CSCI 4560/6560 Computational Geometry

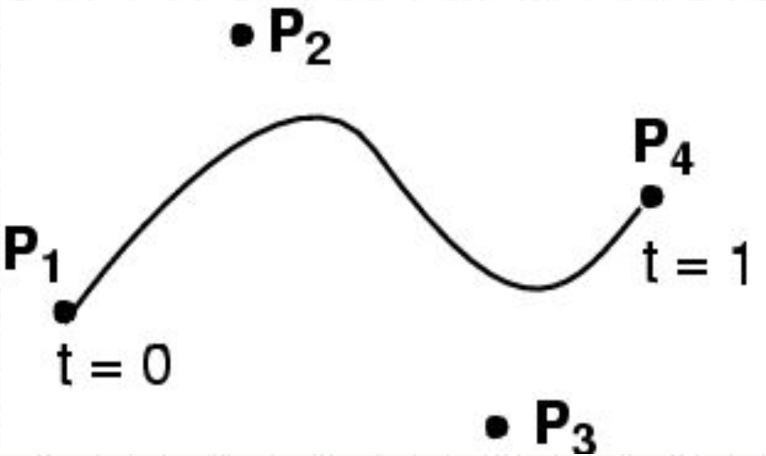
https://www.cs.rpi.edu/~cutler/classes/computationalgeometry/F23/

# Lecture 24: Robot Motion Planning

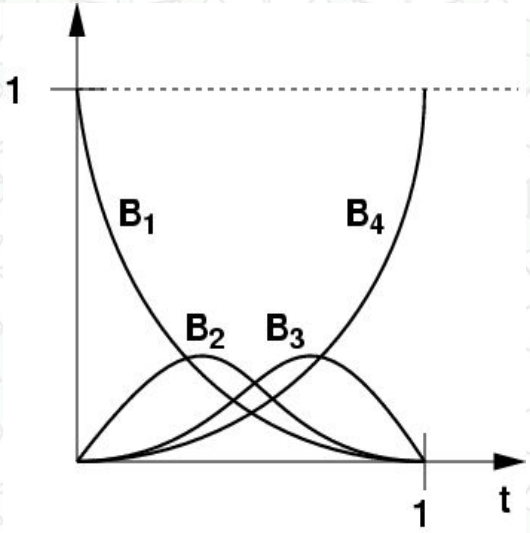
# Outline for Today

- Last Time: Bezier Curves, Polyline Simplification, Clothoid Sketches
- Motivation: Robot Motion Planning
- Previous Lecture: Voronoi Diagram of Segments for Motion Planning
- Degrees of Freedom & Configuration Space
- Trapezoid Map for Motion Planning
- Non-Point, Non-Rotating Robots & Minkowski Sums
- Related Operations: Convolution, Morphology, Accessible Surfaces
- Rotations & Higher Dimensional Configuration Space
- End of Term Schedule: Quiz 2, Sprouts, & Project Presentations

# Cubic Bézier Curve




*Asymmetric:*
*Curve goes through some control points but misses others*




weights sum to 1

**Parametric equation:**
**Function of *t***
***t* varies *0* → *1***

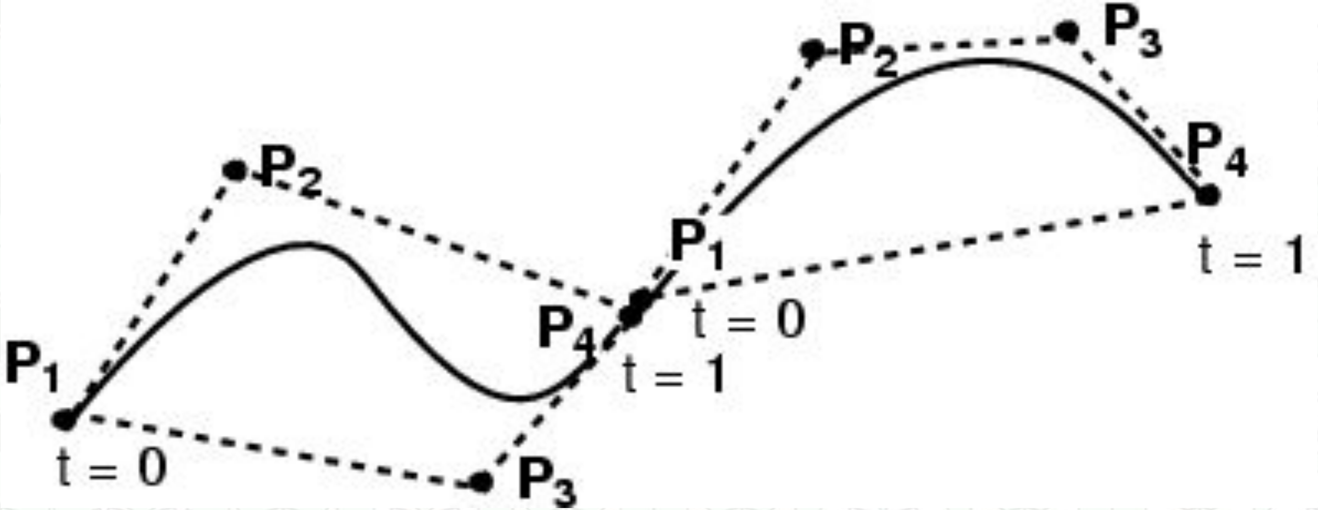
$$Q(t) = (1-t)^3 P_1 + 3t(1-t)^2 P_2 + 3t^2(1-t) P_3 + t^3 P_4$$

control points

# Connecting Cubic Bézier Curves

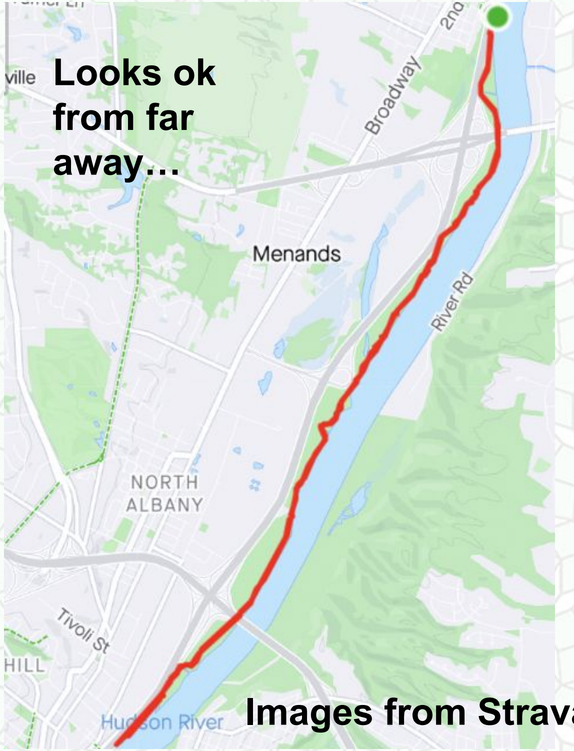
- How can we guarantee $C^0$ continuity?
- How can we guarantee $G^1$ continuity?
- How can we guarantee $C^1$ continuity?
- Can't guarantee higher $C^2$ or higher continuity

Asymmetric: Curve goes through some control points but misses others



# Noisy GPS Running Data

- Can overestimate distance by ~10% !!



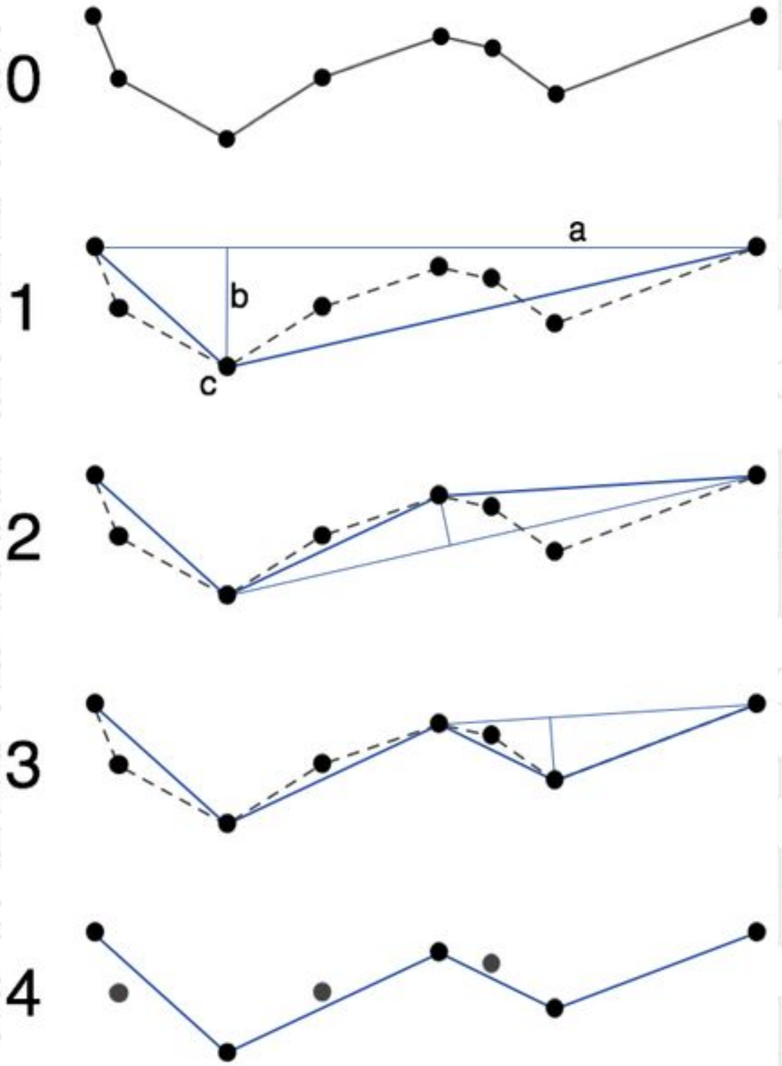


Images from Strava


# Polyline Simplification: Ramer–Douglas–Peucker

- Originally developed for cartography
- Reduce number of points necessary to represent a polyline
- Identify most important points
- Discards points that are < ε from the simplified shape

https://commons.wikimedia.org/wiki/File:Douglas_Peucker.png



# Long Tiny Loops by Dan Aminzade

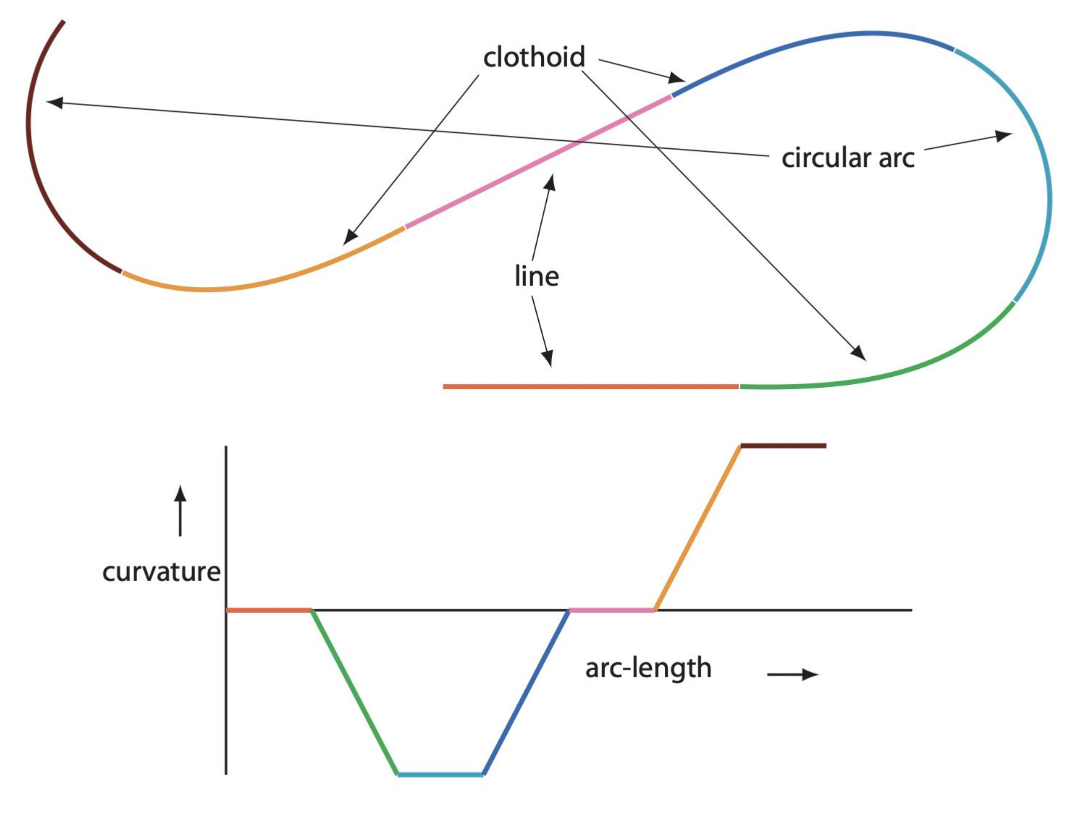
- Extract GPS data from Strava API
- Ramer-Douglas-Peucker:
  Simplify input (remove false
  positive intersections due to noise)
- Verify closed loop
- Check for segment intersections
- Compute convex hull
- Rotating calipers maximum diameter

→ Compute final score
= distance / max diameter


https://longtinyloop.com/faq

# Piecewise Clothoid + Circular Arc + Line

- Aesthetically pleasing
- Fairness
- Can ensure G2 or G3 continuity
- Also model sharp discontinuities as appropriate




"Sketching Piecewise Clothoid Curves"
McCrae & Singh, 2008

# Outline for Today

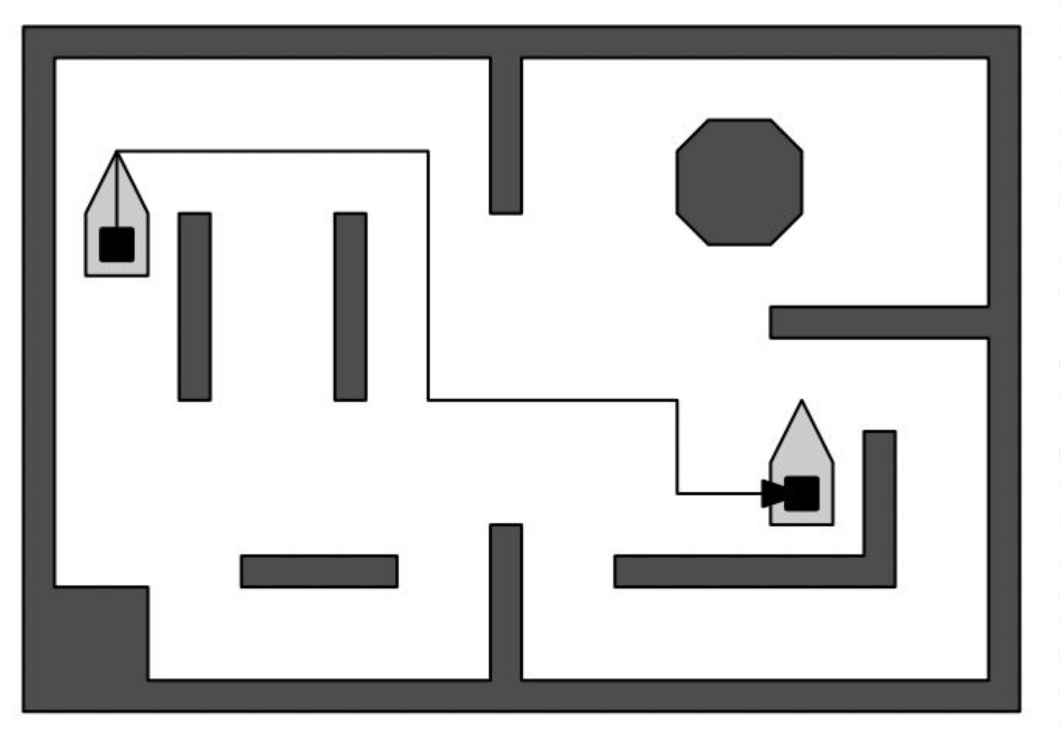
- Last Time: Bezier Curves, Polyline Simplification, Clothoid Sketches
- Motivation: Robot Motion Planning
- Previous Lecture: Voronoi Diagram of Segments for Motion Planning
- Degrees of Freedom & Configuration Space
- Trapezoid Map for Motion Planning
- Non-Point, Non-Rotating Robots & Minkowski Sums
- Related Operations: Convolution, Morphology, Accessible Surfaces
- Rotations & Higher Dimensional Configuration Space
- End of Term Schedule: Quiz 2, Sprouts, & Project Presentations

# Motivation: Robot Motion Planning

- 2D (or 3D)
- Navigate from starting location to end location

- Avoid all obstacles
- Touching/sliding along the obstacles may be allowed (or disallowed)

- Rotation may be allowed (or disallowed)



