Task and Motion Planning (TAMP)

Caelan Garrett

web.mit.edu/caelan

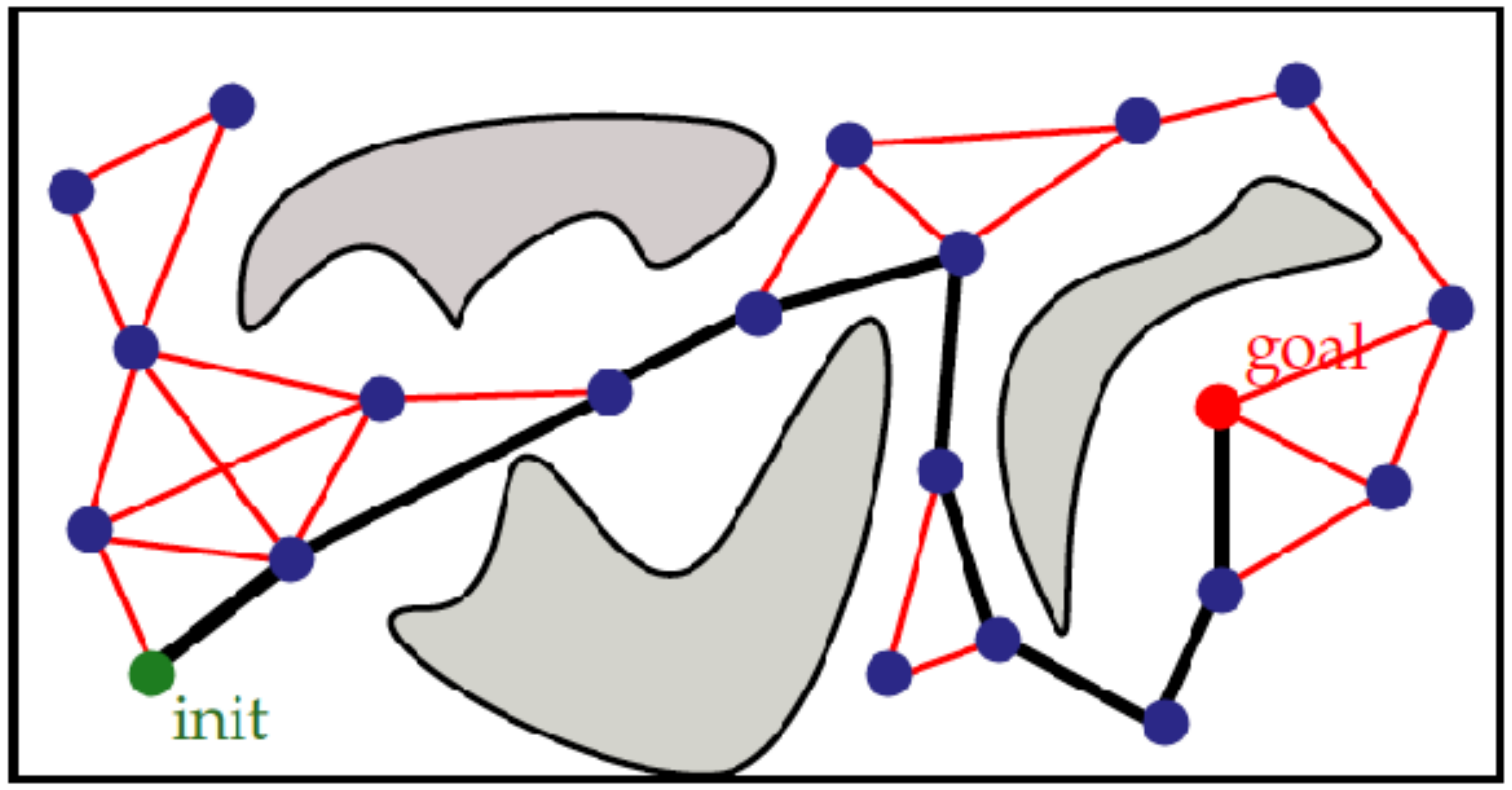
NVIDIA Research

CSE 571: AI Robotics

05/20/2025

(Probable) Roadmap

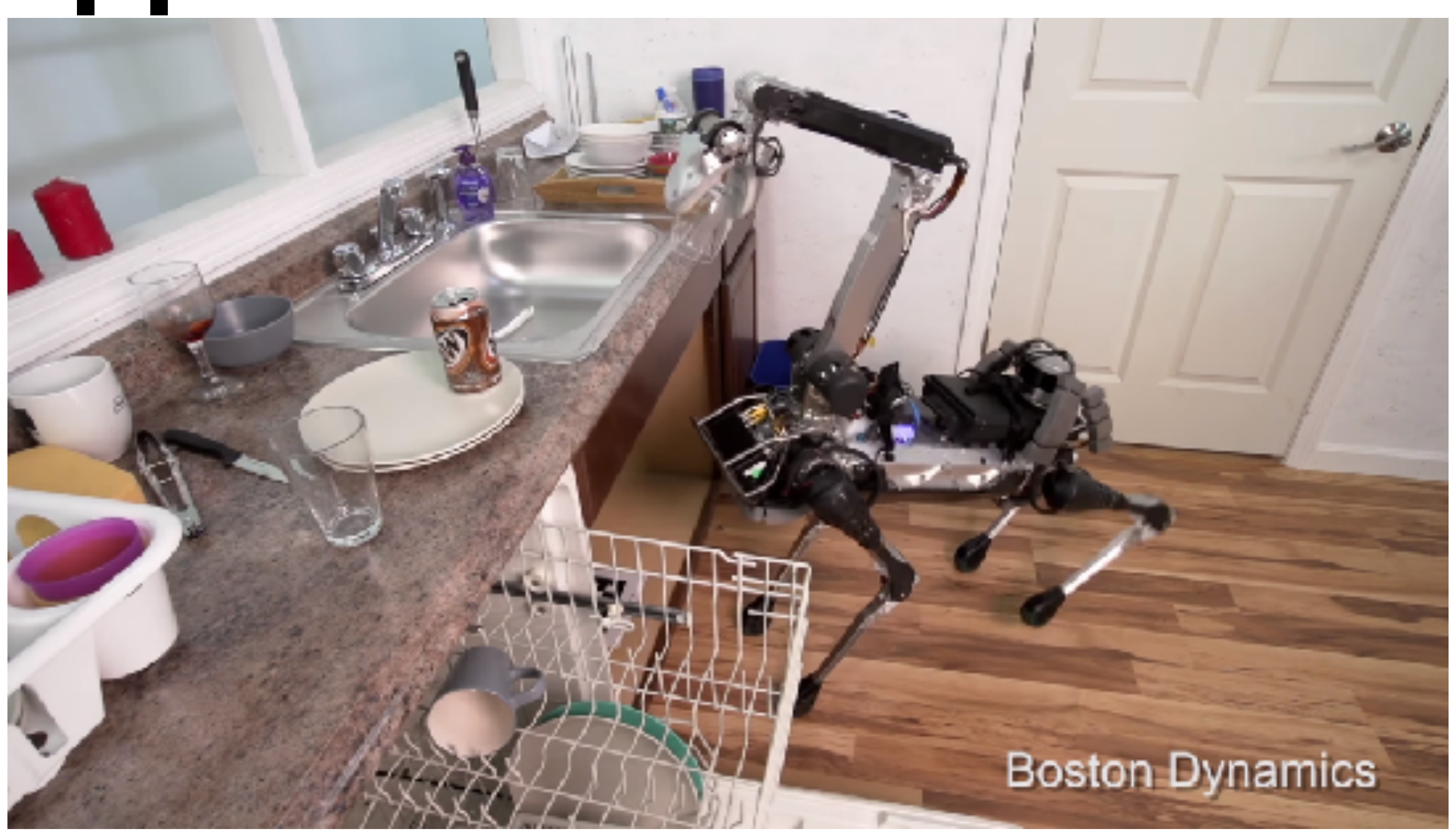
1. **Background**
   1. Task Planning
   2. Review: Motion Planning
2. **Hybrid Planning**
   1. Prediscretized Planning
   2. Multi-Modal Motion Planning
   3. Task and Motion Planning
3. **PDDLStream for TAMP**
4. **TAMP Extensions and Ongoing Work**
   1. GPU Acceleration, Stochasticity, Partially Observability, Imitation Learning



[Fig from Erion Plaku]

Planning for Autonomous Robots

- Robot must select both **high-level** actions & **low-level** controls
- **Application areas:** semi-structured and human environments



Household


Warehouse fulfilment

Food service


Construction

Task and Motion Planning (TAMP)

- Search in a **factored, hybrid** space
  - **Discrete** and **continuous** variables & actions

- **Variables**
  - **Continuous:** robot configuration, object poses, door joint positions,
  - **Discrete:** is-on, is-in-hand, is-holding-water, is-cooked, …

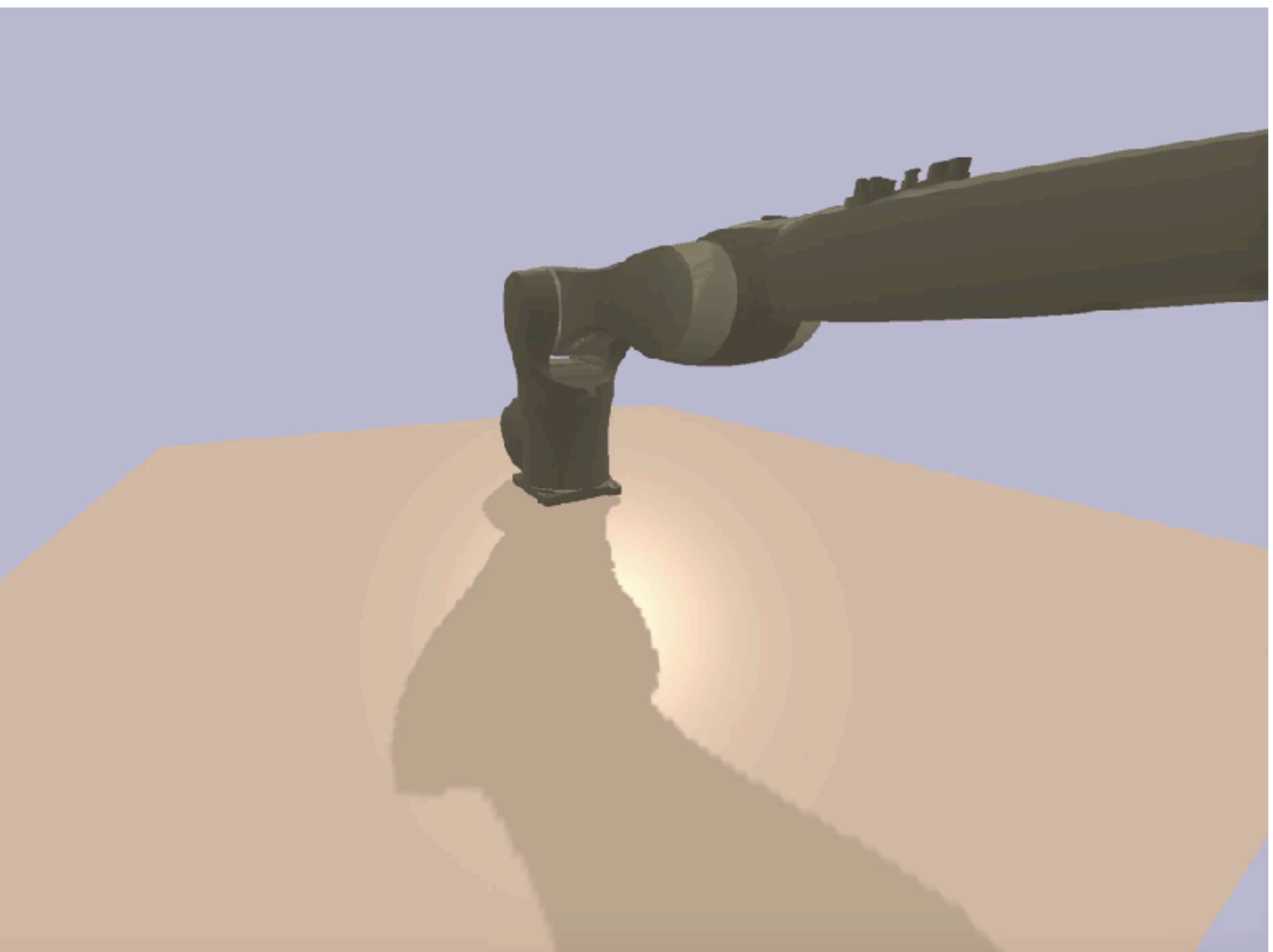
- **Actions:** move, pick, place, push, pull, pour, detect, cook, …

