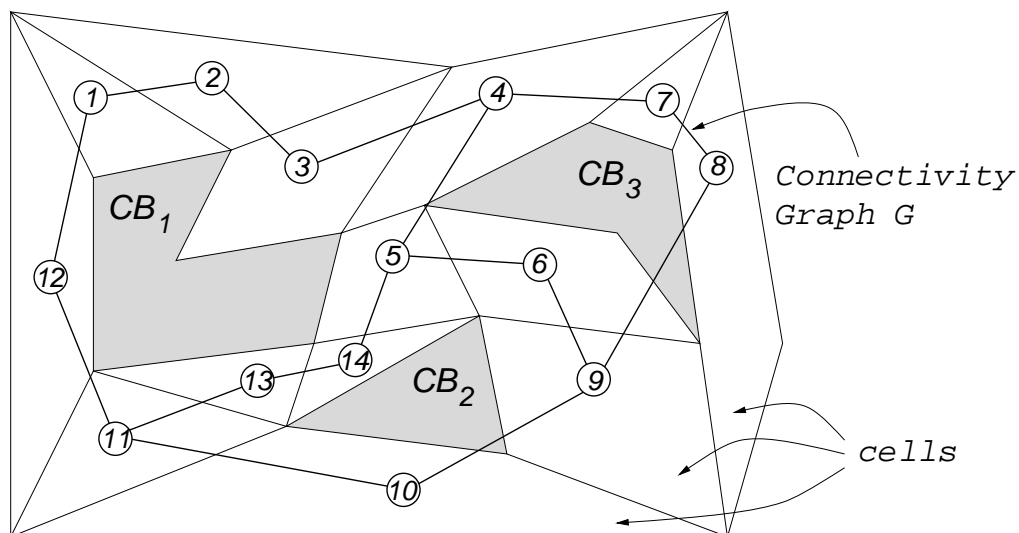
Definitions:

A **convex polygonal decomposition** \mathcal{K} of \mathcal{C}_{free} is a finite collection of convex polygons, called **cells**, such that the interiors of any two cells do not intersect and the union of all cells is \mathcal{C}_{free} .

Two cells $k, k' \in \mathcal{K}$ are **adjacent** iff $k \cap k'$ is a line segment of non-zero length (i.e., not a single point)

The **connectivity graph** associated with a convex polygonal decomposition \mathcal{K} of \mathcal{C}_{free} is an undirected graph G where

- nodes in G correspond to cells in \mathcal{K}
- nodes connected by edge in G iff corresponding cells adjacent in \mathcal{K}

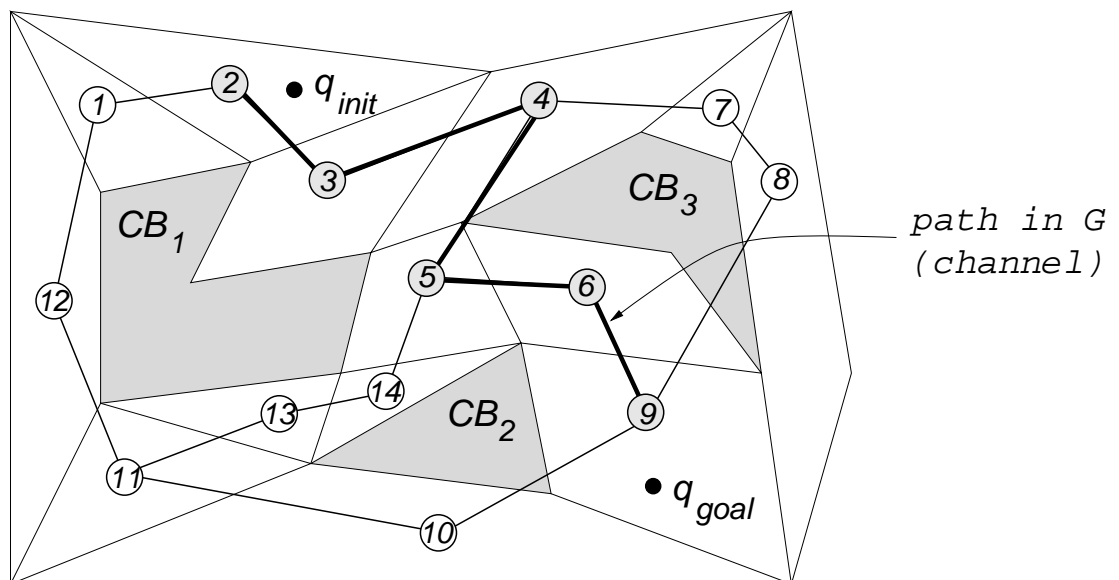


Path Planning with a Convex Polygonal Decomposition

input: configurations \mathbf{q}_{init} and \mathbf{q}_{goal} , and \mathcal{CB} which is a polygonal region

output: a path in \mathcal{C}_{free} connecting \mathbf{q}_{init} and \mathbf{q}_{goal}

1. Build \mathcal{K} , the convex polygonal decomposition of \mathcal{CB}
2. Construct the connectivity graph G of \mathcal{K}
3. locate the cells k_{init} and k_{goal} in \mathcal{K} containing \mathbf{q}_{init} and \mathbf{q}_{goal}
4. find a path in G between the nodes corresponding to k_{init} and k_{goal}
 - corresponds to a sequence of cells forming a **channel** in \mathcal{C}_{free}
5. find a free path from \mathbf{q}_{init} to \mathbf{q}_{goal} in the channel



Getting a path from a channel

Let k_1, k_2, \dots, k_p denote the channel in \mathcal{C}_{free} , where $k_1 = k_{init}$ and $k_p = k_{goal}$.

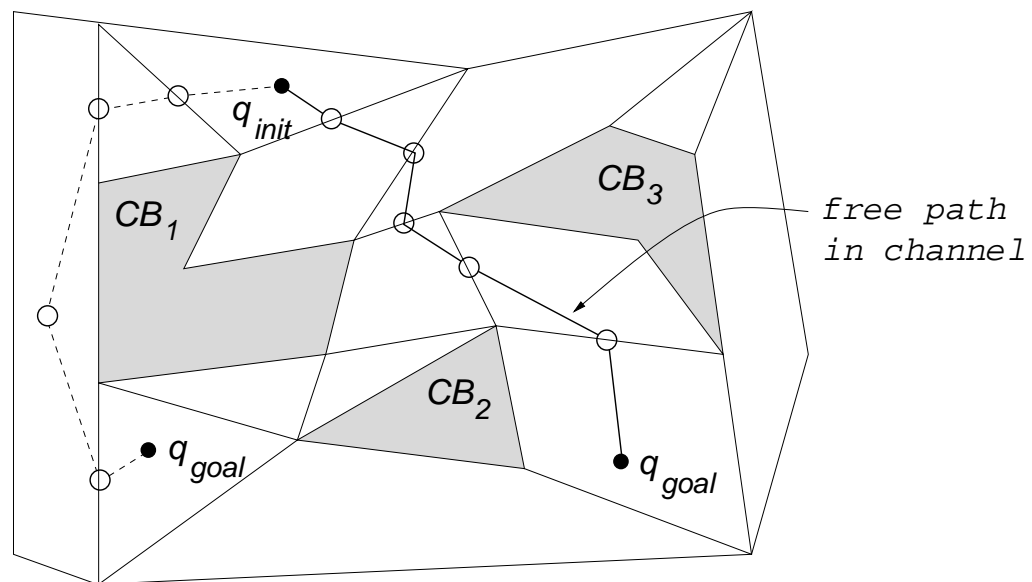
Goal: find a path (1-dimensional curve) from \mathbf{q}_{init} to \mathbf{q}_{goal} that is contained in the interior of the channel (and thus in \mathcal{C}_{free})

Note: since each k_i is convex, the straight line between any two configurations (points) in k_i lies in k_i (and thus in \mathcal{C}_{free})

Idea: use midpoints of cell boundaries as crossing points between cells

Let Q_i denote the midpoint of the line segment common to both k_i and k_{i+1} in the channel, i.e., Q_i is the midpoint of $\partial k_i \cap \partial k_{i+1}$

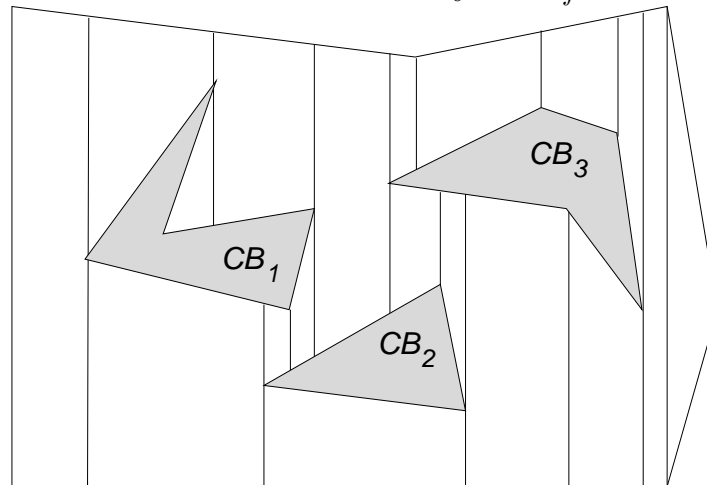
Then, $\mathbf{q}_{init}, Q_1, Q_2, \dots, Q_{p-1}, \mathbf{q}_{goal}$ is a path in the channel



Caveat: if $Q_{i-1}Q_i$ lies in ∂k_i , then the path above is only semi-free (i.e., it has contact). If we want a free path, then we need to insert another point between Q_{i-1} and Q_i that lies in the interior of k_i

Building \mathcal{K} – Trapezoidal Decomposition

Basic Idea: at every vertex of \mathcal{CB} , extend a vertical line up and down in \mathcal{C}_{free} until it touches a C-obstacle or the boundary of \mathcal{C}_{free}