*Computational Geometry Algorithms and Applications*,
de Berg, Cheong, van Kreveld and Overmars, Chapter 13

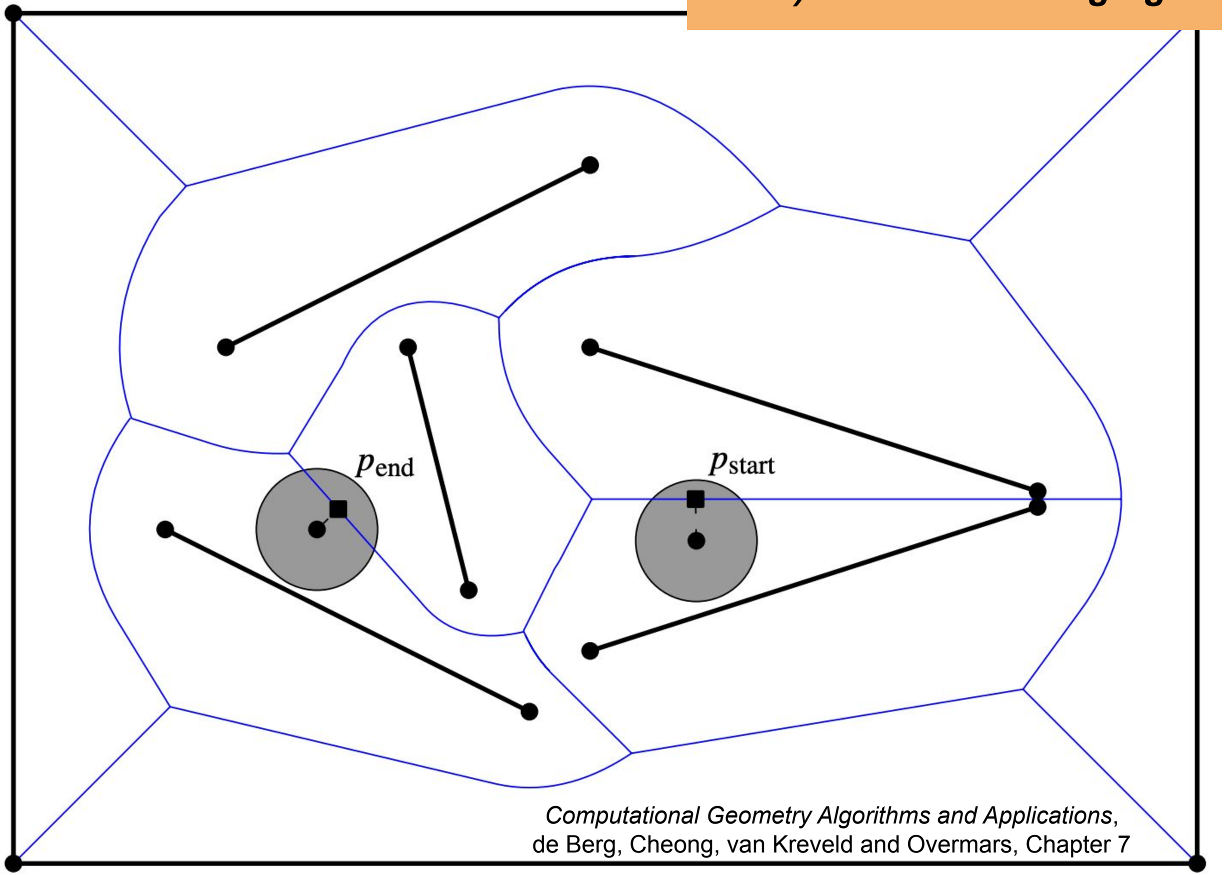
# Outline for Today

- Last Time: Bezier Curves, Polyline Simplification, Clothoid Sketches
- Motivation: Robot Motion Planning
- Previous Lecture: Voronoi Diagram of Segments for Motion Planning
- Degrees of Freedom & Configuration Space
- Trapezoid Map for Motion Planning
- Non-Point, Non-Rotating Robots & Minkowski Sums
- Related Operations: Convolution, Morphology, Accessible Surfaces
- Rotations & Higher Dimensional Configuration Space
- End of Term Schedule: Quiz 2, Sprouts, & Project Presentations

# Voronoi Diagram of Line Segments

***Proper implementation (robustness, floating point, etc) is extra challenging***

- Voronoi Diagram w/ segments has parabolic curved segments
- But is still *O(n)* in complexity - (# of segments)
- And can be computed in *O(n log n)*
- *But why is this useful?*

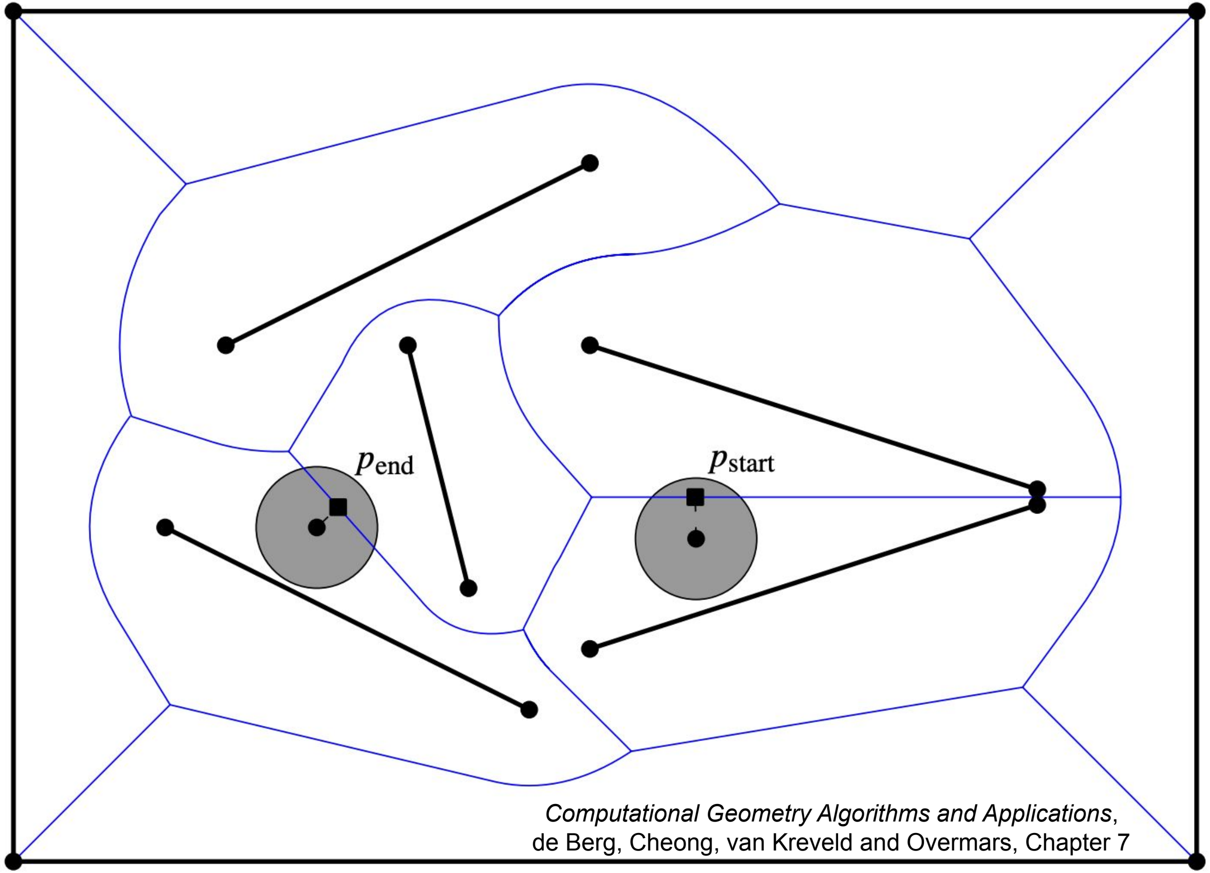



*Computational Geometry Algorithms and Applications*, de Berg, Cheong, van Kreveld and Overmars, Chapter 7

# Application: Robotics & Motion Planning

- Let's move a circular/disk robot from the start position to the end position.

- Step 1: Project the robot center to the closest Voronoi edge (line segment or parabolic curve)

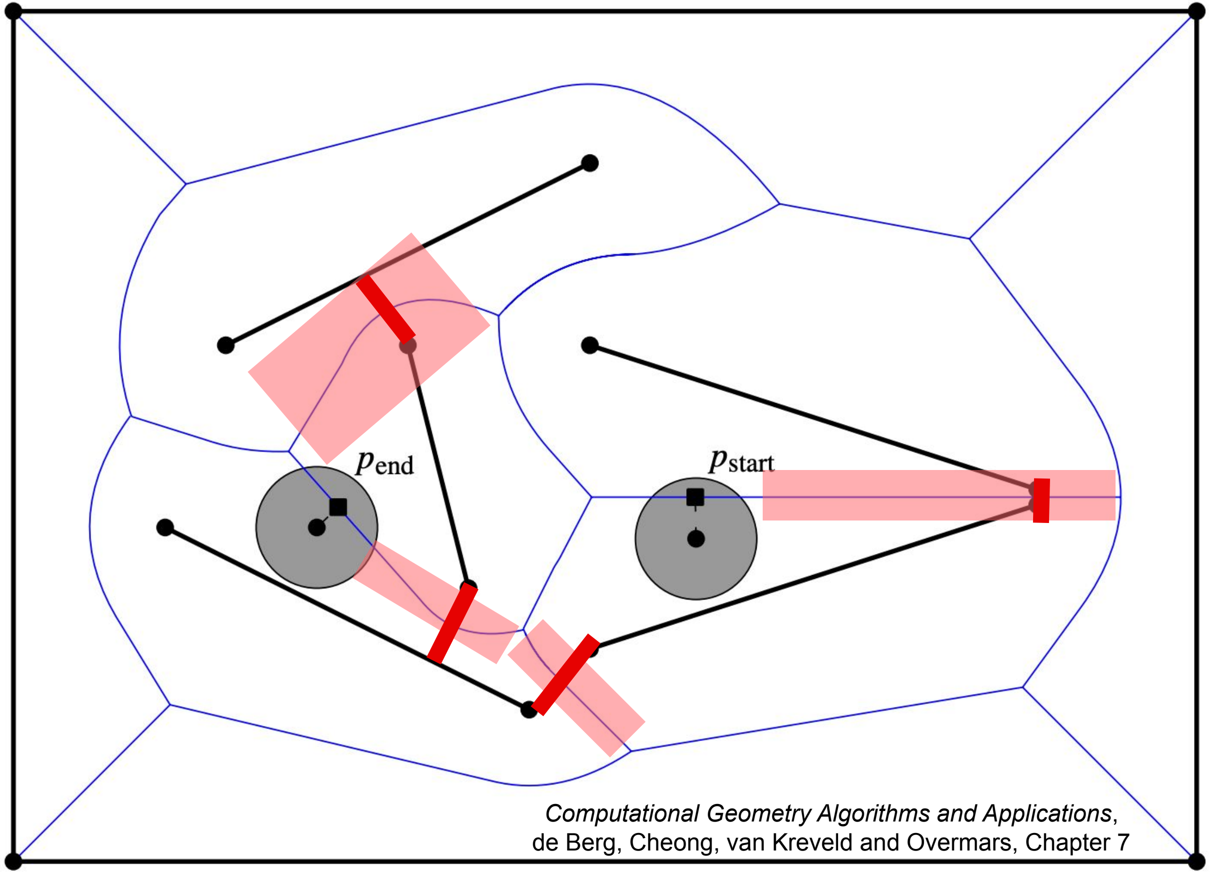



*Computational Geometry Algorithms and Applications*,
de Berg, Cheong, van Kreveld and Overmars, Chapter 7

# Application: Robotics & Motion Planning

- Step 1: Project the robot center to the closest Voronoi edge (line segment or parabolic curve)

- Step 2: Remove edges from the diagram graph with smallest distance to segment < radius.

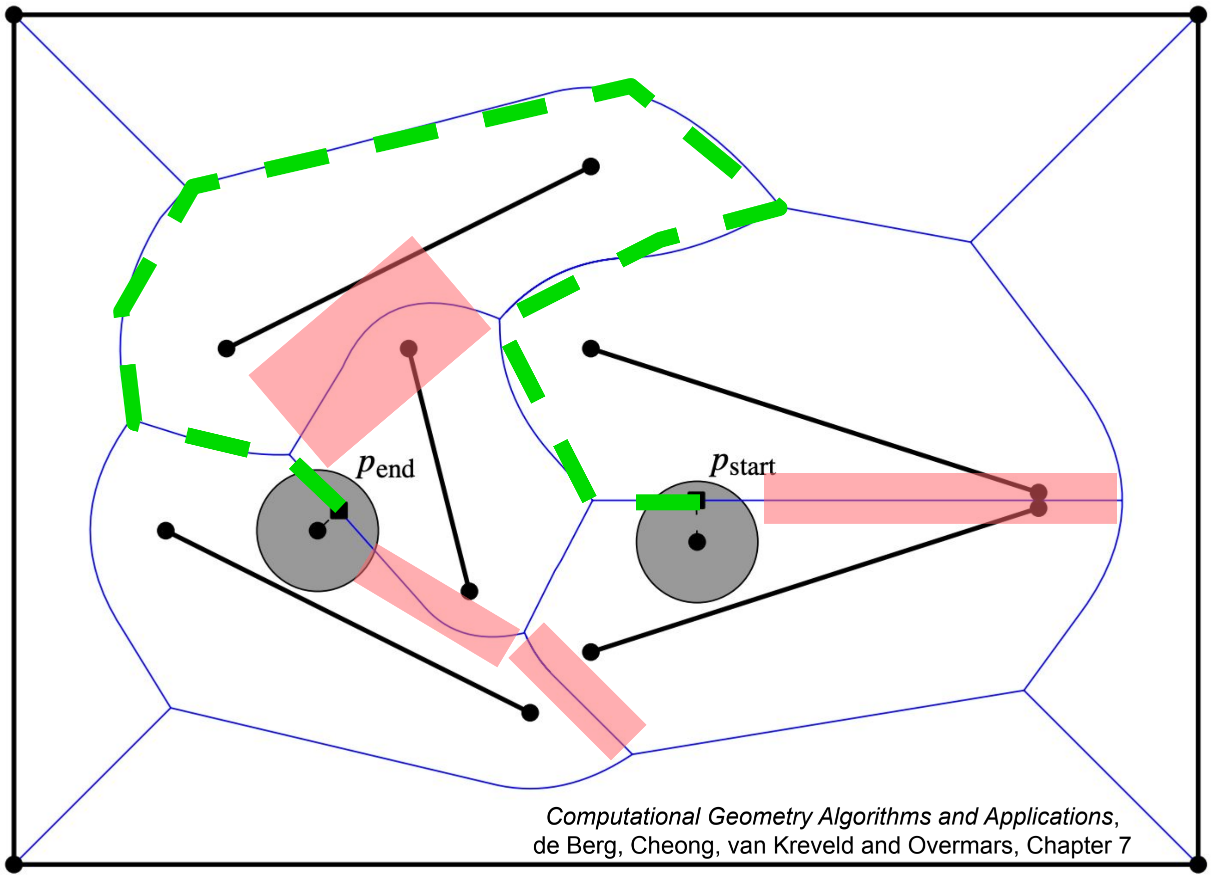



*Computational Geometry Algorithms and Applications*, de Berg, Cheong, van Kreveld and Overmars, Chapter 7

# Application: Robotics & Motion Planning

- Step 2: Remove edges from the diagram graph with smallest distance to segment < radius.

- Step 3: Search the remaining graph for a connected path from start to end.