Cooking and Serving the “Blockoli”

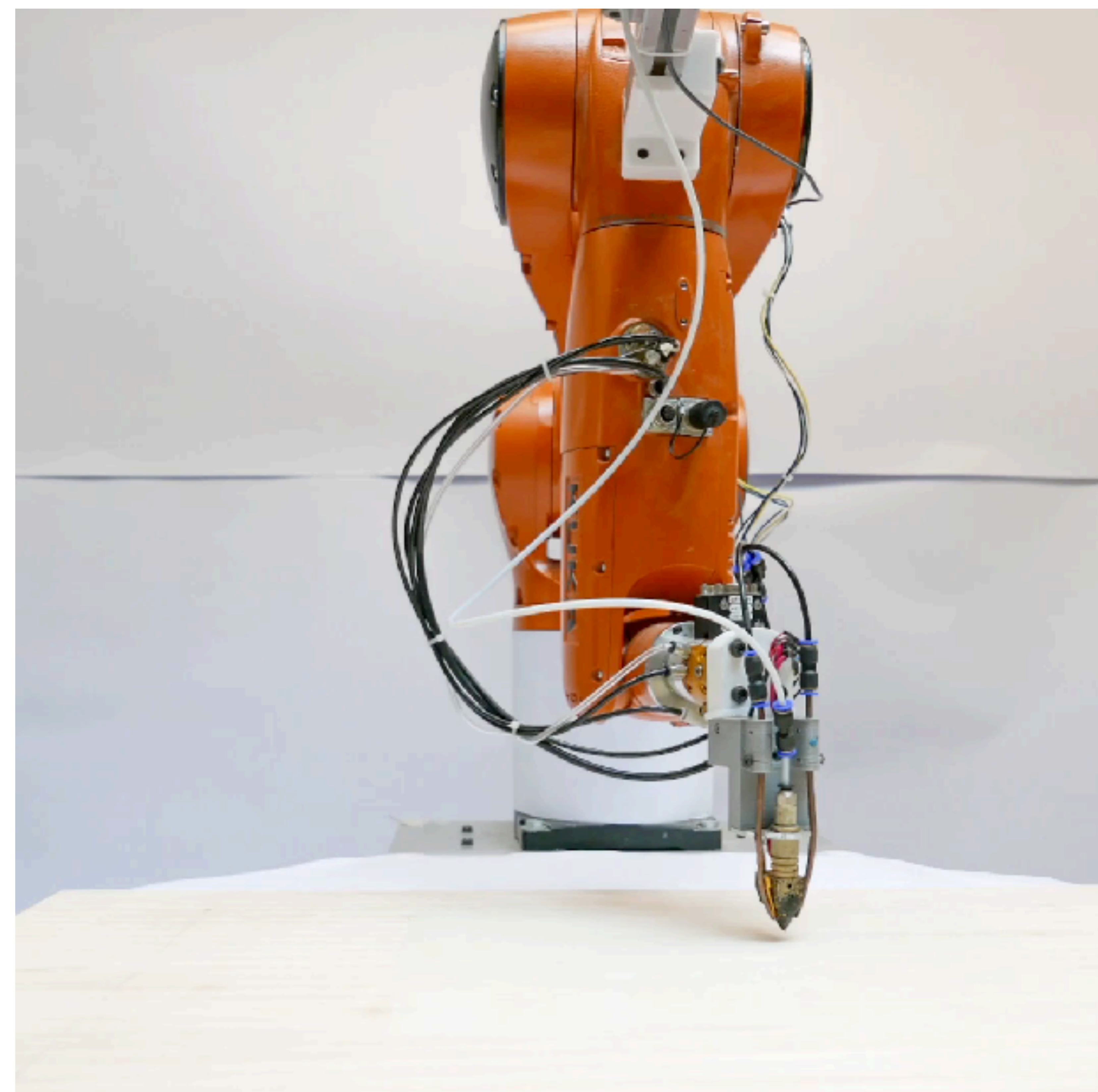
Preparing Coffee

Automated Fabrication

- Plan sequence of **306** 3D printing extrusions (actions)
- Collision, kinematic, **stability** and **stiffness** constraints



[Huang, Garrett, & Mueller 2018]



Problem Class

- **Discrete-time**
  - Plans are finite sequences of controls
- **Deterministic** (not an MDP for now)
  - Actions always produce the intended effect
  - Solutions are **plans** (instead of policies)
- **Observable** (not a POMDP for now)
  - Access to the full world state
- **Hybrid**
  - States & controls composed of mixed **discrete-continuous** variables

Task Planning

Task (Classical, Symbolic) Planning

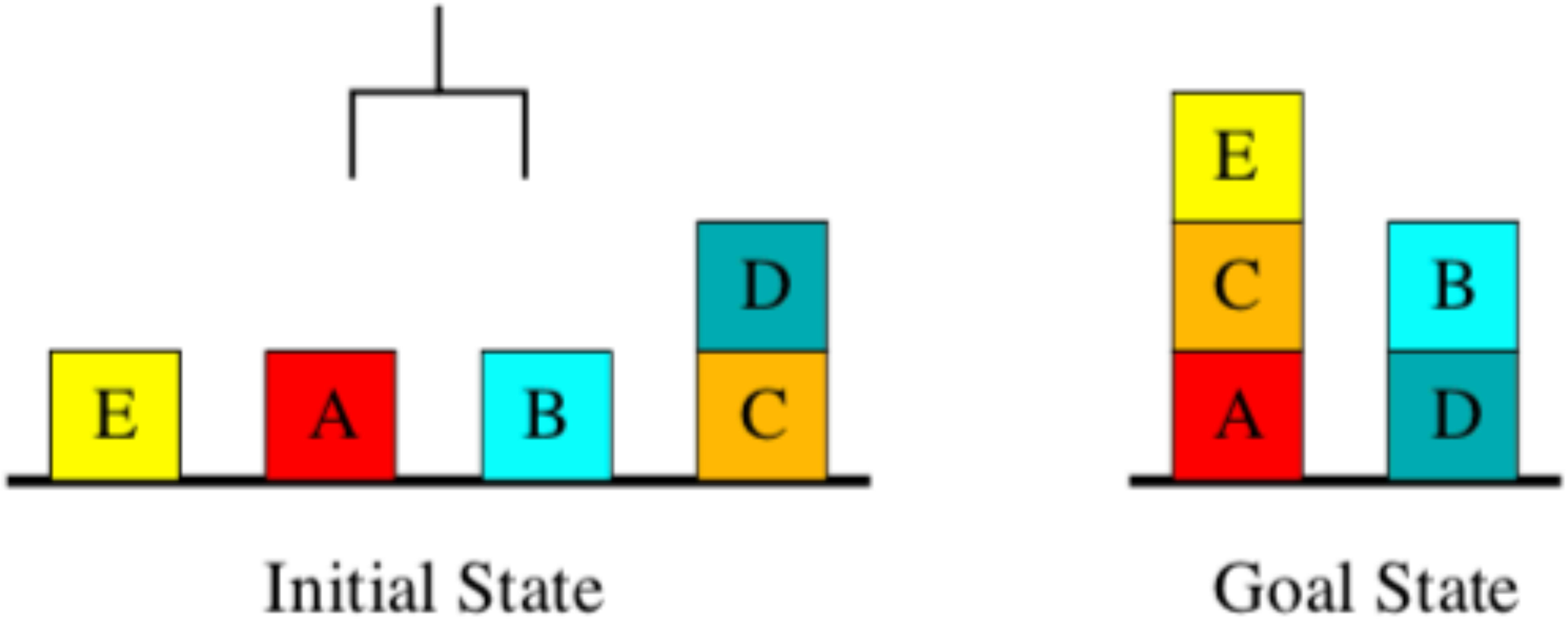
- **Discrete** problems with **many variables**
  - Often enormous (2^N) but **finite** state-spaces

- Problems typically described using an **action language**
  - **Propositional Logic** (STRIPS) [Fikes 1971][Aeronautiques 1998]
  - **Planning Domain Description Language** (PDDL)

- Develop **domain-independent** algorithms
  - Can apply to **any problem** expressible using PDDL
- Exploit **factoring** and **sparsity** to develop algorithms

Classical Planning Representations

Blocksworld domain

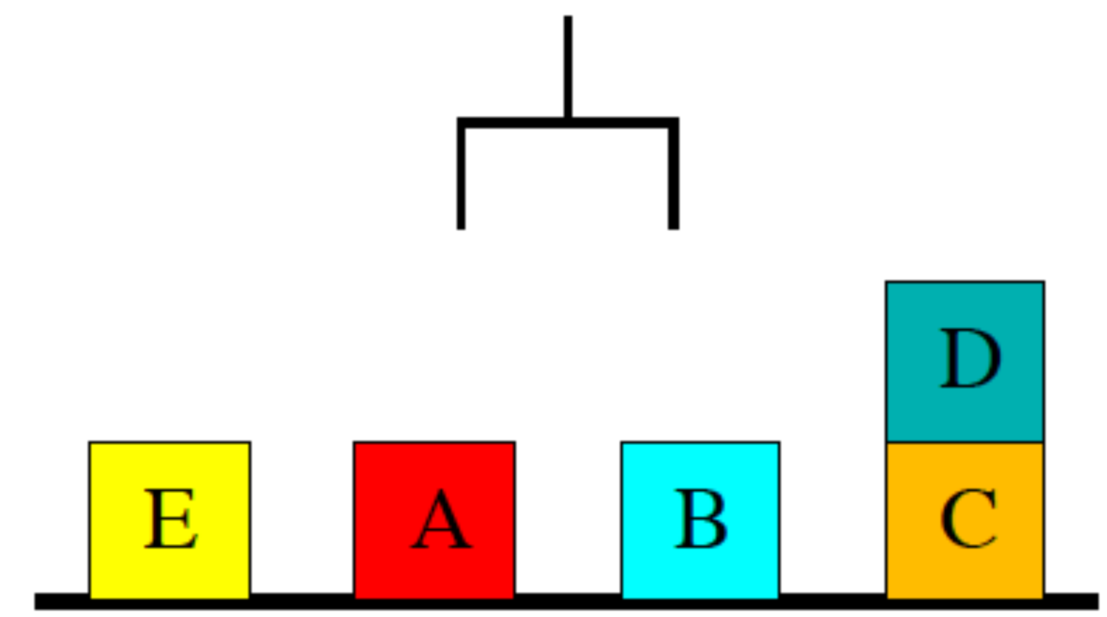


- Facts: $on(x, y)$, $onTable(x)$, $clear(x)$, $holding(x)$, $armEmpty()$.
- Initial state: $\{onTable(E), clear(E), \ldots, onTable(C), on(D, C), clear(D), armEmpty()\}$.
- Goal: $\{on(E, C), on(C, A), on(B, D)\}$.
- Actions: $stack(x, y)$, $unstack(x, y)$, $putdown(x)$, $pickup(x)$.
- $stack(x, y)$? $pre : \{holding(x), clear(y)\}$
  $add : \{on(x, y), armEmpty()\}$
  $del : \{holding(x), clear(y)\}$.

