2) Cell decomposition approaches

Note Title 1/26/2010

Partition C_{free} into a

set of non-overlapping regions, called cells whose union is free. Build an adjacency relationship between cells — call it a connectivity graph — whose nodes are cells and edges between nodes represent if the two corresponding cells share a boundary. A sequence of adjacent cells connecting point to goal

constitutes a channel. A path can in c free

then be extracted from the channel.

What is a cell?

1) generally should be a simple shape, so that counting two pts within "convex" a cell is easy.
   ↓
   st. line

2) adjacency relationship should be relatively easy to test; constructy

a path crossing the boundary should be easy.

Types of cell decomp.:

1) approximate: cell shapes are pre-def.

faster resolution comp.

arb. level of res.

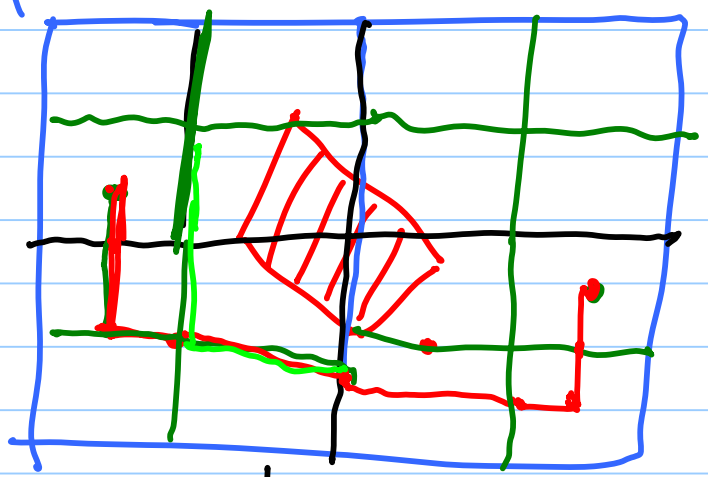
harder more applied for low dim c-spaces

2) exact: cell shape is determined by certain critical events in C_{free}.

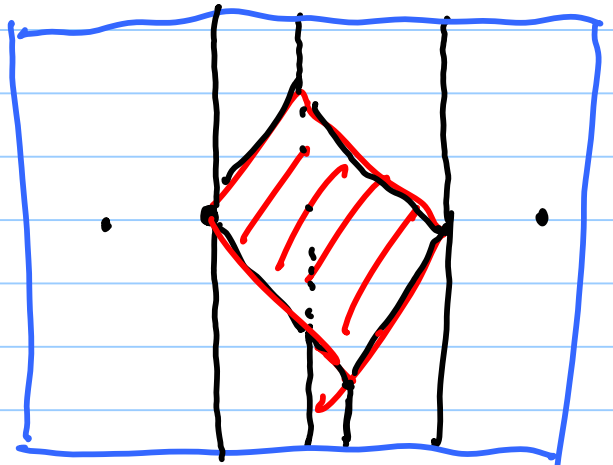
will lead to complete alg. but probably more complex computations are needed

polygonal / trapezoidal decomp for polygonal c-space.

Approx

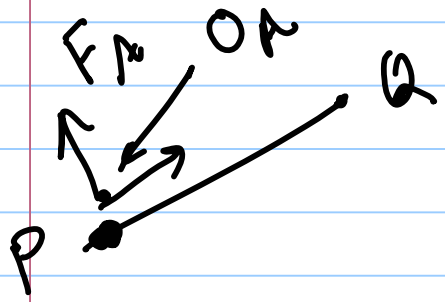


resolution complete



& complete alg.

Non-trivial Case: moving a ladder in a polygonal environment



in plane

General Case : Semi-algebraic sets

Recall they are described by boolean comb. of polynomial inequalities. → rational coeff.
↘ (obs / robot surfaces)

Path Planning prob. described as a "Tarski" sentence

Collin's decomp" → hierarchical way of decomposing "C_{free}" into cells and

determining adjacency relationship.

polynomial in geometric complexity

doubly exp. in 2^{2^d} dim. of d of c-space

Note Canny reduced this to 2^d via his "silhouette algorithm" which builds a "roadmap".

Approx. Cell decomp:

1) Represent C_{free} as a union of cells of pre-defined shape, often rectangloids. generally hierarchical
2) easier to implement
3) provides less insight into the structure of C_{free}
4) time & space complexity is exp in dim of C-space. useful for low

dim, often ≤ 4.

Gen of structure:

see lecture 9a