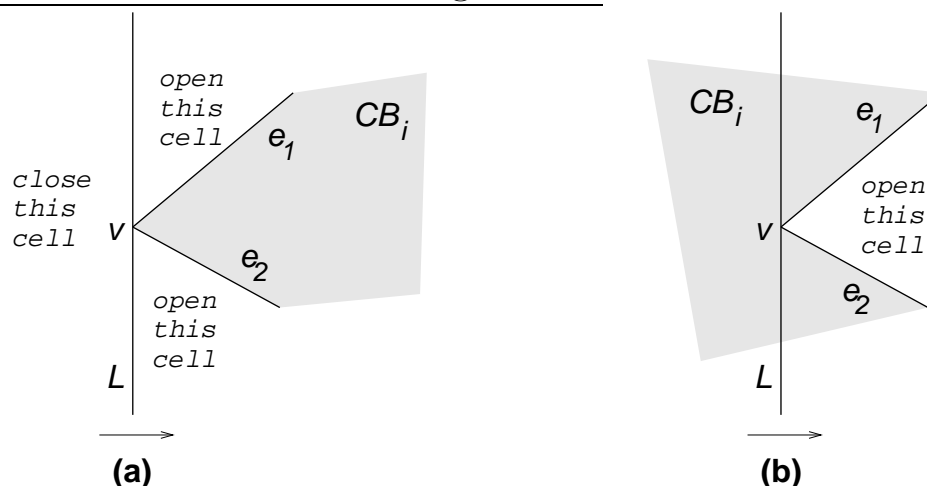
SWEEPLINE ALGORITHM: BUILD TRAPEZOIDAL DECOMPOSITION

input: boundary representation of \mathcal{CB}

output: trapezoidal decomposition of \mathcal{CB}

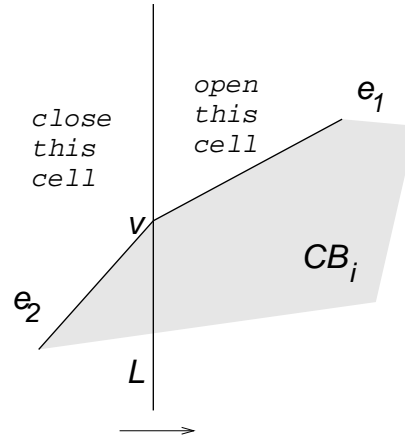
1. sort vertices of \mathcal{CB} by x -coordinate (assume general position–unique x s)
2. 'sweep' a vertical line L from left to right, stopping at each vertex v in \mathcal{CB}
note: v is incident on exactly two edges e_1 and e_2 in \mathcal{CB}

case 1: e_1 and e_2 both lie to the right of L

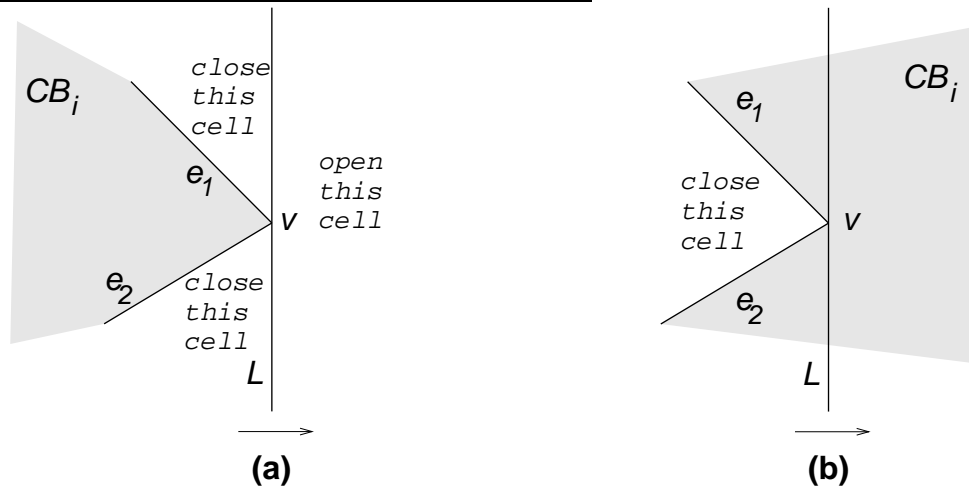


Building \mathcal{K} – Trapezoidal Decomposition (cont'd)

case 2: exactly one of e_1 and e_2 lies to the right of L



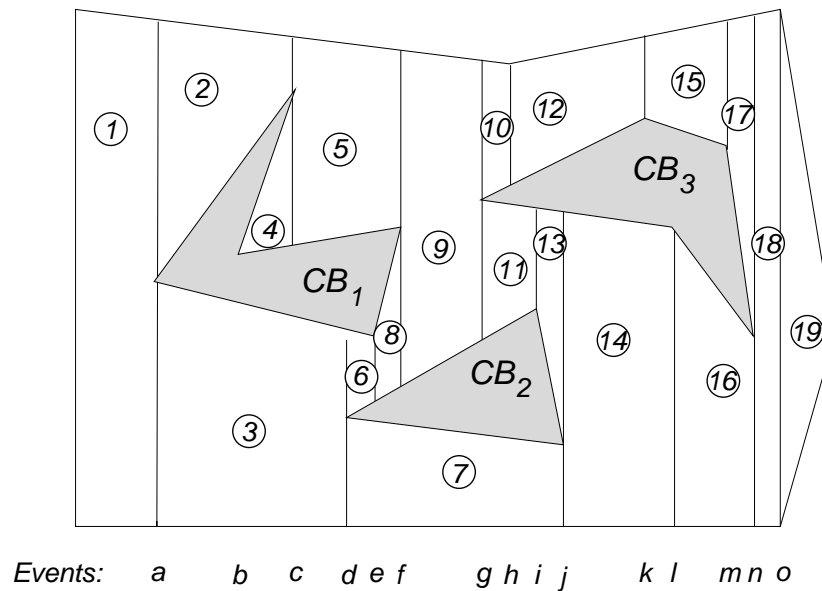
case 3: e_1 and e_2 both lie to the left of L



Complexity: $O(n \log n)$, where n is number of vertices in \mathcal{CB}

- initial sorting – $O(n \log n)$
- if maintain a sorted list of cells currently intersected by L , then each 'event' processed in $O(\log n)$ time (find cell containing v and update L 's list of intersected cells) – $O(n \log n)$ time for all events
- can determine adjacency relations between cells (and G) during sweep (and also cells containing \mathbf{q}_{init} and \mathbf{q}_{goal})

Ex: Trapezoidal Decomposition Sweepline Algorithm



event a (case 1(a)):

- just prior to event a , L intersects cell 1
- at event a we close cell 1 and open cells 2 and 3
- just after event a , L intersects cells 2 and 3 (in that order)

event b (case 1(b)):

- just prior to event b , L intersects cells 2 and 3 (in that order)
- at event b we open cell 4
- just after event b , L intersects cells 2, 4, and 3 (in that order)

event c (case 3(a)):

- just prior to event c , L intersects cells 2, 4, and 3 (in that order)
- at event c we close cells 2 and 4 and open cell 5
- just after event c , L intersects cells 5 and 3 (in that order)

event e (case 2):

- just prior to event e , L intersects cells 5, 6, and 7 (in that order)
- at event e we close cell 6 and open cell 8
- just after event e , L intersects cells 5, 8, and 7 (in that order)

Summary – Convex Polygonal Decomposition when $\mathcal{C} = \mathbb{R}^2$

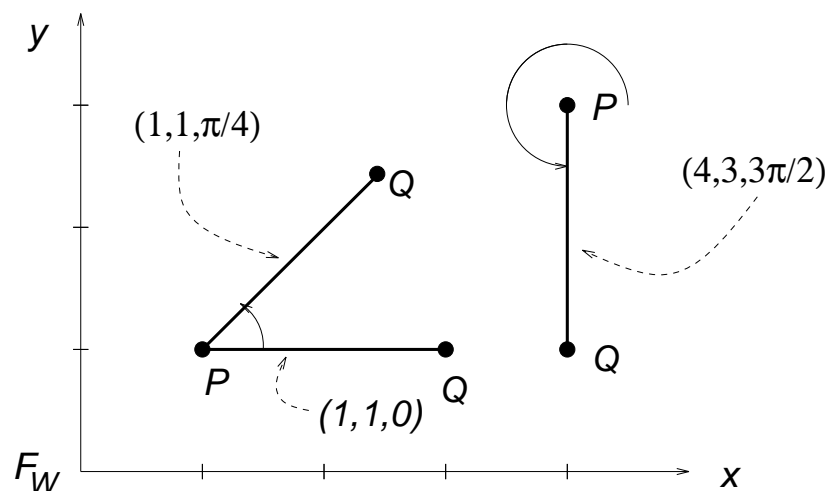
When $\mathcal{C} = \mathbb{R}^2$ and \mathcal{CB} is a polygonal region:

- the trapezoidal decomposition is an exact cell decomposition of \mathcal{C}_{free}
- trapezoidal decomposition can be built in $O(n \log n)$ time [Chazelle, 1987]
- trapezoidal decomposition *is not* an optimal convex decomposition (i.e., it does not have a minimal number of cells)
- it is NP-Hard to compute an optimal convex decomposition of a polygon (\mathcal{C}_{free}) with holes (\mathcal{CB}_i) [Lingas, 1982]
- the trapezoidal (vertical) decomposition can be extended to the case when $\mathcal{C} = \mathbb{R}^3$ – use sweep-plane

Exact Cell Decomposition for moving a ladder in the plane

We'll now consider how to compute an exact cell decomposition of \mathcal{C}_{free} for a harder, but still very particular case:

- the robot \mathcal{A} is a line segment of length d (called a *ladder*, *bar*, or *rod*) which can translate and rotate
- the workspace $\mathcal{W} = \mathbb{R}^2$, and obstacles are polygons so \mathcal{B} is a polygonal region (we'll assume \mathcal{B} is a manifold with boundary)
- the configuration space $\mathcal{C} = \mathbb{R}^2 \times [0, 2\pi)$



The Robot \mathcal{A} : is a line segment PQ of length d

- P is \mathcal{A} 's reference point (i.e., $P = \mathcal{O}_{\mathcal{A}}$, origin of $\mathcal{F}_{\mathcal{A}}$)
- the x-axis of $\mathcal{F}_{\mathcal{A}}$ coincides with the line containing PQ
- parameterize configurations of \mathcal{A} by (x, y, θ)
 - $(x, y) \in \mathbb{R}^2$ – position of P
 - $\theta \in [0, 2\pi)$ – offset of PQ from \mathcal{F}_W 's x-axis (horizontal)

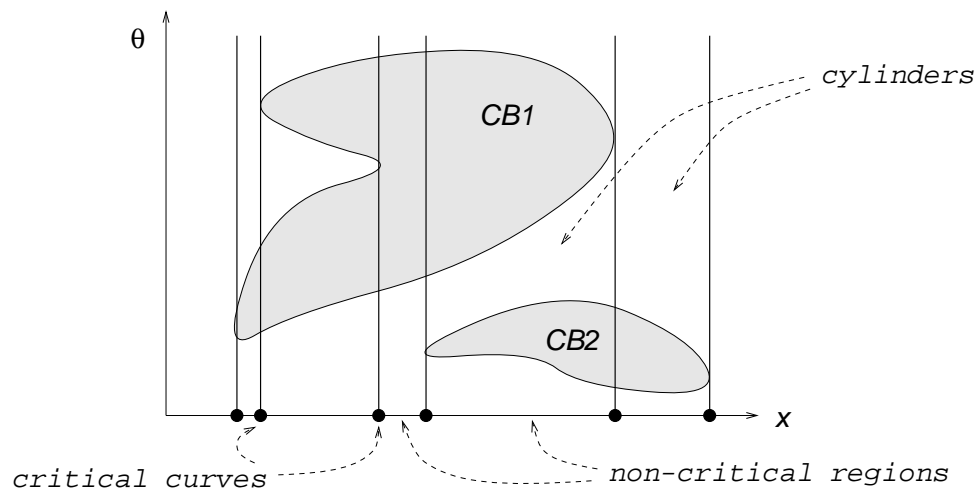
The Critical Curve Method of Schwartz and Sharir