[Figs from Hector Geffner]

First-Order Action Languages

- **Predicate:** Boolean function `On(?b1, ?b2)=`**`True`**`/`**`False`**
- **Facts** (literals): instantiated predicates `On(D, C)=`**`True`**
- **State:** set of facts `{On(A, B)=`**`False`**`, On(D, C)=`**`True`**`, …}`
  - Equivalently, Boolean state variables
  - **Closed-world** assumption
    - Unspecified facts are **false**



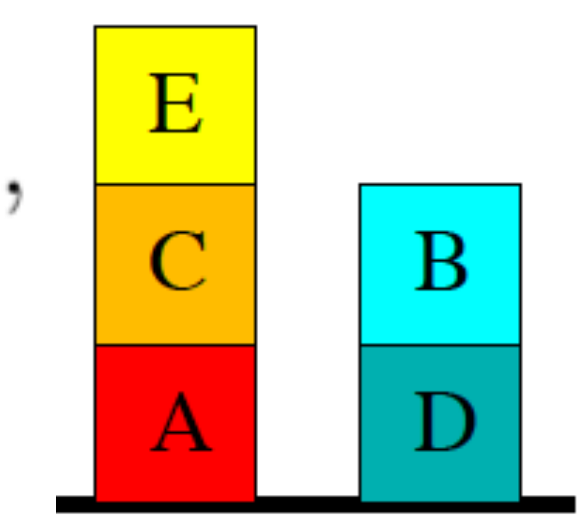
Initial State

Facts: $on(x, y)$, $onTable(x)$, $clear(x)$, $holding(x)$, $armEmpty()$.

Initial state: $\{onTable(E), clear(E), \ldots, onTable(C), on(D, C), clear(D), armEmpty()\}$.

Goal: $\{on(E, C), on(C, A), on(B, D)\}$.

Actions: $stack(x, y)$, $unstack(x, y)$, $putdown(x)$, $pickup(x)$.



Goal State

(Lifted) Action Schema

- A tuple of free **parameters**
- A **precondition** formula tests applicability
- An **effect** formula modifies the state (as a **delta**)
- Logical **conjunctions** encode factoring

```
(:action stack
 :parameters (?b1, ?b2)
 :precondition {
   Holding(?b1), Clear(?b2)}
 :effect {ArmEmpty(),
   On(?b1, ?b2),
   Clear(?b1)
   ¬Holding(?b1),
   ¬Clear(?b2)}
```

```
(:action unstack
 :parameters (?b1, ?b2)
 :precondition {ArmEmpty(),
   On(?b1, ?b2),
   Clear(?b1)}
 :effect {Holding(?b1),
  Clear(?b2),
   ¬Clear(?b1),
   ¬ArmEmpty(),
   ¬On(?b1, ?b2)}
```

Planning Approaches

- **State-space** search: [Bonet 2001] [Hoffman 2001] [Helmert 2006]
  - **Progression** (forward) or regression (backward)
  - Best-first **heuristic search** algorithms
- **Partial-order** planning [Penberthy 1992]
  - Search directly over plans (**plan-space**)
- Planning as **Satisfiability** [Kautz 1999]
  - Compile to **fixed-horizon** SAT instance
  - SAT is **NP-Complete,** Planning is **PSPACE-Complete**
  - **Increase horizon if formula unsatisfiable**
- Large Language Models (**LLMs**)

LLMs for Task Planning

- Large Language Models (LLMs) proficient at commonsense reasoning
- But they struggle at **easy** International Planning Competition (IPC) benchmark problems

Domain	Method	Instances correct				
		GPT-4	GPT-3.5	I-GPT3.5	I-GPT3	GPT-3
Blocksworld (BW)	One-shot	206/600 (34.3%)	37/600 (6.1%)	54/600 (9%)	41/600 (6.8%)	6/600 (1%)
	Zero-shot	210/600 (34.6%)	8/600 (1.3%)	-	-	-
	COT	214/600 (35.6%)	-	-	-	-
Logistics Domain	One-shot	28/200 (14%)	1/200 (0.5%)	6/200 (3%)	3/200 (1.5%)	-
	Zero-shot	15/200 (7.5%)	1/200 (0.5%)	-	-	-
Mystery BW (Deceptive)	One-shot	26/600 (4.3%)	0/600 (0%)	4/600 (0.6%)	14/600 (2.3%)	0/600 (0%)
	Zero-shot	1/600 (0.16%)	0/600 (0%)	-	-	-
	COT	54/600 (9%)	-	-	-	-

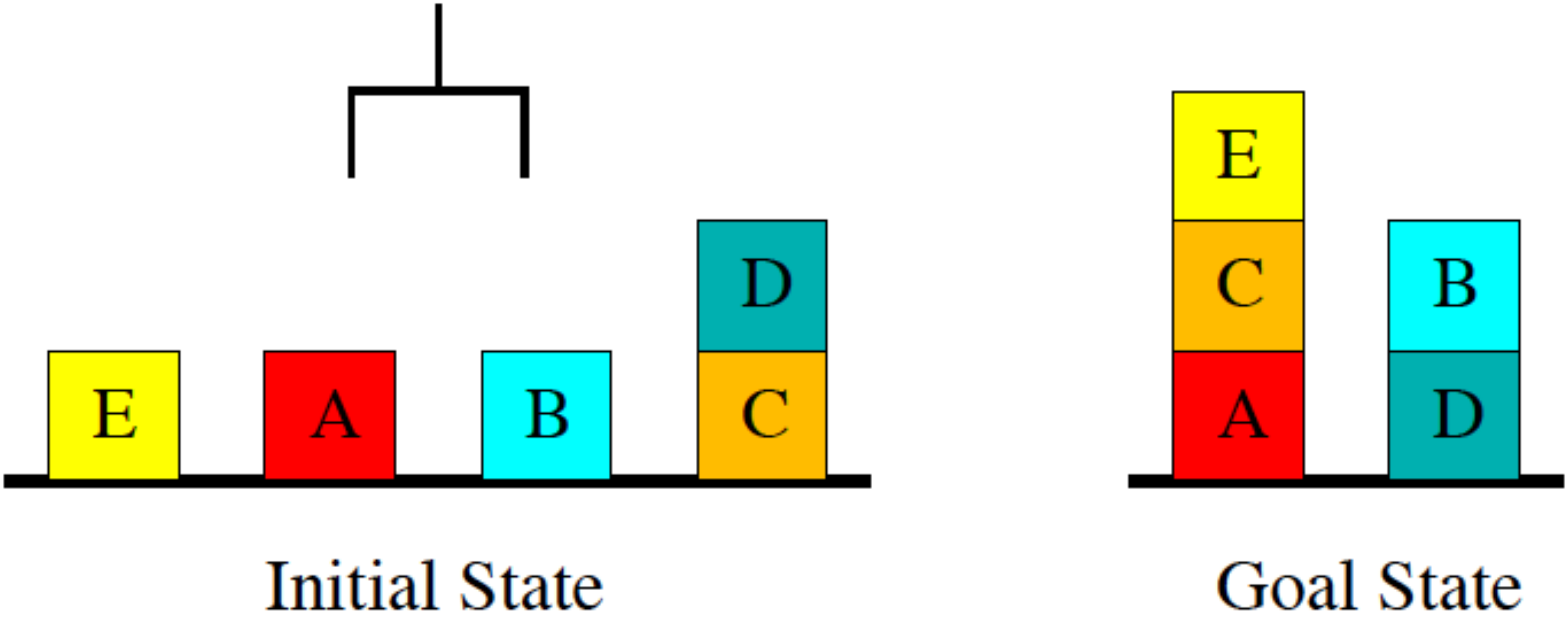
[Valmeekam 2023]

Forward Best-First Search

- For a state s
  - Path **cost:** $g(s)$
  - **Heuristic** estimate: $h(s)$
  - Open list **sorted** by priority $f(s)$
- **Weighted A*:** $f(s) = g(s) + wh(s)$
  - Uniform cost search: $w = 0 \implies f(s) = g(s)$
  - A* search: $w = 1 \implies f(s) = g(s) + h(s)$
  - **Greedy** best-first search: $w = \infty \implies f(s) = h(s)$
- How do we estimate $h(s)$?
  - No obvious metric (no metric-space embedding)

Predict the Minimum Plan Length

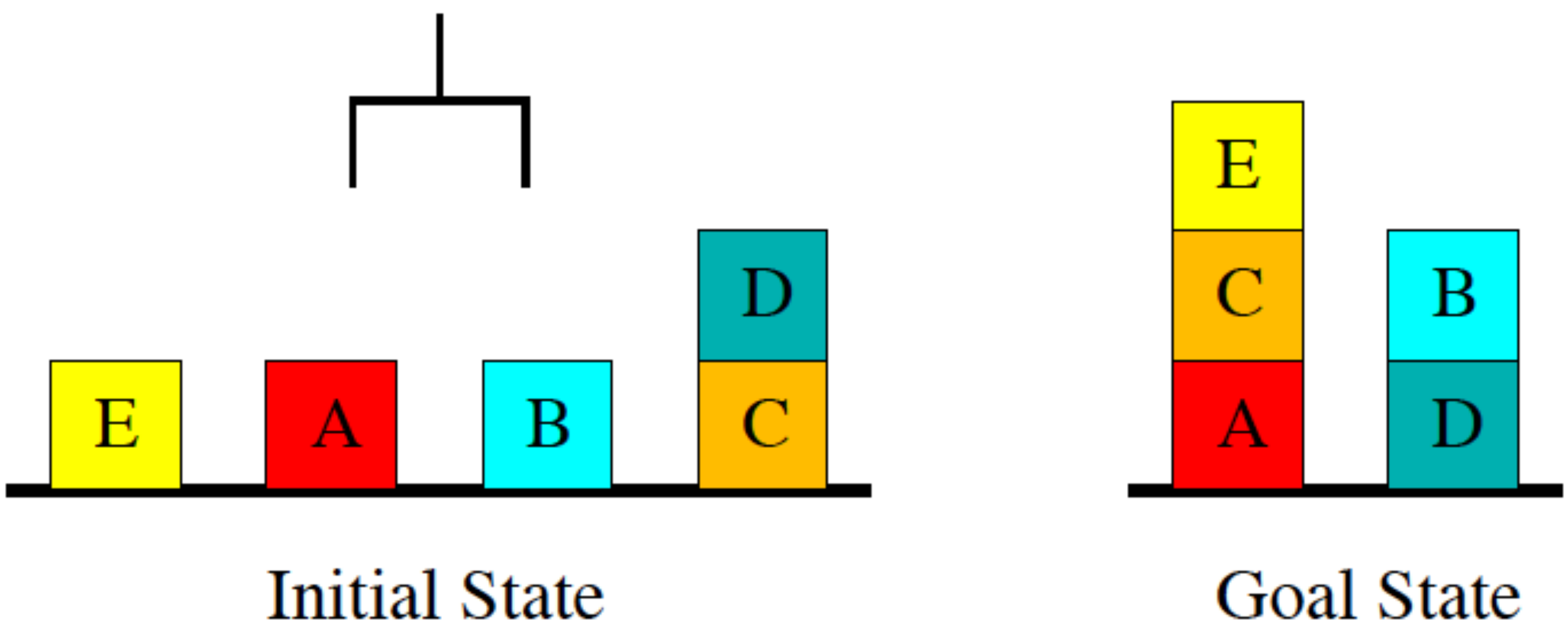
- Can stack / unstack anywhere on the ground
- Hint: is an **even** number



