

Approximate 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 6 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.

Approximate Cell Decomposition

Idea: construct a collection \mathcal{K} of non-overlapping *cells* such that the union of all the cells *approximately* (and *conservatively*) covers the free C-space

Cell Characteristics:

- *cells should have a standard, simple shape* so that it is both easy to compute the decomposition (different from exact decomposition) and easy to compute a path between any two configurations in a cell (same as before)
- it should be pretty easy to test the adjacency of two cells, i.e., whether they share a boundary (same as before)
- it should be pretty easy to find a path crossing the portion of the boundary shared by two adjacent cells (same as before)

Advantages:

- simple, uniform decompositions
- easier and more efficient to implement
- adaptive—can achieve desired accuracy (resolution)

Disadvantages:

- large storage requirements (practical for dof ≤ 4)
- lose accuracy and contact info (don't use C-constraints)
- lose completeness (may fail to find paths)
- decompositions not based on \mathcal{CB}

Bottom Line: we sacrifice exactness for simplicity and efficiency

Approximate Decomposition into Rectangloids

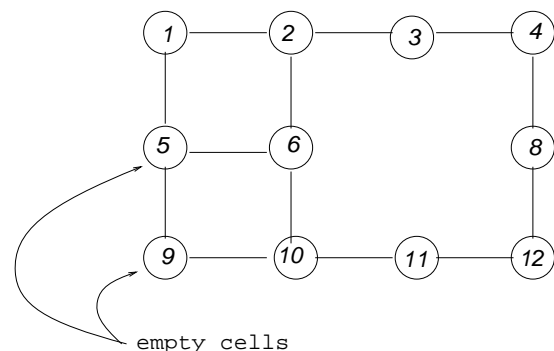
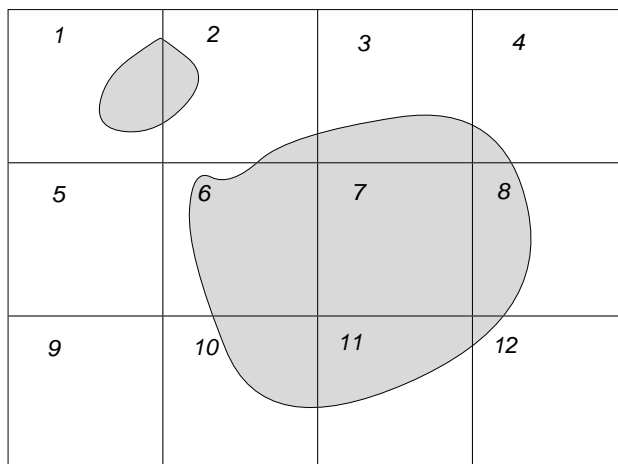
Let Ω be a bounded, closed rectangloid of \mathbb{R}^m representing all configurations of \mathcal{A} (m is the dimension of \mathcal{C})

A **rectangloid decomposition** \mathcal{P} of Ω is a finite collection of interior disjoint rectangloids (cells) k_1, k_2, \dots, k_r such that $\Omega = \cup_i k_i$

– cells k_i and k_j are adjacent if they share a boundary

Each cell k_i is either:

1. **EMPTY**: if its interior is contained in \mathcal{C}_{free} , i.e., $int(k_i) \cap \mathcal{CB} = \emptyset$
2. **FULL**: if it is contained in \mathcal{CB} , i.e., $int(k_i) \subseteq \mathcal{CB}$
3. **MIXED**: otherwise (its interior is intersected by the boundary of \mathcal{CB})



The **connectivity graph** associated with the decomposition \mathcal{P} of Ω is the undirected graph G :