The algorithm we'll look at uses the *critical curve* technique and is due to Schwartz and Sharir (more efficient methods exist, but this is a nice example)

Basic Idea:

- decompose the set of positions of P (the x-y plane corresponding to positions of \mathcal{A} 's origin) into two-dimensional **non-critical regions**
 - the 'structure' of the C-obstacles is 'constant' above (in θ direction) a non-critical region, i.e., the set of C-surfaces intersected by a line perpendicular to x-y plane does not change in a non-critical region
 - boundaries (in x-y plane) of non-critical regions are **critical curves**
- 'lift' 2d non-critical regions into 3d cells (**cylinders**)
- compute adjacency relationships between cylinders (connectivity graph G)

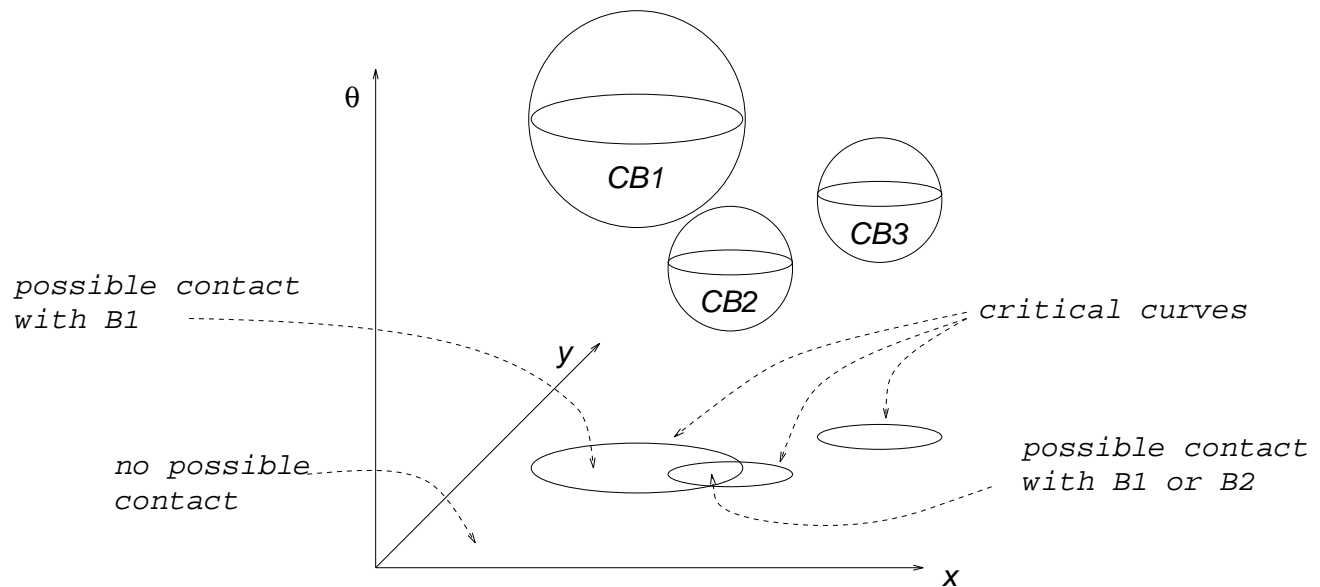
An example with two-dimensional C-space



- critical 'curves' are points on x -axis
- non-critical 'regions' are intervals on x -axis
- 'cylinders' are rectangles

Critical Curves for three-dimensional \mathcal{C}

An example with $\mathcal{C} = \mathbb{R}^2 \times [0, 2\pi)$ and spherical C-obstacles



- 3 critical curves are obtained by projecting the spheres onto the x - y plane
- critical curves partition the x - y plane into the 5 non-critical regions
- within each non-critical region, a line perpendicular to the x - y plane intersects the same set of C-surfaces
- at a critical curve, the set of C-surfaces intersected by a line perpendicular to the x - y plane changes
- within a non-critical region, the value of θ will determine which, if any, of the C-obstacles overlap with \mathcal{A} , i.e., the non-critical region only identifies which types of contact are possible, not what contacts actually will occur

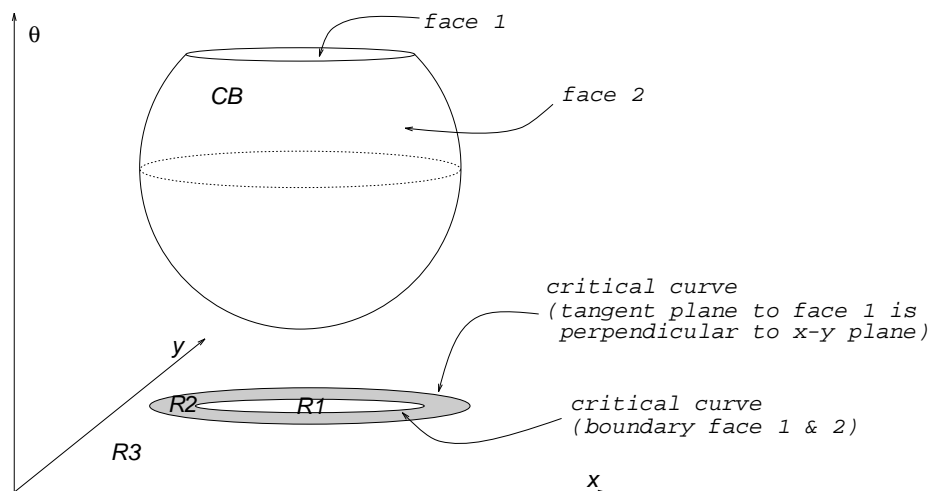
Critical Curves for a Ladder

Returning now to the segment PQ translating and rotating amongst polygonal obstacles in the plane... $\mathcal{CB} \in \mathbb{R}^2 \times [0, 2\pi)$ is bounded by faces, each of which is a ruled surface corresponding to some type of contact

- Type-A surface: segment PQ contains an obstacle vertex
- Type-B surface: P or Q is contained in an obstacle edge

The **critical curves** are **projections onto the x-y plane** of the following curves that live in $\mathbb{R}^2 \times [0, 2\pi)$

1. projections of curves that bound C-obstacle faces (i.e., C-obstacle edges)
2. projections of curves in C-obstacle faces where the tangent plane to the face is perpendicular to the x-y plane



example: 2 critical curves partition x-y plane into 3 non-critical regions:

- In $R1$, 2 possible kinds of contact (face 1 or face 2)
- In $R2$, 1 possible contact (face 2 only)
- In $R3$, no possible contact

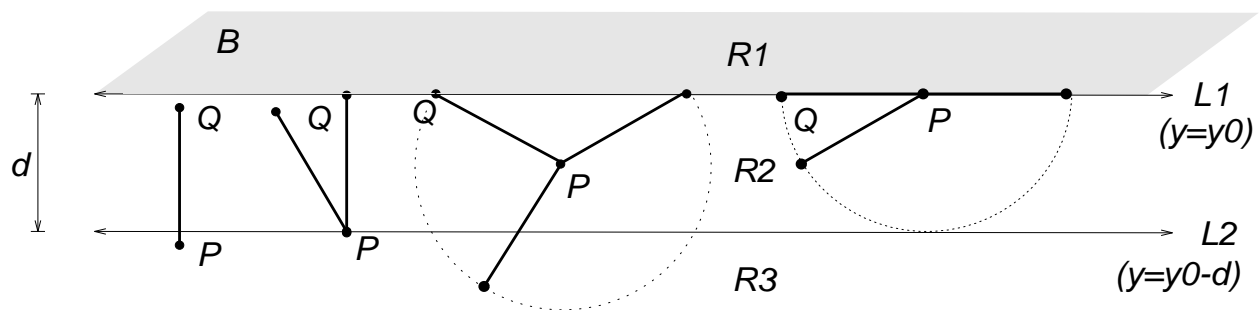
Key: *critical curves are places where possible kinds of contact change*

An Example

Key: *critical curves are places where possible kinds of contact change, i.e., one type becomes impossible and another type becomes possible*

So, to find the critical curves (and non-critical regions) we need to look for (x, y) positions of P where the types of contact can change

Example: \mathcal{B} is a closed half-plane bounded by a line $L1$



- if P is on the far side of $L2$, then no contact can occur
- if P is on $L2$, then $Q-B$ contact possible
- if P is between $L1$ and $L2$, then $Q-B$ contact possible
- if P is on $L1$, then $P-B$ contact, and $Q-B$ possible

two critical curves

- $L1$ ($y = y_0$): $P-B$ contact and $Q-B$ contact possible
- $L2$ ($y = y_0 - d$): $Q-B$ contact possible

three non-critical regions

- $R1$ (completely in \mathcal{CB}): $\{(x, y) \in \mathbb{R}^2 | y > y_0\}$
- $R2$ ($Q-B$ contact possible): $\{(x, y) \in \mathbb{R}^2 | y_0 - d < y < y_0\}$
- $R3$ (no contact possible): $\{(x, y) \in \mathbb{R}^2 | y < y_0\}$

A Taxonomy of Critical Curves

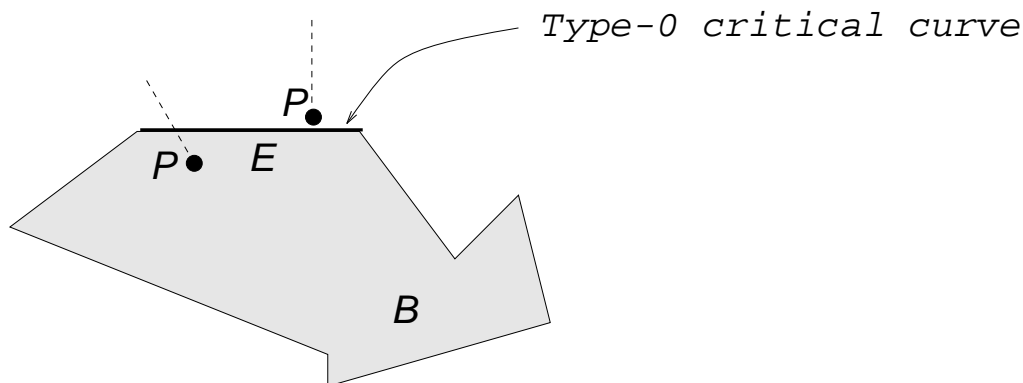
Two ways to get critical curves...

