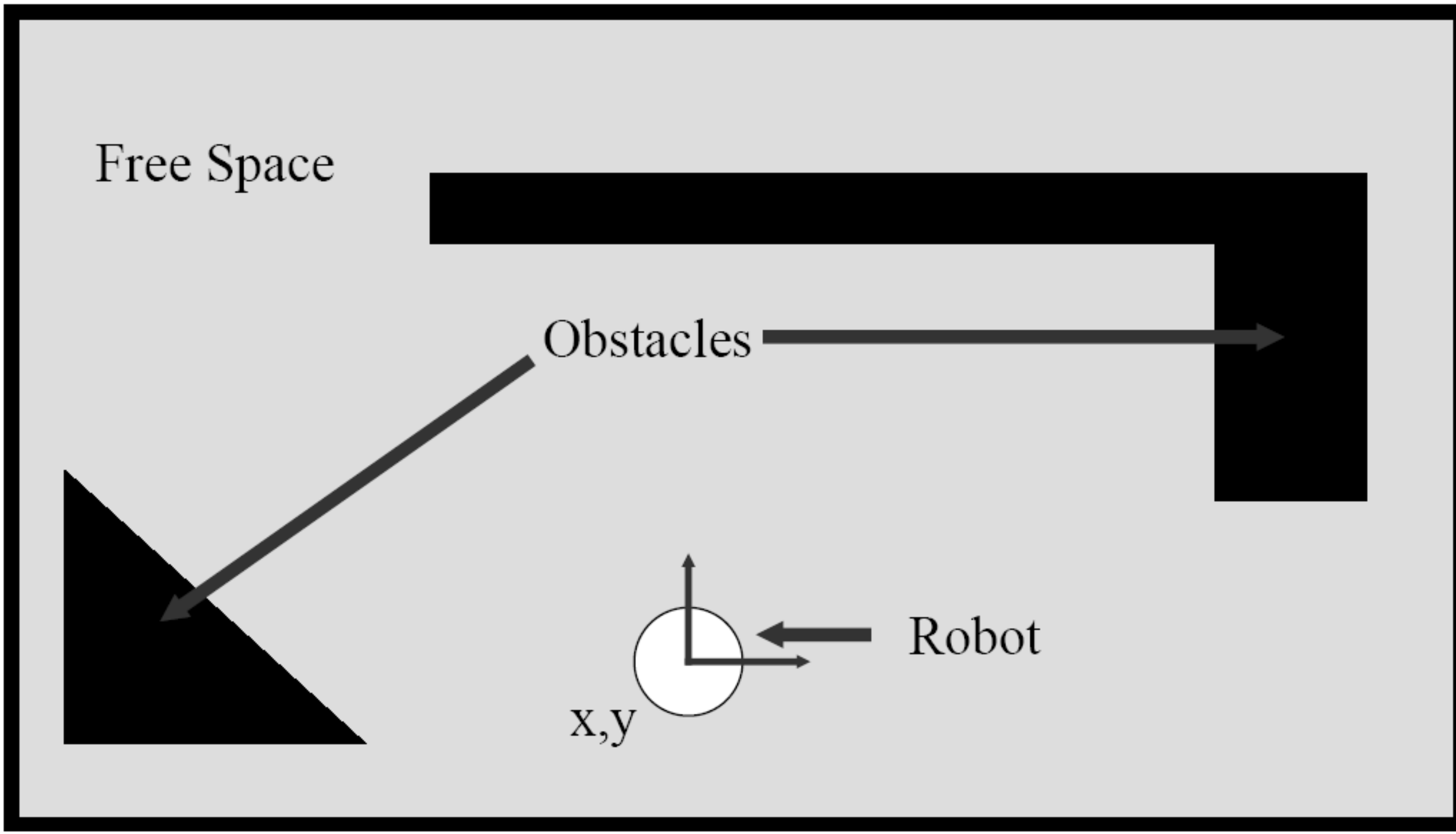
UNIVERSITY OF SOUTH CAROLINA

CSCE 574 ROBOTICS

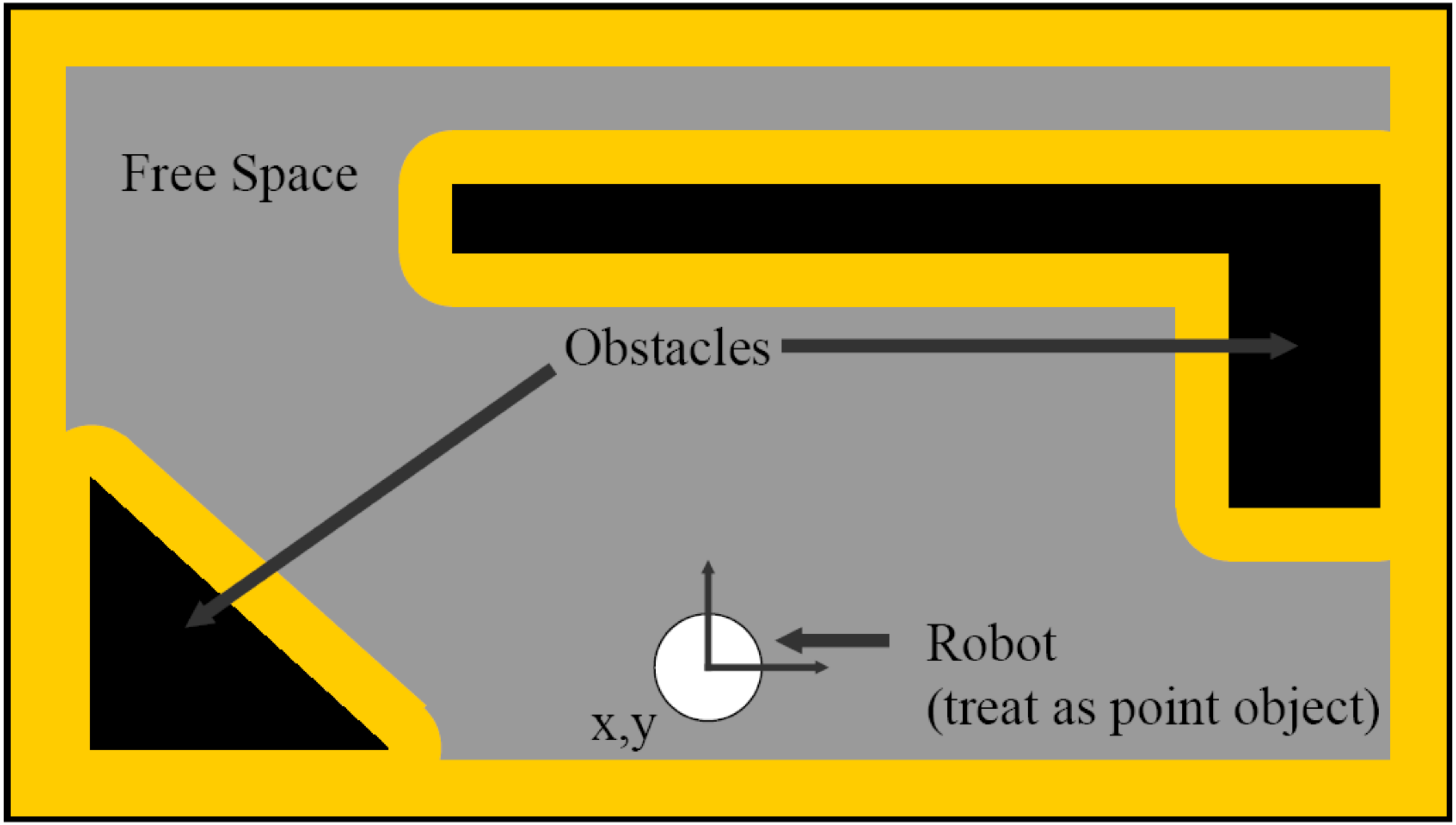
Configuration Space

Ioannis Rekleitis

Configuration Space



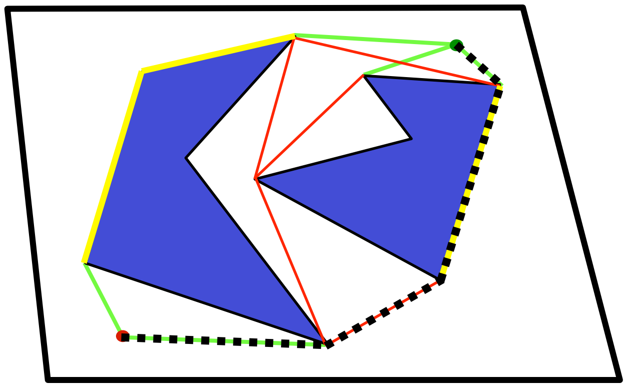
Configuration Space



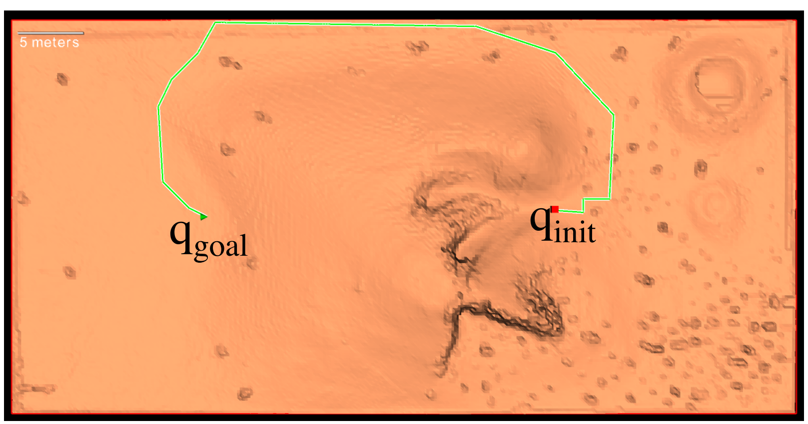
Definition

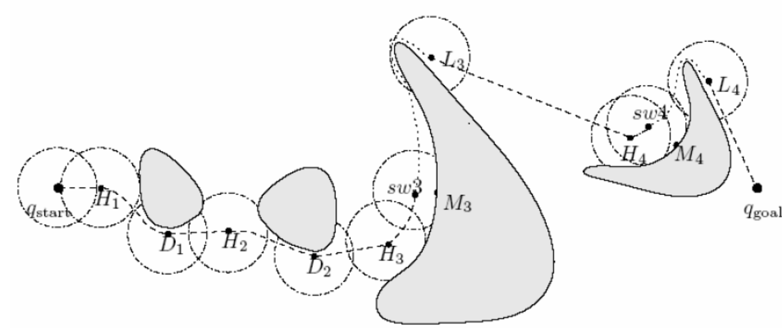
- A robot configuration is a specification of the positions of all robot points relative to a fixed coordinate system
- Usually a configuration is expressed as a "vector" of position/orientation parameters

What is a Path?

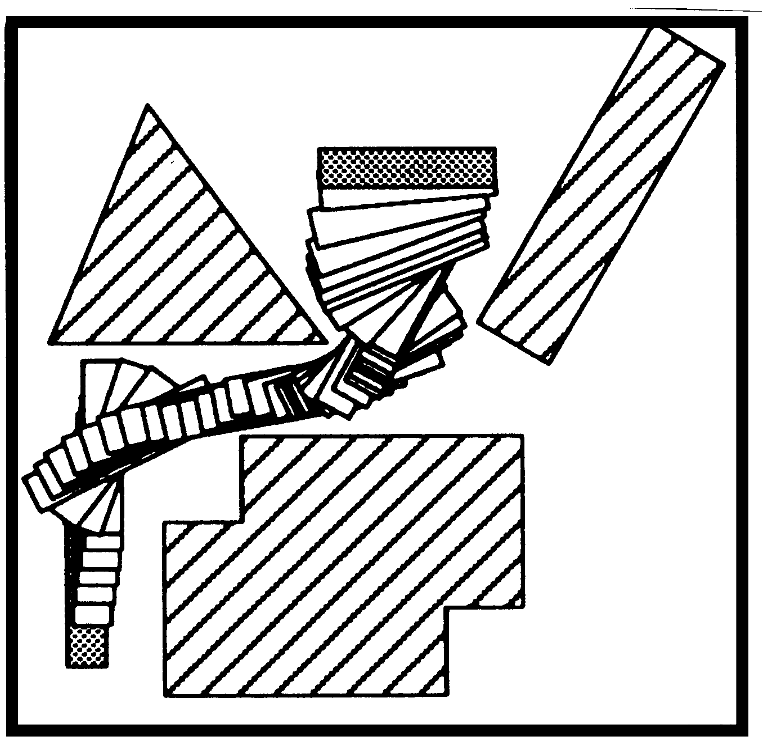


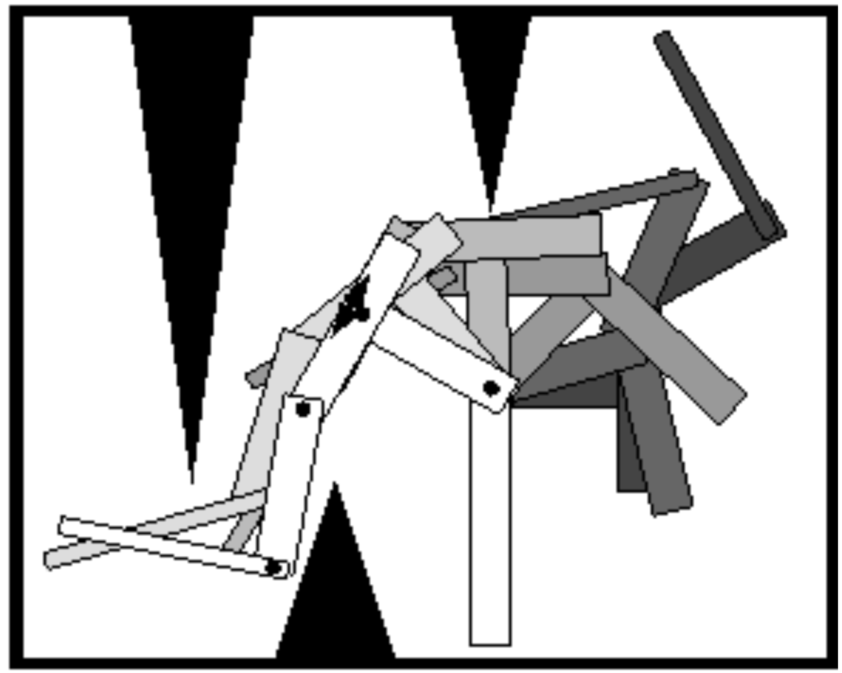
- q_{init}
- q_{goal}



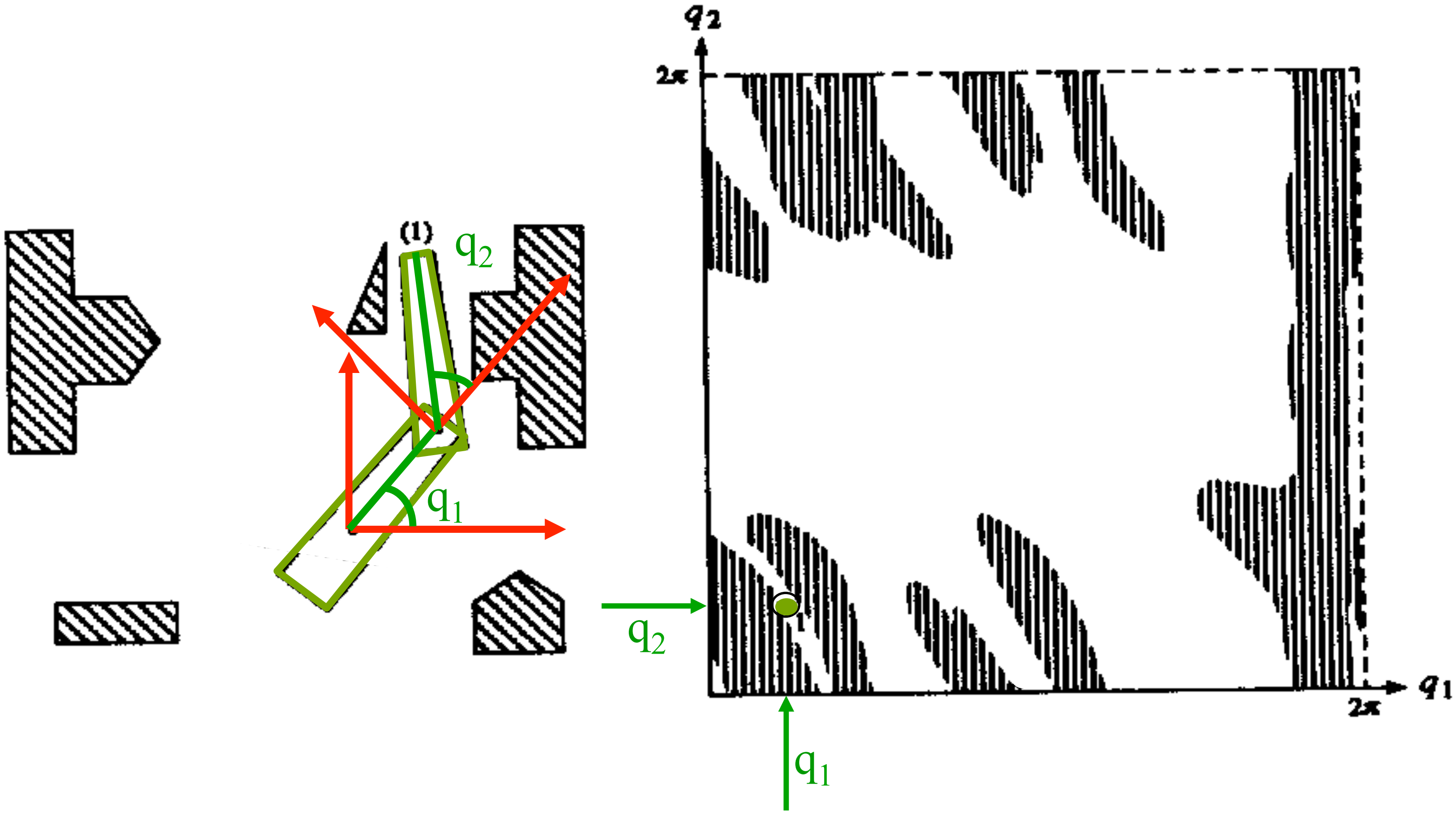


What is a Path?

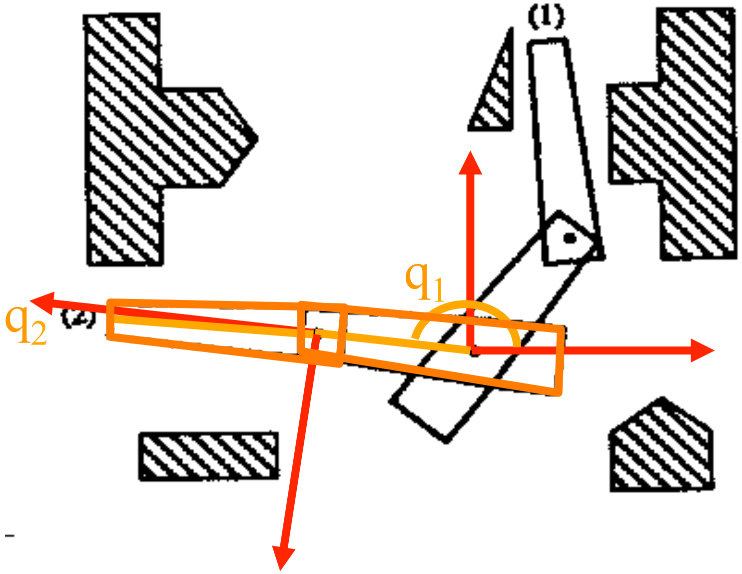


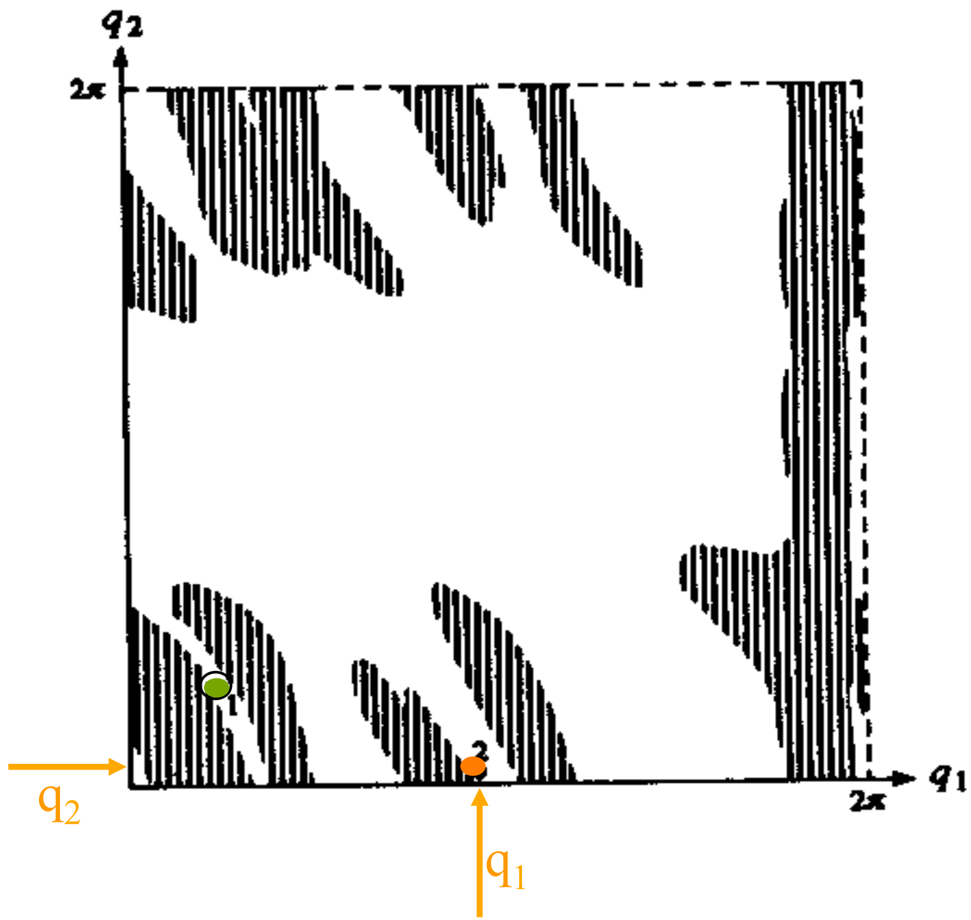


Tool: Configuration Space (C-Space C)

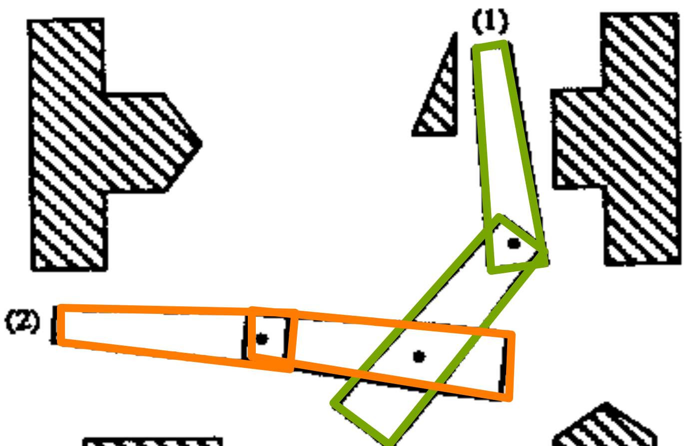


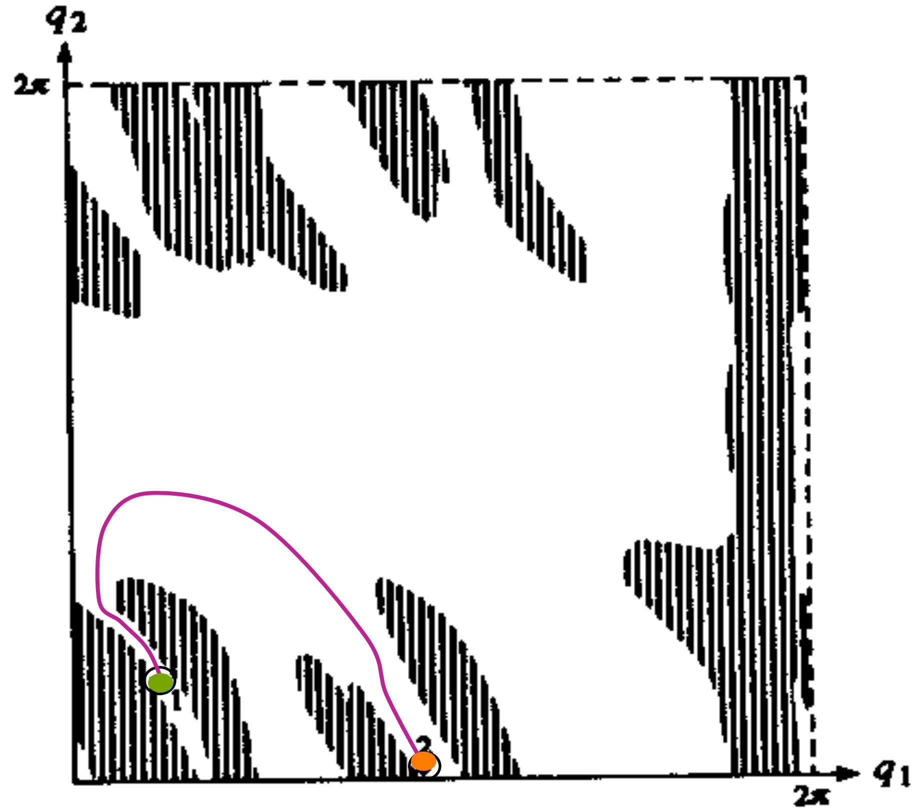
Tool: Configuration Space (C-Space C)



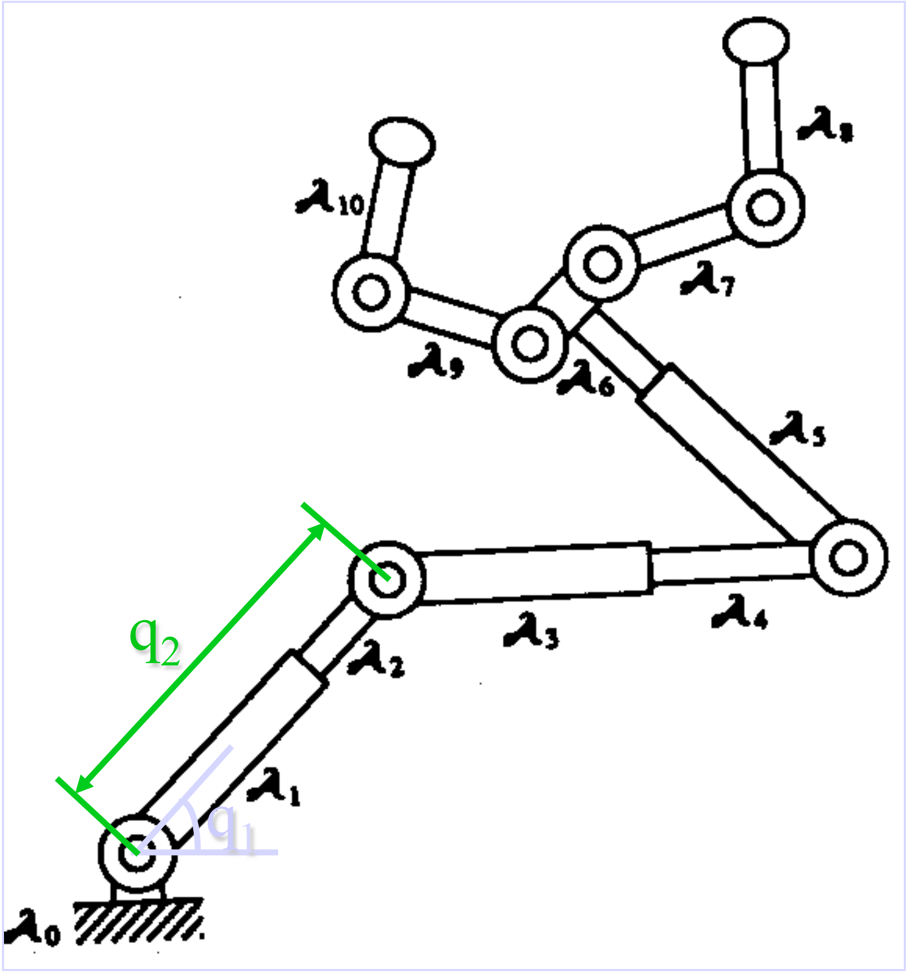


Tool: Configuration Space (C-Space C)





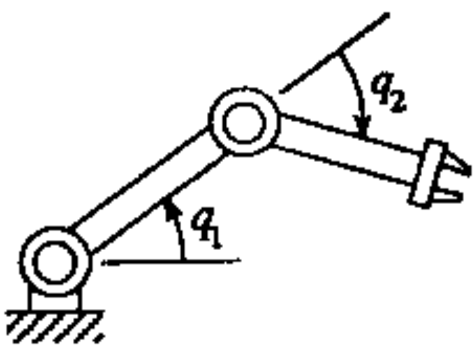
Articulated Robot Example



$$q = (q_1, q_2, \ldots, q_{10})$$

Configuration Space of a Robot

- Space of all its possible configurations
- But the topology of this space is usually not that of a Cartesian space



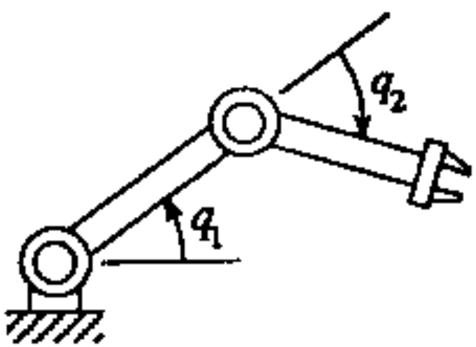
Configuration Space of a Robot

- Space of all its possible configurations
- But the topology of this space is usually not that of a Cartesian space