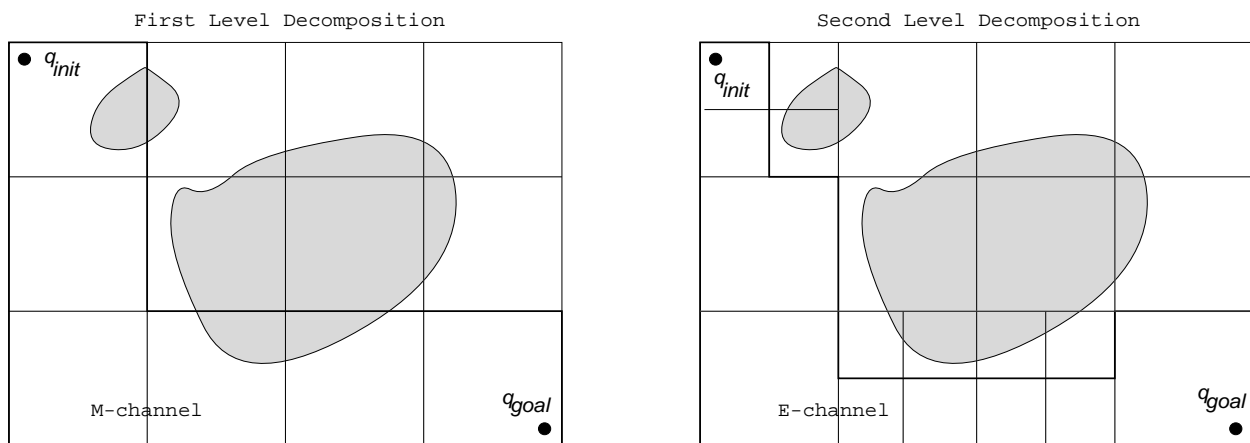
- nodes in G correspond to **EMPTY** and **MIXED** cells in \mathcal{P}
- nodes connected by edge in G iff corresponding cells adjacent in \mathcal{P}

Hierarchical Path Planning

Problem: maybe \mathbf{q}_{init} and \mathbf{q}_{goal} can only be connected by a M-channel in \mathcal{P}

Idea: Construct an E-channel from the M-channel by constructing *rect-angloid decompositions* of the MIXED cells in the M-channel

- do this recursively until an E-channel is obtained (or until no M-channel exists – no free path exists in channel)
- let \mathcal{P}_i denote the i th decomposition ($\mathcal{P} = \mathcal{P}_0$)



This approach can be made **complete**, i.e., it will return an E-channel whenever \mathbf{q}_{init} and \mathbf{q}_{goal} lie in the same connected component of \mathcal{C}_{free}

- however, worst-case computation time is unbounded!

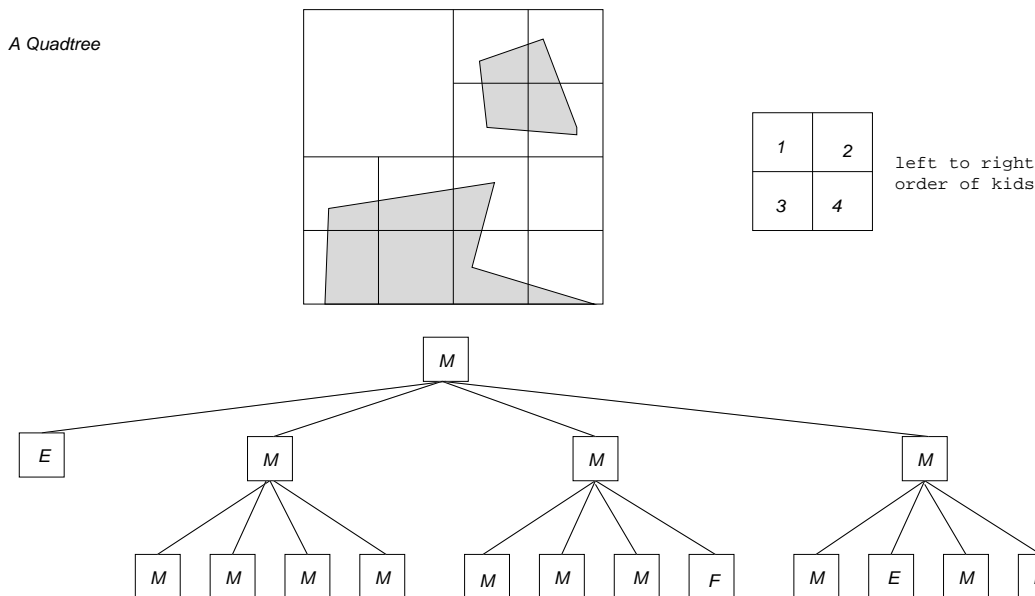
Typically, the recursive decomposition is stopped when the size of the cells reaches some fixed value ϵ (resolution)

