Robotics Basics

Prof. Yunzhu Li
Spring 2024

CS598YL: Deep Learning for Robotic Manipulation
* Course materials adapted from Prof. Saurabh Gupta's ECE598SG at UIUC.

Logistics

- Presentation schedule is released.
- For the syllabus on the website, only papers with solid bullet points need to be presented. Papers with hollow bullet points are recommended but not required.
- After this week, the course format will be primarily student presentation/discussion, guest lectures, and project presentation.

Logistics

- The deadline for the slides for the first presentation is pushed back to tomorrow midnight (11:59 PM on 1/24/2024 Wednesday)
  - 01/30 (Tue) - Haozhe Si, Kulbir Singh Ahluwalia, Xin Xu
  - 02/01 (Thu) - Bryan Zhang, Vaibhav Dhanesh Raheja, Jose Cuaran

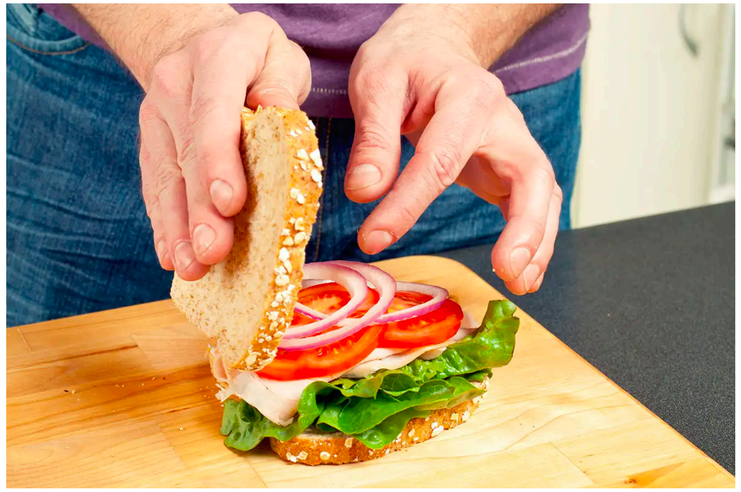
Logistics

- The feedback session for your presentation is on Thursday 1/25
  - 1:15-1:30 pm: Haozhe Si
  - 1:30-1:45 pm: Kulbir Singh Ahluwalia
  - 1:45-2:00 pm: Xin Xu
  - 2:00-2:15 pm: Bryan Zhang
  - 2:15-2:30 pm: Vaibhav Dhanesh Raheja
  - 2:30-2:45 pm: Jose Cuaran

Today’s Agenda

- Robotic Tasks - Manipulation
- Terminology
- Observation and camera calibration
- How to move your robot?
  - Task space & configuration space
  - Forward kinematics and inverse kinematics
  - Path planning

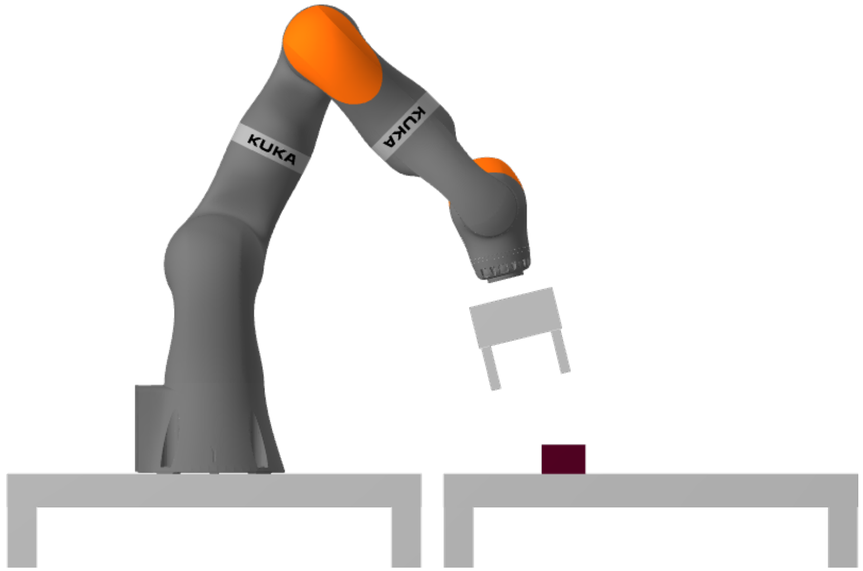
Robotic Tasks - Manipulation

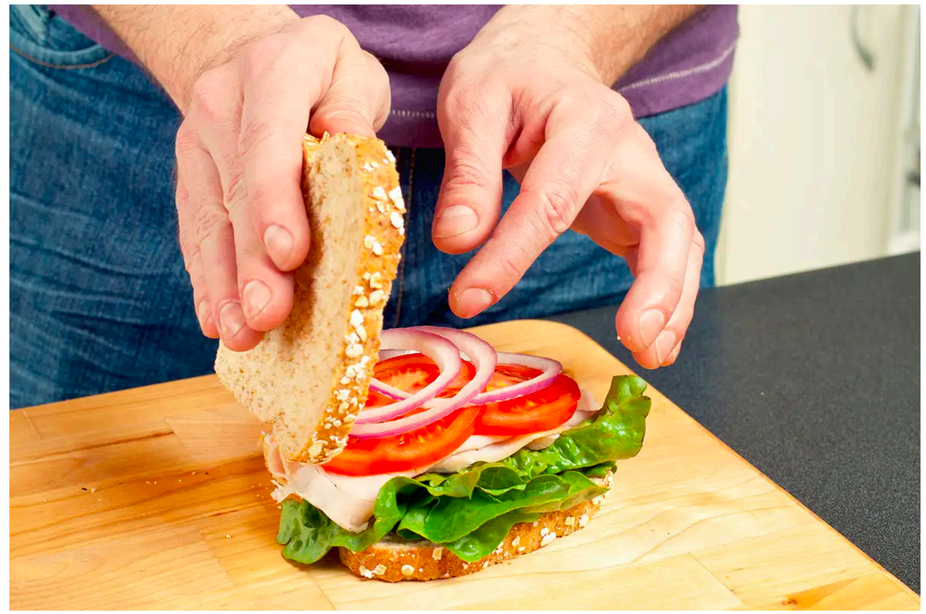


Robotic Tasks - Manipulation

Robotic Tasks - Manipulation

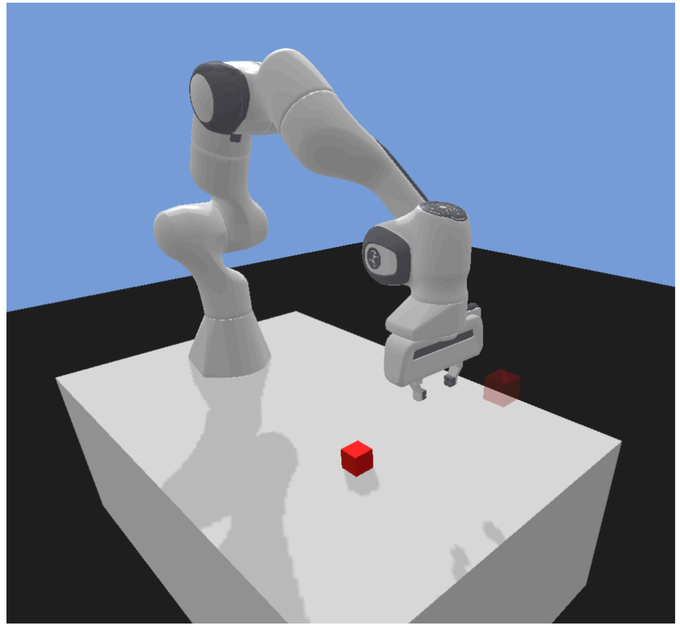
- Task horizon





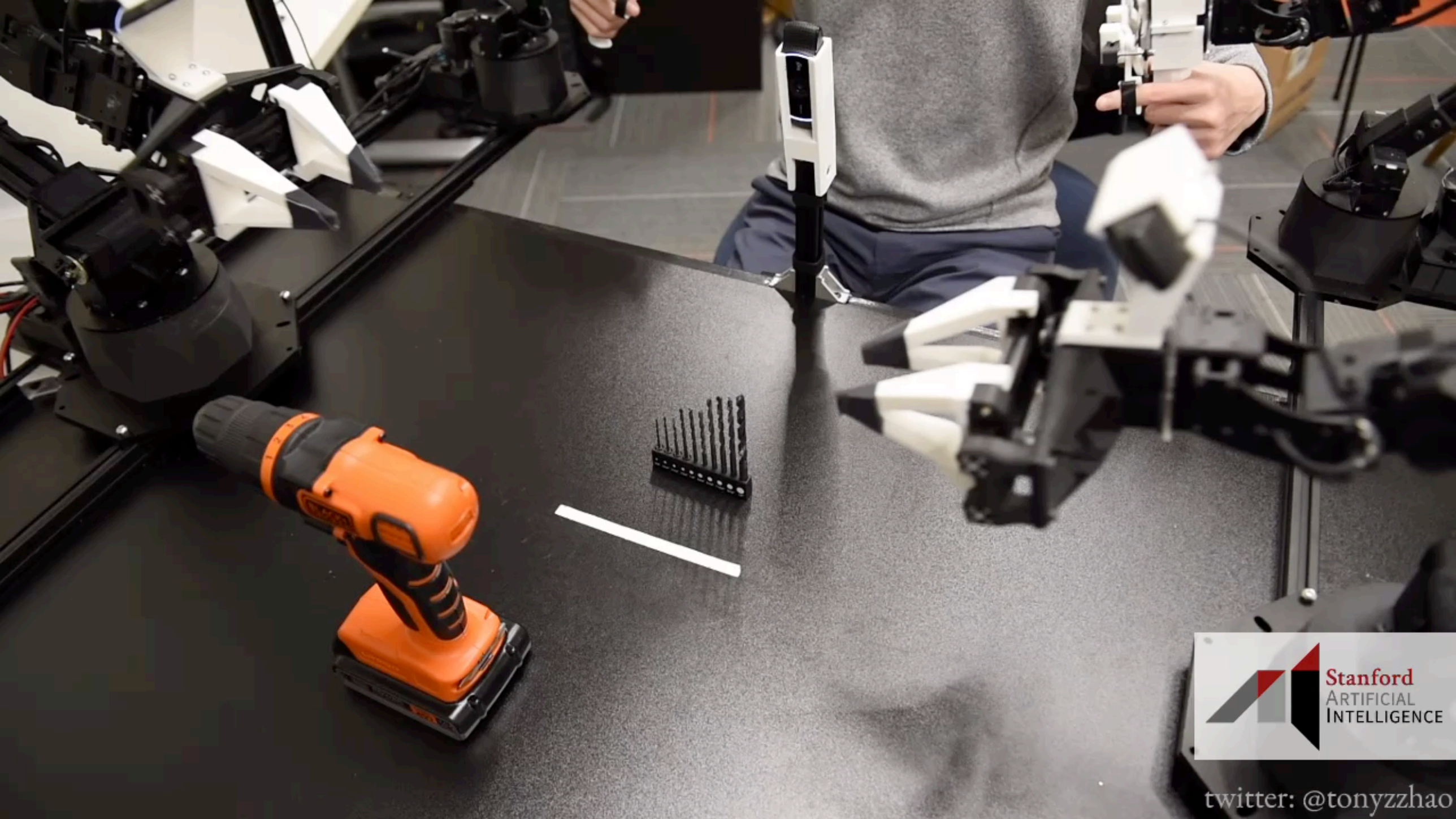
Robotic Tasks - Manipulation

- Task horizon
- Number of objects involved
- Types of objects involved



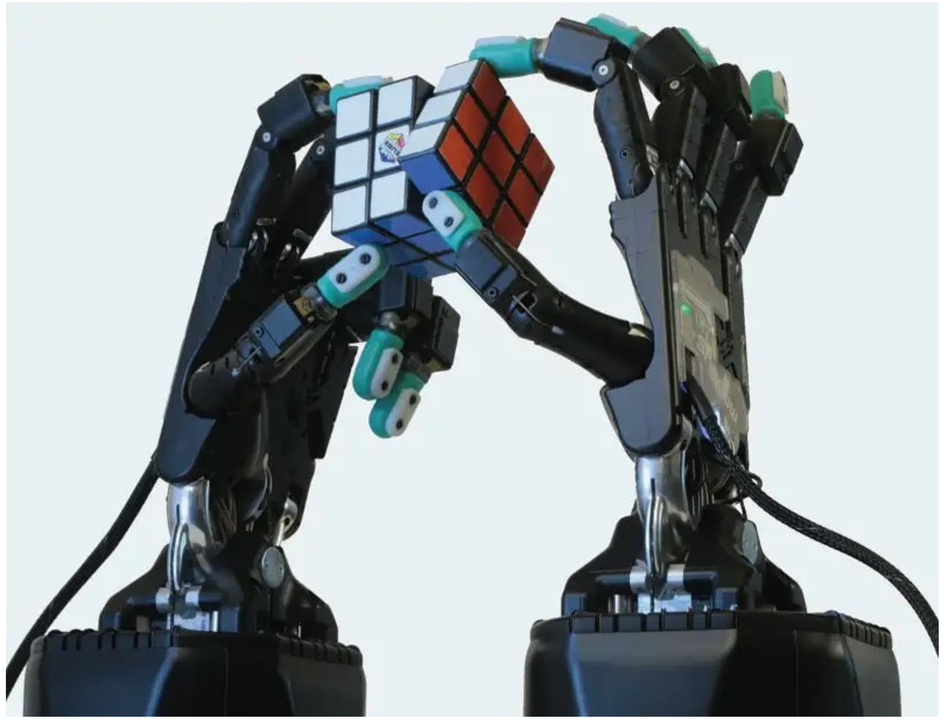
Robotic Tasks - Manipulation

- Task horizon
- Number of objects involved
- Types of objects involved
- Your robot embodiment



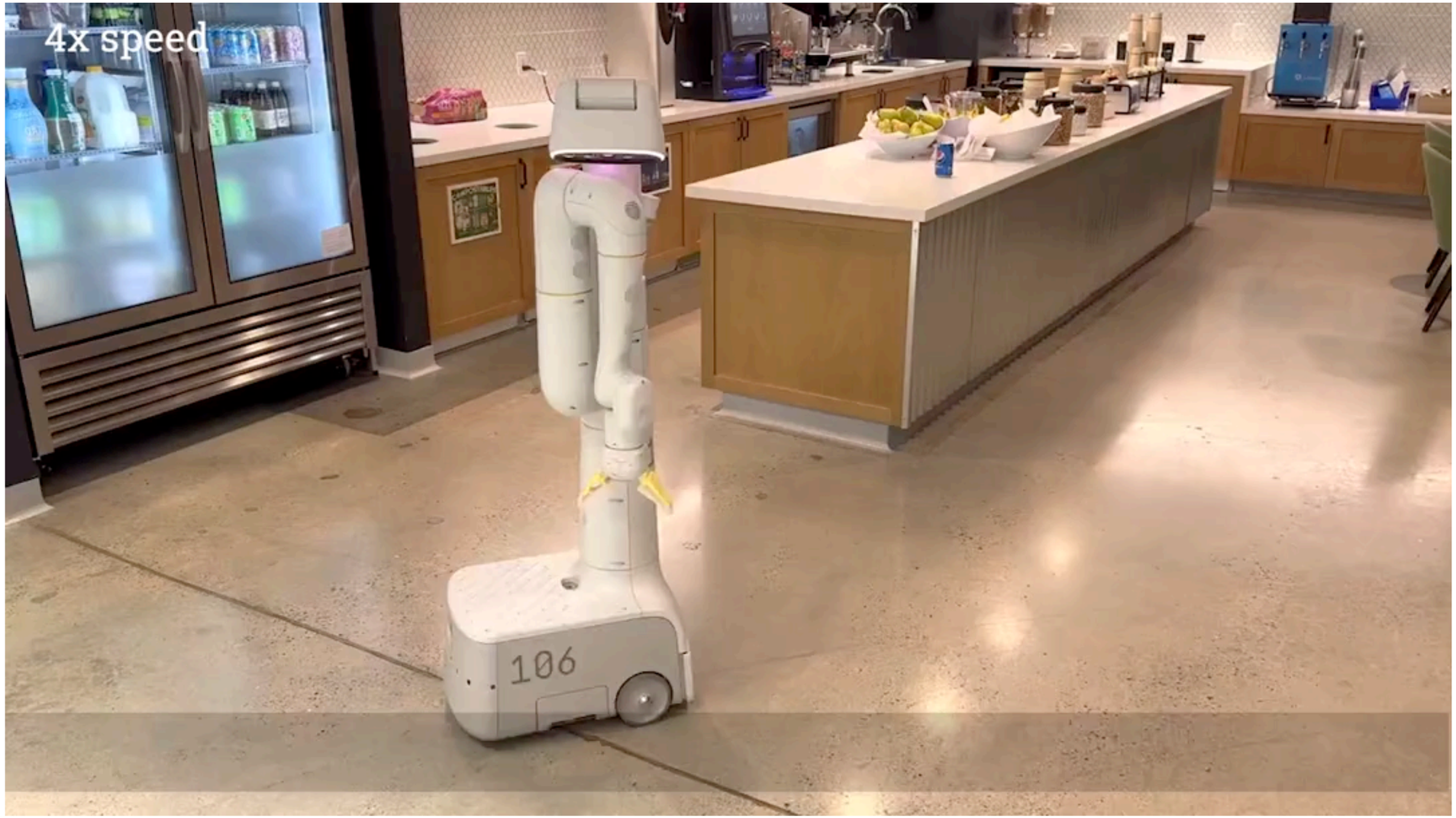
twitter: @tonyzzhao

The need for dexterous hands




[-1, 0, 0]
[-1, 0, 0]
[-1, 0, 0]
[1, 0, 0]
[1, 0, 0]
[1, 0, 0]
[0, 1, 0]
[0, 1, 0]
[0, 1, 0]
[0, -1, 0]
[0, -1, 0]
[0, -1, 0]
[0, 0, -1]
[0, 0, -1]
[0, 0, -1]
[0, 0, 1]
[0, 0, 1]
[0, 0, 1]
[-1, -1, -1]
[-1, -1, 1]
[-1, 0, -1]
[-1, 0, 1]
[1, 0, -1]
[0, -1, -1]
*The tactile sensors have been blurred to preserve confidentiality.

The need for mobile manipulator



Robotic Tasks - Manipulation

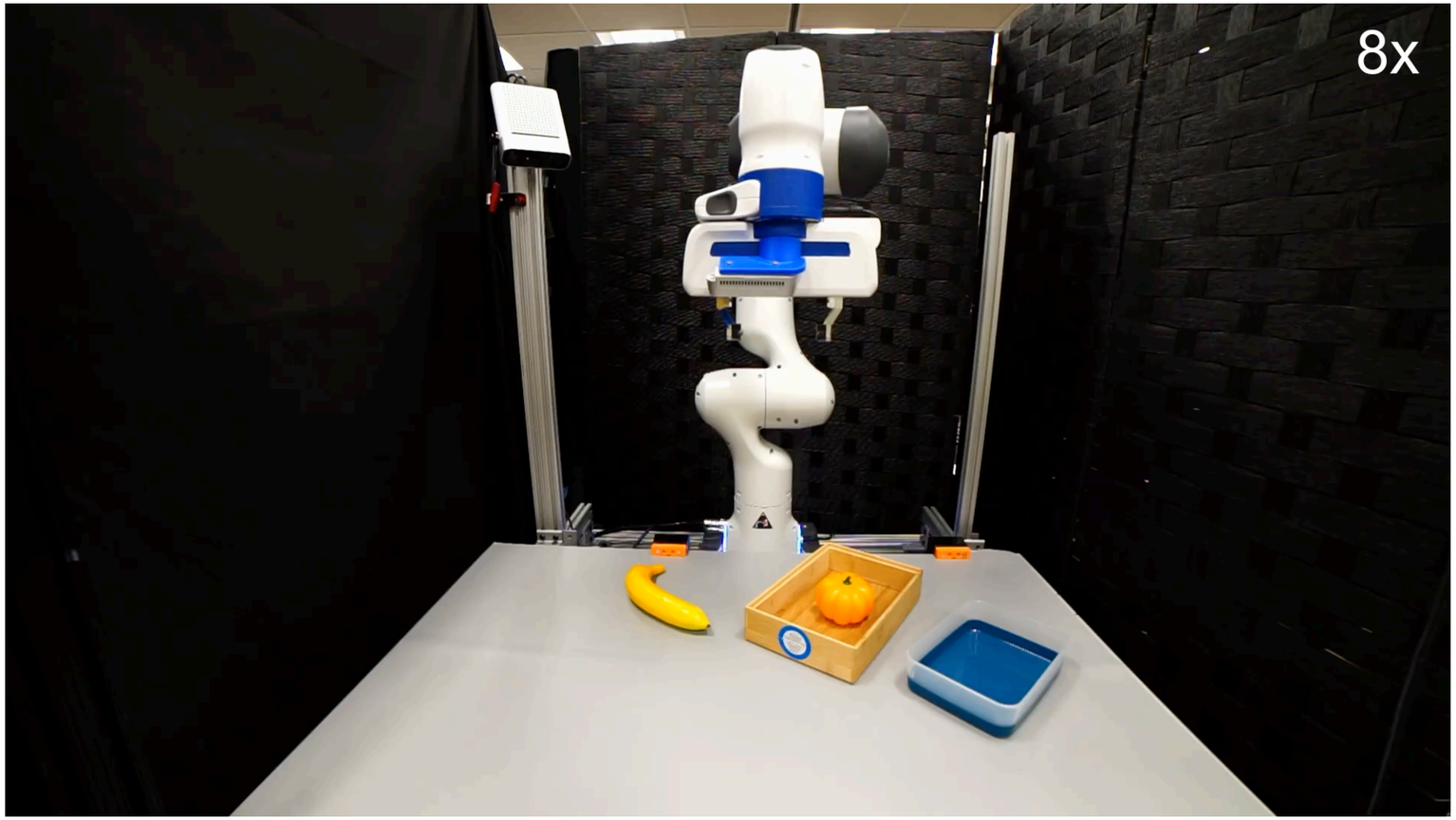
- Task horizon
- Number of objects involved
- Types of objects involved
- Your robot embodiment