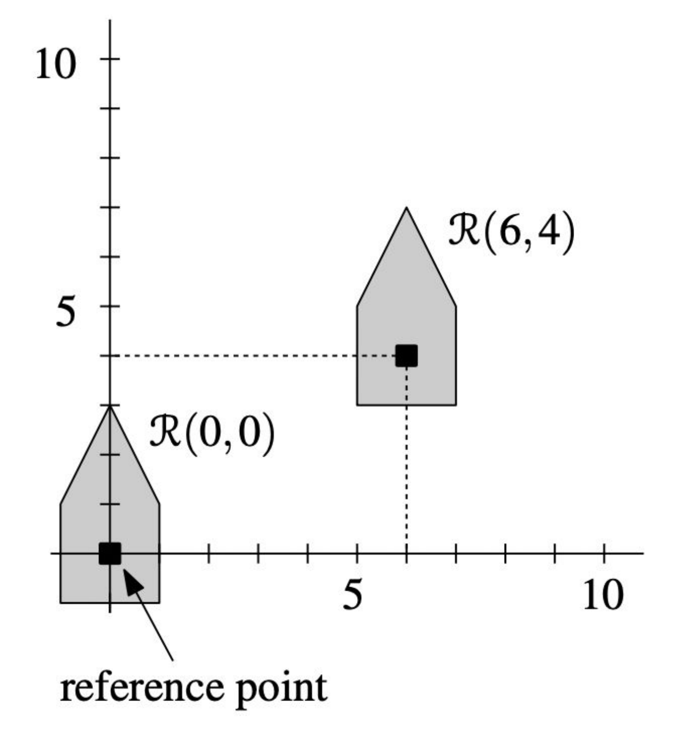

*Computational Geometry Algorithms and Applications*,
de Berg, Cheong, van Kreveld and Overmars, Chapter 7

# Outline for Today

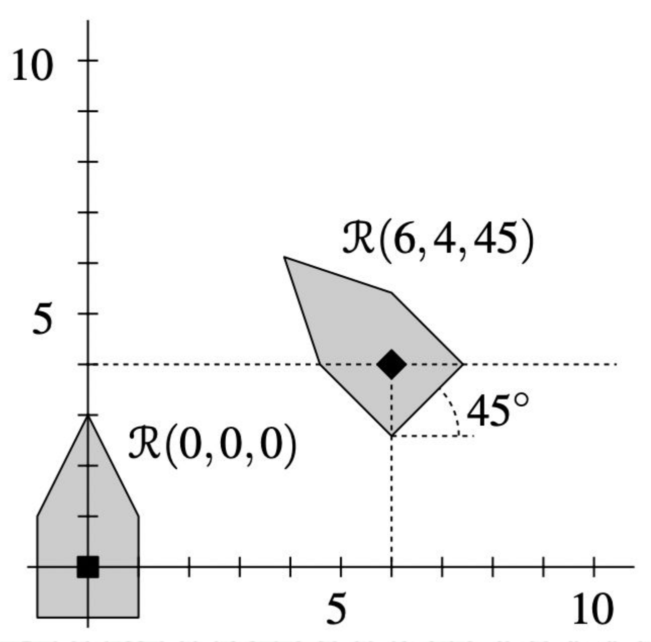
- Last Time: Bezier Curves, Polyline Simplification, Clothoid Sketches
- Motivation: Robot Motion Planning
- Previous Lecture: Voronoi Diagram of Segments for Motion Planning
- Degrees of Freedom & Configuration Space
- Trapezoid Map for Motion Planning
- Non-Point, Non-Rotating Robots & Minkowski Sums
- Related Operations: Convolution, Morphology, Accessible Surfaces
- Rotations & Higher Dimensional Configuration Space
- End of Term Schedule: Quiz 2, Sprouts, & Project Presentations

# Robot Degree of Freedom (DOF)

**2D w/ Translation only → 2 DOF**



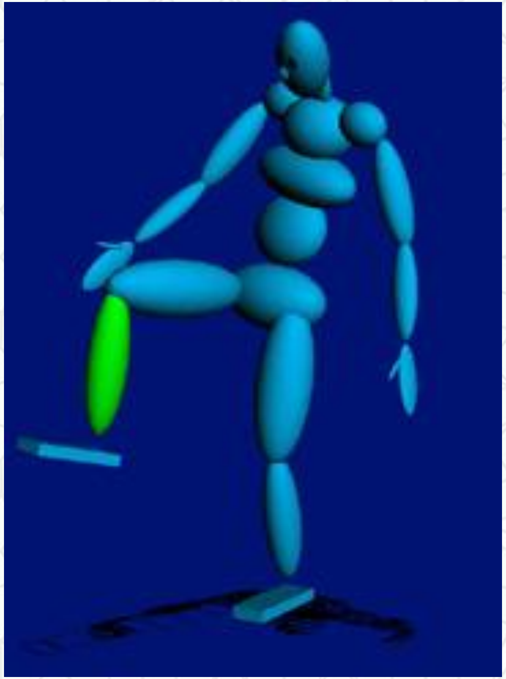

**2D w/ Translation & Rotation → 3 DOF**



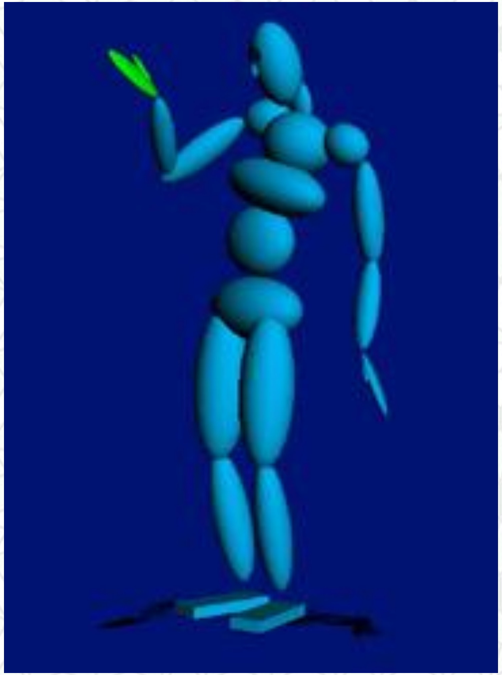

*Computational Geometry Algorithms and Applications*,
de Berg, Cheong, van Kreveld and Overmars, Chapter 13

# Degree of Freedom (DOF)

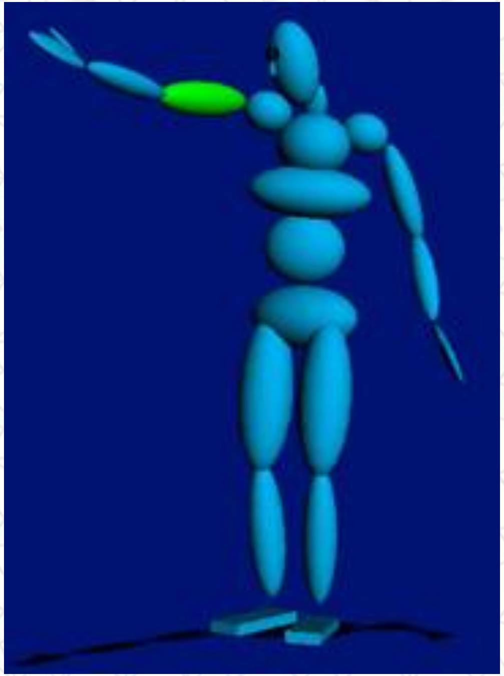
- 3D w/ Translation & up to 3 Rotational DOF → up to 6 total DOF



1 Rotational DOF: knee



2 Rotational DOF: wrist

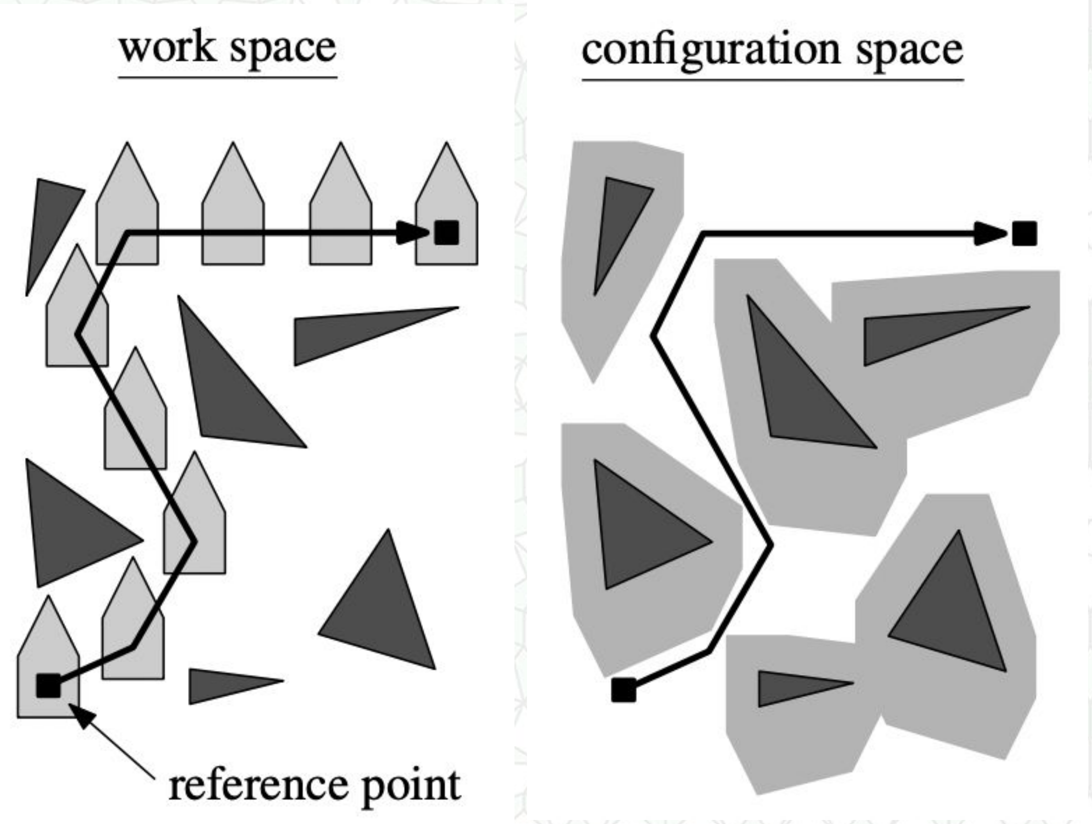


3 Rotational DOF: arm

# Configuration Space

**2D w/ translation only → 2 DOF**

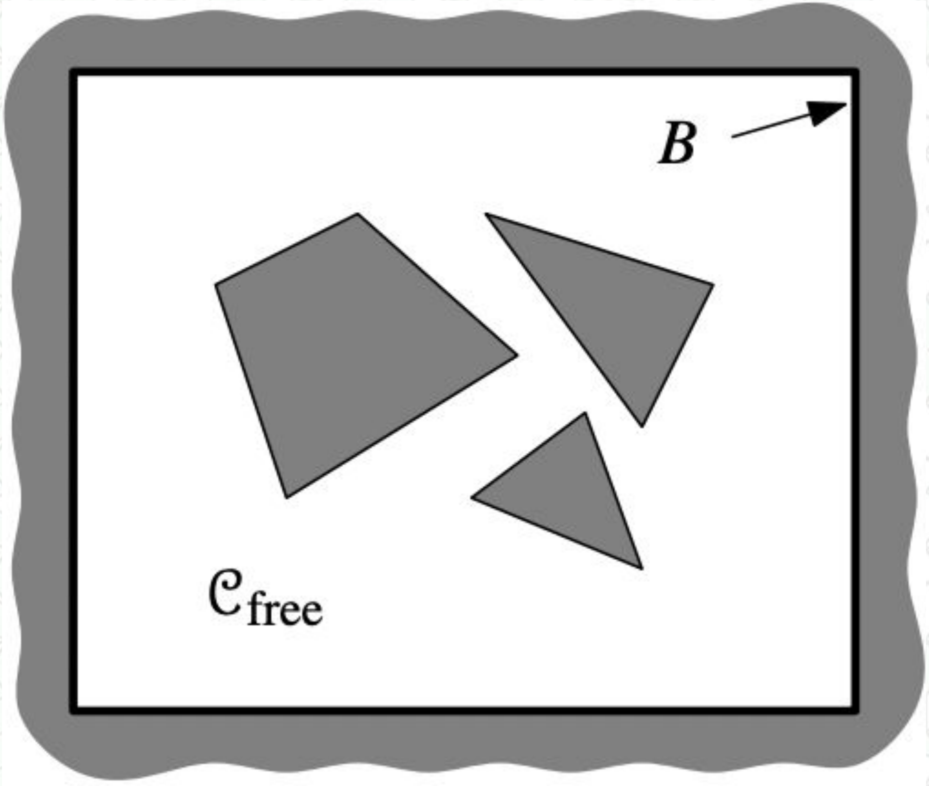
- The dimensions of configuration space match the DOF of the robot
- Usually configuration space is higher dimensional than the environment/workspace
- It is often useful to construct, visualize, and even solve the problem in “configuration space”




*Computational Geometry Algorithms and Applications*,
de Berg, Cheong, van Kreveld and Overmars, Chapter 13

# Determine the Boundaries of the Free Space

- Initially assume a point robot (rotation is thus irrelevant)
- How do we efficiently represent & plan within this free space?



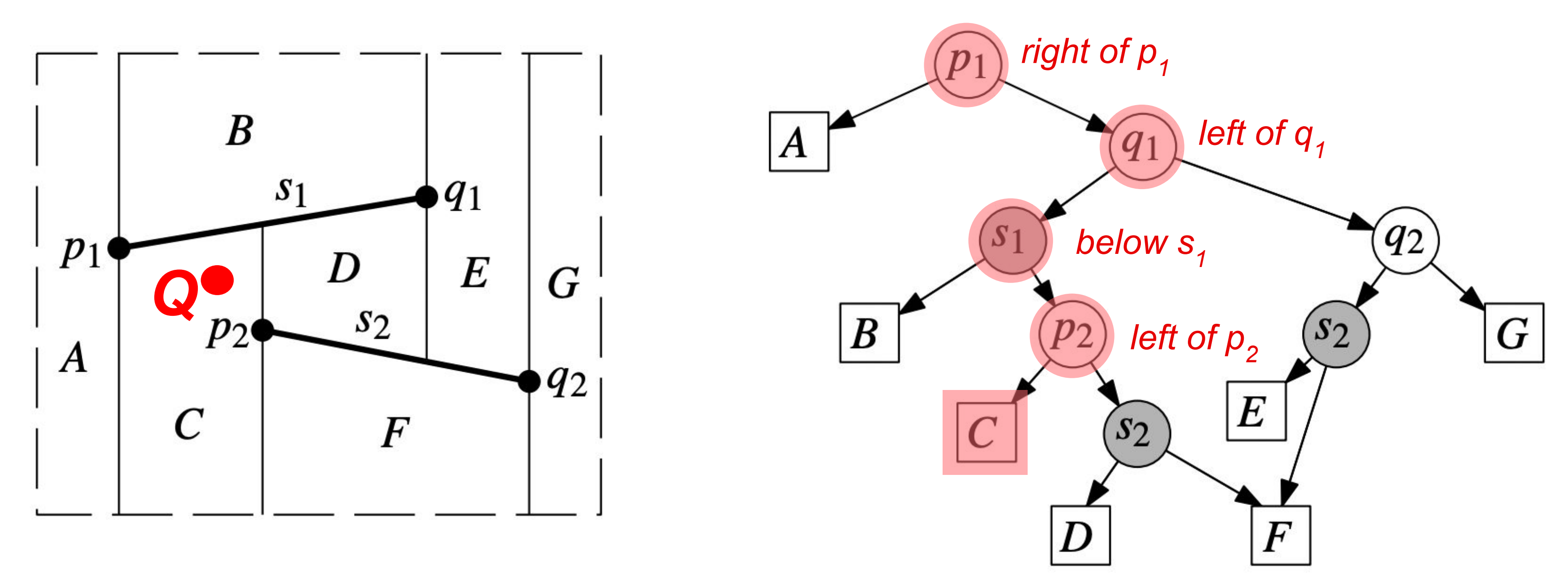

*Computational Geometry Algorithms and Applications*,
de Berg, Cheong, van Kreveld and Overmars, Chapter 13

# Outline for Today

- Last Time: Bezier Curves, Polyline Simplification, Clothoid Sketches
- Motivation: Robot Motion Planning
- Previous Lecture: Voronoi Diagram of Segments for Motion Planning
- Degrees of Freedom & Configuration Space
- Trapezoid Map for Motion Planning
- Non-Point, Non-Rotating Robots & Minkowski Sums
- Related Operations: Convolution, Morphology, Accessible Surfaces
- Rotations & Higher Dimensional Configuration Space
- End of Term Schedule: Quiz 2, Sprouts, & Project Presentations

# Trapezoidal Map & Directed Acyclic Graph

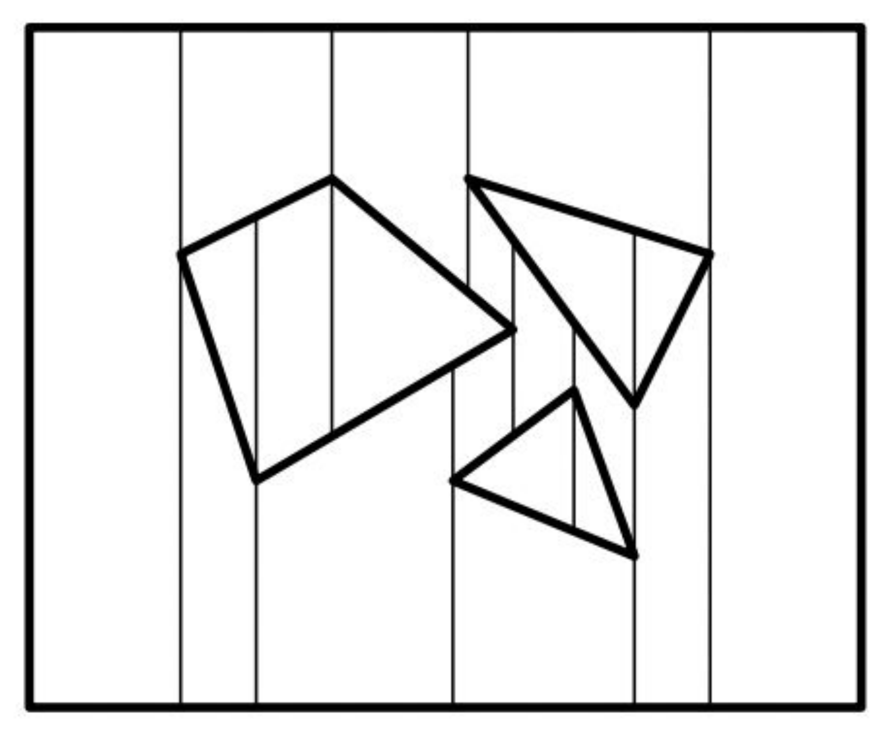
- *n* = # of segments
- size (# of nodes) = *O(n)*
- height = *O(log n) expected* (using Randomized Incremental Construction)