- worst-case computation time is now bounded (and adjustable)
- method is no longer complete, but it is **resolution-complete**, i.e., it finds an E-channel whenever one exists provided that ϵ is small enough
- might stop decomposition if cell is 'mostly' full (conservative)

Decomposing a MIXED Cell: Divide-And-Label

A 2^m -**Tree Decomposition** of MIXED cell Ω is tree T (m dimension of \mathcal{C}):

- each node is a rectangloid and is labeled as EMPTY, FULL, or MIXED
- the degree of T is 2^m , i.e., each internal node has 2^m children, and all children of a node have the same dimension
- the root of T is Ω and only MIXED nodes have children



If $m = 2$ we have a *quadtree* and if $m = 3$ we have an *octree*

- the **depth** of a node (distance from root) determines the dimensions of the cell (depends on size of Ω)
- the **height** h of the tree (depth of the deepest leaf) determines the resolution of the decomposition (the deepest node is the smallest)
- in the worst-case, the tree has 2^{mh} nodes (in practice usually much less)

Now we need to see how to label cells as EMPTY, FULL, or MIXED

Labeling Cells when $\mathcal{C} = \mathbb{R}^2$

Assume $\mathcal{W} = \mathbb{R}^2$, \mathcal{A} and \mathcal{B}_i are polygons, and \mathcal{A} can translate (but not rotate). Thus $\mathcal{C} = \mathbb{R}^2$.

Lets decompose \mathcal{A} and \mathcal{B}_i into convex pieces. Recall from before

$$\text{CB}_{k,l}(\mathbf{q}) = \left(\bigwedge_{i,j} \text{CONST}_{i,j}^{\mathcal{A}}(\mathbf{q}) \right) \wedge \left(\bigwedge_{i,j} \text{CONST}_{i,j}^{\mathcal{B}}(\mathbf{q}) \right)$$

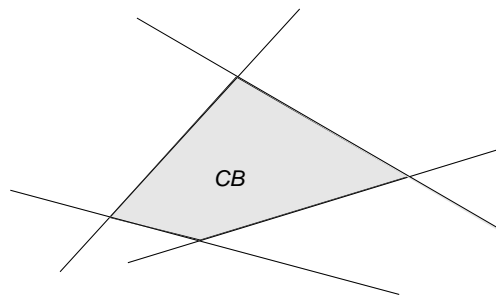
where k and l index the convex pieces of \mathcal{A} and \mathcal{B} , respectively, and

$$\text{CB}(\mathbf{q}) = \bigvee_{k,l} \text{CB}_{k,l}(\mathbf{q})$$

so CB is described by a disjunctive expression called a **C-sentence** of the form

$$\bigvee_i \bigwedge_j e_{i,j}$$

NOTE: i identifies one of the (k, l) pairs from above and j identifies one of the (i, j) pairs from above i.e., every conjunct $\bigwedge_j e_{i,j}$ describes a convex C-obstacle formed from some pair of convex pieces from \mathcal{A} and \mathcal{B}

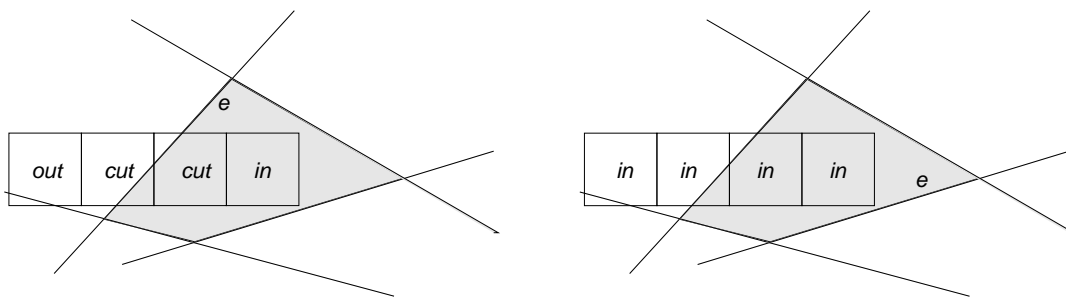


Recall that boundaries of C-obstacles are given by the $f = 0$ equations, and a convex C-obstacle is formed by intersecting halfplanes defined by $f \leq 0$, i.e., edges of C-obstacles come from the f s