Robotic Tasks - Manipulation

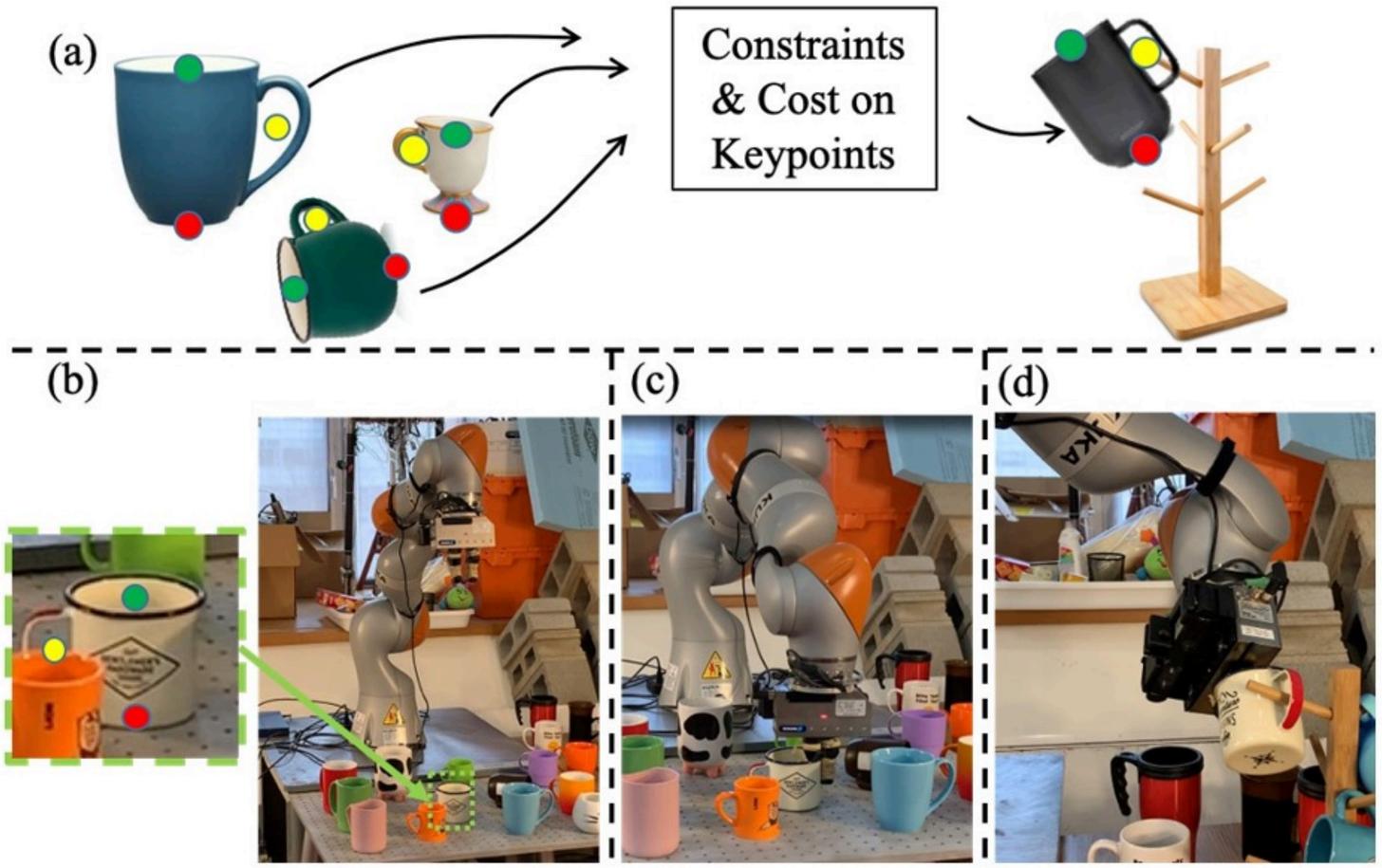
- Task horizon
- Number of objects involved
- Types of objects involved
- Your robot embodiment
- Reliability and robustness

Robotic Tasks - Manipulation

- Task horizon
- Number of objects involved
- Types of objects involved
- Your robot embodiment
- Reliability and robustness



Robotic Tasks - Manipulation



Lucas Manuelli*, Wei Gao*, Peter Florence, Russ Tedrake

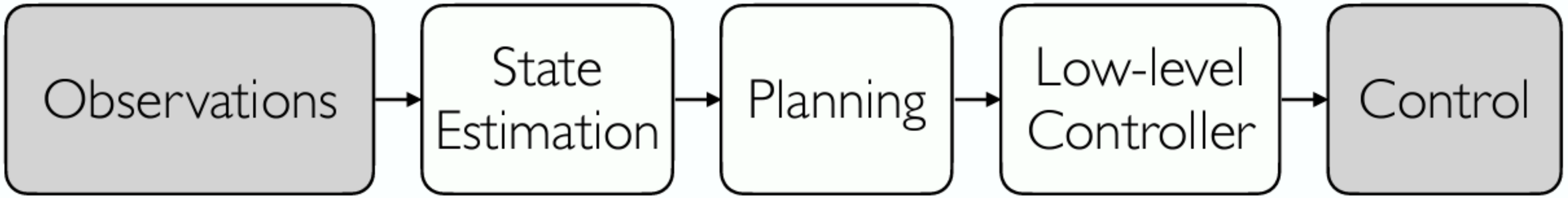
kPAM: KeyPoint Affordances for Category-Level Robotic Manipulation


