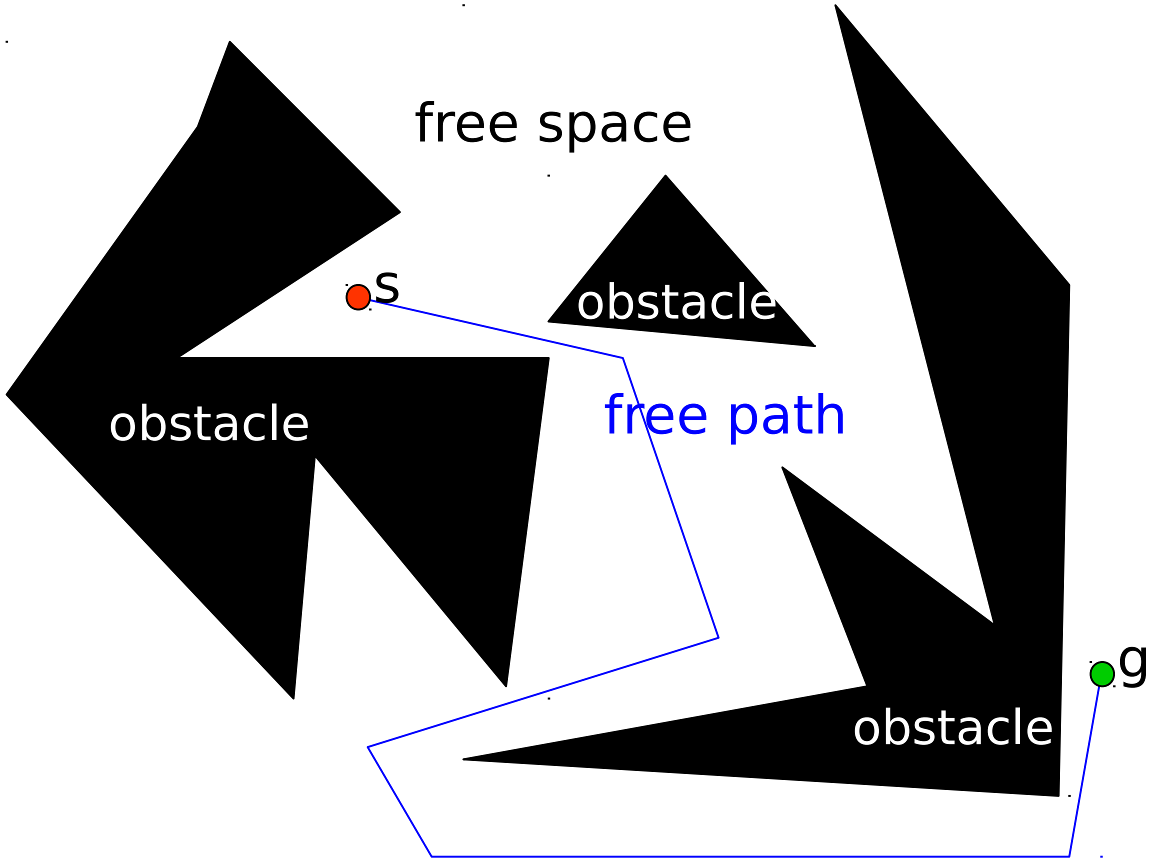
# Cell Decomposition Methods

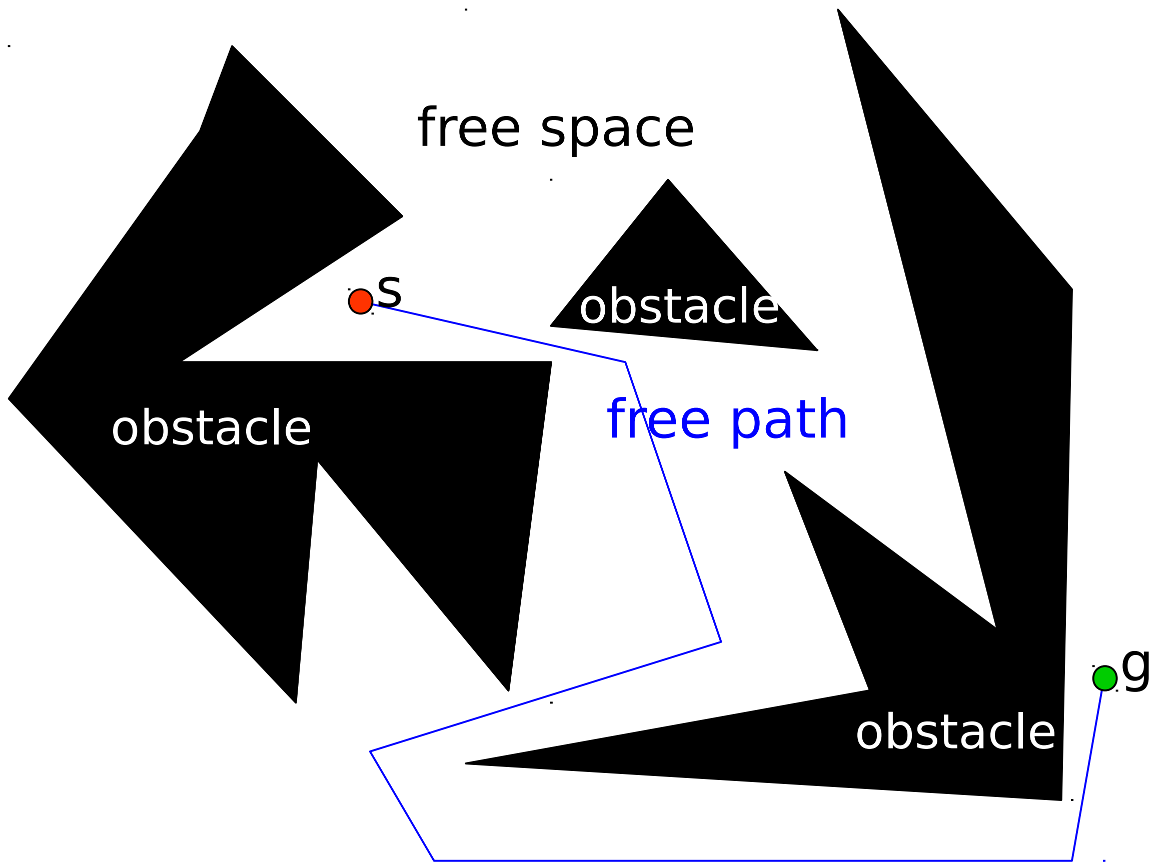



Slides borrowed from: Latombe's book and slides, Choset, Kuffner

# Problem

<u>Given</u>:
– a point-robot (robot is a point in space)
– a start and goal configuration