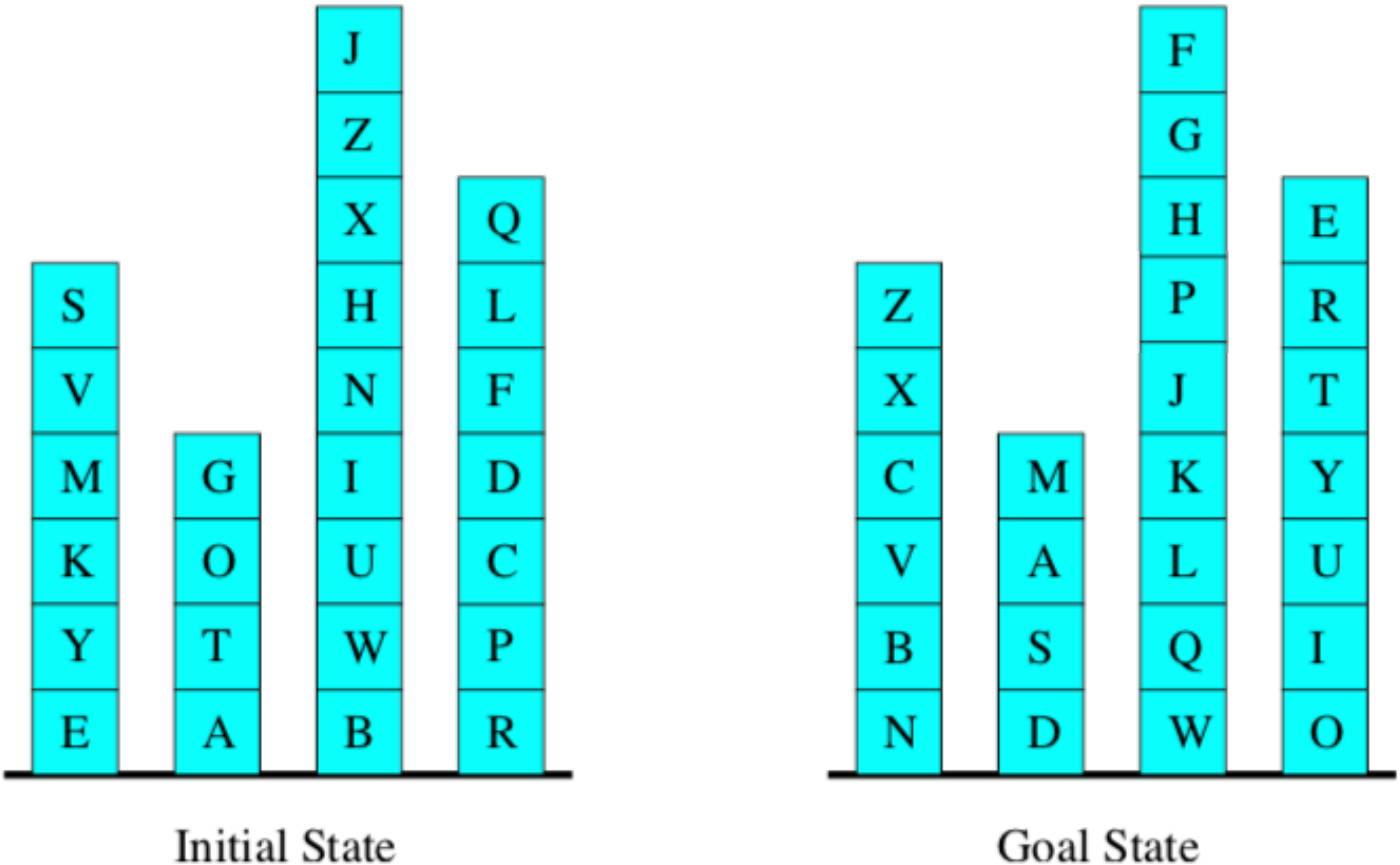
Predict the Minimum Plan Length

- **Solution** (length=8):
  - **unstack**(D, C)
  - **stack**(D, ground)
  - **unstack**(B, ground)
  - **stack**(B, D)
  - **unstack**(C, ground)
  - **stack**(C, A)
  - **unstack**(E, ground)
  - **stack**(E, C)

4/5 Blocks
Move Once



Predict the Minimum Plan Length



Domain-Independent Heuristics

- Estimating $h(s)$ is **nontrivial**
- Can we do it in an a **domain-independent** manner?
- Solve a relaxed, **approximate** planning problem

- Suggestions for how to do this?
  - **Independently** plan for each goal [Lipovetzky 2012]
  - **Remove** some action preconditions [Helmert 2006]
  - <u>Remove negative (**delete**) **effects**</u> [Bonet 2001] [Hoffman 2001]
  - Learn a **value function** on PDDL [Shen 2020]
  - …

Delete-Relaxation Heuristics

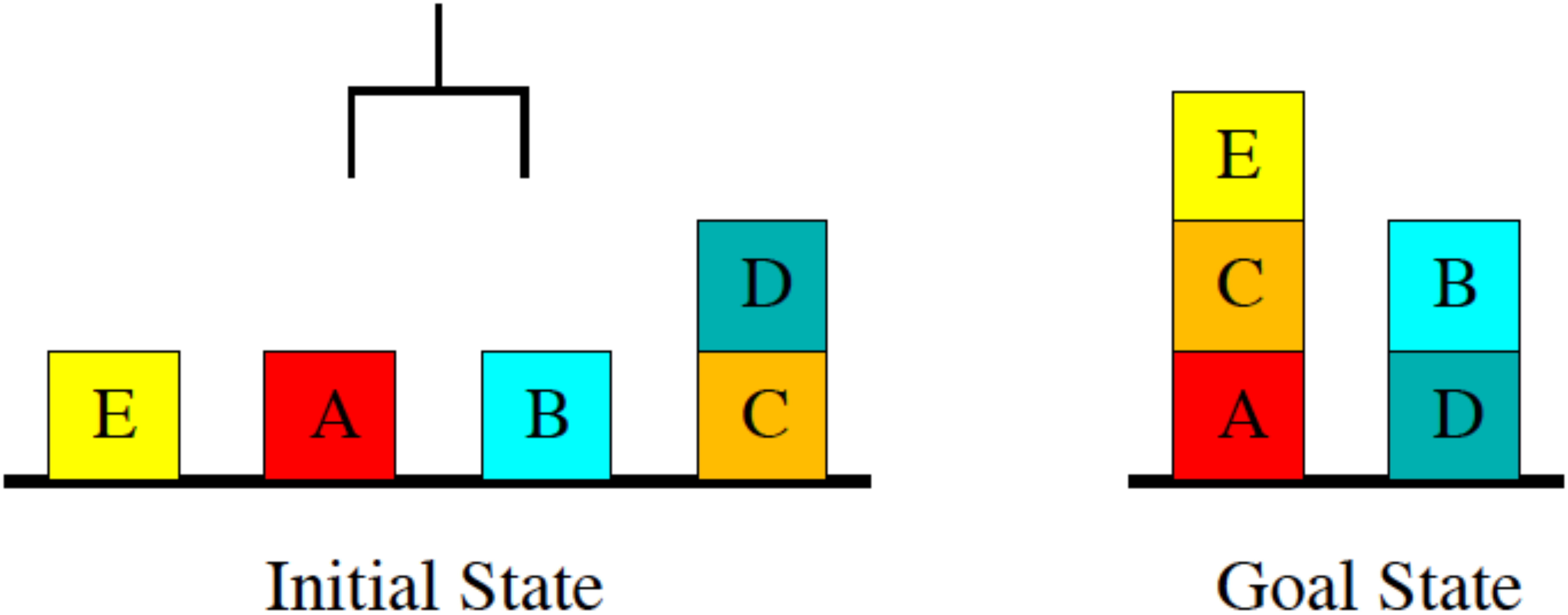
- Remove all negative (¬) effects
  - Solving optimally is **NP-Complete**
  - Can greedily find a short plan in polynomial time
- Basis for both **admissible** and **greedier**, non-admissible heuristics

(:**action stack**
 :**parameters** (?b1, ?b2)
 :**precondition** {
  Holding(?b1), Clear(?b2)}
 :**effect** {ArmEmpty(),
  On(?b1, ?b2),
  Clear(?b1)
  ~~¬Holding(?b1),~~
  ~~¬Clear(?b2)}~~

(:**action unstack**
 :**parameters** (?b1, ?b2)
 :**precondition** {ArmEmpty(),
  On(?b1, ?b2),
  Clear(?b1)}
 :**effect** {Holding(?b1),
  Clear(?b2),
  ~~¬Clear(?b1),~~
  ~~¬ArmEmpty(),~~
  ~~¬On(?b1, ?b2)}~~

Predict the Minimum Delete-Relaxed Plan Length

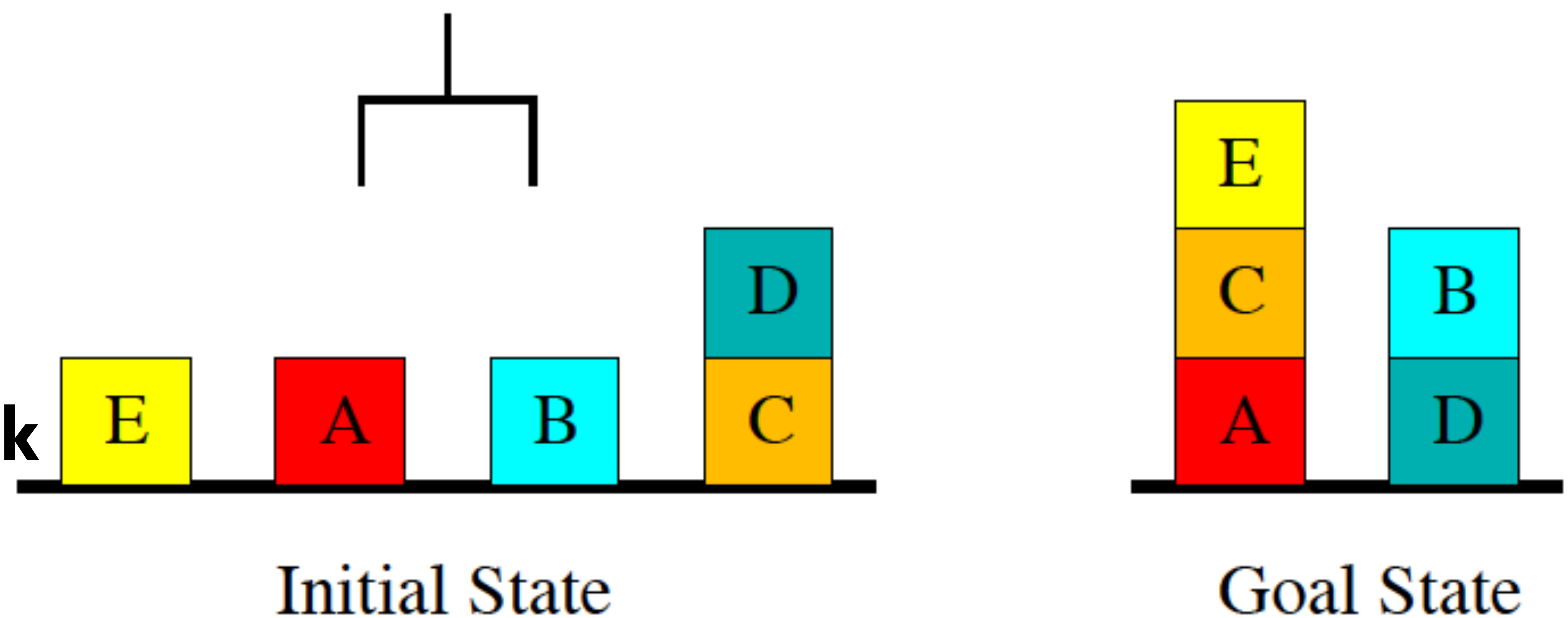
- Can stack / unstack anywhere on the ground
- Hint: is **no greater** than 8