KUKA
1x speed

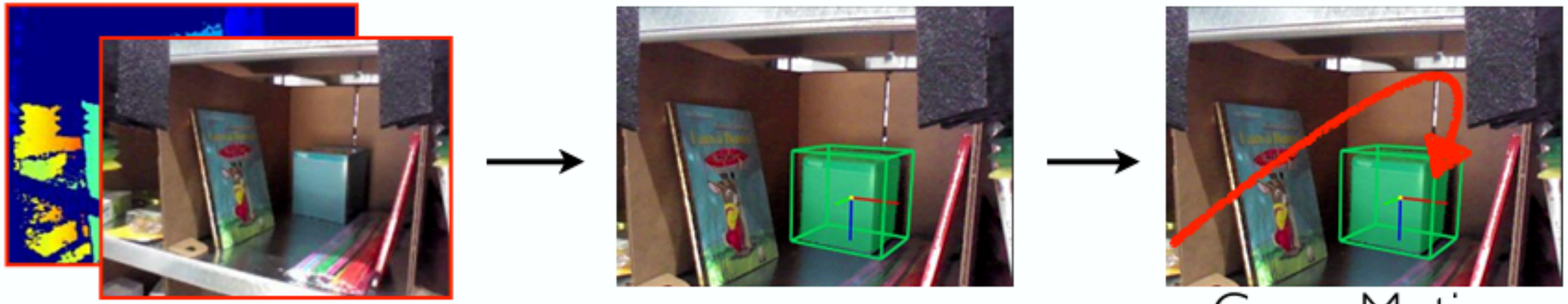

SILGAN
KUKA
KUKA
1x speed


KUKA
KUKA
Domino
SUGAR

Typical Robotics Pipeline



Manipulation

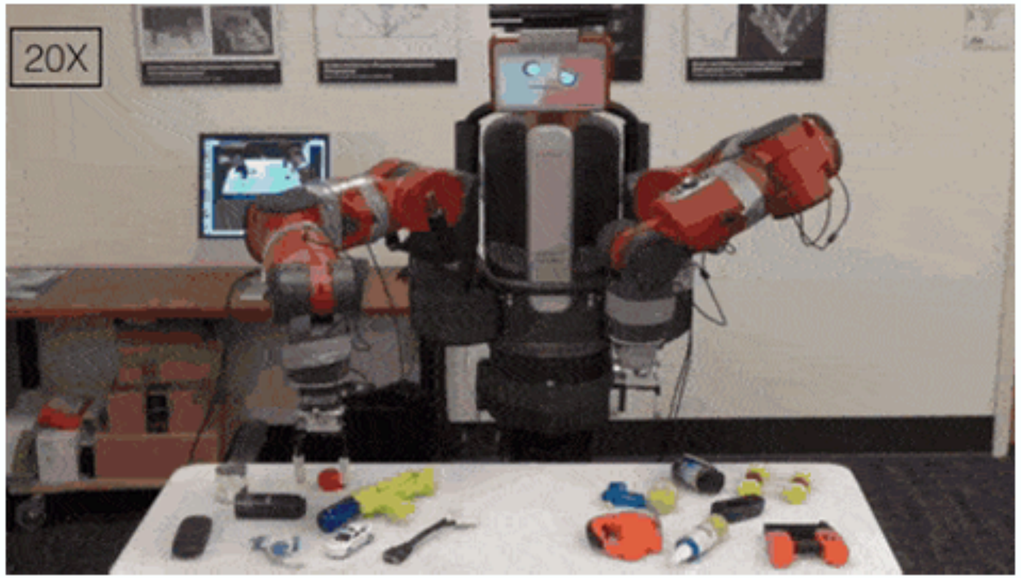


Observed Images

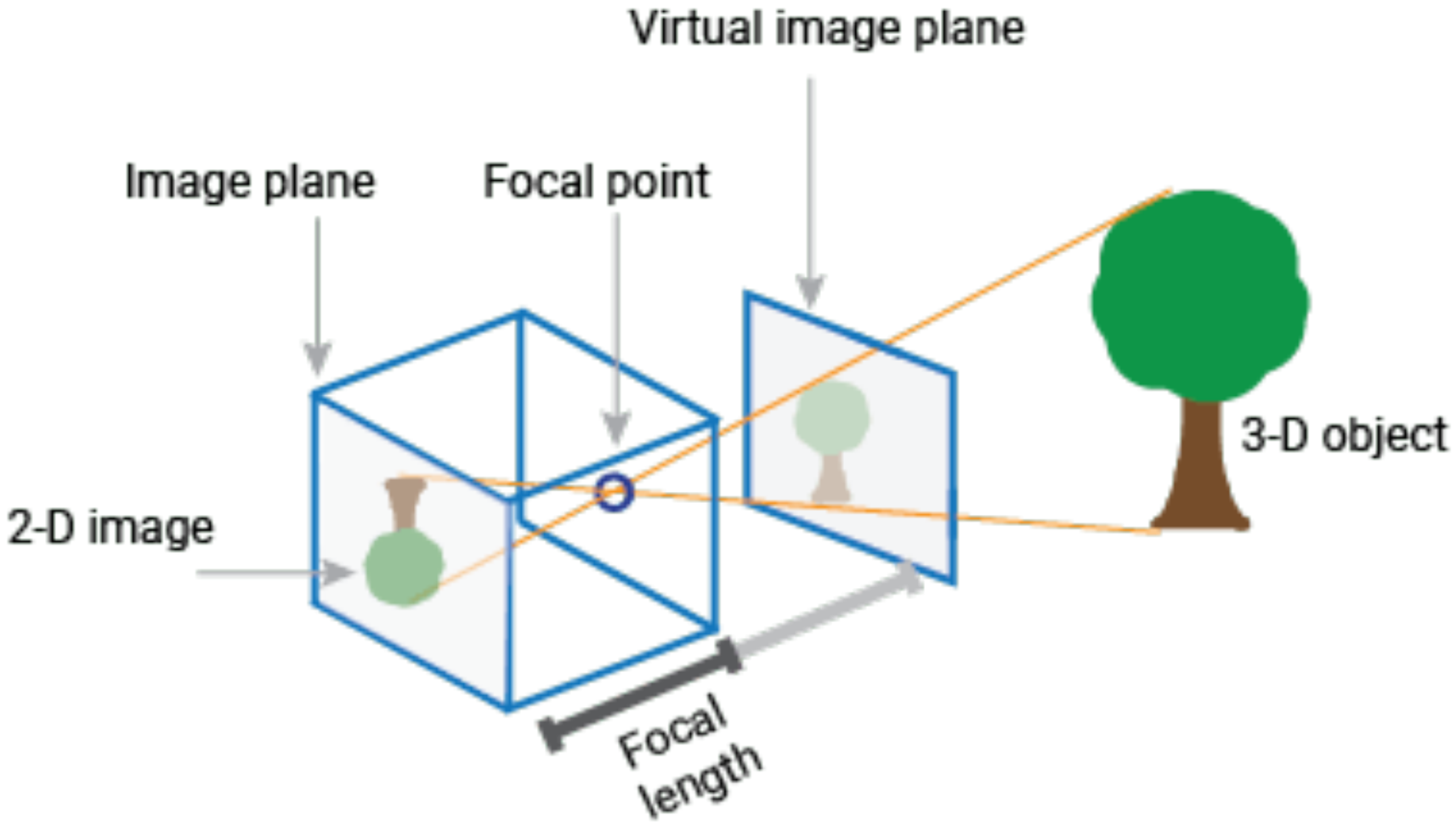
6DOF Pose

Grasp Motion Planning

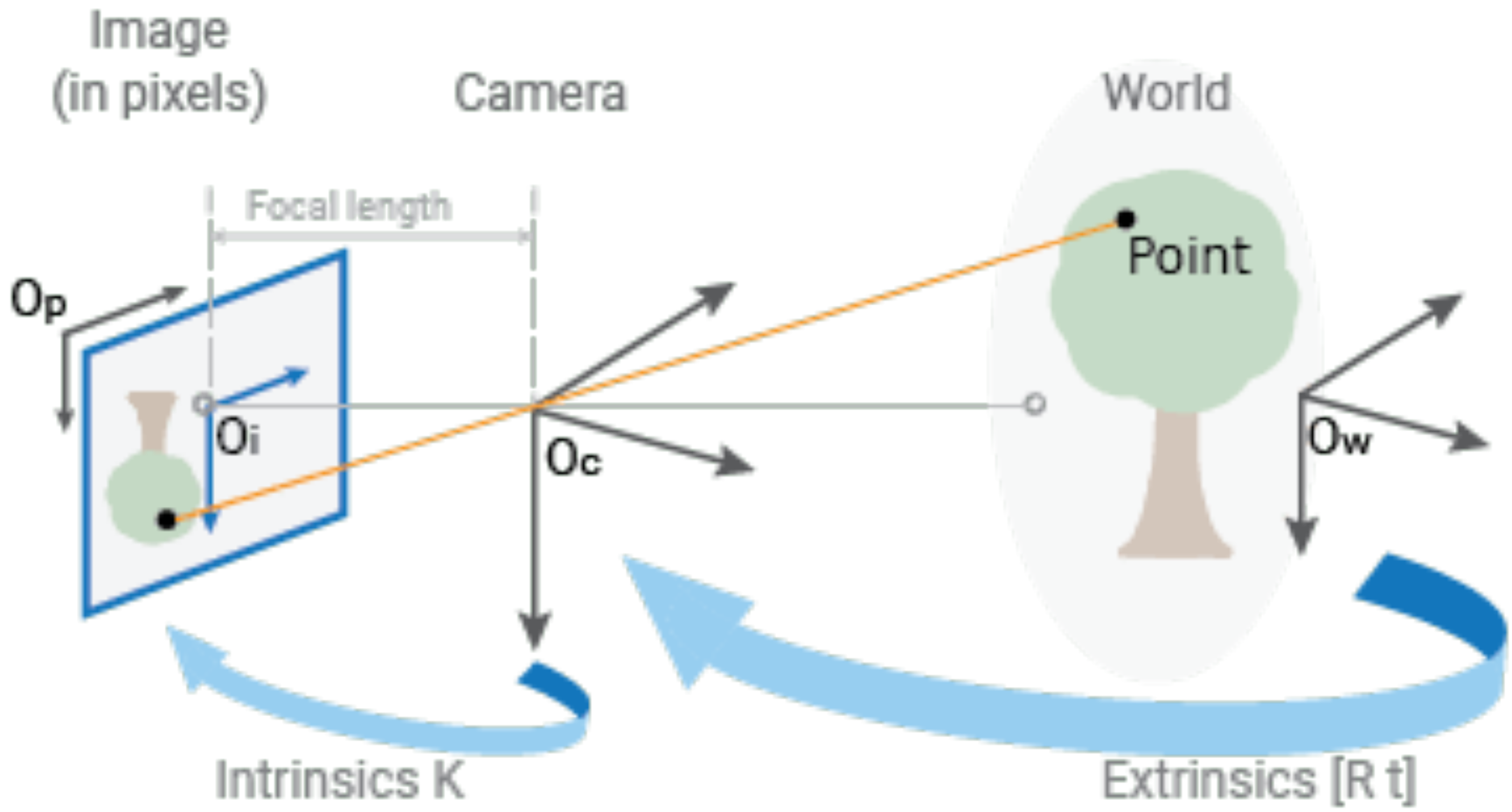
Camera Calibration



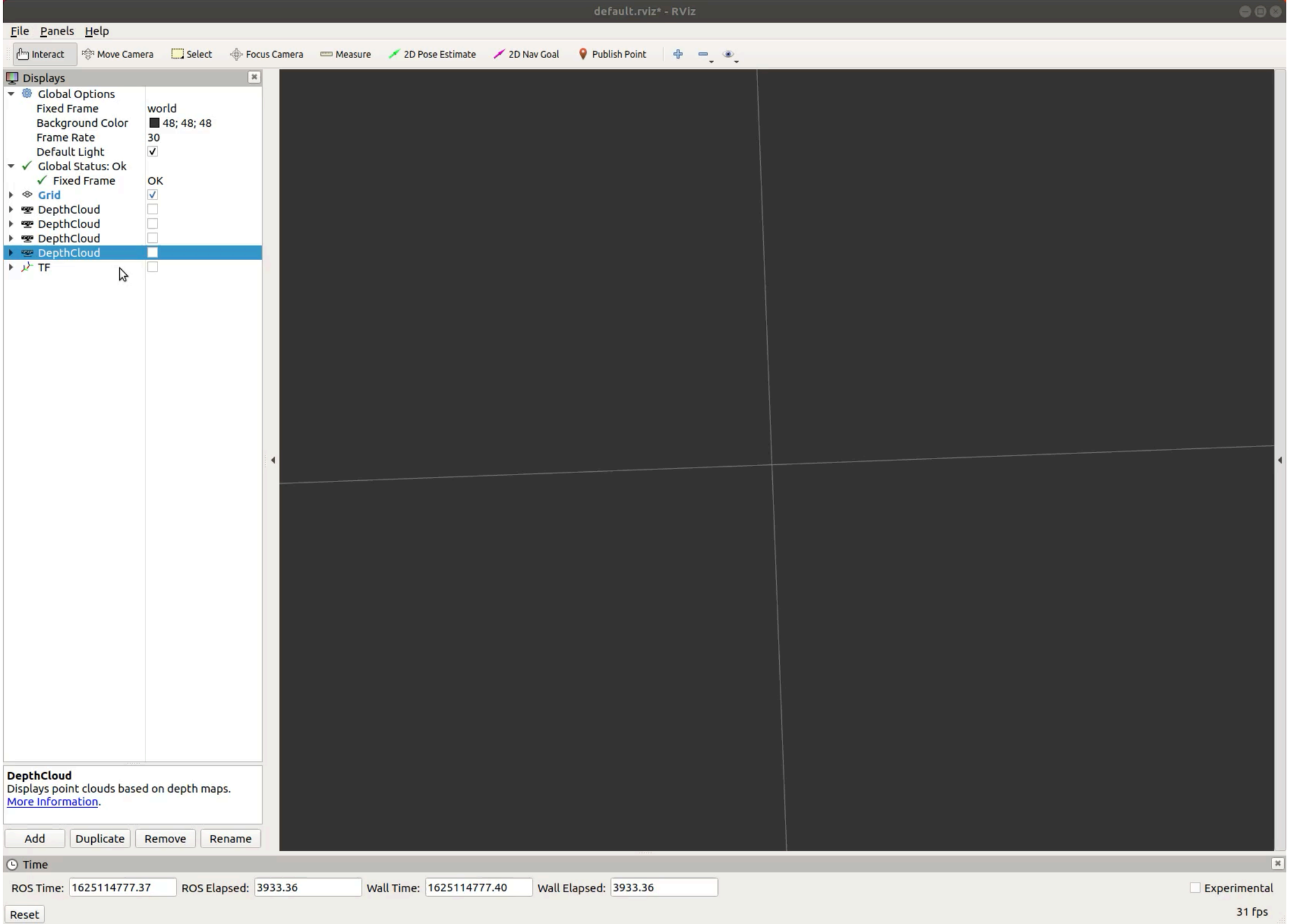
Camera Calibration



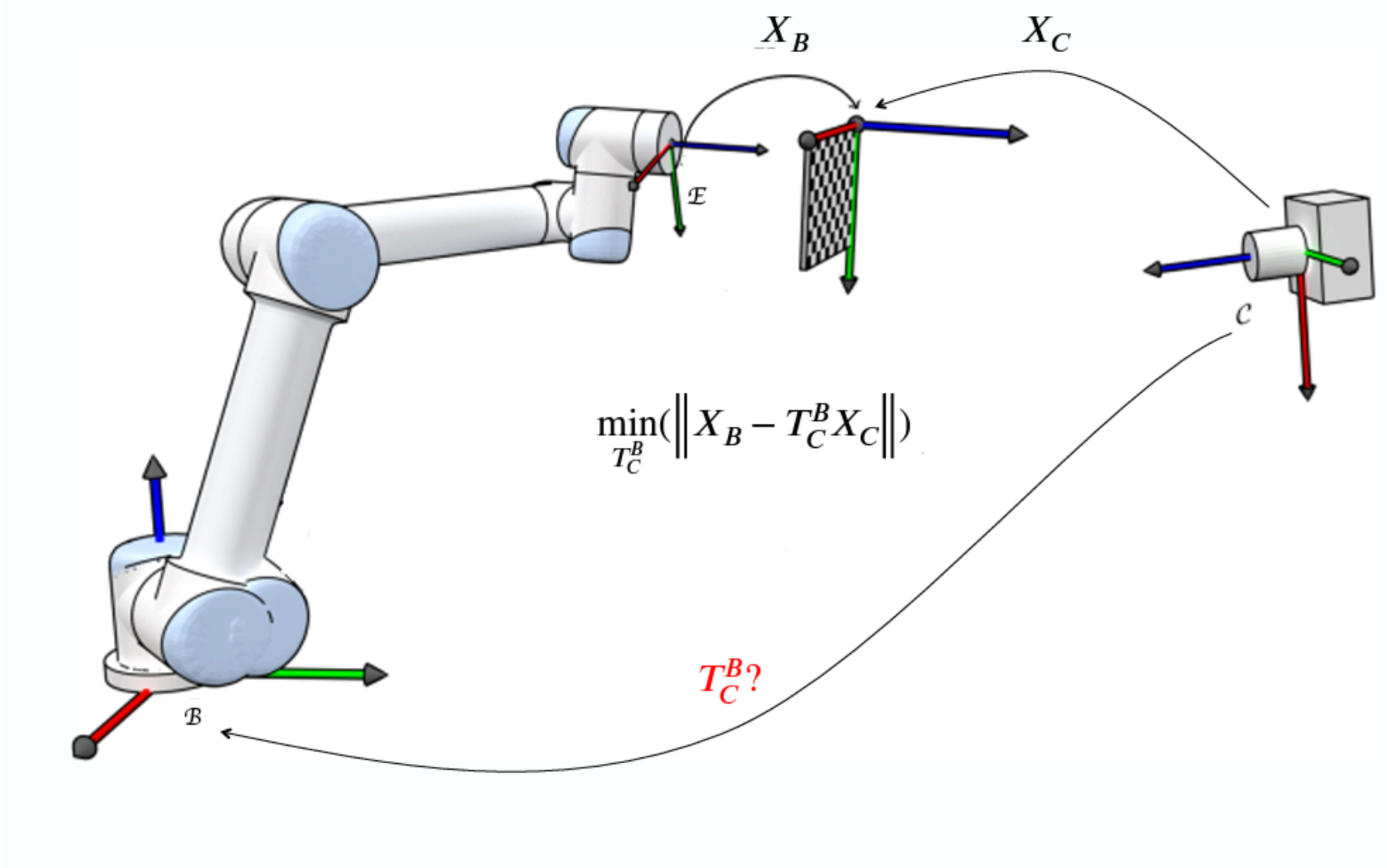
Camera Calibration



Camera Calibration

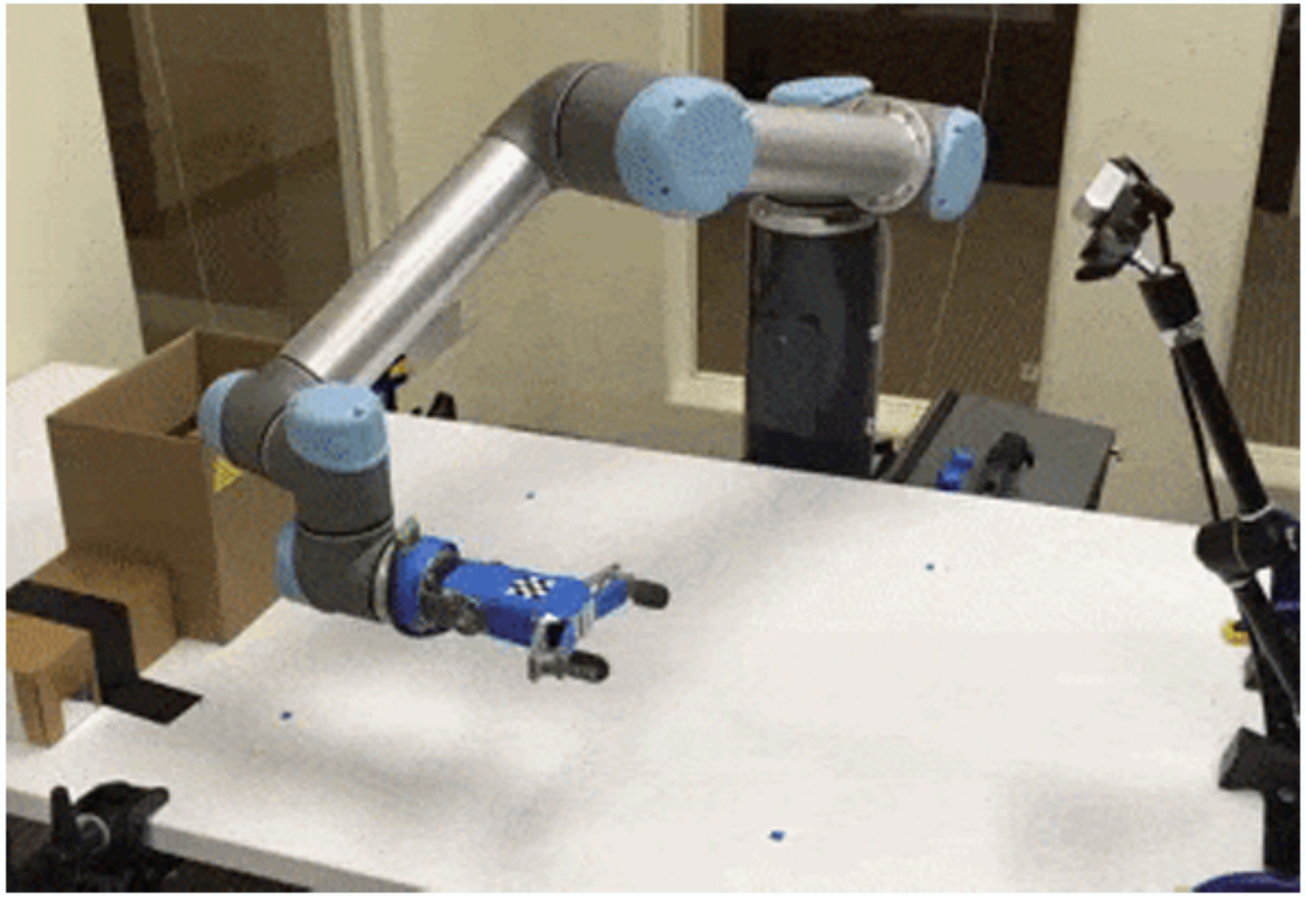

default.rviz* - RViz
File
Panels
Help
Interact
Move Camera
Select
Focus Camera
Measure
2D Pose Estimate
2D Nav Goal
Publish Point
Displays
Global Options
Fixed Frame
world
Background Color
48; 48; 48
Frame Rate
30
Default Light
Global Status: Ok
Fixed Frame
OK
Grid
DepthCloud
DepthCloud
DepthCloud
DepthCloud
TF
DepthCloud
Displays point clouds based on depth maps.
More Information.
Add
Duplicate
Remove
Rename
Time
ROS Time:
1625114777.37
ROS Elapsed:
3933.36
Wall Time:
1625114777.40
Wall Elapsed:
3933.36
Experimental
Reset
31 fps

Camera Calibration



Slide from Dhiraj Gandhi.

Camera Calibration

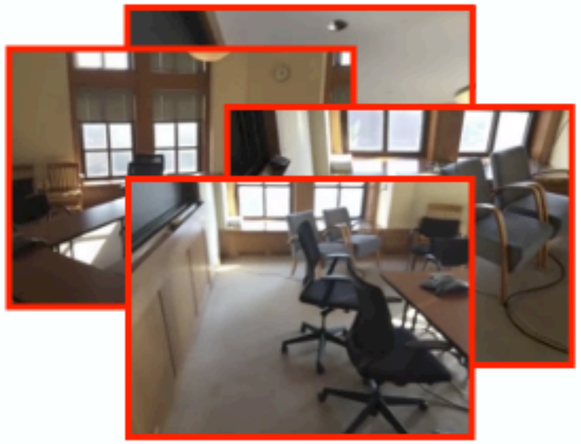


$$\min_{T_C^B} \sum_{i=1}^{n} \left\| X_B^i - T_C^B X_C^i \right\|$$

Observations → State Estimation → Planning → Low-level Controller → Control

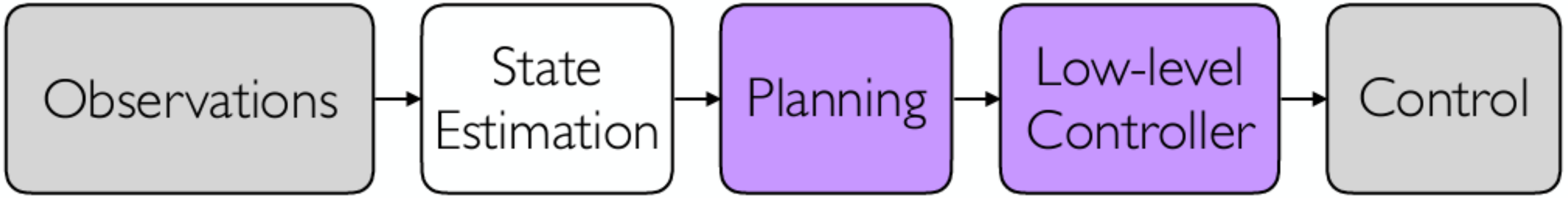
Observed Images

6DOF Pose



Observed Images

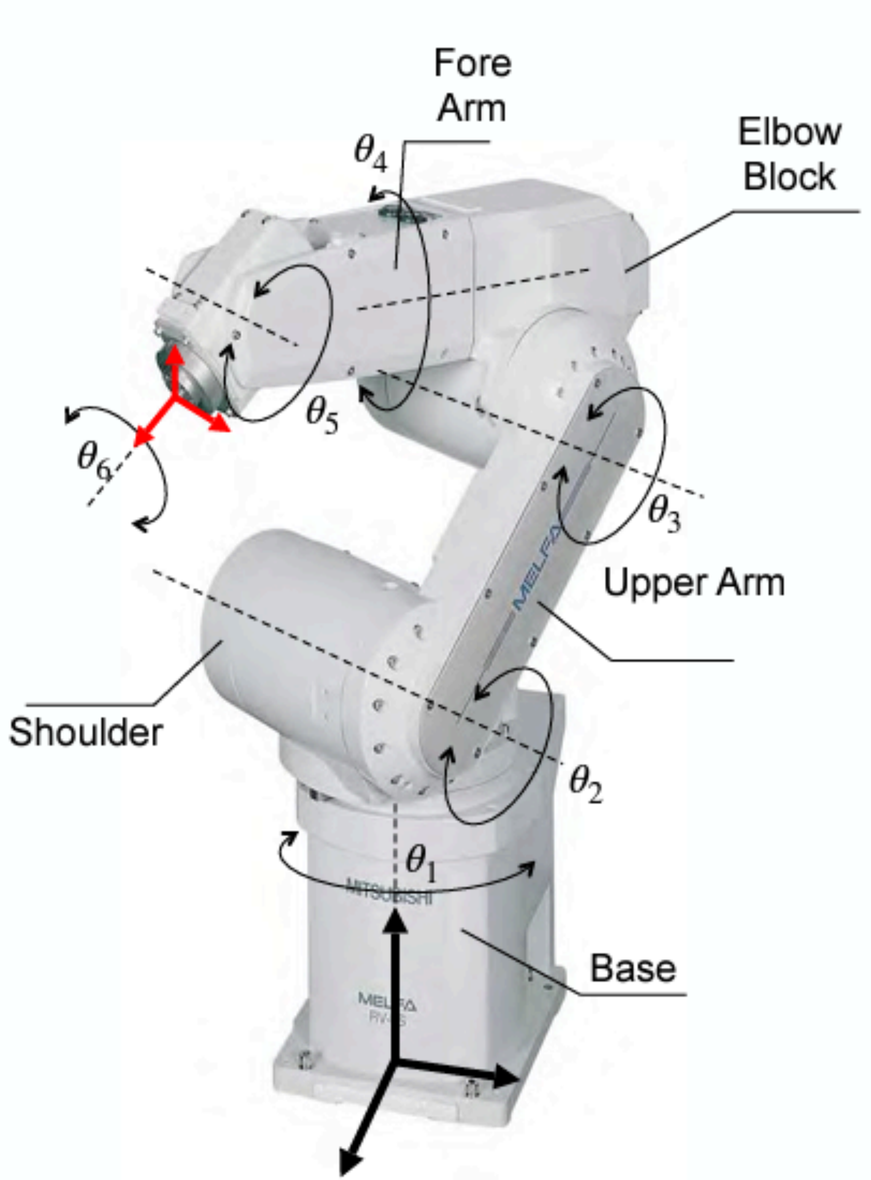
Geometric or Semantic Maps



How to move a robot?

Terminology

- Link
- Joint
- End Effector
- Base
- Sensors

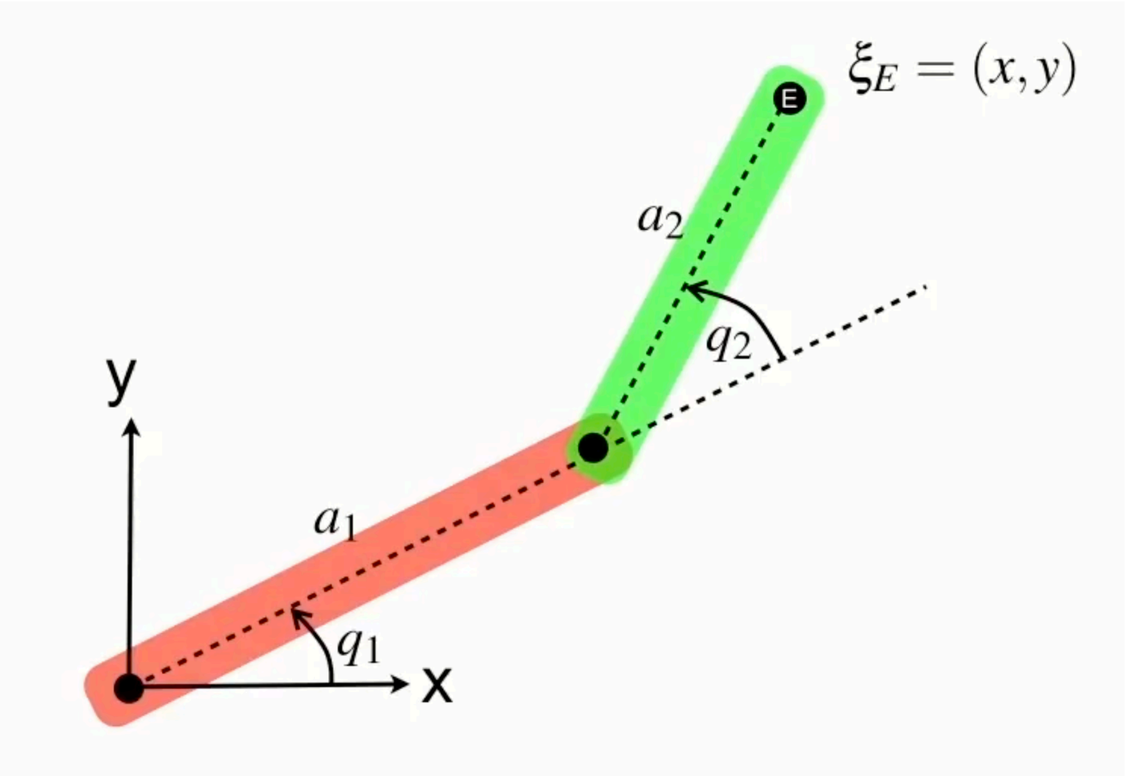


Spaces

- Work Space (or Task Space)
  - This refers to the physical space in which the robot operates. For a robotic arm, the work space might be the volume of space the end-effector (like a gripper or tool) can reach.
- Configuration Space (C-Space)
  - This is a representation of all possible configurations (or positions) of the robot. For robots with multiple joints, the C-space can have multiple dimensions, one for each degree of freedom.

Spaces

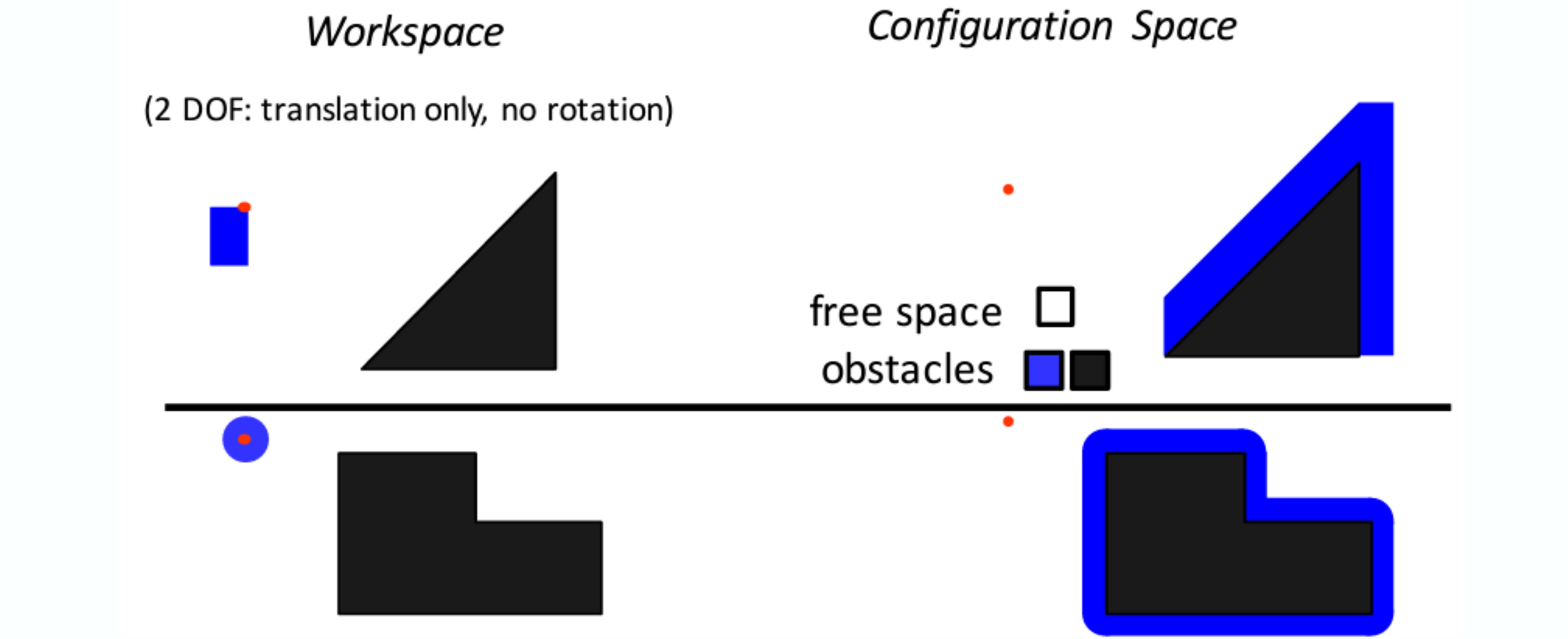
- Link
- Joint
- End Effector
- Base
- Work Space
- Configuration Space



Spaces

- Link
- Joint
- End Effector
- Base
- Work Space
- Configuration Space

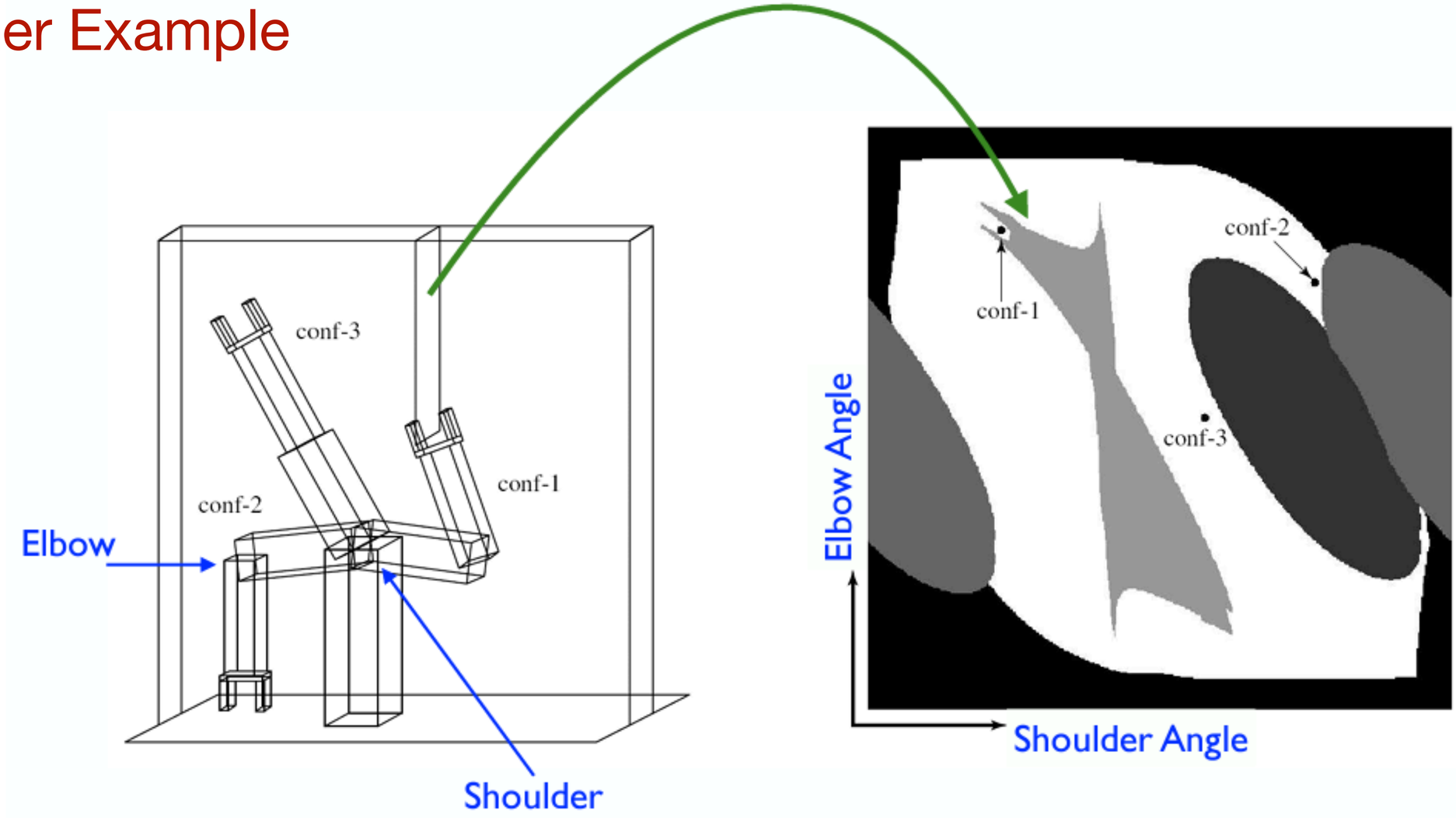
Work space obstacles —> Configuration space obstacles



Slide from Pieter Abbeel.

