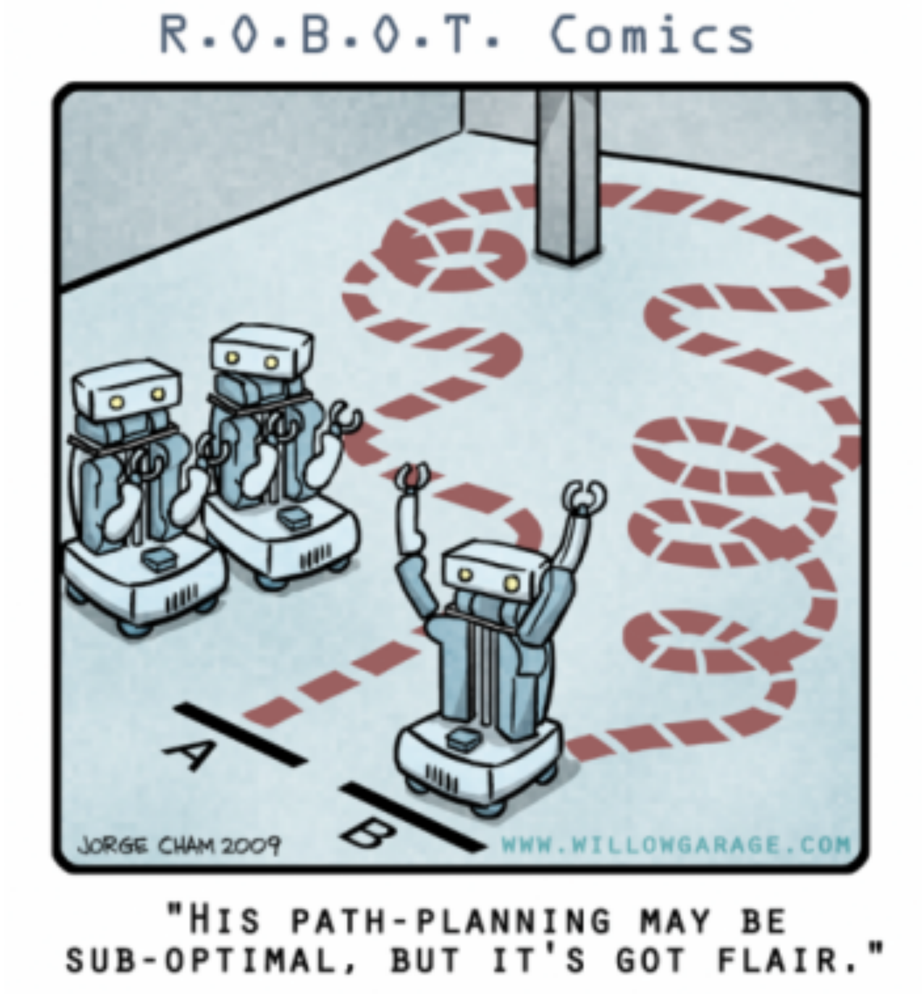
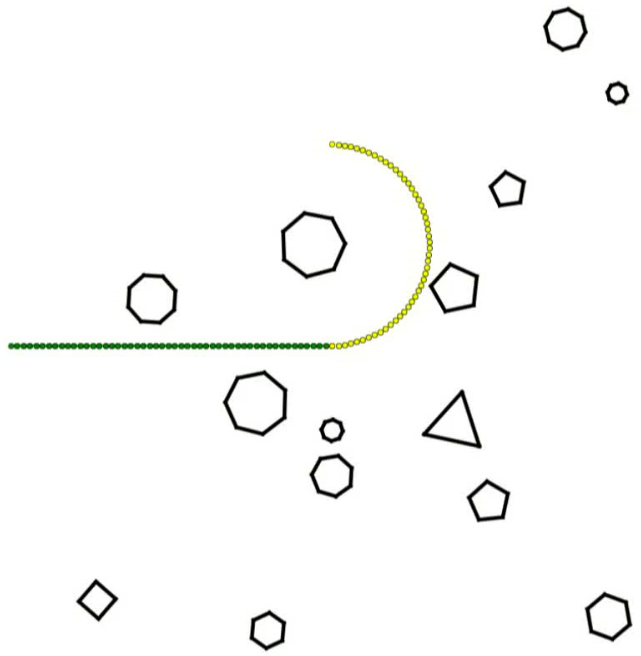
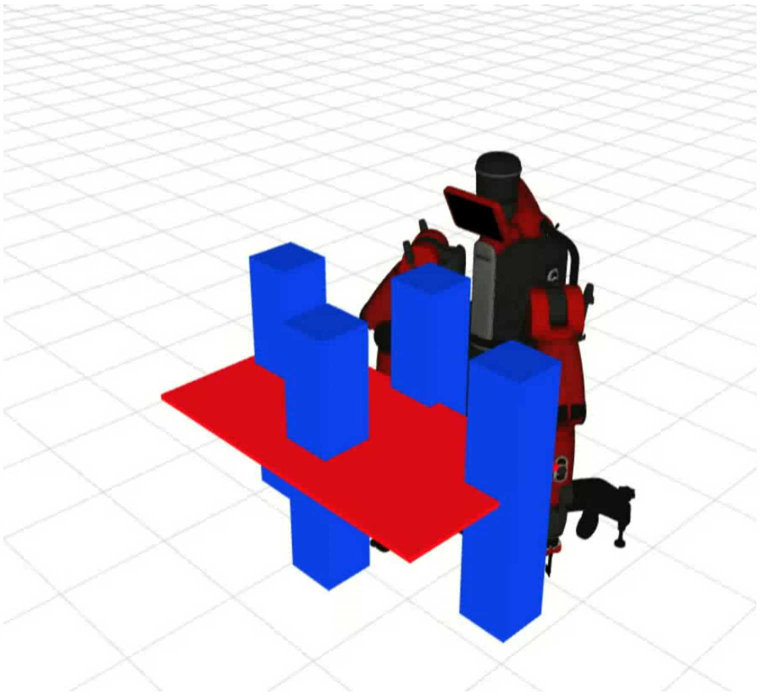
Background: Motion (or Path) Planning



UNIVERSITY OF SOUTH CAROLINA

Examples





Background: Configuration Space - C

- Degree of Freedom (DoF): Each of a number of independently variable factors affecting the range of states in which a system may exist.
- Configuration: A single complete specification of the control parameters of a system
- Configuration Space: The space containing all the possible combinations of the control parameters

Understanding the C – 1D point robot

Workspace:

DoFs :

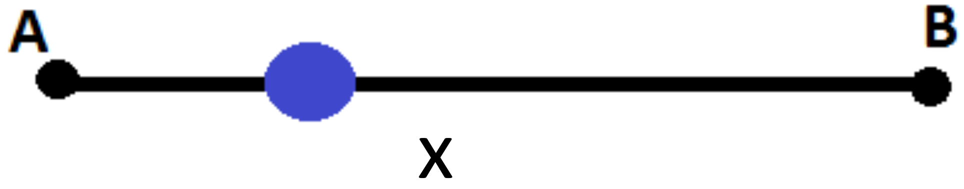
Configuration Space:

C =

Understanding the C – 1D point robot

Workspace:

DoFs : {x}



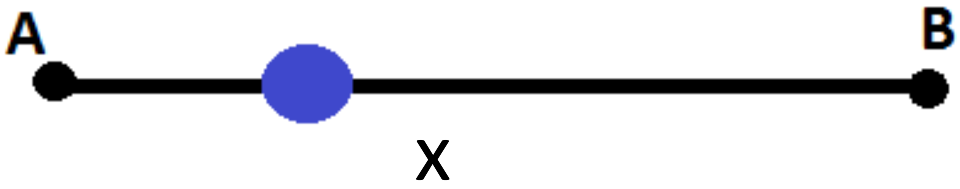
Configuration Space:

C =

Understanding the C – 1D point robot

Workspace:

DoFs : {x}



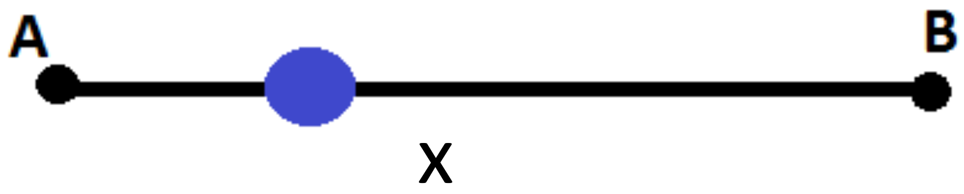
Configuration Space:

C = [A, B]

Understanding the C – 1D point robot

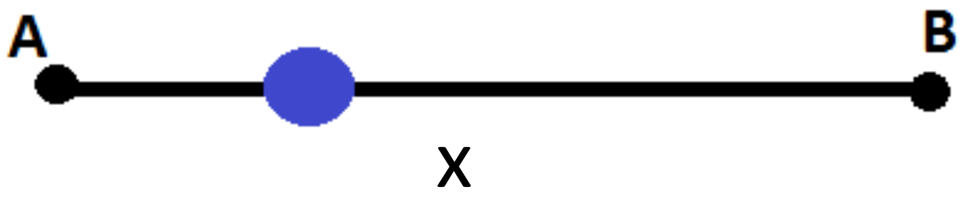
Workspace:

DoFs : {x}



Configuration Space:

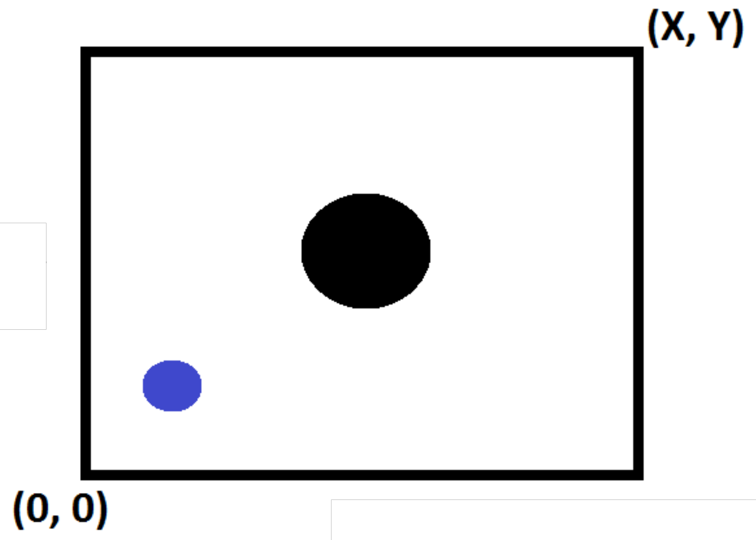
C = [A, B]



Understanding the C – 2D point robot

Workspace

- DoFs :



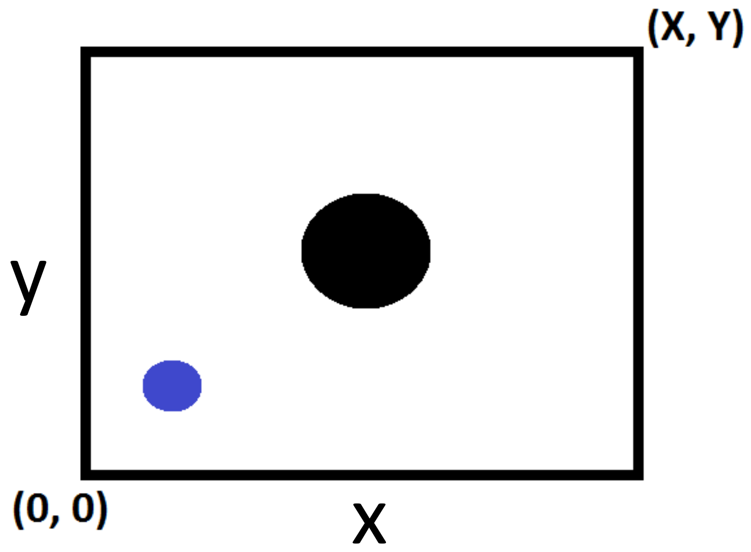
Configuration Space:

- C =

Understanding the C – 2D point robot

Workspace

- DoFs : {x,y}



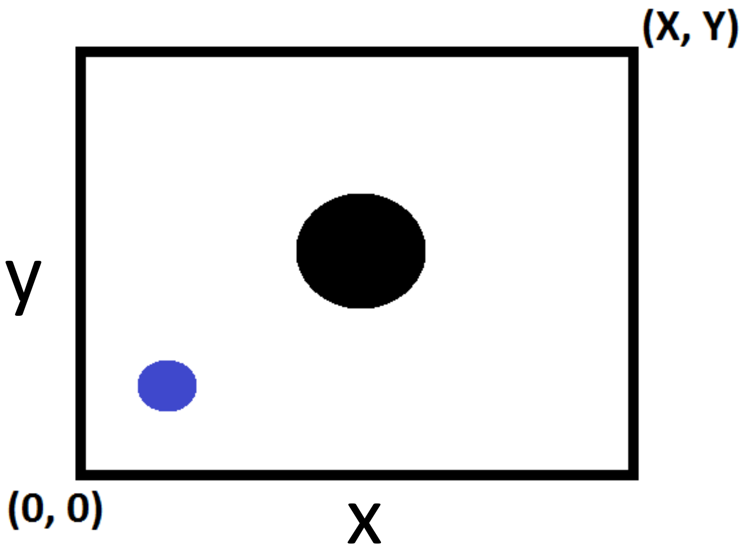
Configuration Space:

- C =

Understanding the C – 2D point robot

Workspace

- DoFs : {x,y}



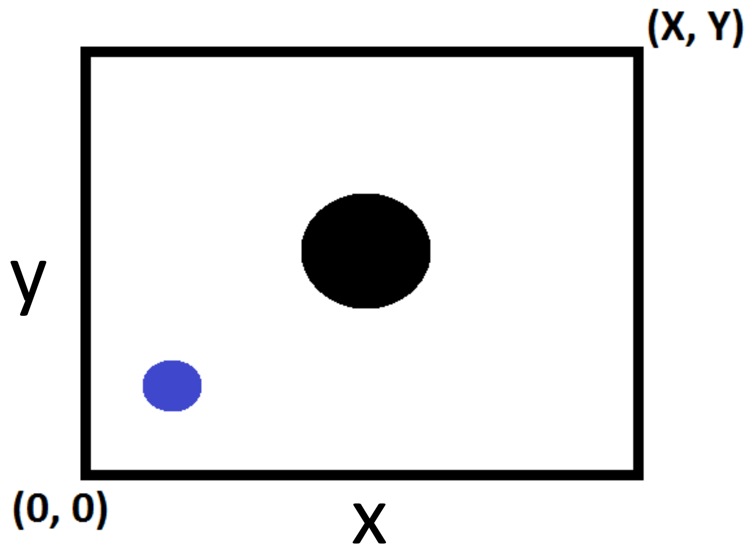
Configuration Space:

- C = [0, X]×[0,Y]

Understanding the C – 2D point robot

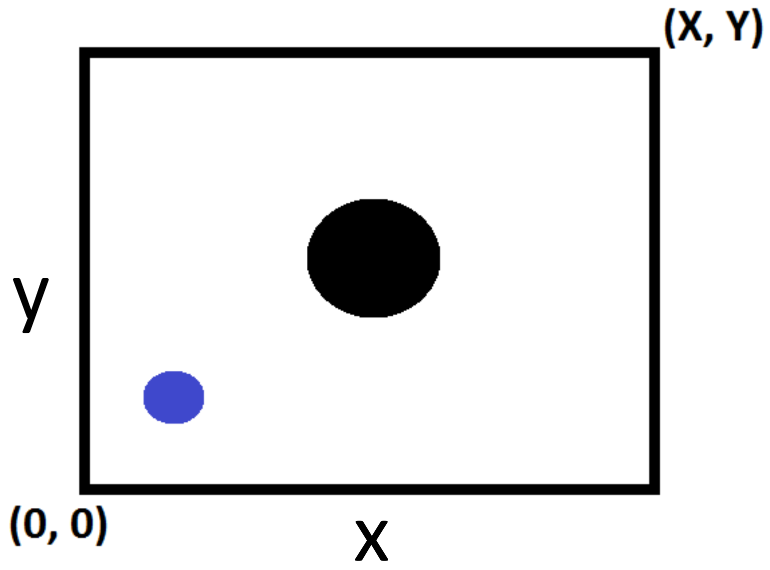
Workspace

- DoFs : {x,y}



Configuration Space:

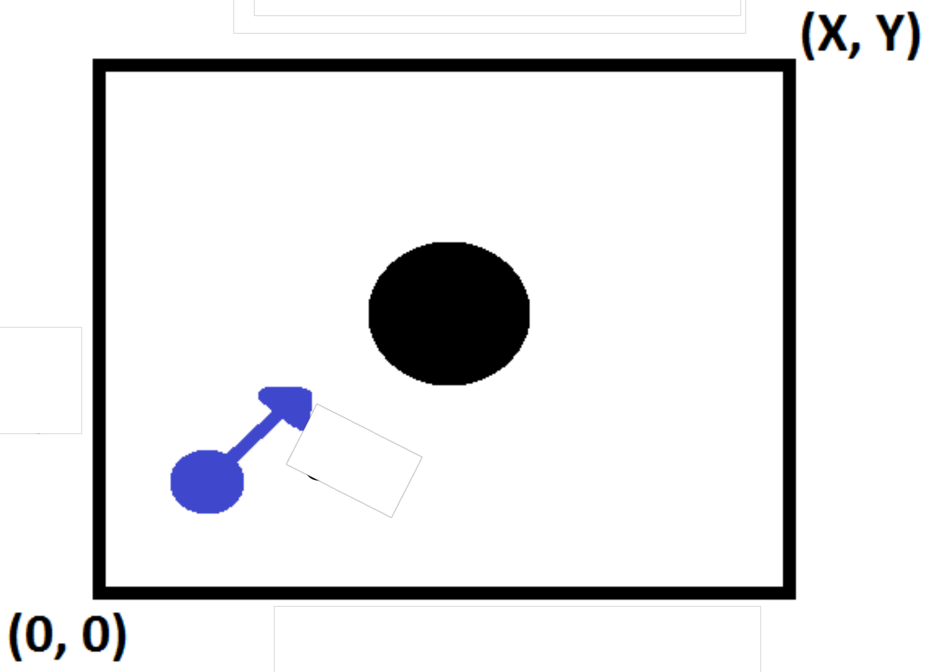
- C = [0, X]×[0,Y]



Understanding the C – 2D point robot extended

Workspace:

- DoFs :



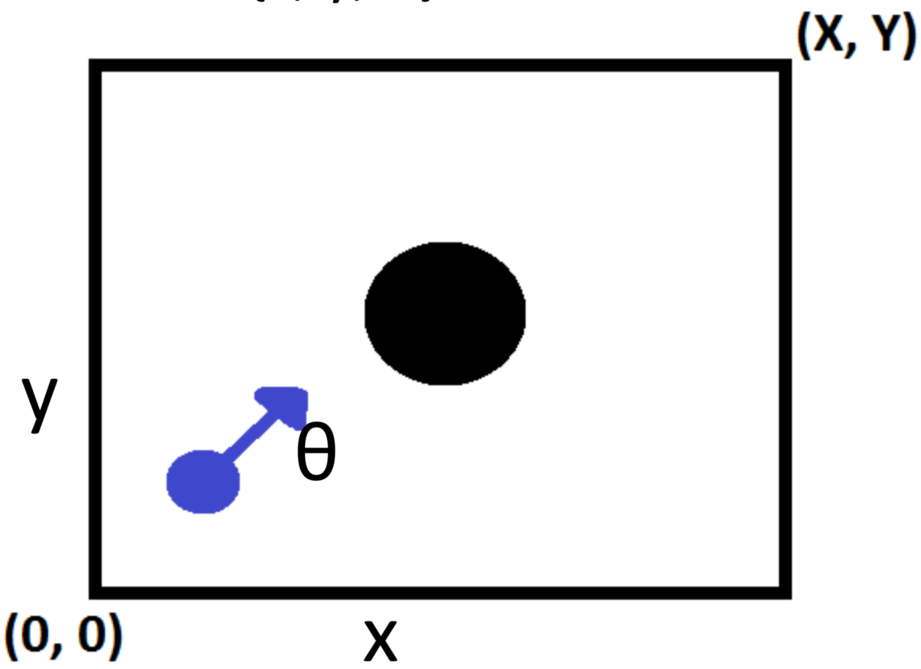
Configuration Space:

- C =

Understanding the C – 2D point robot extended

Workspace:

- DoFs : {x, y, θ}



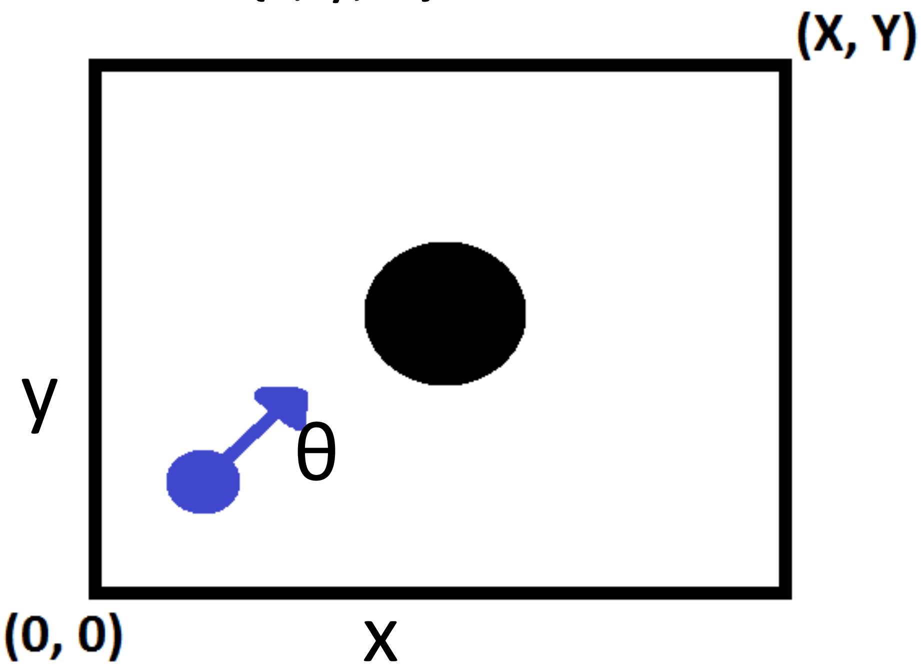
Configuration Space:

- C =

Understanding the C – 2D point robot extended

Workspace:

- DoFs : {x, y, θ}



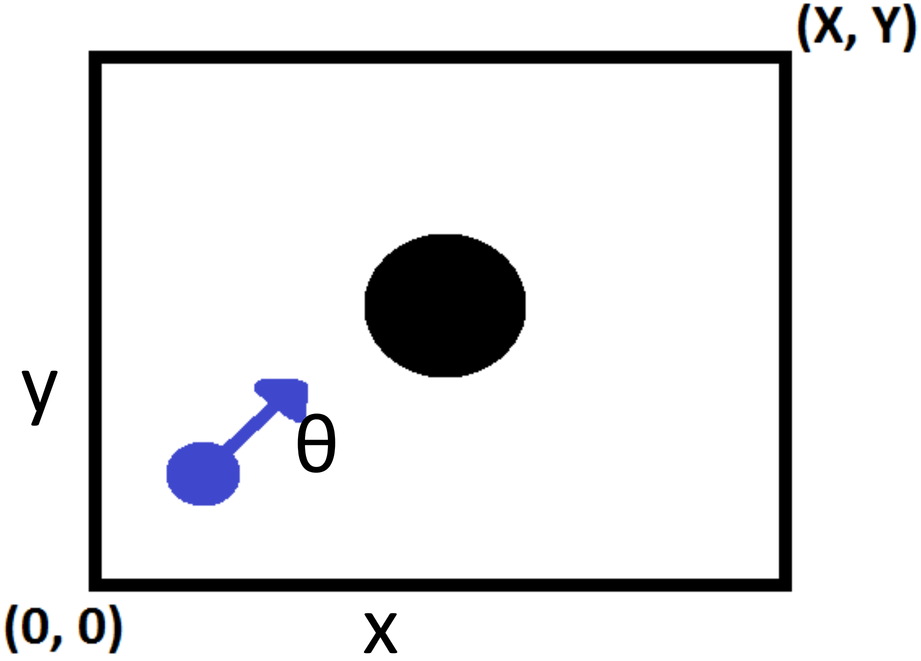
Configuration Space:

- $C = [0, X] \times [0, Y] \times [-\pi, \pi]$

Understanding the C – 2D point robot extended

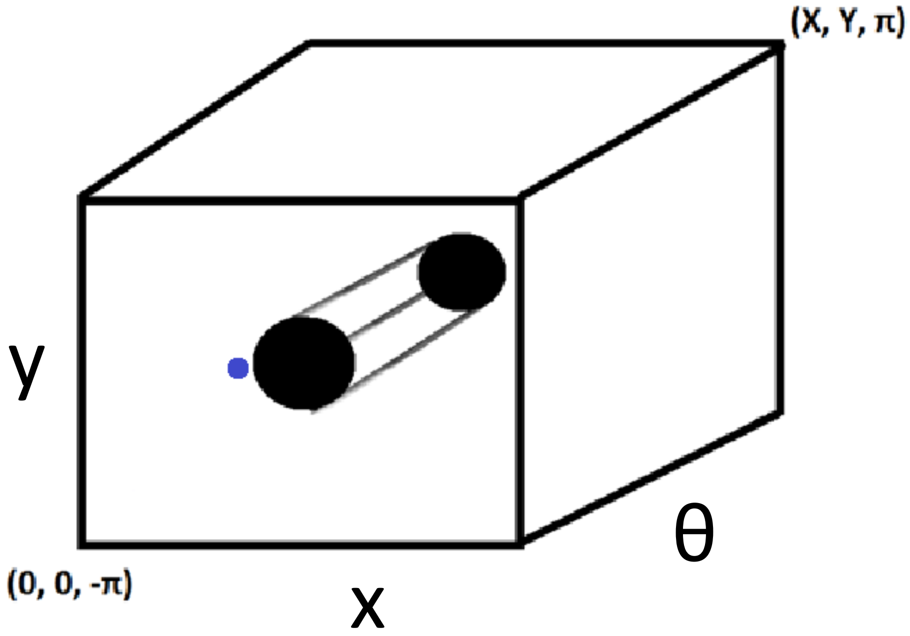
Workspace:

- DoFs : {x, y, θ}



Configuration Space:

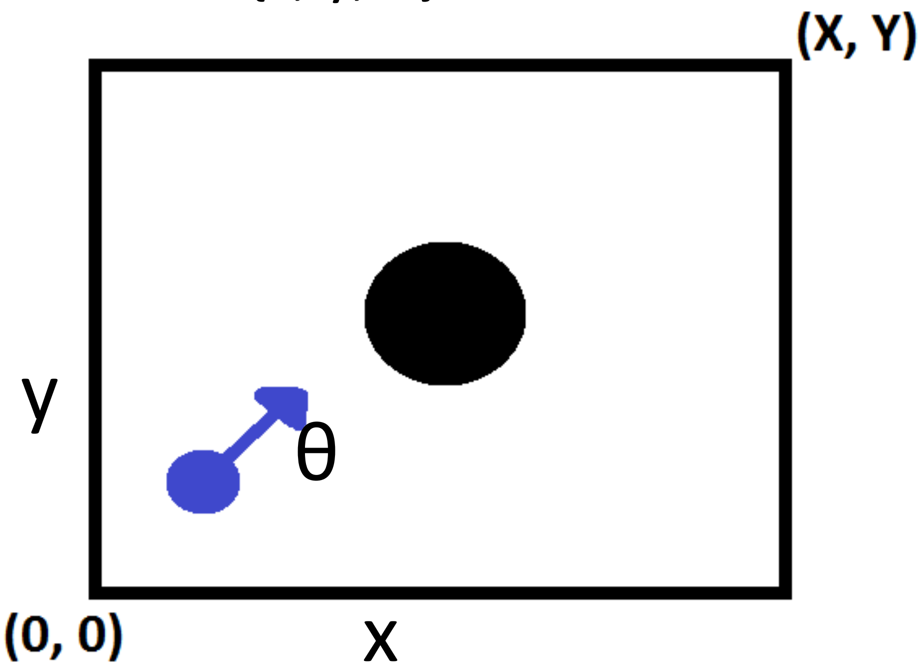
- C = [0, X]×[0, Y]×[-π, π]



Understanding the C – 2D point robot extended

Workspace:

- DoFs : {x, y, θ}



Configuration Space:

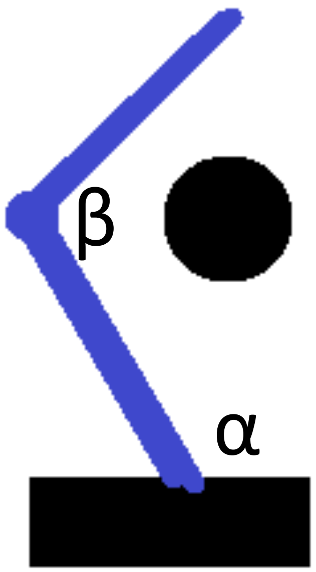
- C = [0, X]×[0, Y]×[-π, π]

In reality, a "wheel" like space!!!

Understanding the C – 2D 2R manipulator

Workspace:

- DoFs :



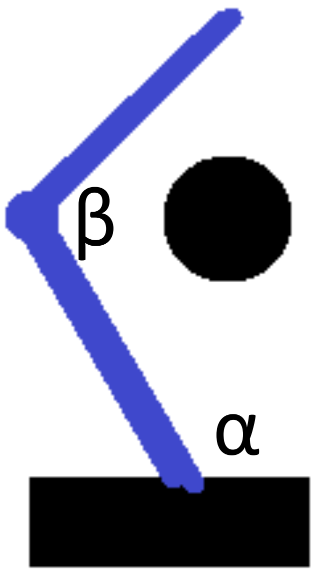
Configuration Space:

- C=

Understanding the C – 2D 2R manipulator

Workspace:

- DoFs : {α, β}



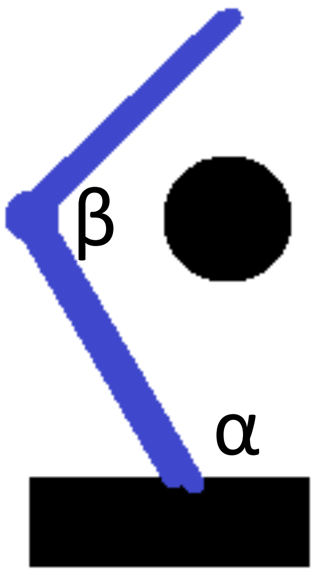
Configuration Space:

- C=

Understanding the C – 2D 2R manipulator

Workspace:

- DoFs : {α, β}



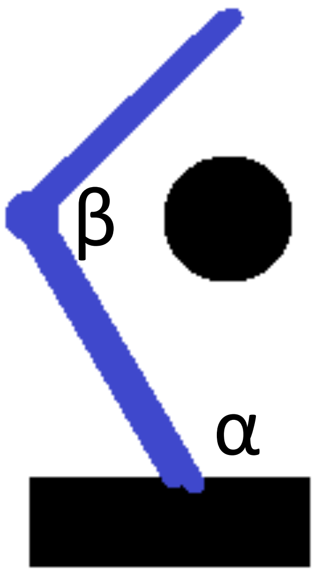
Configuration Space:

- C= [0, π]×[-π, π]

Understanding the C – 2D 2R manipulator

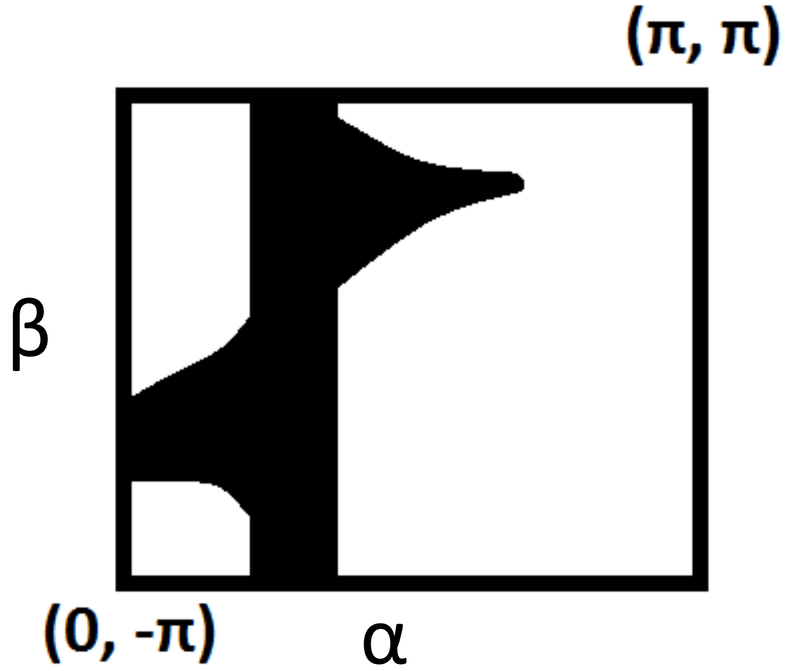
Workspace:

- DoFs : {α, β}



Configuration Space:

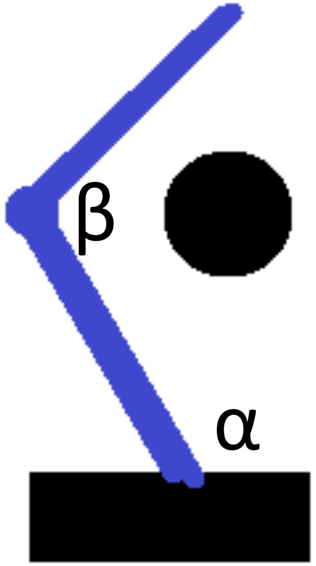
- C= [0, π]×[-π, π]



Understanding the C – 2D 2R manipulator

Workspace:

- DoFs : {α, β}



Configuration Space:

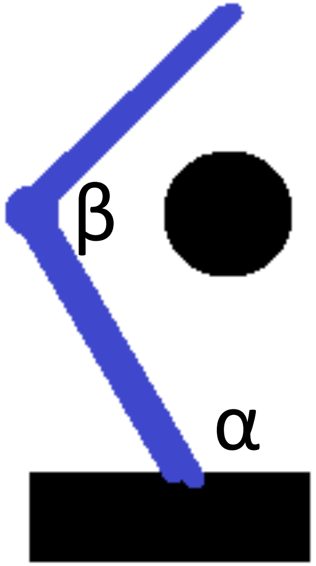
- C= [0, π]×[-π, π]

In reality, a cylinder!!!

Understanding the C – 2D 2R manipulator

Workspace:

- DoFs : {α, β}



Configuration Space:

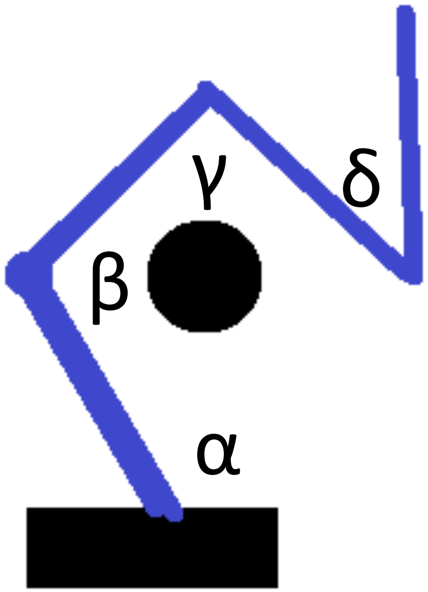
- C= [-π, π]×[-π, π]

A "donut" like shape if also the other joint can revolve 360 degrees.