Predict the Minimum Delete-Relaxed Plan Length

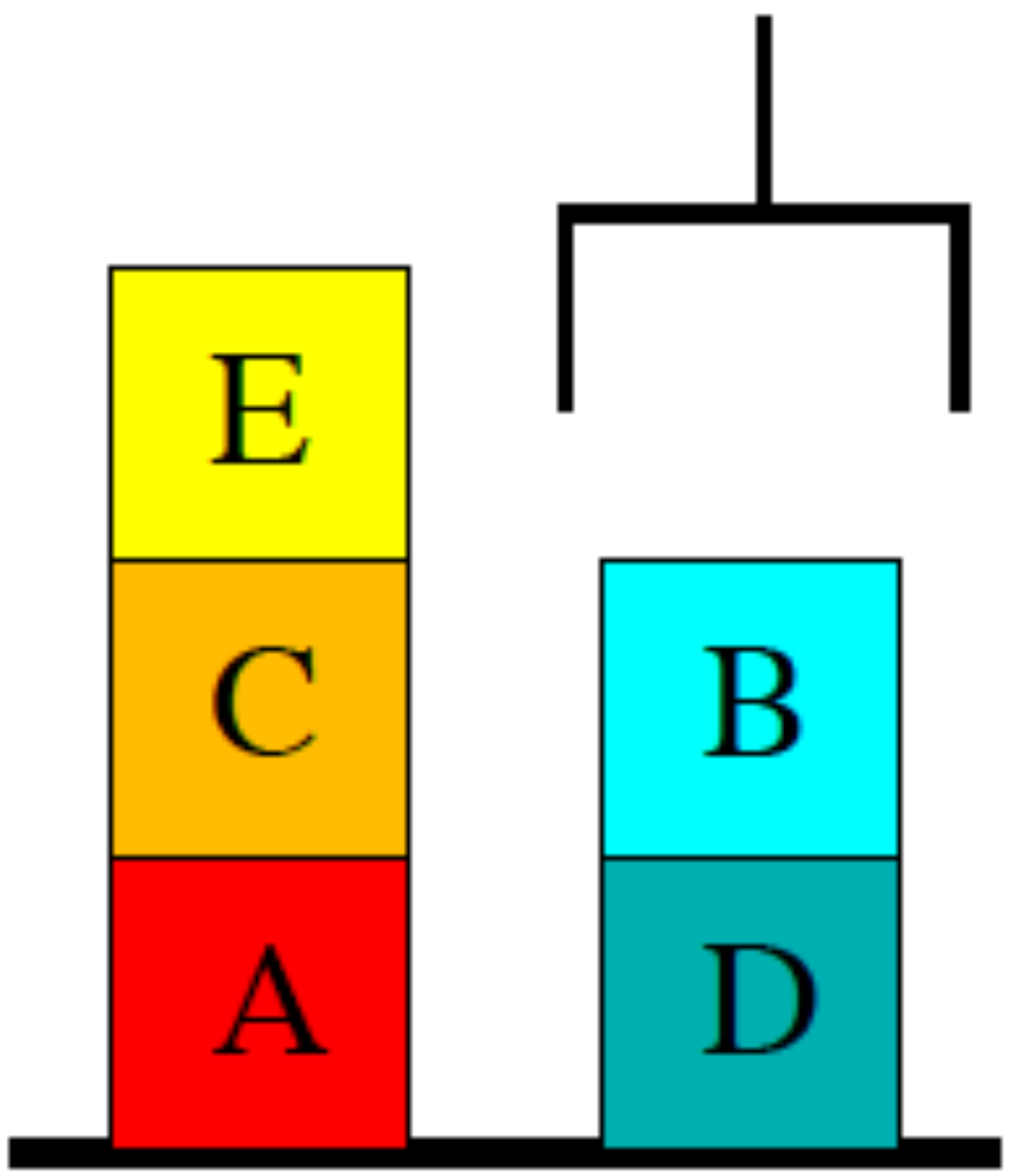
- **Solution** (length=8):
  - **`unstack`**`(D, C)`
  - **`stack`**`(D, ground)`
  - **`unstack`**`(B, ground)`
  - **`stack`**`(B, D)`
  - **`unstack`**`(C, ground)`
  - **`stack`**`(C, A)`
  - **`unstack`**`(E, ground)`
  - **`stack`**`(E, C)`

Still Need to
Unstack + **Stack**
4/5 Blocks

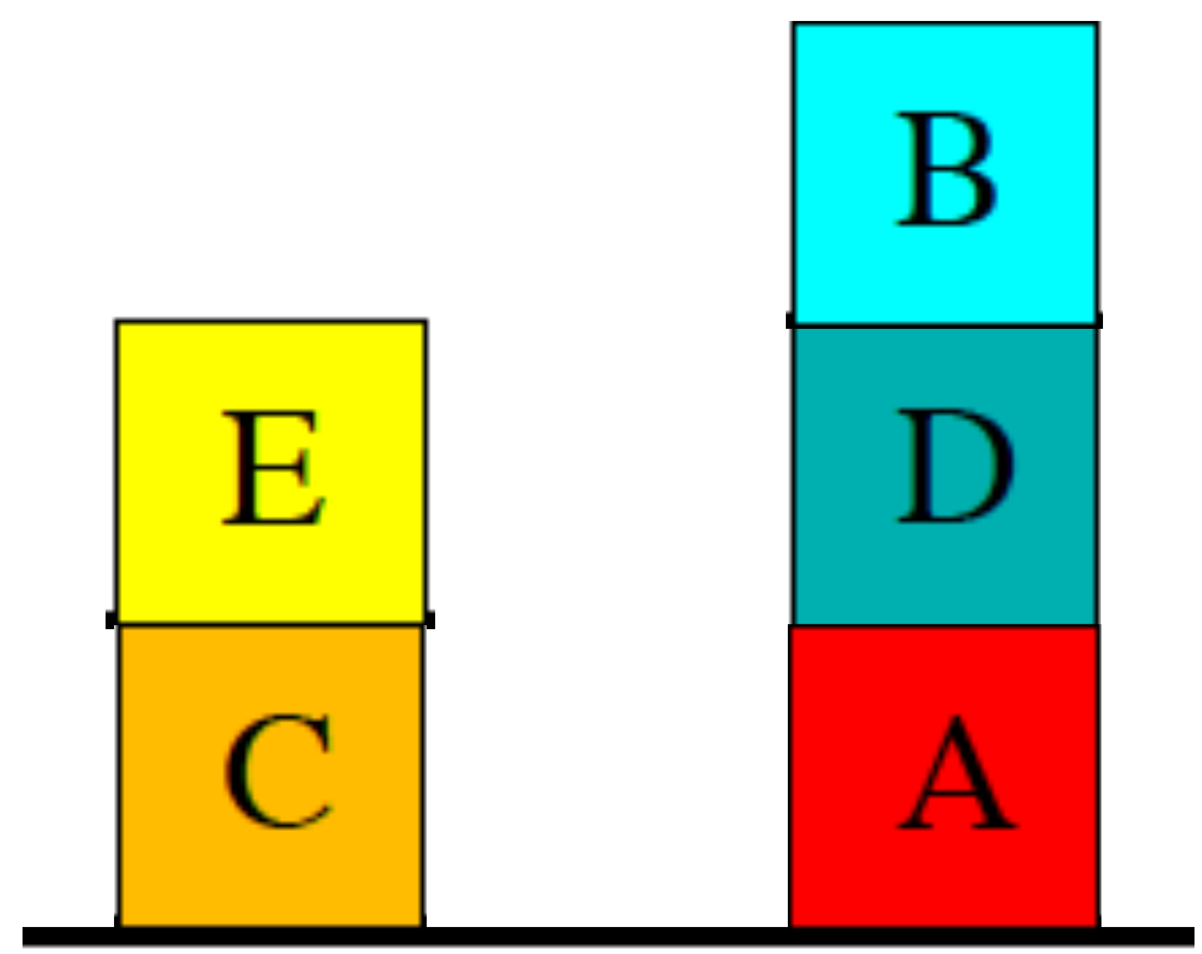


Predict the Minimum Plan Length

- Can **stack** / **unstack** anywhere on the ground
- Hint: is an **even** number