Configuration Space

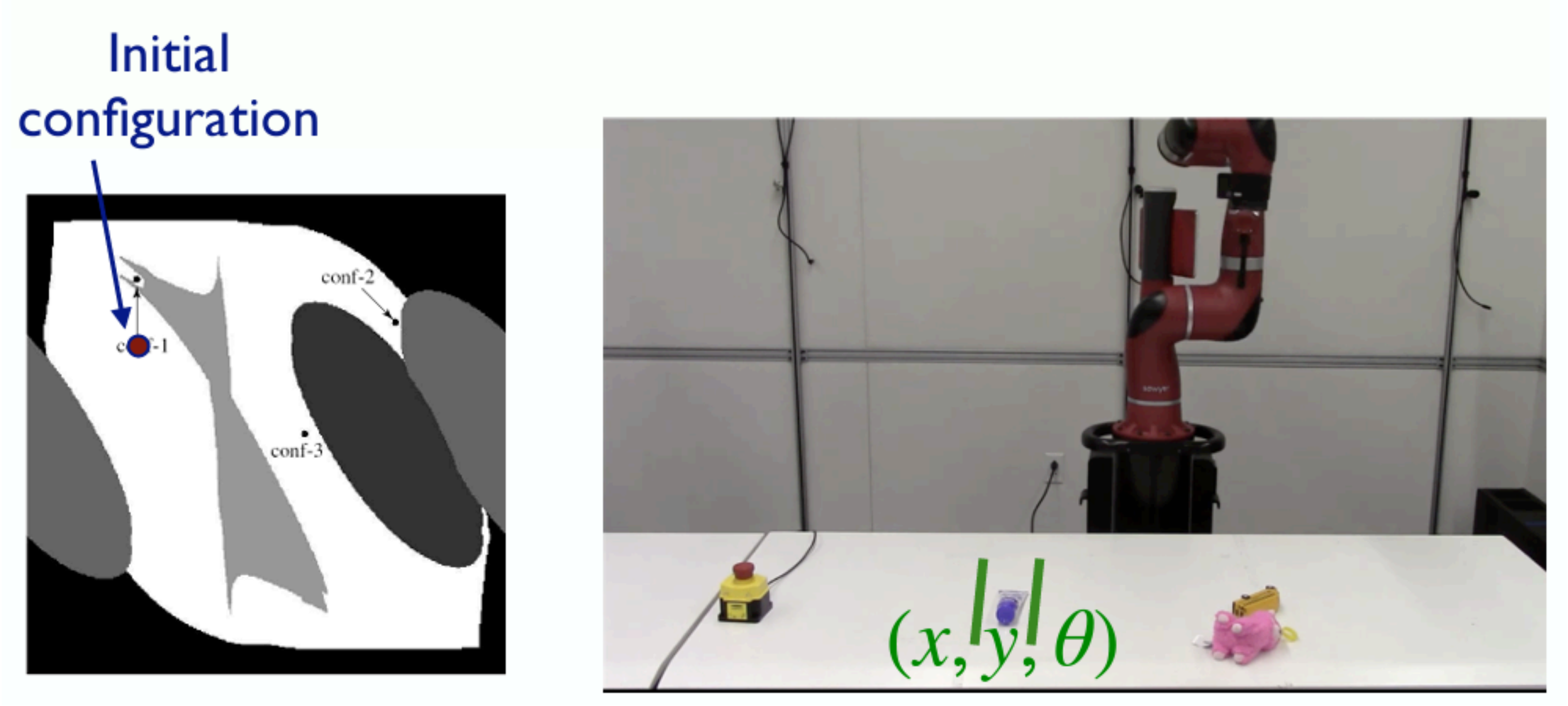
Another Example



Slide from Pieter Abbeel.

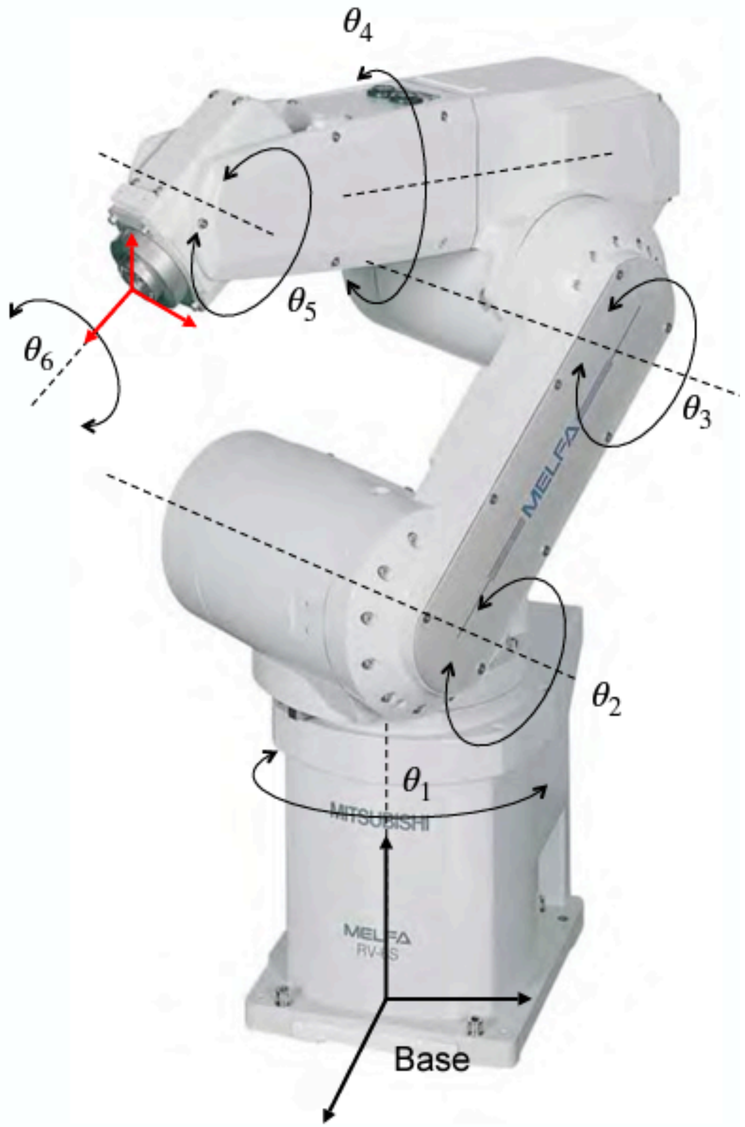
How to move your robot?

Task space to Configuration space



Configuration Space to Task Space

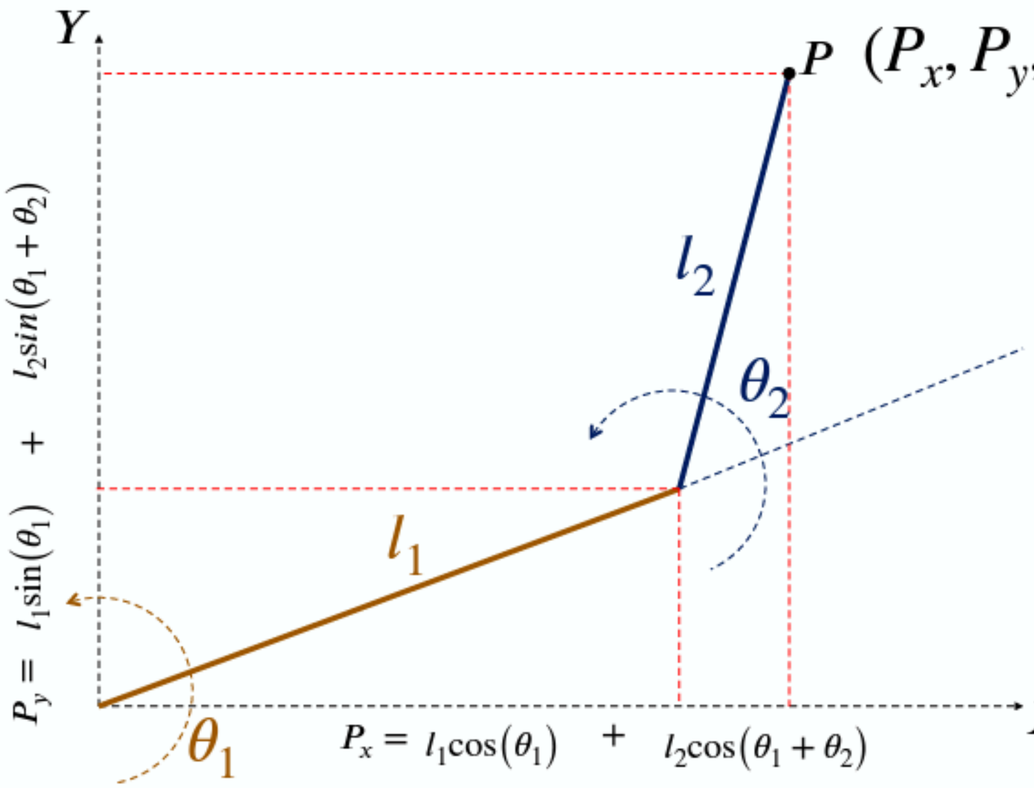
Forward Kinematics



Slide from Dhiraj Gandhi.

Configuration Space to Task Space

Forward Kinematics



$$P_x = l_1 cos(\theta_1) + l_2 cos(\theta_1 + \theta_2)$$

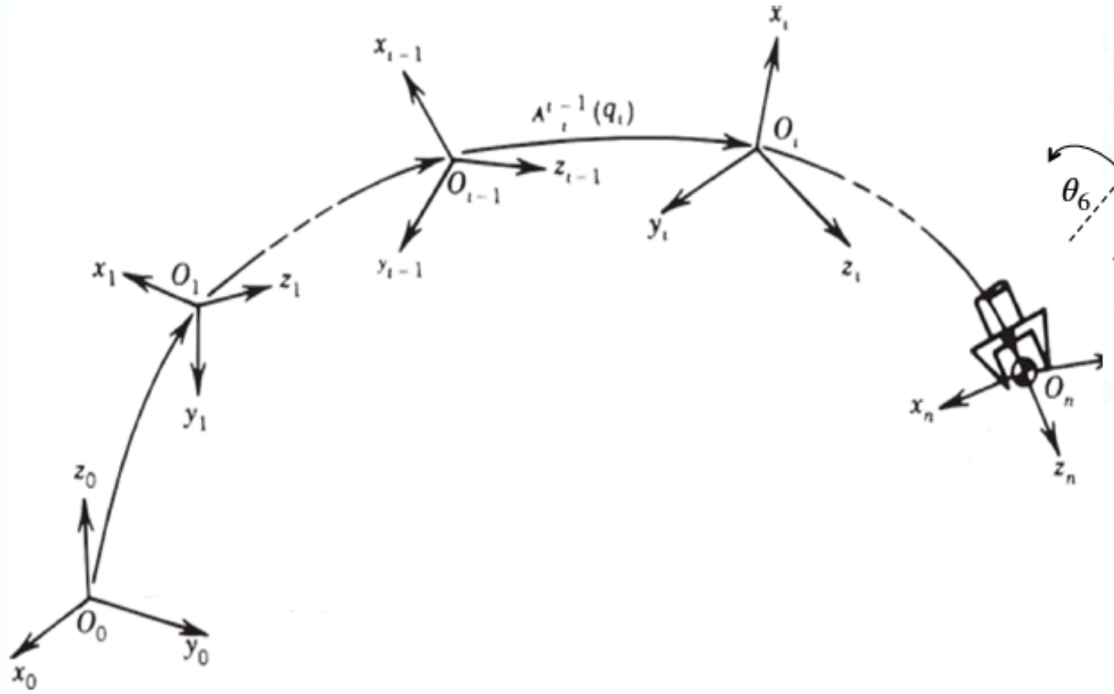
$$P_y = l_2 sin(\theta_1) + l_2 sin(\theta_1 + \theta_2)$$

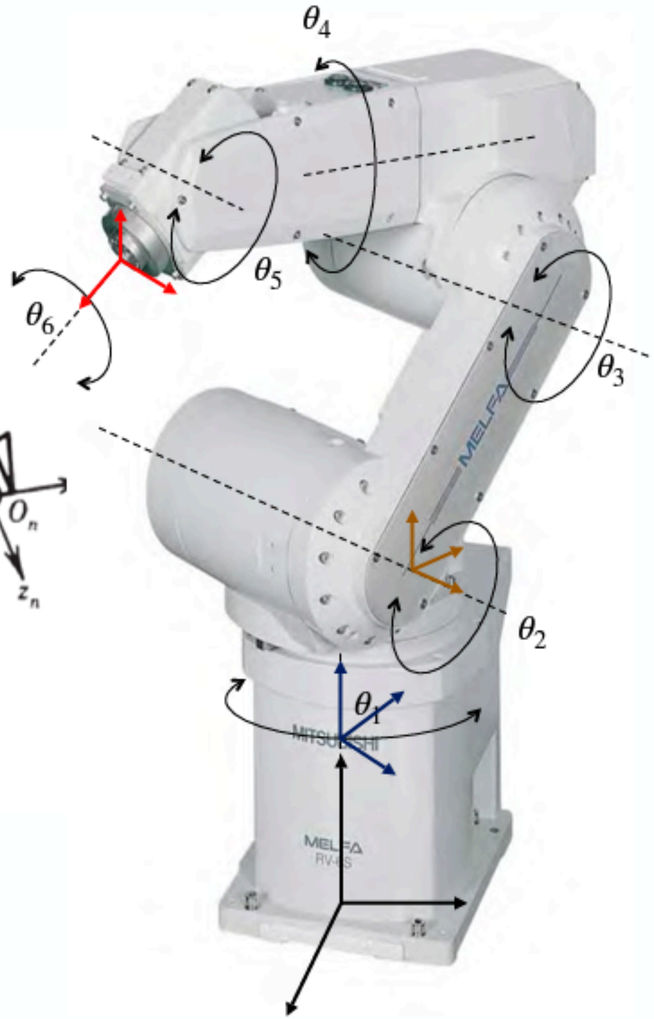
$$P_\theta = \theta_1 + \theta_2$$

Slide from Dhiraj Gandhi.

Configuration Space to Task Space

Forward Kinematics

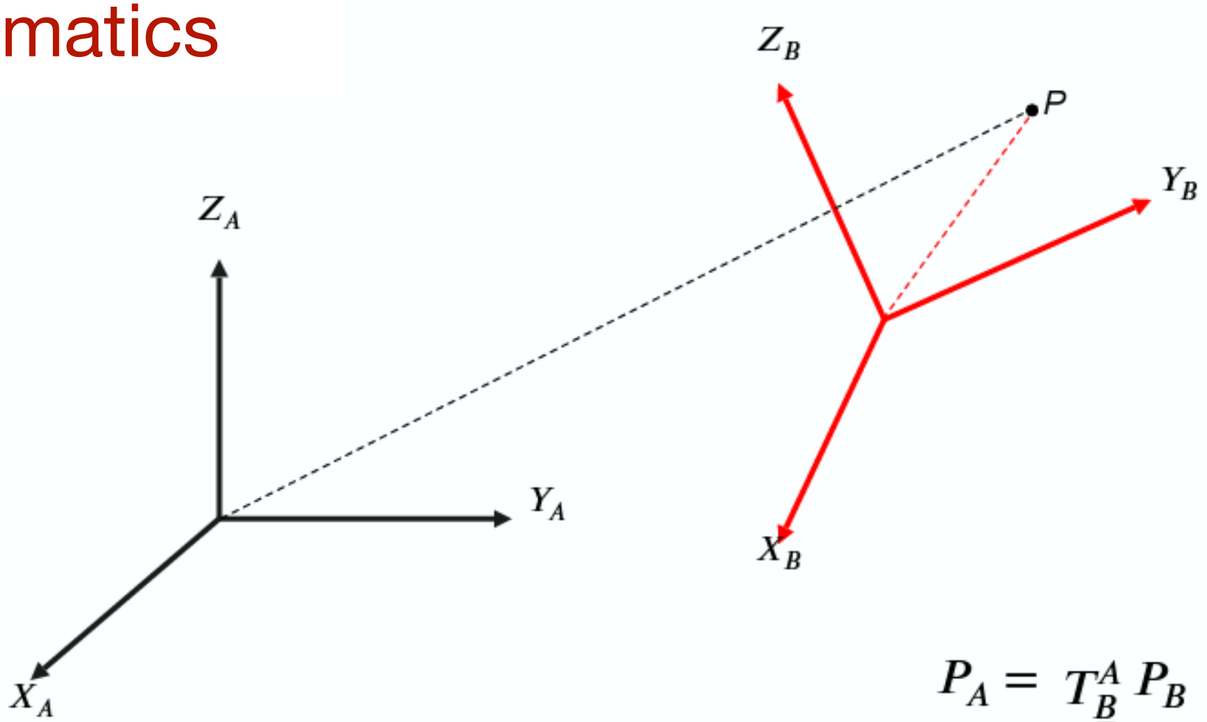




Slide from Dhiraj Gandhi.

Configuration Space to Task Space

Forward Kinematics



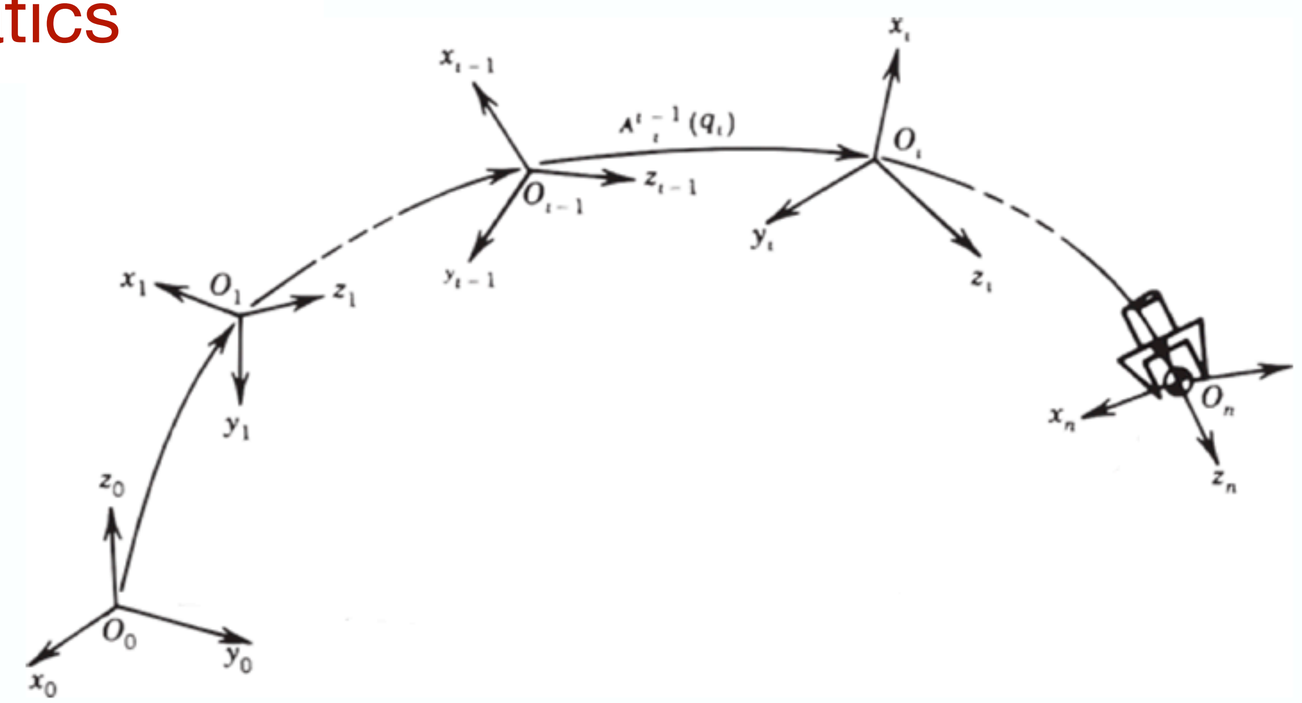
$$P_A = T_B^A P_B$$

$$T_B^A = \begin{bmatrix} r_{11} & r_{21} & r_{31} & \Delta x \\ r_{12} & r_{22} & r_{32} & \Delta y \\ r_{13} & r_{23} & r_{33} & \Delta z \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Slide from Dhiraj Gandhi.

Configuration Space to Task Space

Forward Kinematics

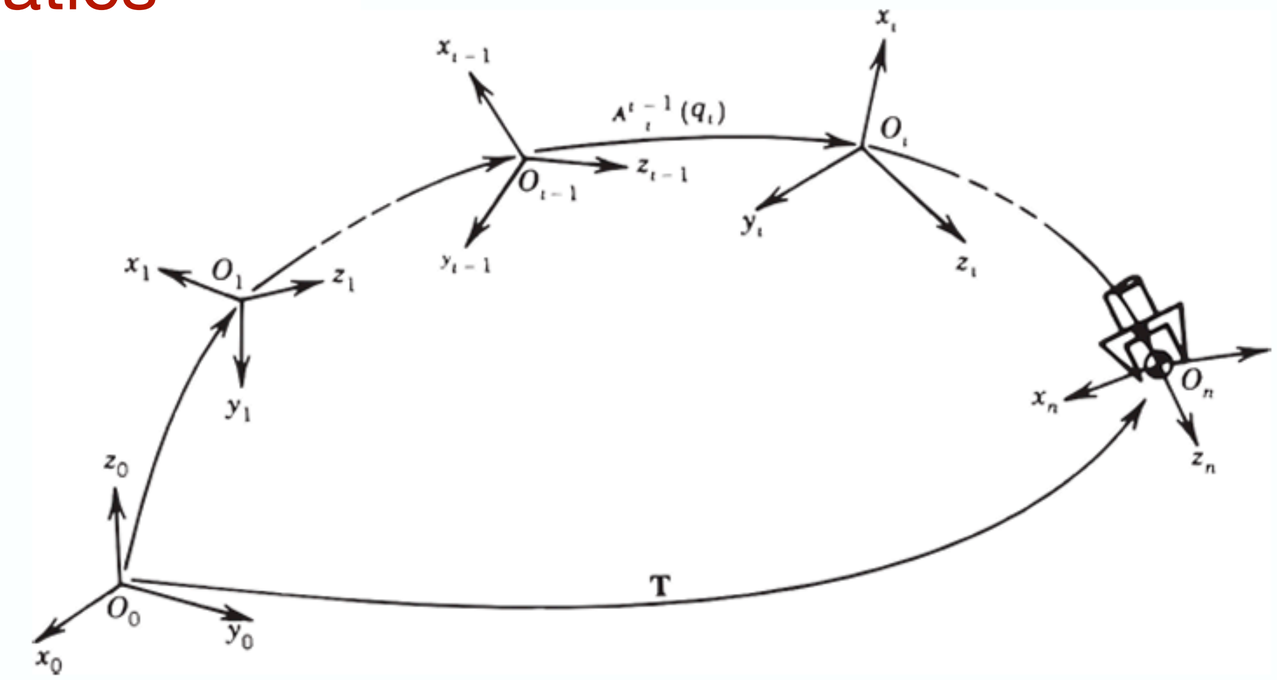


$$T_1^0(\theta_1)\, T_2^1(\theta_2)\ldots \quad T_{n-1}^{n-2}(\theta_n) \quad T_n^{n-1}(\theta_n)$$

Slide from Dhiraj Gandhi.