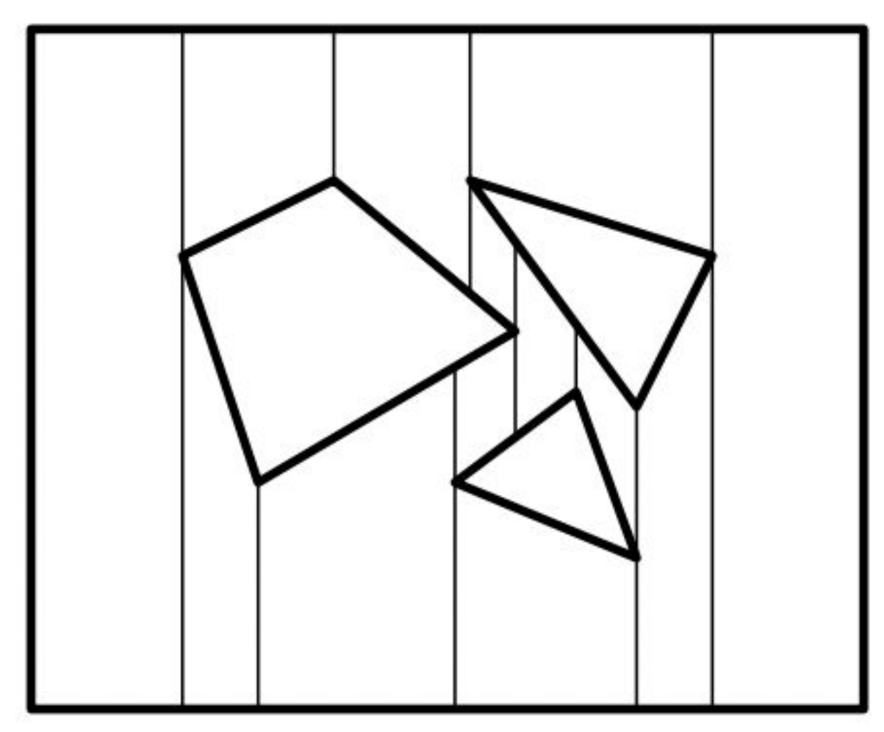

*Computational Geometry Algorithms and Applications*, de Berg, Cheong, van Kreveld and Overmars, Chapter 6

# Build Trapezoidal Map of Free Space

*Insert all obstacle boundaries*



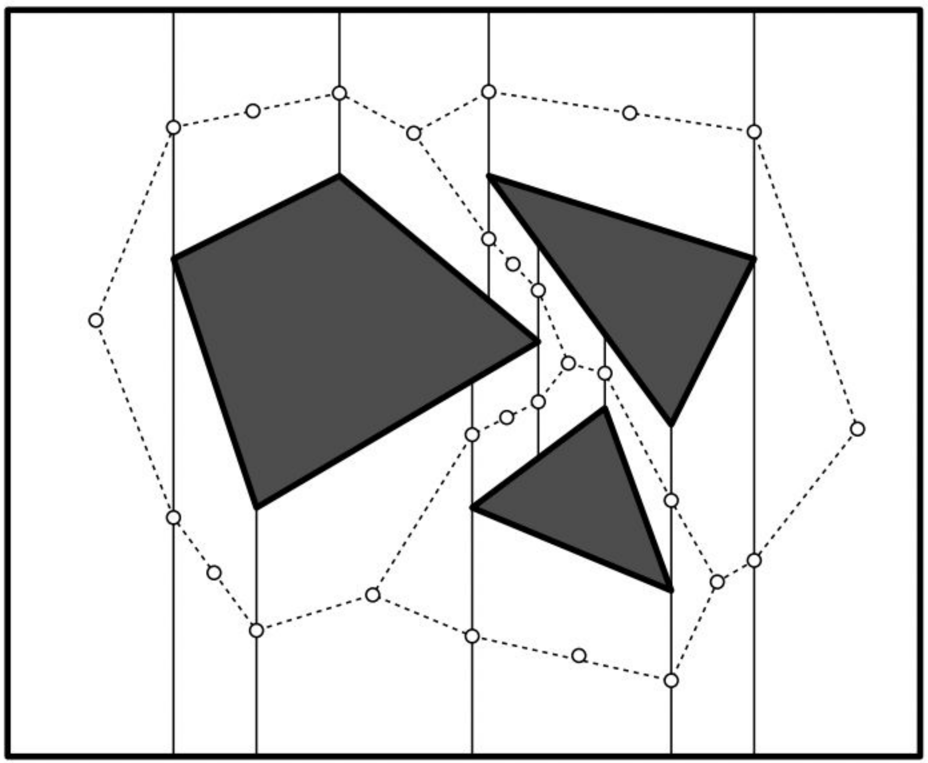
*Remove trapezoids inside obstacles*



*Computational Geometry Algorithms and Applications*,
de Berg, Cheong, van Kreveld and Overmars, Chapter 13

# Motion Planning Graph

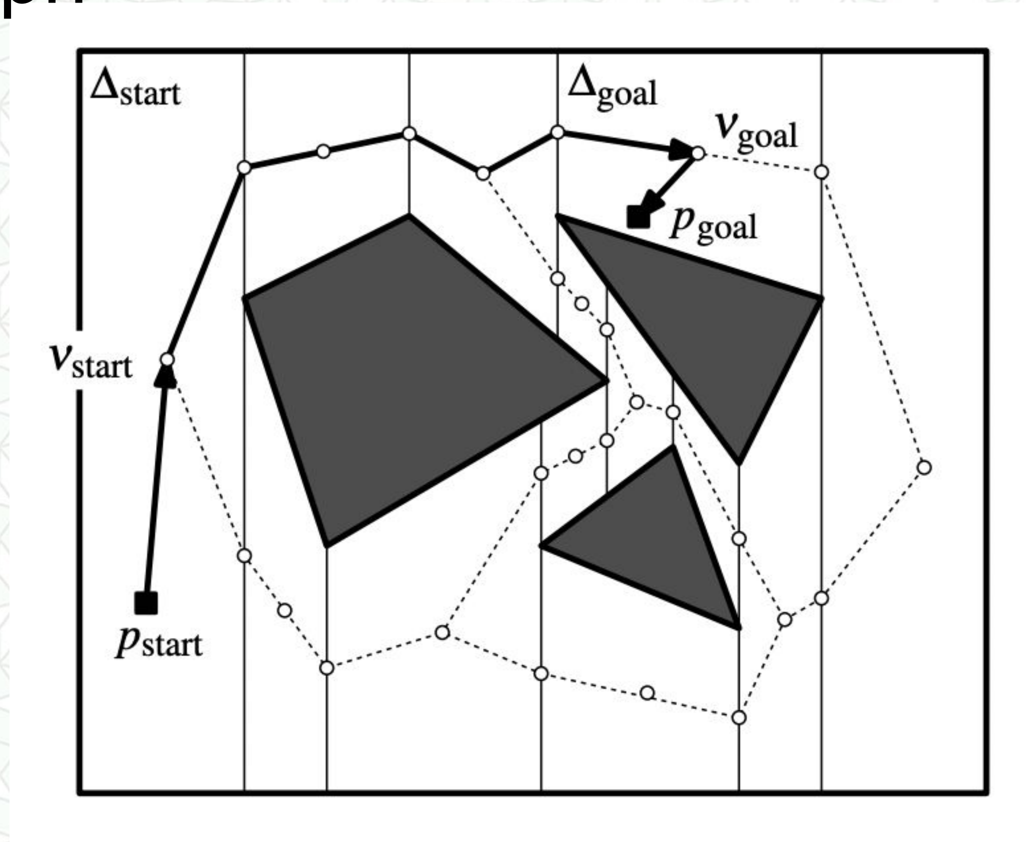
- Add graph node within each trapezoid
- Add graph node at midpoint of each vertical edge
- Connect two graph nodes if they share a vertical edge



*Computational Geometry Algorithms and Applications*,
de Berg, Cheong, van Kreveld and Overmars, Chapter 13

# Motion Planning Graph

- Locate which trapezoid contains the start & end points
- Follow a straight line path from the start point to the graph node within the containing trapezoid
- Perform breadth first search on the graph to find a path from start to end (if any exists)

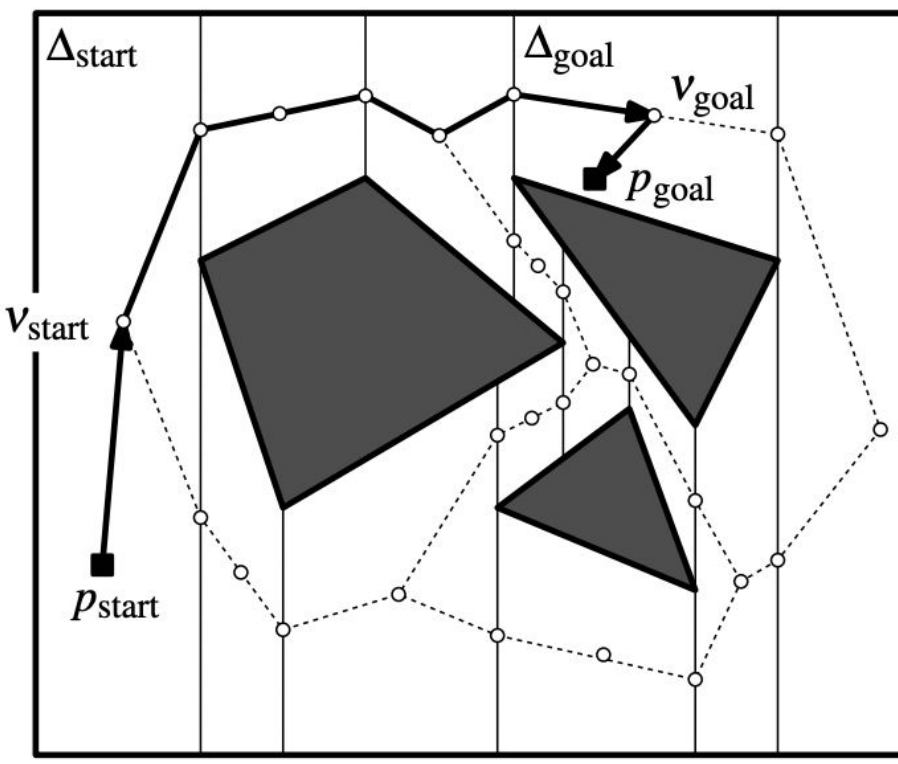



*Computational Geometry Algorithms and Applications*,
de Berg, Cheong, van Kreveld and Overmars, Chapter 13

# Motion Planning Graph - Analysis

where *n* is the # of line segments for the obstacles + environment boundary

- Size of Trapezoid Map
- Build Trapezoid Map
- Locate start/end trapezoid
- Breadth first search

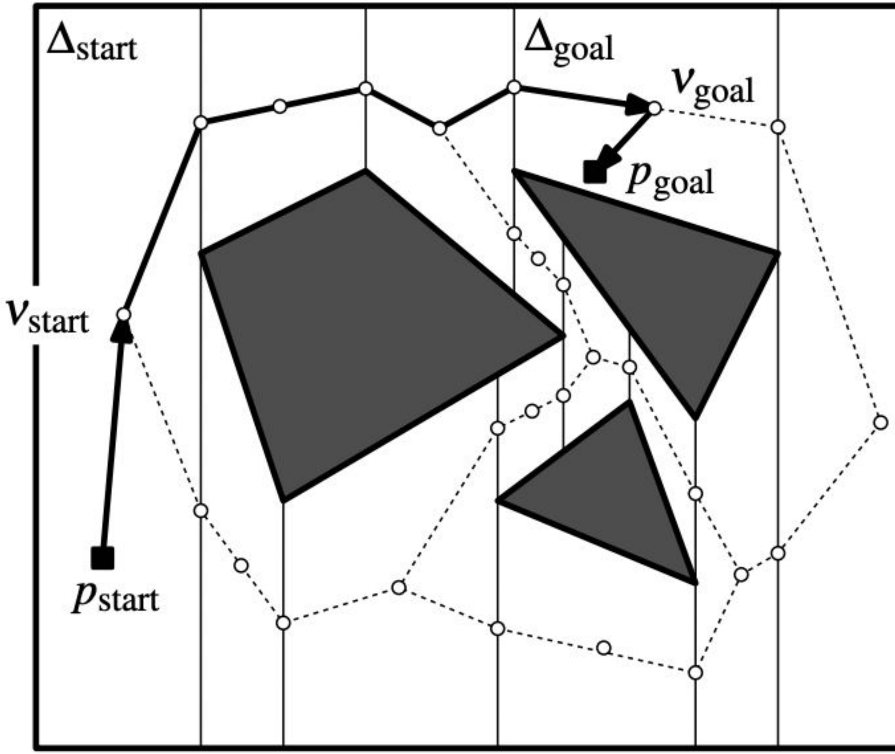



*Computational Geometry Algorithms and Applications*, de Berg, Cheong, van Kreveld and Overmars, Chapter 13

# Motion Planning Graph - Analysis

where *n* is the # of line segments for the obstacles + environment boundary

- Size of Trapezoid Map
  → *O(n)*
- Build Trapezoid Map
  → *O(n log n)*
  *Randomized incremental construction*
- Locate start/end trapezoid
  → *O(log n)*
- Breadth first search
  → *O(n)*
  *Finds a path, not the shortest path*




*Computational Geometry Algorithms and Applications*,
de Berg, Cheong, van Kreveld and Overmars, Chapter 13

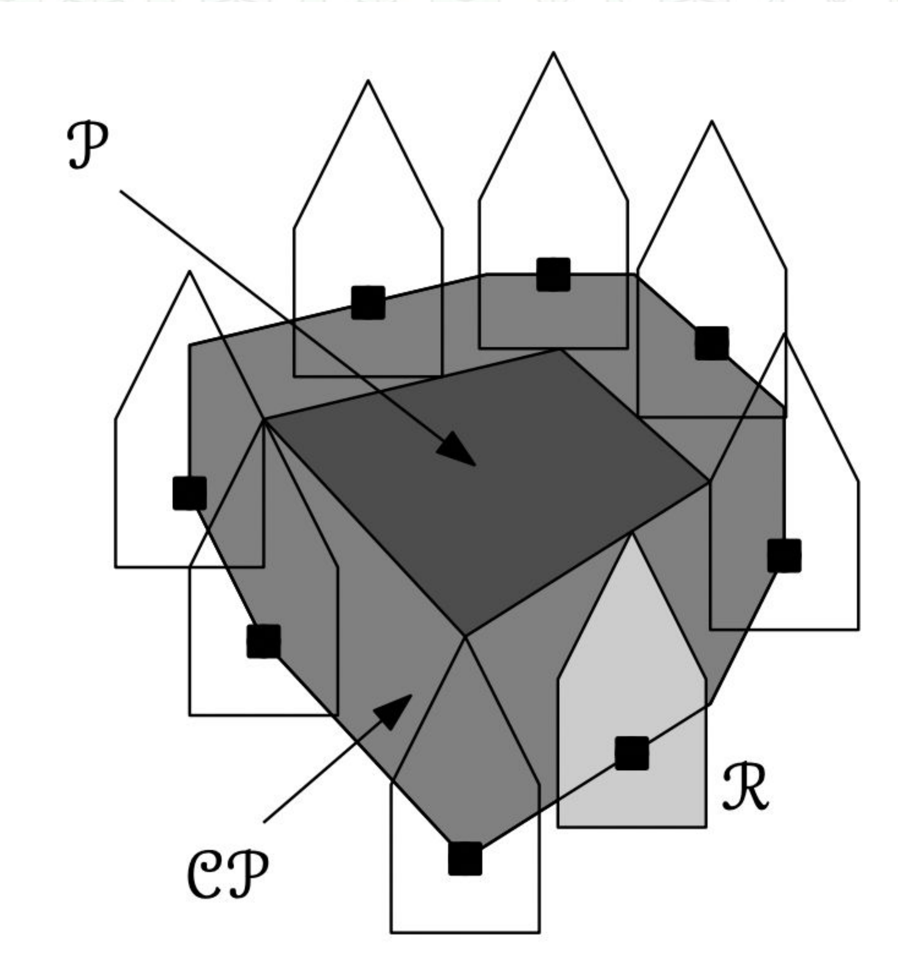
# Outline for Today

- Last Time: Bezier Curves, Polyline Simplification, Clothoid Sketches
- Motivation: Robot Motion Planning
- Previous Lecture: Voronoi Diagram of Segments for Motion Planning
- Degrees of Freedom & Configuration Space
- Trapezoid Map for Motion Planning
- Non-Point, Non-Rotating Robots & Minkowski Sums
- Related Operations: Convolution, Morphology, Accessible Surfaces
- Rotations & Higher Dimensional Configuration Space
- End of Term Schedule: Quiz 2, Sprouts, & Project Presentations

# Non-Point Robots

- Initially, let's ignore rotation
- How close can the robot get to the obstacle?