Cells and C-Constraints

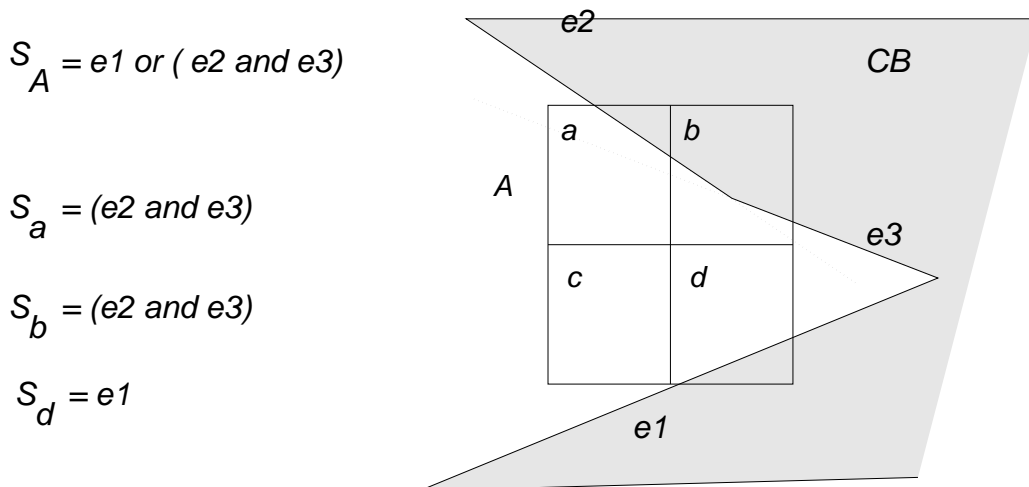
Given a C-constraint e_{ij} (a halfplane), a cell is either

1. *inside* e_{ij} if all of its four vertices are inside e_{ij}
2. *outside* e_{ij} if all of its four vertices are outside e_{ij}
3. *cut* by e_{ij} otherwise