Configuration Space to Task Space

Forward Kinematics



$$T = T_1^0(\theta_1)\, T_2^1(\theta_2) \ldots \quad T_{n-1}^{n-2}(\theta_n) \quad T_n^{n-1}(\theta_n)$$

$$= \begin{bmatrix} r_{11} & r_{21} & r_{31} & \Delta x \\ r_{12} & r_{22} & r_{32} & \Delta y \\ r_{13} & r_{23} & r_{33} & \Delta z \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

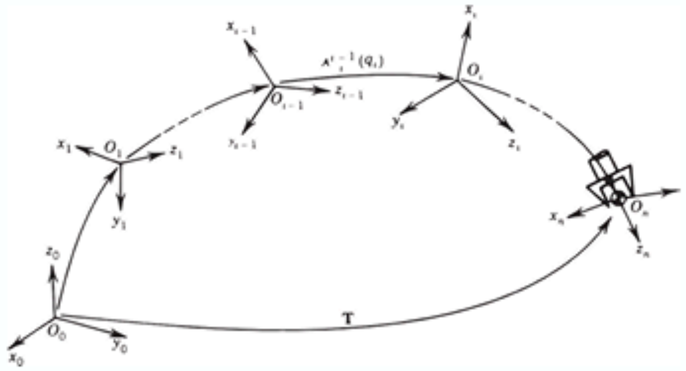
Maps configuration space to work space

$$x = f(\theta) = f(\theta_1, \theta_2, .., \theta_{n-1}, \theta_n)$$

Slide adapted from Dhiraj Gandhi.

Configuration Space to Task Space

Inverse Kinematics



$$x = f(\theta)$$

Maps configuration space to work space

Numerical IK

Solve for θ_d in:

$$x_d - f(\theta_d) = 0$$

Analytical IK

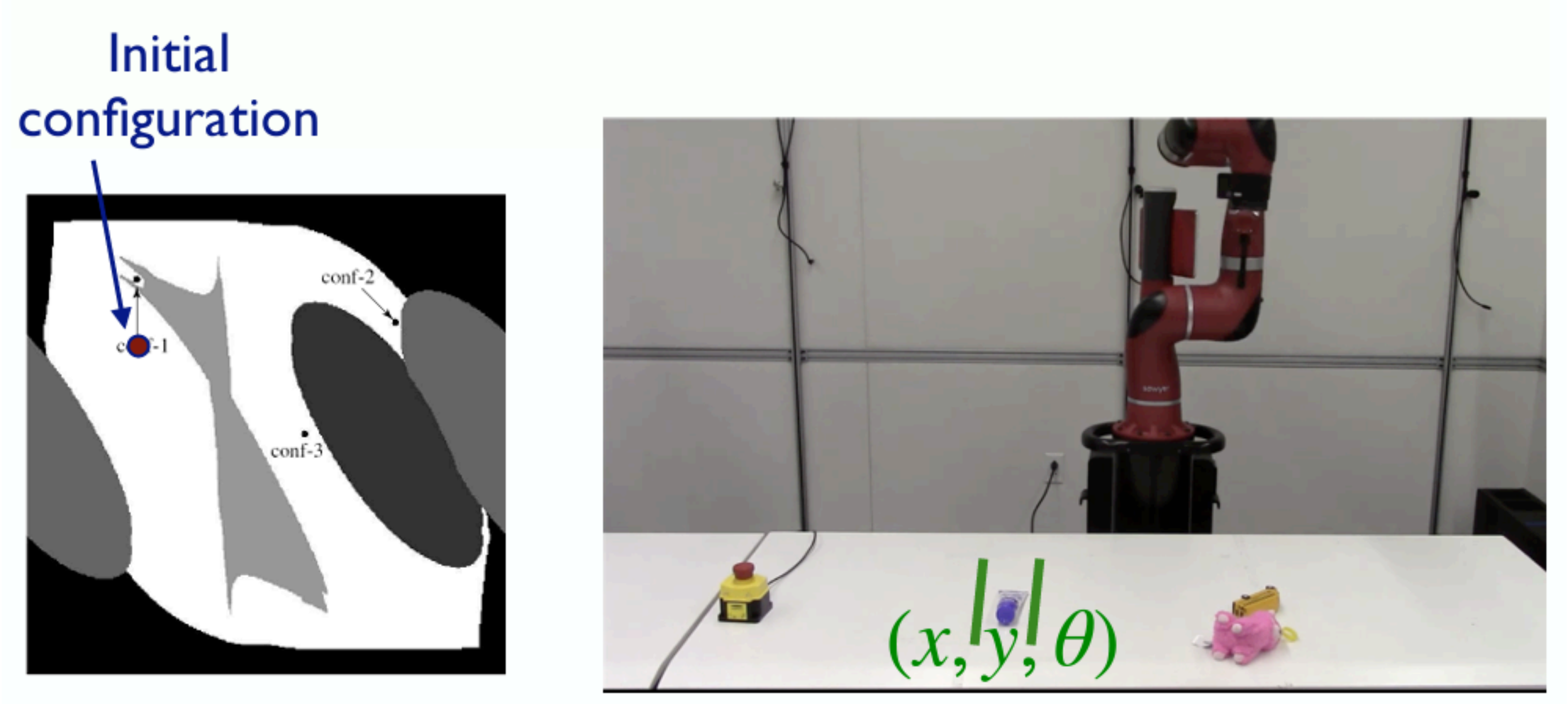
- Robot Specific
- Fast
- Characterize the solution space

Find configuration(s) that map to a given work space point

Slide adapted from Dhiraj Gandhi, Modern Robotics

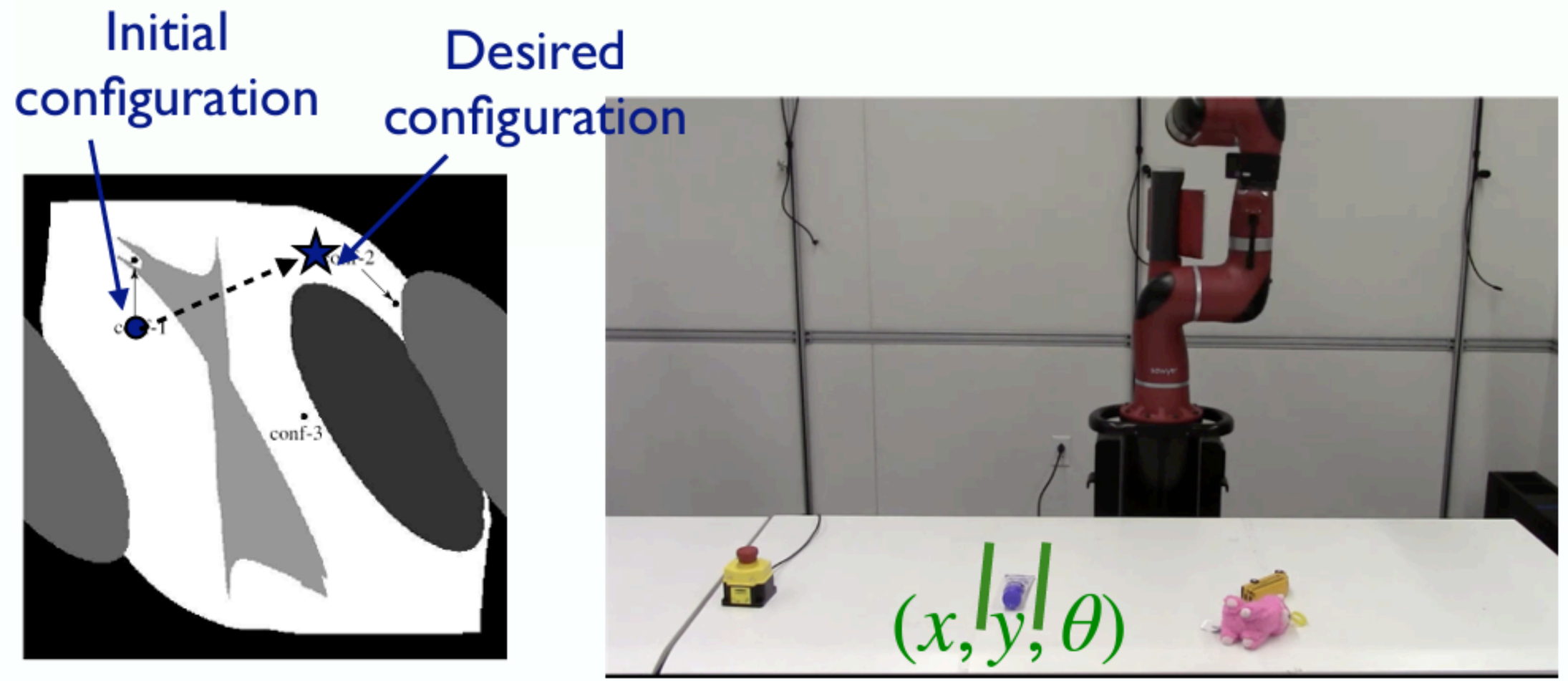
How to move your robot?

- Task space to Configuration space

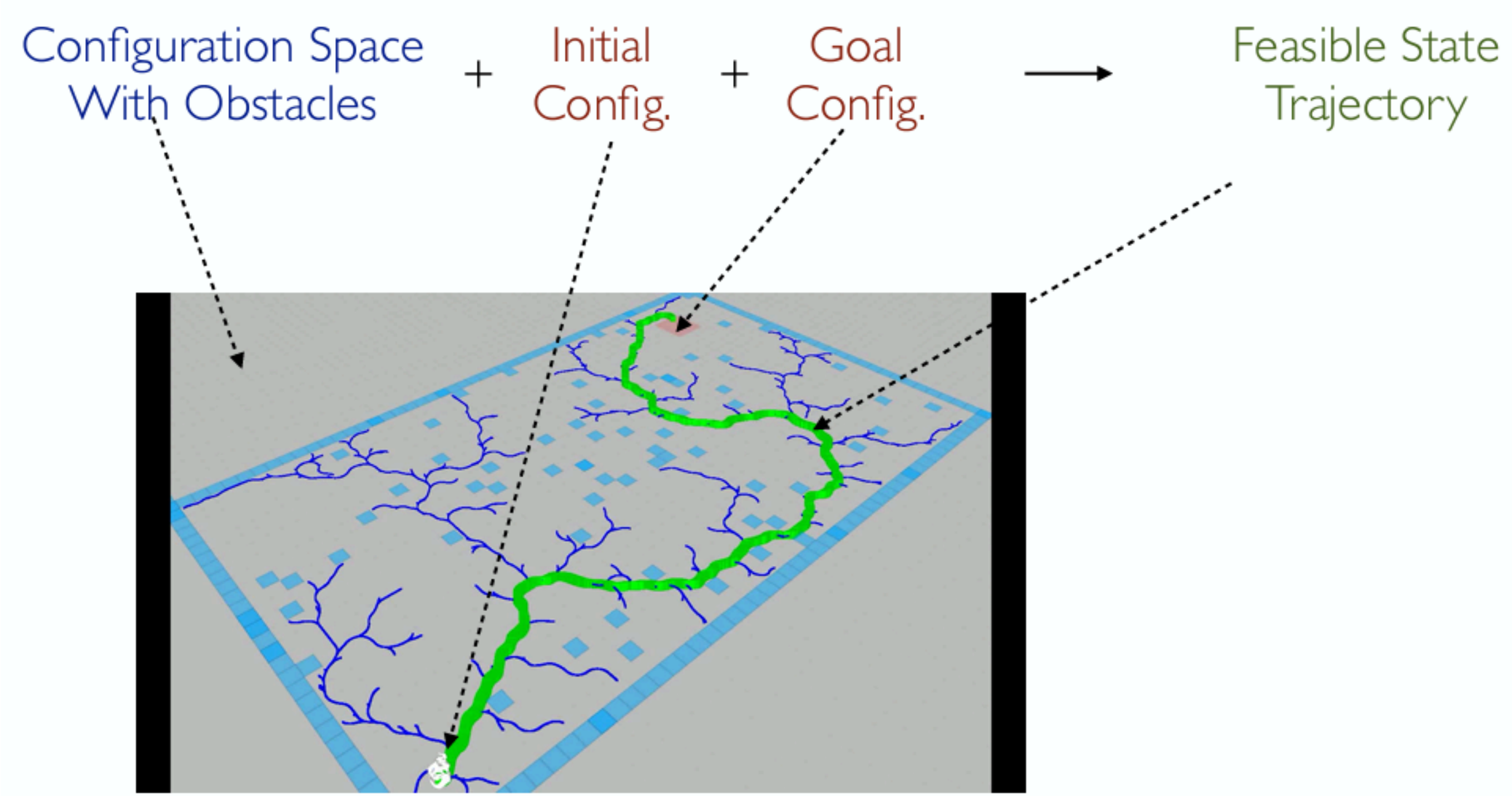


How to move your robot?

- Task space to Configuration space
- Configuration space trajectory



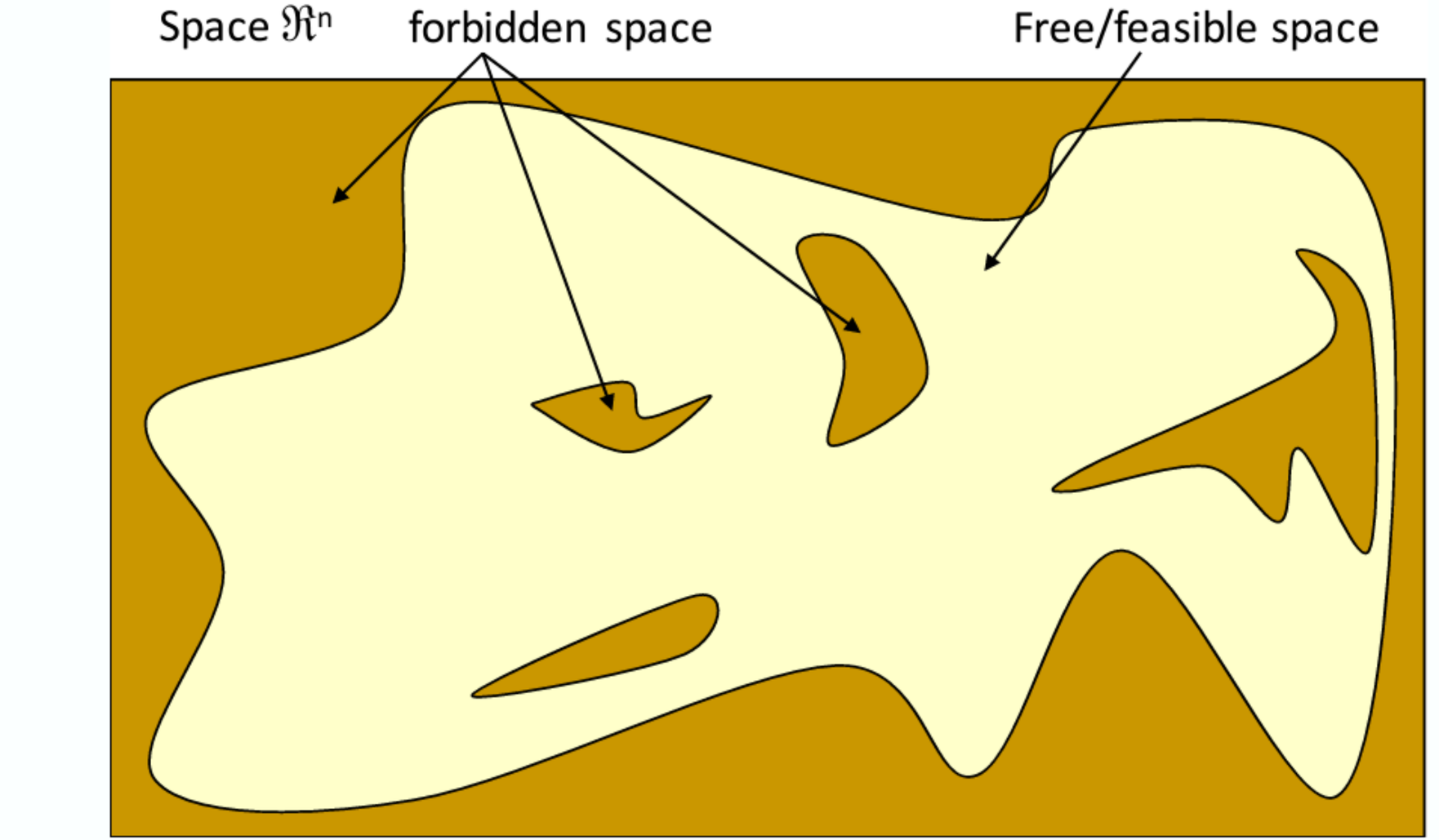
Path Planning



Path Planning

- Complete Methods
- Grid Methods
- Sampling Methods
- Potential Fields
- Trajectory Optimization

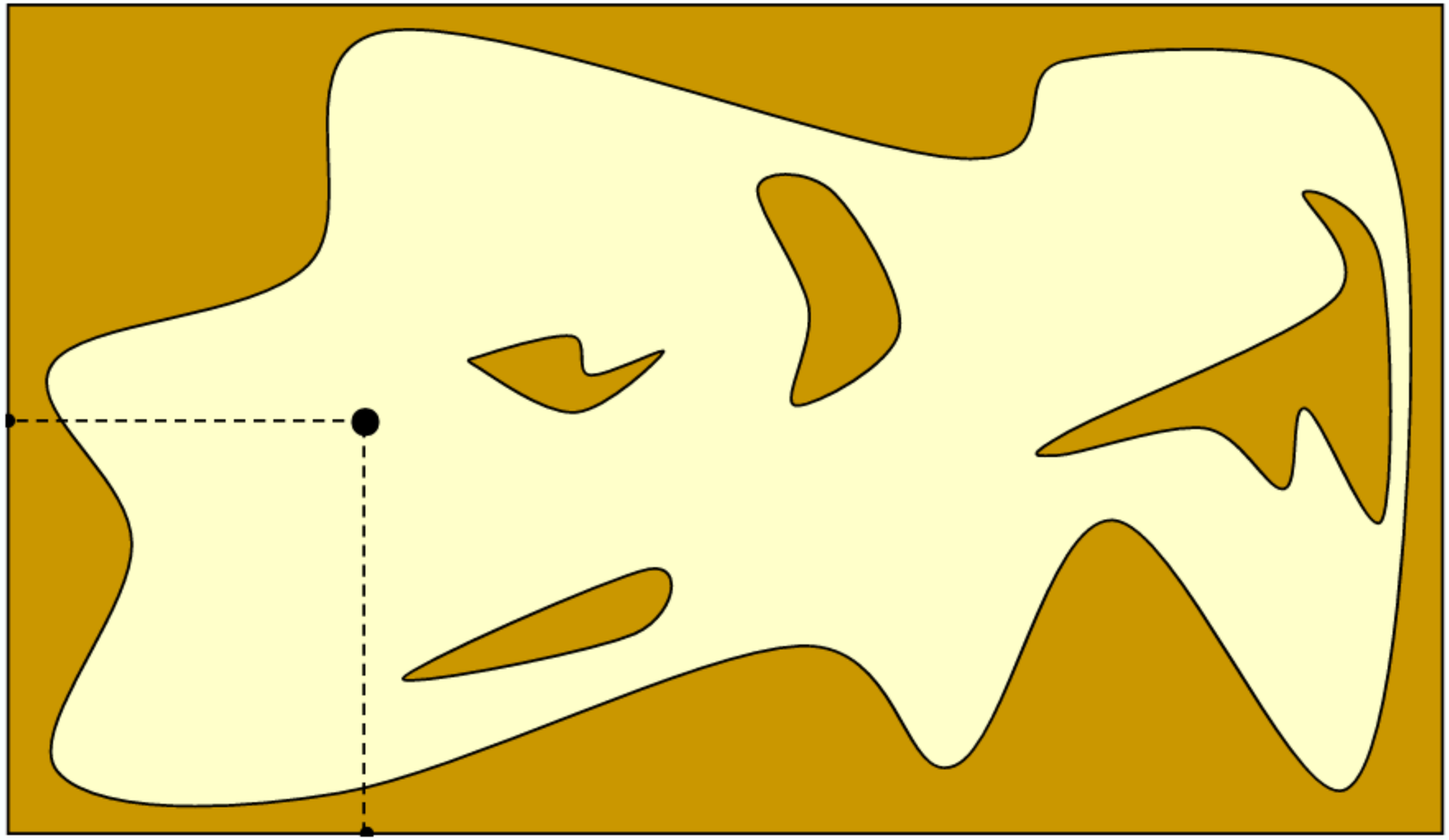
Probabilistic Roadmaps



Slide from Pieter Abbeel.