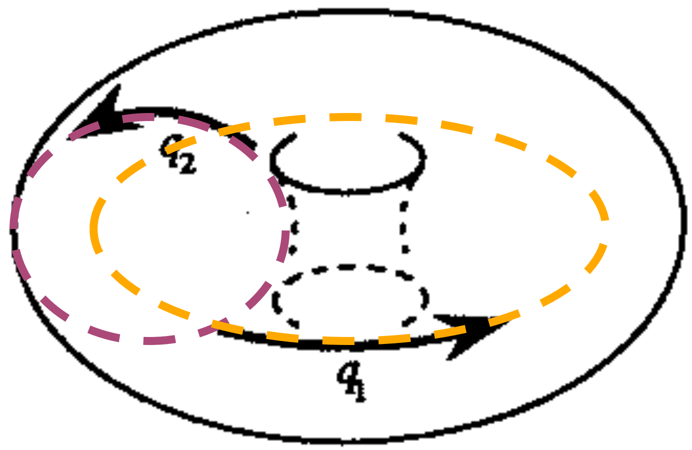
$C = S^1 \times S^1$

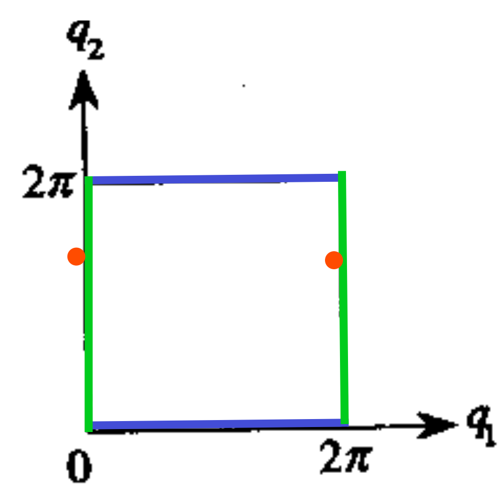
Configuration Space of a Robot

- Space of all its possible configurations
- But the topology of this space is usually not that of a Cartesian space



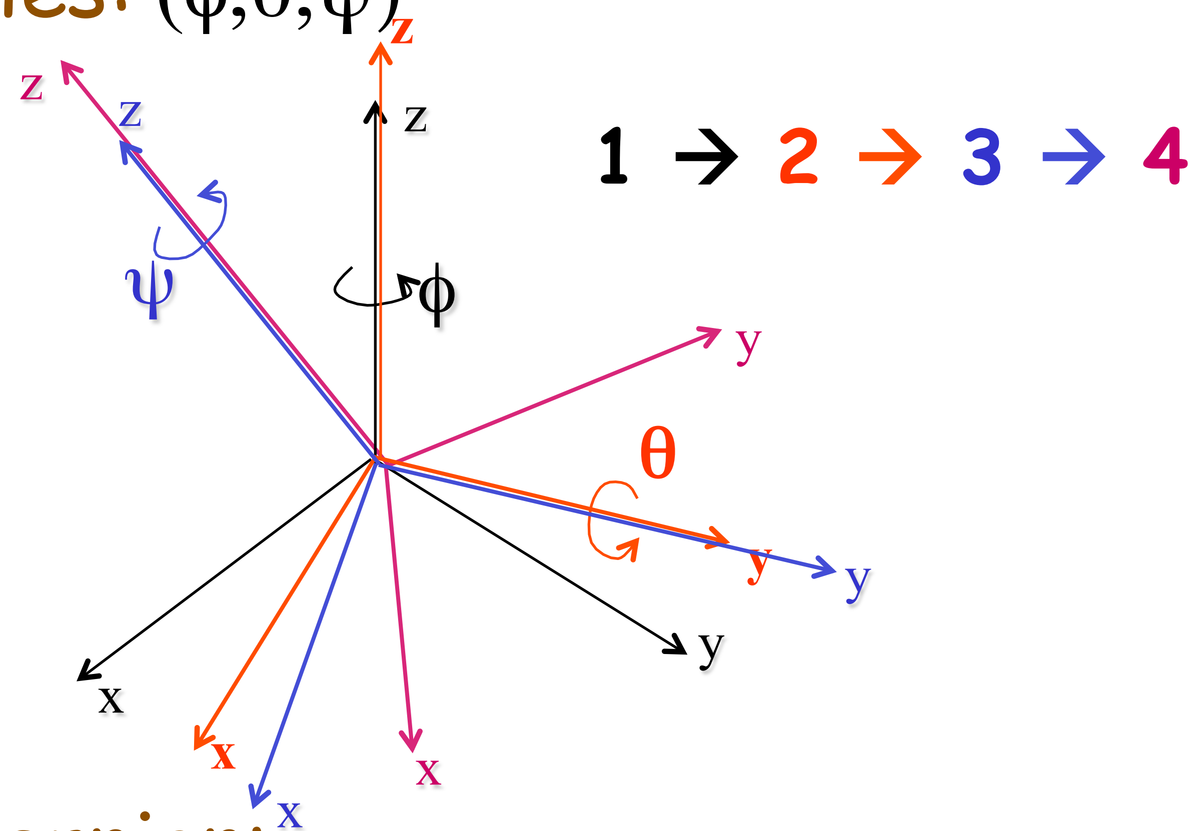
$C = S^1 \times S^1$





Parameterization of SO(3)

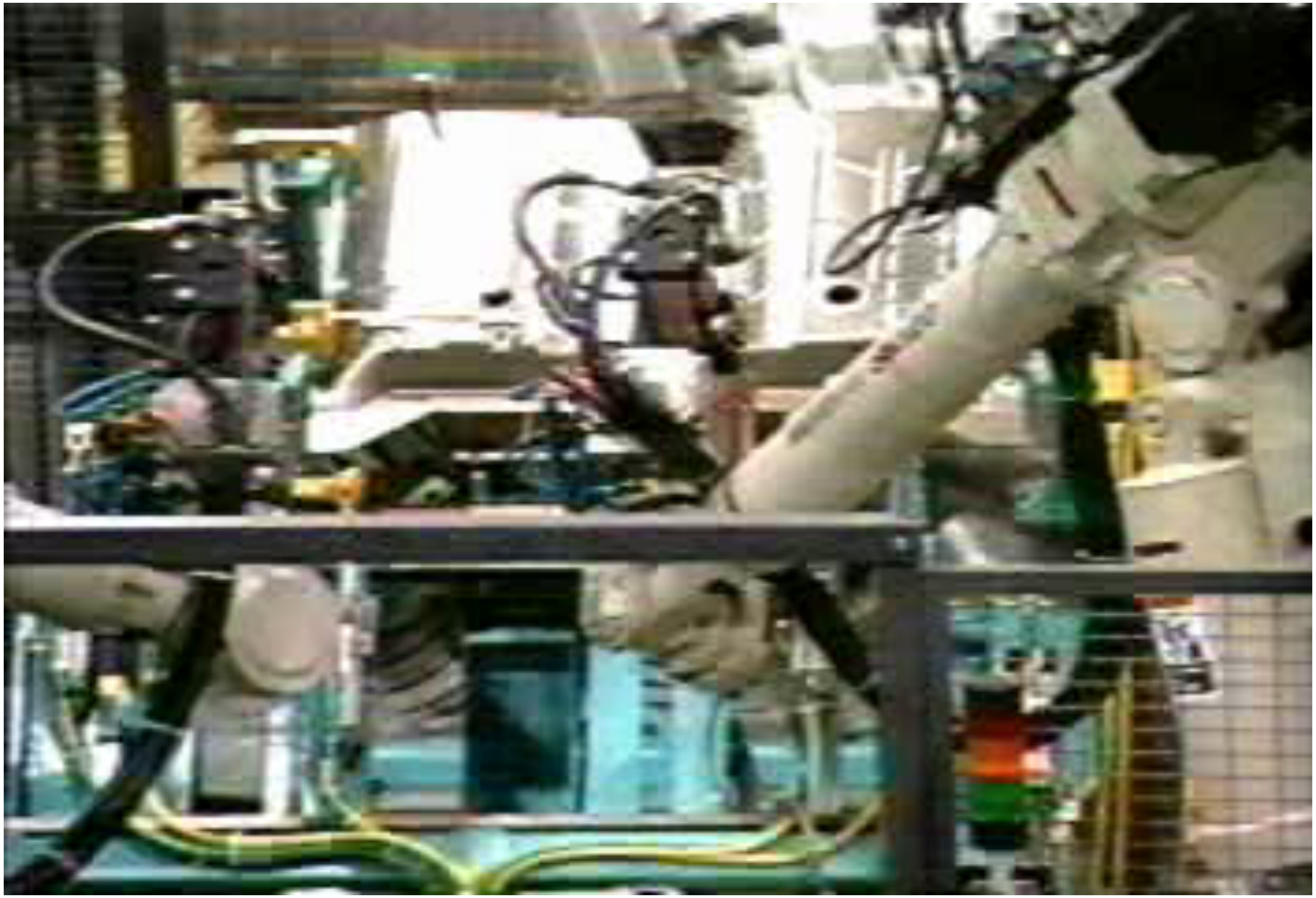
- Euler angles: (ϕ,θ,ψ)



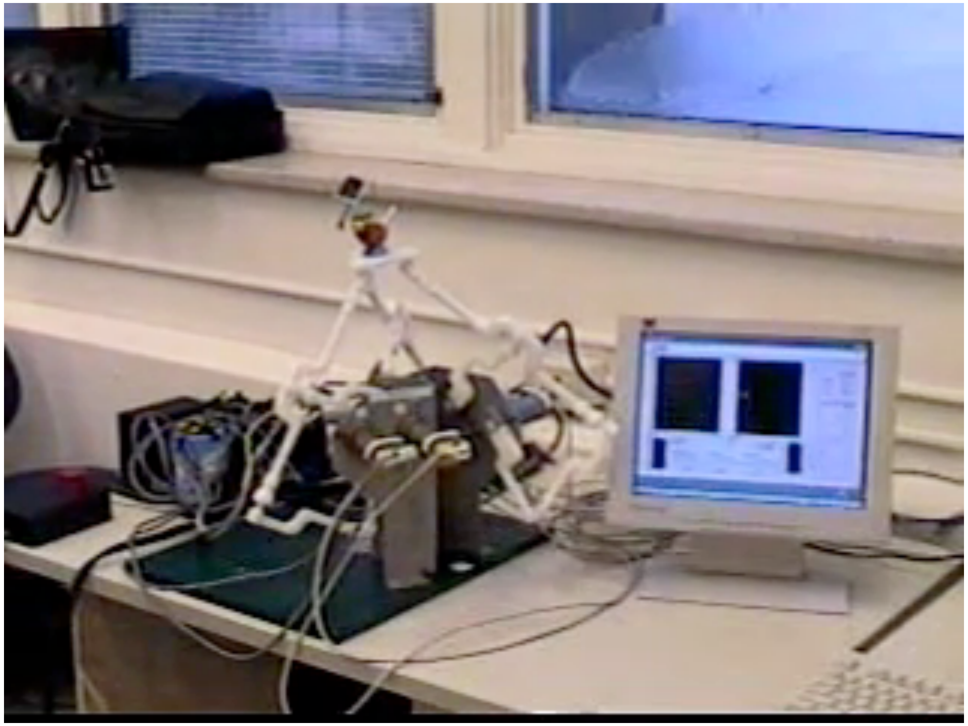
- Unit quaternion:

$$(\cos\theta/2,\ n_1 \sin\theta/2,\ n_2 \sin\theta/2,\ n_3 \sin\theta/2)$$

A welding robot



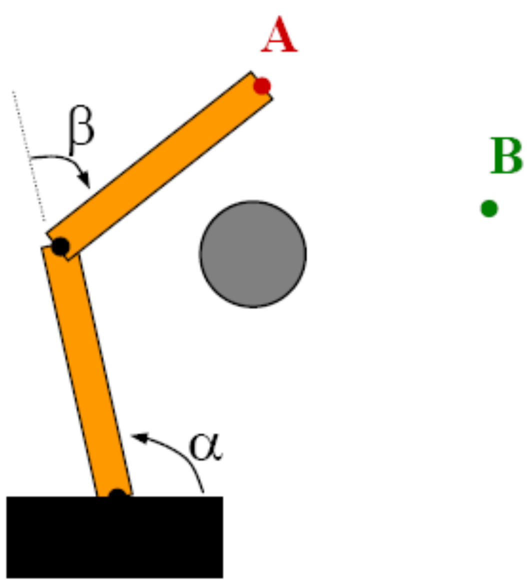
A Stuart Platform



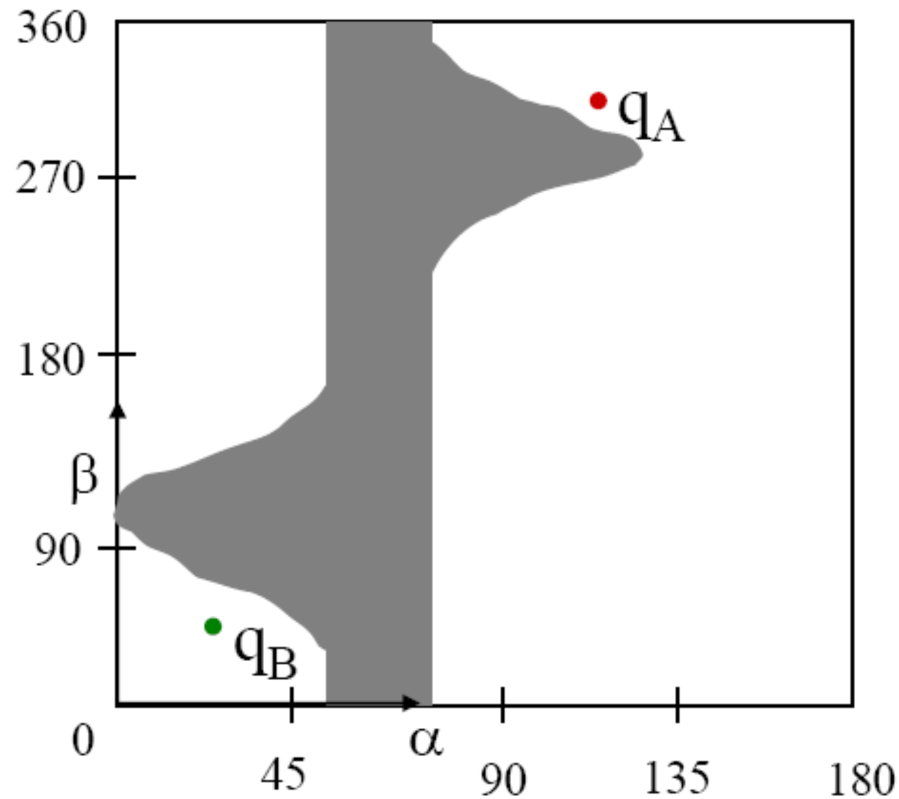
Barrett WAM arm on a mobile platform

Configuration Space Obstacle

Reference *configuration*



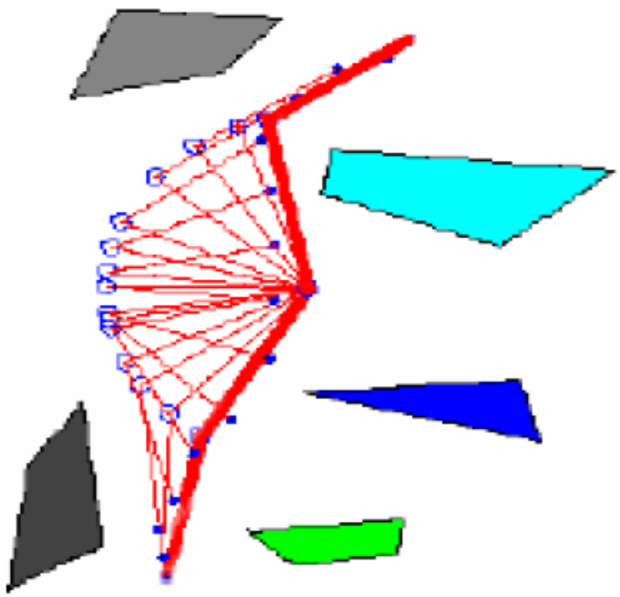
How do we get from A to B ?