- The obstacle boundaries in configuration space will be expanded
- The origin / reference point of the robot is important

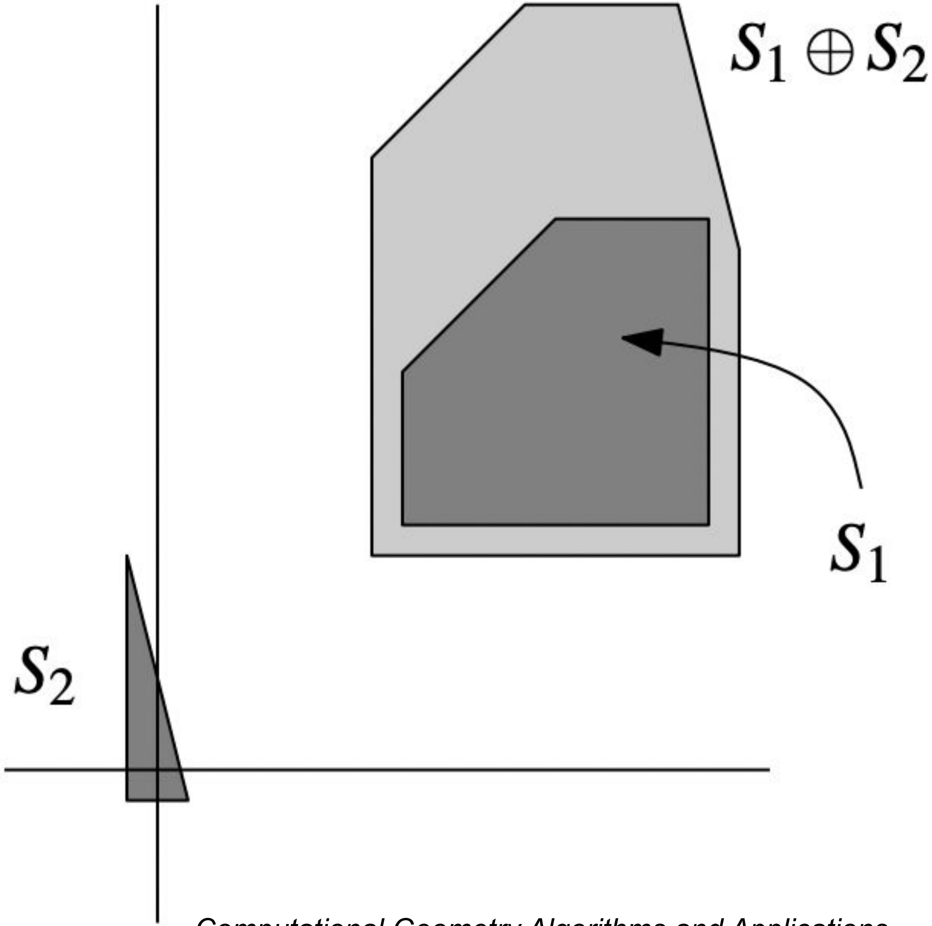



*Computational Geometry Algorithms and Applications*,
de Berg, Cheong, van Kreveld and Overmars, Chapter 13

# Minkowski Sum ⊕

Related to:

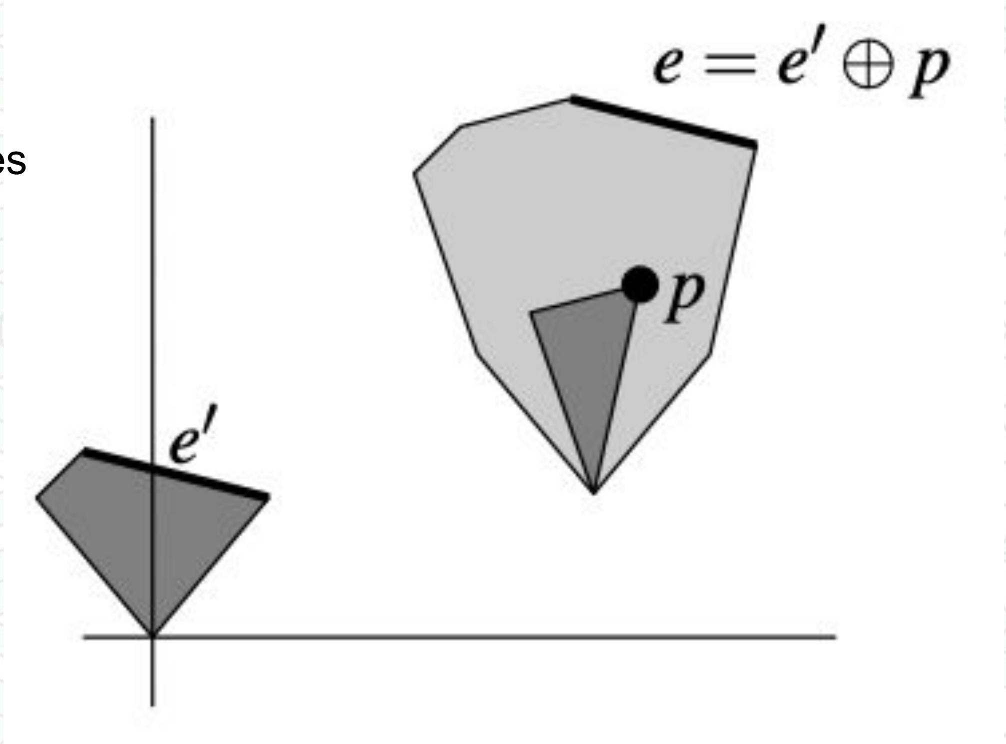
- Convolution
- Morphology
  - Dilation
  - Erosion
  - Opening
  - Closing
- Accessible Surfaces




*Computational Geometry Algorithms and Applications*,
de Berg, Cheong, van Kreveld and Overmars, Chapter 13

# Complexity of Minkowski Sum ⊕

- Given:
    - Convex robot with *n = 4* edges
    - Convex obstacle with *m = 3* edges
- How many edges does the resulting shape hape?

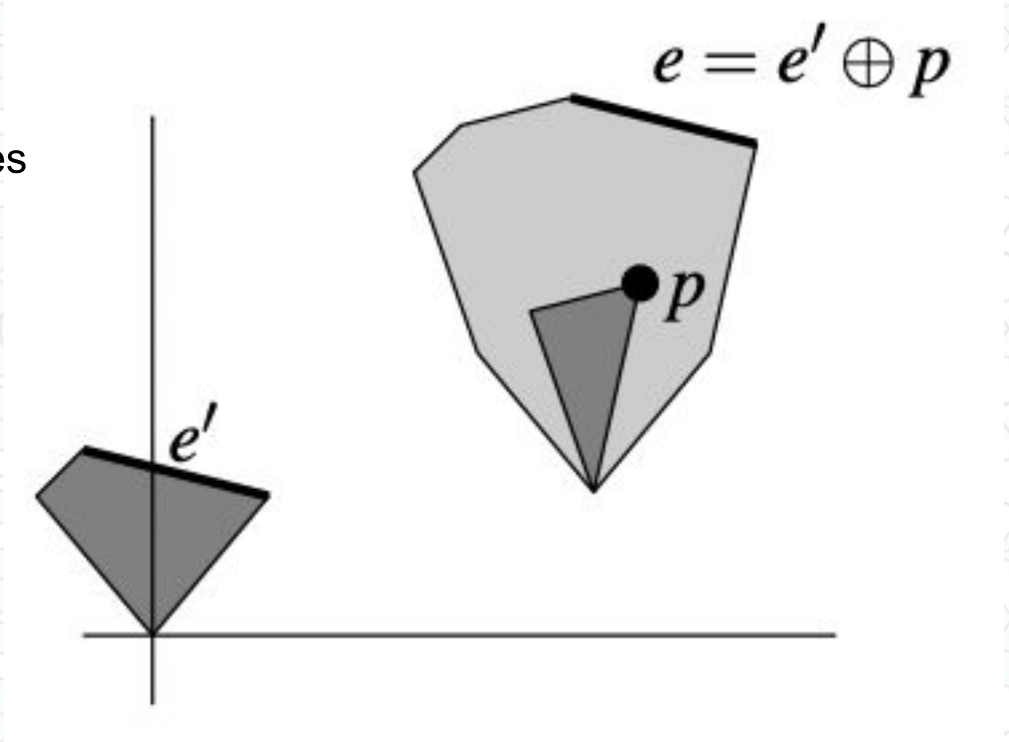



*Computational Geometry Algorithms and Applications*,
de Berg, Cheong, van Kreveld and Overmars, Chapter 13

# Complexity of Minkowski Sum ⊕

- Given:
  - Convex robot with *n = 4* edges
  - Convex obstacle with *m = 3* edges
- How many edges does the resulting shape hape?
  - ***n+m* = 7 edges**

- Each edge in the Minkowski sum is defined by an edge on one shape and a point on the other shape
- If two or more edges of the robot and obstacle are parallel, it will have fewer than n+m edges

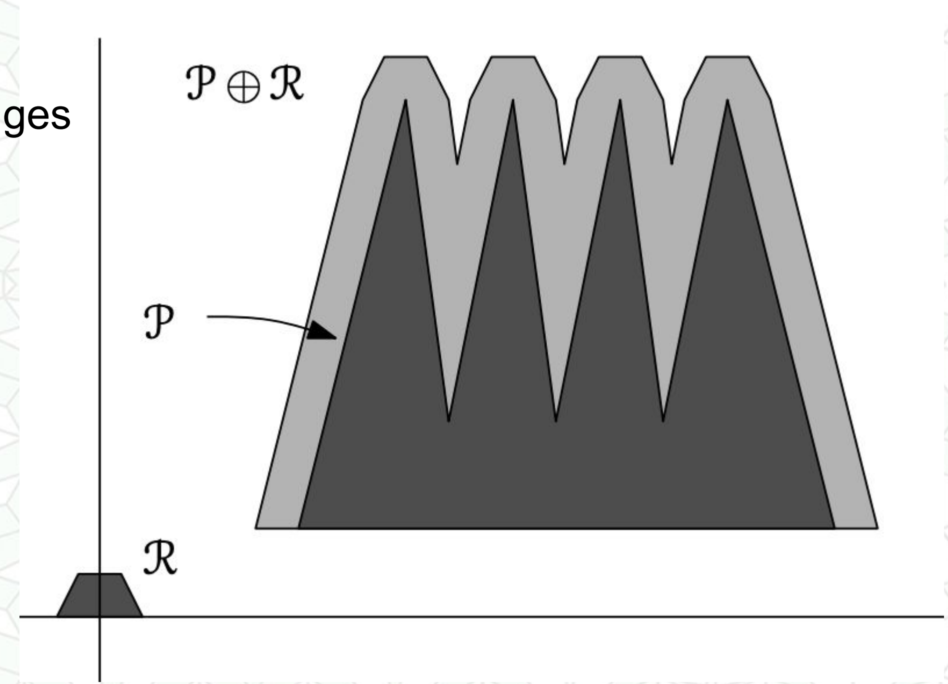



*Computational Geometry Algorithms and Applications*,
de Berg, Cheong, van Kreveld and Overmars, Chapter 13

# Complexity of Minkowski Sum ⊕

- Given:
  - Convex robot with *n* edges
  - **Non-convex obstacle** with *m* edges
- How many edges does the resulting shape hape?
  - ***O(nm)*** **edges**

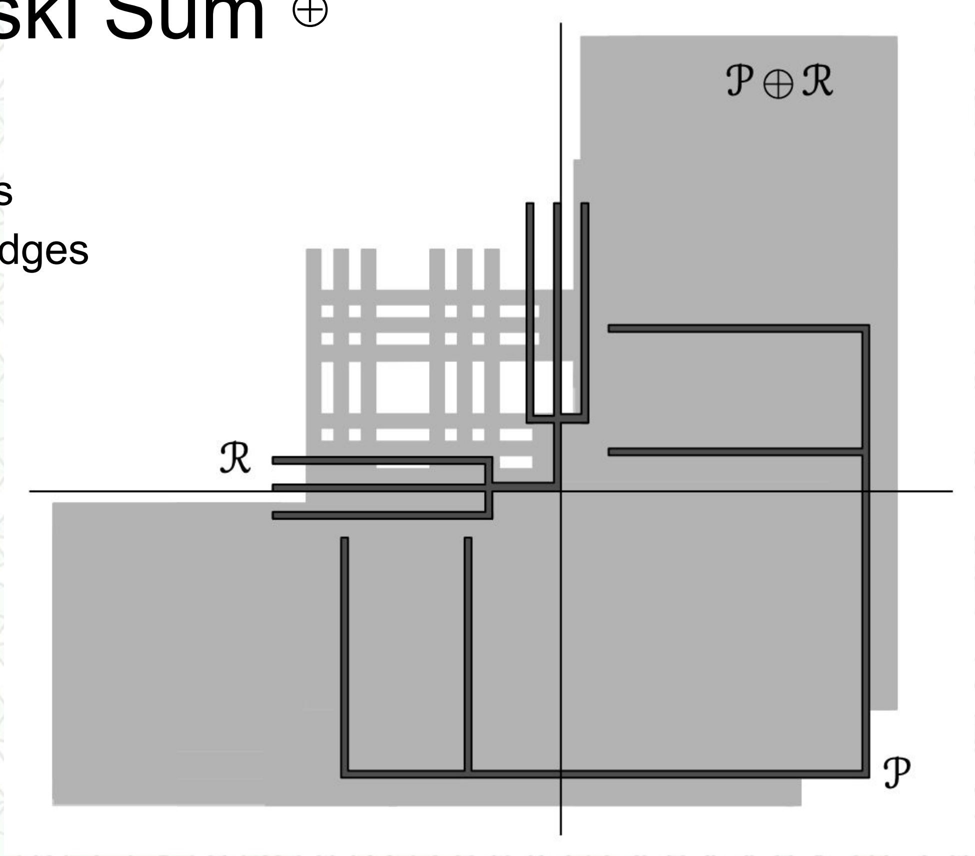
- Why? How to compute?
  - Triangulate the obstacle into m+2 triangles
  - Compute the minkowski sum of robot with each triangle
  - Combine via union operation




*Computational Geometry Algorithms and Applications*,
de Berg, Cheong, van Kreveld and Overmars, Chapter 13

# Complexity of Minkowski Sum ⊕

- Given:
  - **Non-convex robot** with $n$ edges
  - **Non-convex obstacle** with $m$ edges
- How many edges does the resulting shape hape?
  - ***$O(n^2m^2)$ edges***