Probabilistic Roadmaps

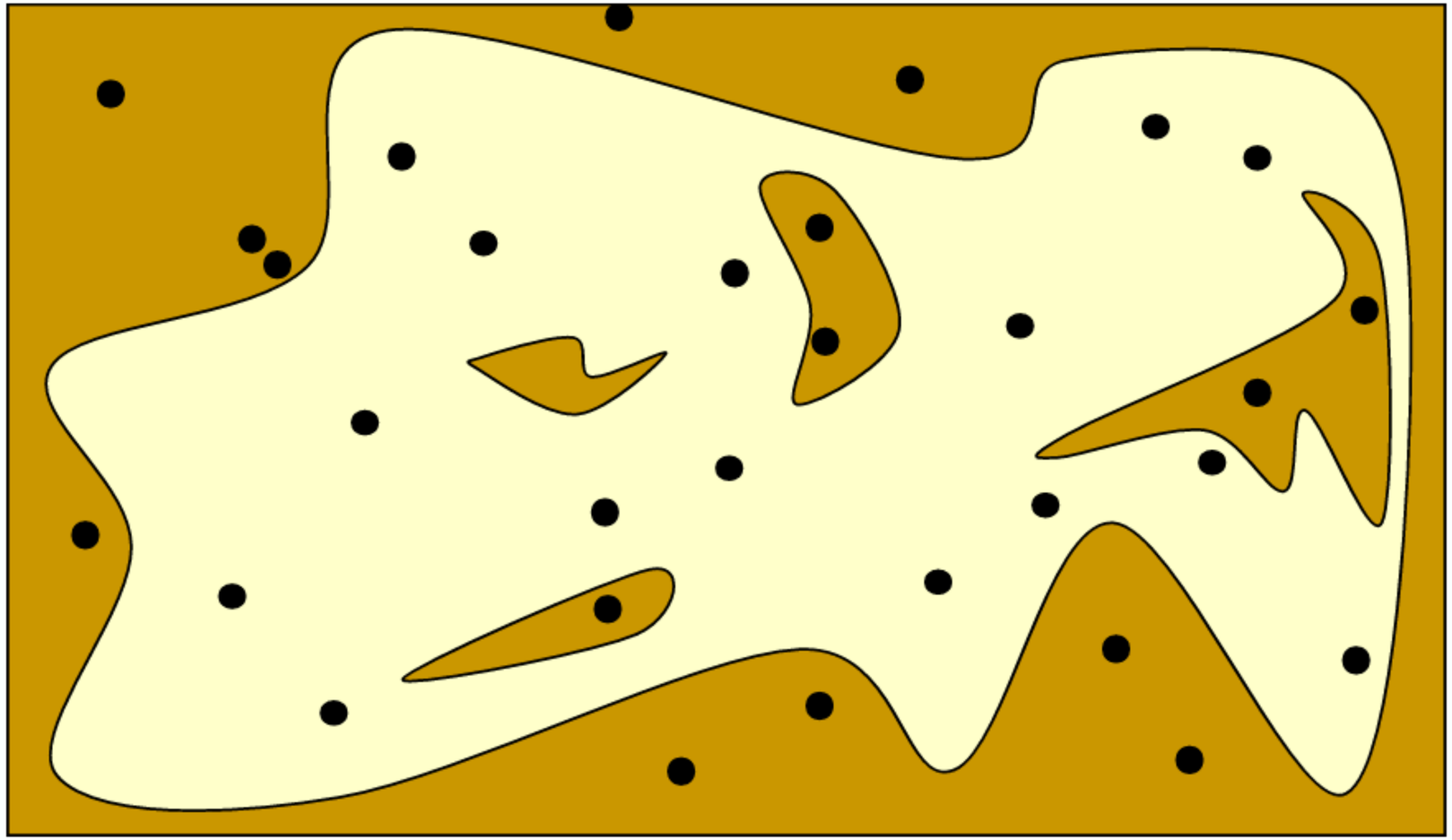
Randomly Sample Configurations



Slide from Pieter Abbeel.

Probabilistic Roadmaps

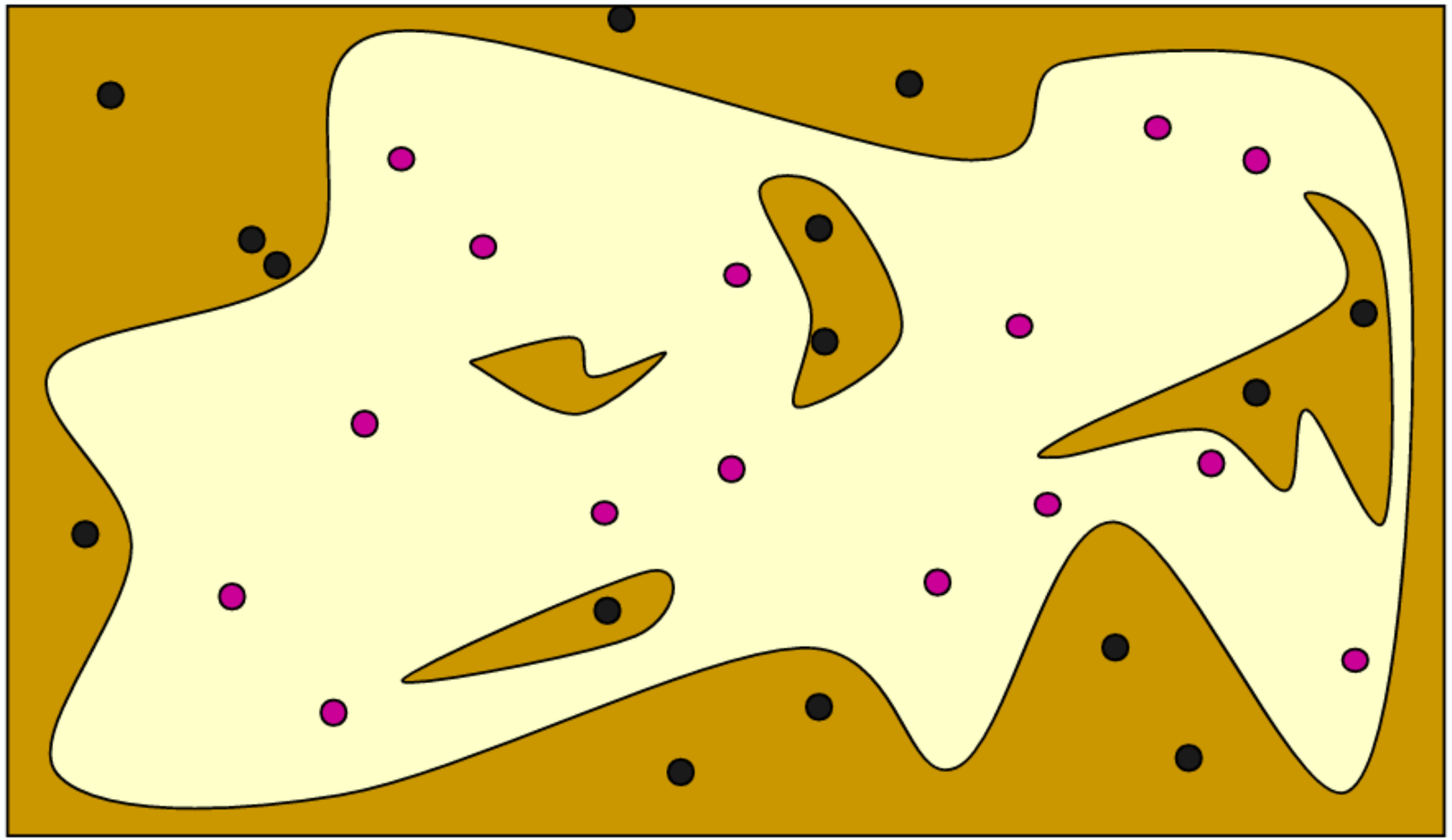
Randomly Sample Configurations



Slide from Pieter Abbeel.

Probabilistic Roadmaps

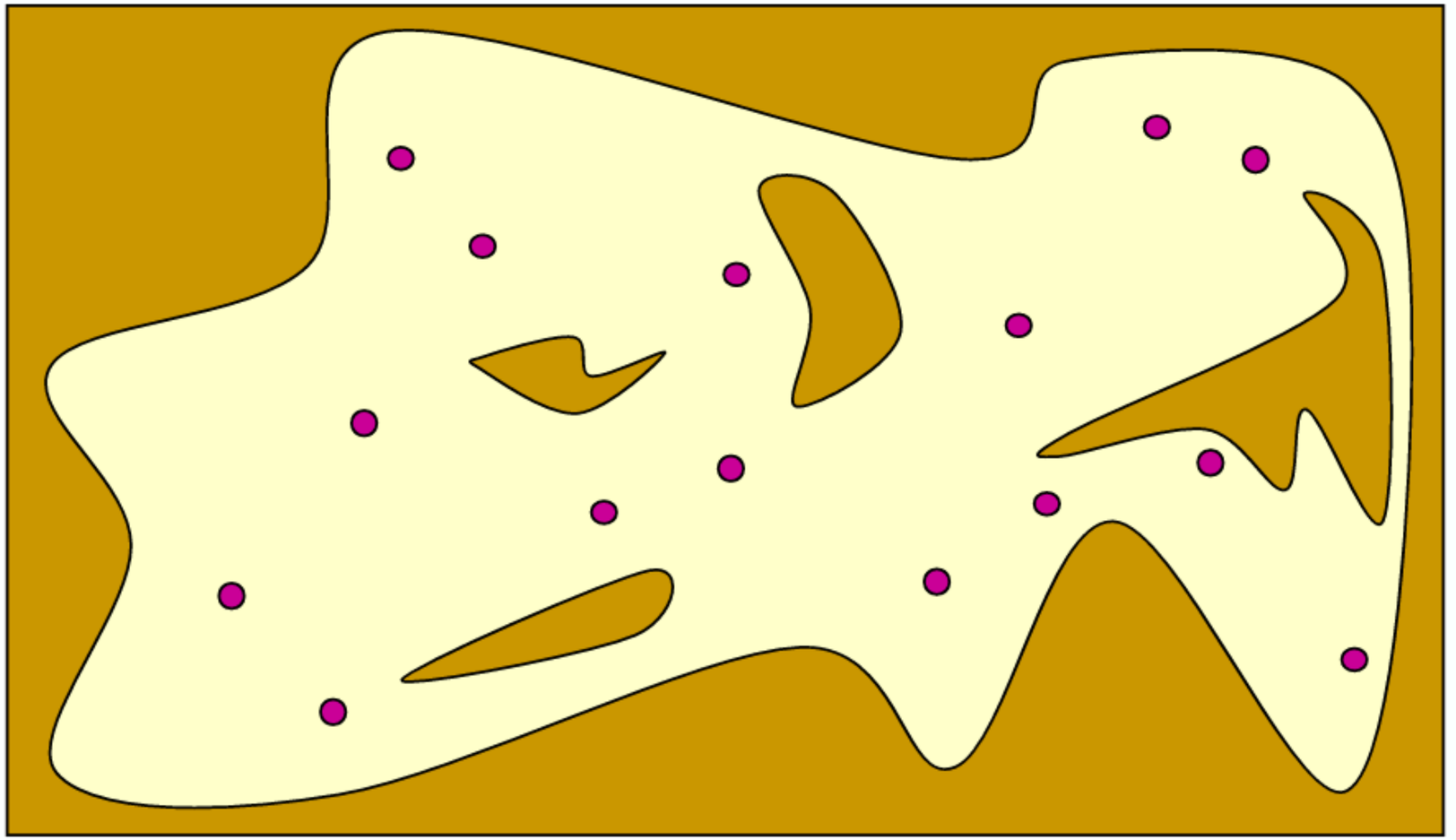
Test Sampled Configurations for Collisions



Slide from Pieter Abbeel.

Probabilistic Roadmaps

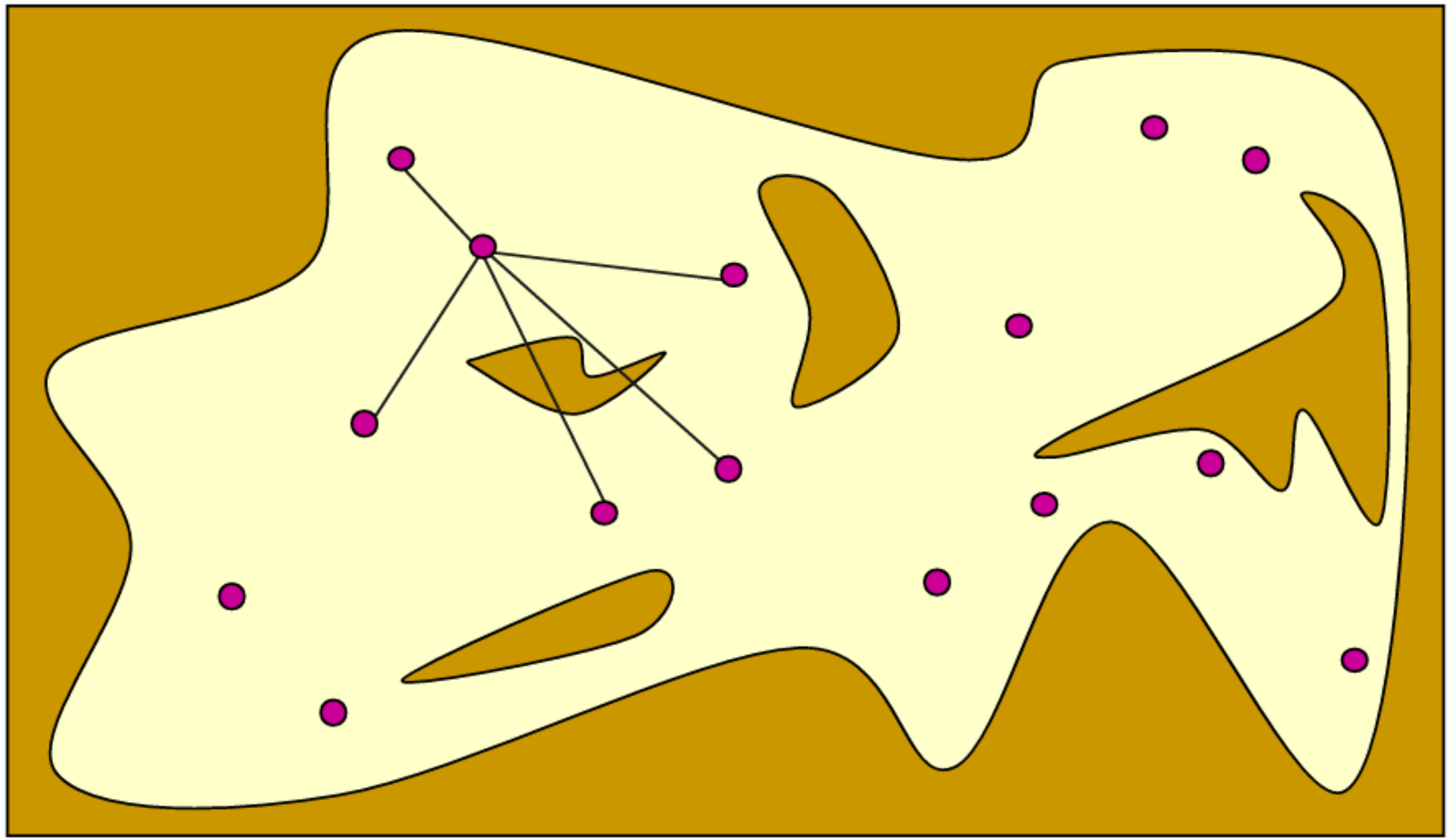
The collision-free configurations are retained as milestones



Slide from Pieter Abbeel.

Probabilistic Roadmaps

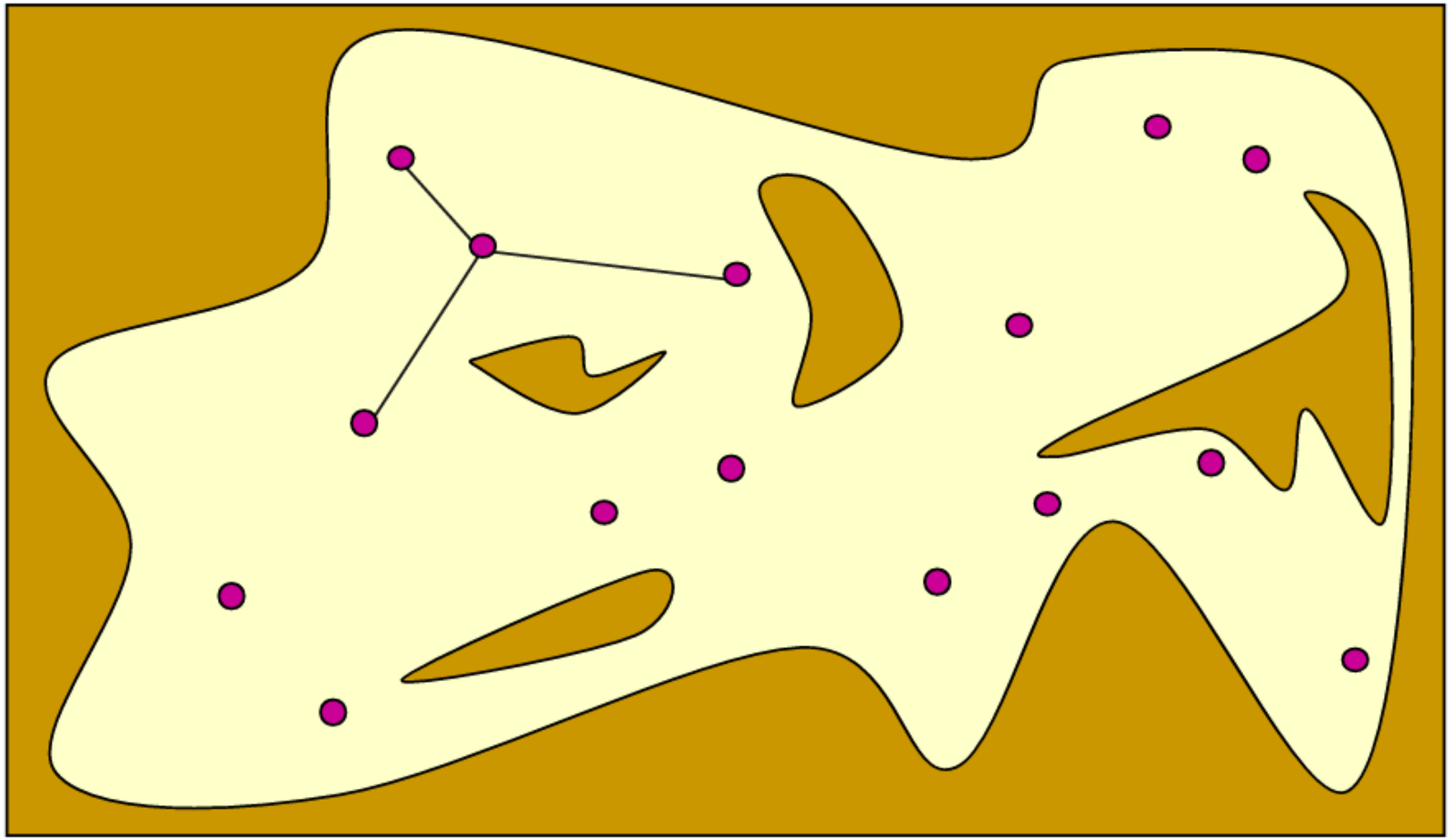
Each milestone is linked by straight paths to its nearest neighbors



Slide from Pieter Abbeel.

Probabilistic Roadmaps

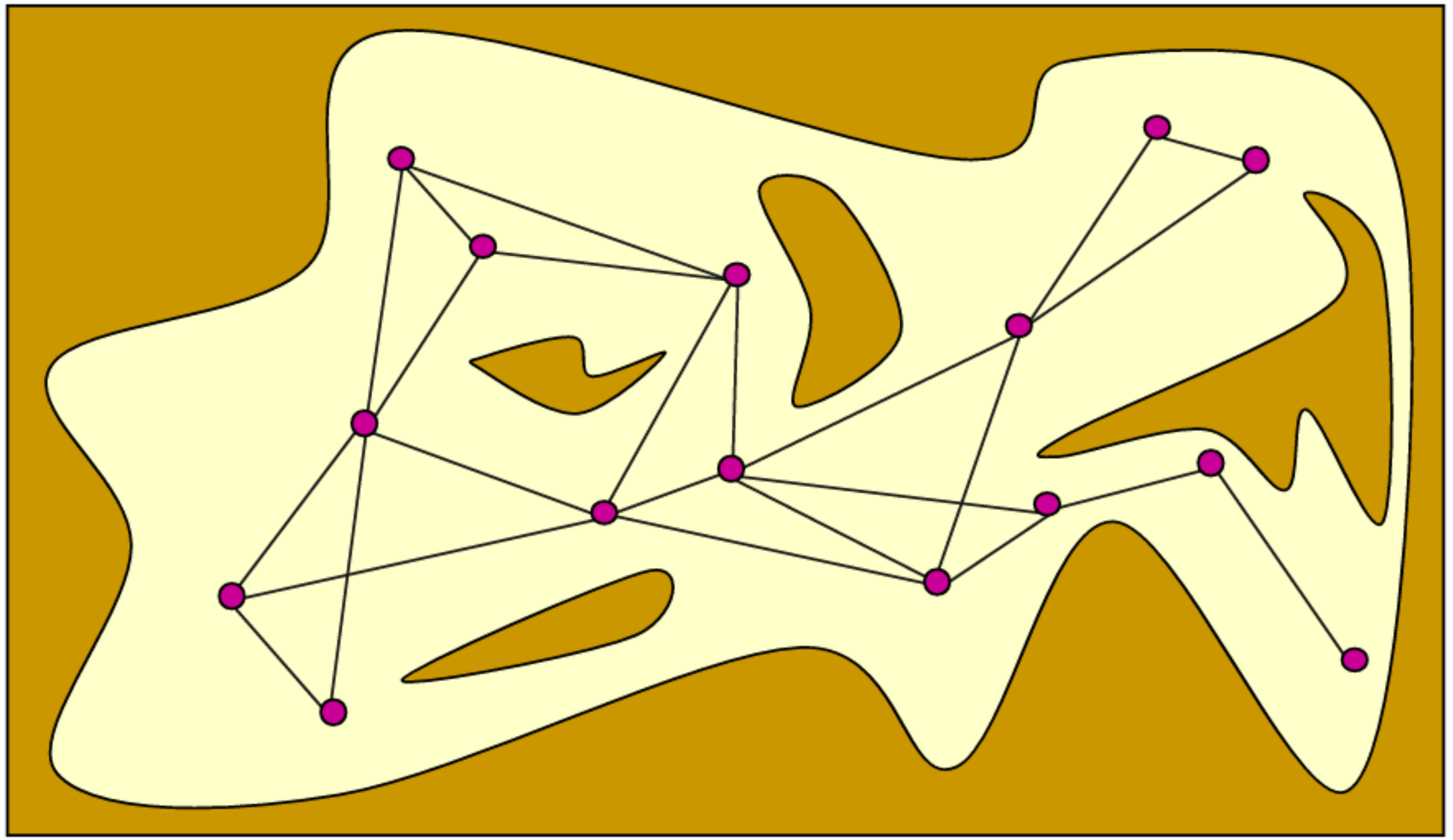
Paths that undergo collisions are removed



Slide from Pieter Abbeel.