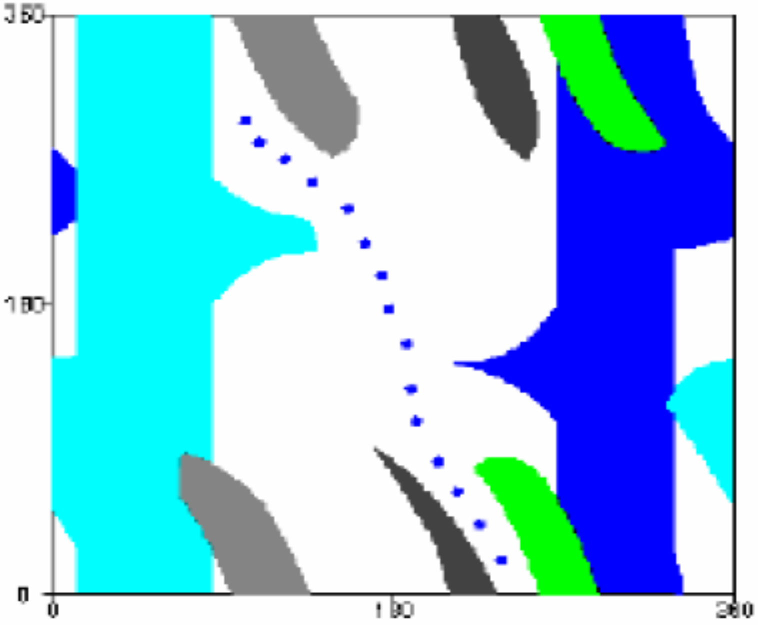


An obstacle in the robot's workspace

The C-space representation of this obstacle…

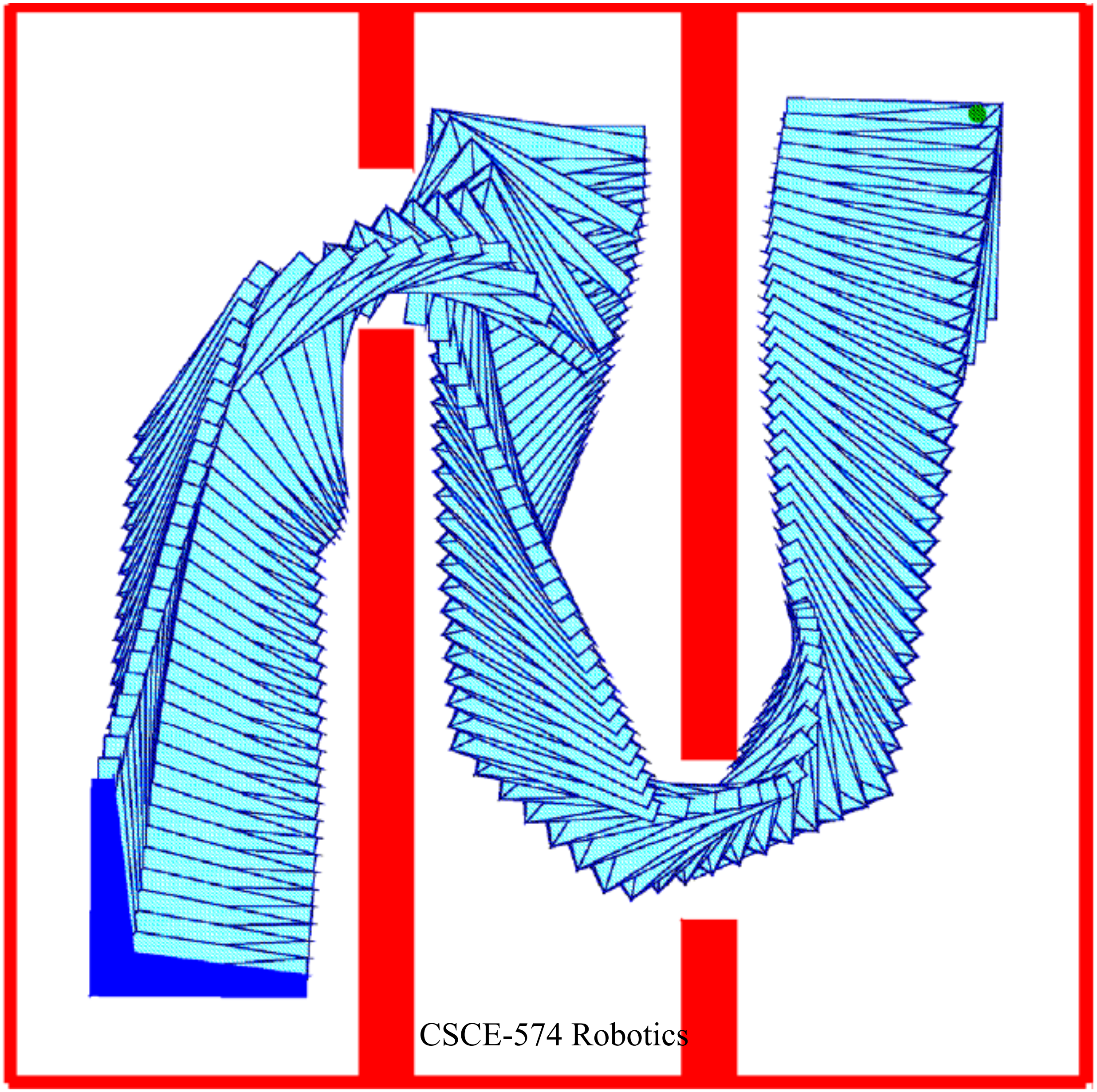
Two link path



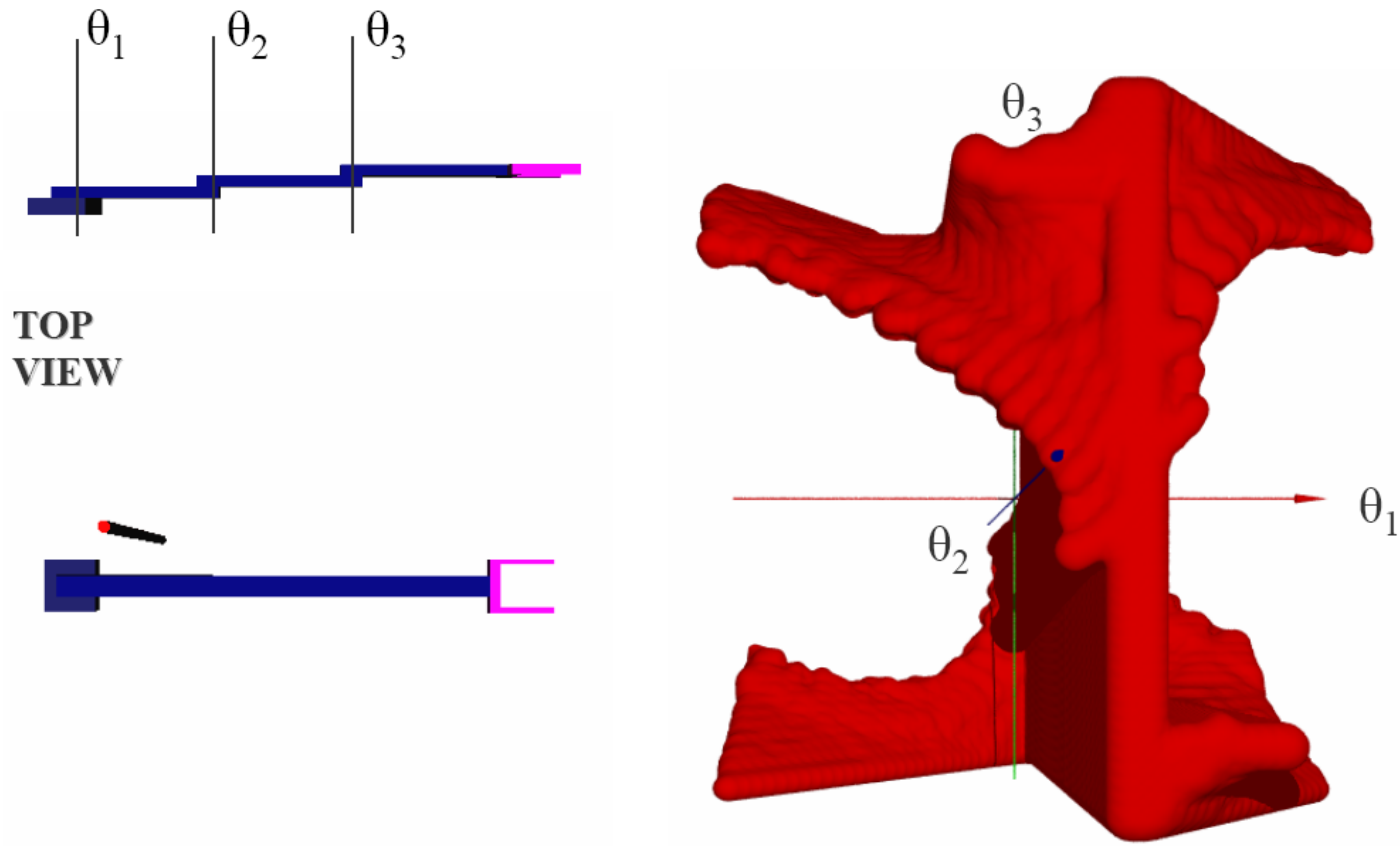


Thanks to Ken Goldberg

2D Rigid Object



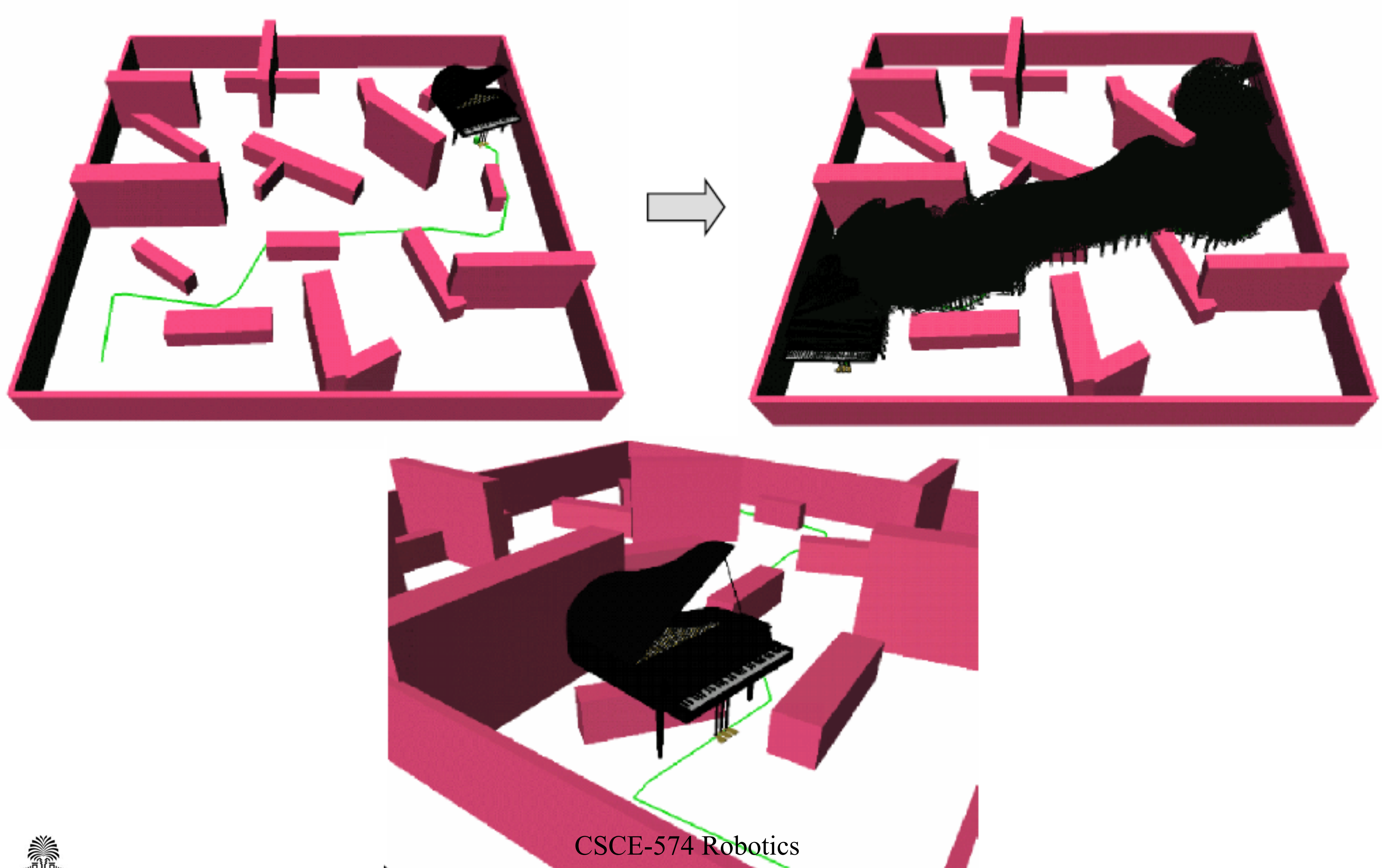
The Configuration Space



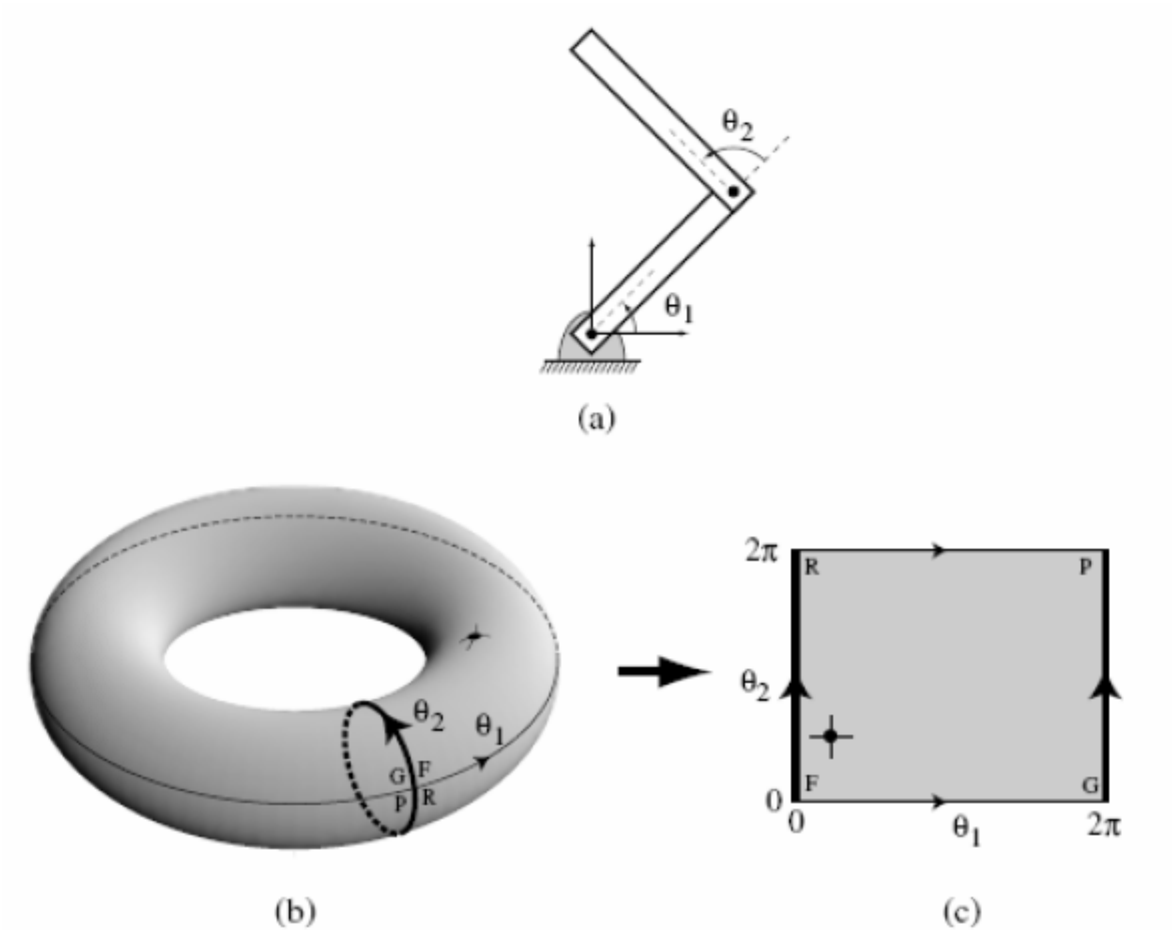
workspace

C-space

Moving a piano



Parameterization of Torus

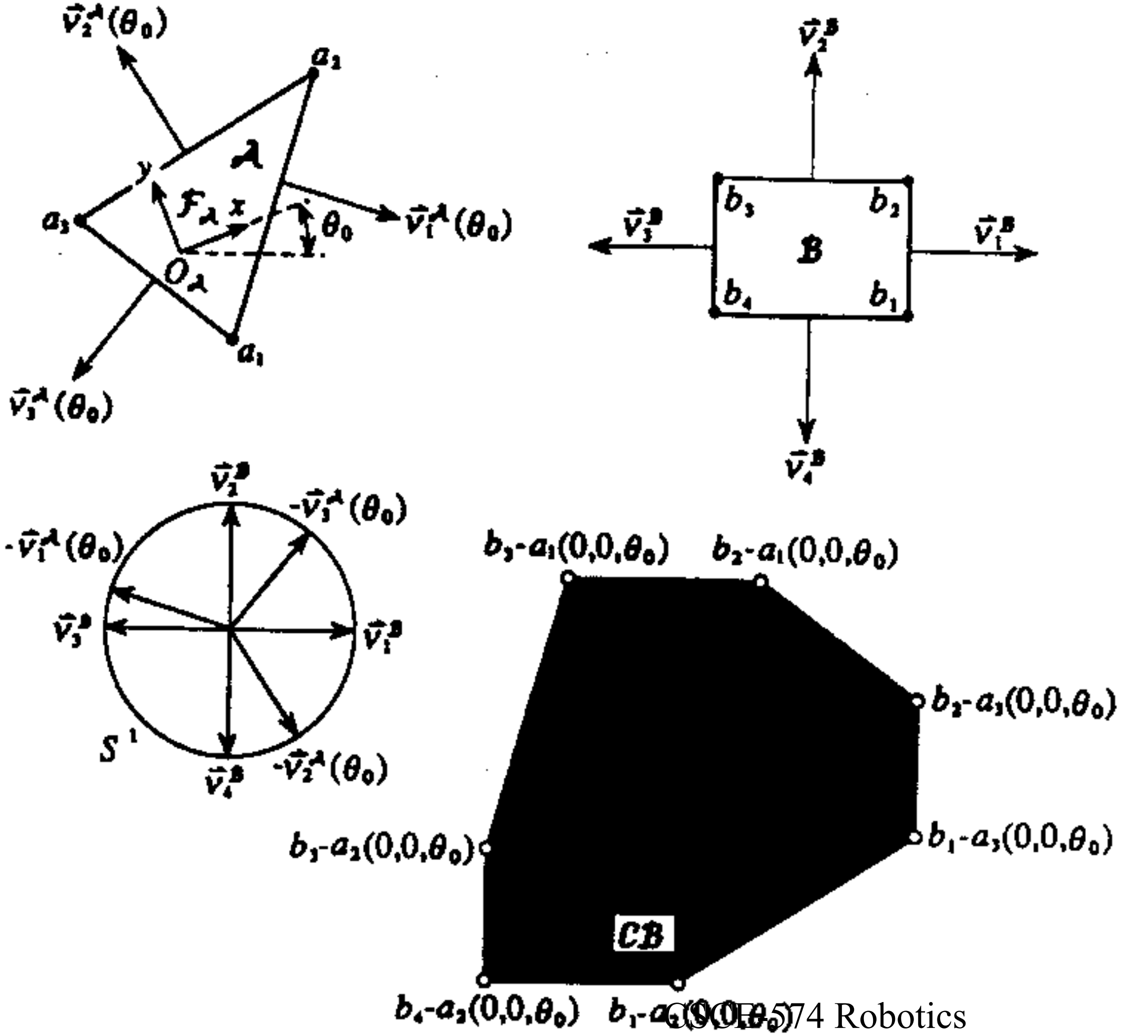


$(\theta_1, \theta_2) \in \mathbb{R}^2,$

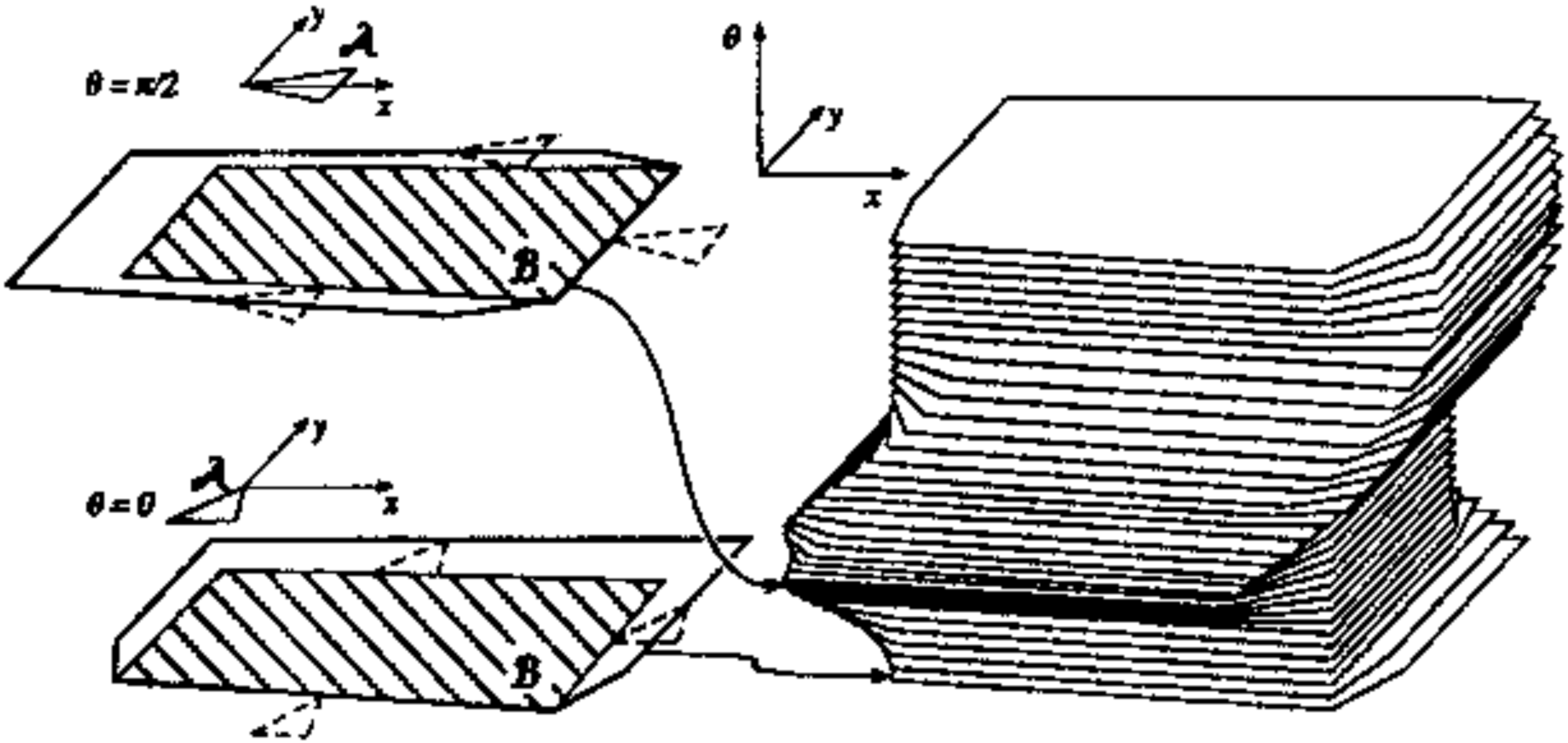
problems at $\theta_i = \{0, 2\pi\}.$

Linear-Time Computation of C-Obstacle in 2-D

(convex polygons)

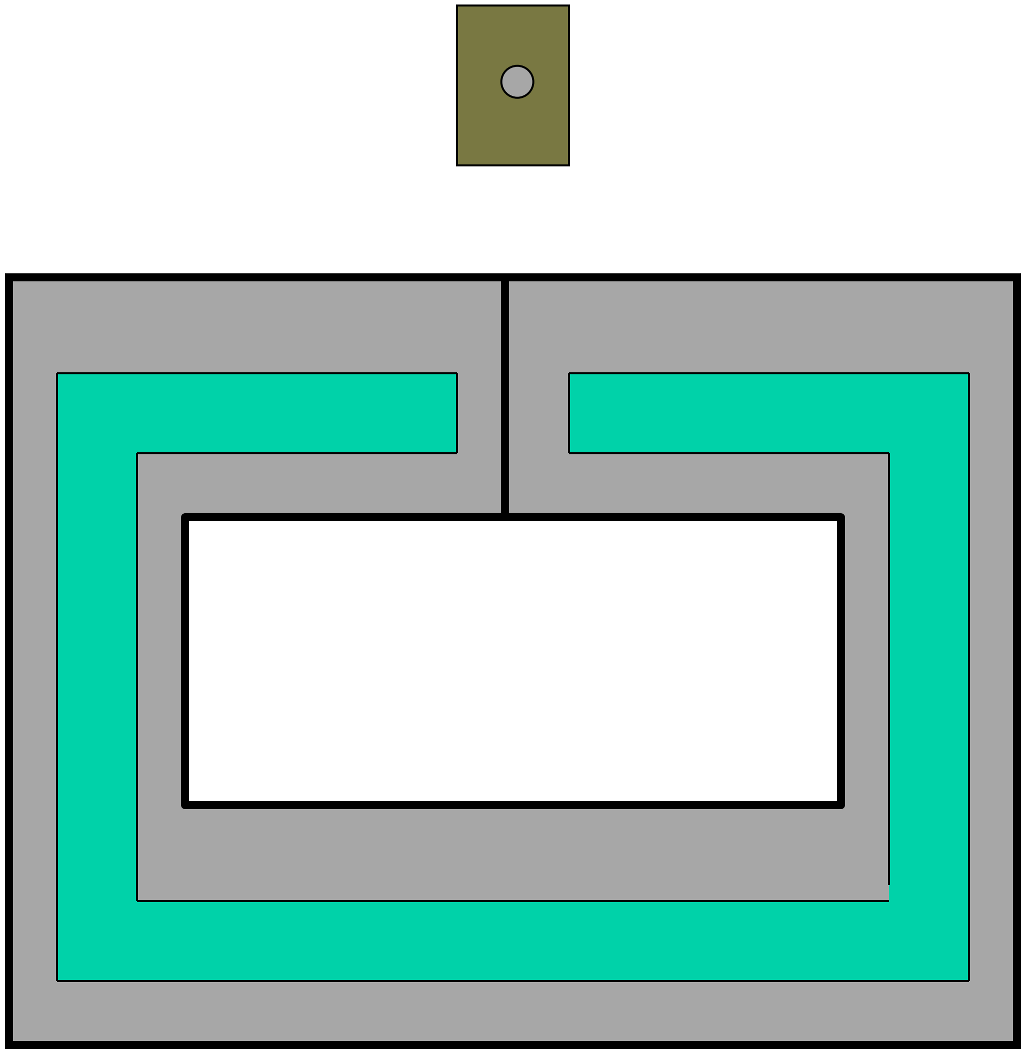


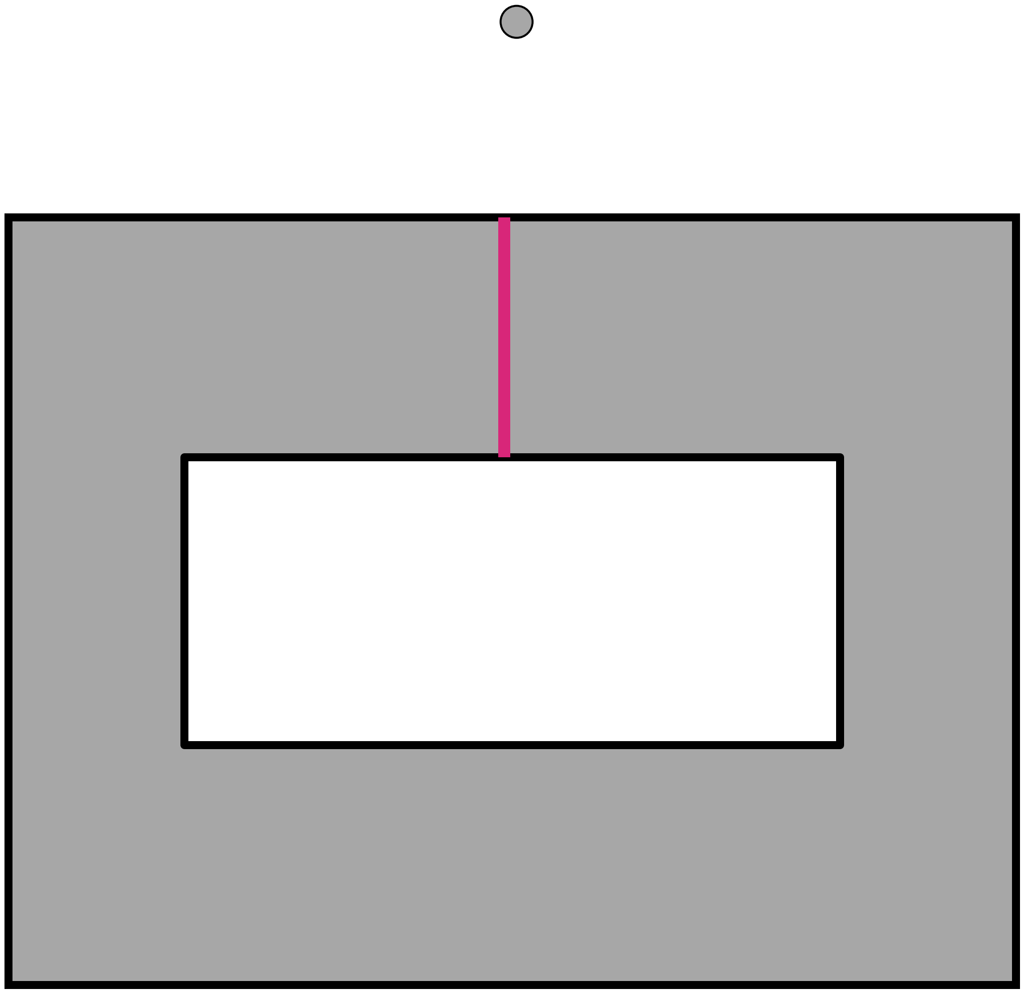
Rigid Robot Translating and Rotating in 2-D



Free and Semi-Free Paths

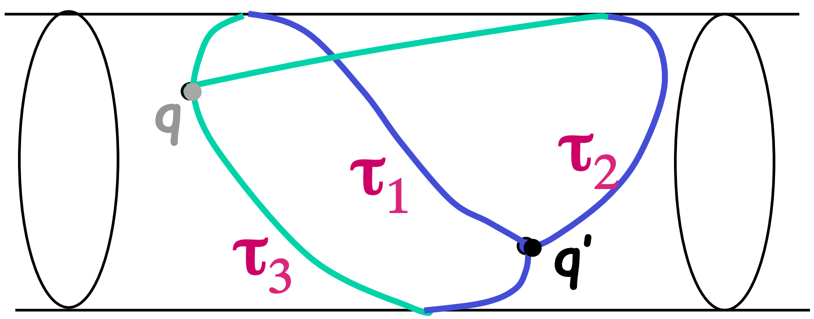
- A free path lies entirely in the free space F
- A semi-free path lies entirely in the semi-free space





Notion of Homotopic Paths

- Two paths with the same endpoints are homotopic if one can be continuously deformed into the other
- R x S^1 example:

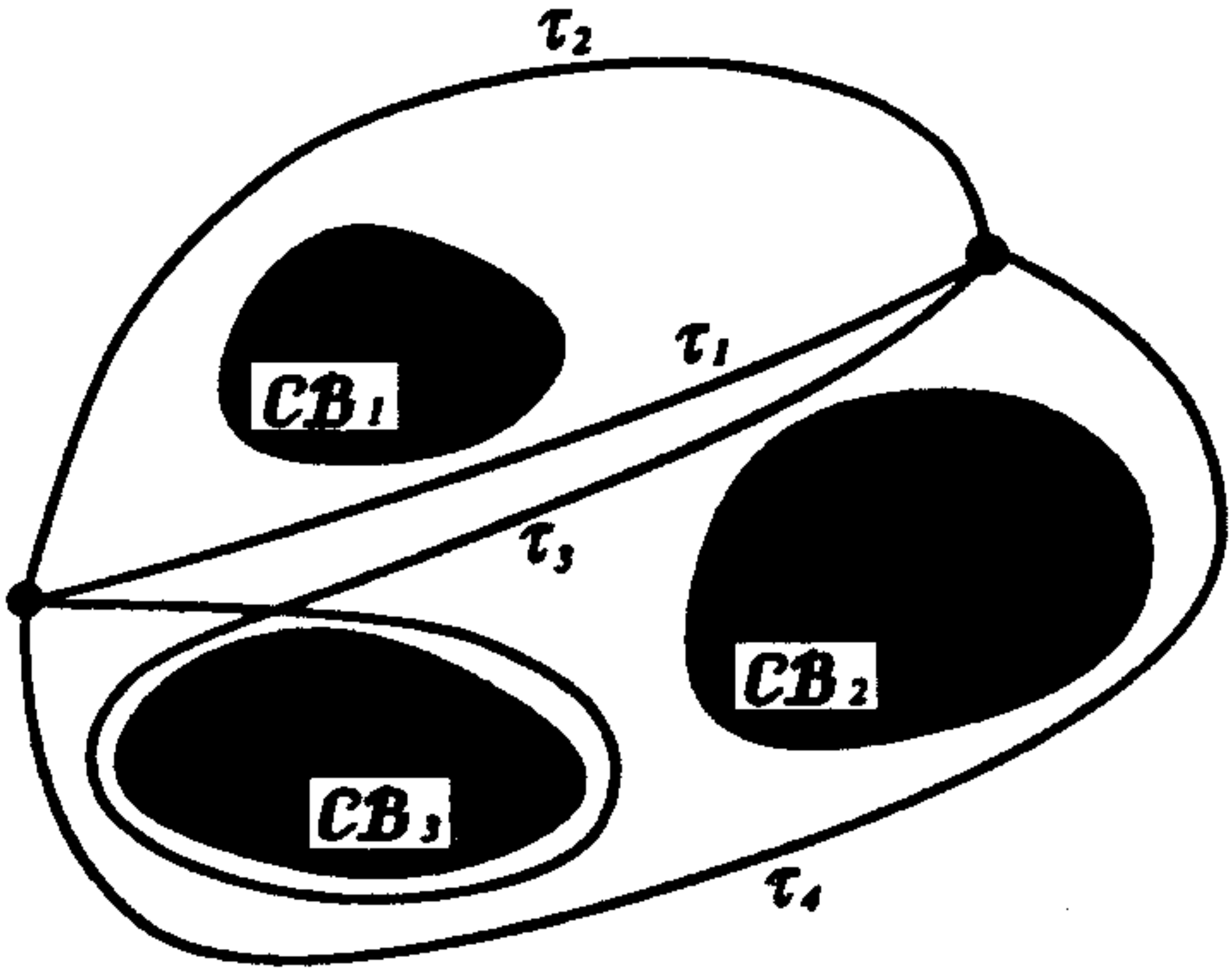


- τ_1 and τ_2 are homotopic
- τ_1 and τ_3 are not homotopic
- In this example, infinity of homotopy classes

Connectedness of C-Space

- C is connected if every two configurations can be connected by a path
- C is simply-connected if any two paths connecting the same endpoints are homotopic
  Examples: $\mathbf{R}^2$ or $\mathbf{R}^3$
- Otherwise C is multiply-connected
  Examples: S^1 and SO(3) are multiply- connected:
  - In S^1, infinity of homotopy classes
  - In SO(3), only two homotopy classes

Classes of Homotopic Free Paths



Probabilistic Roadmaps PRMs

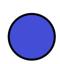
The basic idea behind PRM is to take random samples from the configuration space of the robot, testing them for whether they are in the free space, and use a local planner to attempt to connect these configurations to other nearby configurations. The starting and goal configurations are added in, and a graph search algorithm is applied to the resulting graph to determine a path between the starting and goal configurations.

Kavraki, L. E.; Svestka, P.; Latombe, J.-C.; Overmars, M. H. (1996), "Probabilistic roadmaps for path planning in high-dimensional configuration spaces", IEEE Transactions on Robotics and Automation 12 (4): 566–580.

Rapidly-exploring Random Trees

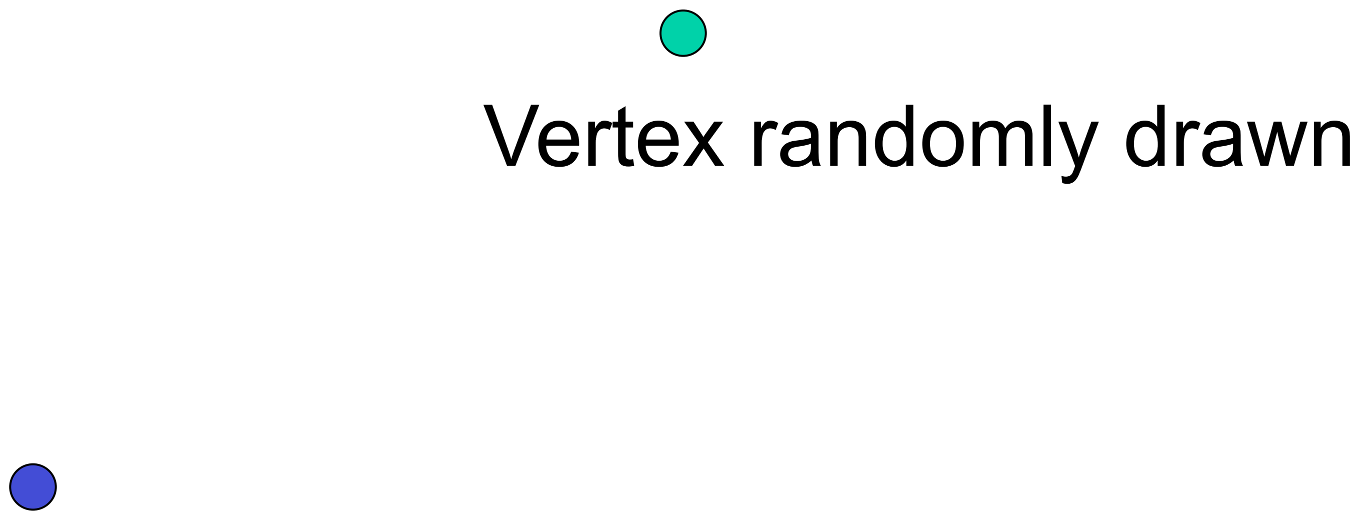
- A point P in C is randomly chosen.
- The nearest vertex in the RRT is selected.
- A new edge is added from this vertex in the direction of P, at distance ε.
- The further the algorithm goes, the more space is covered.