<u>Find</u>:
– path from start to goal that does not result in a collision



# Motion planning framework

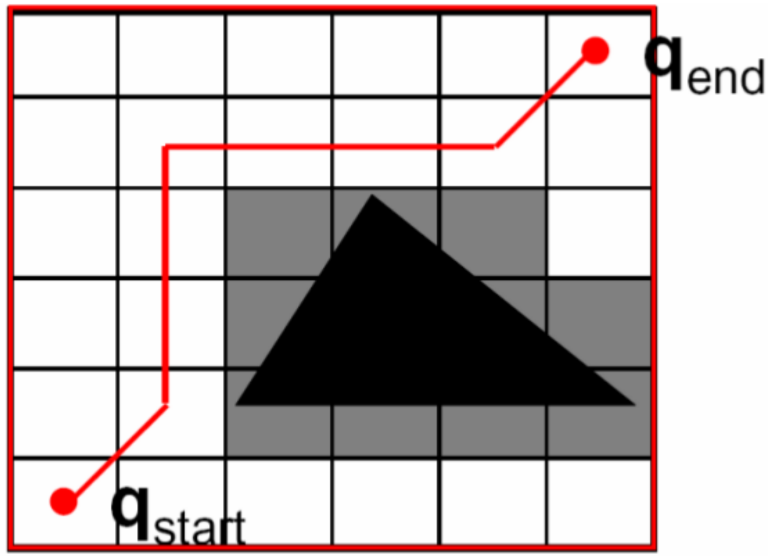
Continuous representation

Discretization

Graph searching

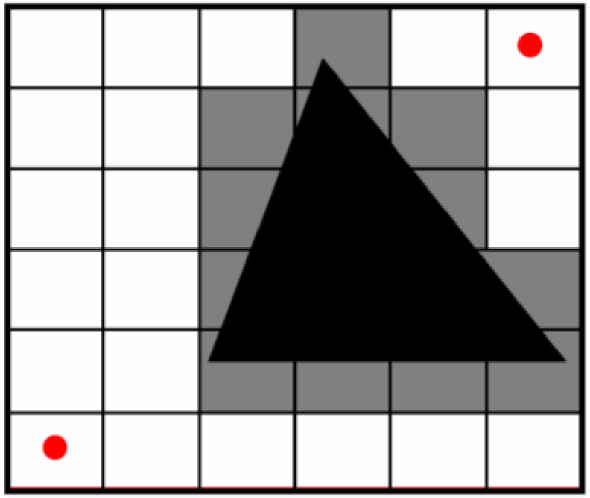
(blind, best-first, A*)

# Approximate Cell Decomposition



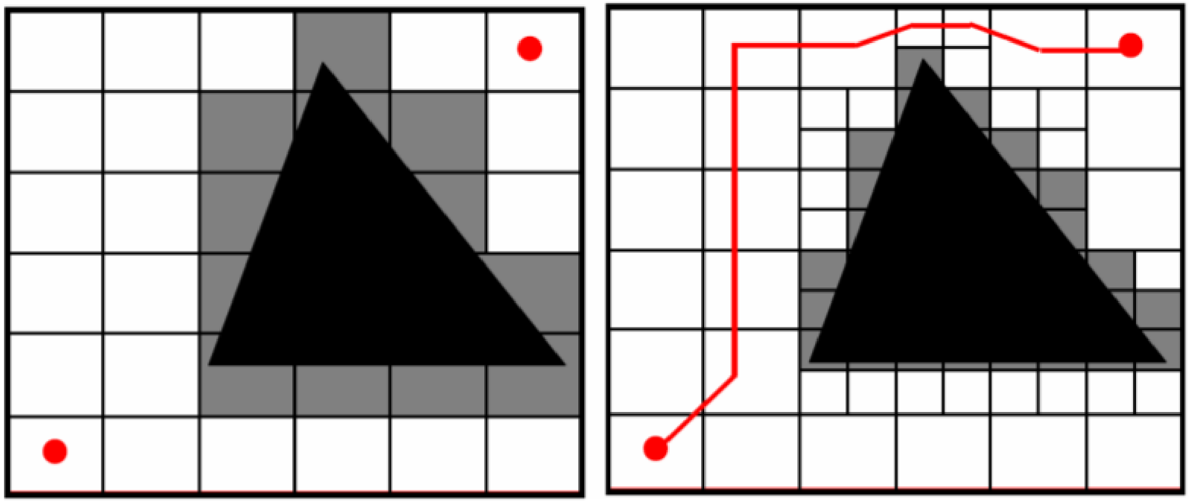

- Define a discrete grid in C-Space
- Mark any cell of the grid that intersects $\mathcal{C}_{obs}$ as blocked
- Find path through remaining cells by using (for example) A* (e.g., use Euclidean distance as heuristic)
- Cannot be *complete* as described so far. Why?