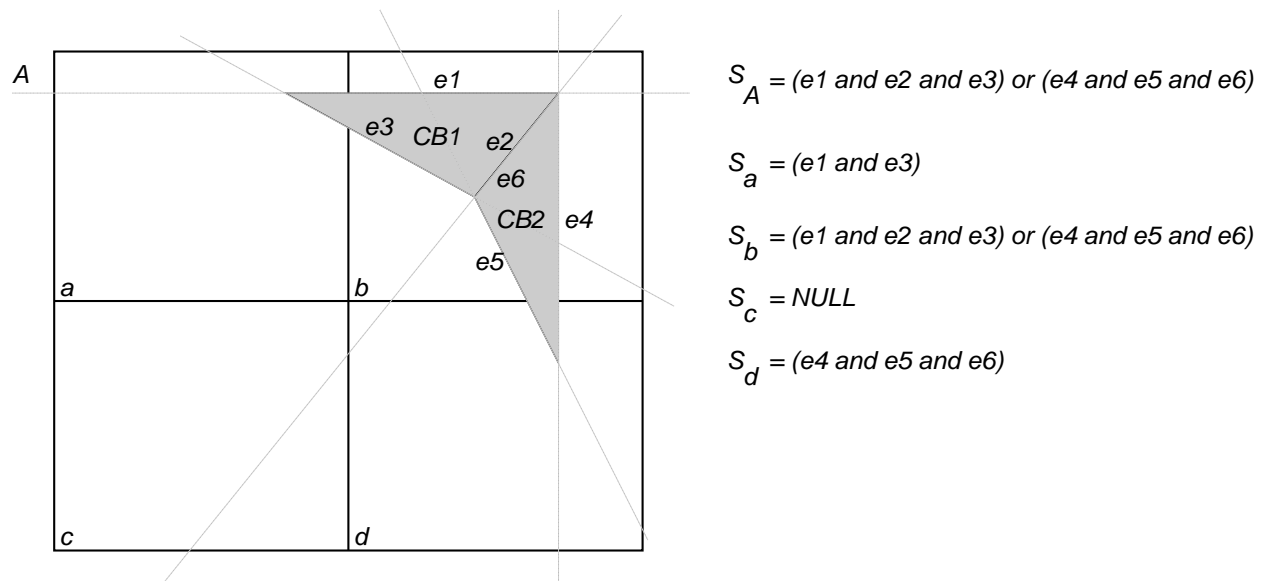


Basic Idea: Associate a C-sentence with each cell that describes the C-obstacles that intersect that cell

- label cells by testing their vertices with the C-sentence of the parent cell
- as we label a cell, construct its C-sentence from the C-sentence of its parent



Labeling a New Cell



PROCEDURE: LABEL(k, S)

input: new cell k , and S , the C-sentence for k 's parent

output: k 's label and its C-sentence S_k

for each conjunct $\Lambda \in S$ /* Λ describes a convex CB-piece*/

for each C-constraint $e \in \Lambda$ /* e is halfplane containing CB-piece edge*/

if k is inside e , then $\Lambda = \Lambda - \{e\}$ /*any subcell of k also in e */