Initial State



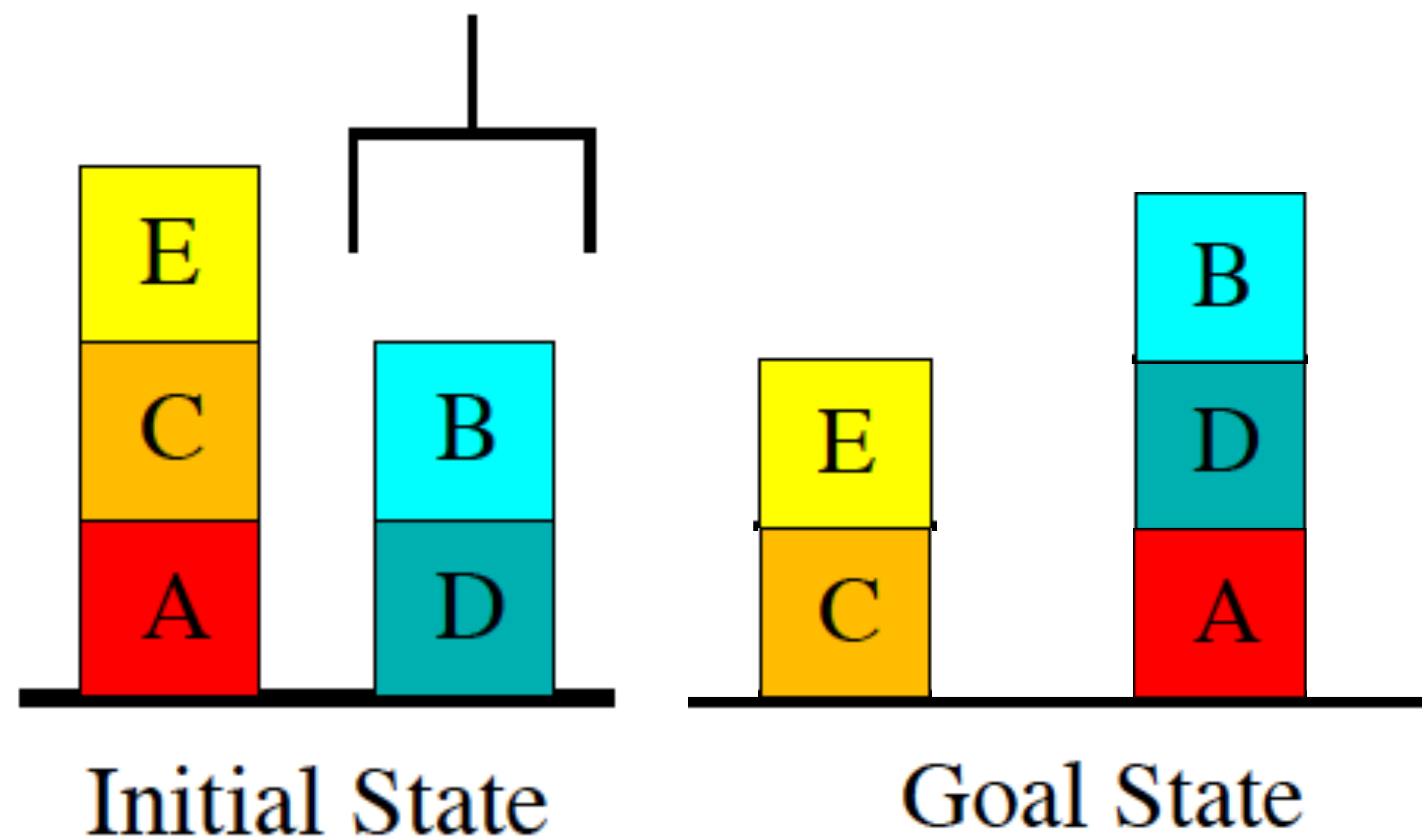
Goal State

Predict the Minimum Plan Length

- **Solution** (length=12):
  - **unstack**(E, C)
  - **stack**(E, ground)
  - **unstack**(C, A)
  - **stack**(C, ground)
  - **unstack**(E, ground)
  - **stack**(E, C)
  - **unstack**(B, D)
  - **stack**(B, ground)
  - **unstack**(D, ground)
  - **stack**(D, A)
  - **unstack**(B, ground)
  - **stack**(B, D)