1. compute a boundary representation of \mathcal{CB} , project all the edges onto the x-y plane, and find all faces with tangent planes perpendicular to the x-y plane and project the curves in these faces to x-y plane
2. construct the critical curves by working directly with the obstacles in \mathcal{W} – consider all possible contacts (i.e., vertices/edges of each \mathcal{B}_i and \mathcal{A})

Schwartz and Sharir (1983) took the latter approach. They studied the problem of moving the ladder $\mathcal{A} = PQ$ in the plane and came up a taxonomy of critical curves.

Type-0 Curves:

Let E be an obstacle edge. E is a Type-0 critical curve.

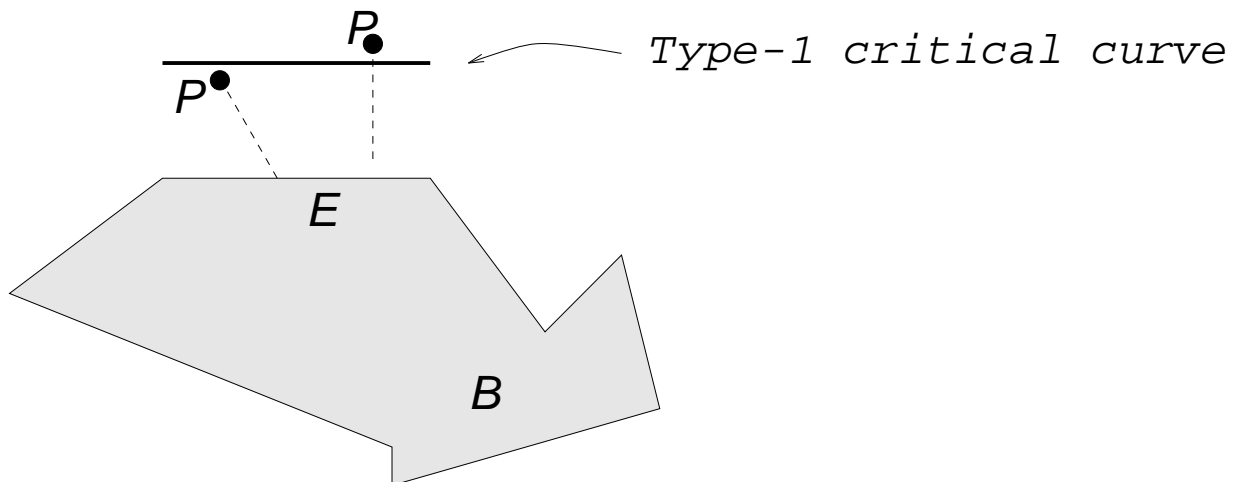


- if P is inside an obstacle, then $(x, y, \theta) \in \mathcal{CB}$ for **all** values of θ
- if P is outside an obstacle, then there may be some values of θ where $(x, y, \theta) \notin \mathcal{CB}$

Taxonomy of Critical Curves (cont'd)

Type-1 Curves:

Let E be an obstacle edge. The straight line segment at distance d from and parallel to E is a Type-1 critical curve.

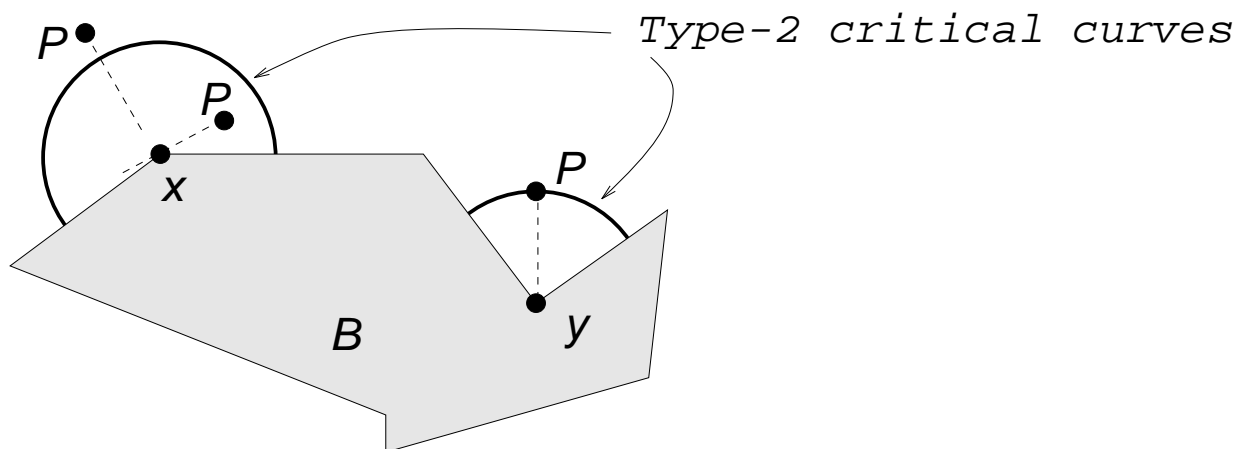


- if P is inside this segment (between it and E), then contact between PQ and E might occur for some values of θ
- if P is outside this segment, then no contact between PQ and E is possible for any value of θ

Taxonomy of Critical Curves (cont'd)

Type-2 Curves:

Let X be an obstacle vertex. The circular arc of radius d centered at X and bounded by the supporting half lines of the obstacle edges incident to X is a Type-2 critical curve.

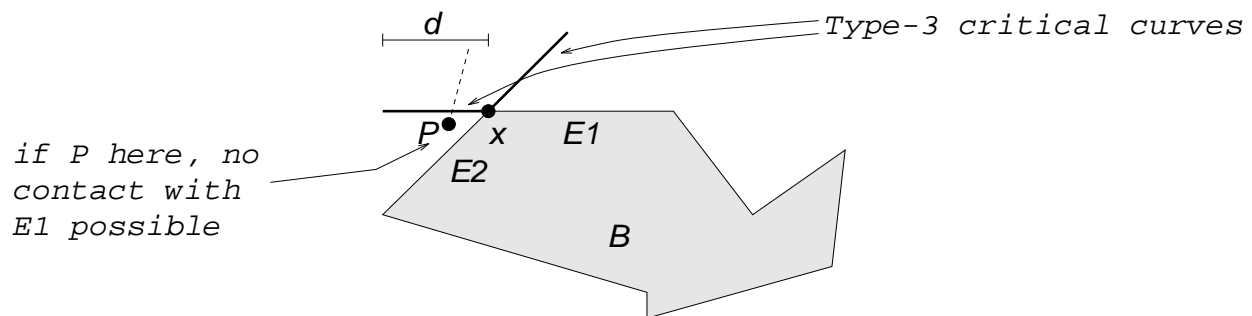


- if P is inside this arc, then contact between PQ and X might occur for some values of θ
- if P is outside this arc, then no contact between PQ and X is possible for any value of θ

Taxonomy of Critical Curves (cont'd)

Type-3 Curves:

Let E be an obstacle edge, and let X be a *convex* vertex that is an endpoint of E . The straight line segment traced by P as PQ slides along E and Q is contained in E is a Type-3 critical curve.

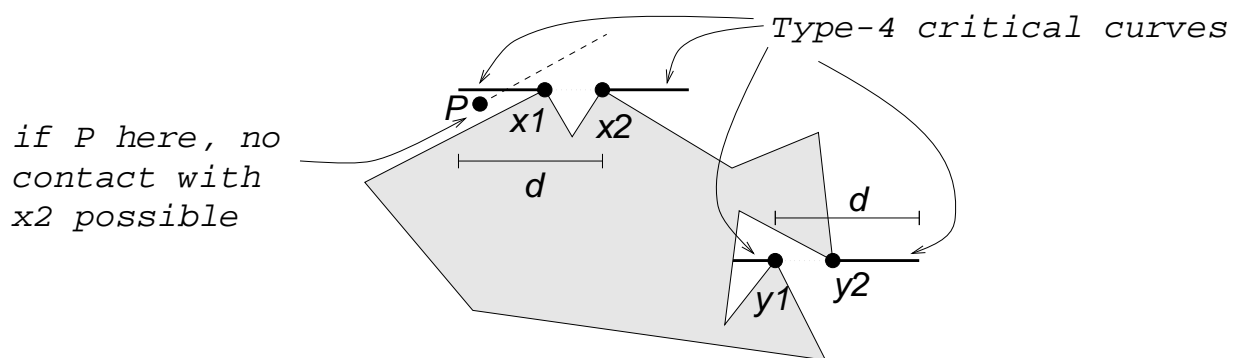


- if P is between this segment and the other edge incident on X , then no contact between PQ and E is possible (except at X)
- if P is not between this segment and the other edge incident on X , then contact between PQ and E may be possible for some values of θ

Taxonomy of Critical Curves (cont'd)

Type-4 Curves:

Let X_1 and X_2 be convex obstacle vertices ($\leq d$ apart) such that the line passing through them is tangent to \mathcal{B} at both X_1 and X_2 . The line segment traced by P as PQ slides while touching both X_1 and X_2 is a Type-4 critical curve.



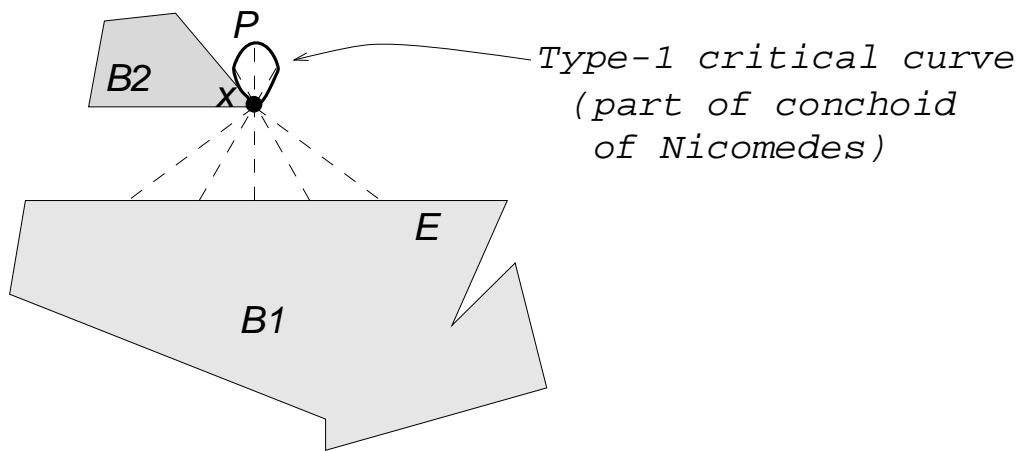
- if P is between this segment and the edge incident to X_1 (X_2) on the 'far' side of \mathcal{B} , then no contact between PQ and X_2 (X_1) is possible
- if P is not between this segment and the edge incident to X_1 (X_2) on the 'far' side of \mathcal{B} , then contact between PQ and X_2 (X_1) may be possible for some values of θ

Taxonomy of Critical Curves (cont'd)

Type-5 Curves:

Let E be an obstacle edge and let X be a convex obstacle vertex that is not an endpoint of E . Let h be the distance from X to E .