Probabilistic Roadmaps

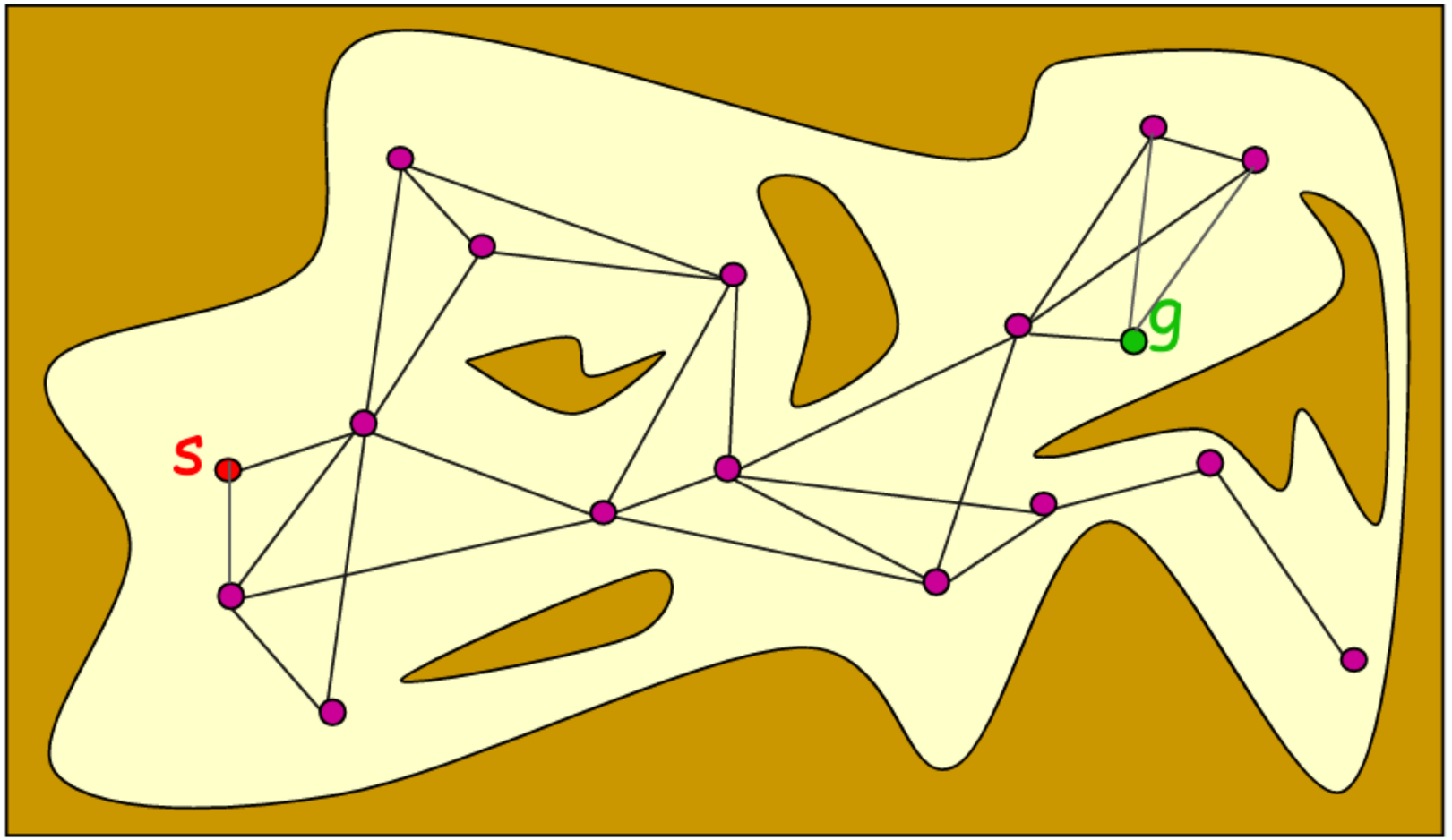
The collision-free links are retained as local paths to form the PRM



Slide from Pieter Abbeel.

Probabilistic Roadmaps

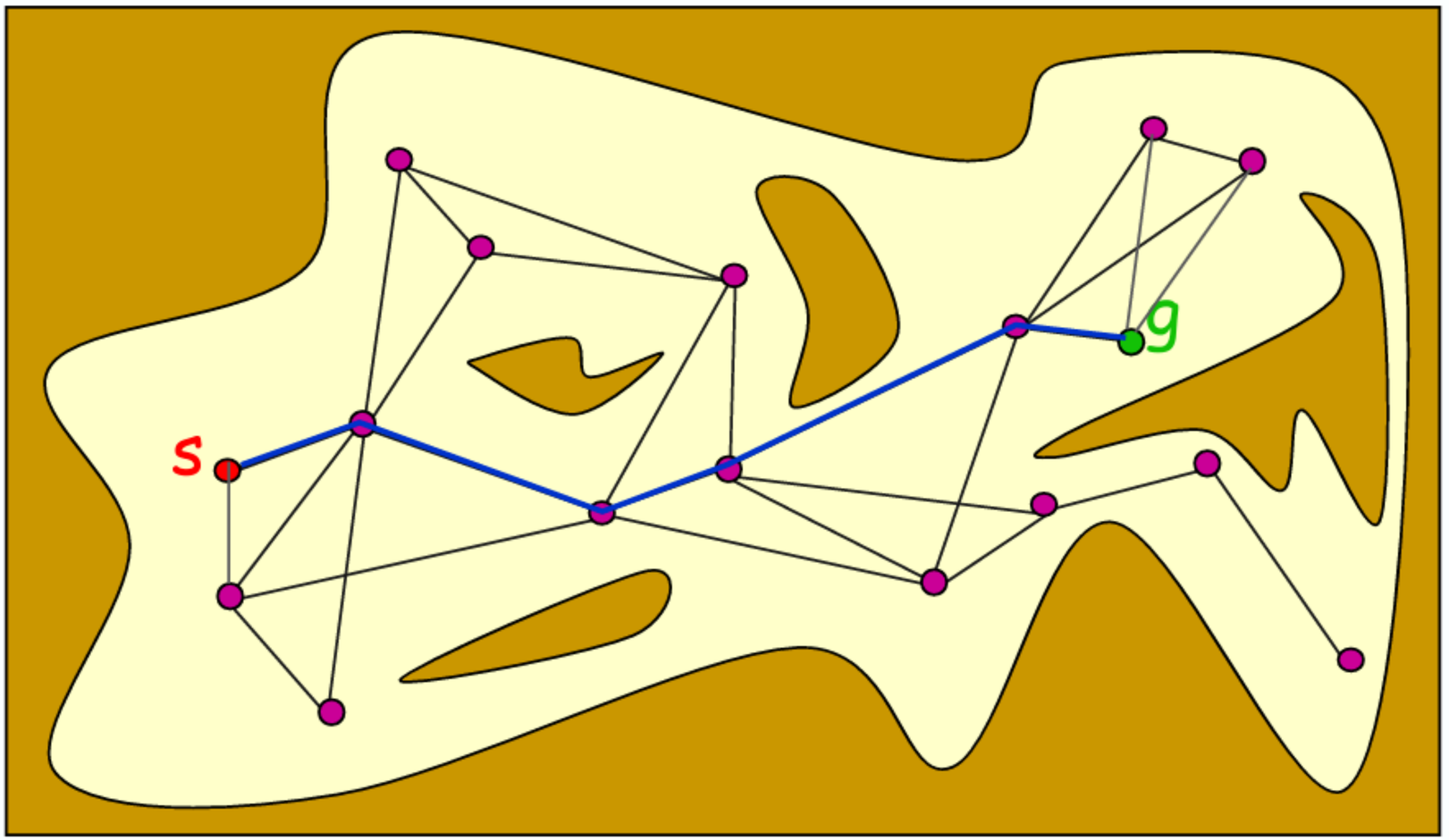
The start and goal configurations are included as milestones



Slide from Pieter Abbeel.

Probabilistic Roadmaps

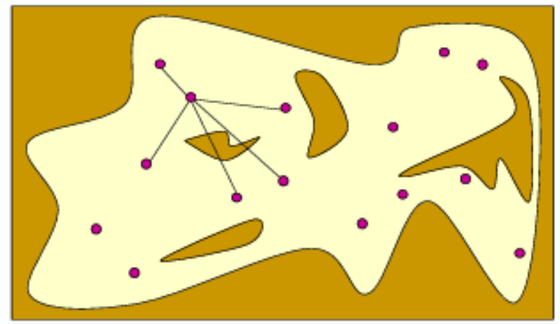
The PRM is searched for a path from s to g



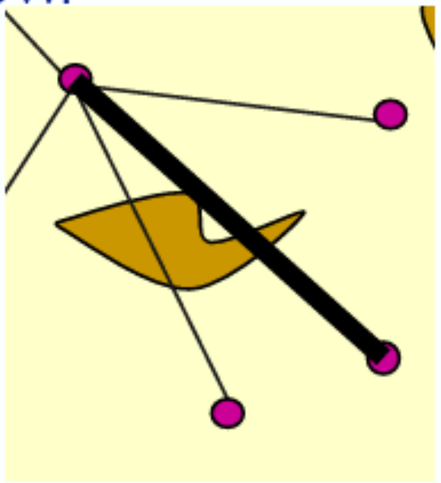
Slide from Pieter Abbeel.

Probabilistic Roadmaps

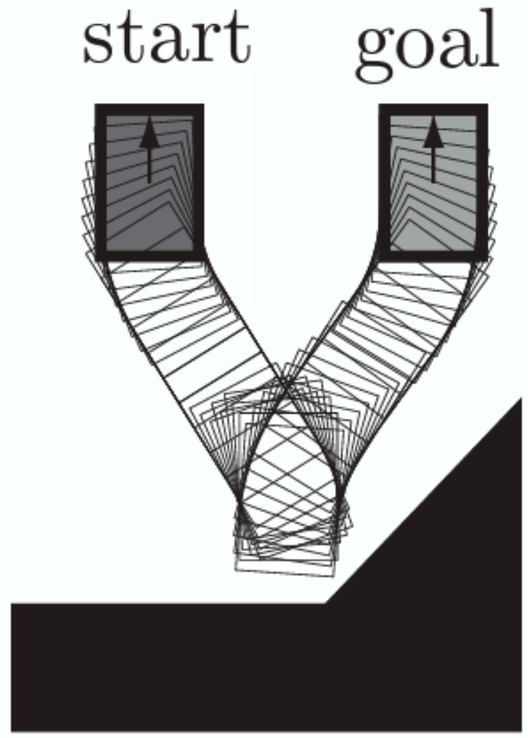
Challenging to link milestones



Collision checking can be slow.

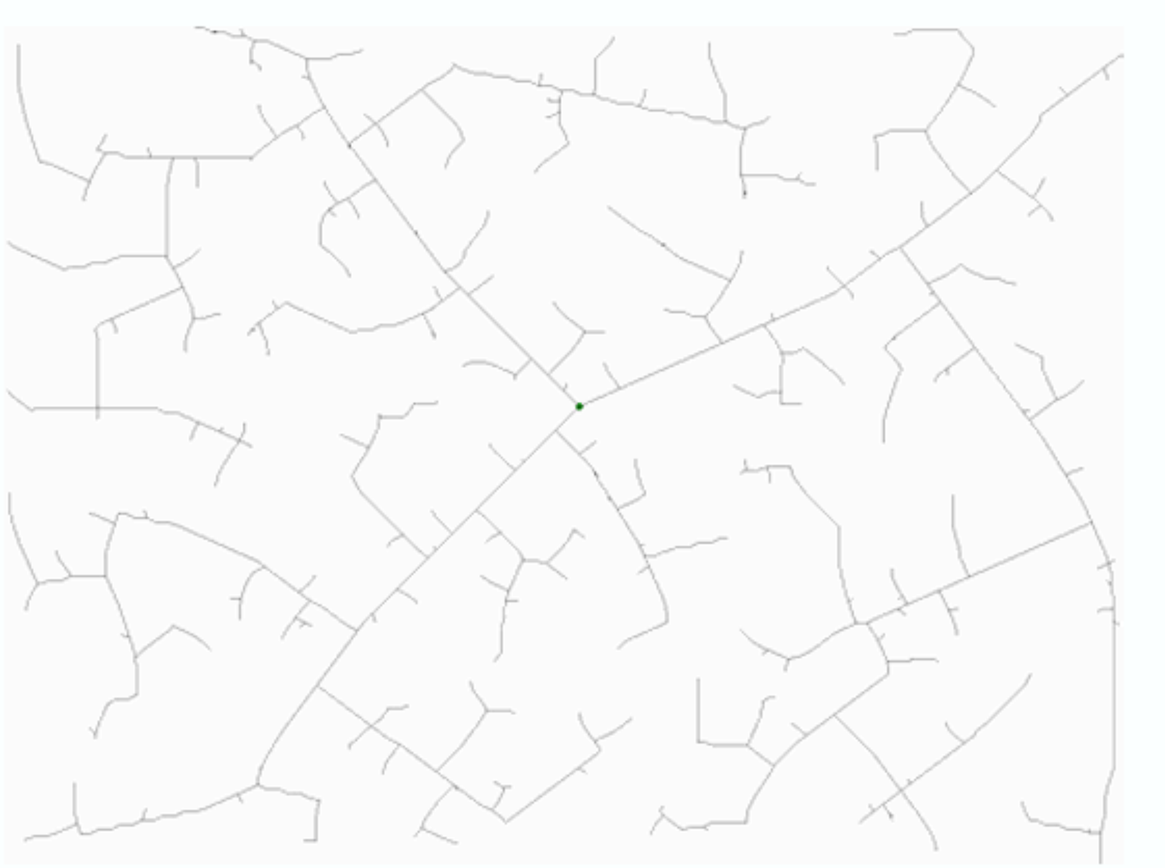


All straight line paths may not be feasible, or a good measure of distance between states.



Rapidly Exploring Random Trees (RRTs)

- Kinodynamic planning
- Build up a tree through generating “next states” in the tree by executing random controls



Rapidly Exploring Random Trees (RRTs)

- Build up a tree through generating “next states” in the tree by executing random controls

GENERATE_RRT($x_{init}, K, \Delta t$)

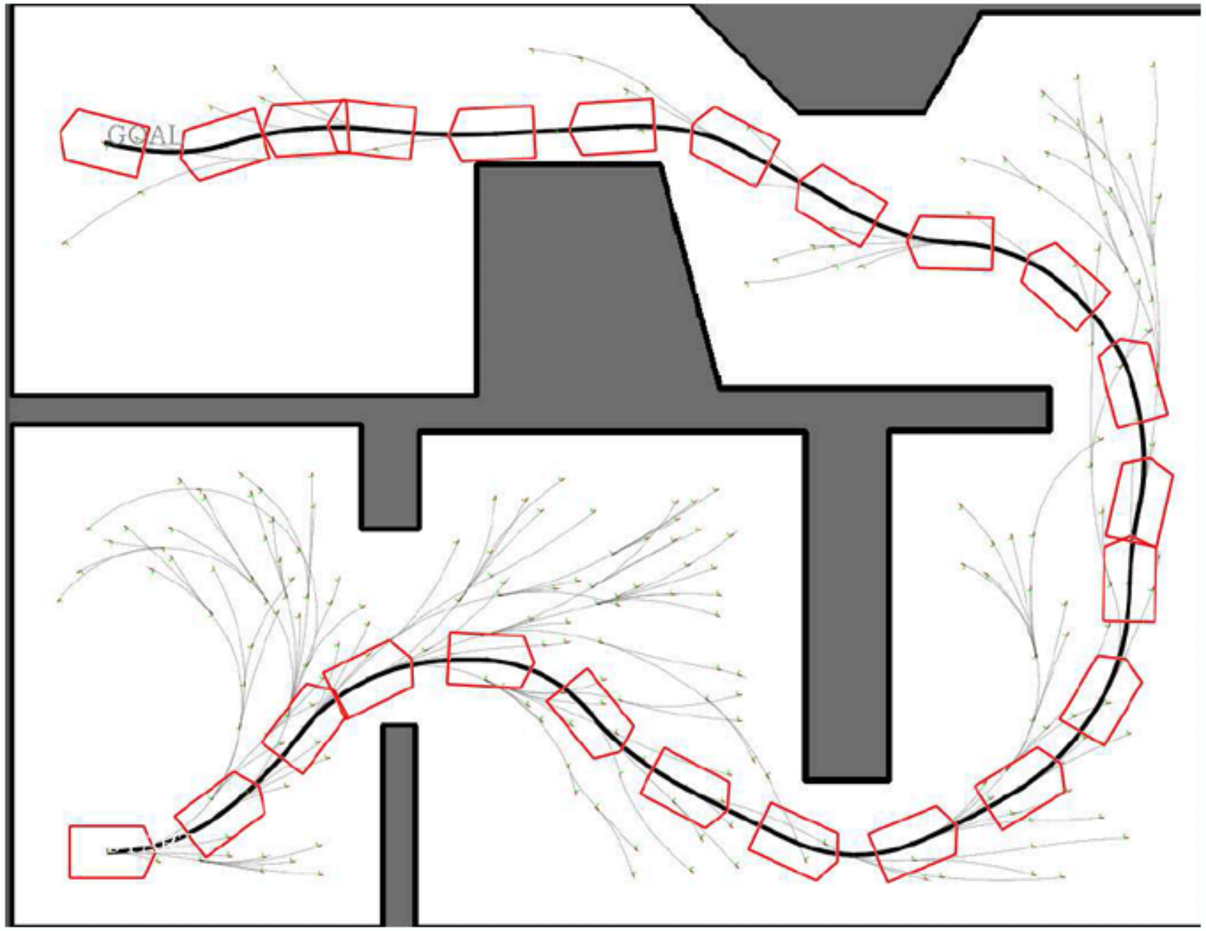
```
1  T.init(x_init);
2  for k = 1 to K do
3      x_rand ← RANDOM_STATE();
4      x_near ← NEAREST_NEIGHBOR(x_rand, T);
5      u ← SELECT_INPUT(x_rand, x_near);
6      x_new ← NEW_STATE(x_near, u, Δt);
7      T.add_vertex(x_new);
8      T.add_edge(x_near, x_new, u);
9  Return T
```

SELECT_INPUT(x_{rand}, x_{near})

- Two point boundary value problem
  - If too hard to solve, often just select best out of a set of control sequences. This set could be random, or some well chosen set of primitives.