Rapidly-expanding Random Trees



Starting vertex

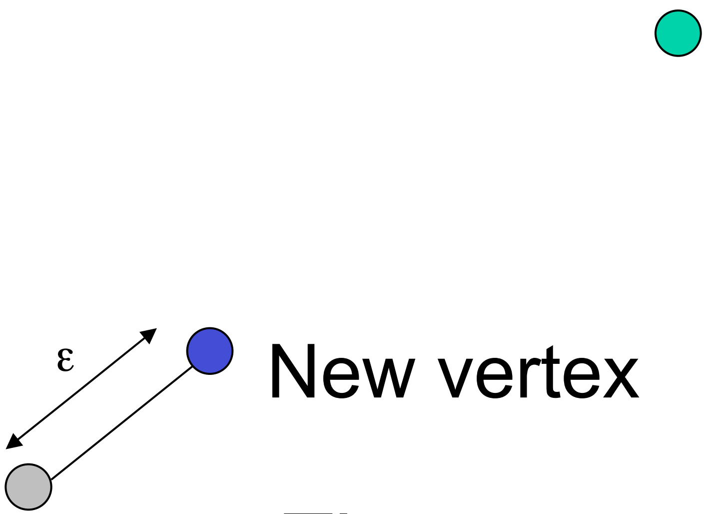
Rapidly-expanding Random Trees



Rapidly-expanding Random Trees

Nearest vertex

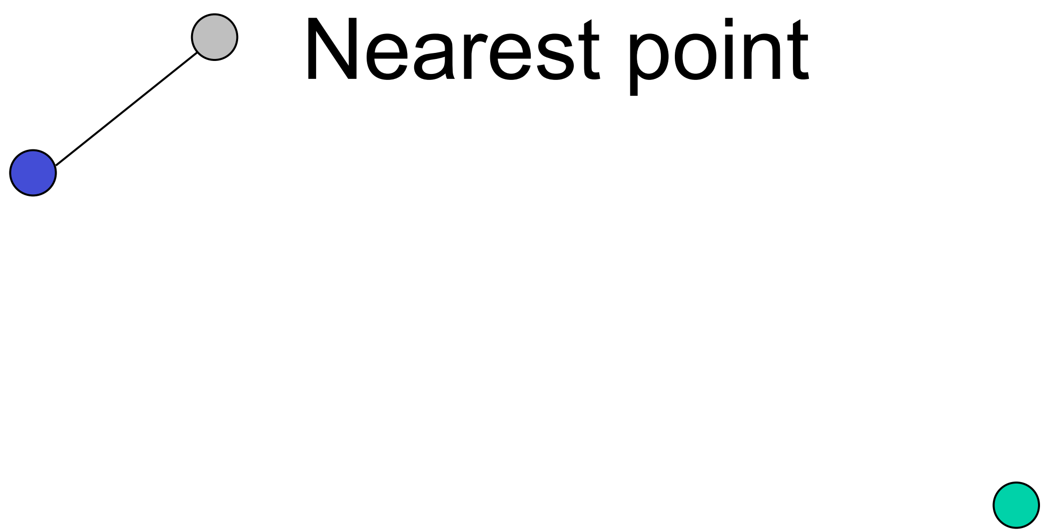
Rapidly-expanding Random Trees



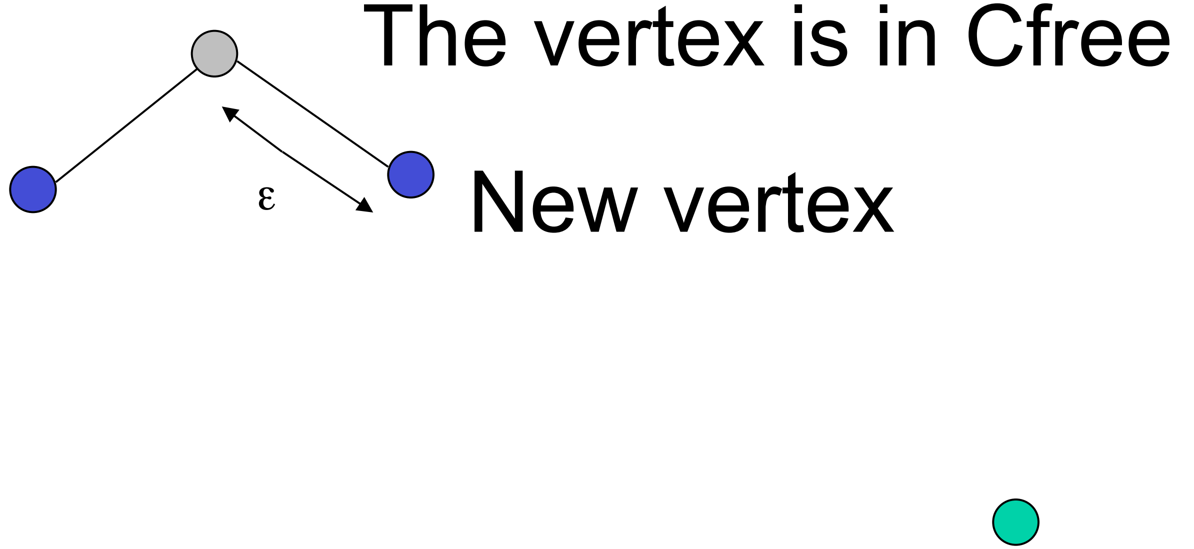
The vertex is in Cfree

Rapidly-expanding Random Trees

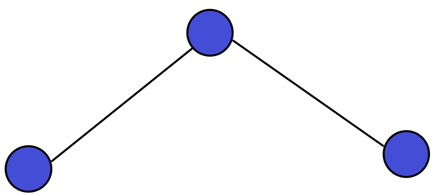
Rapidly-expanding Random Trees



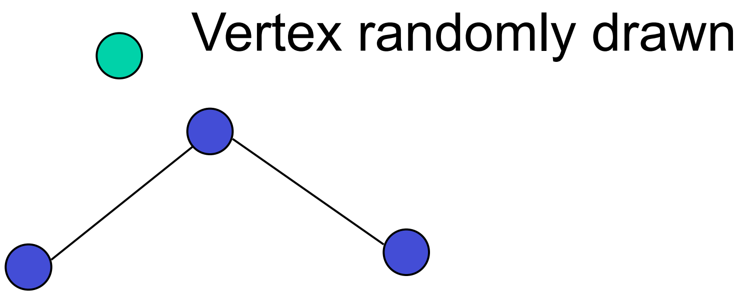
Rapidly-expanding Random Trees



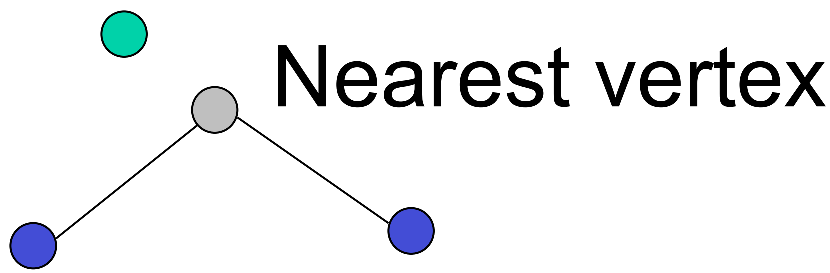
Rapidly-expanding Random Trees



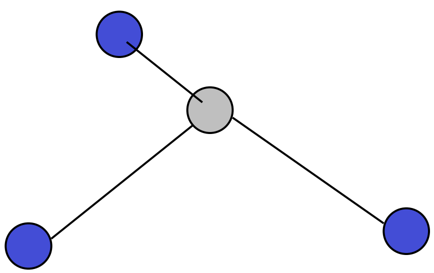
Rapidly-expanding Random Trees



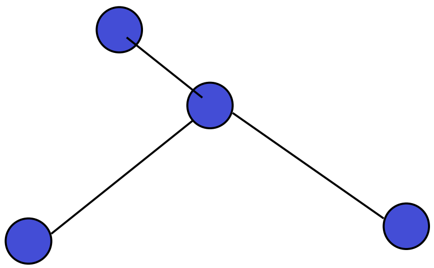
Rapidly-expanding Random Trees



Rapidly-expanding Random Trees



Rapidly-expanding Random Trees

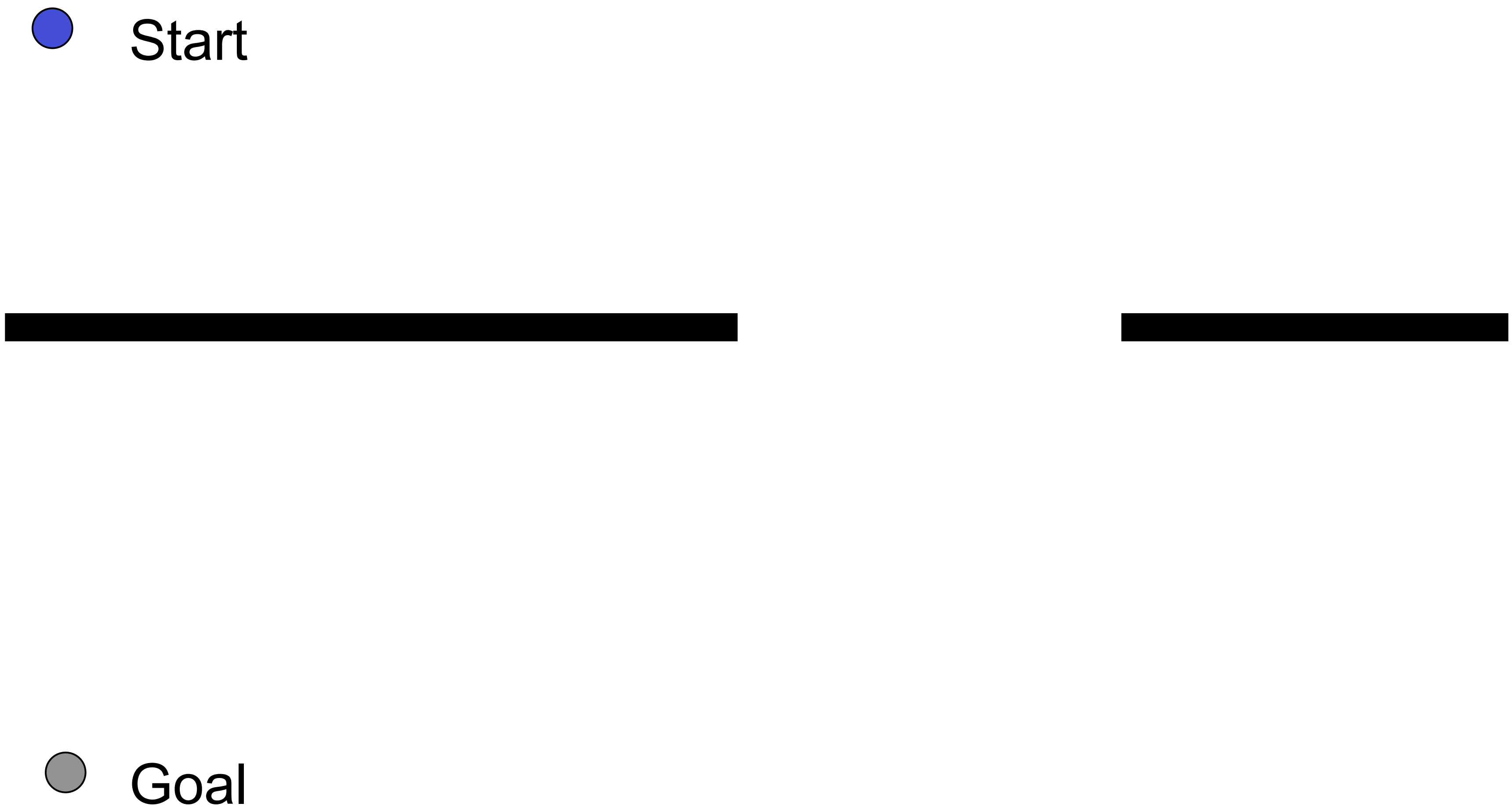


And it continues…

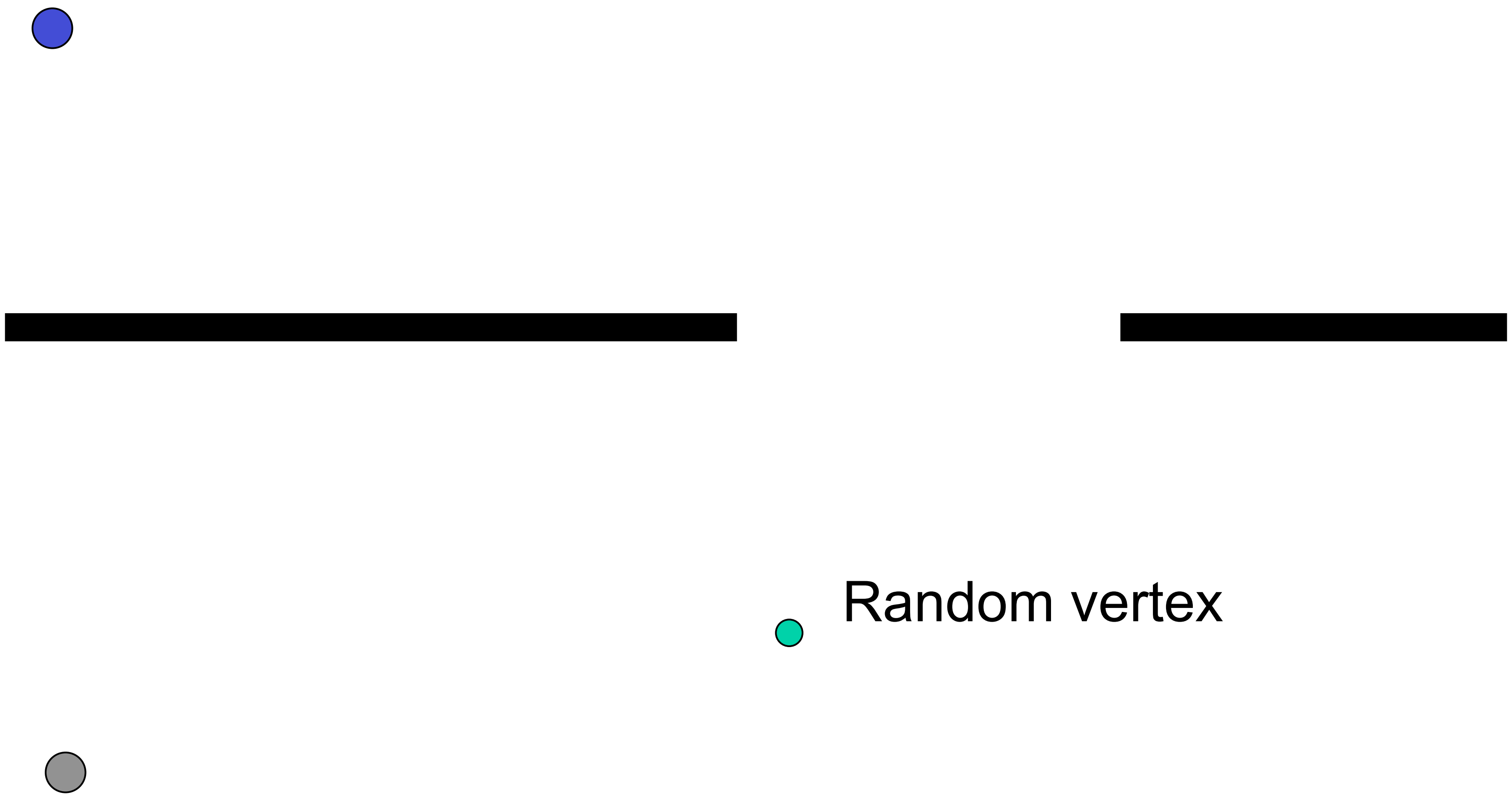
RRT-Connect

- We grow two trees, one from the beginning vertex and another from the end vertex
- Each time we create a new vertex, we try to greedily connect the two trees