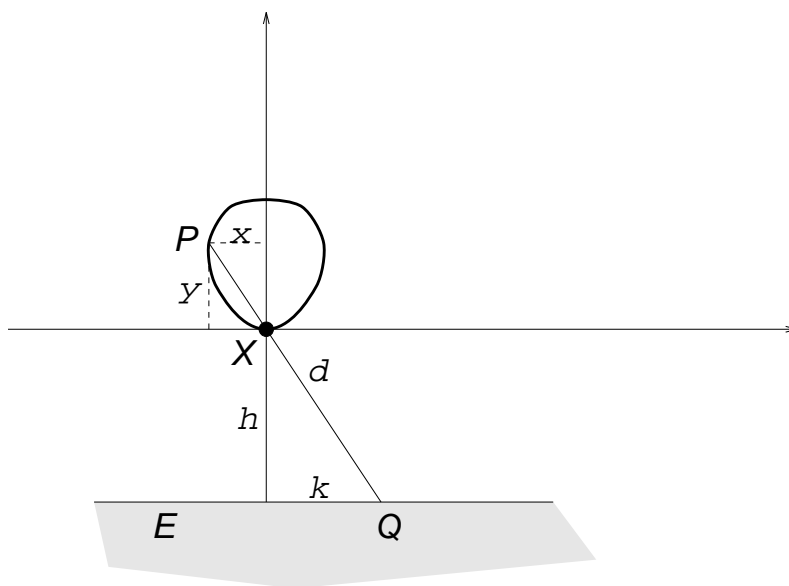
If $h < d$, the curve traced by P as PQ moves in a way such that it simultaneously touches E and X , while being tangent to \mathcal{B} at X , is a Type-5 critical curve.



- if P is inside this curve, then contact between PQ and X is not possible but contact with E may be possible for some values of θ
- if P is on this curve, then simultaneous contact between PQ and X and E is possible
- if P is outside this curve, then simultaneous contact between PQ and X and E is not possible

Taxonomy of Critical Curves (cont'd)

A closer look at Type-5 Curves



Letting the origin be X , and the x -axis parallel to E we have:

$$d^2 = (y + h)^2 + (x + k)^2 \quad (1)$$

$$\frac{y}{x} = \frac{h}{k} \quad (2)$$

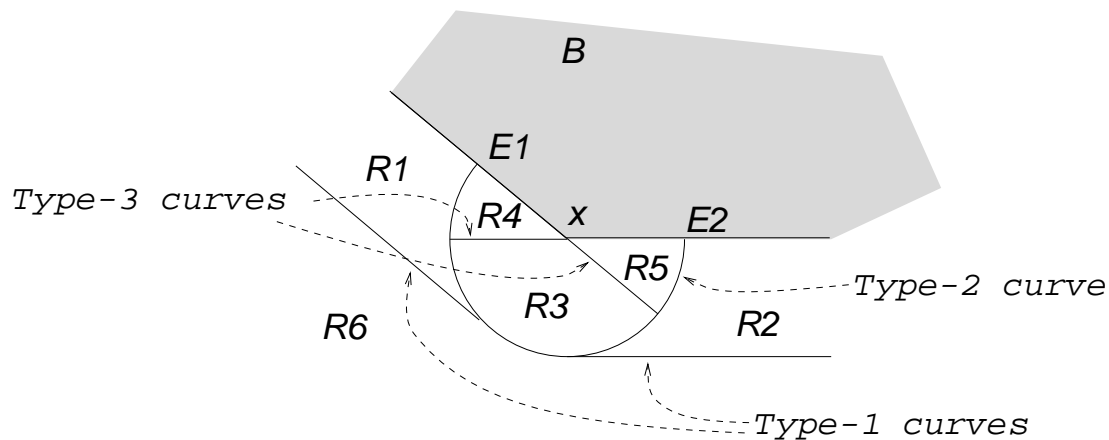
Thus, substituting hx/y for k in (1) we get (eventually):

$$x^2 = y^2 \left(\frac{d^2}{(y + h)^2} - 1 \right) \quad (3)$$

When $y \geq 0$, this equation represents a Type-5 critical curve (which is a subset of the conchoid of Nicomedes).

Example: An Arrangement of Critical Curves

The union of all critical curves is called an **arrangement** of critical curves. It partitions the x-y plane into a set of **non-critical regions** bounded by critical curve **sections** (the 'edges' in the arrangement).



Each non-critical region corresponds to different possible kinds of contact. For example, when P is in region:

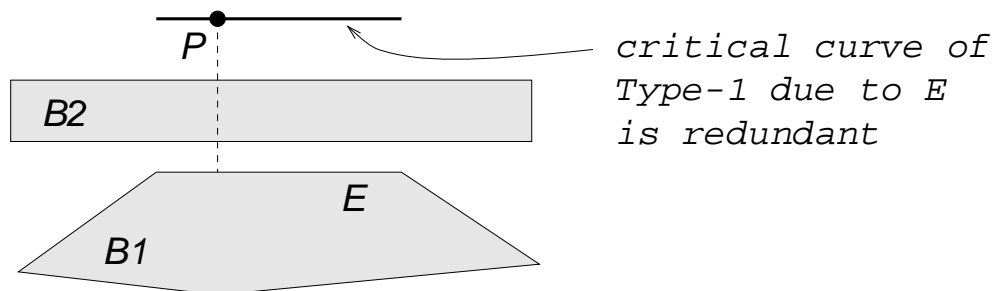
- **R1**: there is possible contact between Q and $E1$ (Type-B)
- **R2**: there is possible contact between Q and $E2$ (Type-B)
- **R3**: there is possible contact between Q and both $E1$ and $E2$ (Type-B)
- **R4**: there is possible contact between Q and $E1$ (Type-B) and possible contact between PQ and x (Type-A)
- **R5**: there is possible contact between Q and $E2$ (Type-B) and possible contact between PQ and x (Type-A)
- **R6**: there is no possible contact

Complexity Critical Curve Arrangements

1. $O(n^2)$ critical curves (n is number of vertices of \mathcal{B})
 - $O(n)$ critical curves of Types 0-3 (one edge or one vertex of \mathcal{B})
 - $O(n^2)$ critical curves of Types 4-5 (two vertices or vertex/edge of \mathcal{B})
 2. each critical curve is algebraic of degree 1 (Types 0, 1, 3, 4), 2 (Type-2), or 4 (Type-5)
 - two critical curves intersect in $O(1)$ points (unless they coincide, in which case we 'perturb' things a bit and we're ok since \mathcal{B} is a manifold)
 3. $O(n^4)$ intersection points (sections) in an arrangement of critical curves
 - $O(n^2)$ intersections (sections) on each critical curve (by 1 and 2)
-

Redundant Critical Curves

Sometimes a critical curve may be generated by the above rules even if the particular contact is not feasible. Such a curve is called **redundant**.



Because of \mathcal{B}_2 , PQ can never be such that P is on the Type-1 critical curve due to E while Q is on E .

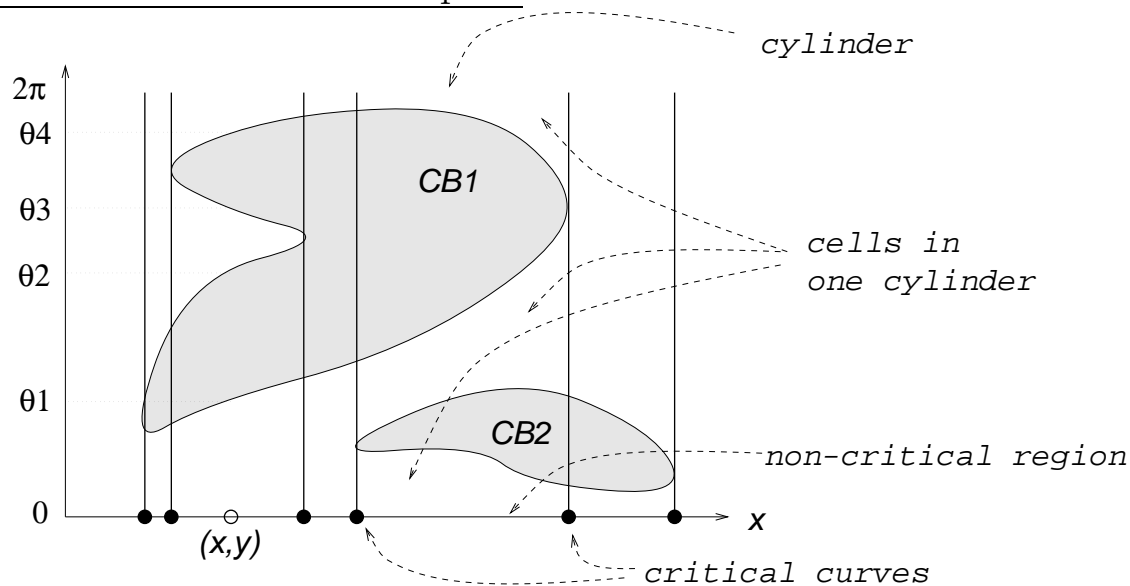
These will not cause a problem (we will see how to get rid of them easily)

Example: A More Complex Arrangement of Critical Curves

Exact Decomposition of \mathcal{C}_{free}

Basic Idea: We extend the non-critical regions in the θ direction perpendicular to the x-y plane to form cylinders. Each cylinder is partitioned into a disjoint set of *free* cells (bounded by C-obstacles).

Example: Two-dimensional C-space



Partitioning cylinders into cells...

Definition: $F(x, y)$ is the **set of all free orientations** of PQ when P is at a non-critical position (x, y) , i.e.,

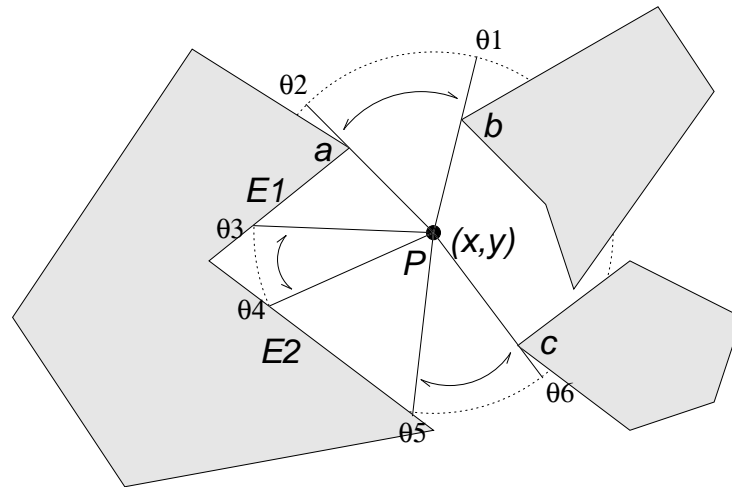
$$F(x, y) = \{\theta \mid (x, y, \theta) \in \mathcal{C}_{free}\}$$

If all orientations of PQ at position (x, y) are free, then $F(x, y) = [0, 2\pi)$. Otherwise, $F(x, y)$ is a finite collection of open, maximal intervals.