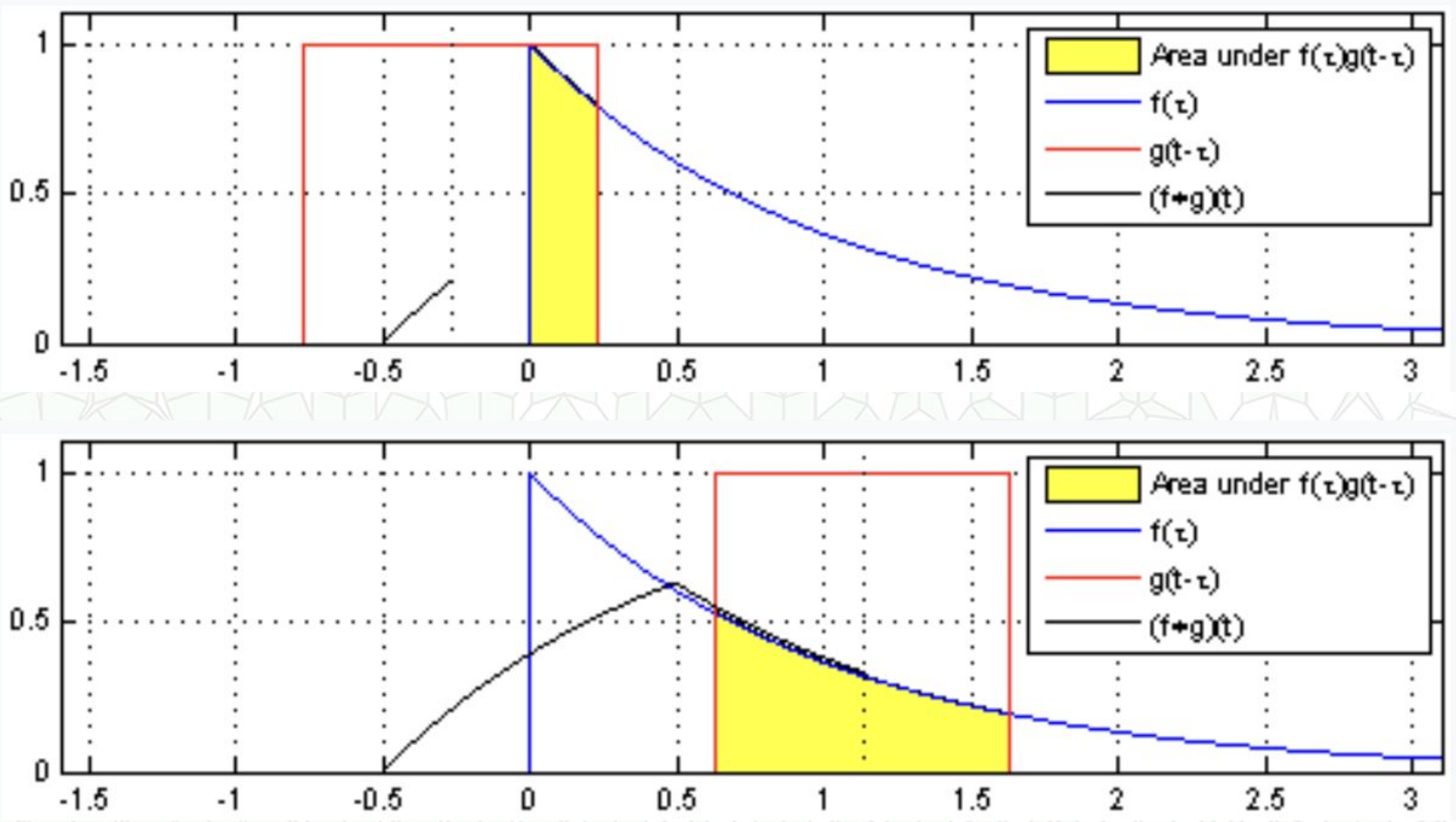



*Computational Geometry Algorithms and Applications*,
de Berg, Cheong, van Kreveld and Overmars, Chapter 13

# Outline for Today

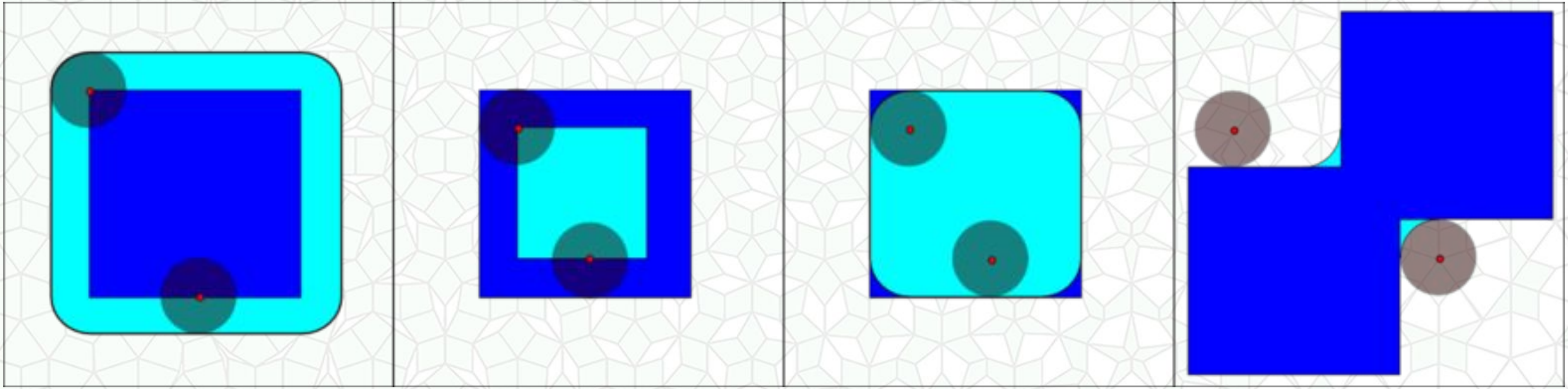
- Last Time: Bezier Curves, Polyline Simplification, Clothoid Sketches
- Motivation: Robot Motion Planning
- Previous Lecture: Voronoi Diagram of Segments for Motion Planning
- Degrees of Freedom & Configuration Space
- Trapezoid Map for Motion Planning
- Non-Point, Non-Rotating Robots & Minkowski Sums
- Related Operations: Convolution, Morphology, Accessible Surfaces
- Rotations & Higher Dimensional Configuration Space
- End of Term Schedule: Quiz 2, Sprouts, & Project Presentations

# Convolution ✻ "Flip & Slide" from Signals & Systems



# Morphology for Computer Vision

- For Noise Removal and other Image Processing Tasks



Dilation

Erosion

Opening
(Erosion first,
then Dilation)

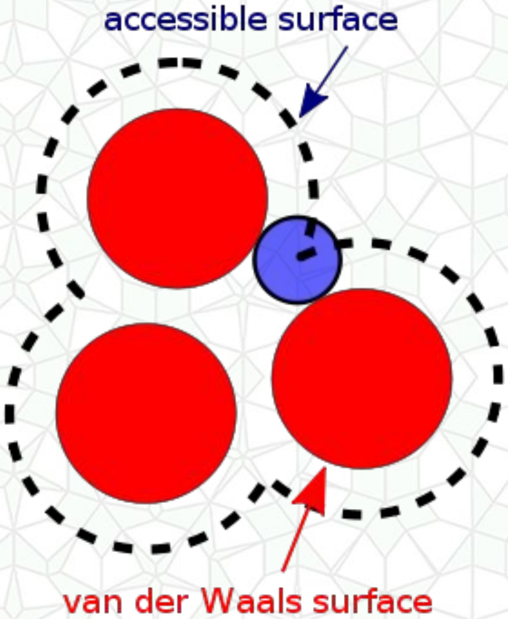
Closing
(Dilation first,
then Erosion)

https://en.wikipedia.org/wiki/Mathematical_morphology

# Weathering, Accessible Surfaces

https://en.wikipedia.org/wiki/Accessible_surface_area

- Simulate water flow & removal of surface dirt



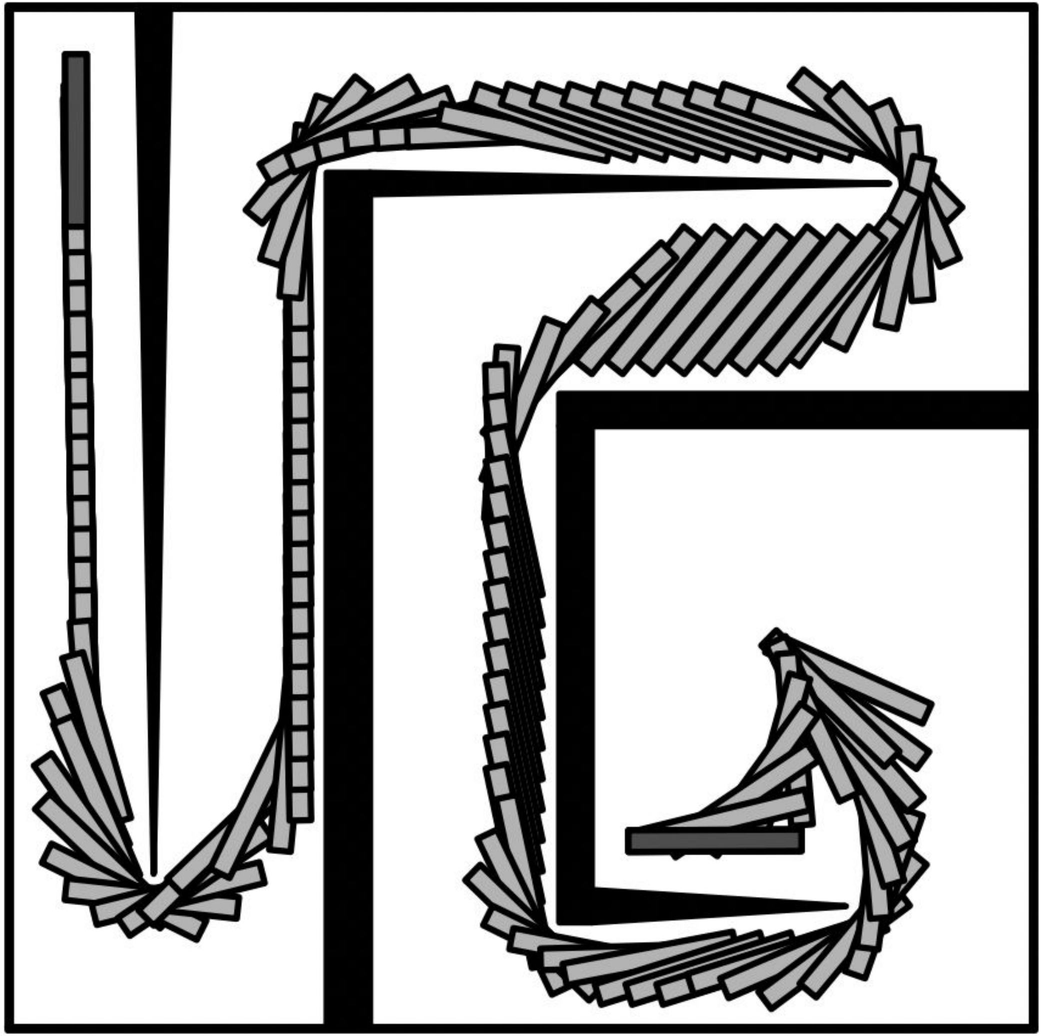

"Flow and Changes in Appearance" Dorsey, Pedersen, & Hanrahan, SIGGRAPH 1996

# Outline for Today

- Last Time: Bezier Curves, Polyline Simplification, Clothoid Sketches
- Motivation: Robot Motion Planning
- Previous Lecture: Voronoi Diagram of Segments for Motion Planning
- Degrees of Freedom & Configuration Space
- Trapezoid Map for Motion Planning
- Non-Point, Non-Rotating Robots & Minkowski Sums
- Related Operations: Convolution, Morphology, Accessible Surfaces
- Rotations & Higher Dimensional Configuration Space
- End of Term Schedule: Quiz 2, Sprouts, & Project Presentations

# What about Rotating Robots?

- Rotation may be necessary to complete the task

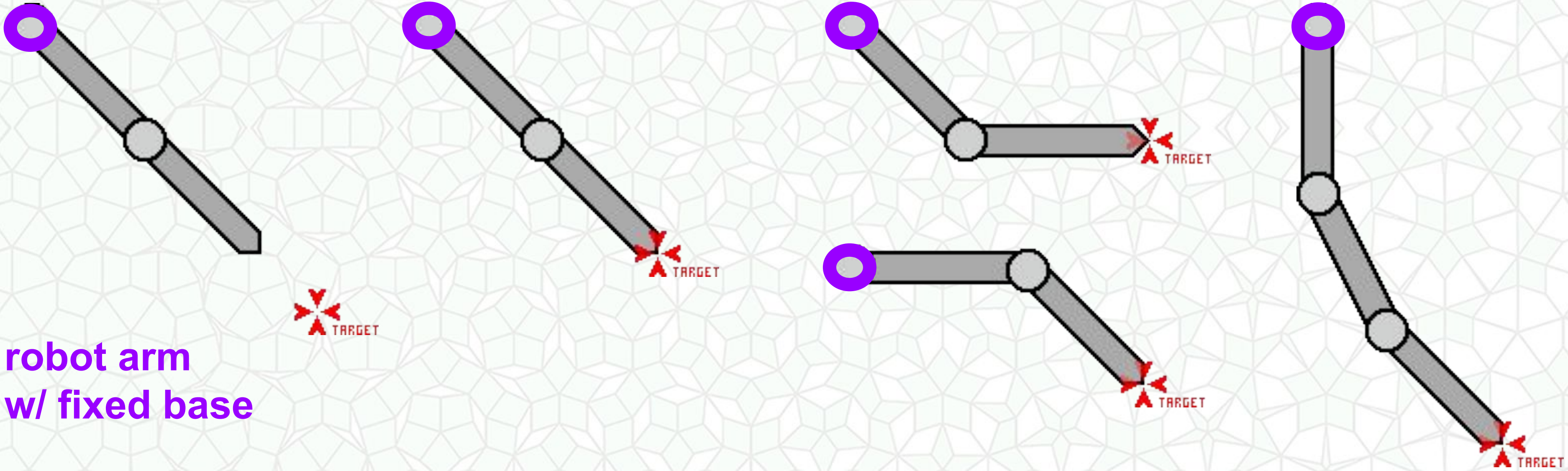


*Computational Geometry Algorithms and Applications*, de Berg, Cheong, van Kreveld and Overmars, Chapter 13

# Searching Configuration Space

Application:
Robot Motion Planning

*No solutions* | *One solution* | *Two solutions (2D)* | *Many solutions*

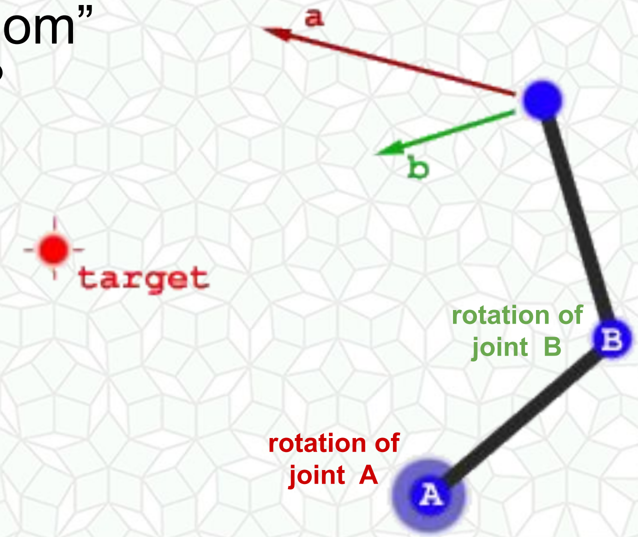



"The good-looking textured light-sourced bouncy fun smart and stretchy page"
Hugo Elias, http://freespace.virgin.net/hugo.elias/ (stale link)

# Searching Configuration Space

- What are the unknowns?

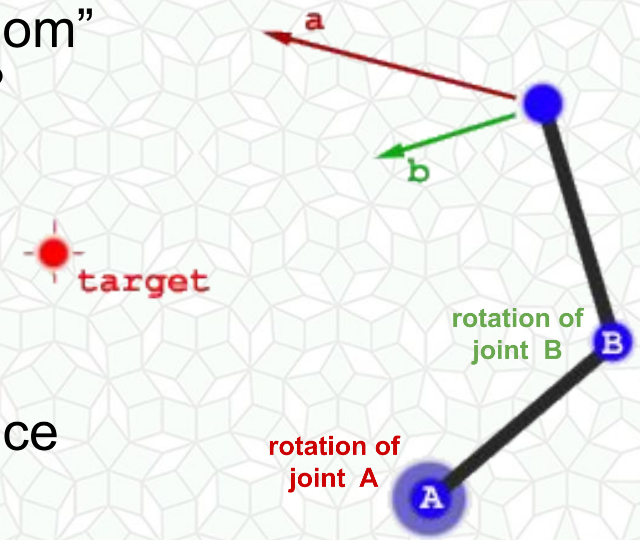
- What are the "degrees of freedom" of our robot arm?