# Approximate Cell Decomposition



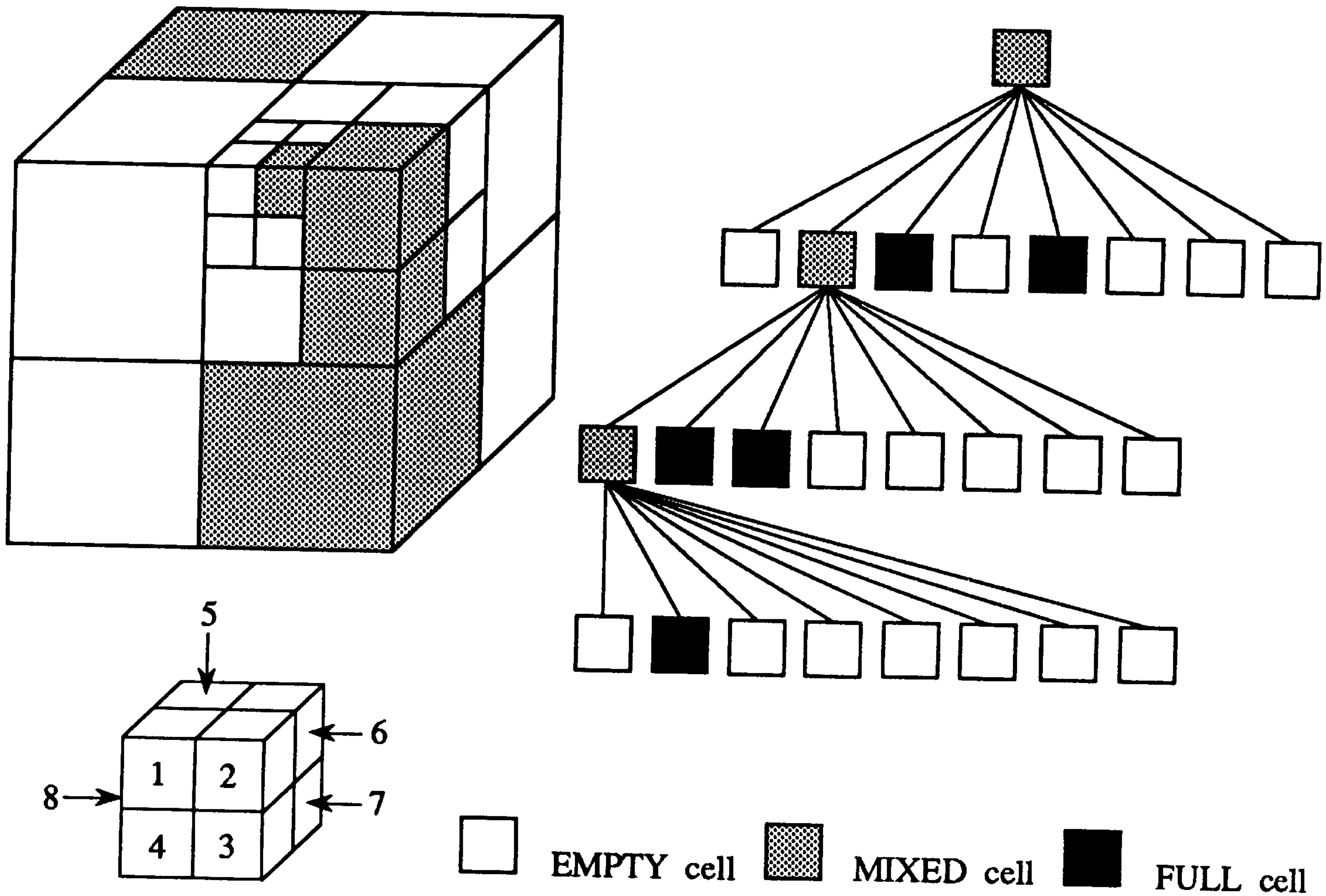
- Cannot find a path in this case even though one exists

# Approximate Cell Decomposition



- Cannot find a path in this case even though one exists
- Solution:
- Distinguish between
  - Cells that are entirely contained in $\mathcal{C}_{obs}$ (*FULL*) and
  - Cells that partially intersect $\mathcal{C}_{obs}$ (*MIXED*)
- Try to find a path using the current set of cells
- If no path found:
  - Subdivide the *MIXED* cells and try again with the new set of cells