Understanding the C – 2D 4R manipulator

Workspace:

- DoFs :



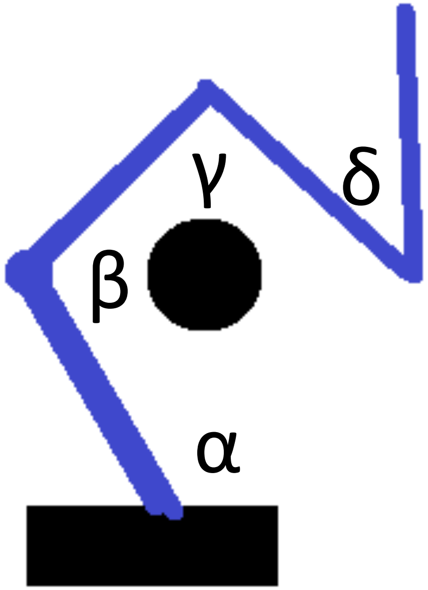
Configuration Space:

- C =

Understanding the C – 2D 4R manipulator

Workspace:

- DoFs : {α, β, γ, δ}



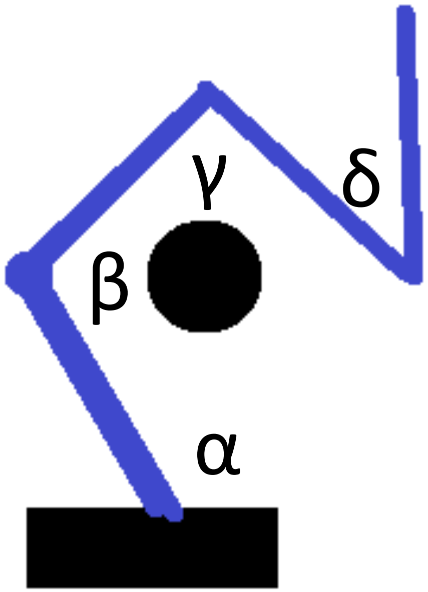
Configuration Space:

- C =

Understanding the C – 2D 4R manipulator

Workspace:

- DoFs : {α, β, γ, δ}



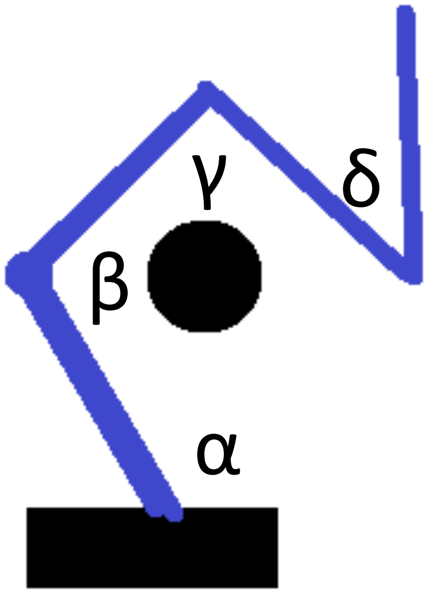
Configuration Space:

- $C = [0, \pi] \times [-\pi, \pi] \times [-\pi, \pi] \times [-\pi, \pi]$

Understanding the C – 2D 4R manipulator

Workspace:

- DoFs : {α, β, γ, δ}



Configuration Space:

- C = [0, π]×[-π, π]×[-π, π]×[-π, π]

- 4-D Space!
- Hard to visualize.

The Motion Planning Problem

- Given an initial configuration and a specific group of goal configurations return a path of valid configurations that move the robot from the initial to the goal state.
- Not an easy problem: P-SPACE Hard! [J. H. Reif., 1979.]

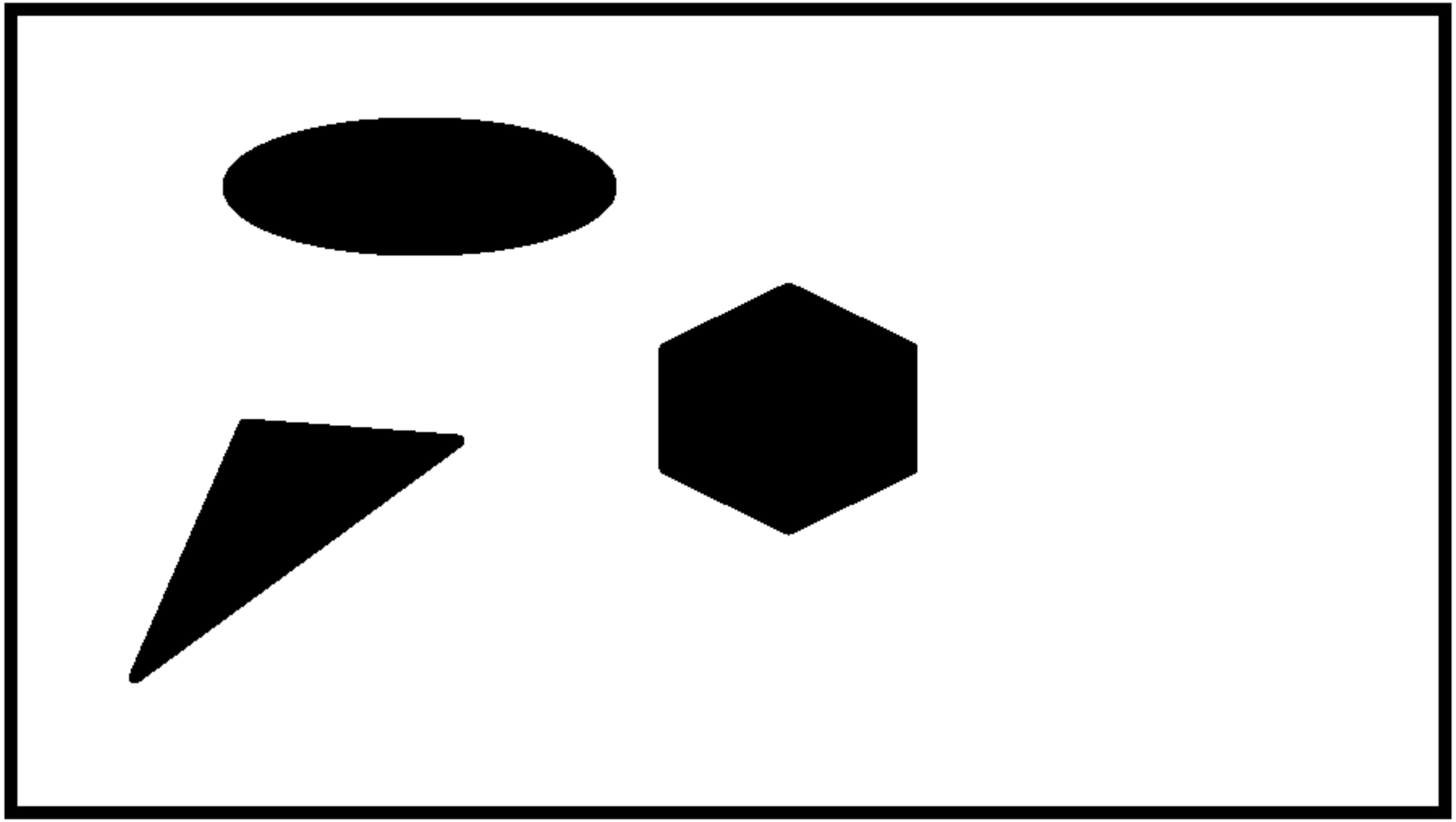
Sampling based techniques

- Sampling based techniques come to the rescue!
  - Graphs are created using random samples inside the C-free
  - By connecting the samples the algorithms try to find a path connecting the initial with the goal configuration.
- Only a function that checks if a random configuration is valid needed.

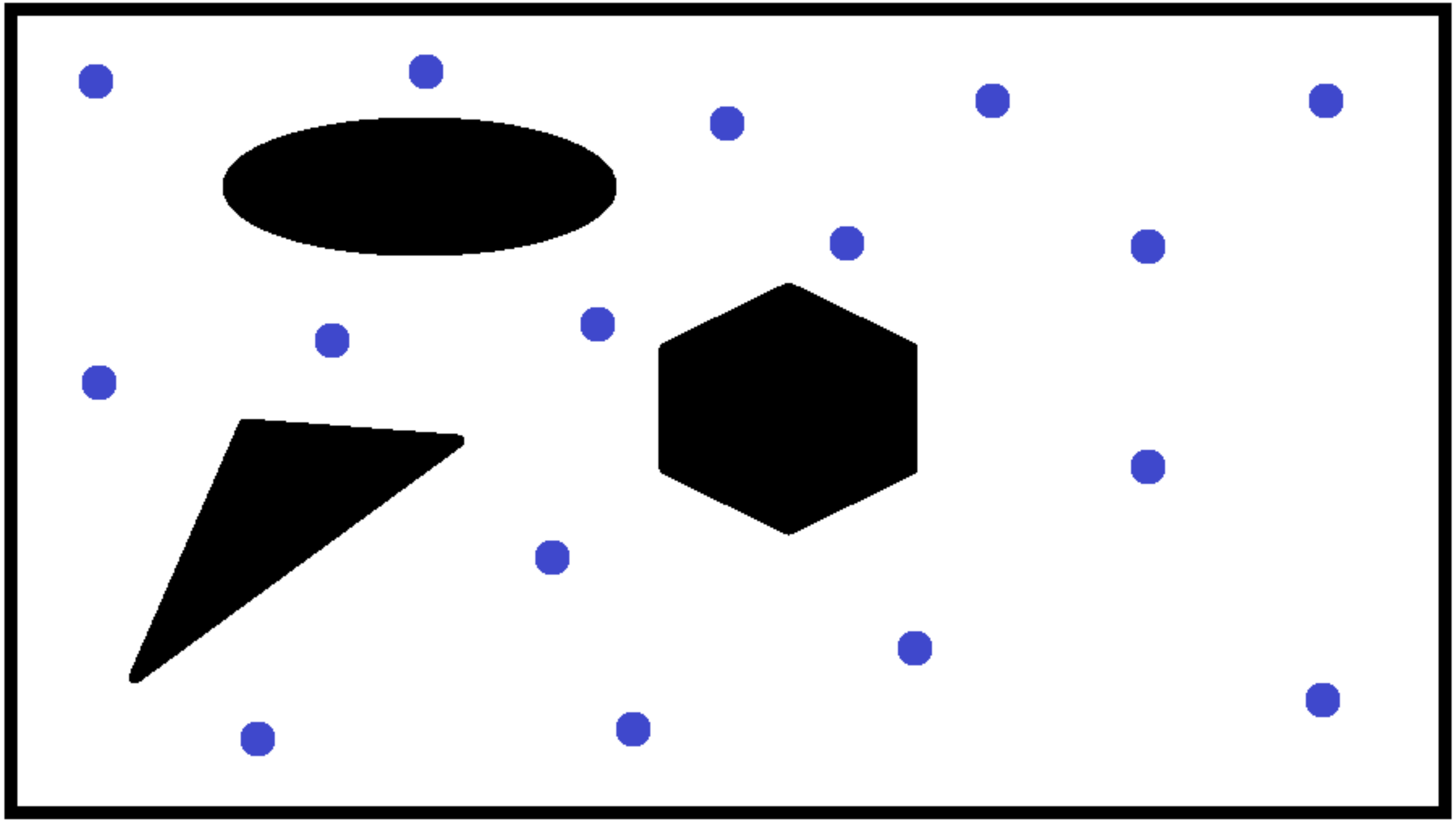
Probabilistic Roadmaps (PRMs)

- Sample uniformly random configurations for a while.
- Keep the valid ones.
- Try to connect the samples creating nodes and by finding the minimum spanning tree.
- Use the structure for each problem by connecting the initial and goal configurations to the graph.