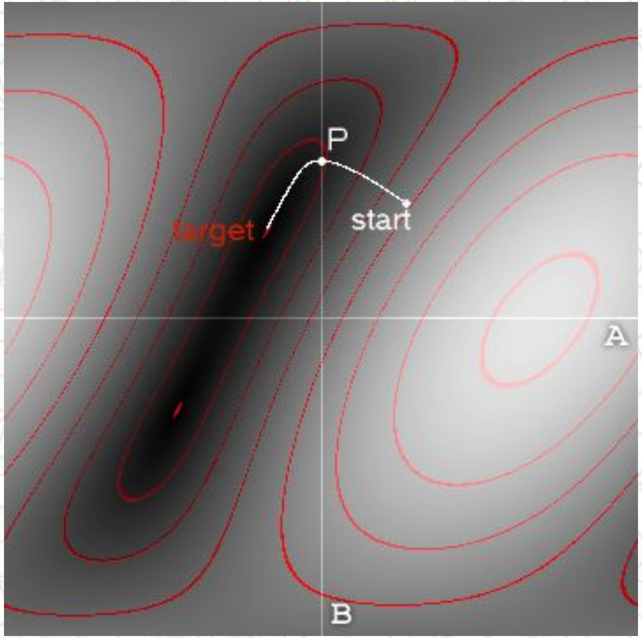

"The good-looking textured light-sourced bouncy fun smart and stretchy page"
Hugo Elias, http://freespace.virgin.net/hugo.elias/ (stale link)

# Searching Configuration Space

- What are the unknowns?
- What are the "degrees of freedom" of our robot arm?
- More degrees of freedom = higher dimensional configuration space




*configuration space shaded by distance to target (darker means closer to goal)*

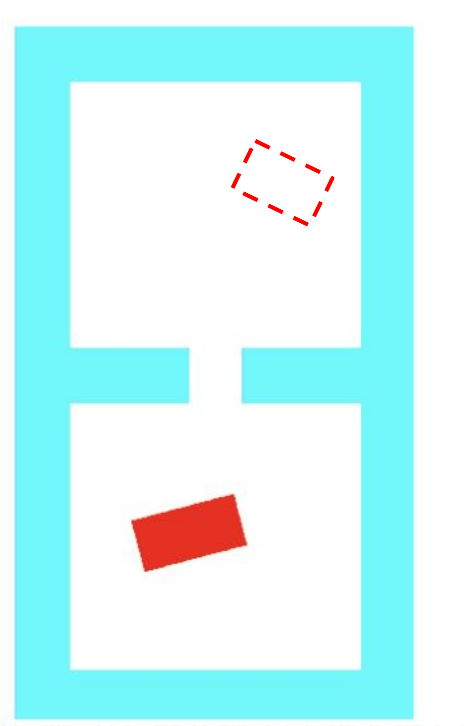



"The good-looking textured light-sourced bouncy fun smart and stretchy page"
Hugo Elias, http://freespace.virgin.net/hugo.elias/ (stale link)

# Searching Configuration Space

"C-Space Tunnel Discovery for Puzzle Path Planning", Zhang, Belfer, Kry, & Voucha, SIGGRAPH 2020.

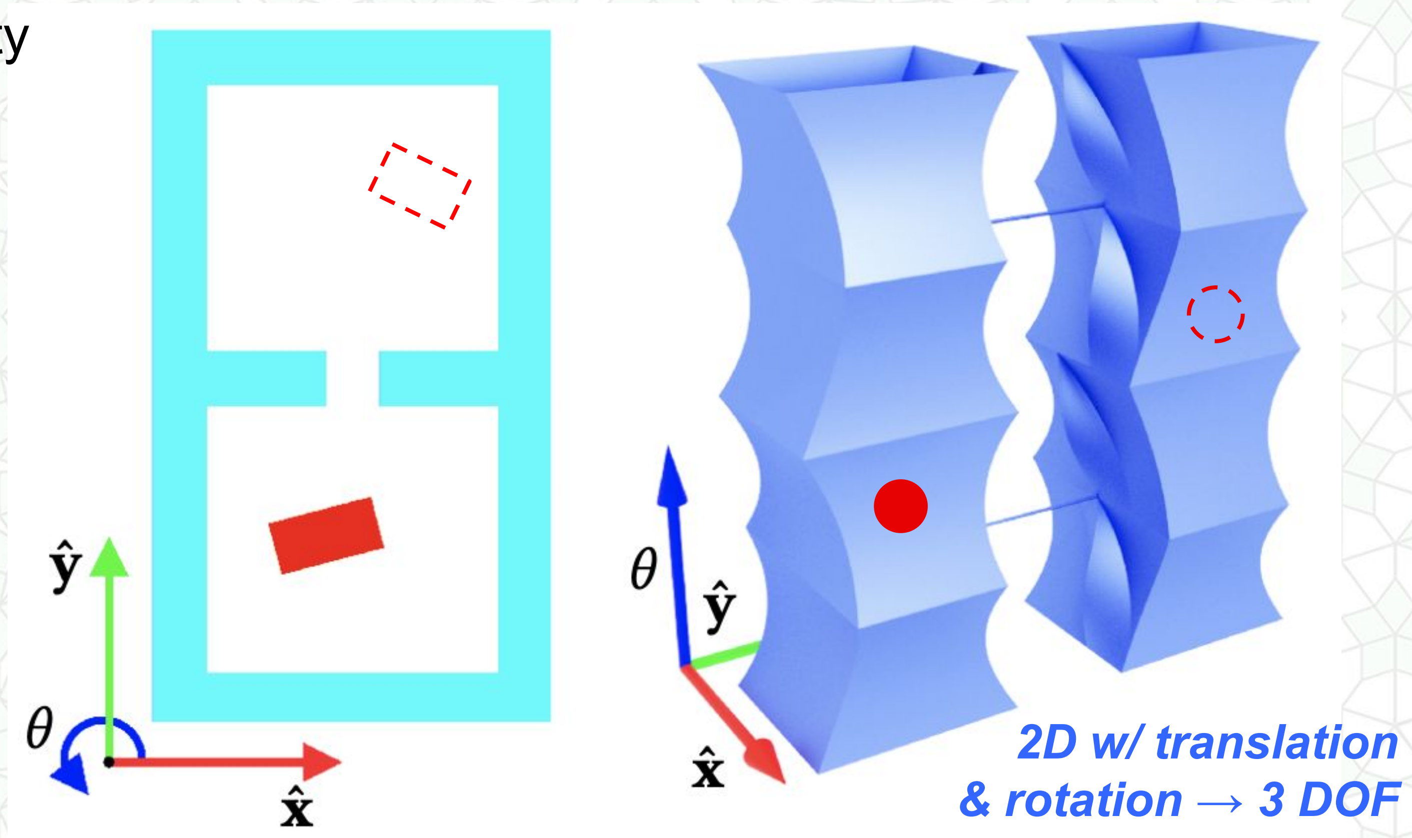
- How many DOF?
- How do we find a solution?
- *Or show none exists?*



# Searching Configuration Space

"C-Space Tunnel Discovery for Puzzle Path Planning", Zhang, Belfer, Kry, & Voucha, SIGGRAPH 2020.

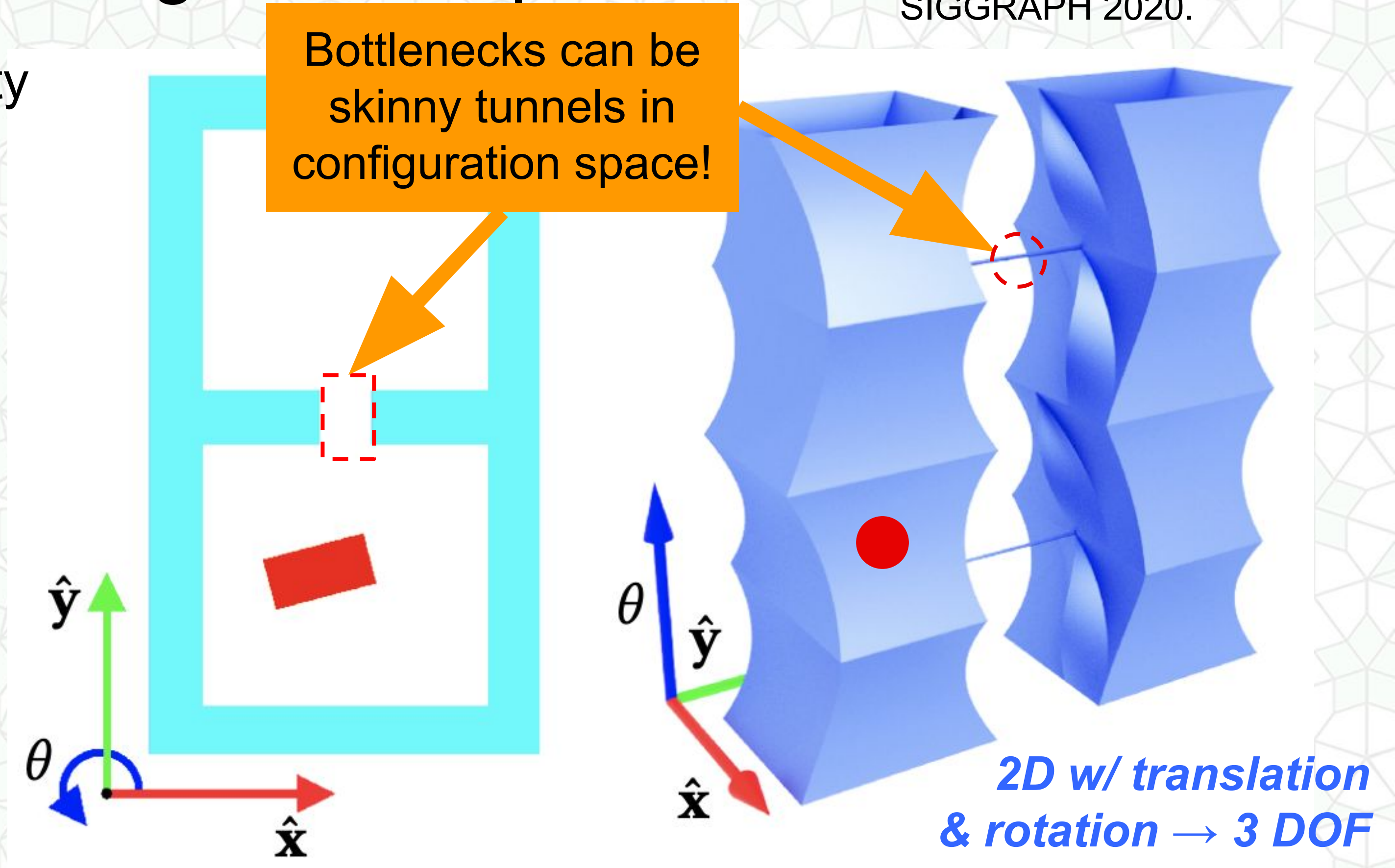
- Dimensionality becomes infeasible to construct & exhaustively search
- Discretized and/or Randomized search is necessary



# Searching Configuration Space

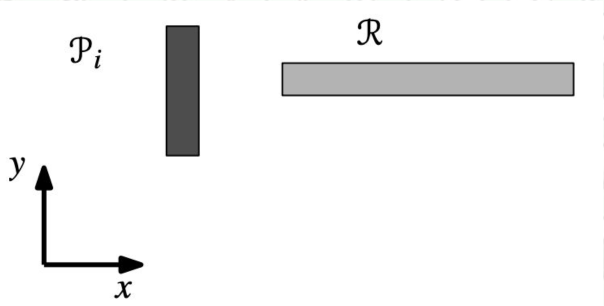
"C-Space Tunnel Discovery for Puzzle Path Planning",
Zhang, Belfer, Kry, & Voucha,
SIGGRAPH 2020.

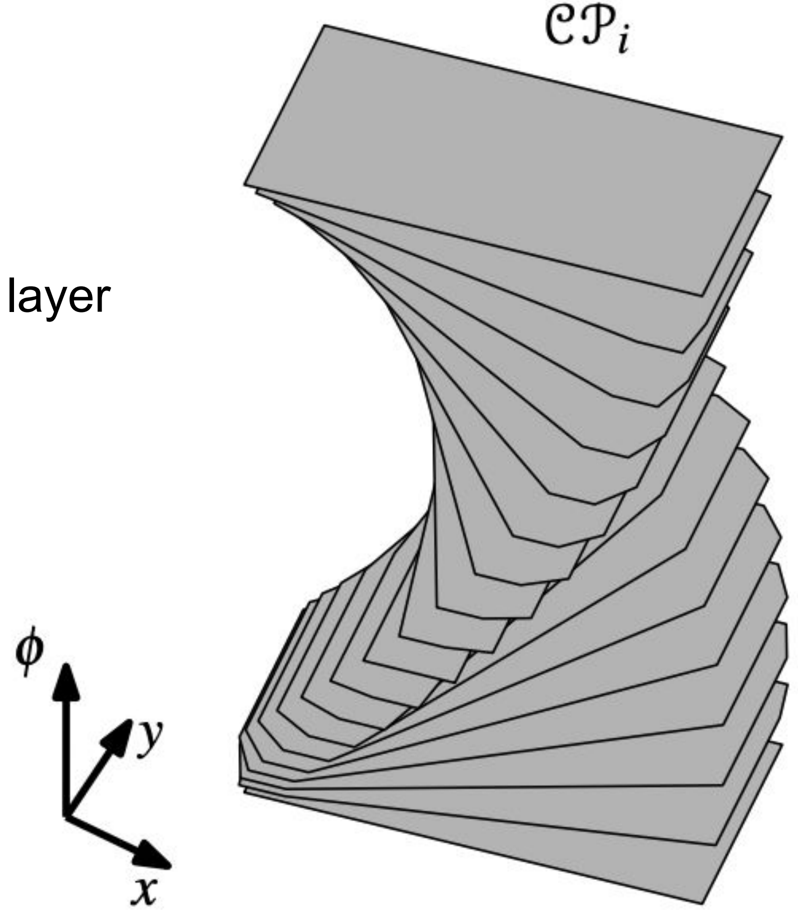
- Dimensionality becomes infeasible to construct & exhaustively search
- Discretized and/or Randomized search is necessary



# Discretized Search

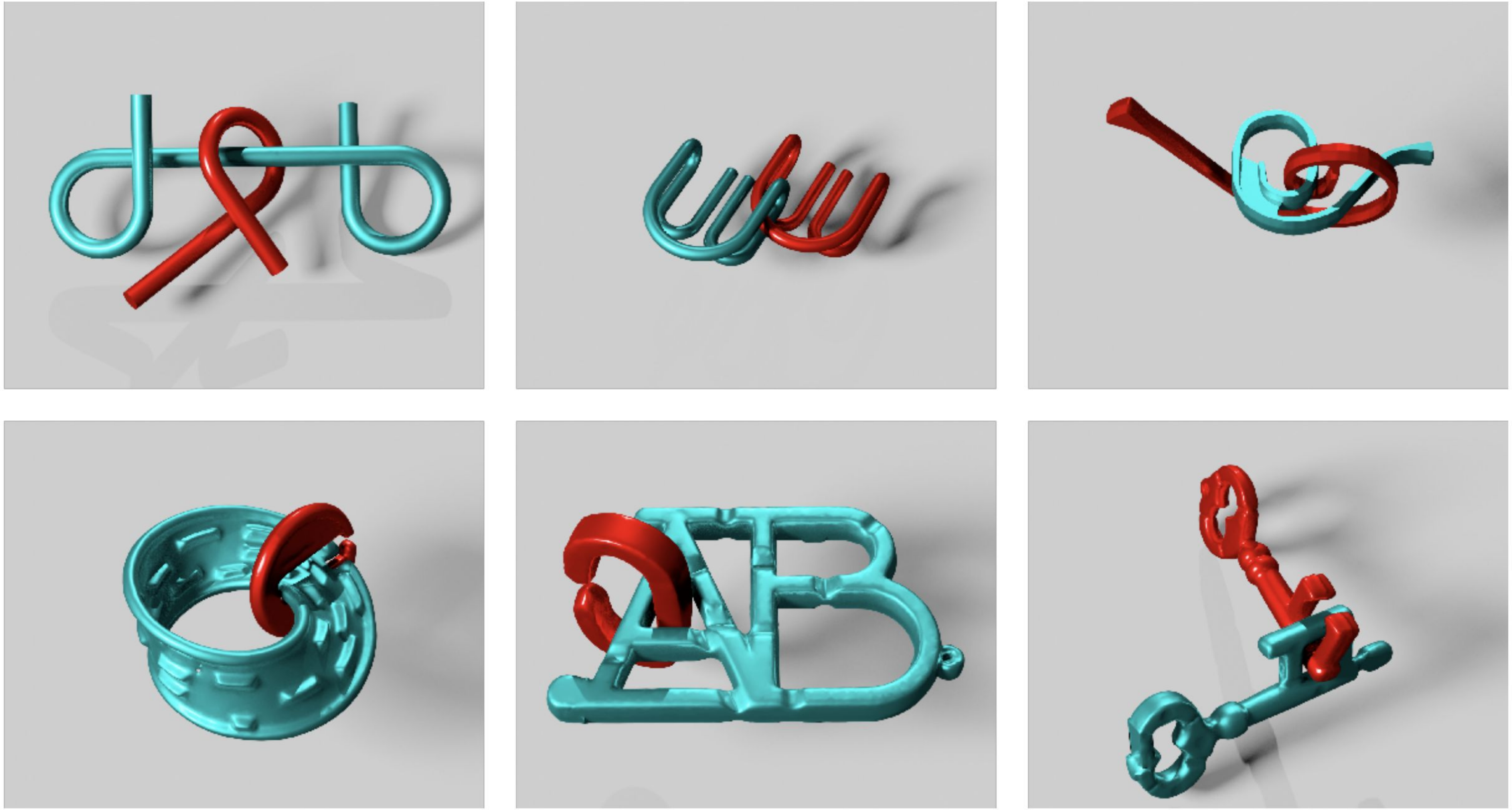
- Discretize problem into fixed step sizes in rotation
- Search a single 2D configuration space layer
- Step up or down a layer
- Because error has been introduced, add extra padding around obstacles



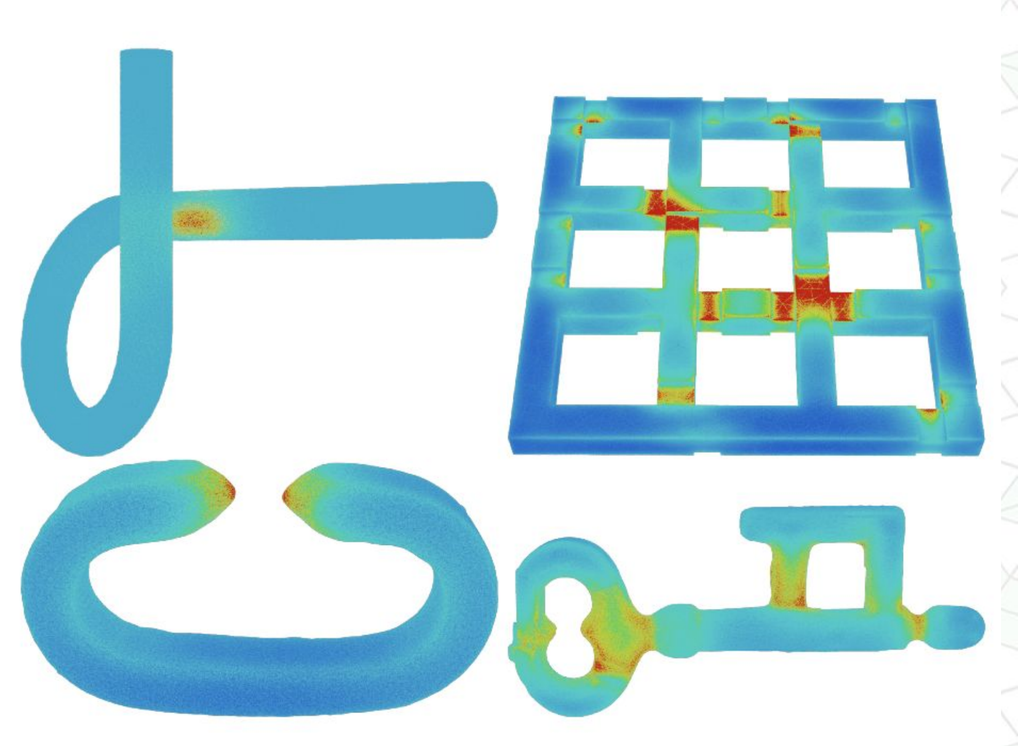

*Computational Geometry Algorithms and Applications*,
de Berg, Cheong, van Kreveld and Overmars, Chapter 13

"C-Space Tunnel Discovery for Puzzle Path Planning",
Zhang, Belfer, Kry, & Voucha, SIGGRAPH 2020.