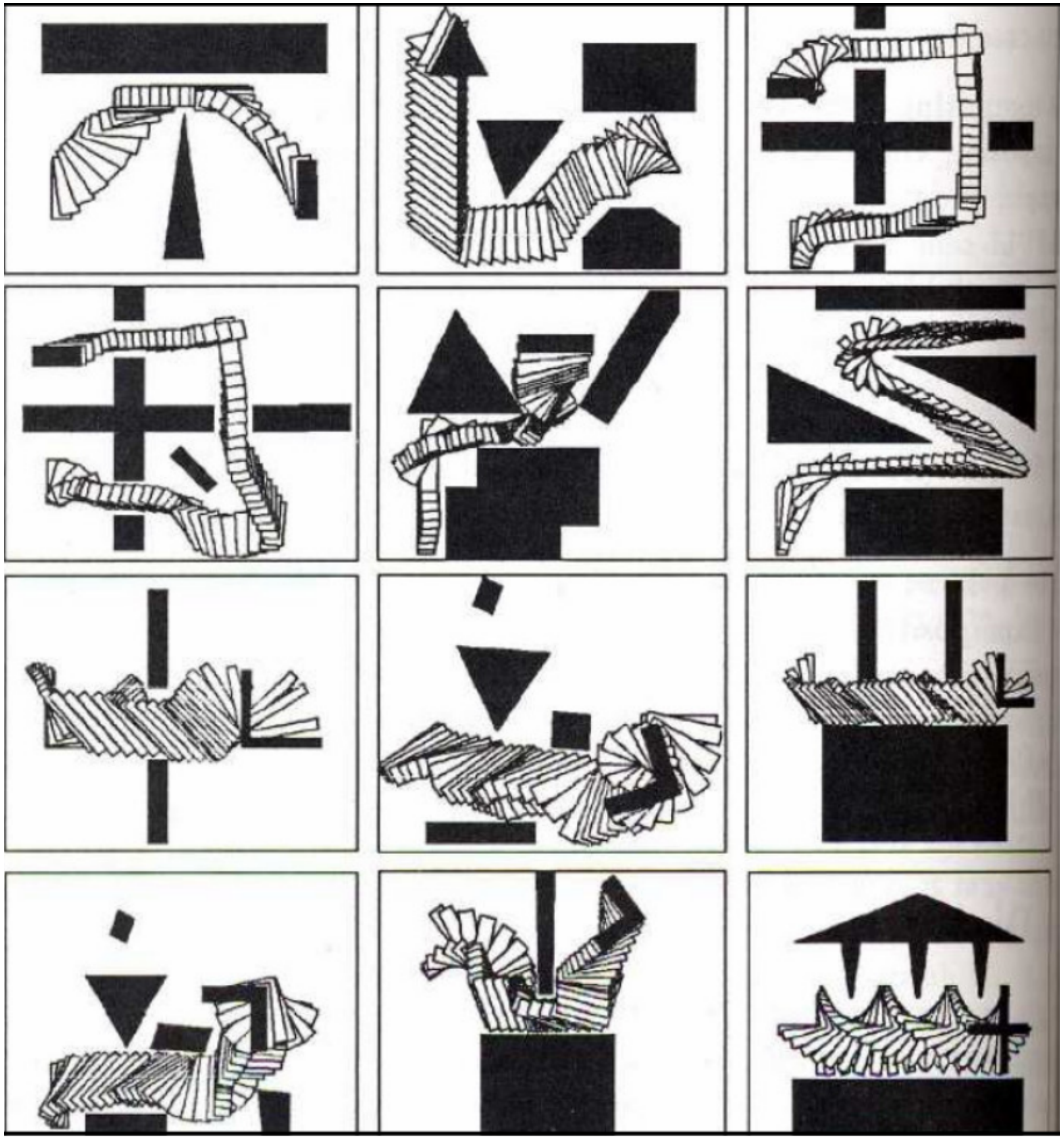
# Approximate Cell Decomposition



# Approximate Cell Decomposition

1. Compute cell decomposition down to some resolution
2. Identify start and goal cells
3. Search for sequence of empty/mixed cells between start and goal cells
4. If no sequence, then exit with no path
5. If sequence of empty cells, then exit with solution
6. If resolution threshold achieved, then exit with failure
7. Decompose further the mixed cells
8. Return to 2

# Approximate Cell Decomposition



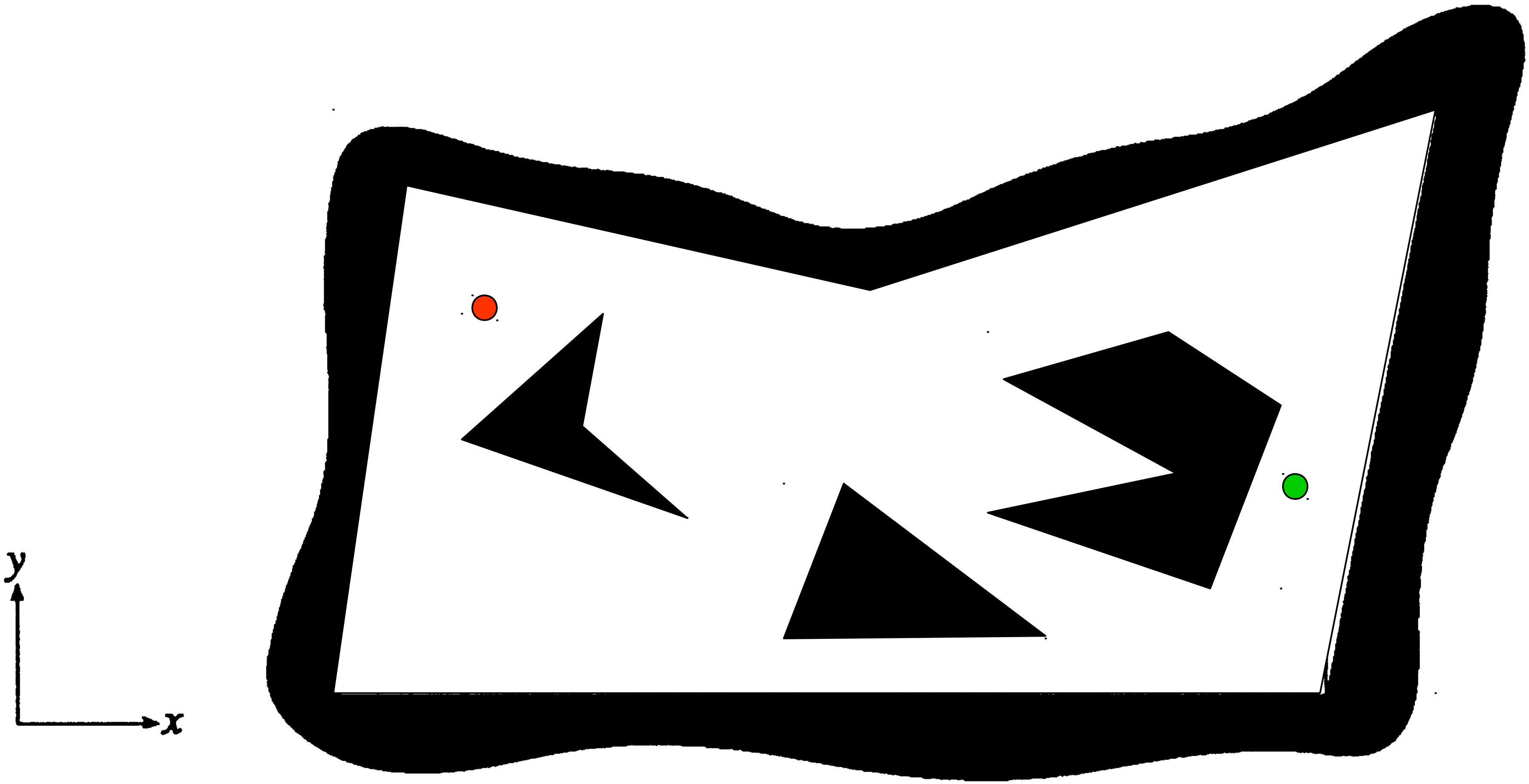
# Approximate Cell Decomposition

- Good:
  - Limited assumptions on obstacle configuration
  - Approach used in practice
  - Find obvious solutions quickly
- Bad:
  - No clear notion of optimality (“best” path)
  - Trade-off completeness/computation
  - Still difficult to use in high dimensions