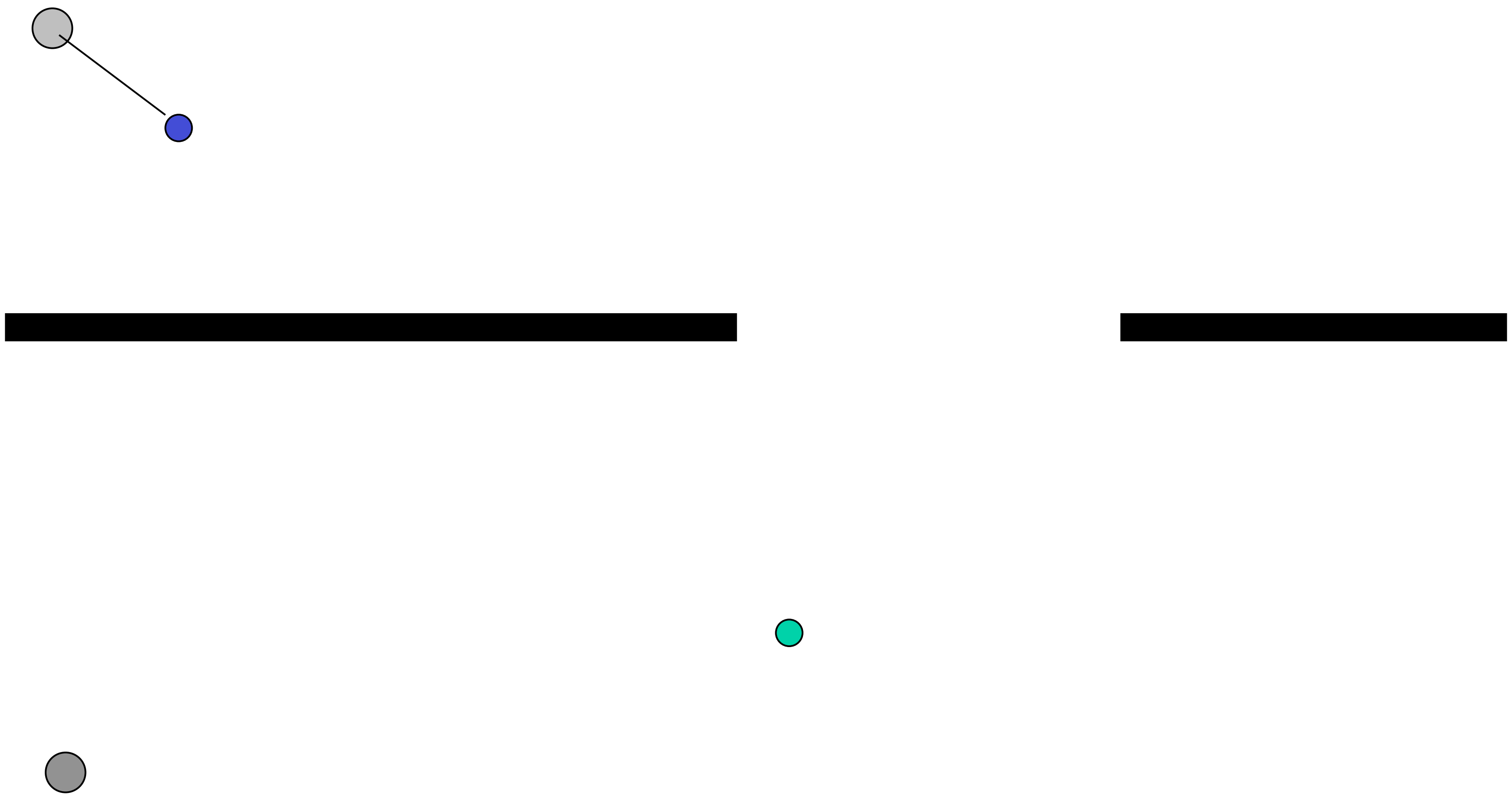
RRT-Connect: example



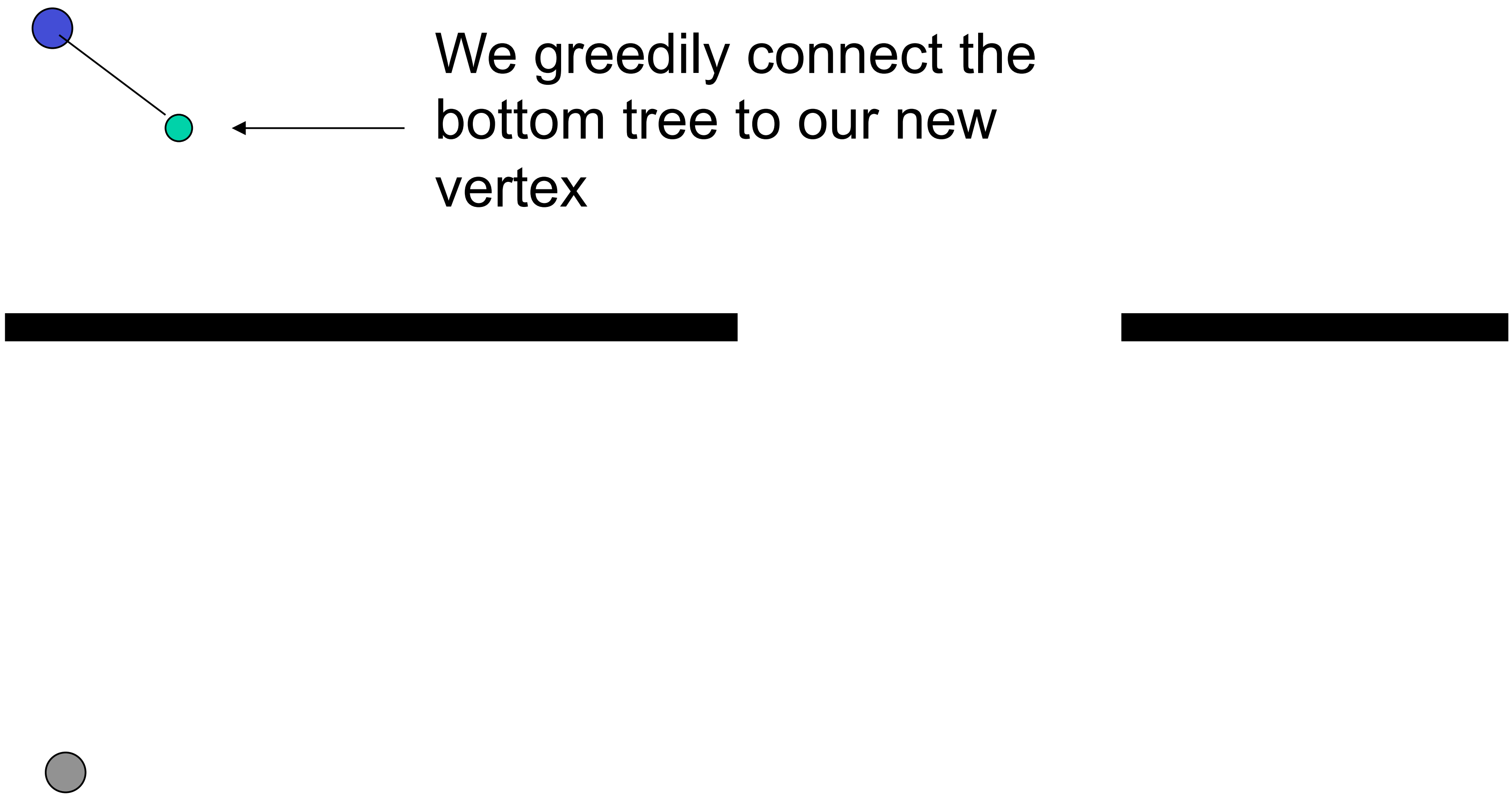
RRT-Connect: example



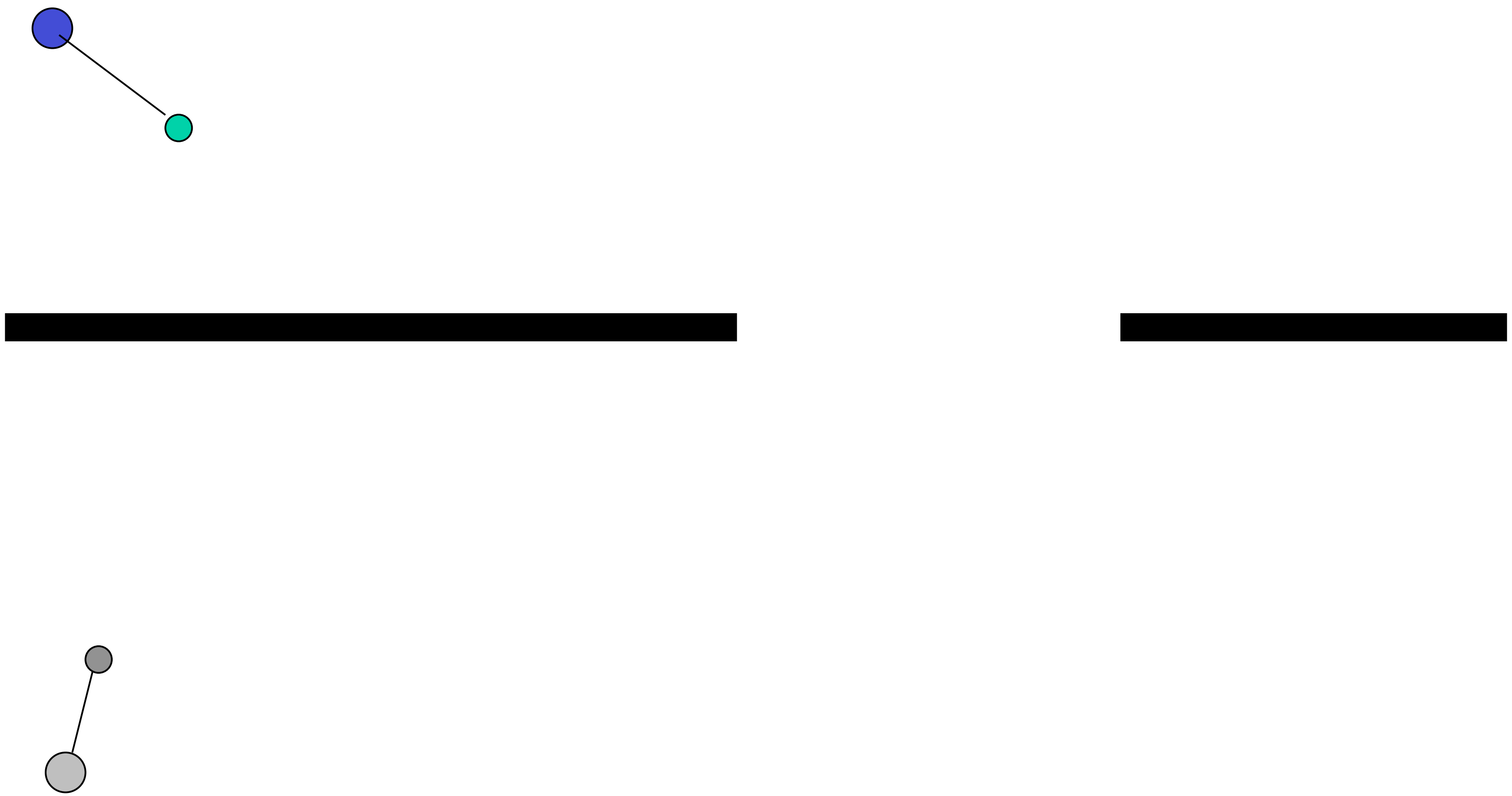
RRT-Connect: example



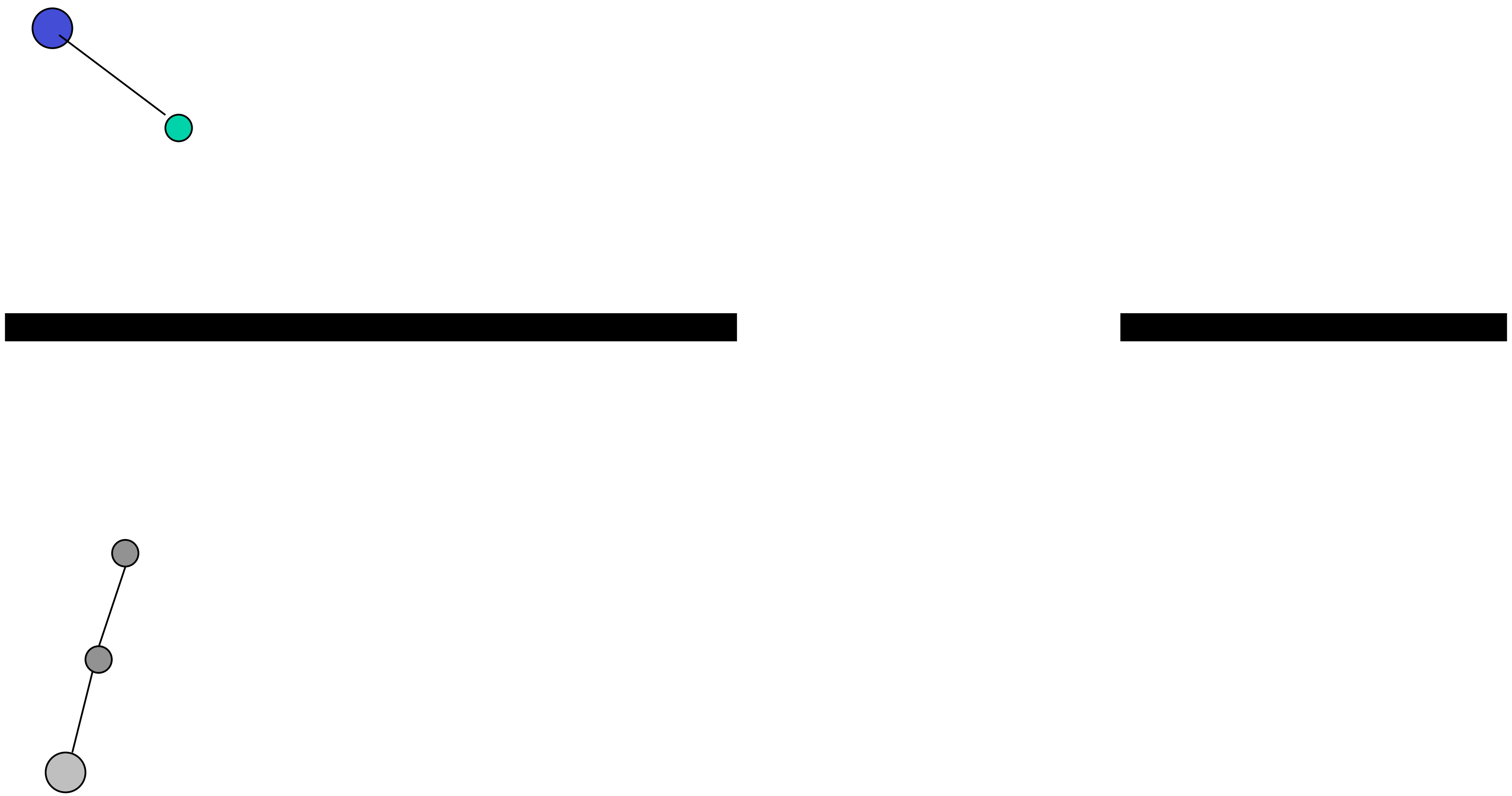
RRT-Connect: example



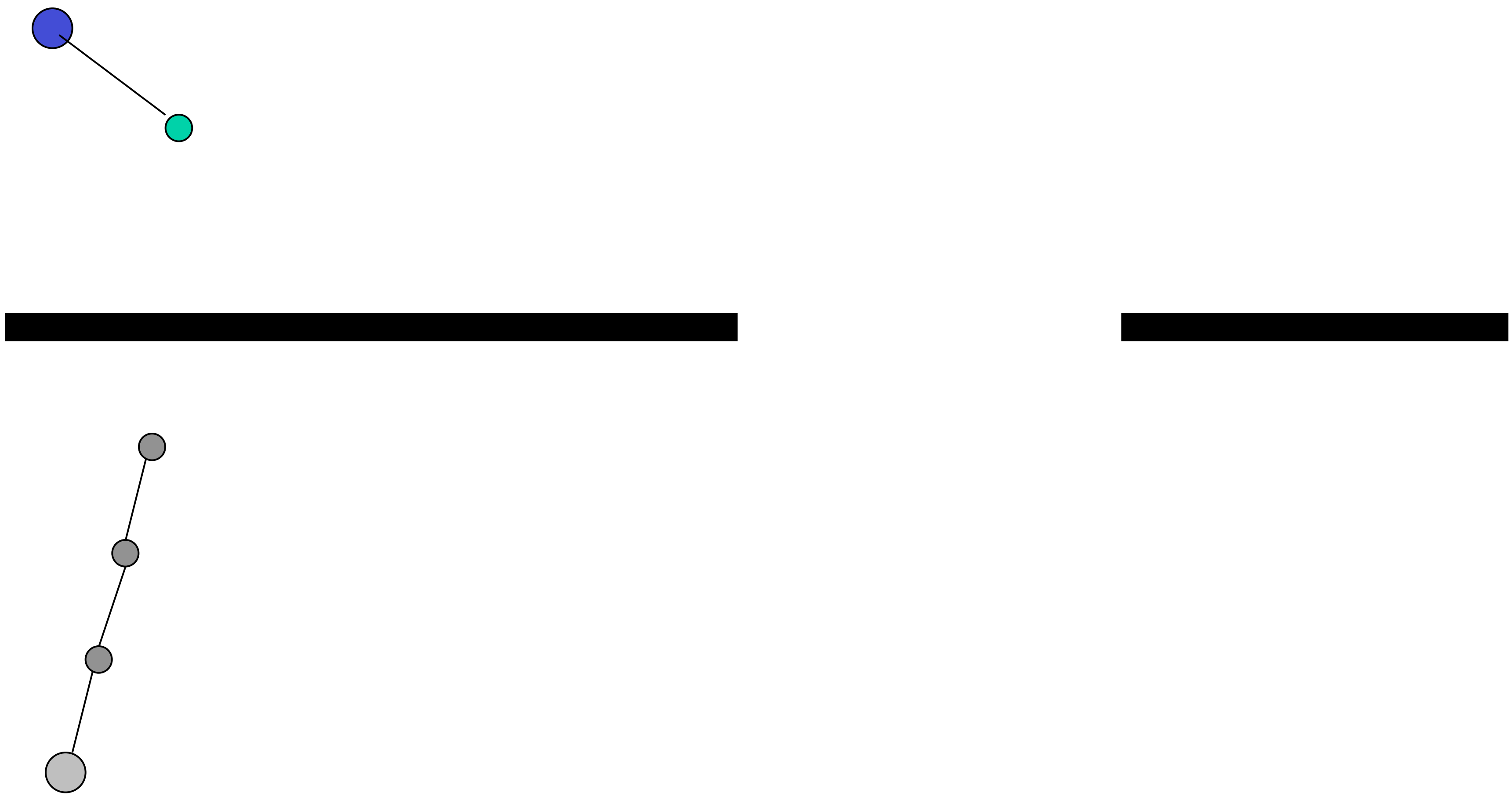
RRT-Connect: example



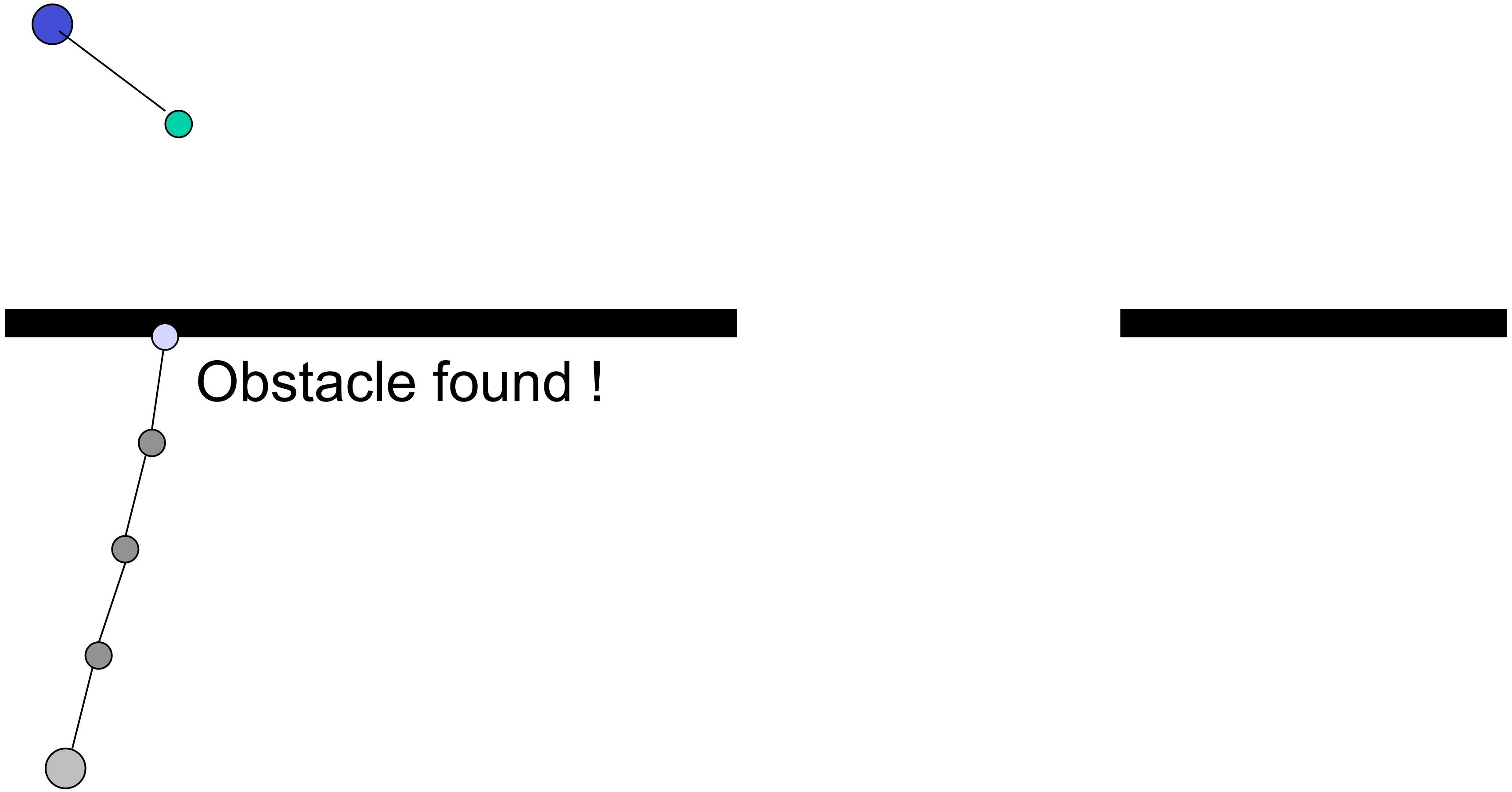
RRT-Connect: example



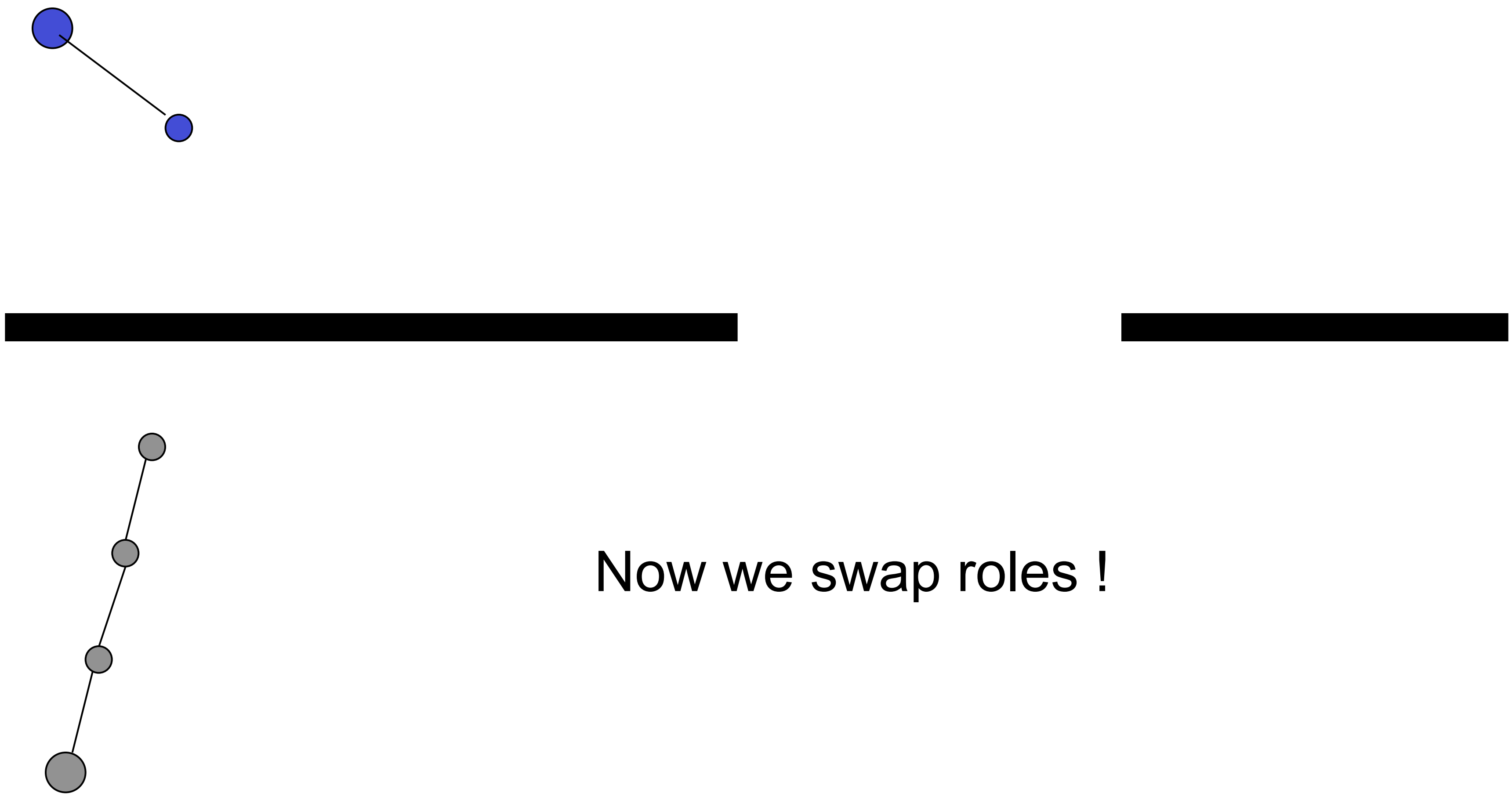
RRT-Connect: example



RRT-Connect: example

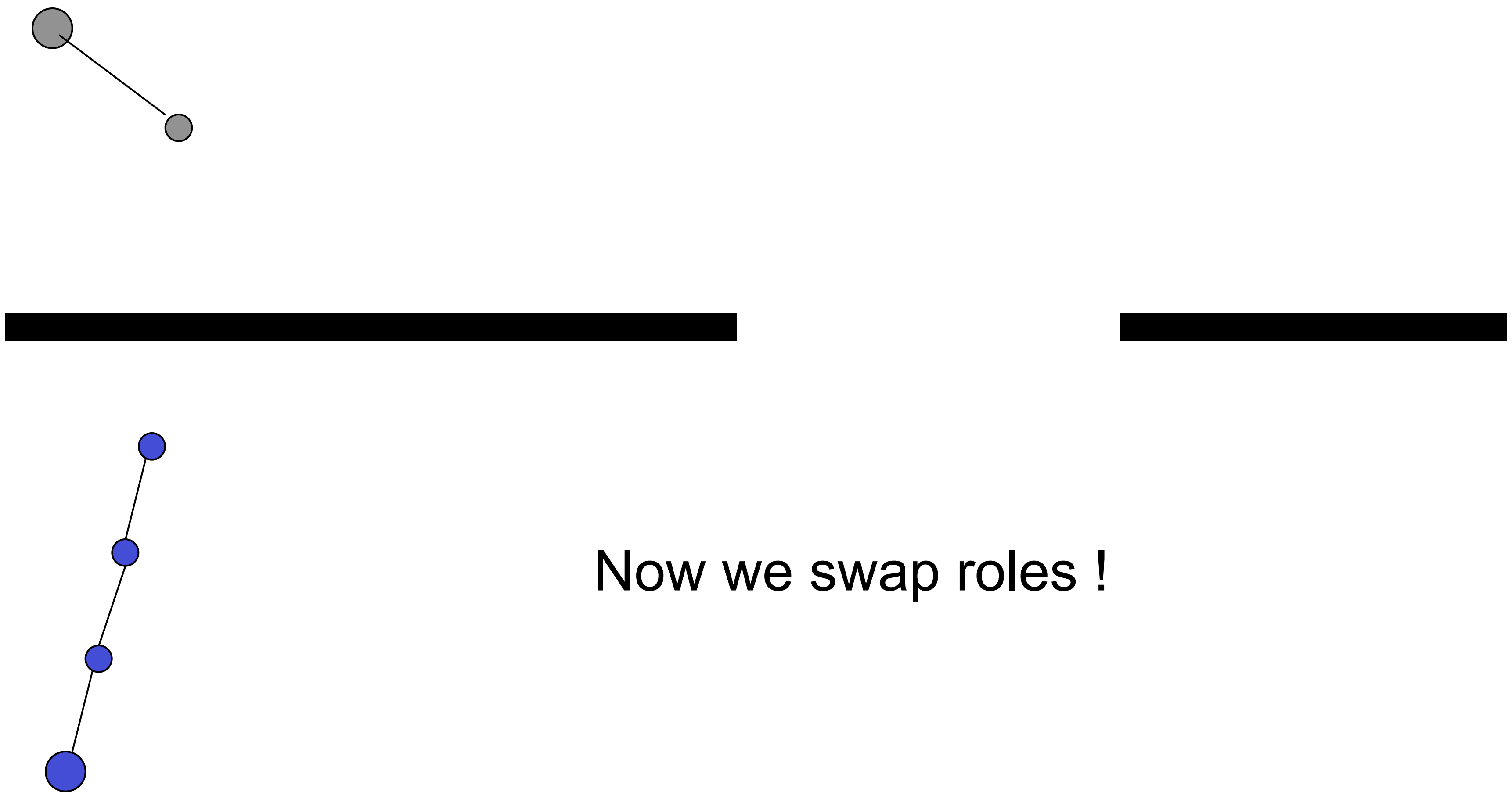


RRT-Connect: example



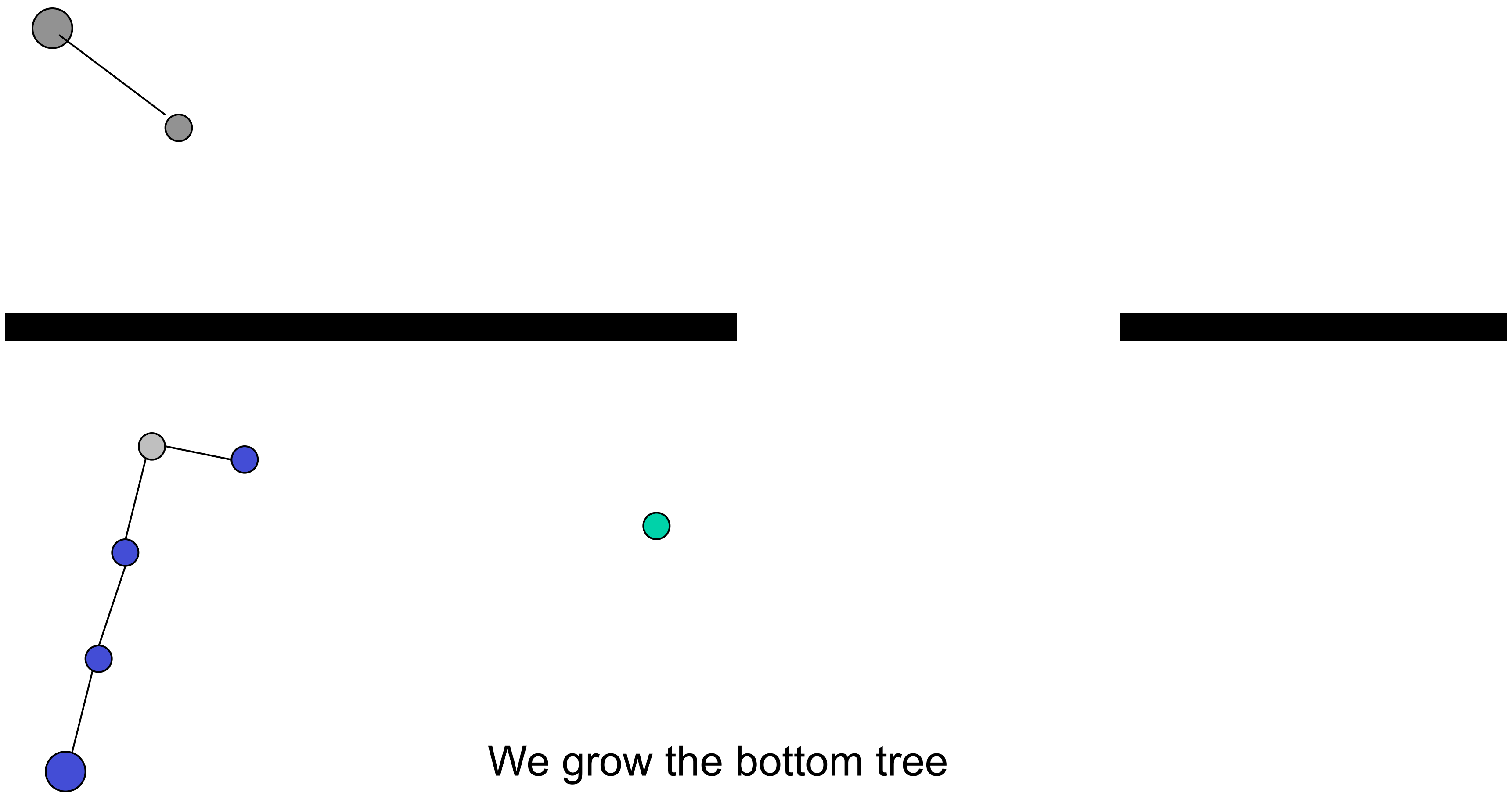
Now we swap roles !

RRT-Connect: example



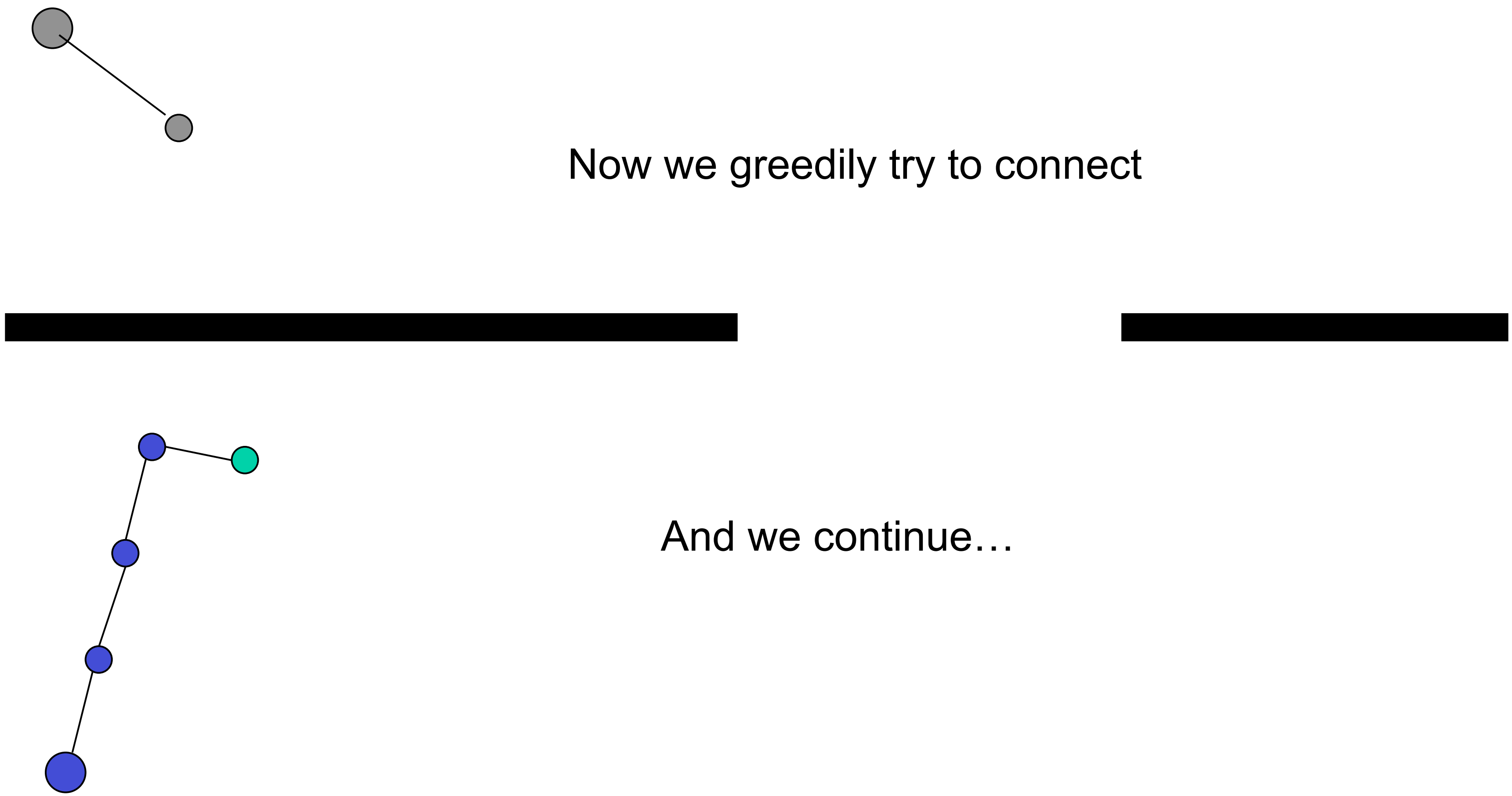
Now we swap roles !