if k is outside e , then $S = S - \{\Lambda\}$ /* k can't intersect CB-piece*/

if $\Lambda \in S$ and $\Lambda = \emptyset$ /* k inside all $e \in \Lambda$ */

then label k as FULL and exit

if $S = \emptyset$

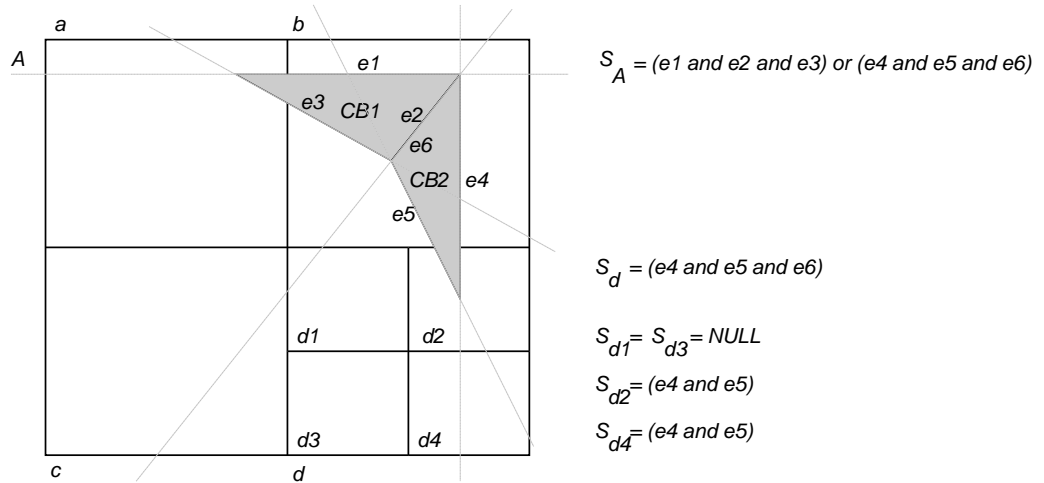
then label k as EMPTY

else label k as MIXED and set $S_k = S$

Complexity: linear in n , the number of C-constraints in S

Labeling is Conservative

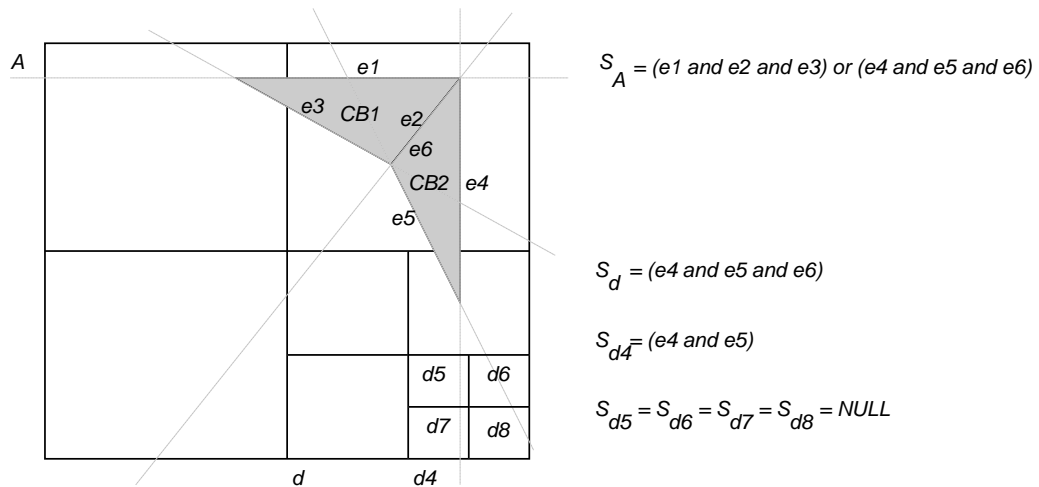
Note: the algorithm sometimes conservatively labels EMPTY cells as MIXED



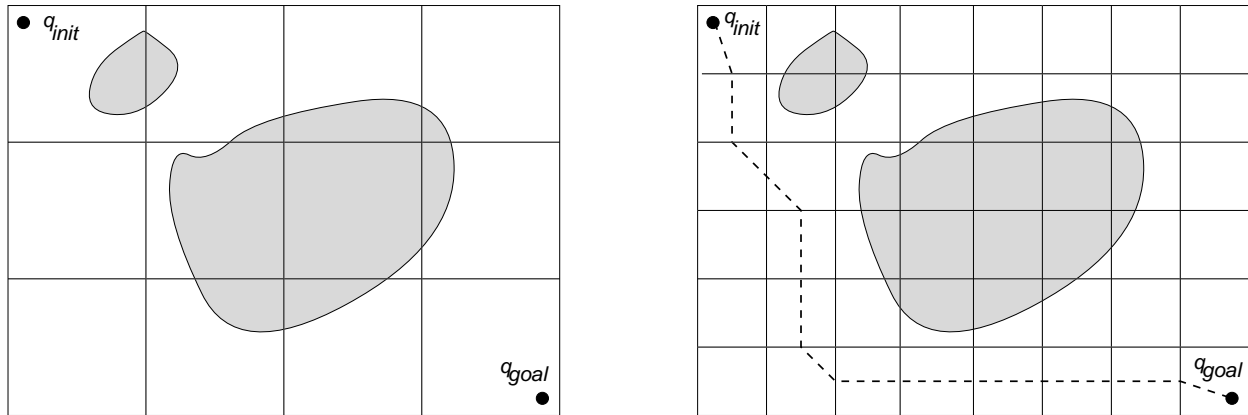
Cell $d4$ is labeled as MIXED because (simultaneously)

- it has vertices outside $e4$ and vertices inside $e4$ **and**
- it has vertices outside $e5$ and vertices inside $e5$