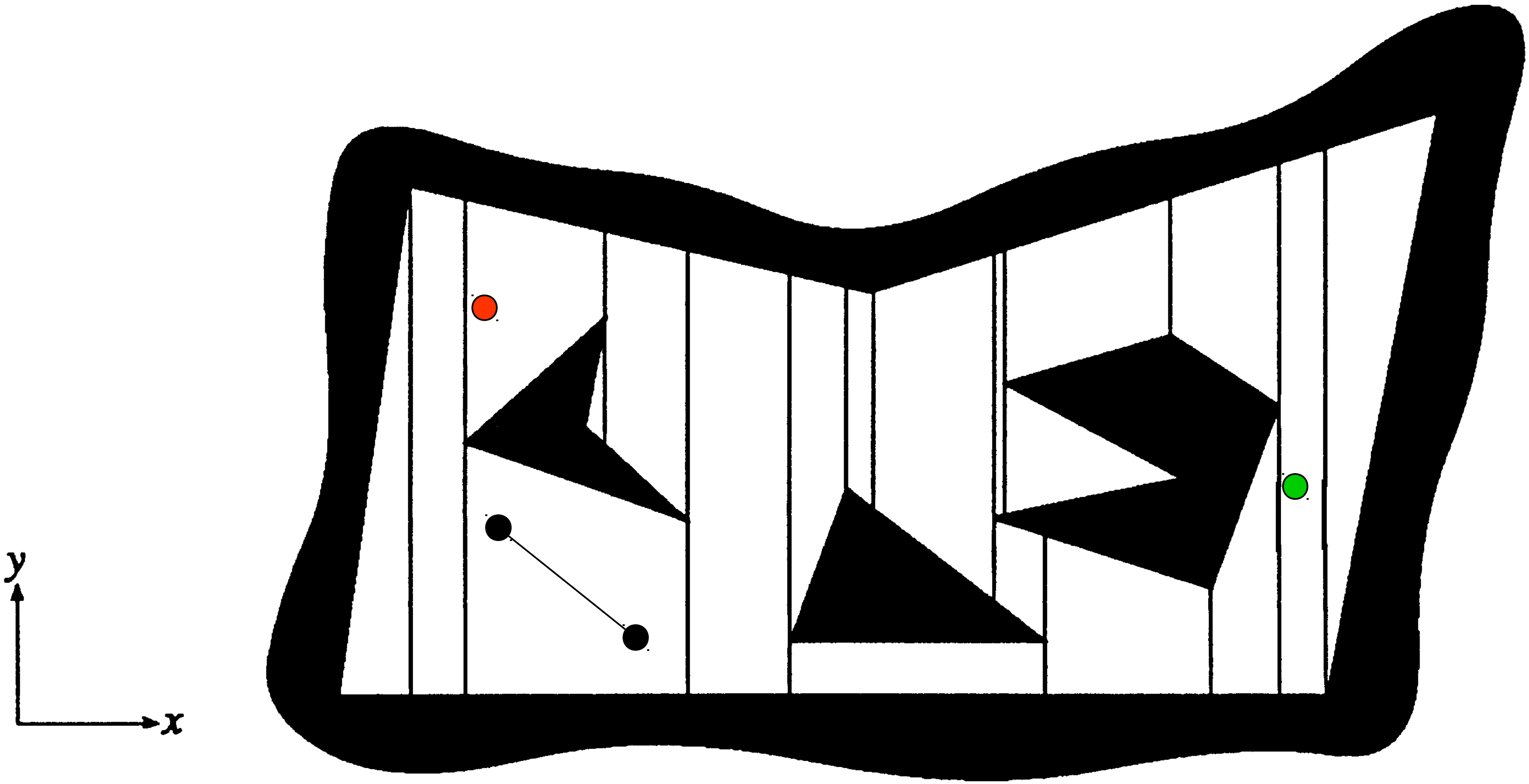
# Approximate Cell Decomposition

- Good:
  - Limited assumptions on obstacle configuration
  - Approach used in practice
  - Find obvious solutions quickly
- Bad:
  - No clear notion of optimality (“best” path)
  - Trade-off completeness/computation
  - Still difficult to use in high dimensions