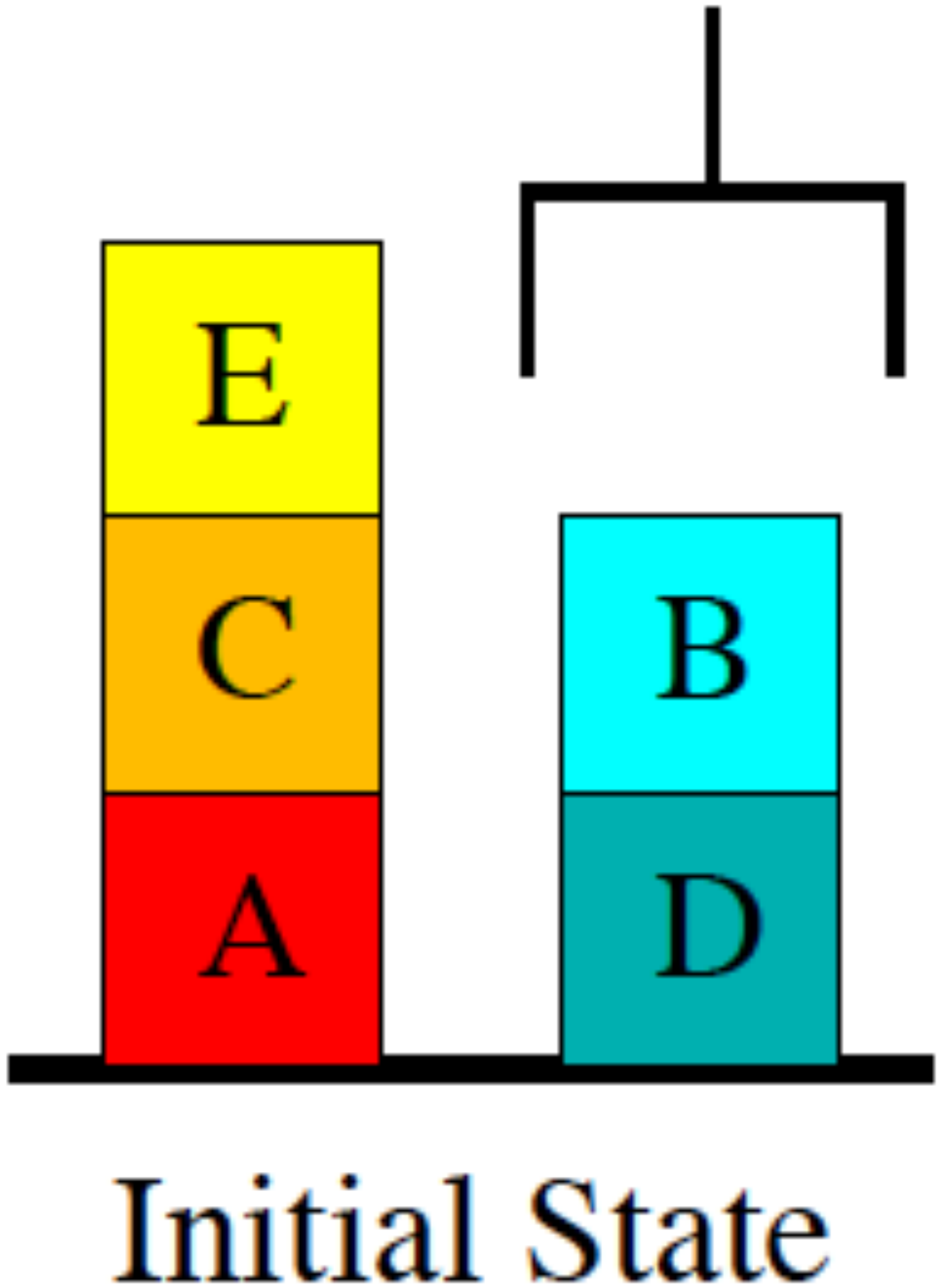
Need to **Restack**

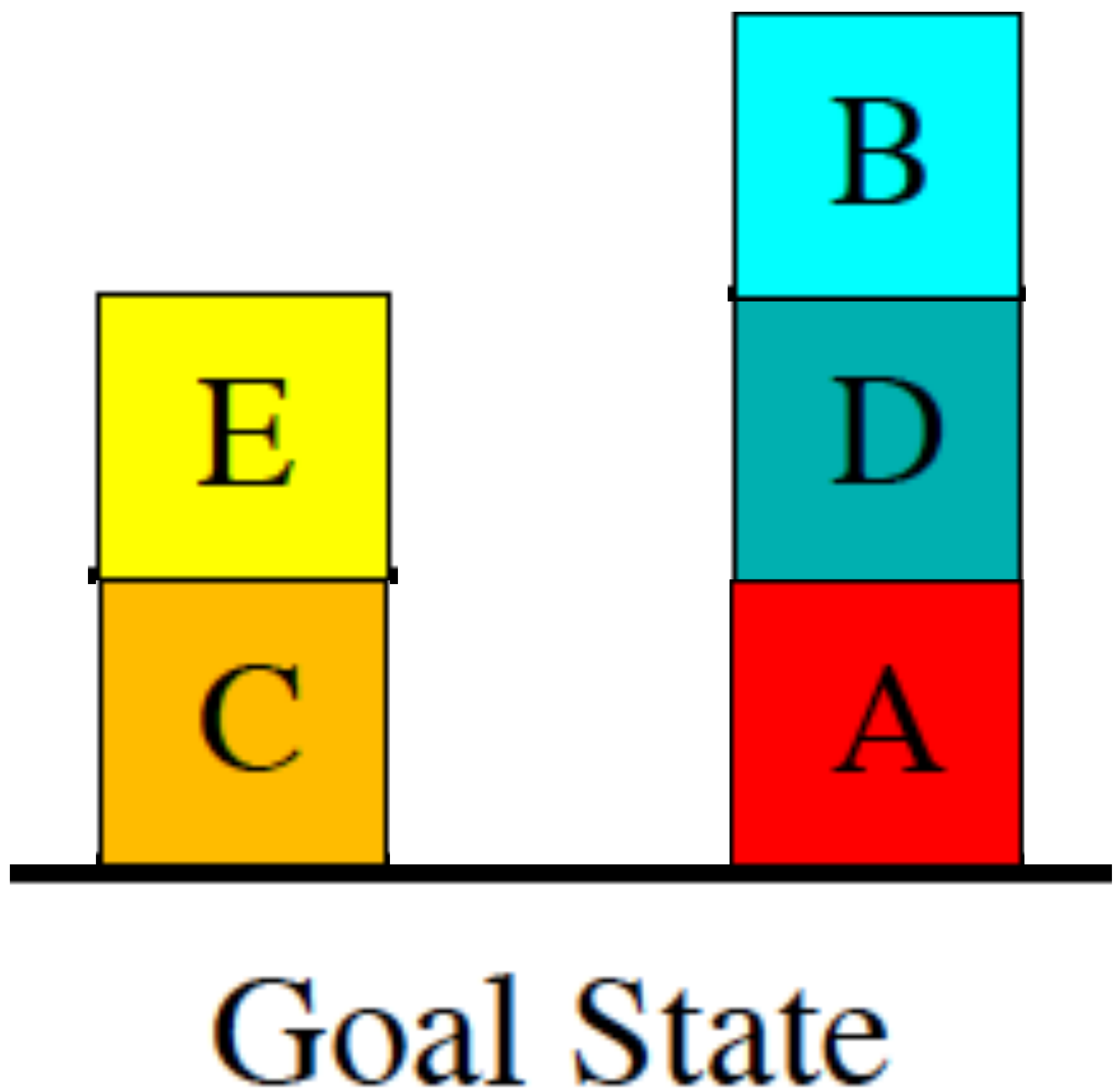
B and **E**



Predict the Minimum Delete-Relaxed Plan Length

- Can stack / unstack anywhere on the ground
- Hint: is **no greater** than 12

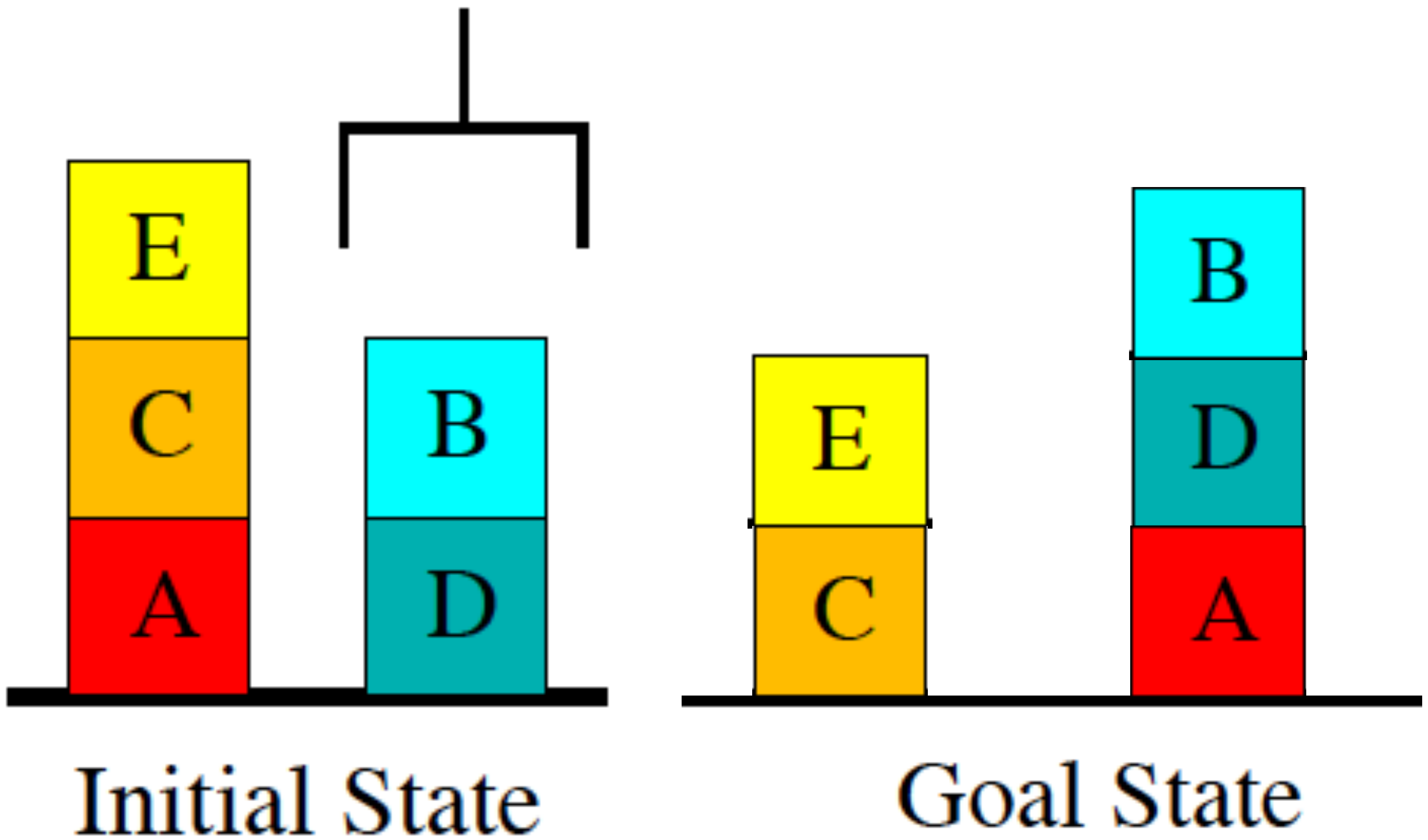




Predict the Minimum Delete-Relaxed Plan Length

- **Solution** (length=5):
  - **unstack**(E, C)
  - **unstack**(C, A)
  - **unstack**(B, D)
  - **unstack**(D, ground)
  - **stack**(D, A)

Only Need to
Unstack + Stack
1/5 Blocks



Motion Planning

Review: Motion Planning

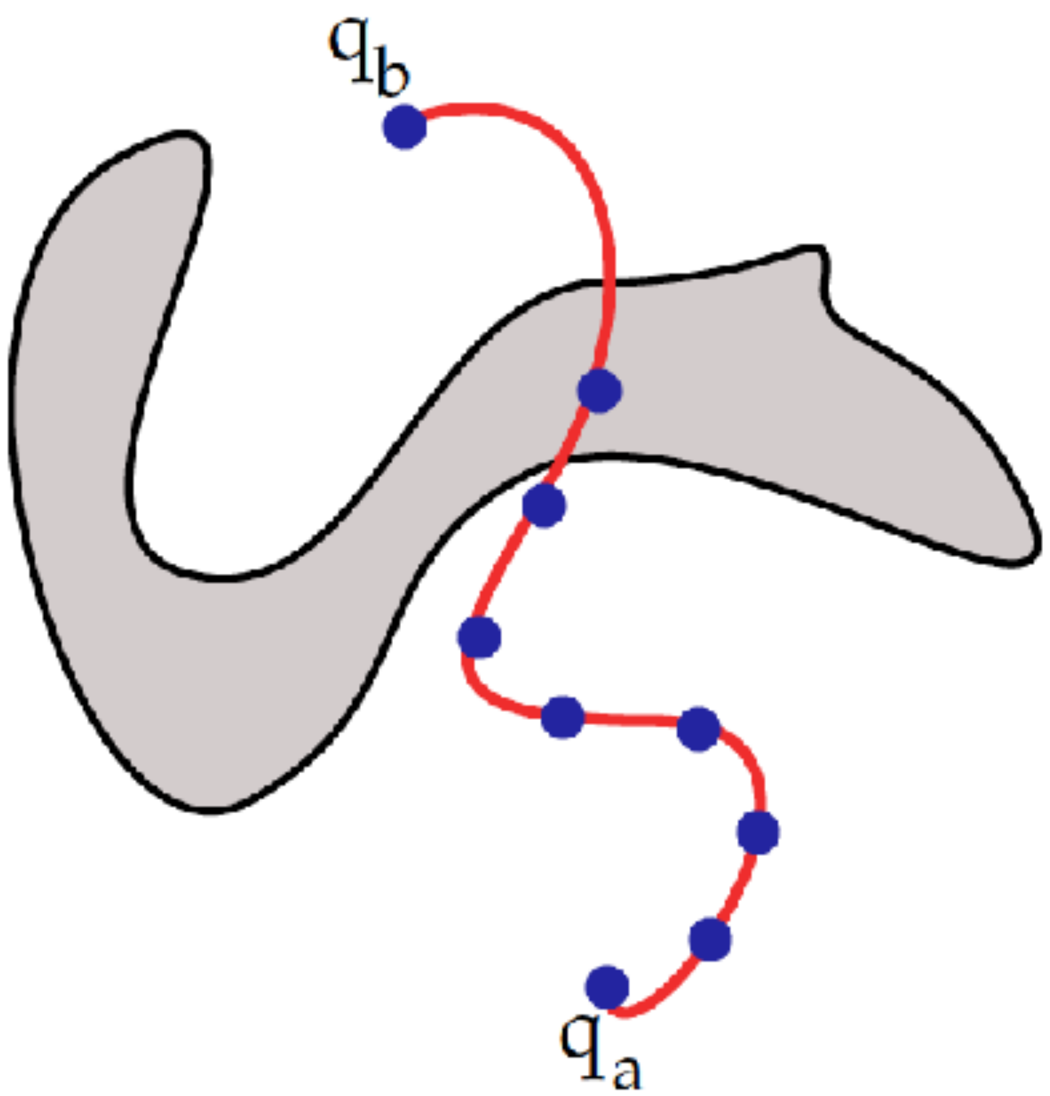
- Plan a **path** for a robot from an initial configuration to a goal configuration that **avoids obstacles**
  - Sequence of **continuous** configurations
  - Configurations often are **high-dimensional**
    - Example: 7 DOFs

- **High-level approaches:**
  - Geometric decomposition
  - <u>Sampling-based</u>
  - Grid-based
  - Optimization-based

Sampling-Based Motion Planning

- **Discretize** configuration space by **sampling**
  - Sampling be deterministic or **random**
- **Implicitly** represent the collision-free configuration space using an blackbox **collision checker**

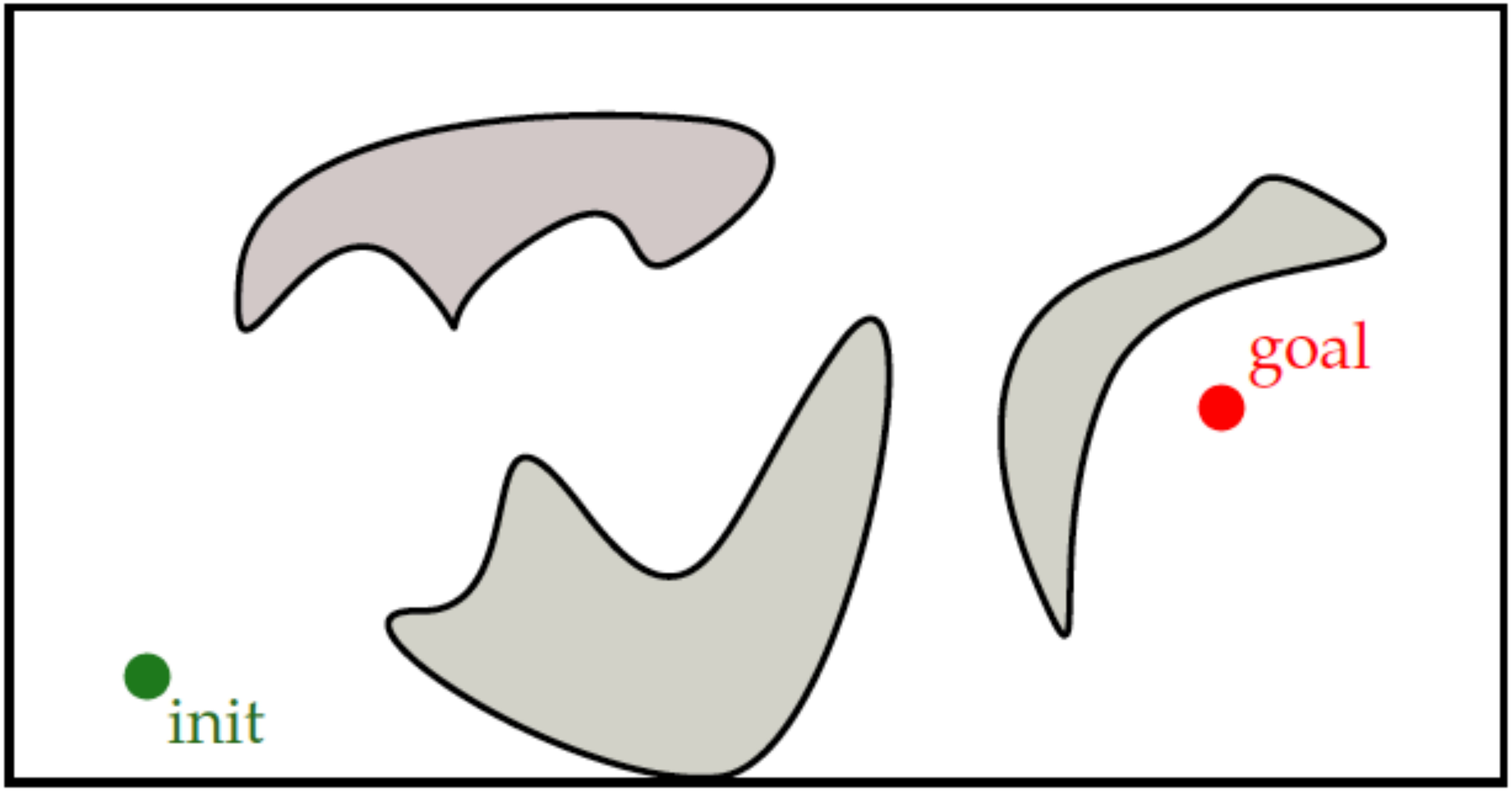
- **Algorithms**
  - Probabilistic Roadmap (PRM)
  - Rapidly-Exploring Random Tree (RRT)
  - Bidirectional RRT (BiRRT)
  - **cuRobo** PRM Global Planning



[Kavraki 1994][Kuffner 2000][LaValle 2006][Sundaralingam 2022]

[Fig from Erion Plaku]