- e.g., $F(x, y) = \{(\theta_2, \theta_3), (\theta_4, \theta_1)\}$

Stops

What do the intervals in $F(x, y)$ look like in \mathcal{W} ?



$$F(x, y) = \{(\theta_1, \theta_2), (\theta_3, \theta_4), (\theta_5, \theta_6)\}$$

Definitions:

A **stop** is the vertex or edge of \mathcal{B} that PQ contacts at the limiting orientations in an interval in $F(x, y)$. Each interval in $F(x, y)$ has two stops – a **clockwise stop** and a **counterclockwise stop**.

- e.g., b and a are stops for (θ_1, θ_2) , $E1$ and $E2$ are stops for (θ_3, θ_4) , and $E2$ and c are stops for (θ_4, θ_5)

$\lambda_1(x, y, s)$ is the value of θ where PQ contacts clockwise stop s with P in position (x, y) . $\lambda_2(x, y, s')$ is the value of θ for counterclockwise stop s'

- e.g., $\lambda_1(x, y, E1) = \theta_3$ and $\lambda_2(x, y, E2) = \theta_4$

$\sigma(x, y)$ is the **set of all pairs of stops** associated with intervals in $F(x, y)$.

If $F(x, y) = [0, 2\pi)$, then $\sigma(x, y) = [\Omega, \Omega]$, where Ω is a special symbol

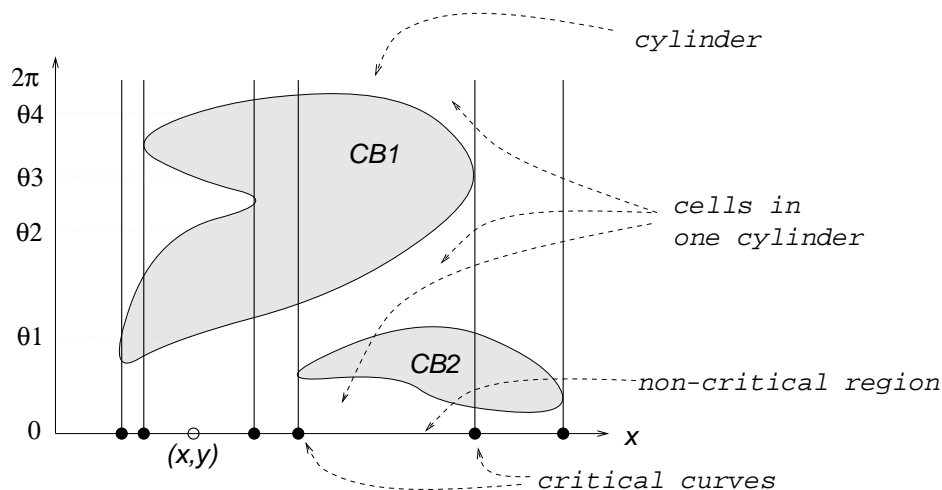
- e.g., $\sigma(x, y) = \{[b, a], [E1, E2], [E2, c]\}$

Stops within a non-critical region and Cells

Fact: Let R be a non-critical region, and let (x, y) and (x', y') be any two points in R . Then the intervals in $F(x, y)$ and $F(x', y')$ have the same set of stops, i.e., $\sigma(x, y) = \sigma(x', y')$. Let $\sigma(R)$ denote this set of stops.

This follows from the fact that all points in a non-critical region have the same possible types of contact.

Basic Idea: We have so far decomposed the x-y plane into a set of non-critical regions. Our exact decomposition of \mathcal{C}_{free} will consist of *free cells* in the 'cylinders' above the non-critical regions.



Definition: Let R be a non-critical region and let $[s_1, s_2] \in \sigma(R)$.

$$cell(R, s_1, s_2) = \{(x, y, \theta) \mid (x, y) \in R \text{ and } \theta \in (\lambda_1(x, y, s_1), \lambda_2(x, y, s_2))\}$$

is called a **cell**, i.e., $cell(R, s_1, s_2)$ is the set of all points in the 'cylinder' above R such that PQ is oriented between the two stops s_1 and s_2 . It is an open, connected subset of \mathcal{C}_{free} . Note, it is maximal in R 's cylinder, but is not necessarily maximal in \mathcal{C}_{free} .

The set of all such cells forms a decomposition of \mathcal{C}_{free} (but not \mathcal{C}).

Building a Connectivity Graph (cell adjacencies)

Definition: cells $cell(R, s_1, s_2)$ and $cell(R', s'_1, s'_2)$ are **adjacent** iff

- (i) the boundaries of R and R' share a critical curve segment β , and
- (ii) $\forall(x, y) \in int(\beta)$,
 $(\lambda_1(x, y, s_1), \lambda_2(x, y, s_2)) \cap (\lambda_1(x, y, s'_1), \lambda_2(x, y, s'_2)) \neq \emptyset$

Thus, if two cells k and k' are adjacent, then any configuration in k can be connected to any configuration in k' by a free path whose projection onto the x - y plane crosses β transversally, with constant orientation in some neighborhood of the crossing point.

Let R and R' be two non-critical regions such that β is a section of the critical curve contained in the boundaries of both R and R' .

Later we will define a **crossing rule** for β that will tell us the adjacency relations between the cells contained in the cylinders above R and R' . (The crossing rule will be based on $\sigma(R)$ and $\sigma(R')$, i.e., the set of stops for the free orientations in R and R' .)

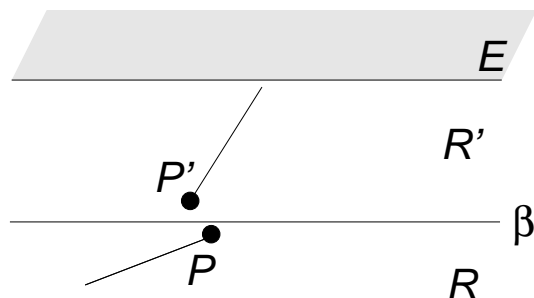
But first we'll look at some examples...

Adjacency Example: β is Type-1 critical curve segment

Let β be a section of a Type-1 critical curve due to an obstacle edge E , and let R' be the region between E and β and R the region on the other side of β . $cell(R, s_1, s_2)$ and $cell(R', s'_1, s'_2)$ are adjacent when P can move (directly) from R to R' in such a way that PQ only translates when P crosses β .

Cases where $cell(R, s_1, s_2)$ and $cell(R', s'_1, s'_2)$ are adjacent are shown below.

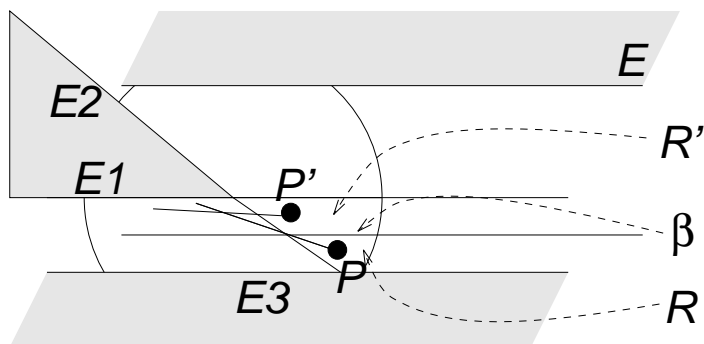
Case 1: $[s_1, s_2] = [\Omega, \Omega]$ and $[s'_1, s'_2] = [E, E]$



$$[s_1, s_2] = [\Omega, \Omega]$$

$$[s'_1, s'_2] = [E, E]$$

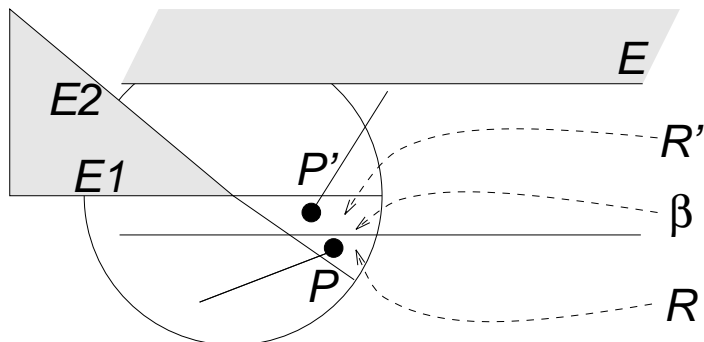
Case 2: $[s'_1, s'_2] = [s_1, s_2]$



$$[s_1, s_2] = [E1, E3]$$

$$[s'_1, s'_2] = [E1, E3]$$

Case 3: $[s'_1, s'_2] = [s_1, E]$ (or $[s'_1, s'_2] = [E, s_2]$)



$$[s_1, s_2] = [E1, E2]$$

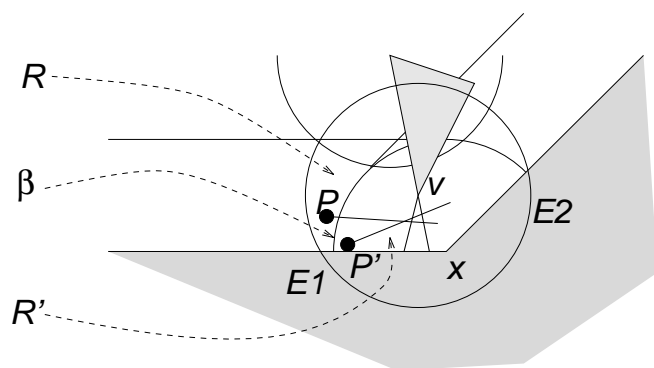
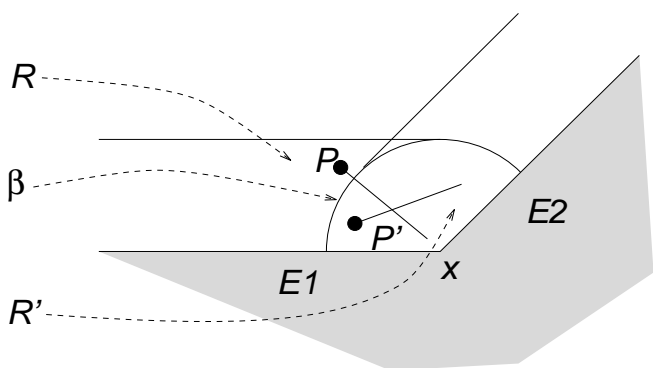
$$[s'_1, s'_2] = [E1, E]$$

Adjacency Example: β is Type-2 critical curve segment

Let β be a section of a Type-2 critical curve due to a concave vertex x of \mathcal{B} and adjacent to two edges $E1$ and $E2$. Assume the 'outer' angle between $E1$ and $E2$ is $\geq \pi/2$. Let β be the 'leftmost' segment of the Type-2 critical curve due to x .