RRT-Connect: example



We grow the bottom tree

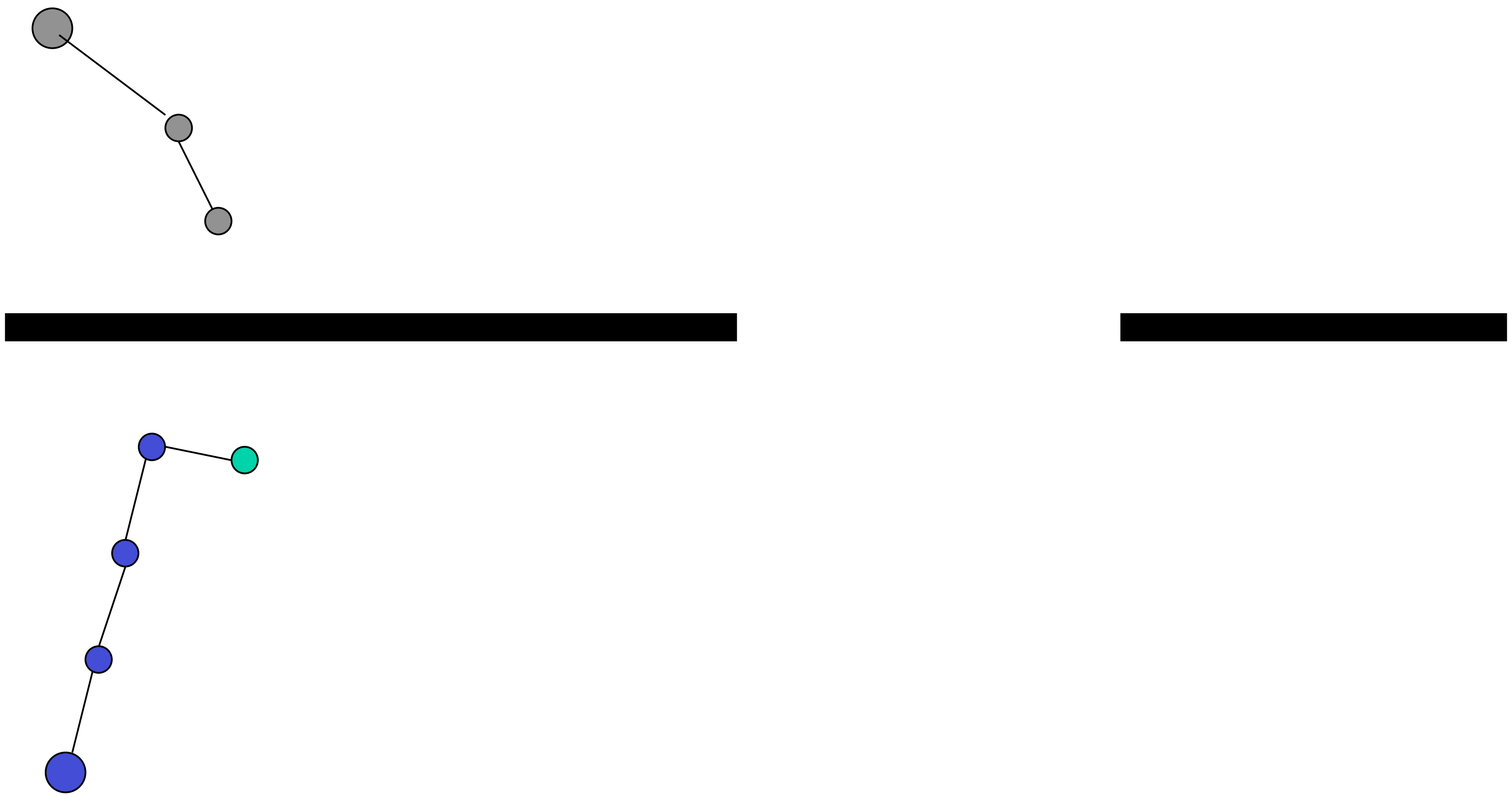
RRT-Connect: example



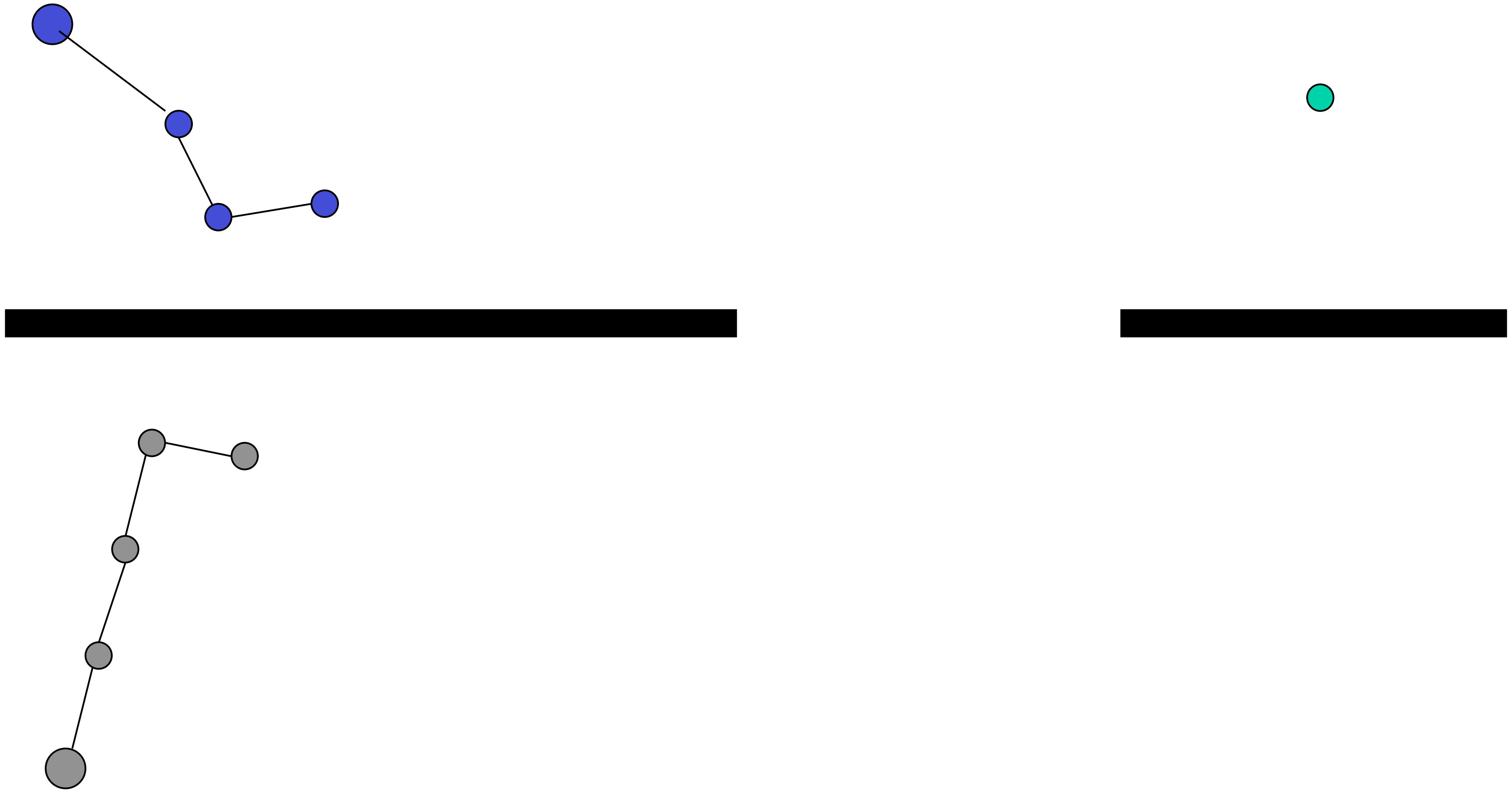
Now we greedily try to connect

And we continue…

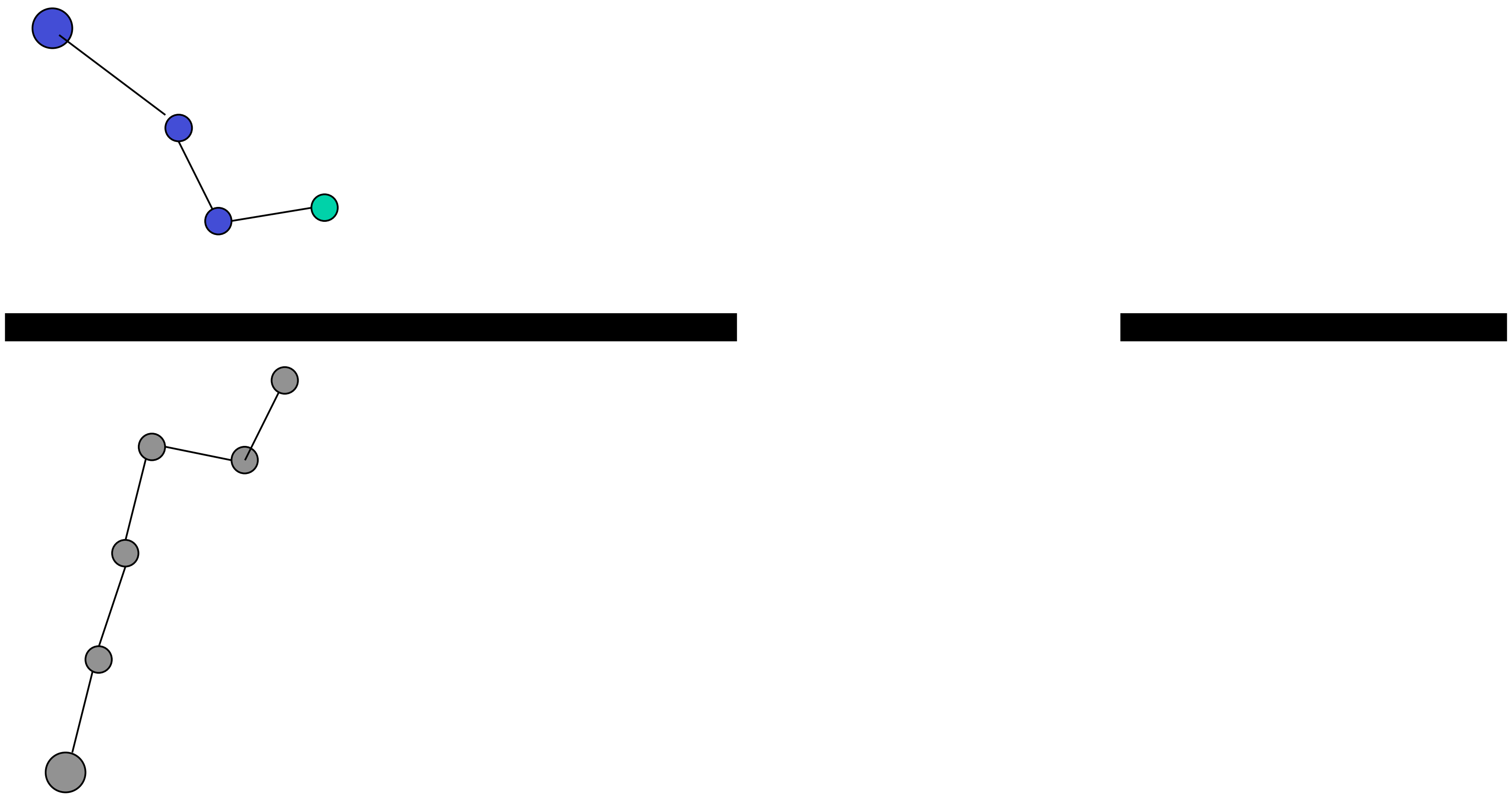
RRT-Connect: example



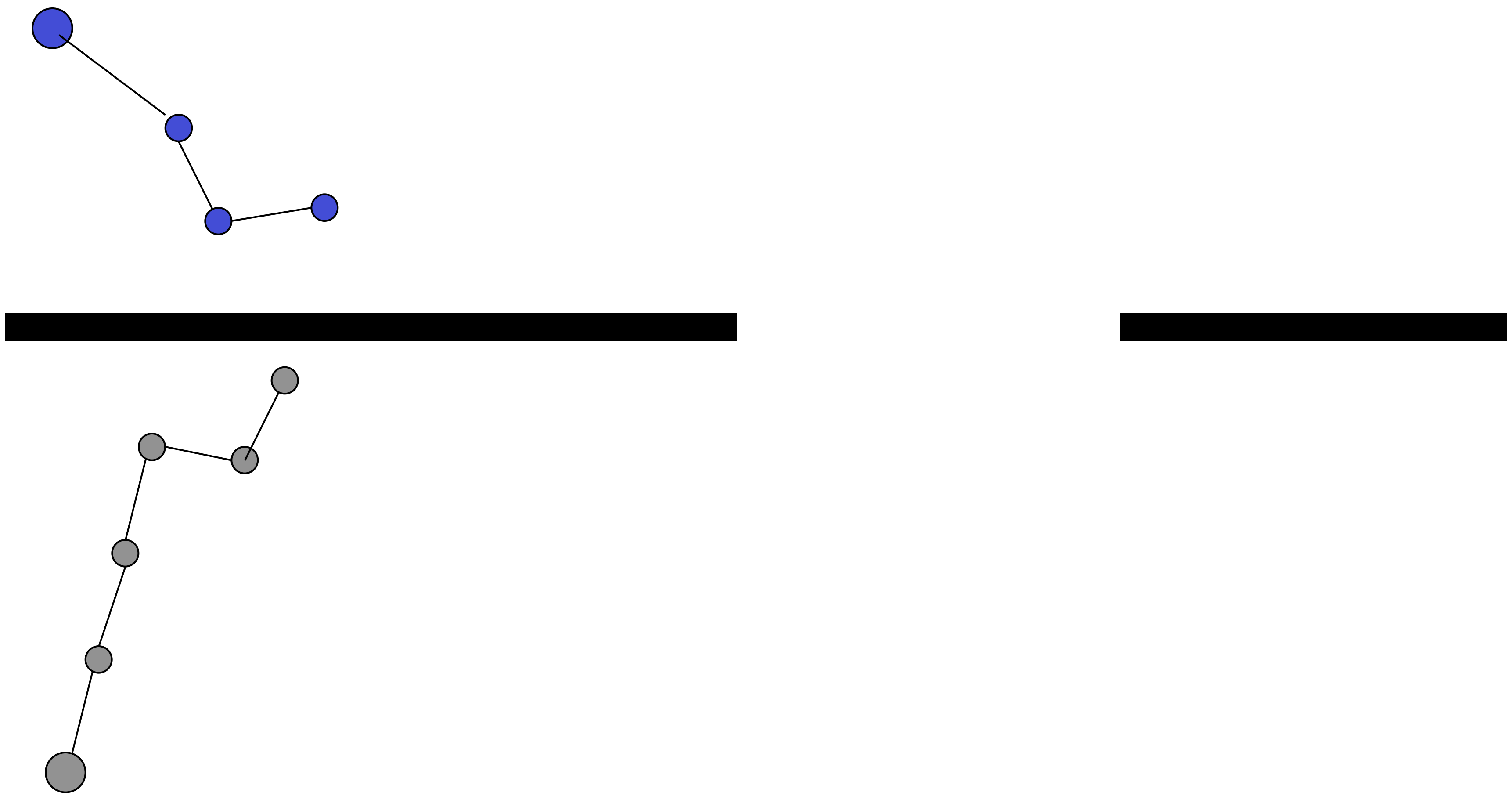
RRT-Connect: example



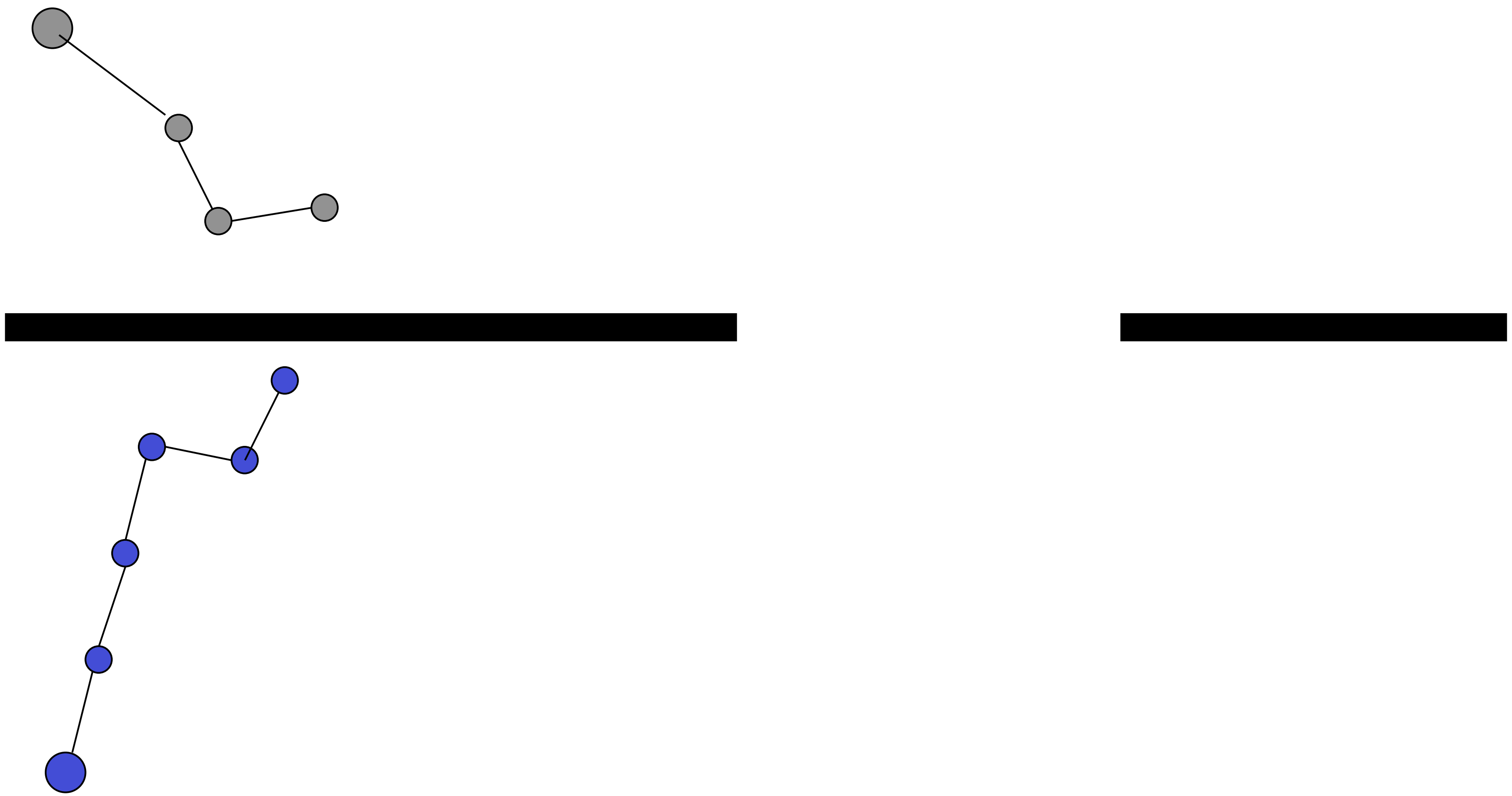
RRT-Connect: example



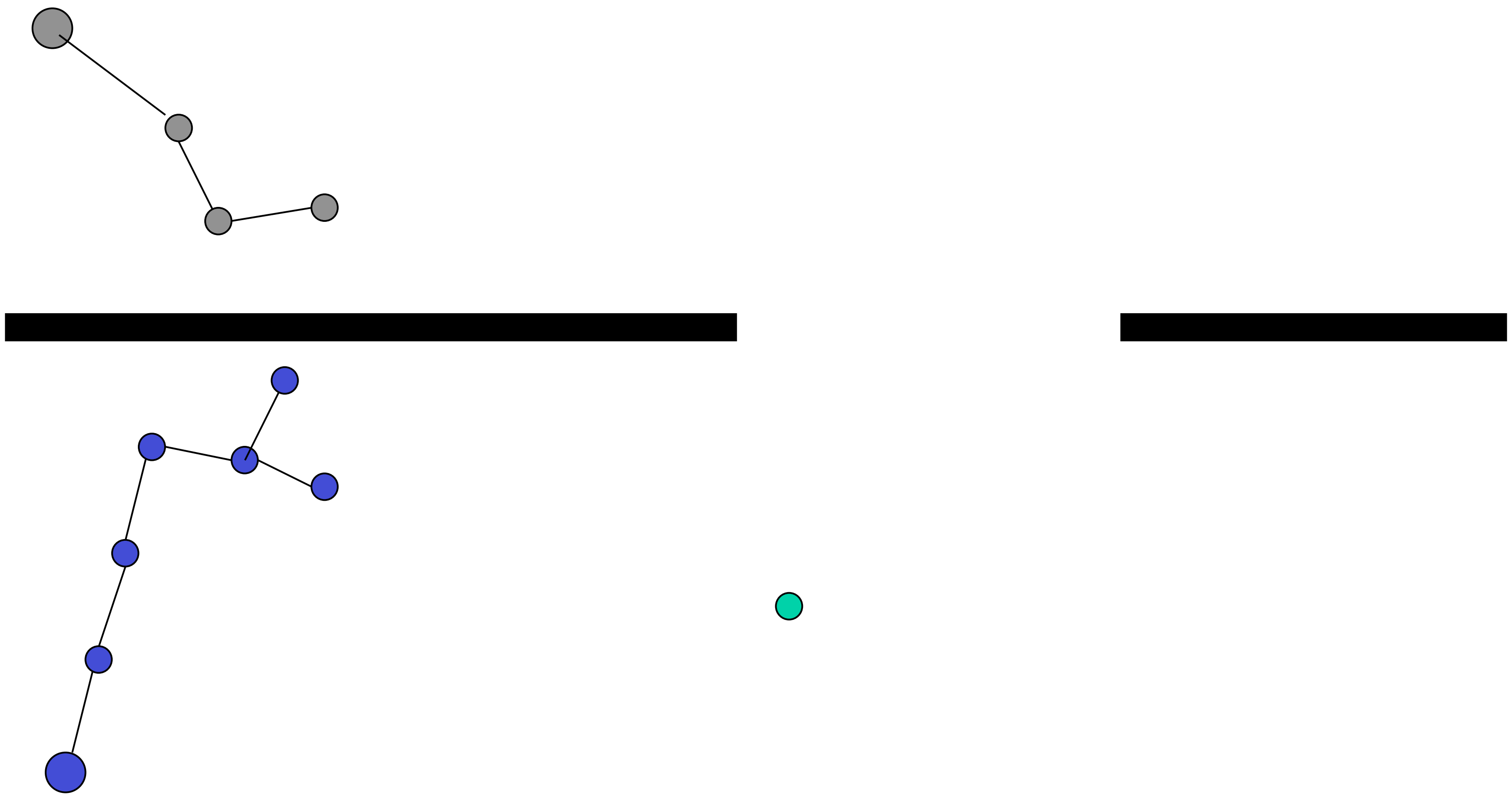
RRT-Connect: example



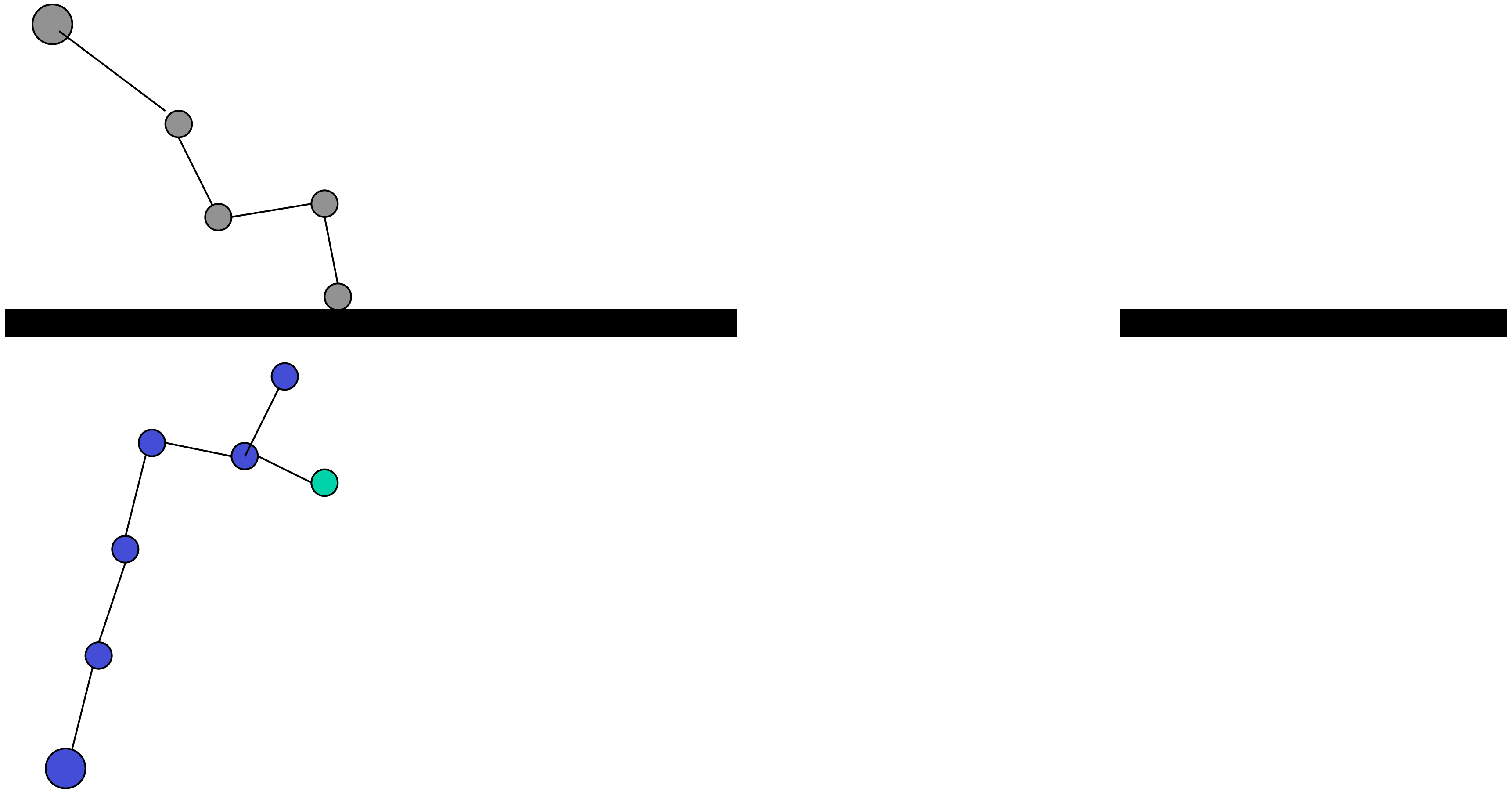
RRT-Connect: example



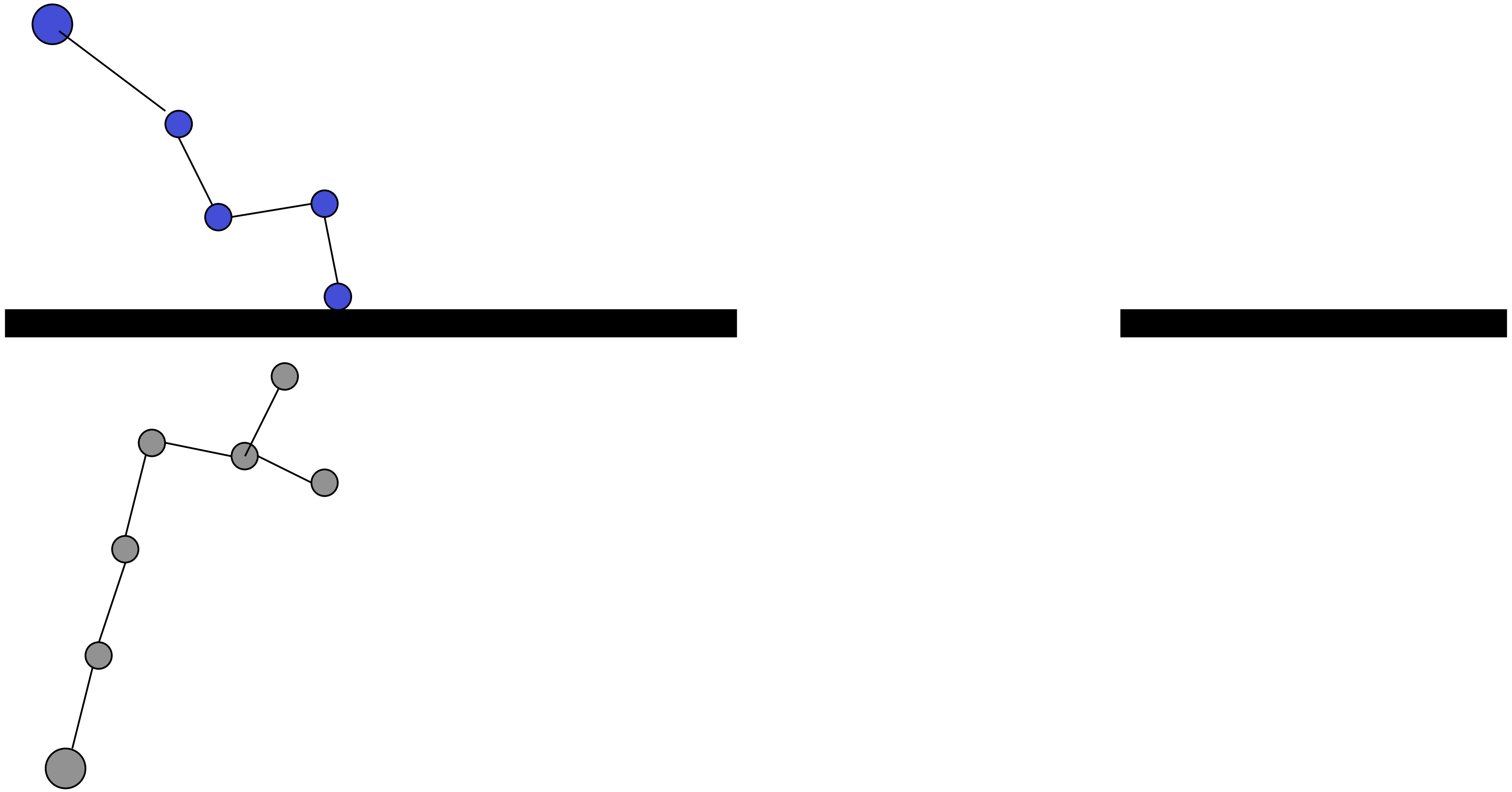
RRT-Connect: example



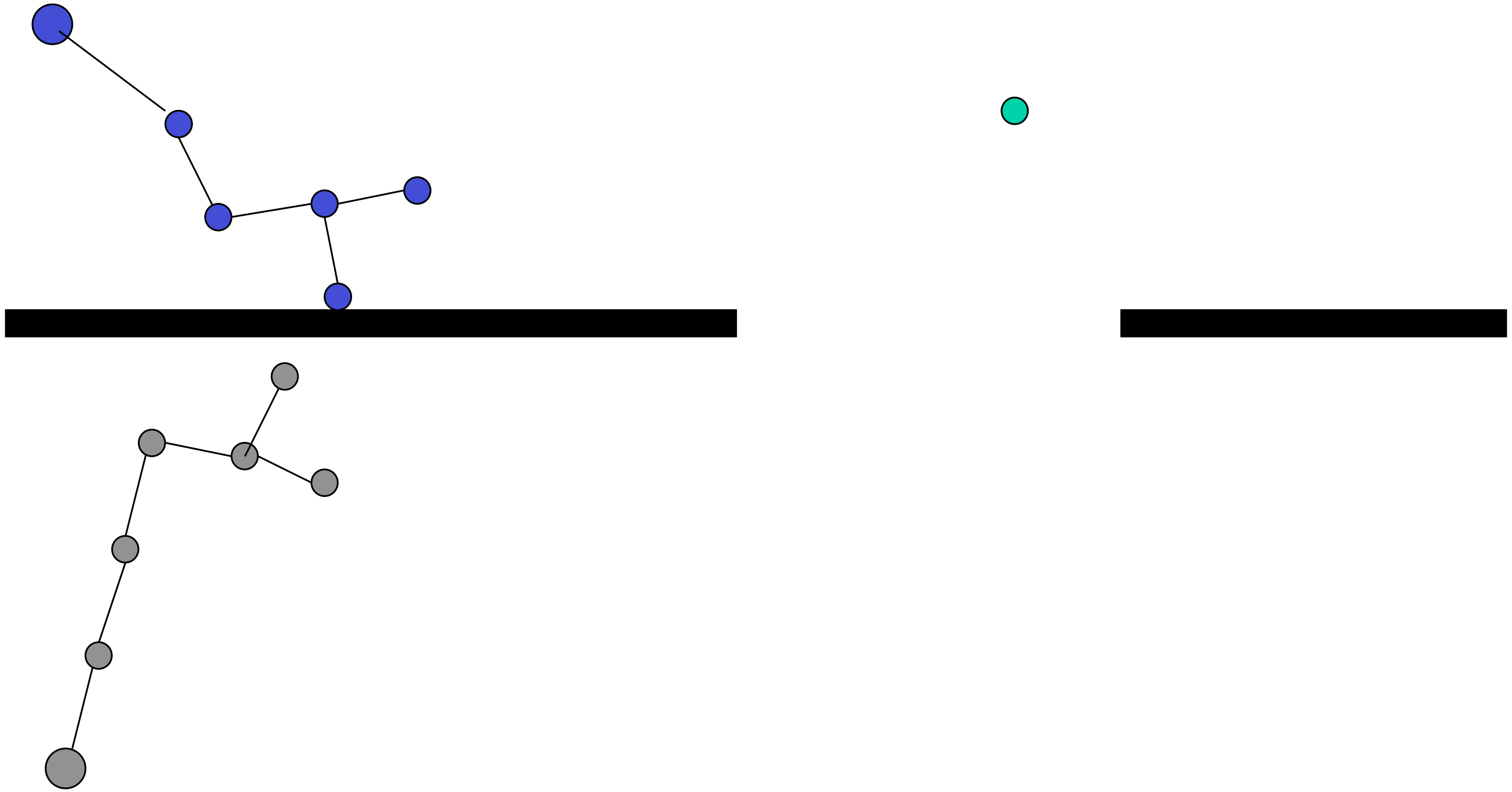
RRT-Connect: example



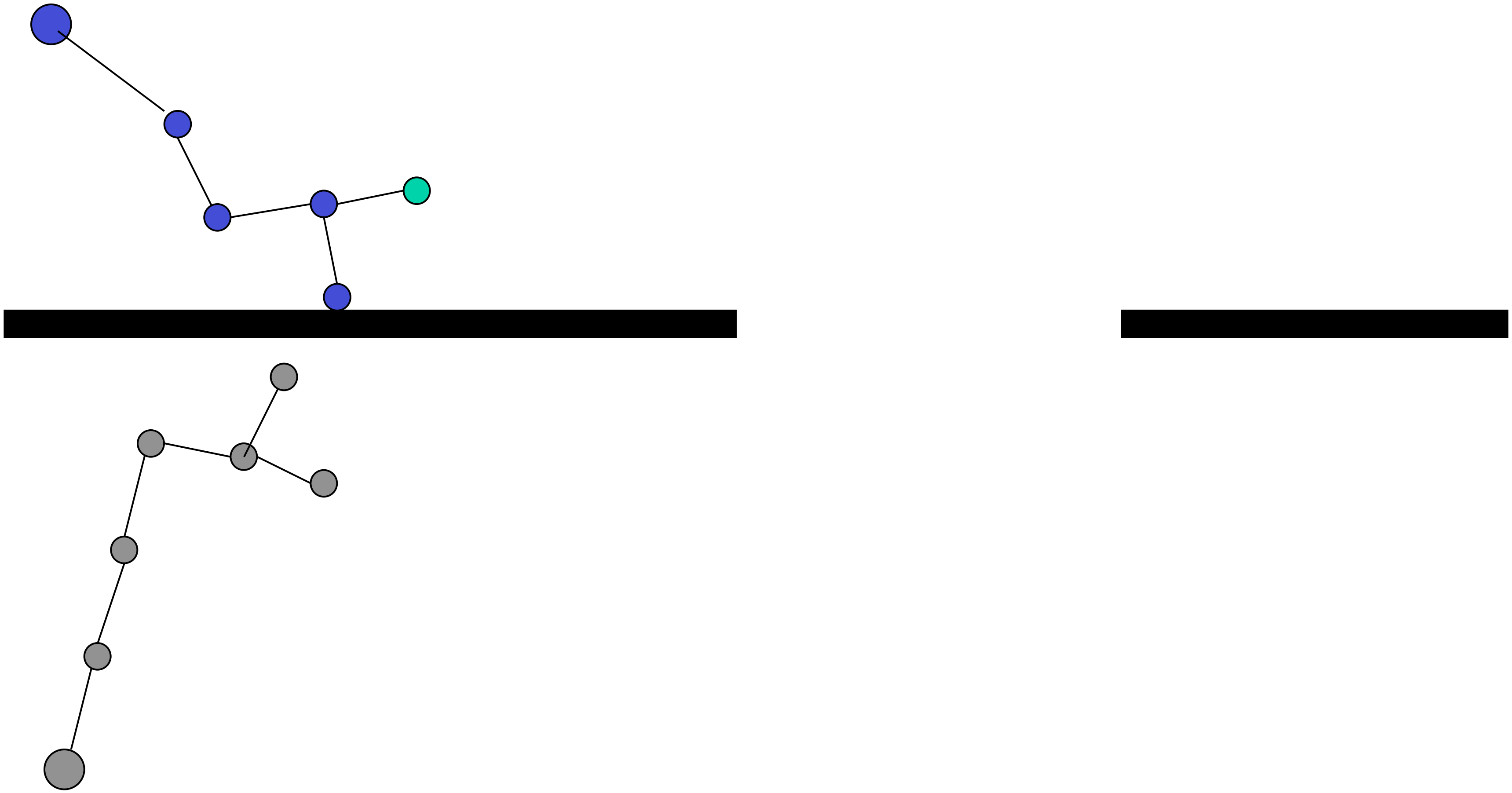
RRT-Connect: example



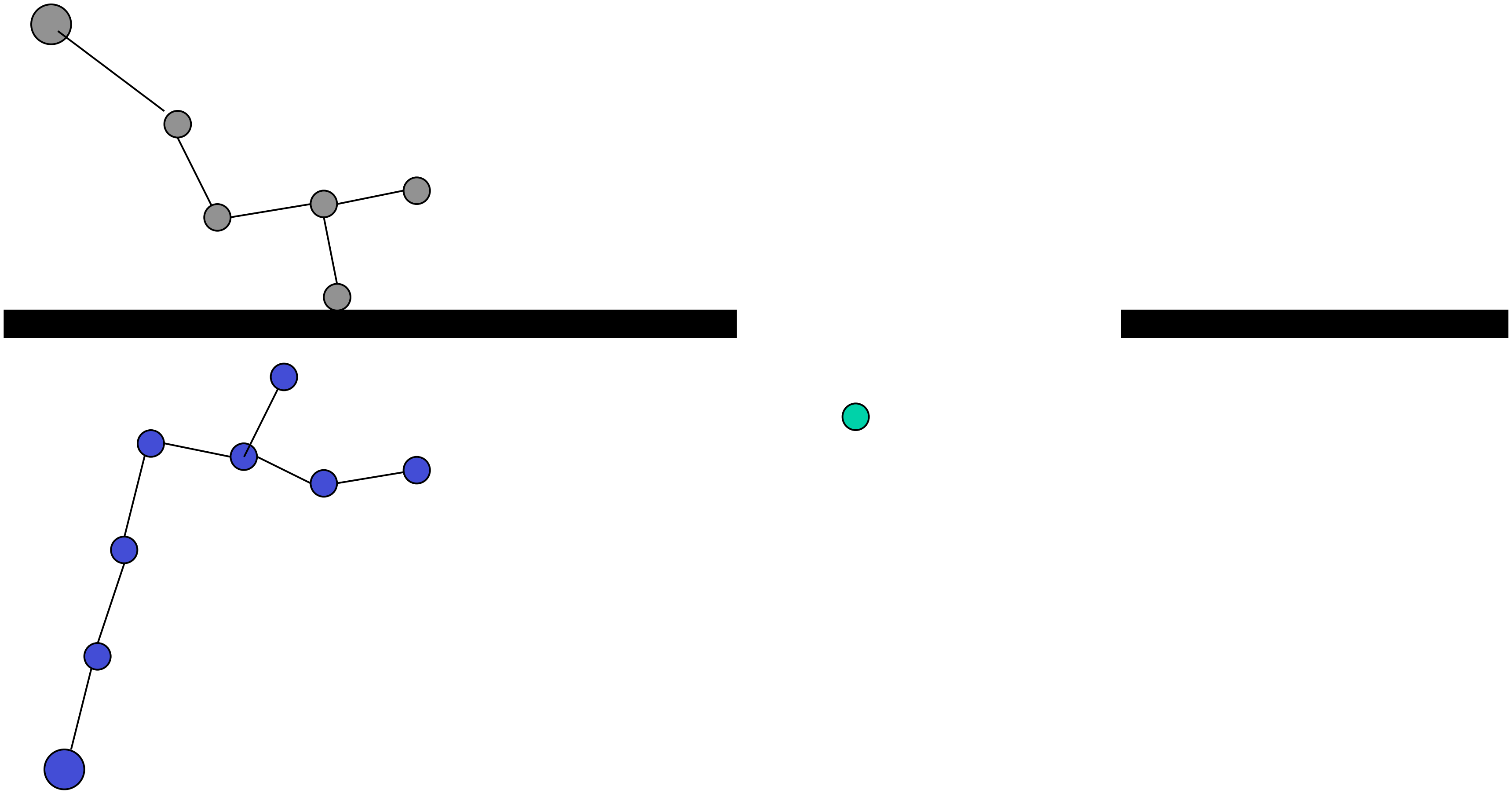
RRT-Connect: example



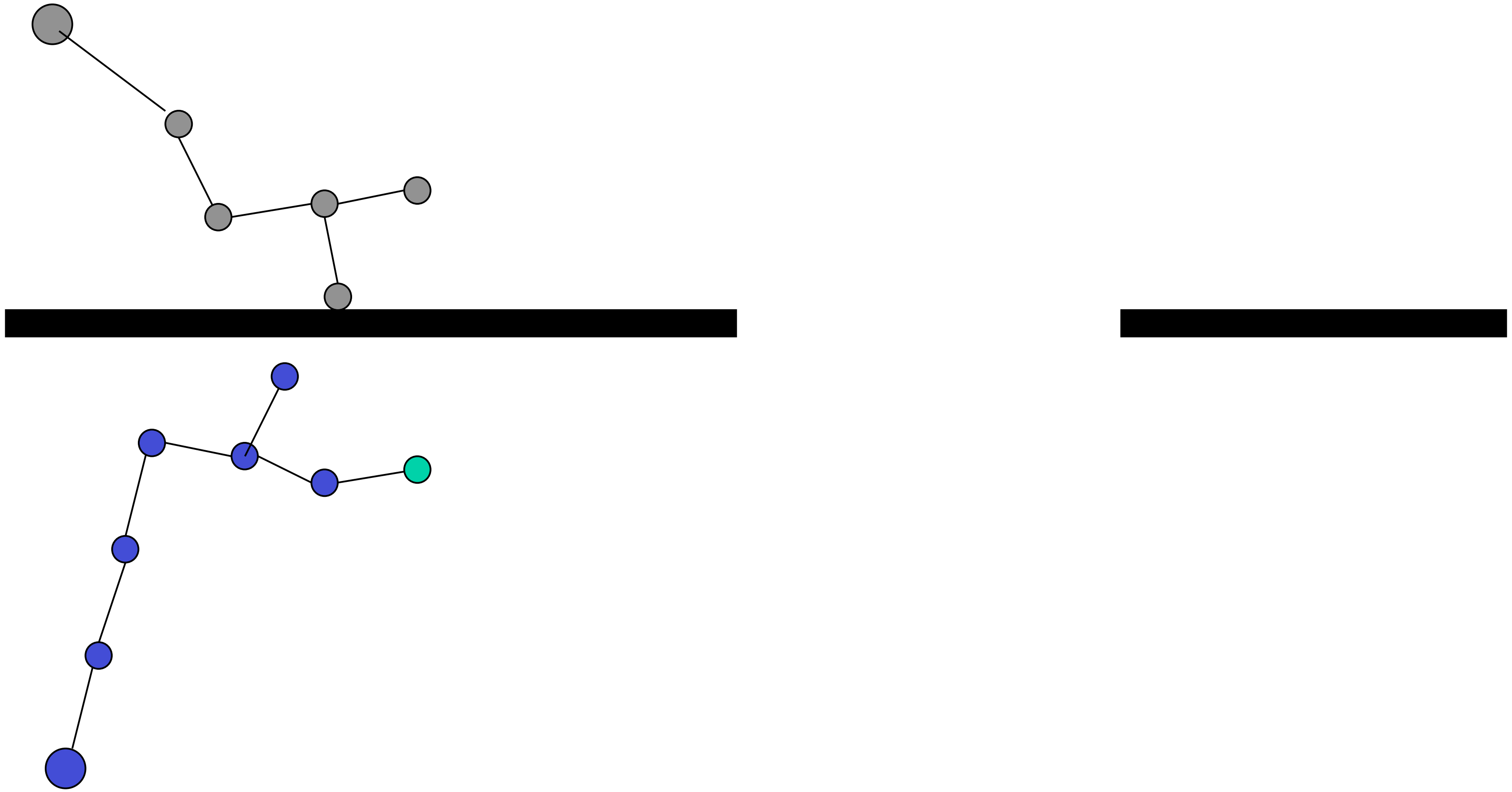
RRT-Connect: example



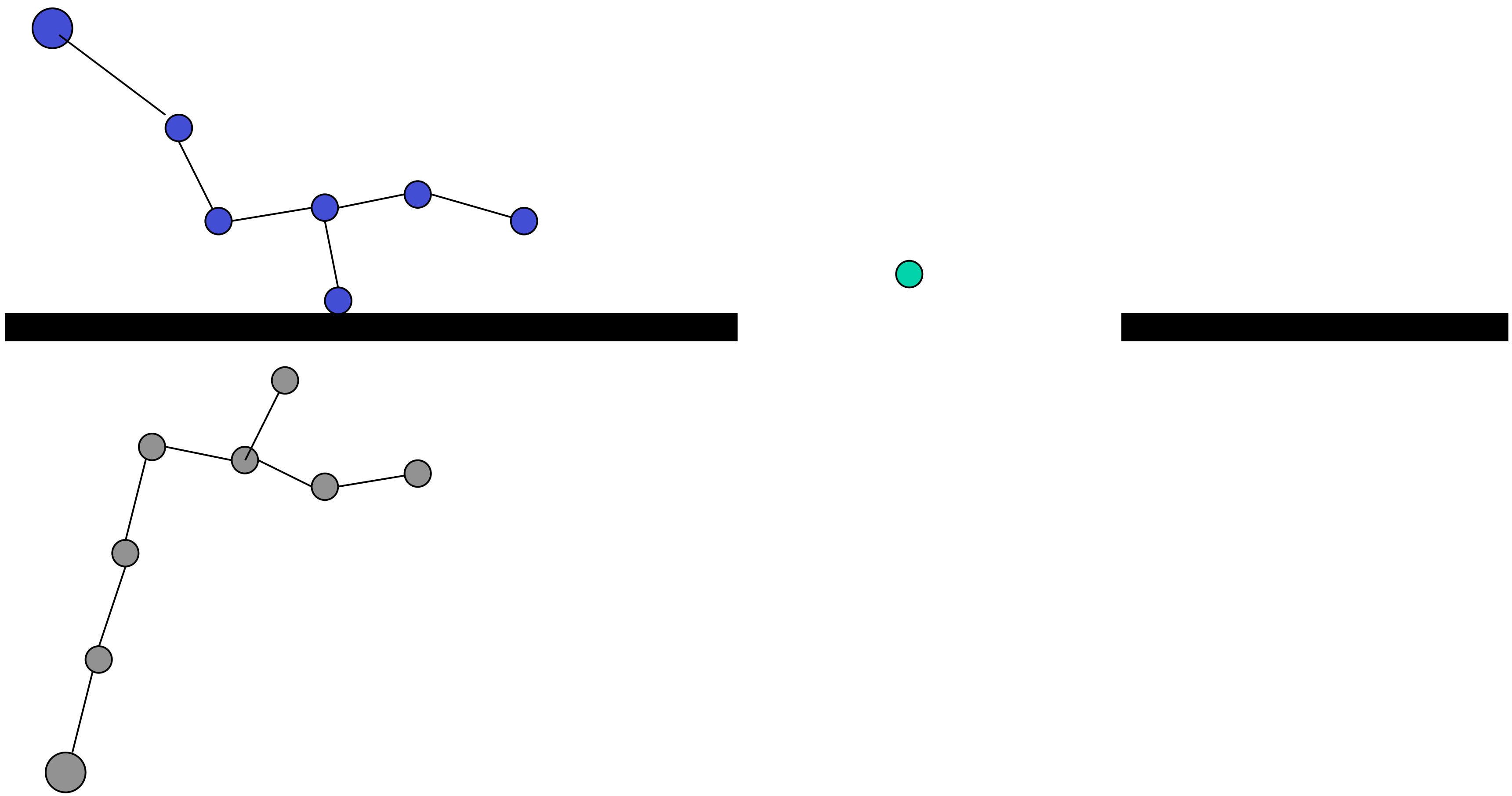
RRT-Connect: example



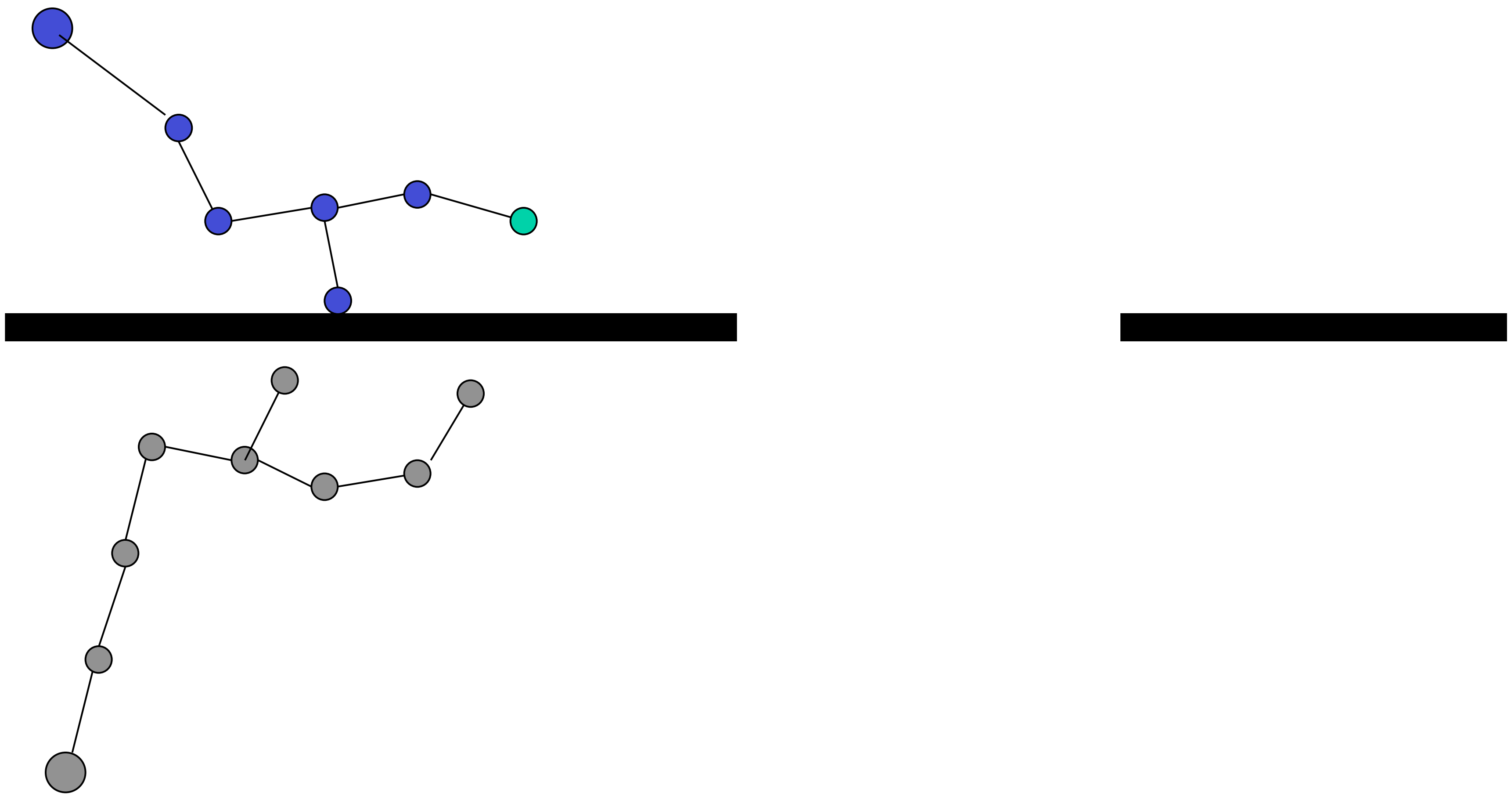
RRT-Connect: example



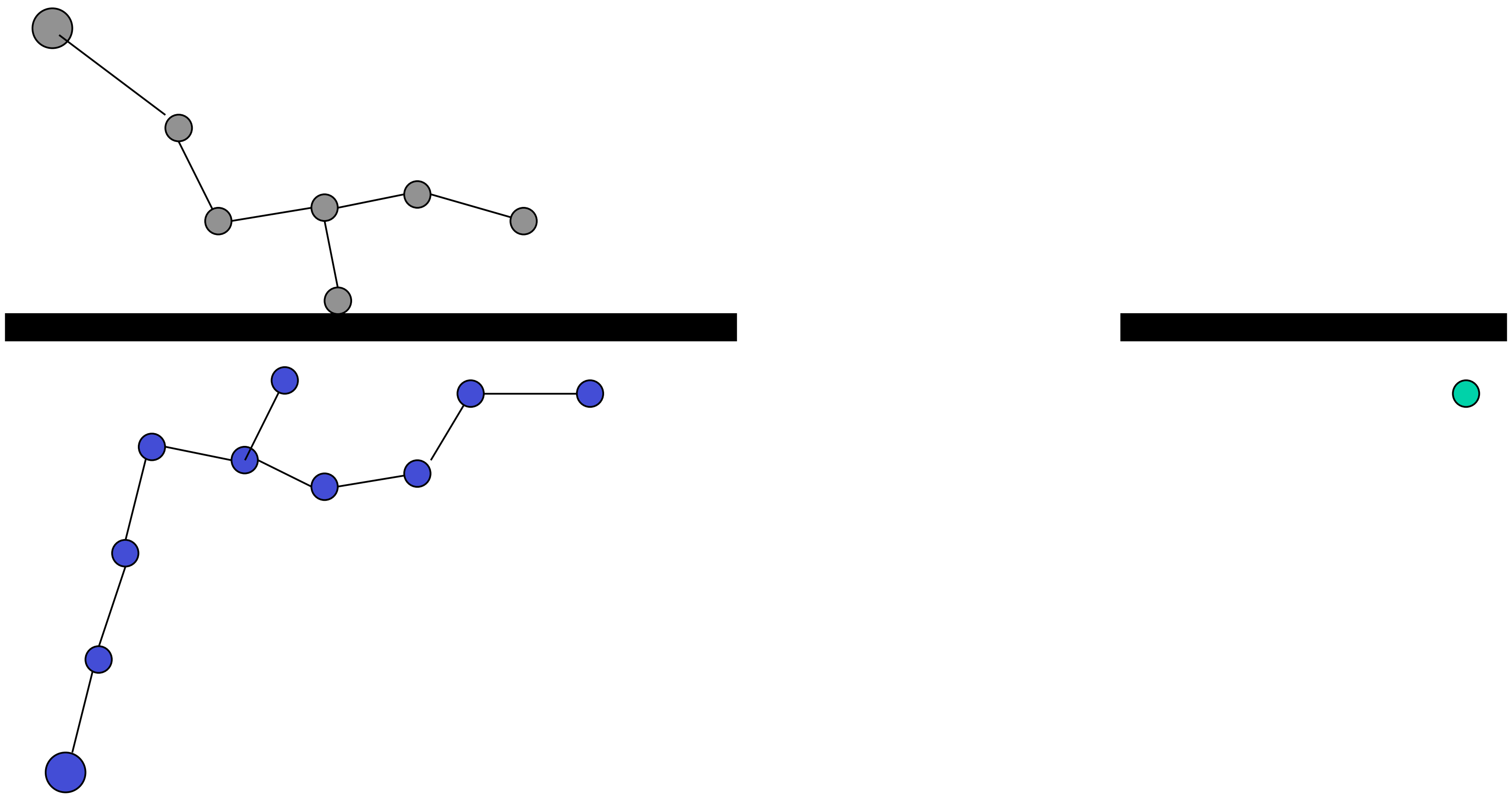
RRT-Connect: example