"C-Space Tunnel Discovery for Puzzle Path Planning", Zhang, Belfer, Kry, & Voucha, SIGGRAPH 2020.

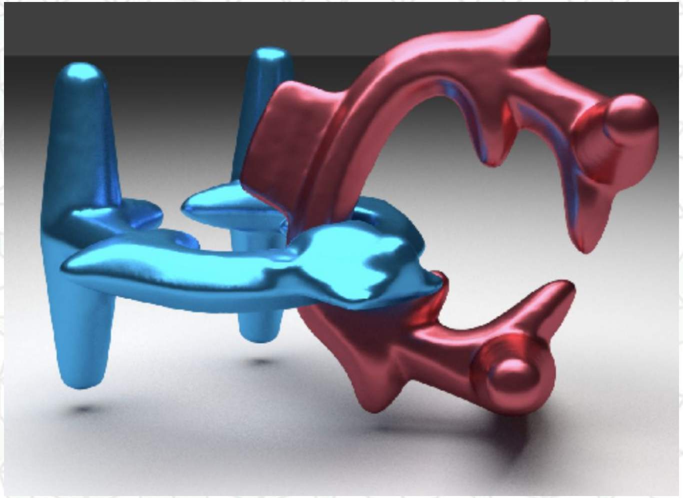
- Limited to puzzles with 2 rigid bodies
  - One is fixed
  - The other moves with translation + rotation = 6 DOF
- **6D search space** is reduced by pre-processing geometry to identify potential geometric pinch points / bottlenecks



"C-Space Tunnel Discovery for Puzzle Path Planning", Zhang, Belfer, Kry, & Voucha, SIGGRAPH 2020.

- Cannot feasibly solve with **3 or more rigid pieces (12+ DOF!)** e.g., *Hanayama Enigma Puzzle*

- Or puzzles with less obvious geometric pinch points / bottlenecks e.g., *Hanayama Elk Puzzle*



"Design of Part Feeding and Assembly Processes with Dynamics", Song,Trinkle, Kumar, & Pang, MEAM 2004.

# Robotics: Automatic Part Sorting & Orienting

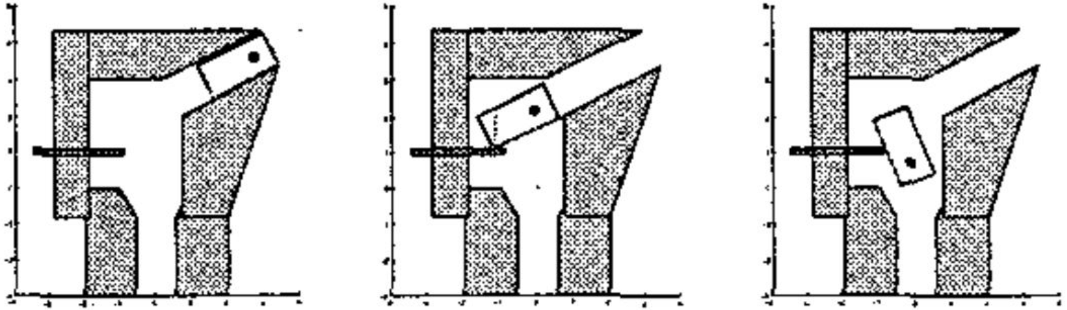


Fig. 9. Peg able to pass through the device with optimal design parameters with center of gravity starting on the right.

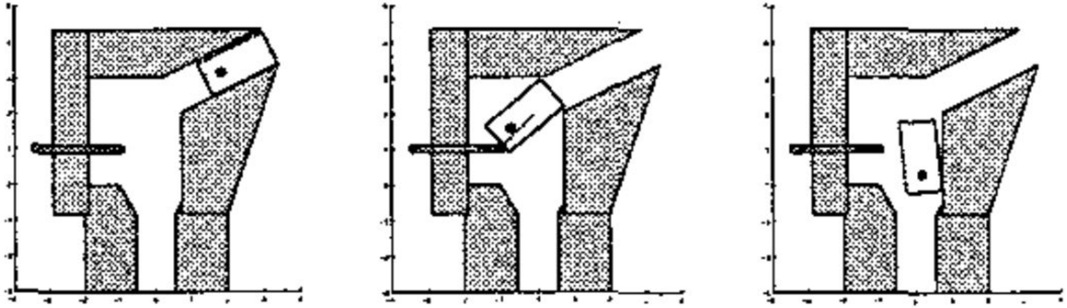


Fig. 10. Peg able to pass through the device with optimal design parameters with center of gravity starting on the left.

# Robotics: Automatic Part Sorting & Orienting

“Using Simulation for Planning and Design of Robotic Systems with Intermittent Contact”, Stephen Berard, RPI PhD 2009.

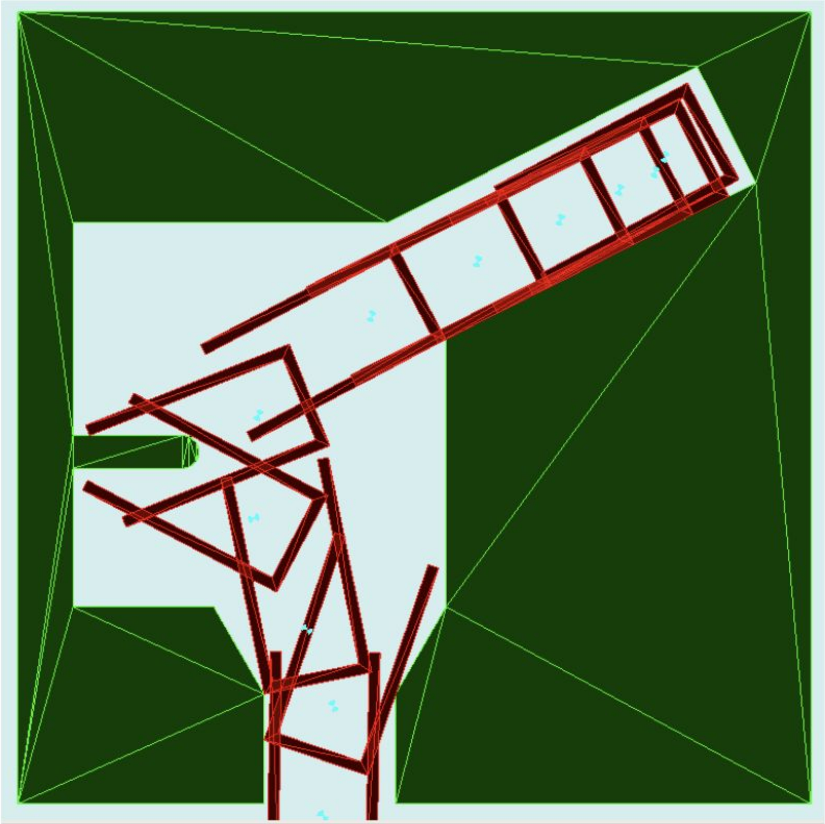
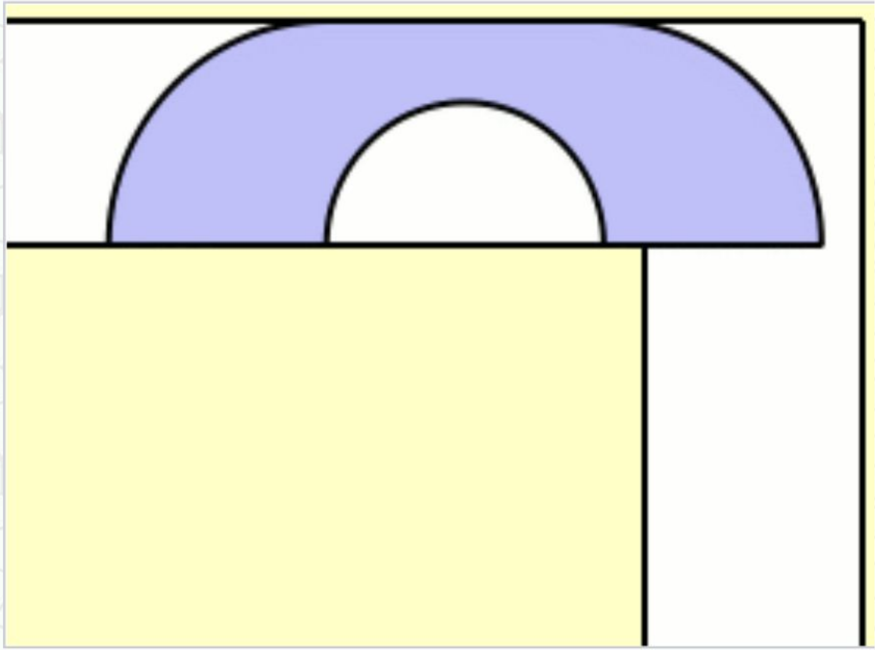
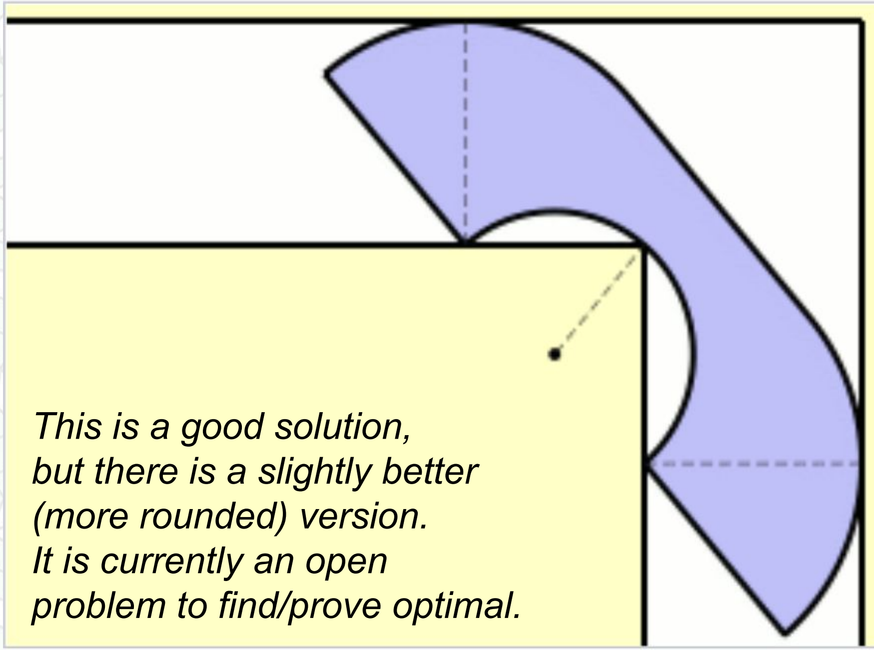


Figure 4.2: Snapshots of the gravity-fed part in the feeder.

# Moving Sofa Problem

- Find the largest rigid shape (by area) that can navigate a 90° corner






https://en.wikipedia.org/wiki/Moving_sofa_problem

# Outline for Today

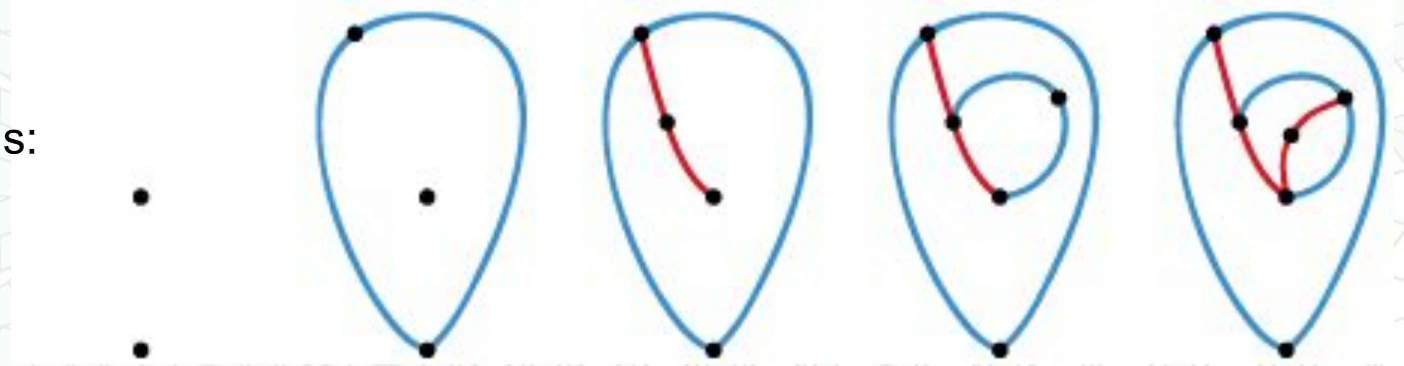
- Last Time: Bezier Curves, Polyline Simplification, Clothoid Sketches
- Motivation: Robot Motion Planning
- Previous Lecture: Voronoi Diagram of Segments for Motion Planning
- Degrees of Freedom & Configuration Space
- Trapezoid Map for Motion Planning
- Non-Point, Non-Rotating Robots & Minkowski Sums
- Related Operations: Convolution, Morphology, Accessible Surfaces
- Rotations & Higher Dimensional Configuration Space
- End of Term Schedule: Quiz 2, Sprouts, & Project Presentations

- Quiz 2 on Friday
    - Optional: You are allowed one double-sided page of notes
    - Quiz will be on paper, with some sketching
    - Like Quiz 1, you have the option to type written answers in a plaintext file on your laptop (but not to make general use of internet or textbook, etc.)

| | | | | |
|---|---|---|---|---|
| Nov 27, Final Project *Progress Post #2* due @ 11:59pm | Nov 28, Lecture 24: Robot Motion Planning<br>*Textbook Reading:*<br>• CGAA Chapter 13 | | | Dec 1, Quiz 2 |
| | Dec 5, Lecture 25: Sprouts & Brussels Sprouts | | Dec 7, Final Project *Written Report* due @ 11:59pm | Dec 8, Final Project Presentations |
| Dec 11-13, *Reading days* *No classes* | | | Dec 14-15, *Other RPI Final Exams* *(no Final Exam for Computational Geometry)* | |
| Dec 18-20, *Other RPI Final Exams* *(no Final Exam for Comptuational Geometry)* | | | | |

Paper and Pencil Games
http://www.papg.com/show?1TMQ

# Sprouts Game Rules



- Draw *n* spots
- Players take turns:
  - Draw a line joining two spots, or a single spot to itself.
  - The line must not cross another line or pass through another spot.
  - Draw a spot on the new line.
  - No more than three lines can emerge from any spot.
- Normal Winning Condition: Winner is last person to make a move
- *Misère Winning Condition: Winner is first person who cannot make a move*