Rapidly Exploring Random Trees (RRTs)

- Build up a tree through generating “next states” in the tree by executing random controls



Slide from Pieter Abbeel.

tree type

Random Tree | RRT | RRT*

obstacle type

narrow passage

number of nodes to add:

1 | 10 | 100 | 200 | 500

exploration bias

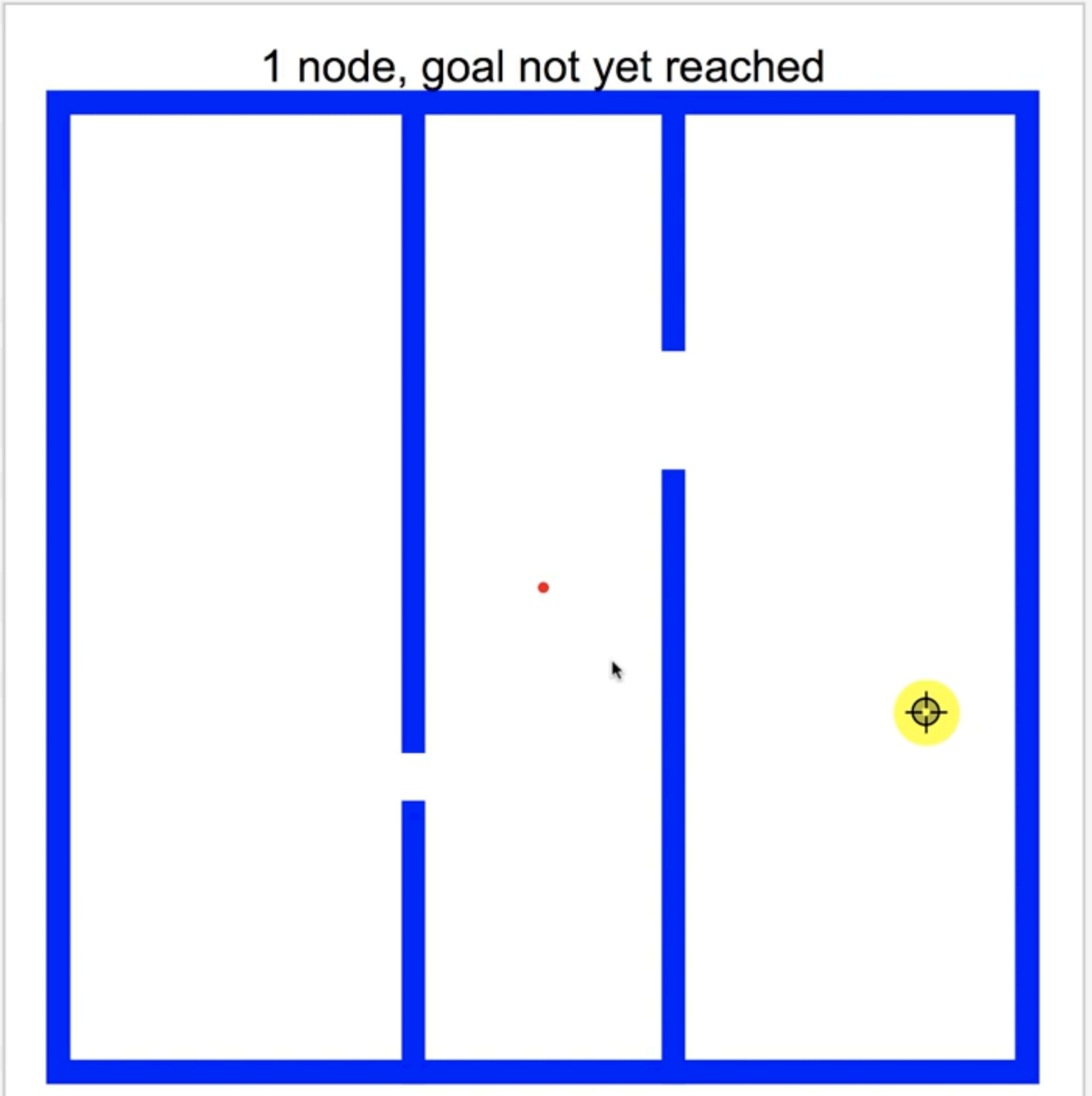
0.

goal radius

1.4

code at demonstrations.wolfram.com/RapidlyExploringRandomTreeRRTAndRRT/

by Aaron Becker and Li Huang



tree type

Random Tree | RRT | RRT*

obstacle type

narrow passage

number of nodes to add:

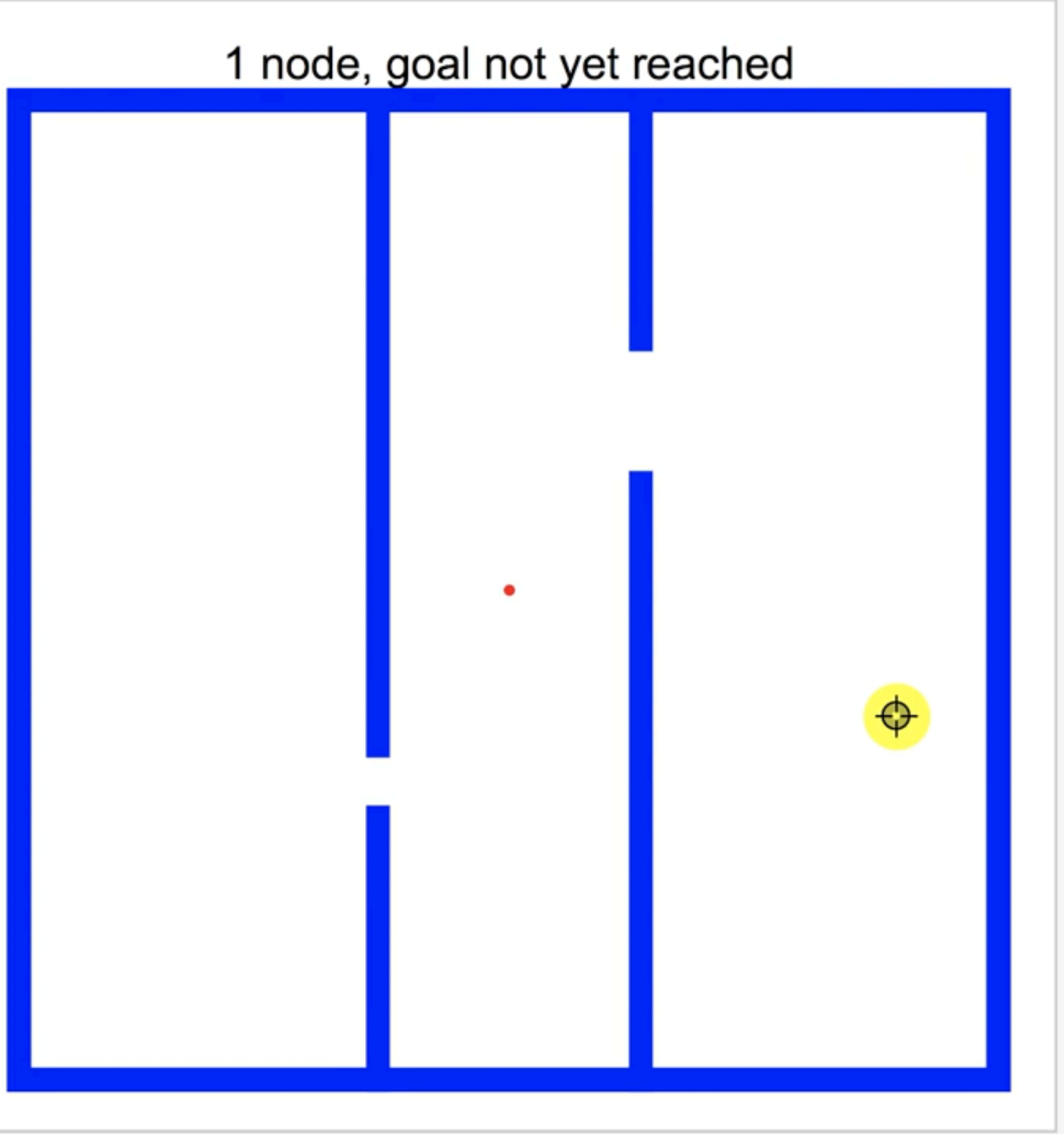
1 | 10 | 100 | 200 | 500

exploration bias

0.

goal radius

1.4



Algorithm BuildRRT
  Input: Initial configuration q_{init}, number of vertices in RRT K, incremental distance Δq)
  Output: RRT graph G

```
G.init(q_init)
for k = 1 to K
  q_rand ← RAND_CONF()
  q_near ← NEAREST_VERTEX(q_rand, G)
  q_new ← NEW_CONF(q_near, q_rand, Δq)
  G.add_vertex(q_new)
  G.add_edge(q_near, q_new)
return G
```

tree type

Random Tree | RRT | RRT*

obstacle type

narrow passage

number of nodes to add:

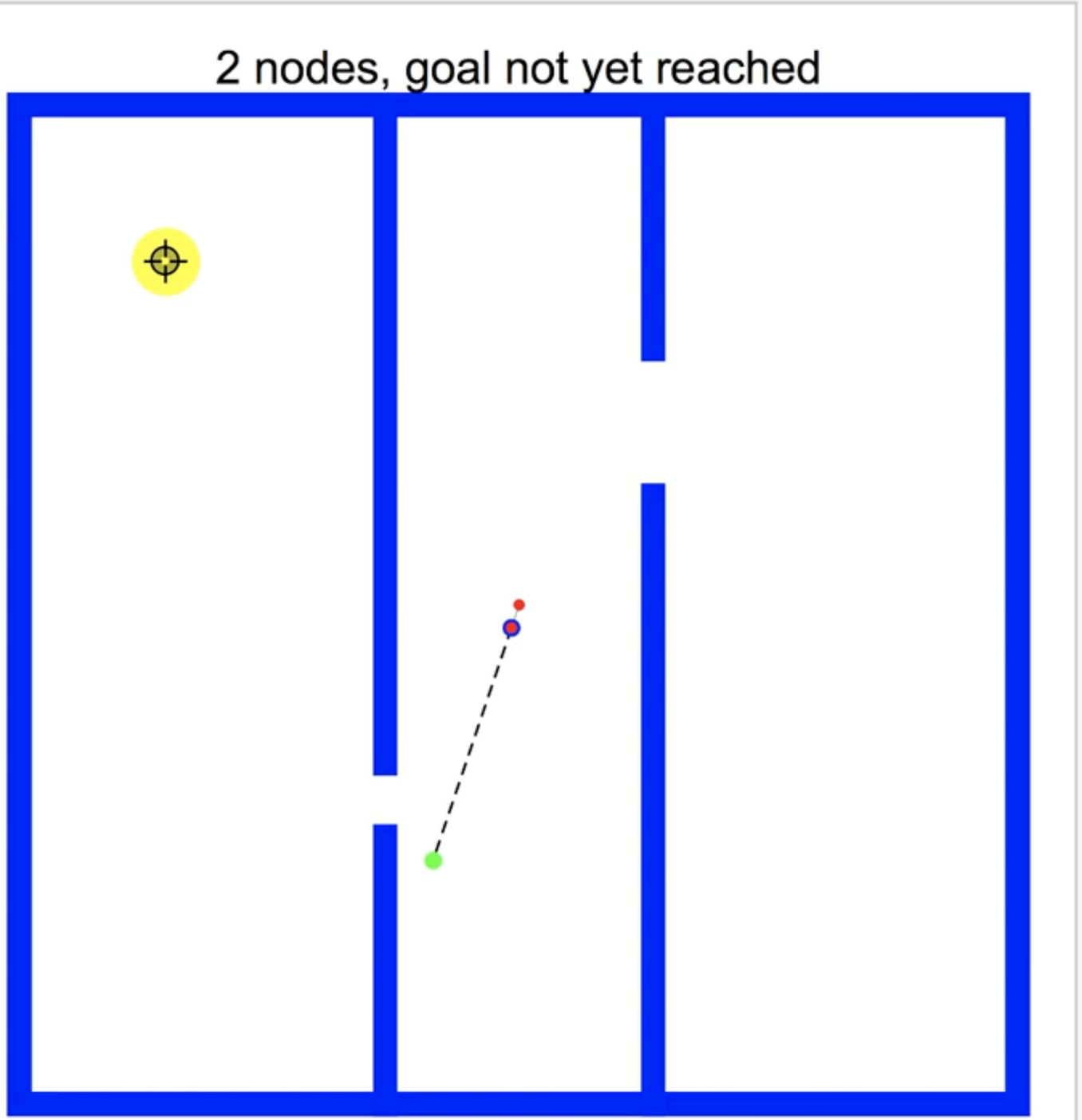
1 | 10 | 100 | 200 | 500

exploration bias

0.

goal radius

1.4



200%

tree type

Random Tree | RRT | RRT*

obstacle type

none

number of nodes to add:

1 | 10 | 100 | 200 | 500

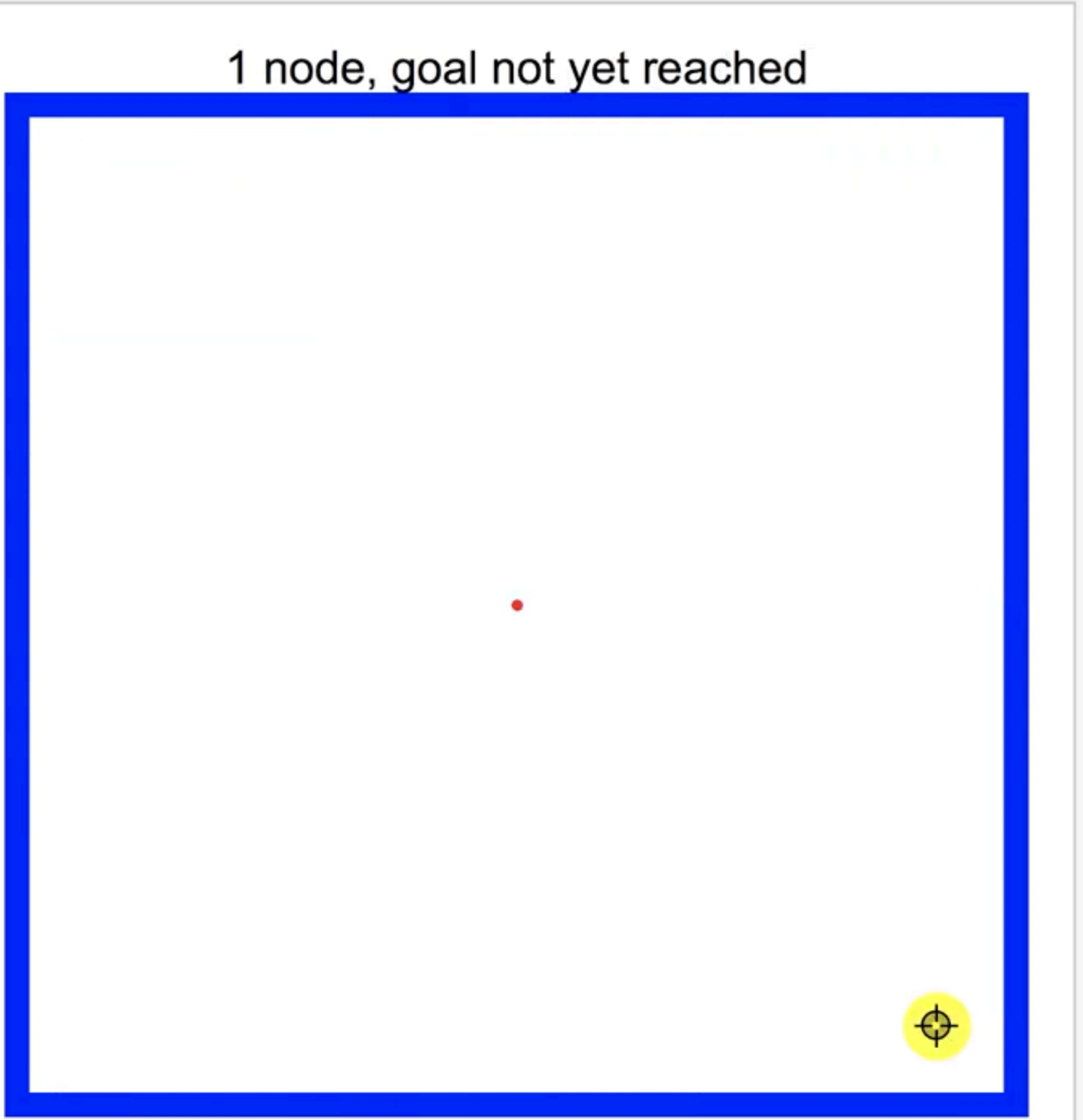
exploration bias

0.

goal radius

1.4

All three trees are *probabilistically complete*, meaning if a nonzero width path exists, the tree will eventually find the path.



tree type

Random Tree | RRT | RRT*

obstacle type

sparse

number of nodes to add:

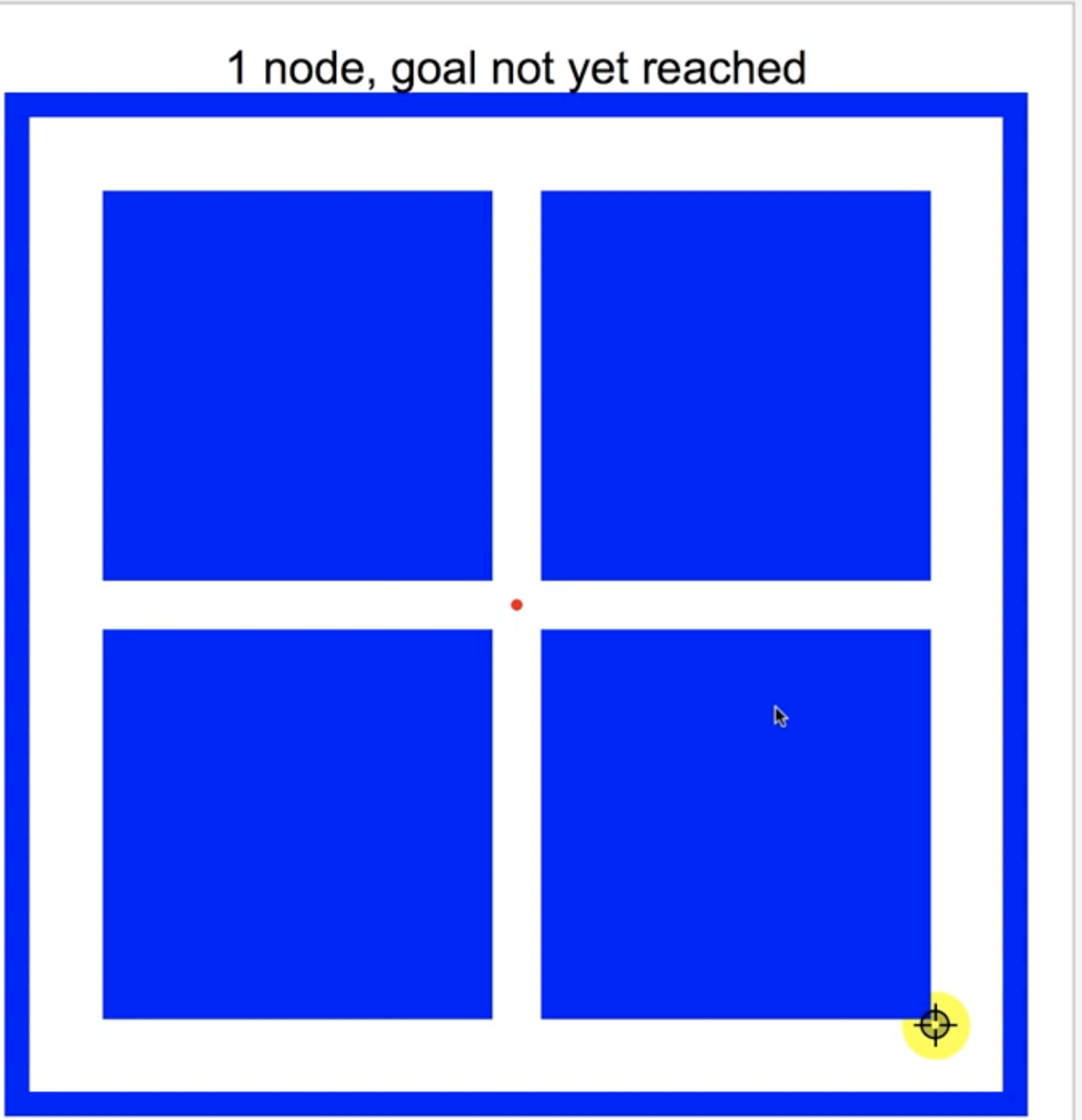
1 10 100 200 500

exploration bias

0.

goal radius

1.4



200%

tree type

Random Tree | RRT | RRT*

obstacle type

concave

number of nodes to add:

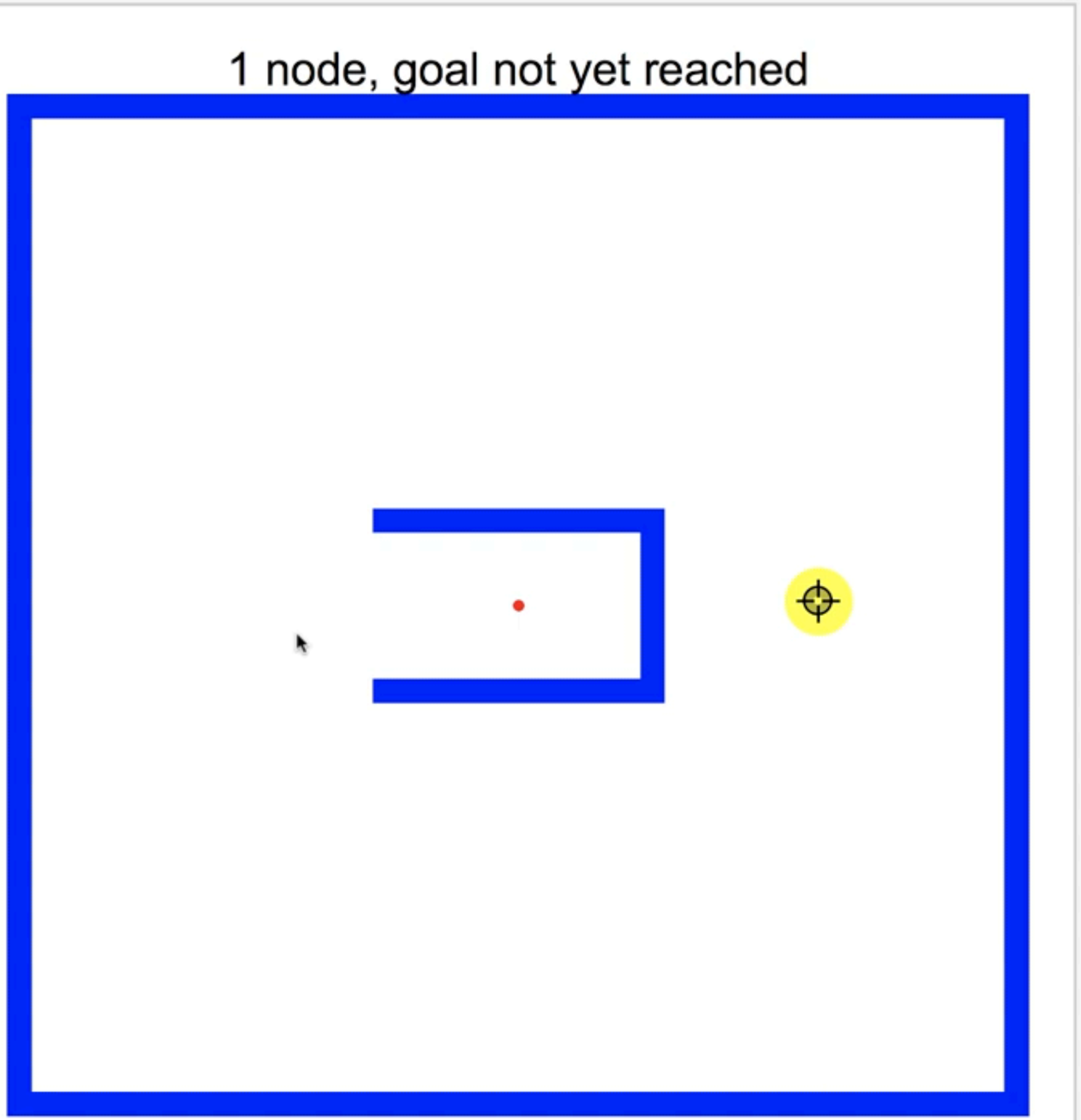
1 | 10 | 100 | 200 | 500

exploration bias

0.5

goal radius

1.4



200%

Today’s Agenda

- Robotic Tasks - Manipulation
- Terminology
- Observation and camera calibration
- How to move your robot?
  - Task space & configuration space
  - Forward kinematics and inverse kinematics
  - Path planning