Cases where $cell(R, s_1, s_2)$ and $cell(R', s'_1, s'_2)$ are adjacent are shown below.

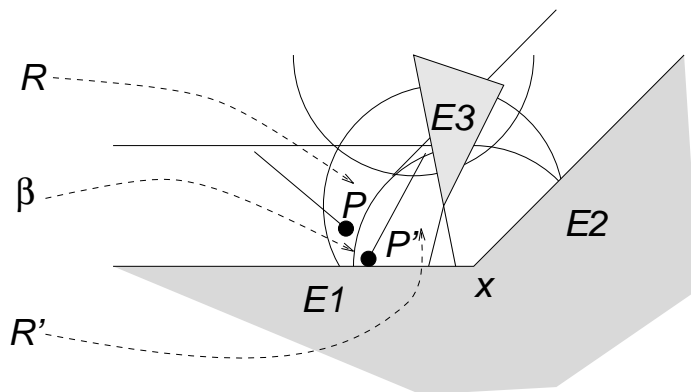
Case 1: $s_1 = E1, s'_1 = E2$ and $s_2 = s'_2$



$$[s_1, s_2] = [E1, E1] \text{ and } [s'_1, s'_2] = [E2, E1]$$

$$[s_1, s_2] = [E1, v] \text{ and } [s'_1, s'_2] = [E2, v]$$

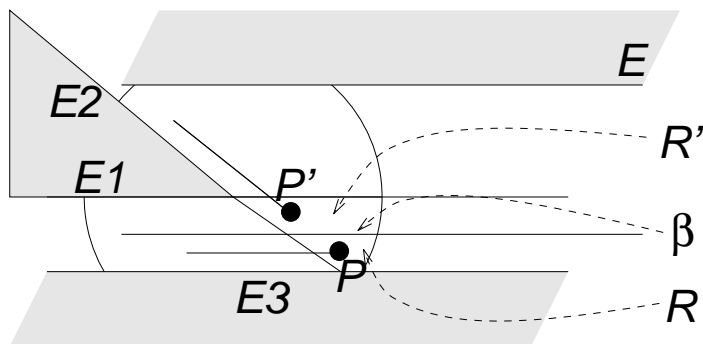
Case 2: $[s'_1, s'_2] = [s_1, s_2]$



$$[s_1, s_2] = [E3, E1] \text{ and } [s'_1, s'_2] = [E3, E1]$$

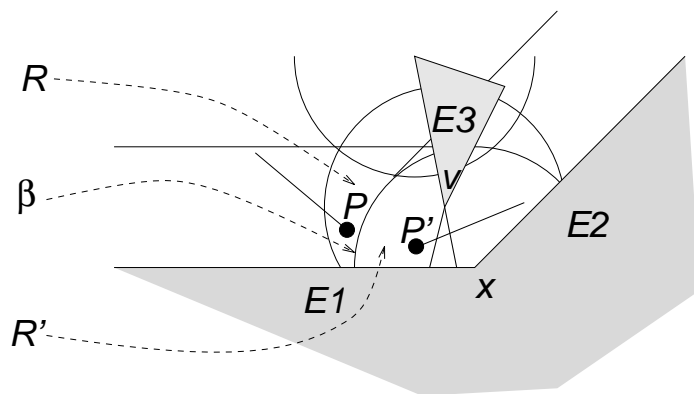
Some Non-Adjacent Cells

Some cases where $cell(R, s_1, s_2)$ and $cell(R', s'_1, s'_2)$ are **not** adjacent.



$$[s_1, s_2] = [E1, E3]$$

$$[s'_1, s'_2] = [E, E2]$$



$$[s_1, s_2] = [E3, E1]$$

$$[s'_1, s'_2] = [E2, v]$$

In the situations above there is no way to move between the two configurations while keeping P in $R \cup R'$.

A Crossing Rule for β (computing adjacent cells)

Let R and R' be two non-critical regions such that β is a critical curve section common to the boundaries of both R and R' . (Assume β is not redundant and does not coincide with any other critical curve.)

Fact: An exhaustive case analysis would show that whenever $int(\beta)$ is crossed transversally the set $\sigma(x, y)$ changes in one of the following ways:

- one new pair of stops appears in $\sigma(x, y)$
 - one old pair of stops disappears from $\sigma(x, y)$
 - one of the stops in a pair $[s_1, s_2]$ currently in $\sigma(x, y)$ changes (i.e., $[s_1, s_2]$ is replaced by $[s'_1, s'_2]$ and either $s_1 = s'_1$ or $s_2 = s'_2$)
-

We can now state the following rule for determining cell adjacencies (when β is not contained in any other critical curve).

Crossing Rule for β

1. for each $[s_1, s_2] \in \sigma(R) \cap \sigma(R')$,
 $cell(R, s_1, s_2)$ is adjacent to $cell(R', s_1, s_2)$ (i.e., cells with the same stops)
2. for each $[s_1, s_2] = \sigma(R) - \sigma(R')$ and $[s'_1, s'_2] = \sigma(R') - \sigma(R)$ (if they exist),
 $cell(R, s_1, s_2)$ is adjacent to $cell(R', s'_1, s'_2)$.

Note that in this case one of the following conditions holds:

- (a) $s_1 = s'_1$, or
- (b) $s_2 = s'_2$, or
- (c) $s_1 = s_2 = \Omega$ or $s'_1 = s'_2 = \Omega$

The Connectivity Graph and Motion Planning

Definition: The **connectivity graph** G corresponding to our exact decomposition of \mathcal{C}_{free} into cells (which are contained in the cylinders above the non-critical regions in the x-y plane) is an undirected graph

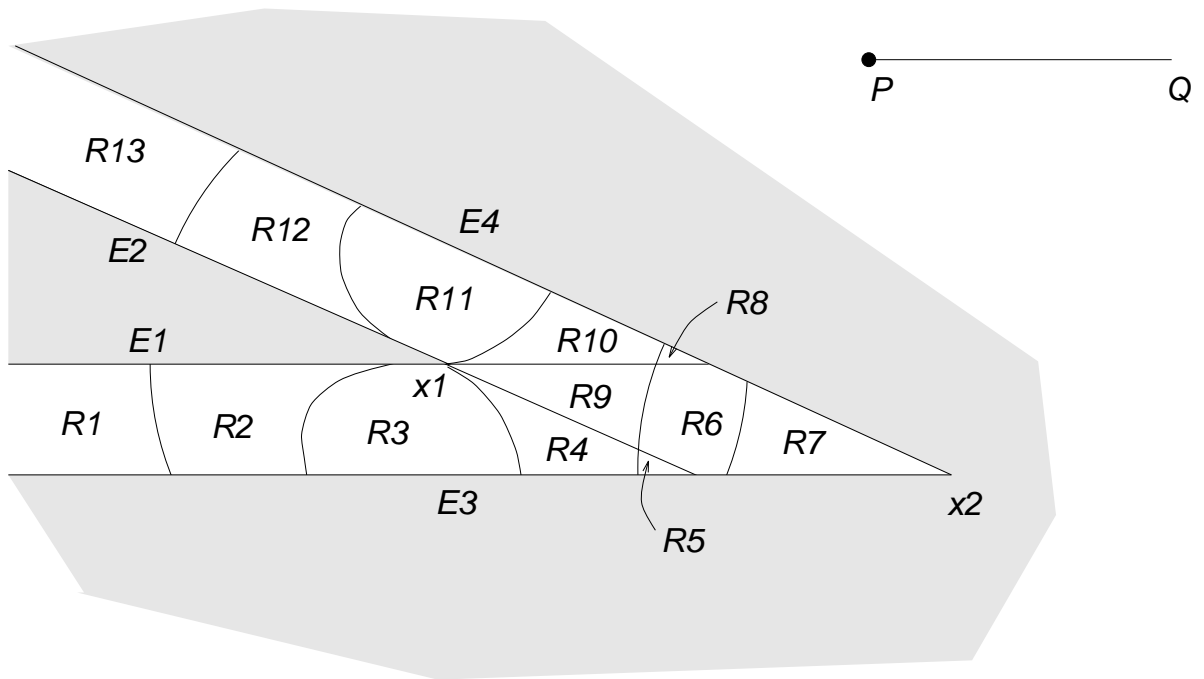
- where *nodes* correspond to cells $cell(R, s_1, s_2)$, where R is a non-critical region and $[s_1, s_2] \in \sigma(R)$
 - two nodes are connected by an *edge* iff their corresponding cells are adjacent
-

Motion Planning:

1. build connectivity graph G (does not depend on \mathbf{q}_{init} or \mathbf{q}_{goal})
 - (a) find all the critical curves
 - (b) compute their arrangement (intersect them)
 - (c) find the non-critical regions R and their stops $\sigma(R)$ (identifying cells $cell(R, s_1, s_2)$ in exact decomposition)
 - (d) determine the cell adjacencies (critical curve segments and stops)
2. locate the cells containing \mathbf{q}_{init} and \mathbf{q}_{goal}
3. find a path in G between the nodes corresponding to the cells containing \mathbf{q}_{init} and \mathbf{q}_{goal}

Fact: This is an **exact** motion planning method, i.e., there exists a free path connecting \mathbf{q}_{init} and \mathbf{q}_{goal} iff there exists a path in G between the nodes corresponding to the cells containing \mathbf{q}_{init} and \mathbf{q}_{goal} .