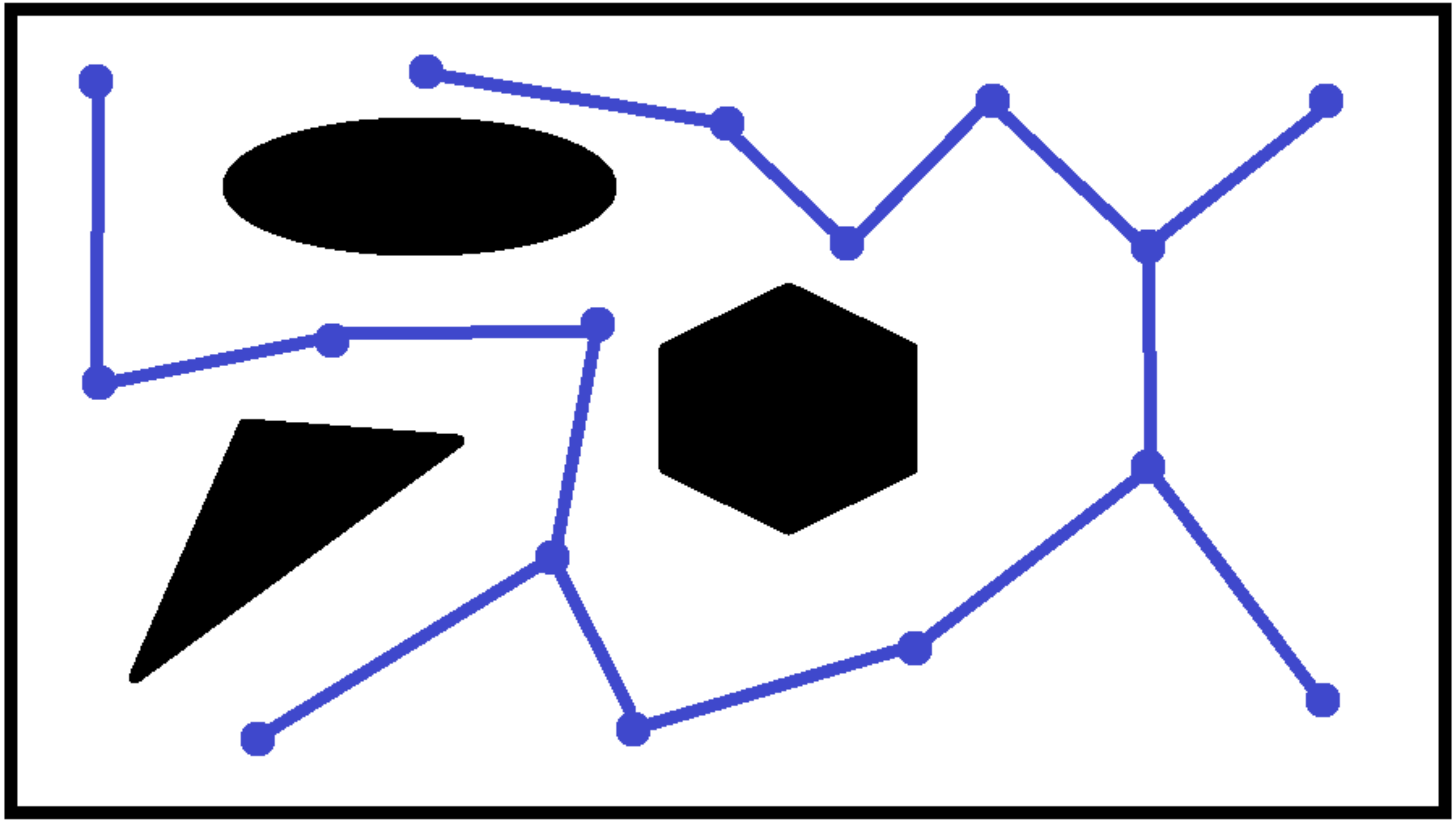
PRMs



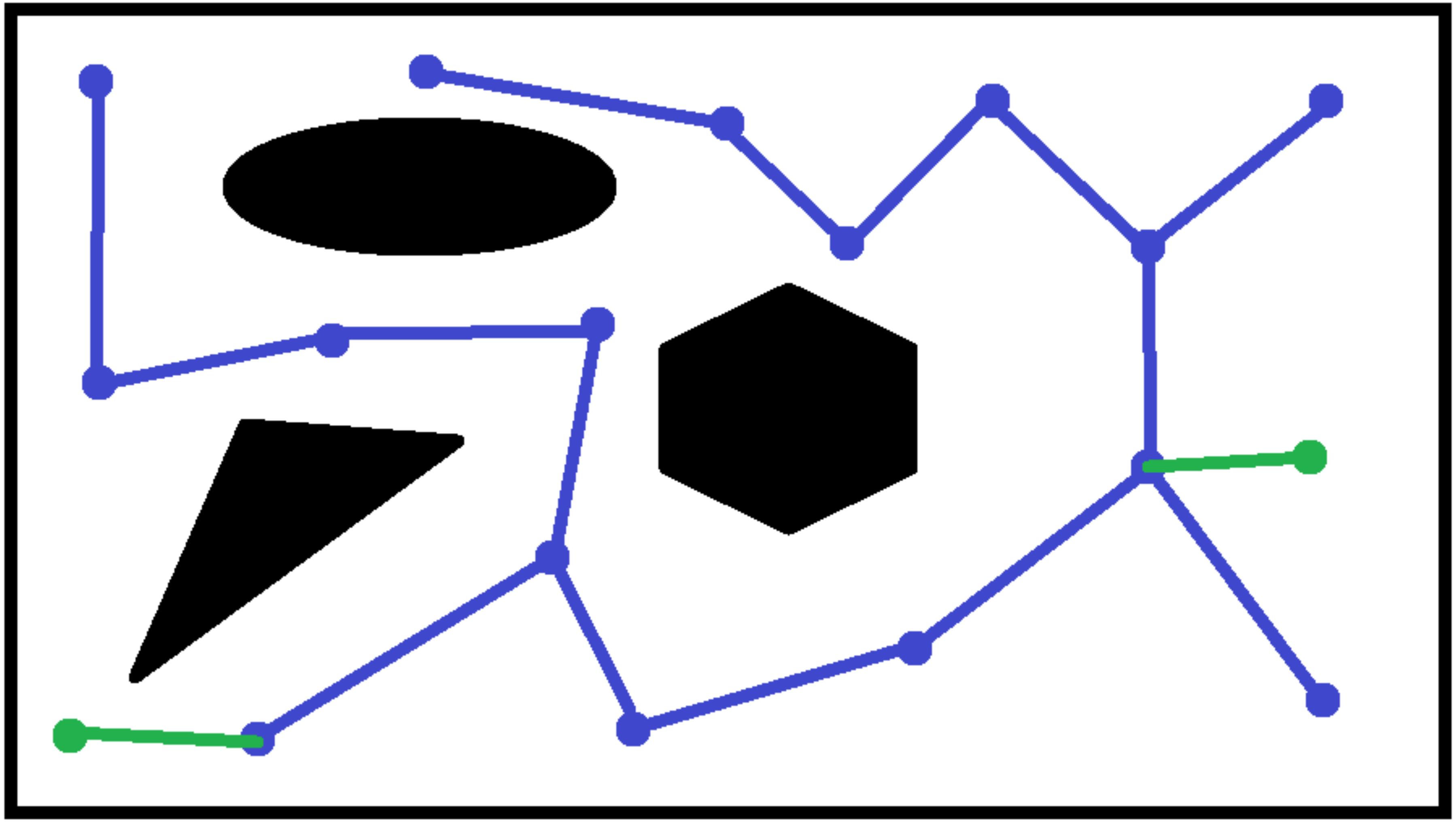
PRMs



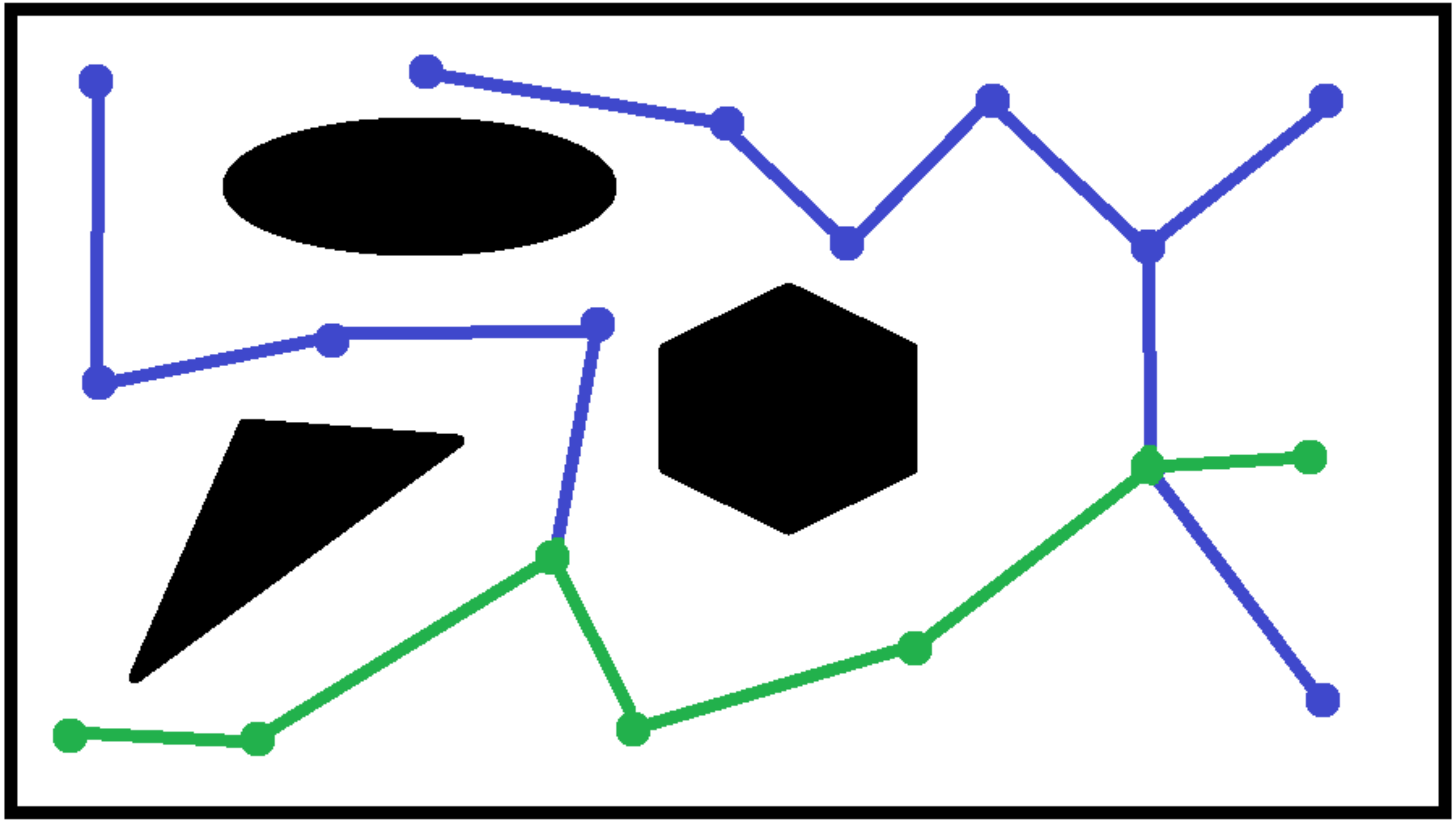
PRMs



PRMs



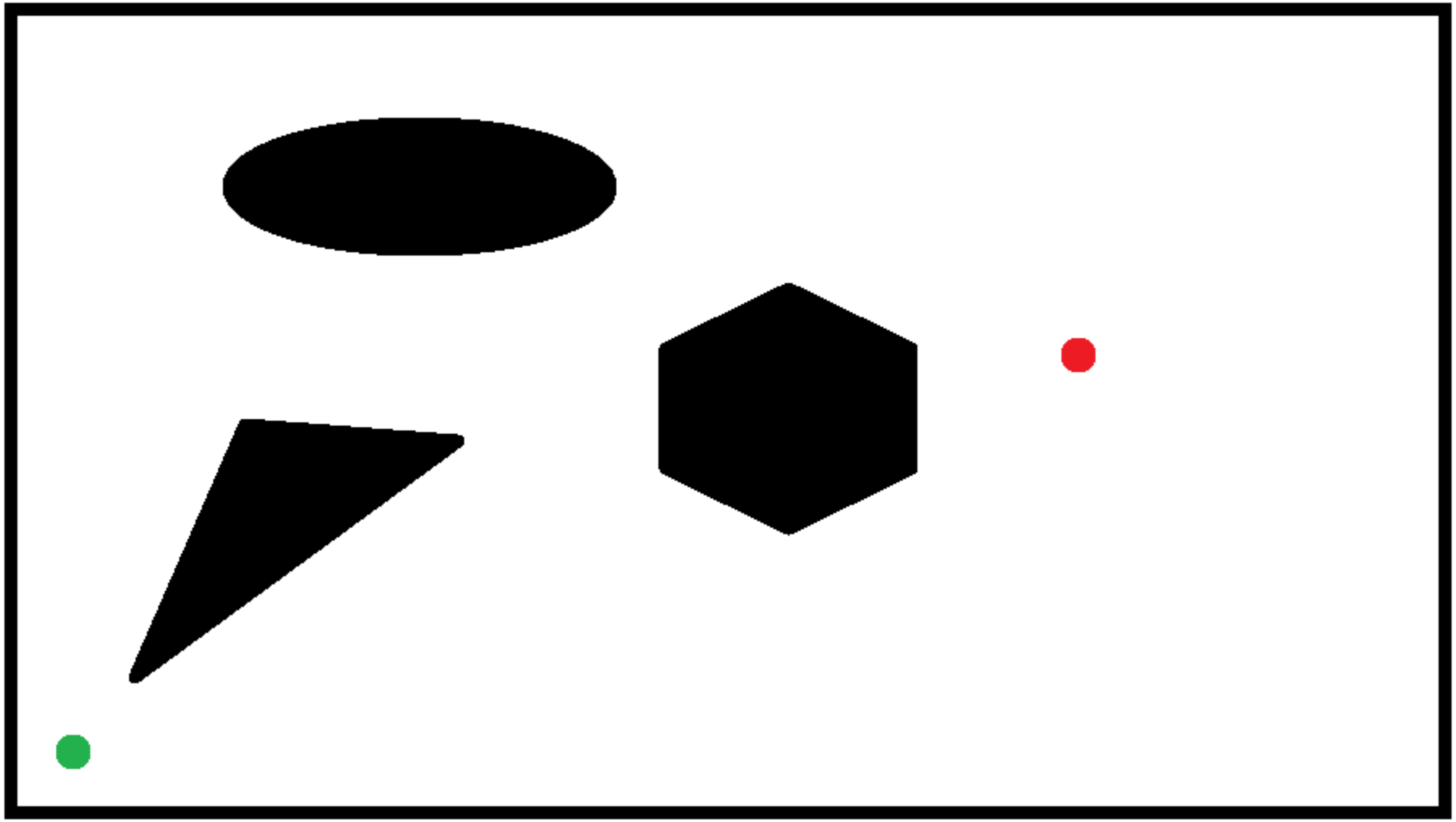
PRMs



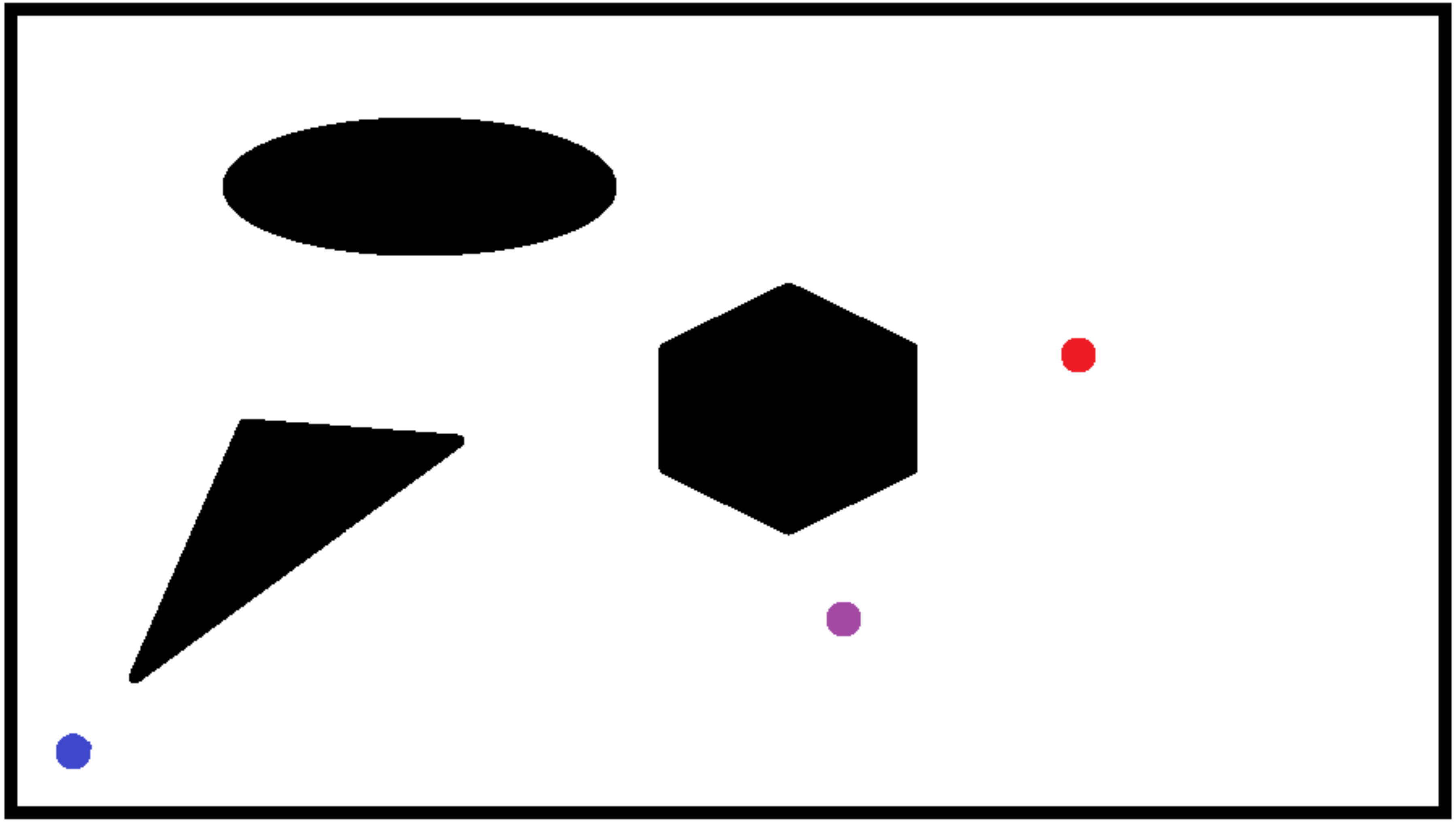
Rapidly exploring Random Trees (RRTs)

- Start from the initial configuration. Iteratively:
  - Sample uniformly a random sample q_{rand}.
  - Find the closer node to that random sample q_{near}.
  - Extend the tree from q_{near} to q_{rand} creating the new node q_{new}.
  - Until timeout or the goal configuration is reached.
  - Then traverse the tree from the goal configuration to the root.