But, never fear, this problem will be fixed in some later iteration



Hierarchical Path Planning with Rectangloid Decomposition



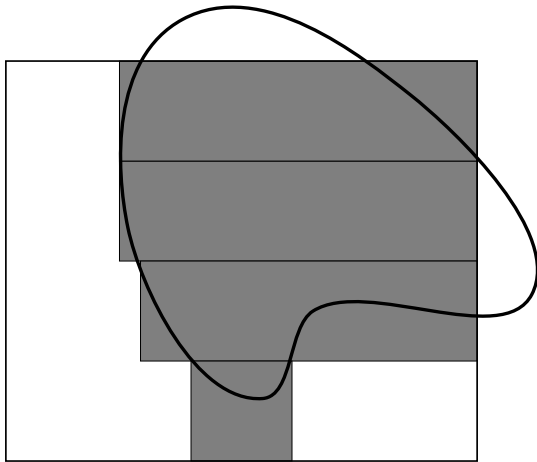
HIERARCHICAL PATH PLANNING WITH A RECTANGLOID DECOMPOSITION

input: Ω and configurations \mathbf{q}_{init} and \mathbf{q}_{goal}

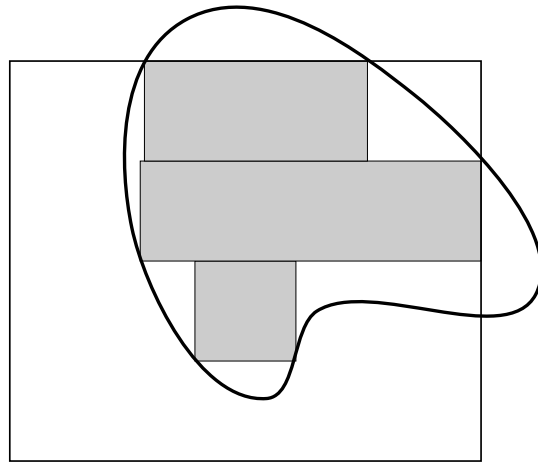
output: a path in \mathcal{C}_{free} connecting \mathbf{q}_{init} and \mathbf{q}_{goal}

1. compute a rectangloid decomposition \mathcal{P}_1 of Ω
 set $i := 1$
 2. locate the cells k_{init} and k_{goal} in \mathcal{P}_i containing \mathbf{q}_{init} and \mathbf{q}_{goal}
 3. search \mathcal{P}_i 's connectivity graph G_i for a channel connecting k_{init} and k_{goal}
 - (a) if an E-channel was found, then report SUCCESS and goto step 5
 - (b) if an M-channel was found, then goto step 4
 - (c) if no channel was found, then report FAILURE
 4. compute a rectangloid decomposition of each MIXED cell in the M-channel
 set $i := i + 1$ and goto step 2
 5. find a free path from \mathbf{q}_{init} to \mathbf{q}_{goal} in the E-channel
 – cross at midpoints of cell boundaries (may need to interior points)
-

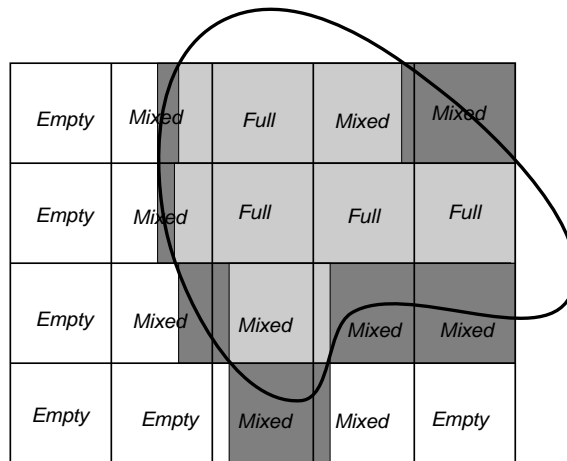
Approximate-and-Decompose Mixed Cell Decomposition



Bounded Approximation



Bounding Approximation



A Decomposition

- [Zhu and Latombe, 1989]
- compute **bounding** and **bounded** approximations
- use these to construct EMPTY, FULL, and MIXED cells
- motivation is to compute fewer cells in total

Dealing with more complicated C-spaces

In principle, the hierarchical planning method with rectangloid decompositions can be applied to any path planning problem

- mixed cell decomposition using 2^m -Trees generalizes also
 - cell labeling depends on \mathcal{C} (text gives method for $\mathcal{C} = \mathbb{R}^2 \times [0, 2\pi)$)
- 'approximate-and-decompose' mixed cell decomposition
 - can produce fewer cells in total
- other mixed cell decomposition methods

Bottom Line: Approximate Cell Decomposition methods are practical only for special cases or low-dimension \mathcal{C} , i.e., $dof \leq 4$, because they produce too many cells (large storage and search)