The Corner Example



Stops

$$\sigma(R_1) = \{[E_1, E_3], [E_3, E_1]\}$$

$$\sigma(R_2) = \{[E_1, E_3], [E_3, X_1]\}$$

$$\sigma(R_3) = \{[E_1, E_3], [E_3, E_4]\}$$

$$\sigma(R_4) = \{[E_1, E_3], [E_3, E_4], [E_4, X_1]\}$$

$$\sigma(R_5) = \{[E_1, E_3], [E_4, X_1]\}$$

$$\sigma(R_6) = \{[E_1, E_3], [E_4, E_2]\}$$

$$\sigma(R_7) = \{[E_4, E_3]\}$$

$$\sigma(R_8) = \{[E_4, E_2], [X_1, E_3]\}$$

$$\sigma(R_9) = \{[E_4, E_2], [E_1, E_3], [E_3, E_4]\}$$

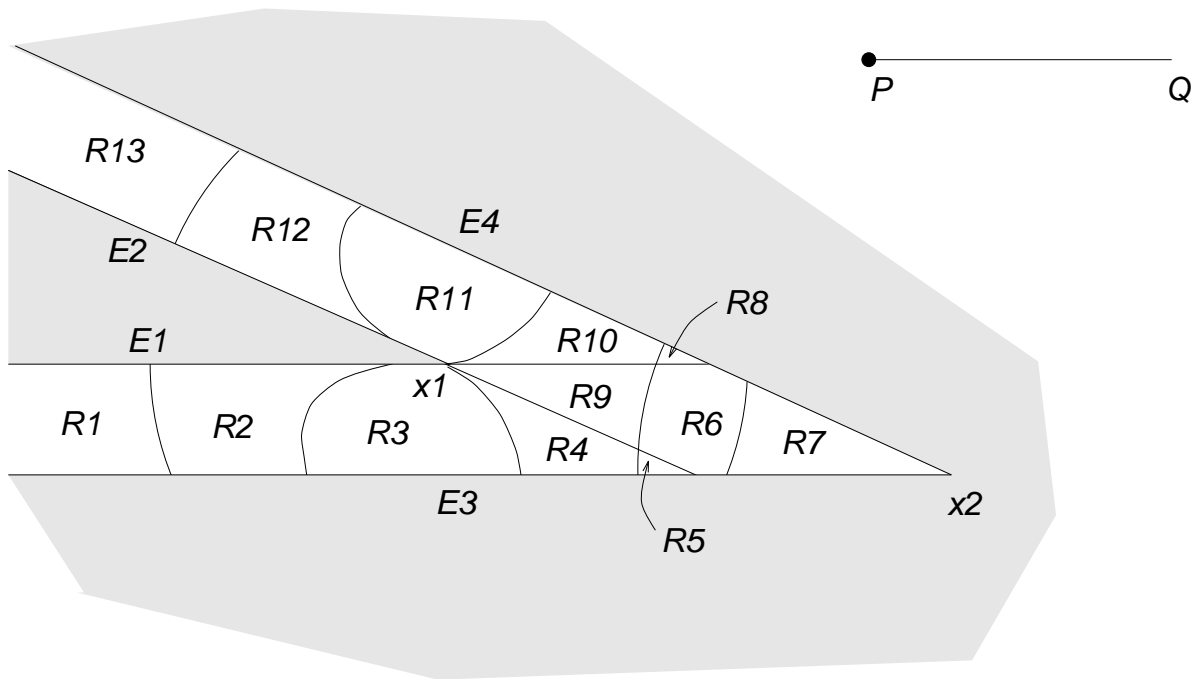
$$\sigma(R_{10}) = \{[E_4, E_2], [X_1, E_3], [E_3, E_4]\}$$

$$\sigma(R_{11}) = \{[E_4, E_2], [E_3, E_4]\}$$

$$\sigma(R_{12}) = \{[E_4, E_2], [X_1, E_4]\}$$

$$\sigma(R_{13}) = \{[E_4, E_2], [E_2, E_4]\}$$

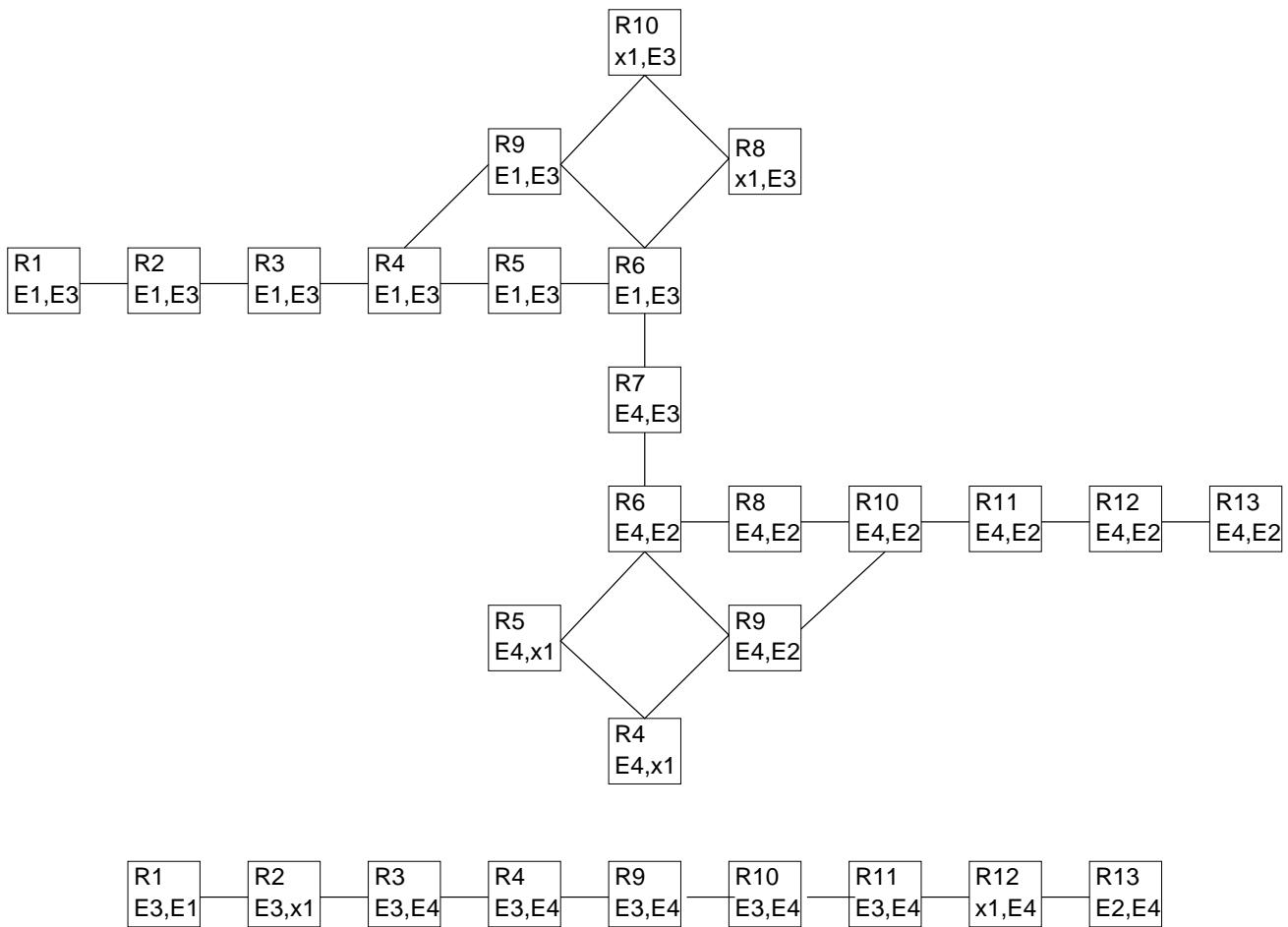
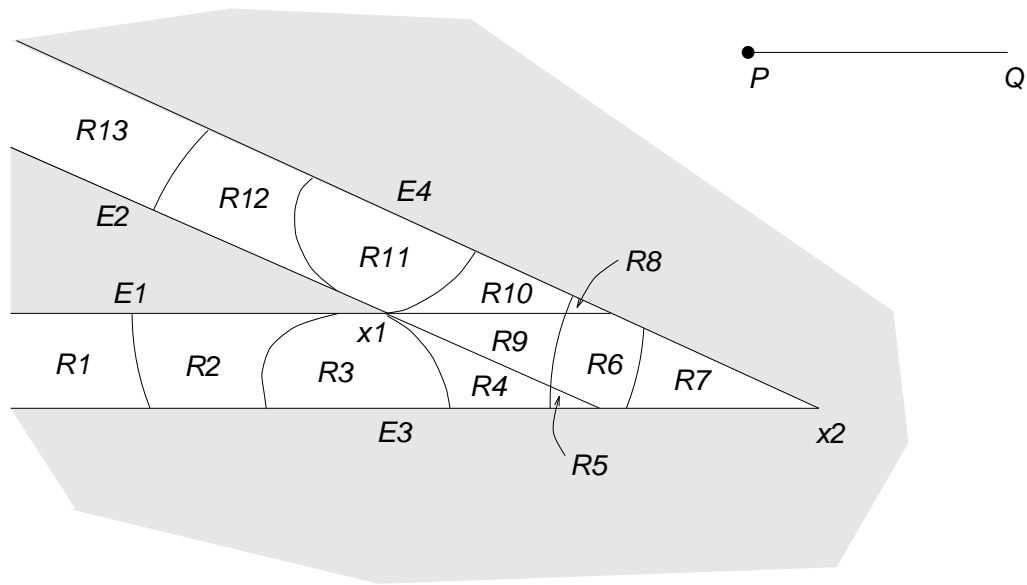
The Corner Example



Some Adjacent Cells

- $\sigma(R_4) = \{[E_1, E_3], [E_3, E_4], [E_4, X_1]\}$ and $\sigma(R_5) = \{[E_1, E_3], [E_4, X_1]\}$
 - $cell(R_4, E_1, E_3)$ is adjacent to $cell(R_5, E_1, E_3)$
 - $cell(R_4, E_4, X_1)$ is adjacent to $cell(R_5, E_4, X_1)$
- $\sigma(R_{12}) = \{[E_4, E_2], [X_1, E_4]\}$ and $\sigma(R_{13}) = \{[E_4, E_2], [E_2, E_4]\}$
 - $cell(R_{12}, E_4, E_2)$ is adjacent to $cell(R_{13}, E_4, E_2)$
 - $cell(R_{12}, X_1, E_4)$ is adjacent to $cell(R_{13}, E_2, E_4)$
- $\sigma(R_6) = \{[E_1, E_3], [E_4, E_2]\}$ and $\sigma(R_7) = \{[E_4, E_3]\}$
 - $cell(R_6, E_1, E_3)$ is adjacent to $cell(R_7, E_4, E_3)$
 - $cell(R_6, E_4, E_2)$ is adjacent to $cell(R_7, E_4, E_3)$

The Corner Example: Connectivity Graph



Exact Cell Decomposition: The General Case

So far we've considered only special cases. It turns out that the 'cylindrical decomposition' for the ladder can be extended to the general case.

In the general case, a **cylindrical algebraic decomposition** can be built that is an exact cell decomposition of \mathcal{C}_{free} suitable for motion planning

- the only constraint is that both \mathcal{A} and \mathcal{B} can be described as **algebraic sets** (e.g., no sin or cos)
- *Collins* gave a constructive proof that such a decomposition exists
 - this particular decomposition is called the **Collins decomposition**
 - algorithm is polynomial in n , the number of polynomials, and their degree, used to describe \mathcal{A} and \mathcal{B}
 - algorithm is doubly exponential in m , the dimension of \mathcal{C} ($O(2^{2^{O(m)}})$)
 - algorithm is recursive, e.g., builds m dimensional cylinders by projecting to $m - 1$ -dimensional space and then lifting, and the $m - 1$ dimensional cells are obtained recursively by the same process
 - decomposition has number of cells doubly exponential in m
- *Schwartz and Sharir* [1983] described a more efficient way to construct the connectivity graph for the Collins decomposition
 - a more efficient way to test cell adjacencies
 - is it only (!) doubly exponential in m , e.g., $O(2^{2^m})$ as opposed to $O(2^{2^{O(m)}})$

Importance of These Results: they establish that a general (exact) motion planning method exists that is polynomial in the geometric complexity. Unfortunately, they are not practical solutions in most cases since they are complex and have high complexity.