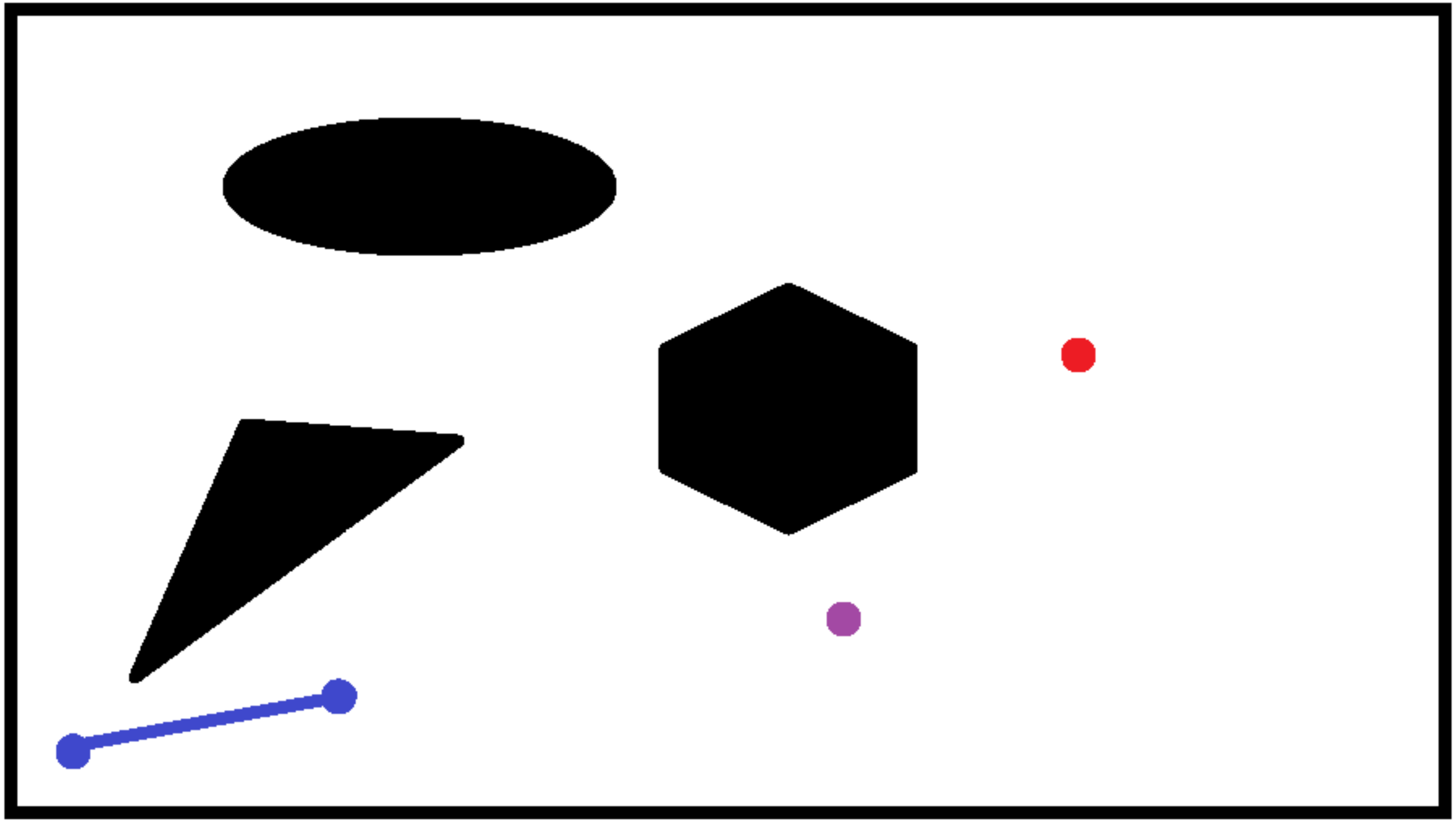
RRTs



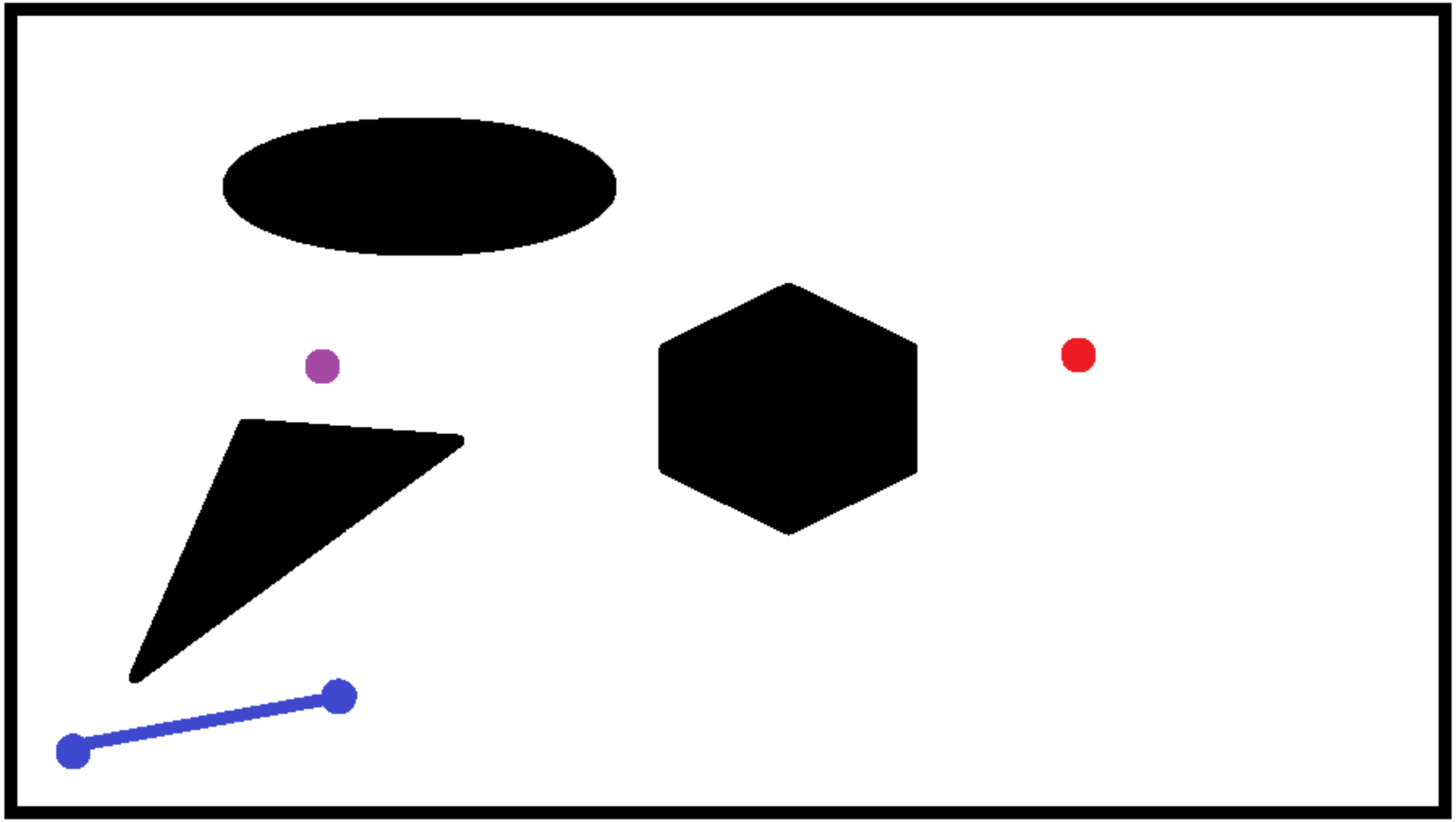
RRTs



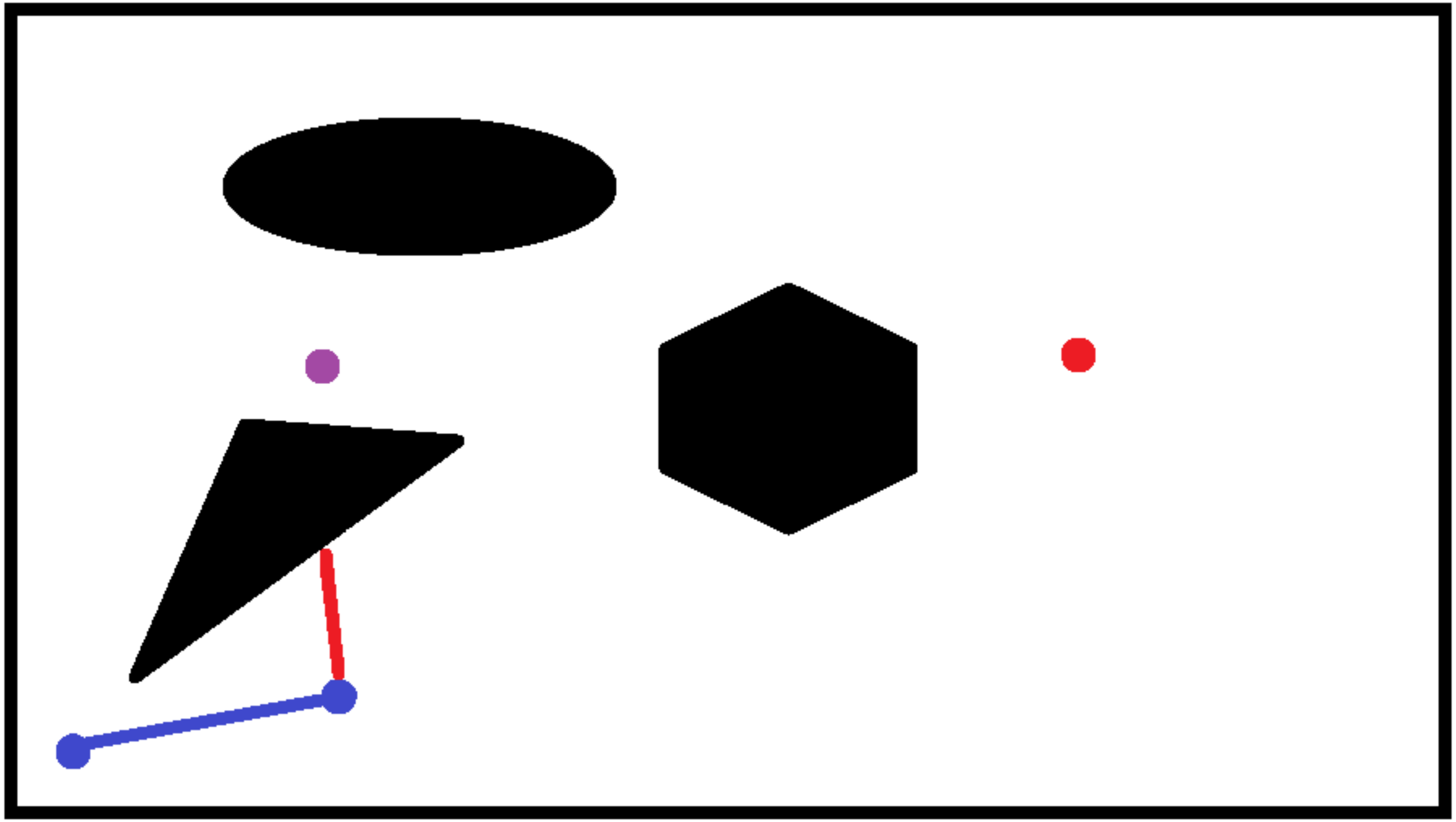
RRTs



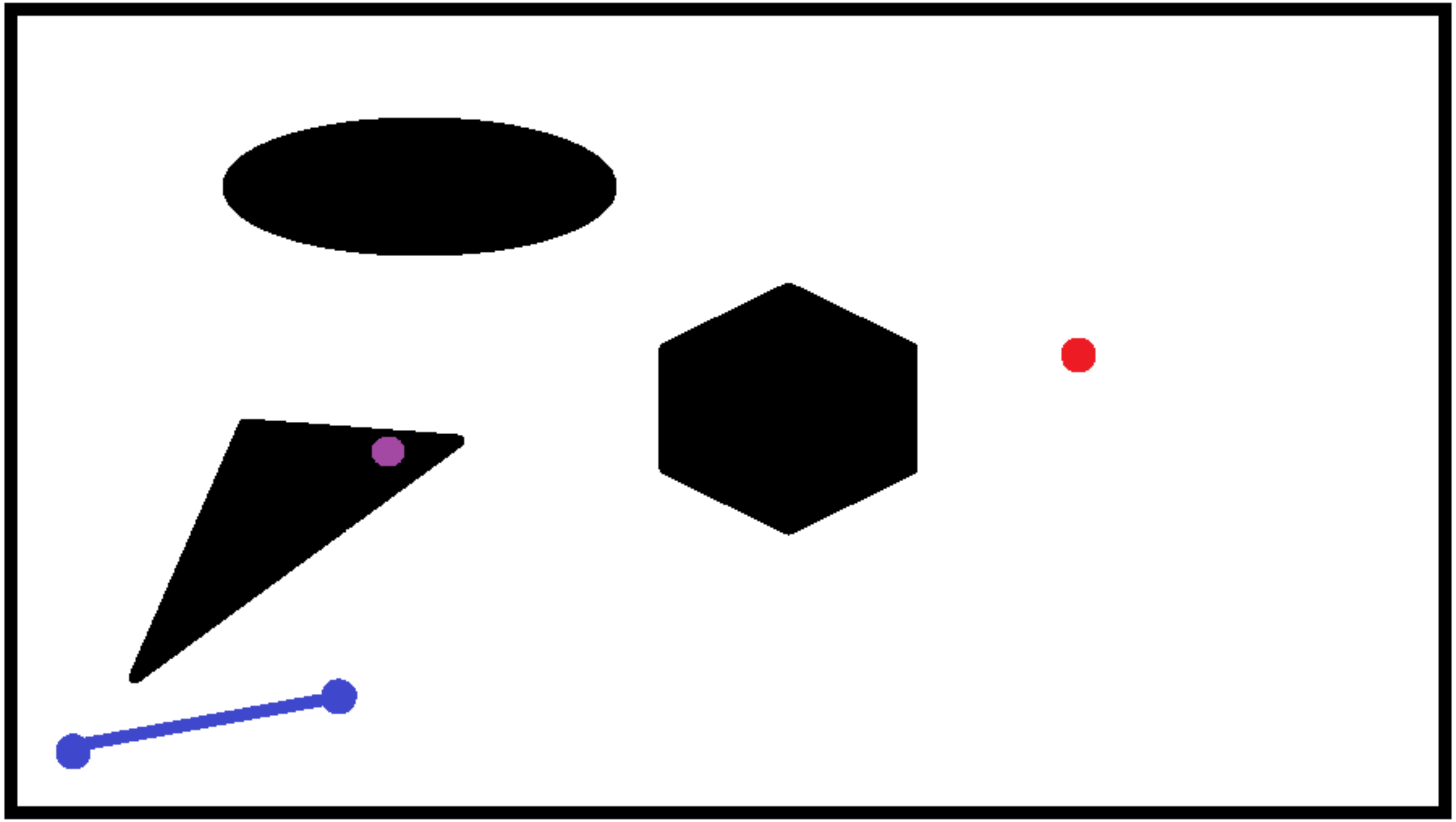
RRTs



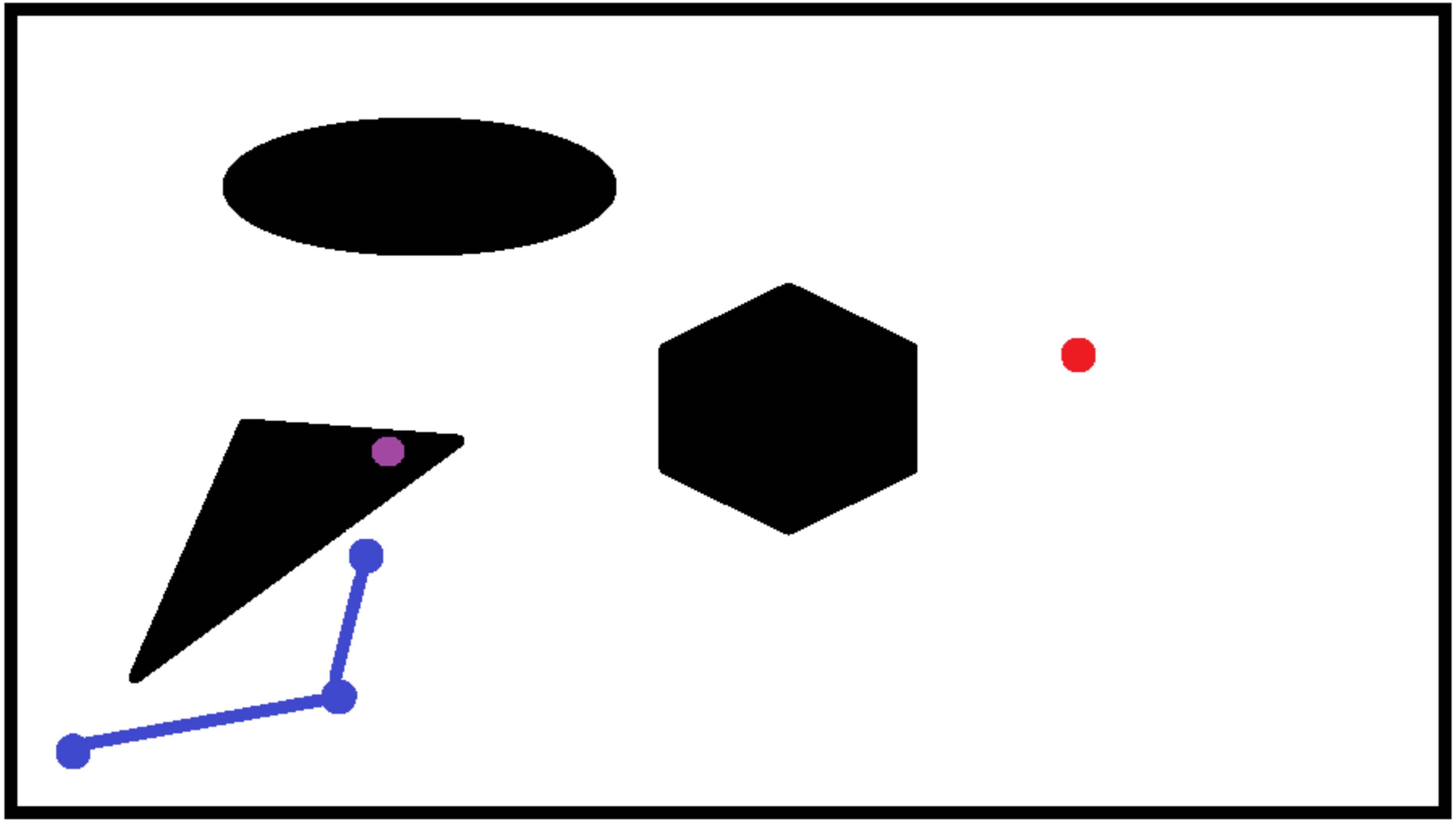
RRTs



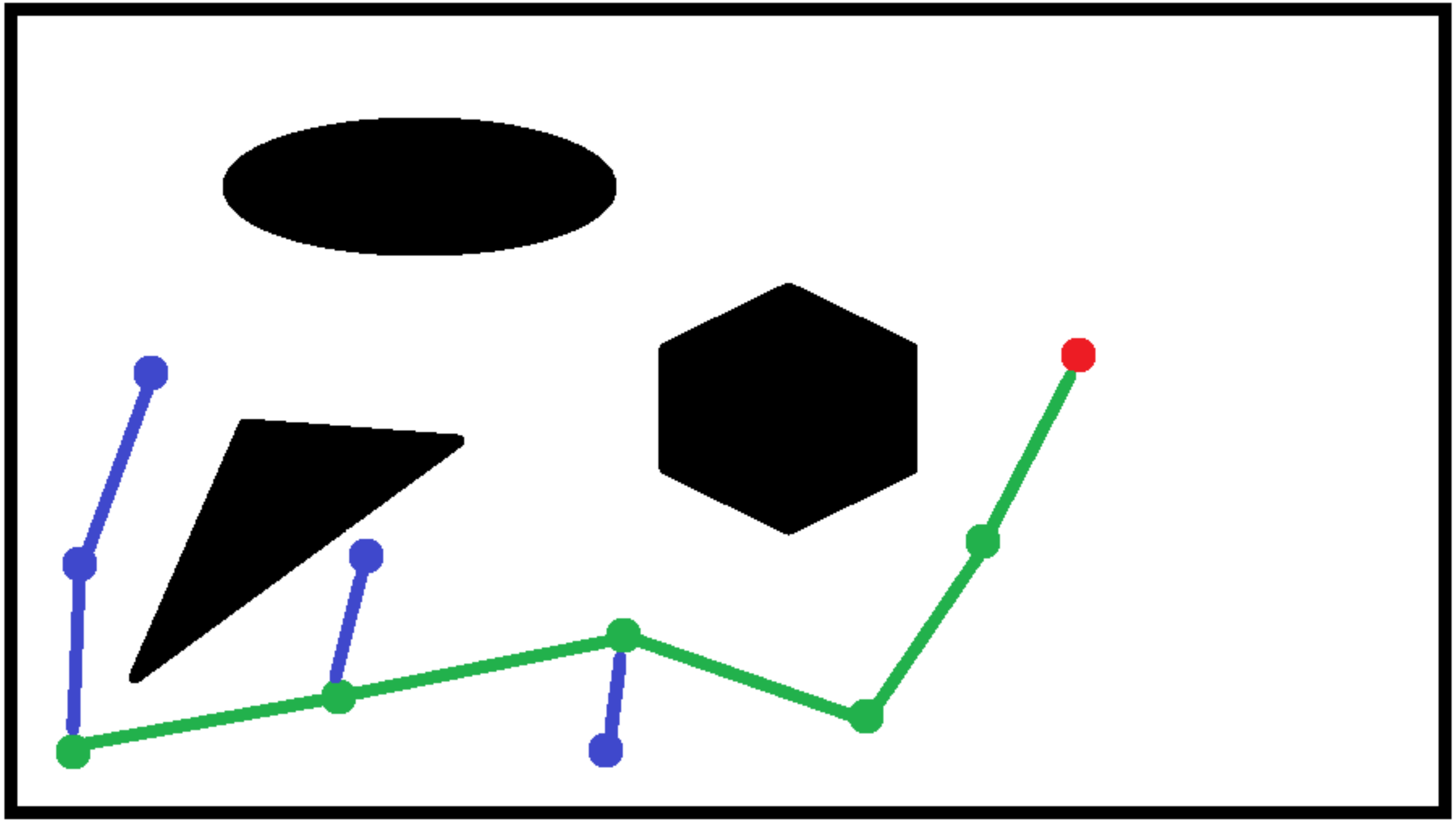
RRTs



RRTs



RRTs



Probabilistic Completeness

- Both RRTs and PRMs have probabilistic completeness guarantees:
  - The probability of a solution to be found, if one exists, tends to 1 in the infinity.
    - Yes. Theoretically, the problem is that hard that the most efficient techniques may need to run for an eternity for a solution!
    - Practically: Solutions can be found in reasonable time, if we are lucky.
- Probabilistic Completeness is the best guarantee that a complete general planner can offer.

PRMs vs RRTs

PRMs