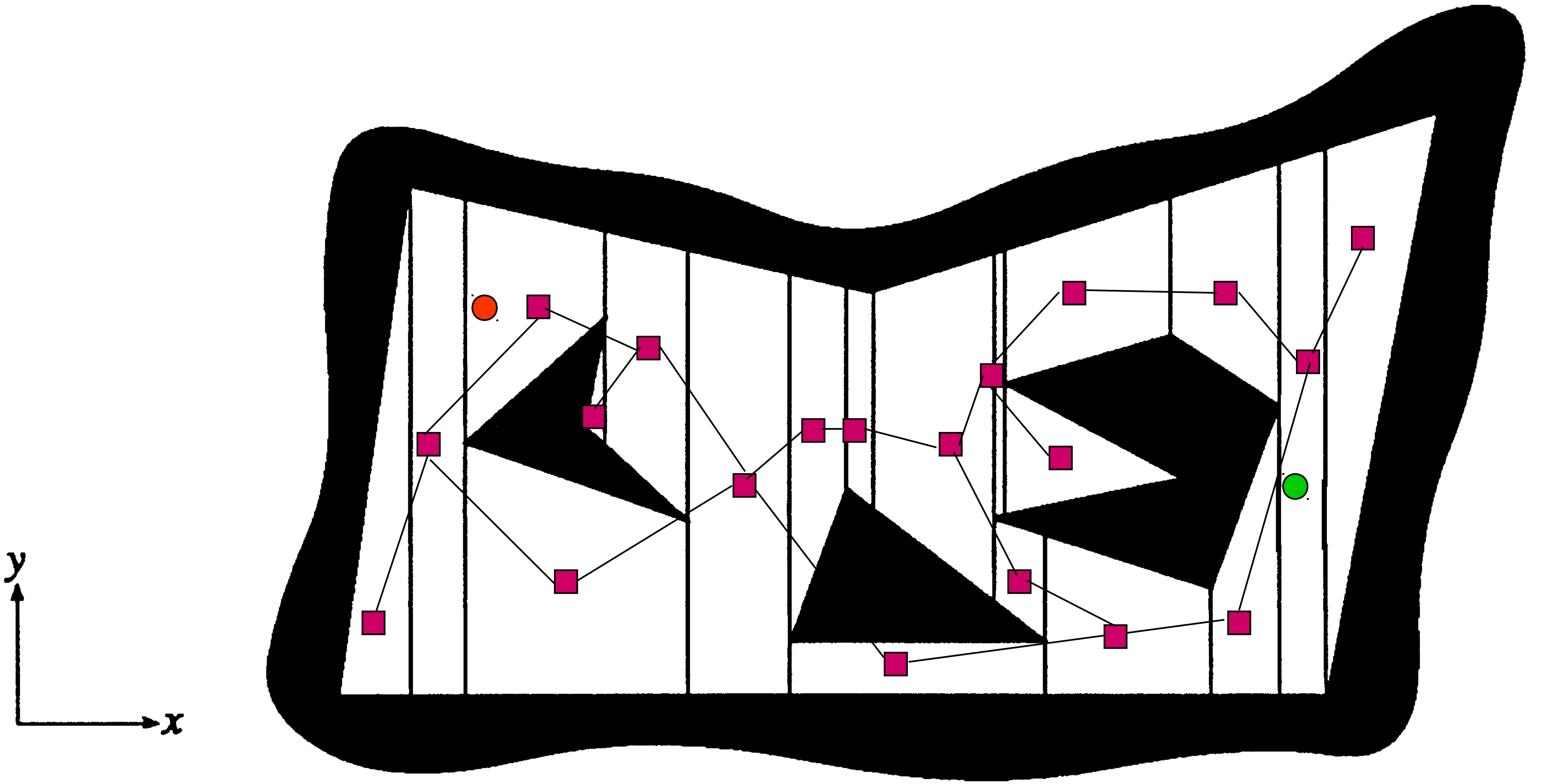
# Exact Cell Decomposition



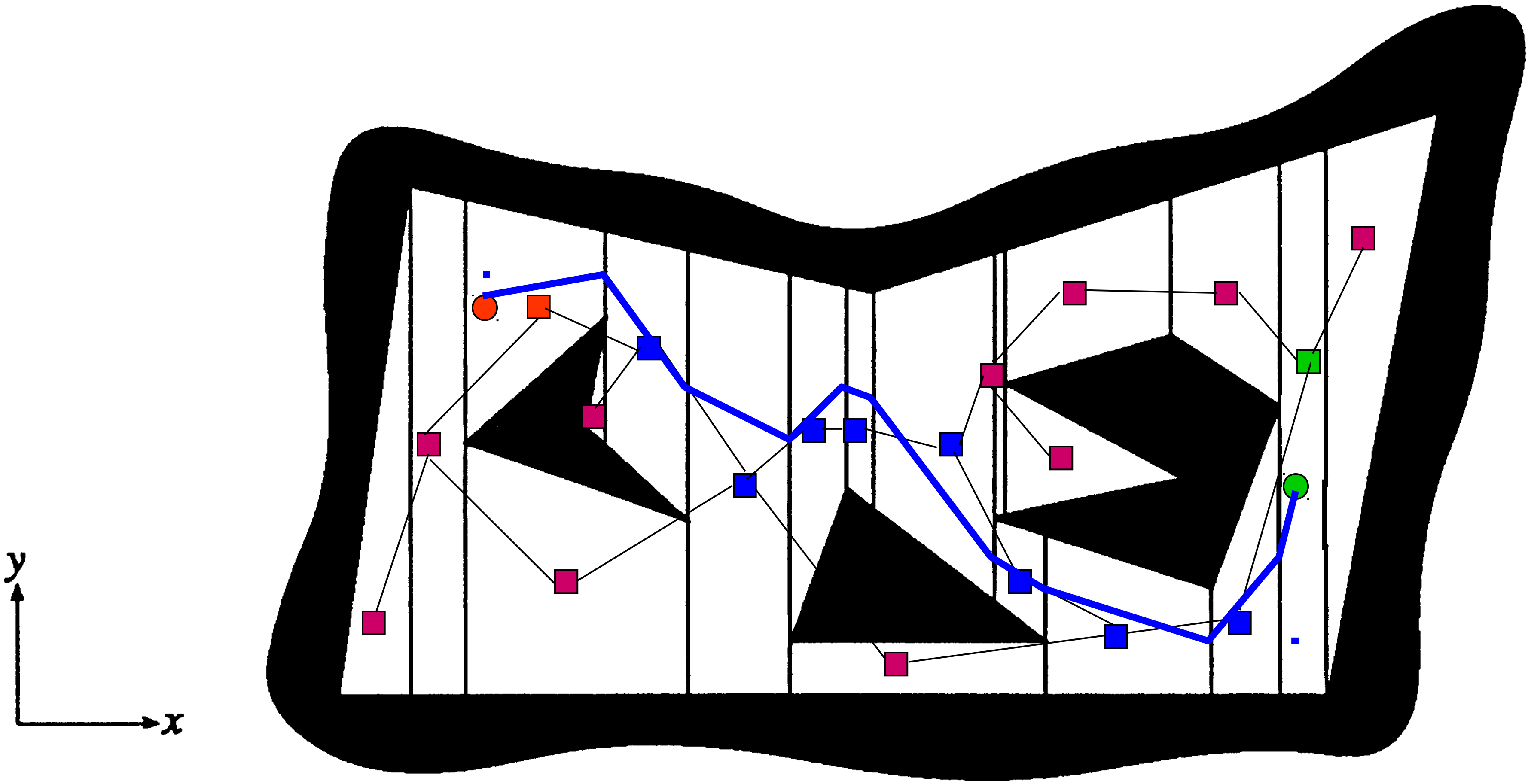
# Exact Cell Decomposition



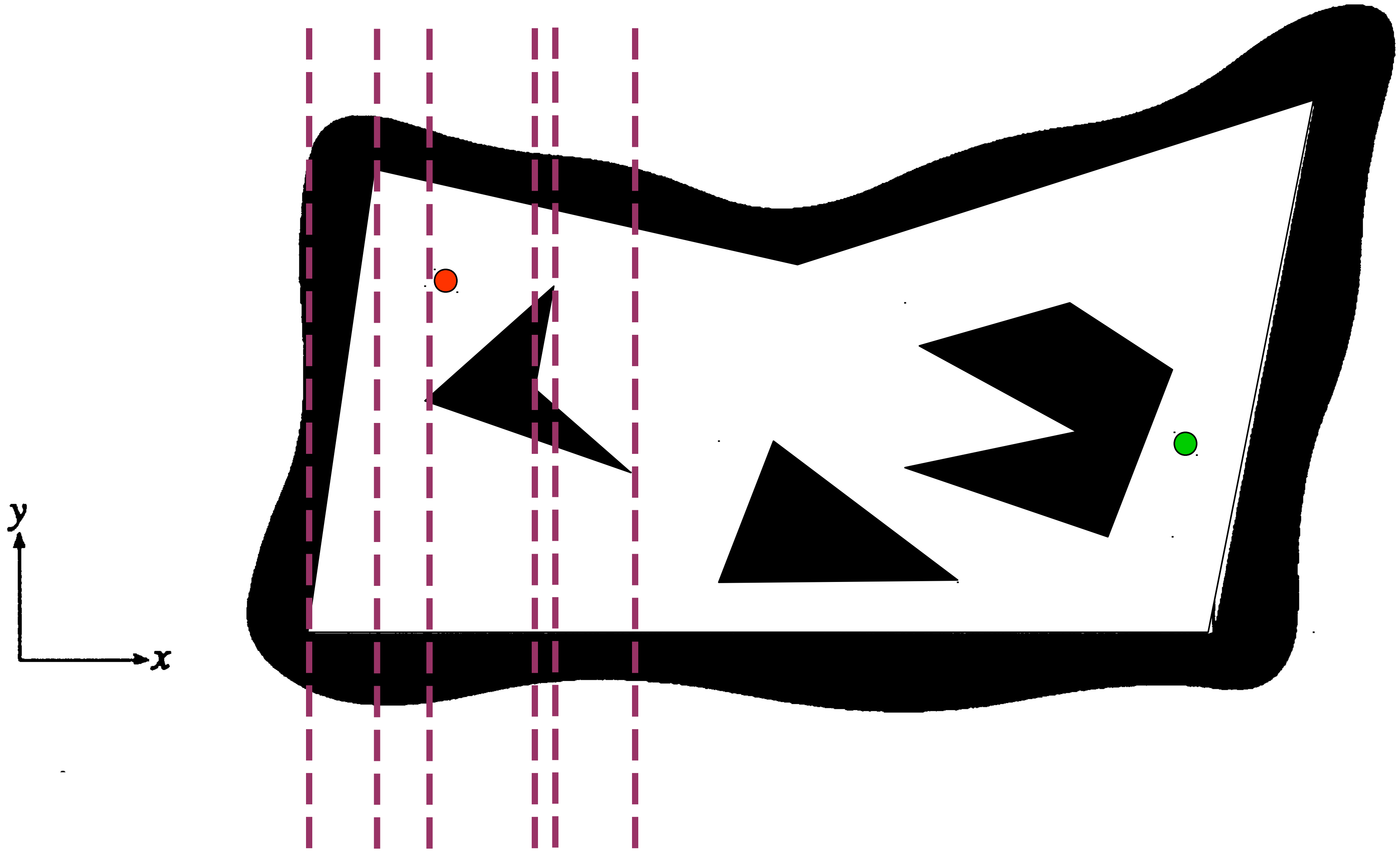
# Exact Cell Decomposition



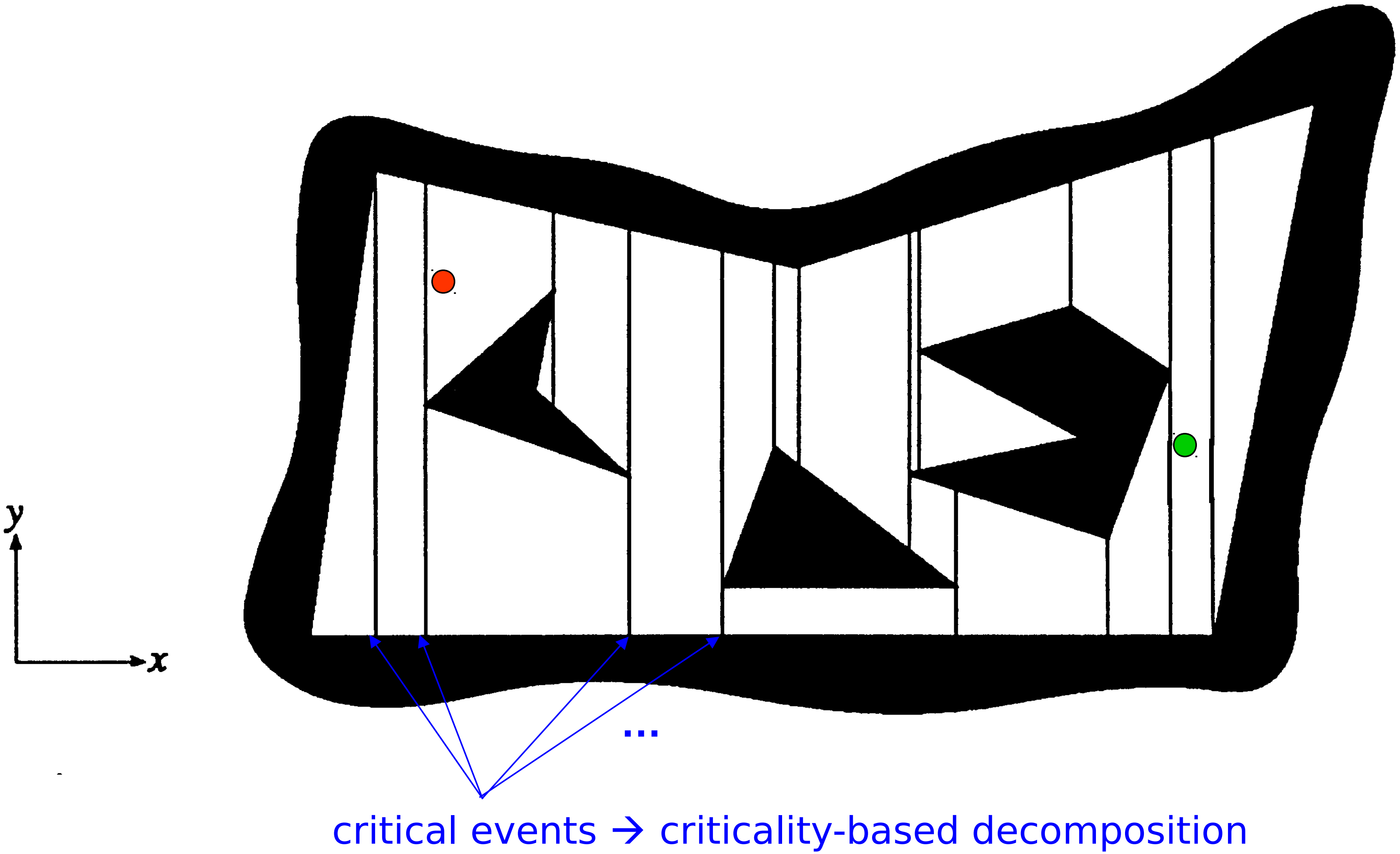
# Exact Cell Decomposition



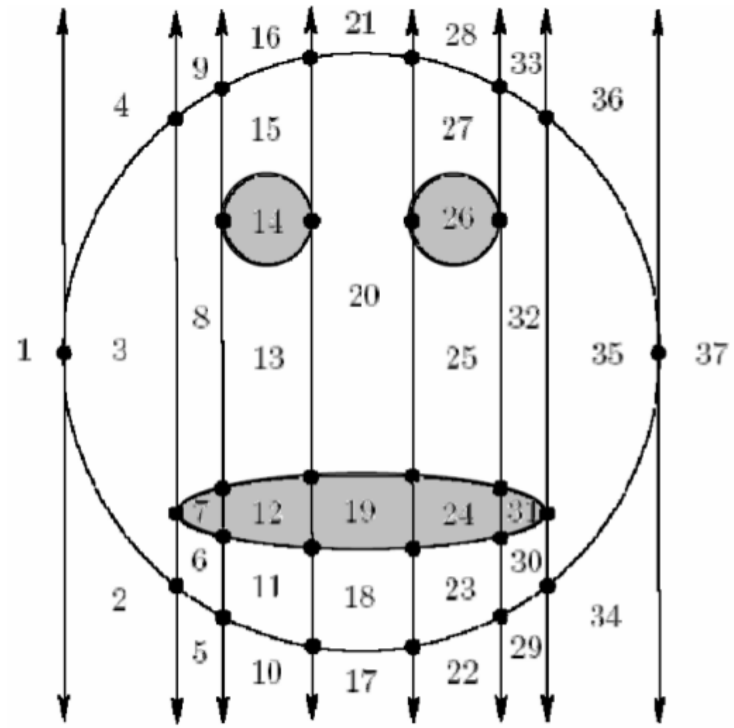