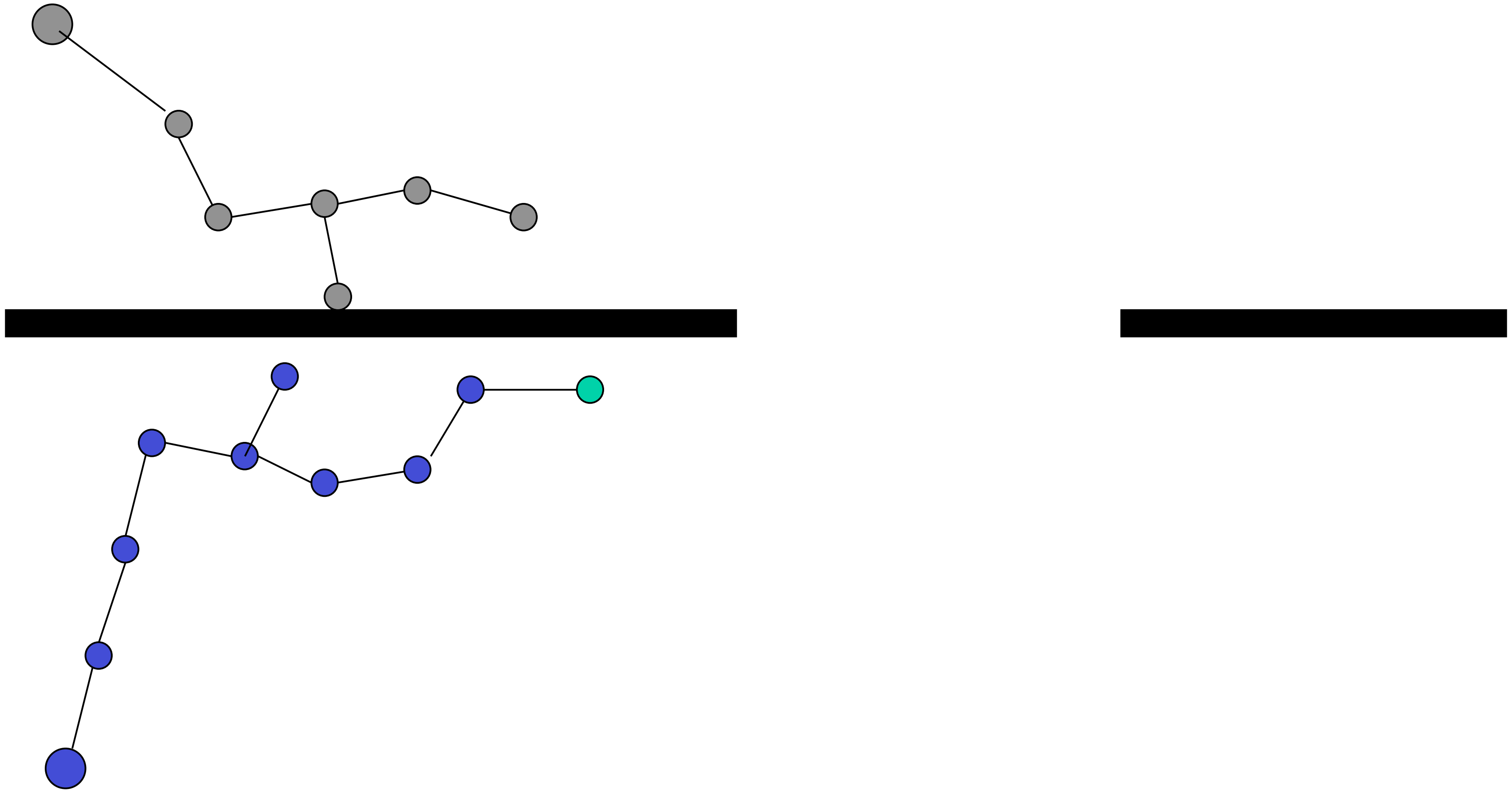
RRT-Connect: example



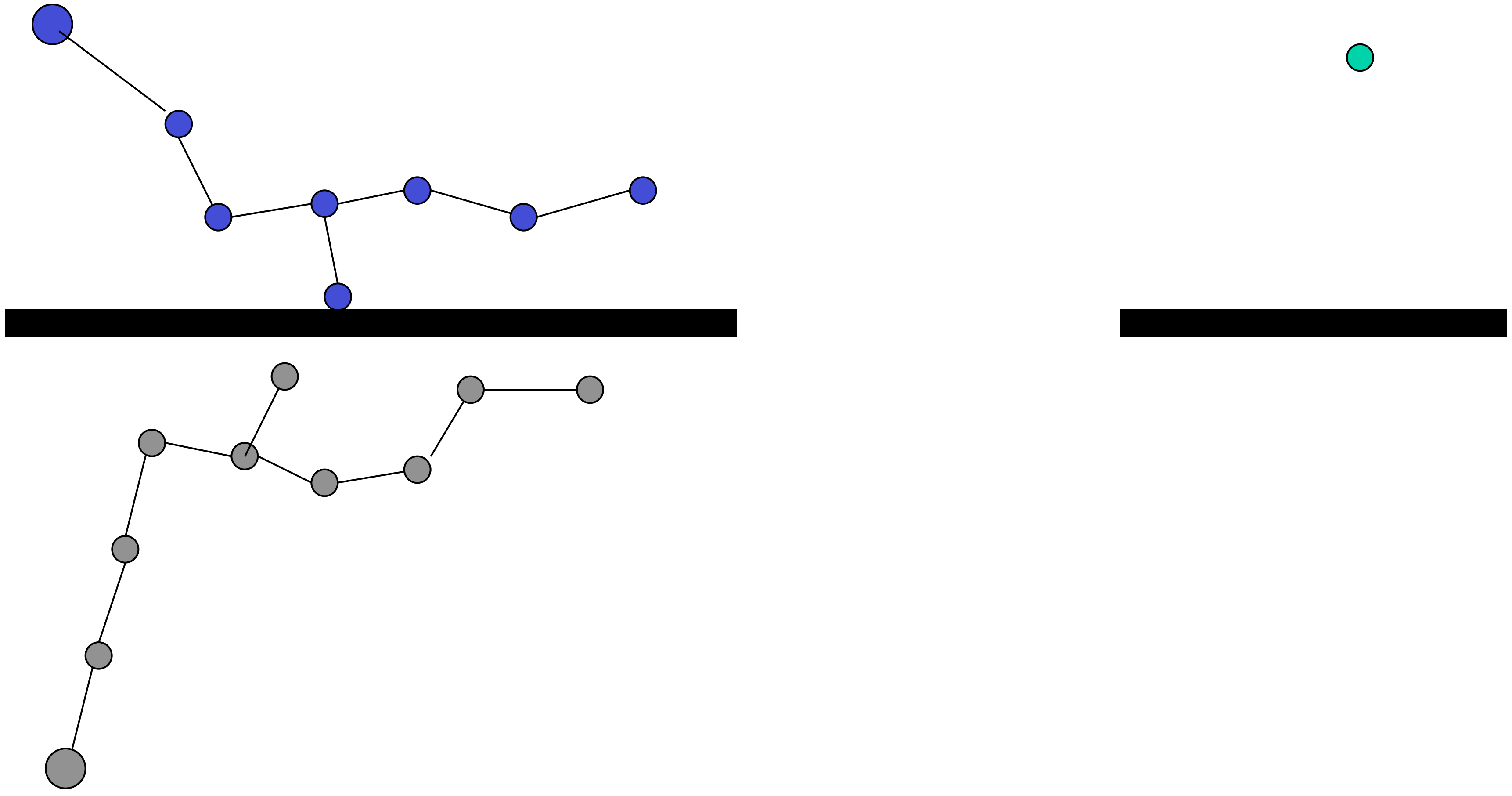
RRT-Connect: example



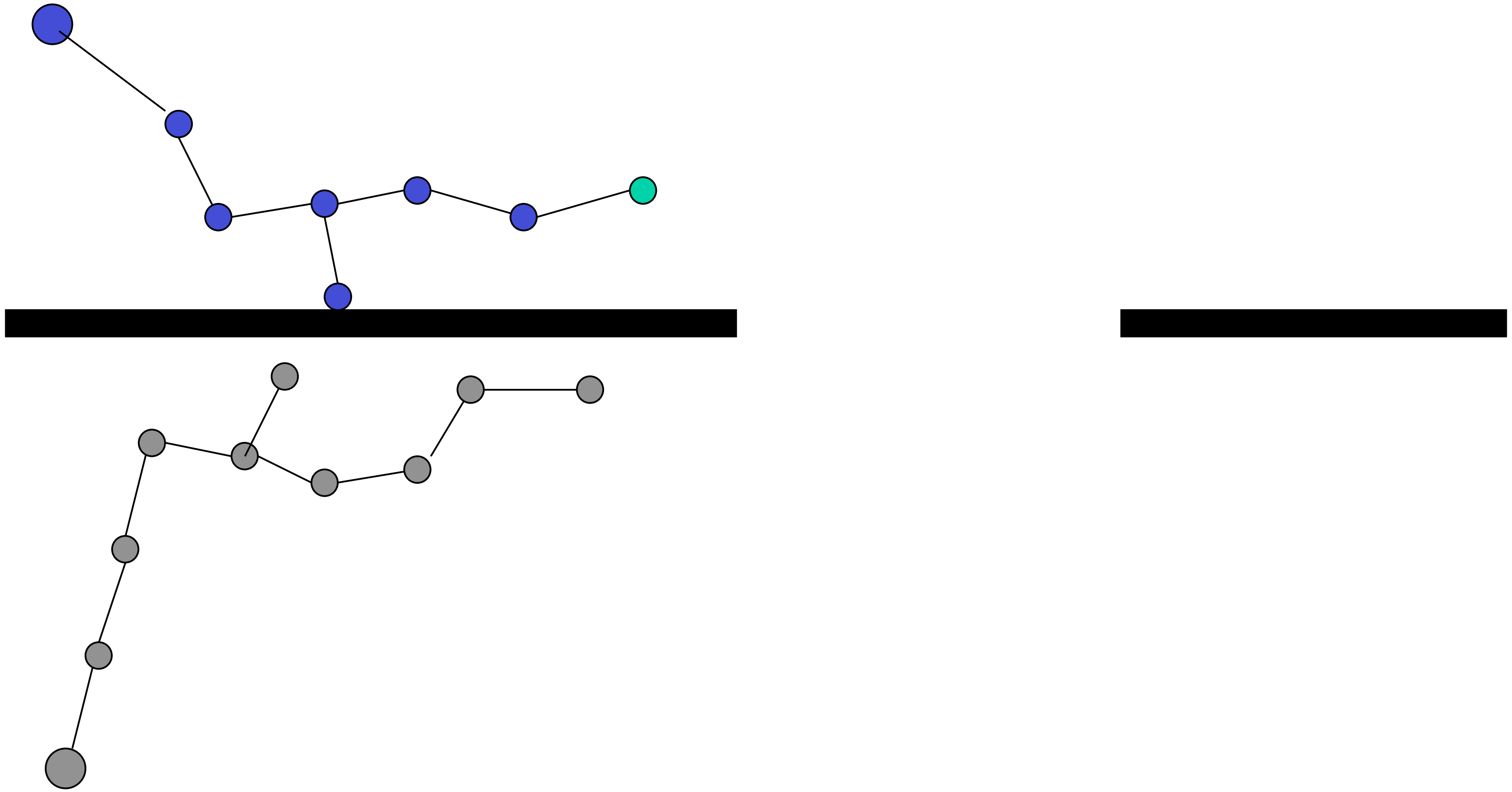
RRT-Connect: example



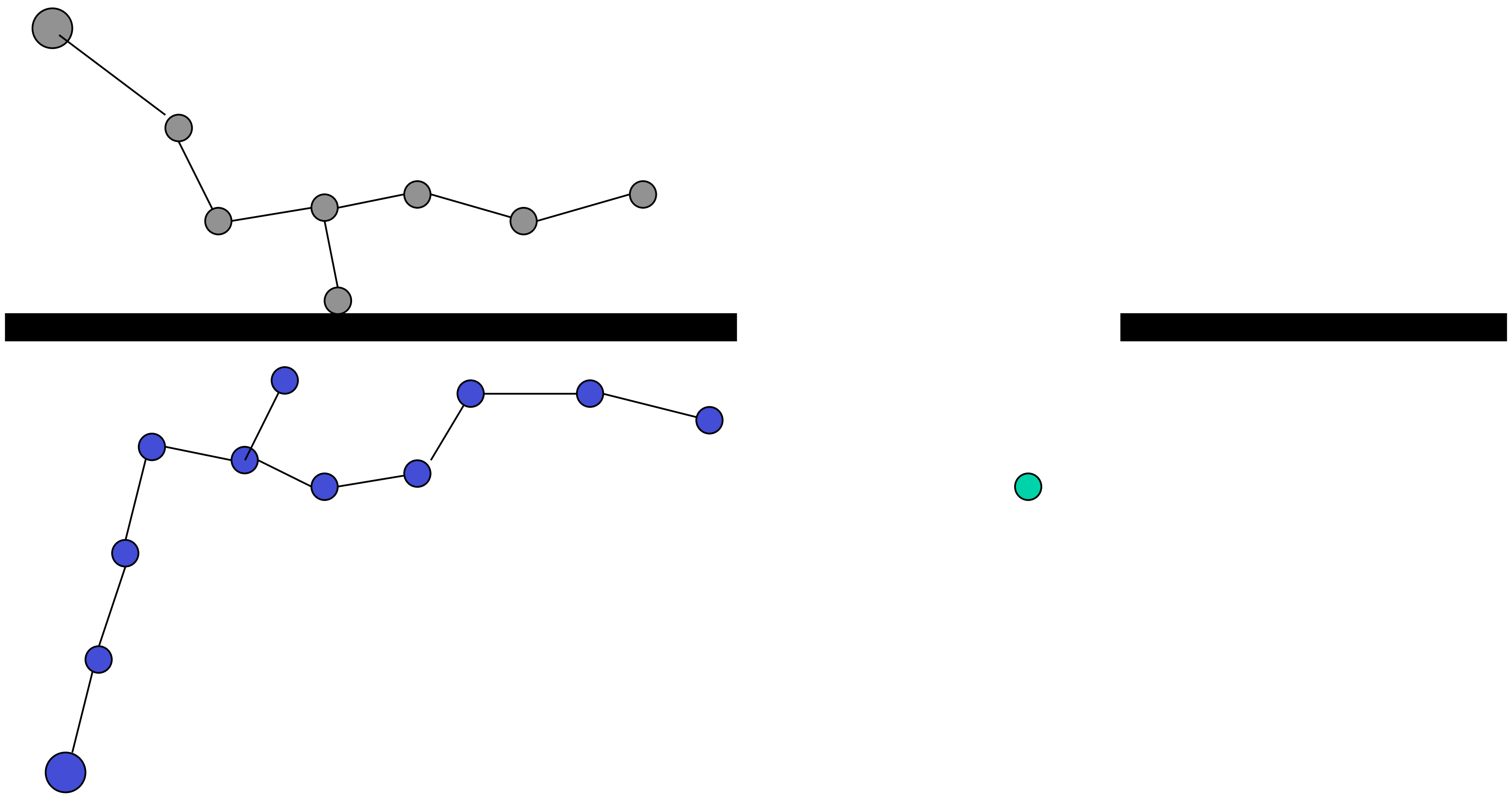
RRT-Connect: example



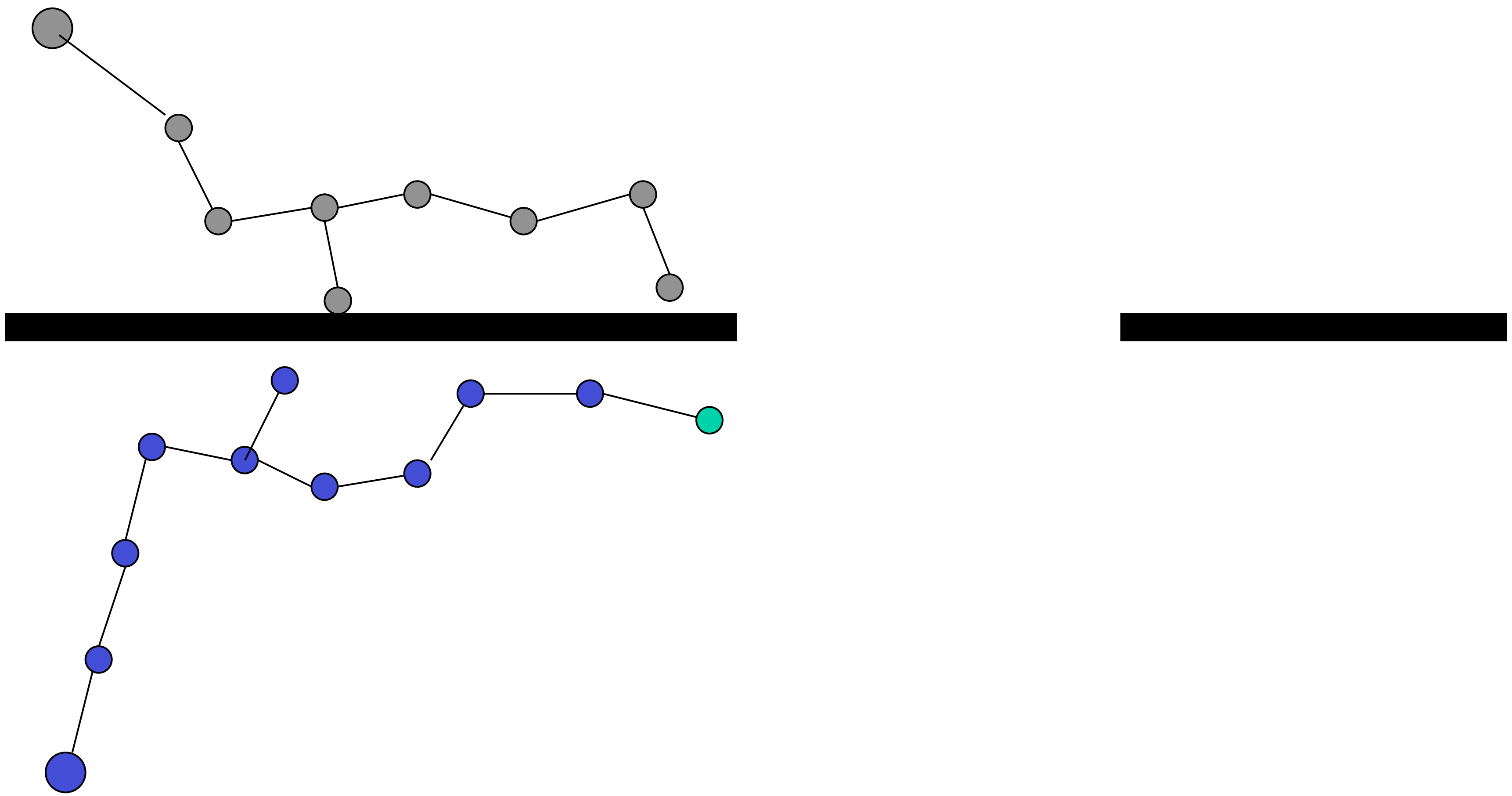
RRT-Connect: example



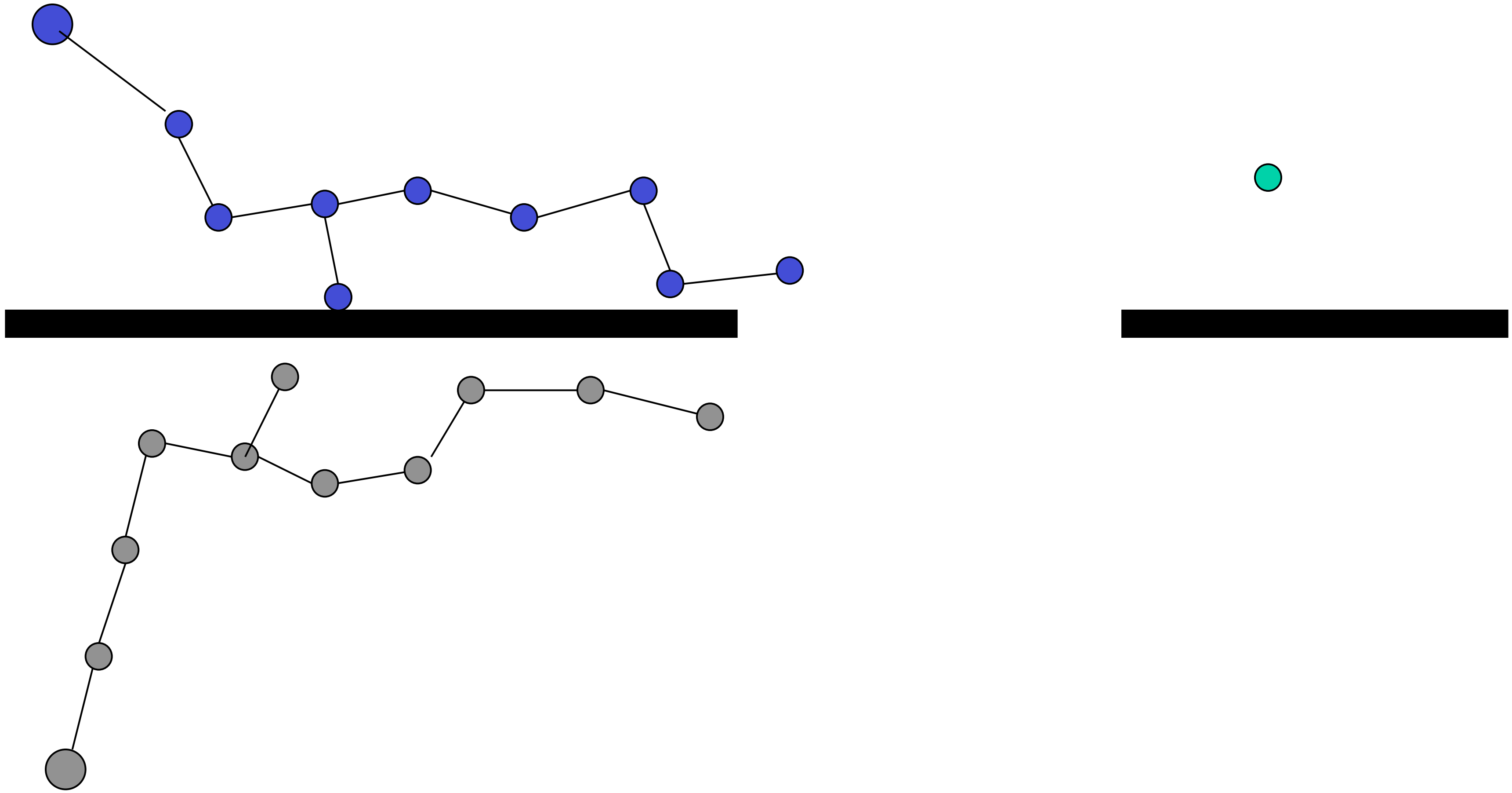
RRT-Connect: example



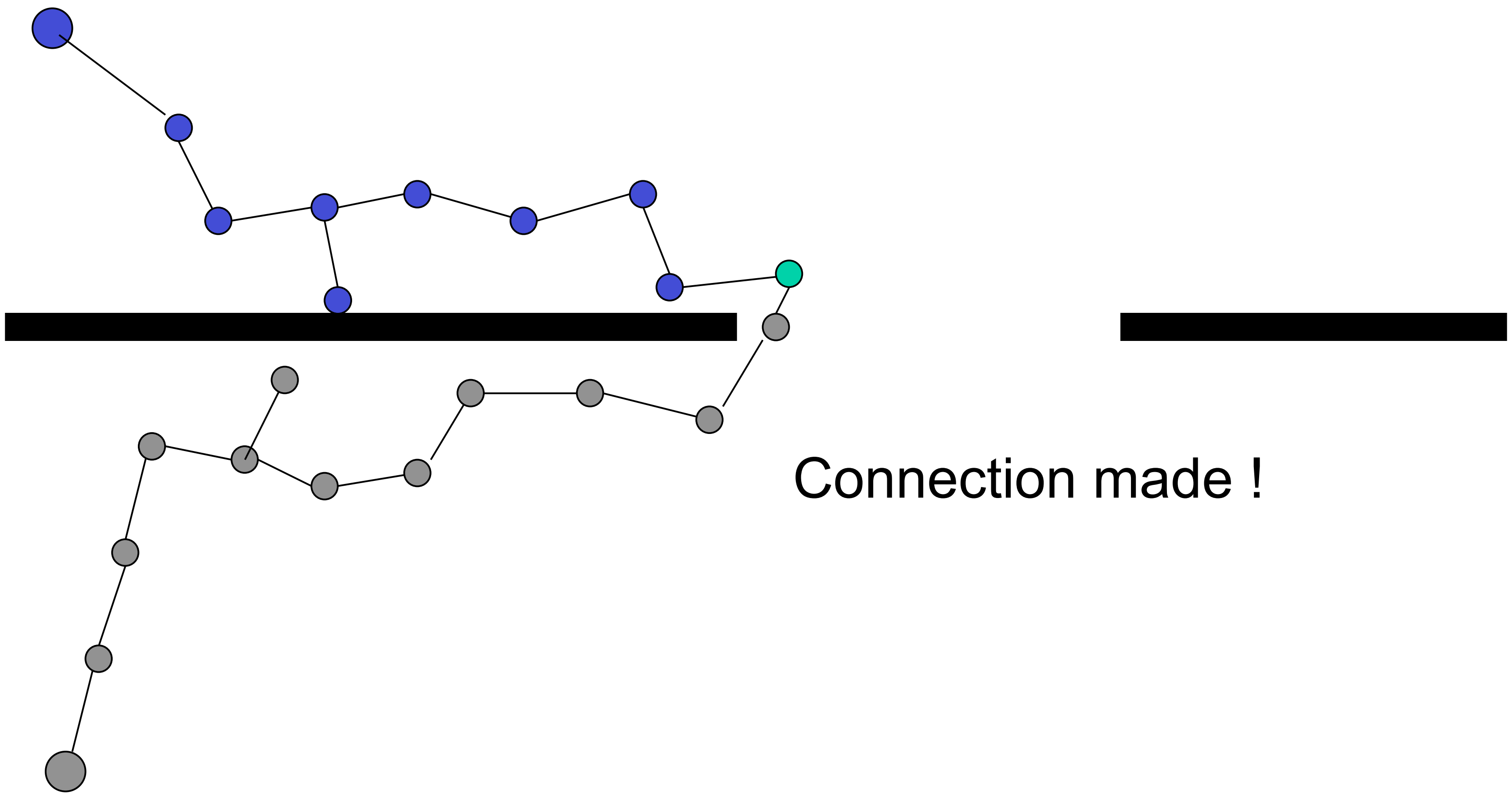
RRT-Connect: example



RRT-Connect: example

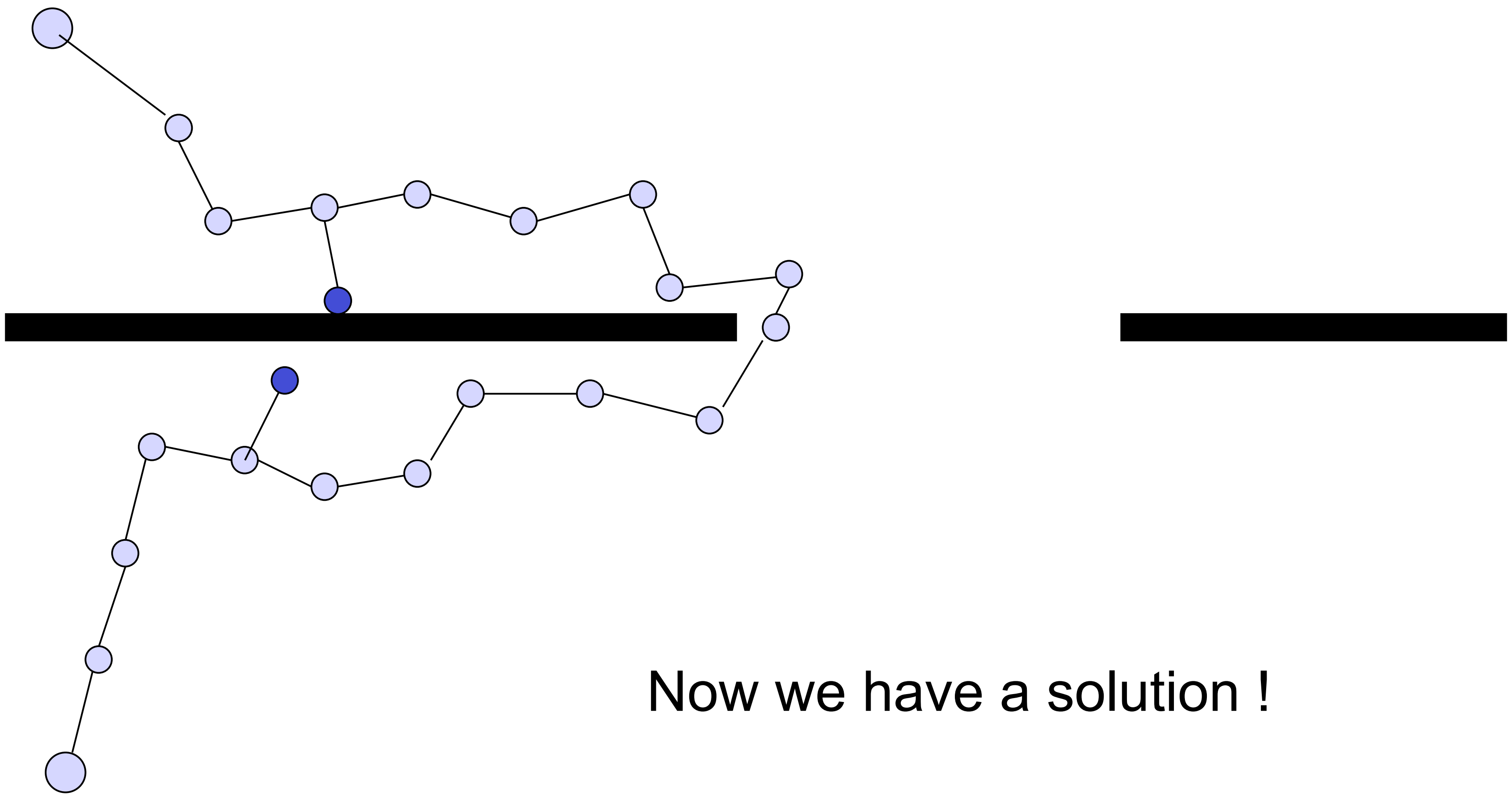


RRT-Connect: example



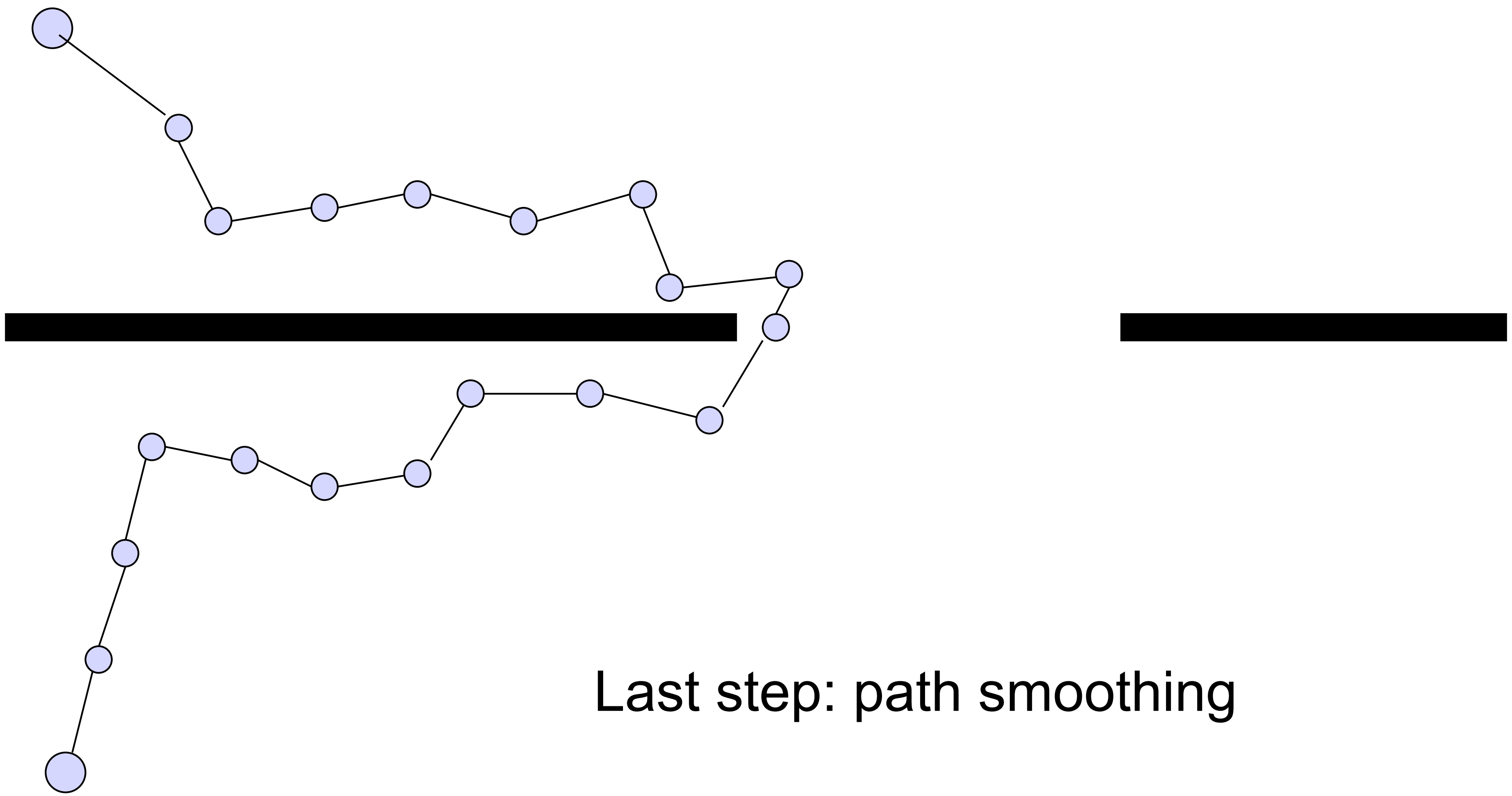
Connection made !

RRT-Connect: example



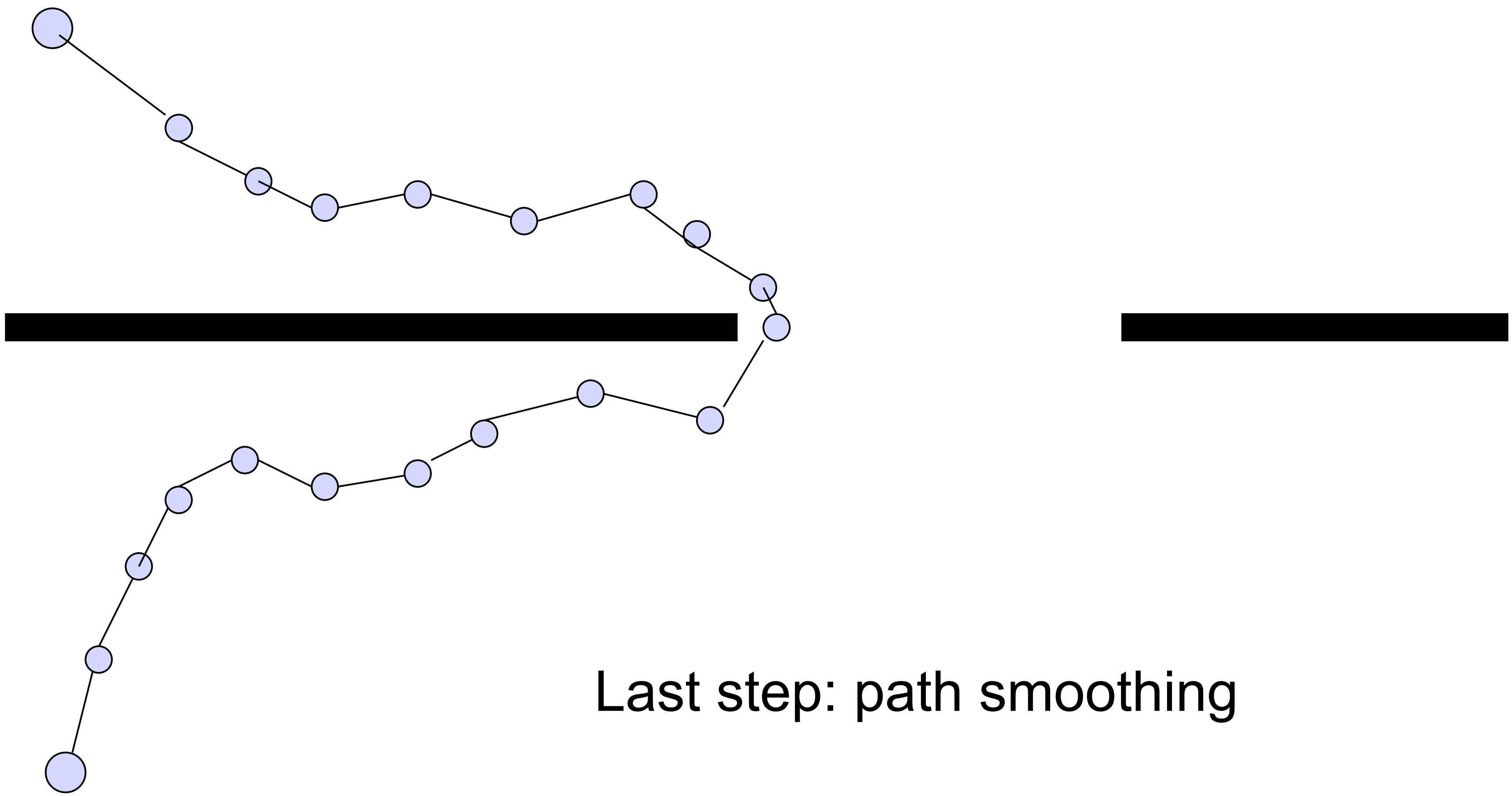
Now we have a solution !

RRT-Connect: example



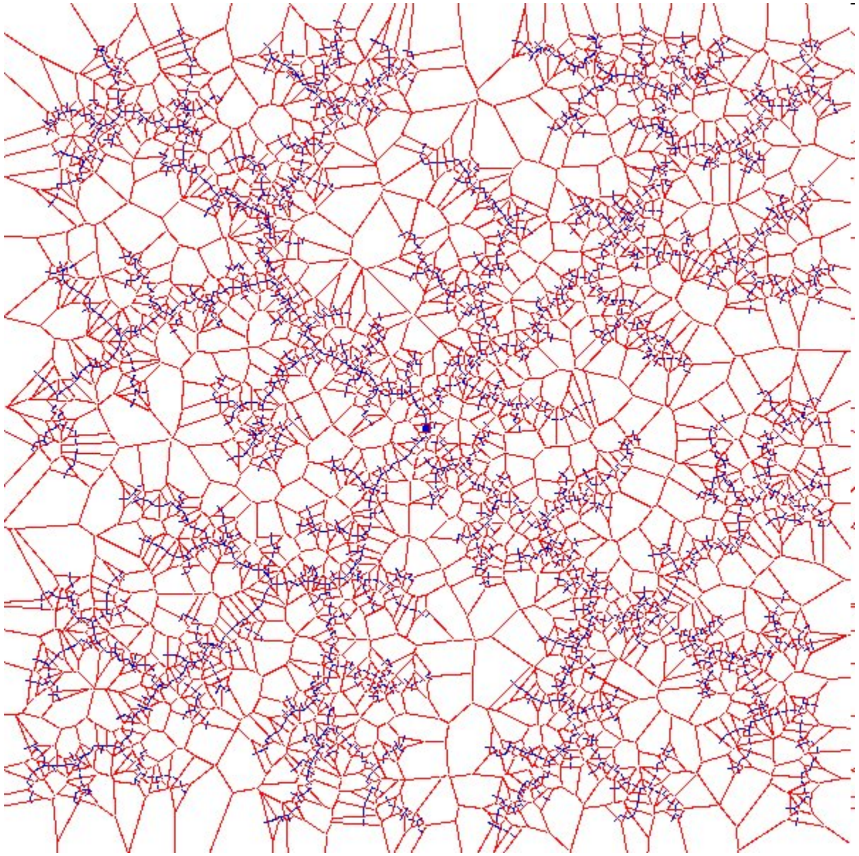
Last step: path smoothing

RRT-Connect: example



Last step: path smoothing

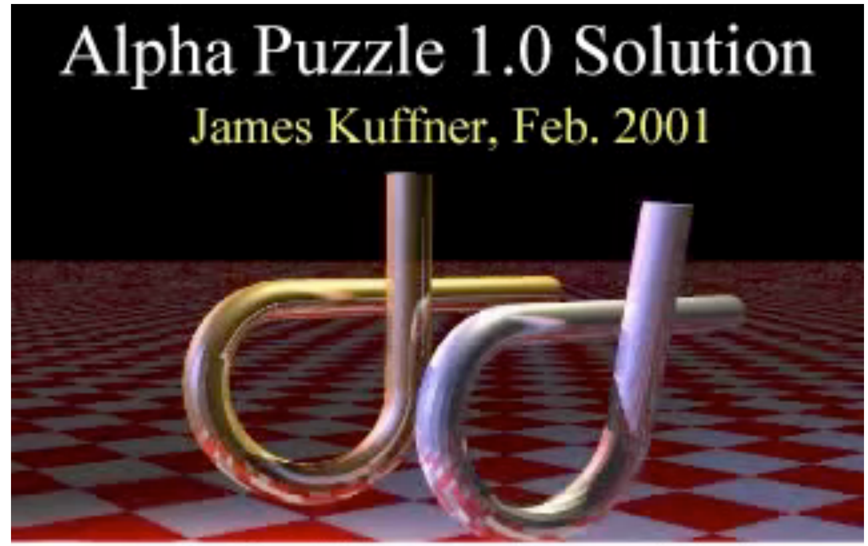
An RRT in 2D



Example from: http://msl.cs.uiuc.edu/rrt/gallery_2drrt.html

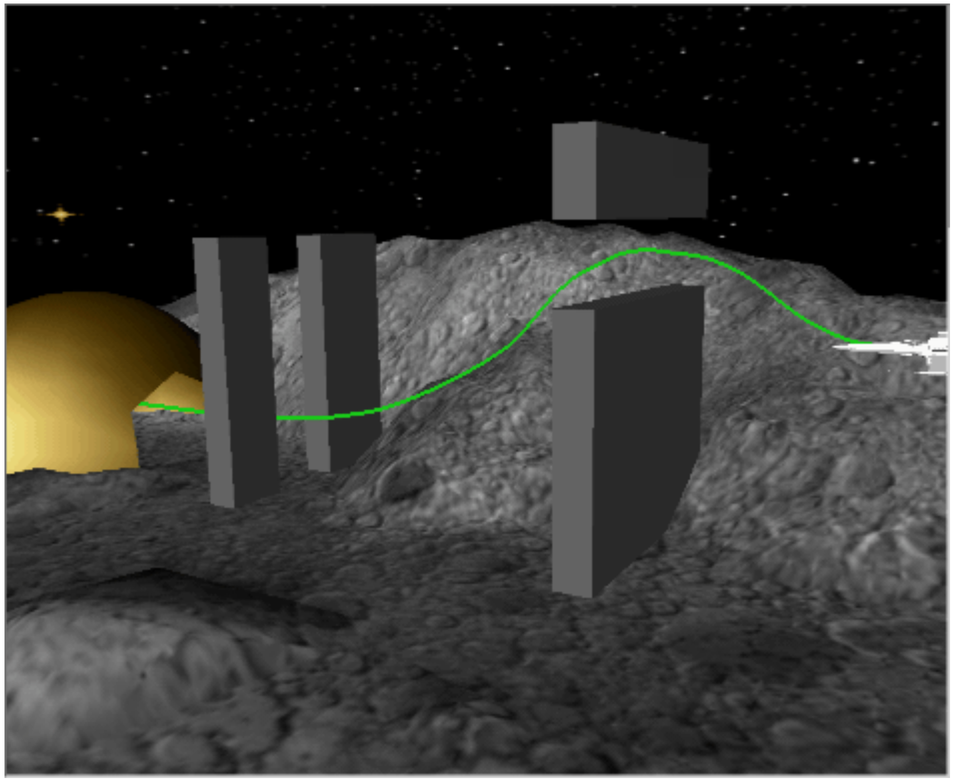
A Puzzle solved using RRTs

The goal is the separate the two bars from each other. You might have seen a puzzle like this before. The example was constructed by Boris Yamrom, GE Corporate Research & Development Center, and posted as a research benchmark by Nancy Amato at Texas A&M University. It has been cited in many places as a one of the most challenging motion planning examples. In 2001, it was solved by using a balanced bidirectional RRT, developed by James Kuffner and Steve LaValle. There are no special heuristics or parameters that were tuned specifically for this problem. On a current PC (circa 2003), it consistently takes a few minutes to solve.



model by DSMFT group, Texas A&M Univ.
original model by Boris Yamrom, GE

Lunar Landing



The following is an open loop trajectory that was planned in a 12-dimensional state space. The video shows an X-Wing fighter that must fly through structures on a lunar base before entering the hangar. This result was presented by Steve LaValle and James Kuffner at the Workshop on the Algorithmic Foundations of Robotics, 2000.