- In general, they need more time.
- Multi-query planners given the same environment.
- Only for holonomic systems. Not ideal for most real mobile robots.
- Should cover the whole C in order to work properly.

RRTs

- They are much faster in most cases.
- Single query planners. More robust to different conditions.
- Both for holonomic and non-holonomic systems.
- They can provide fast solutions only with few samples.

High-Dimensional Systems

High-Dimensional Systems

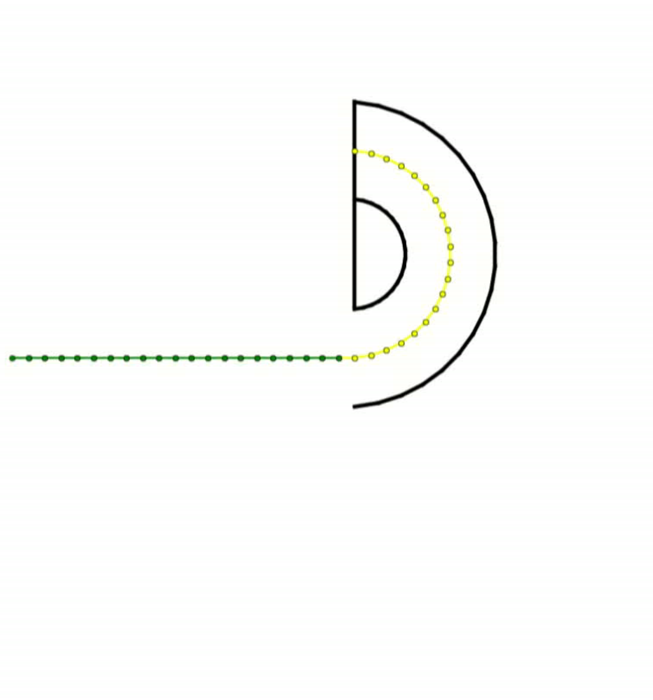
Norwegian University of Science and Technology, Kongsberg Maritime

High-Dimensional Systems

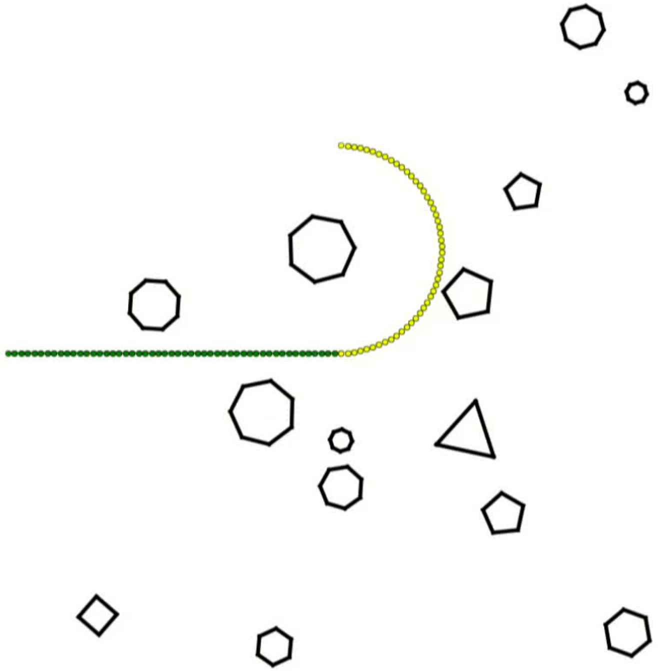
High-Dimensional Systems

Experiments –HR Manipulator

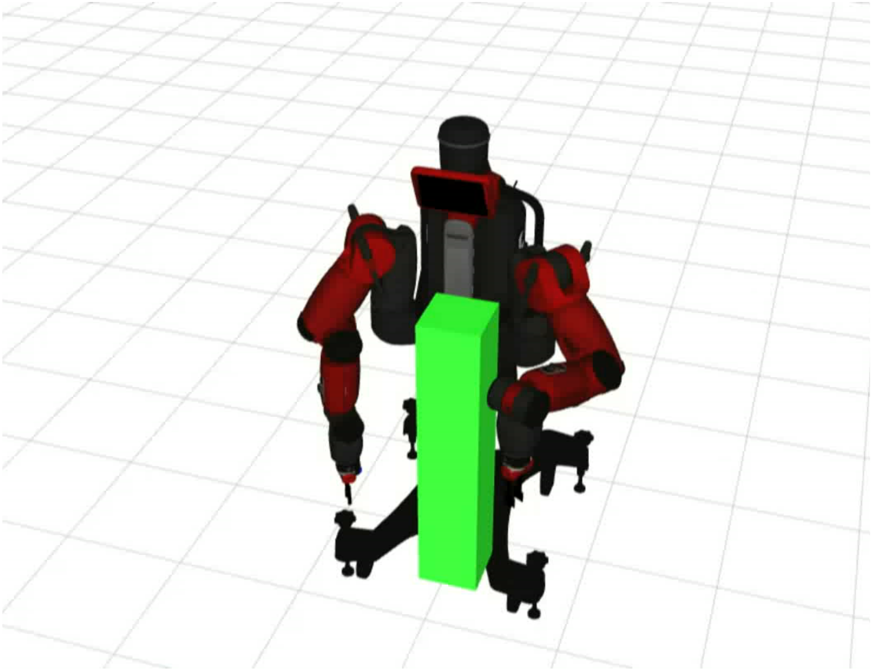
Horn

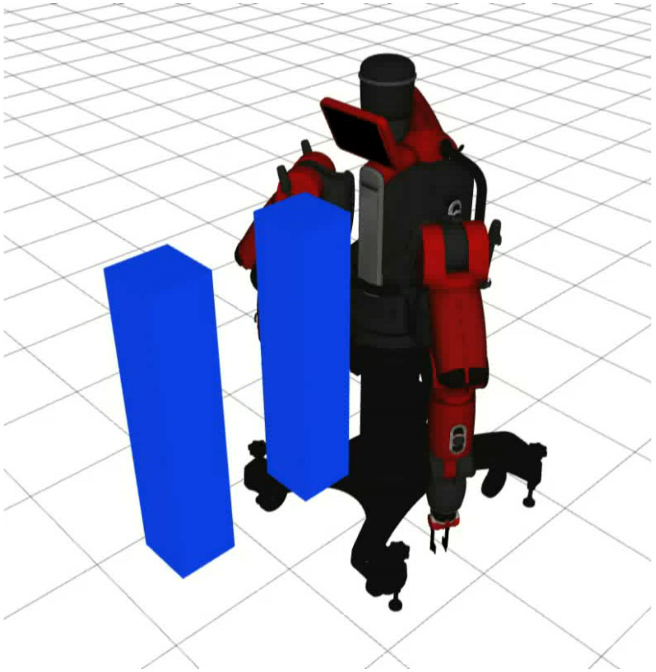


Cluttered



Experiments - Baxter





Experiments - Baxter