# Exact Cell Decomposition



# Exact Cell Decomposition

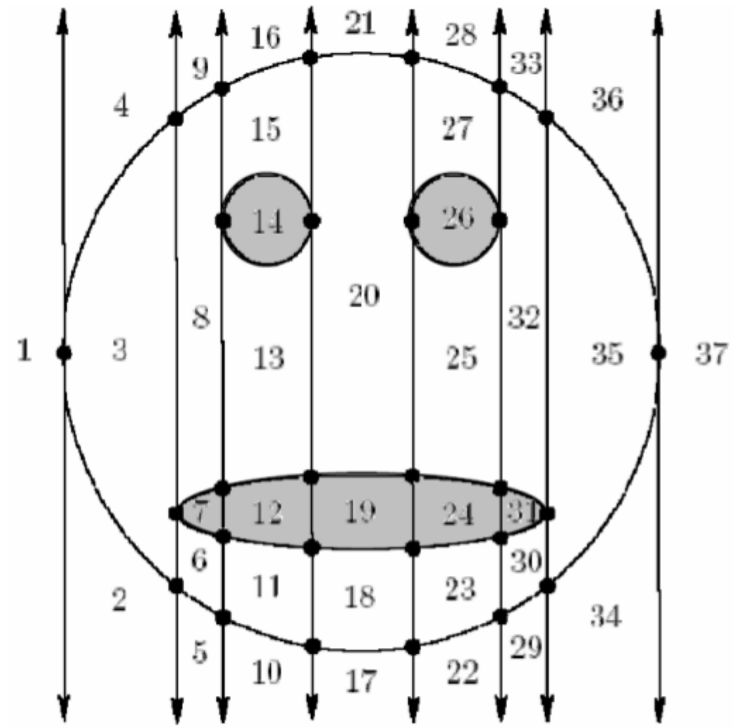


# Exact Cell Decomposition




- A version of exact cell decomposition can be extended to higher dimensions and non-polygonal boundaries ("cylindrical cell decomposition")
- Provides exact solution → completeness
- Expensive and difficult to implement in higher dimensions

# Exact Cell Decomposition




- A version of exact cell decomposition can be extended to higher dimensions and non-polygonal boundaries ("cylindrical cell decomposition")
- Provides exact solution → completeness
- Expensive and difficult to implement in higher dimensions

# Exact Cell Decomposition

The good:
– exact cell decomposition is complete

The bad:
– it doesn't scale well to high dimensions