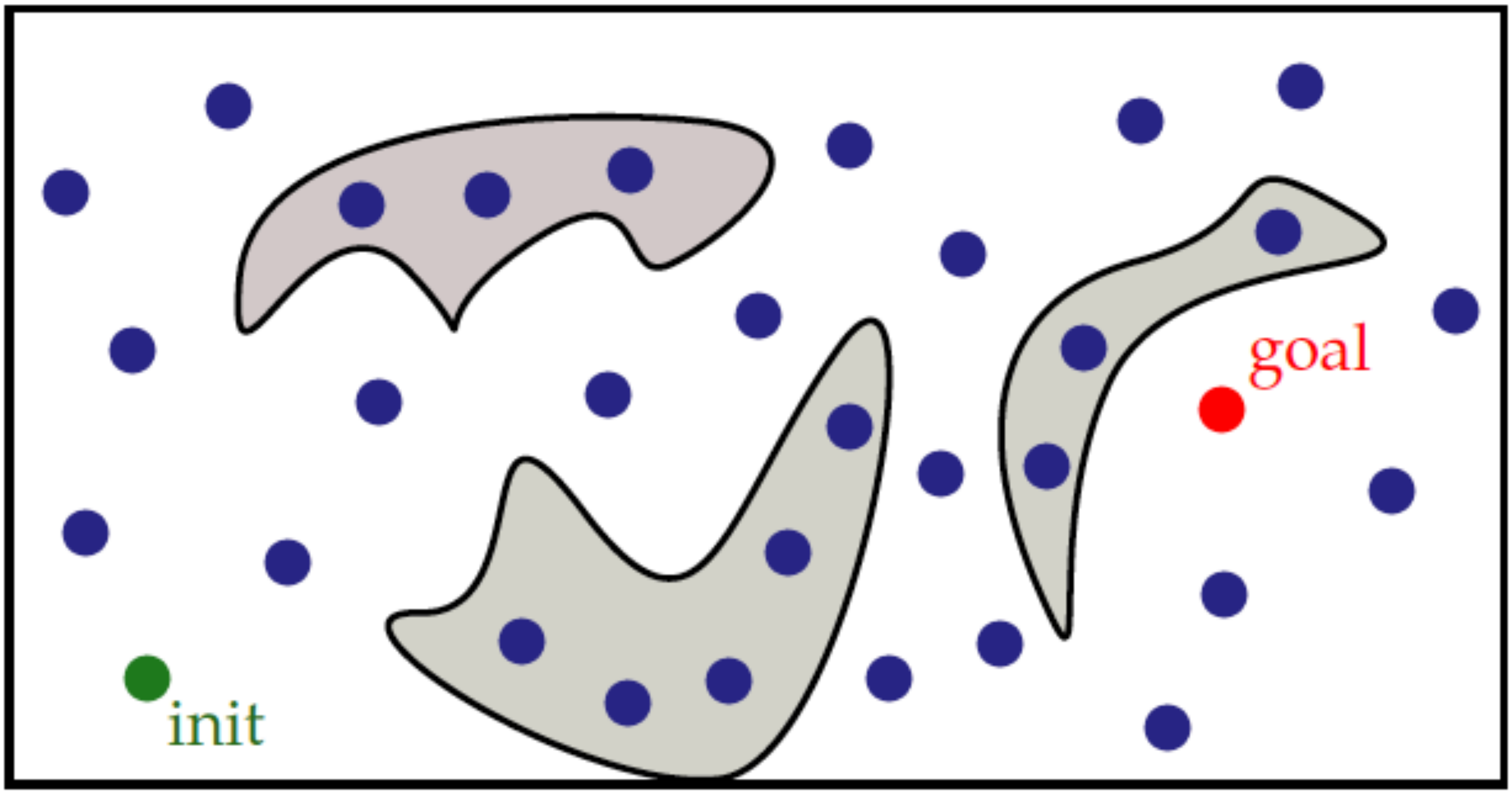
Probabilistic Roadmap (1/7)



[Fig from Erion Plaku]

Find a path from init to goal that avoids the obstacles

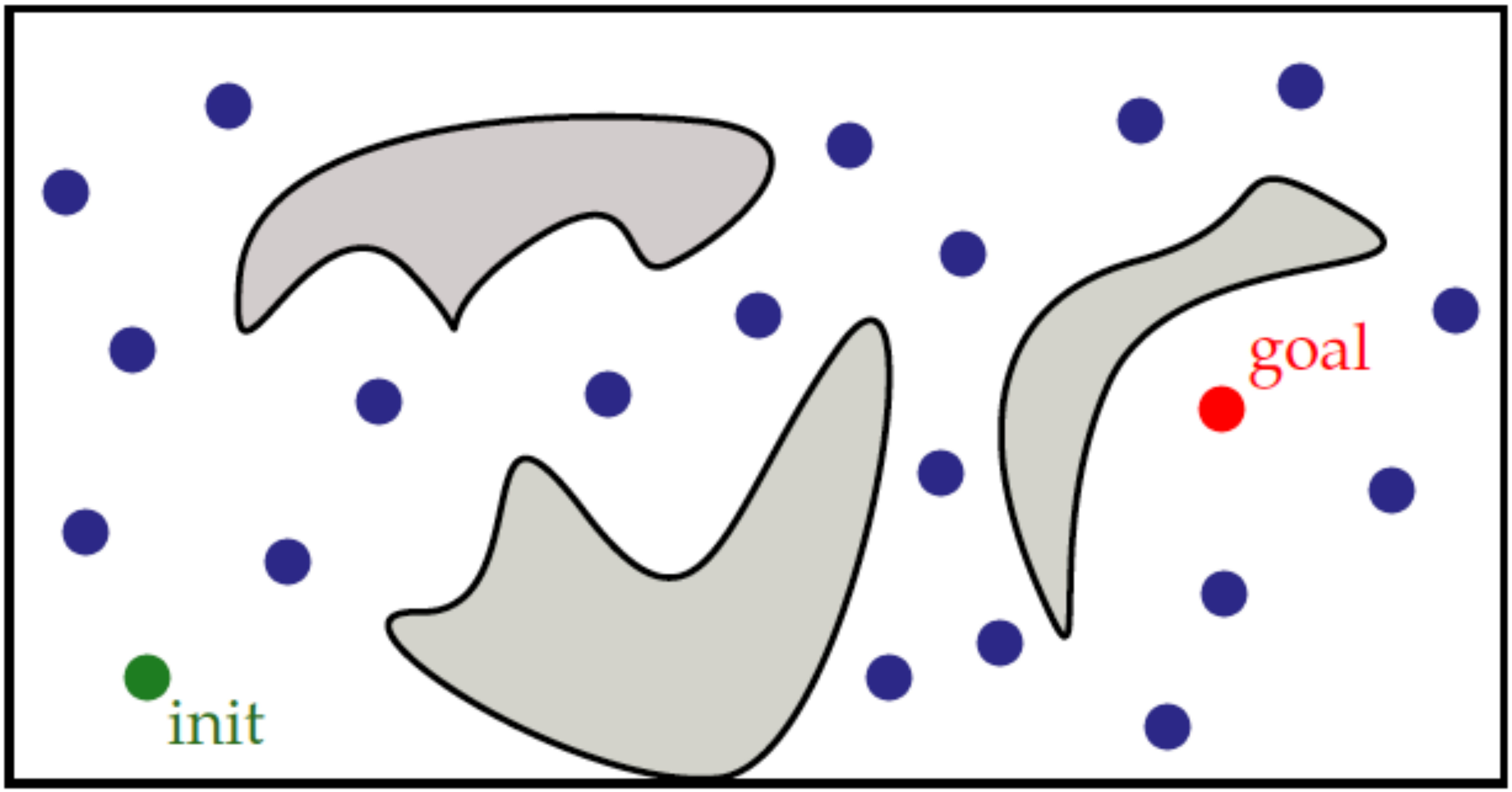
Probabilistic Roadmap (2/7)



[Fig from Erion Plaku]

Sample a set of configurations

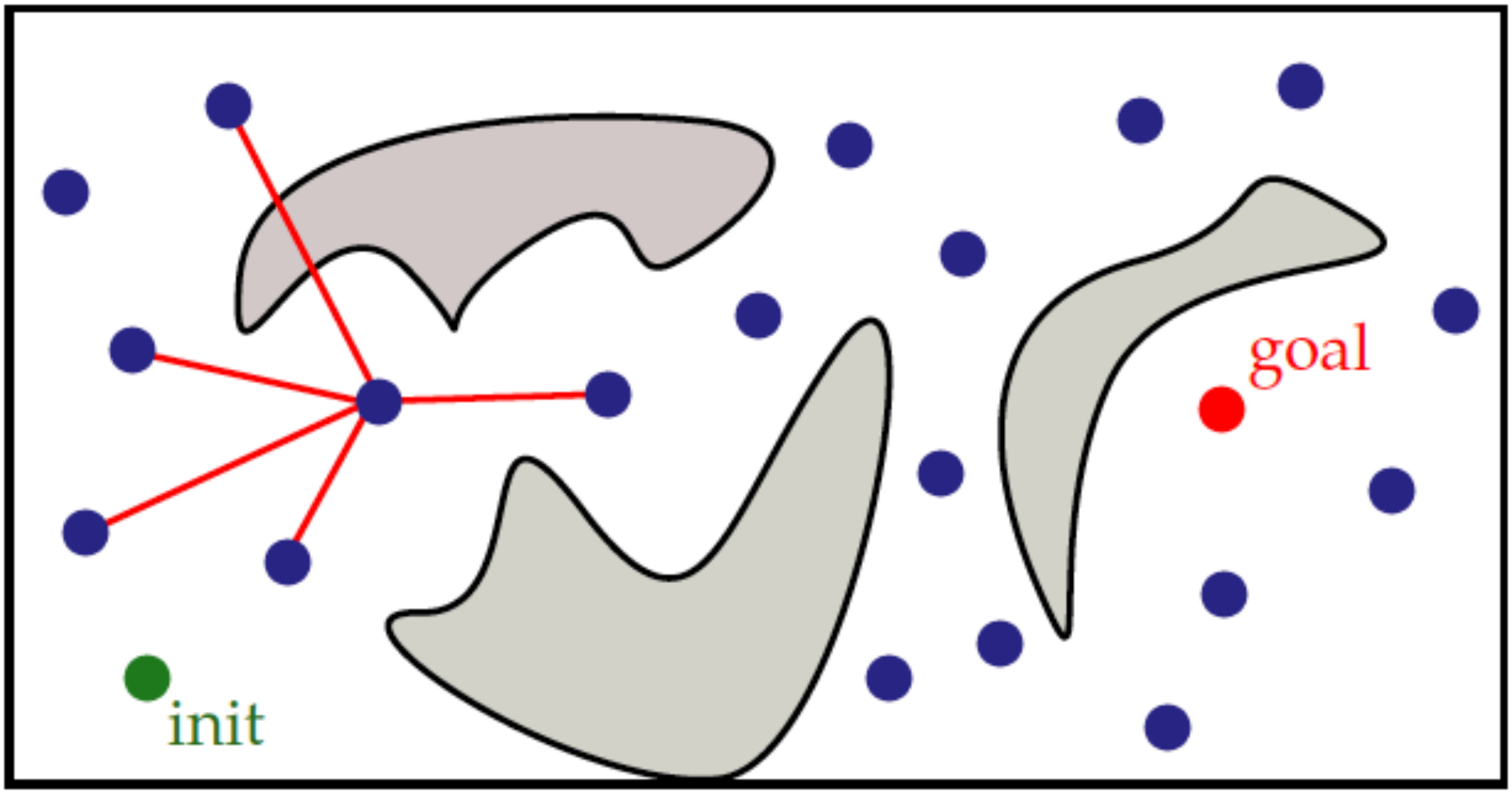
Probabilistic Roadmap (3/7)



[Fig from Erion Plaku]

Remove configurations that collide with the obstacles

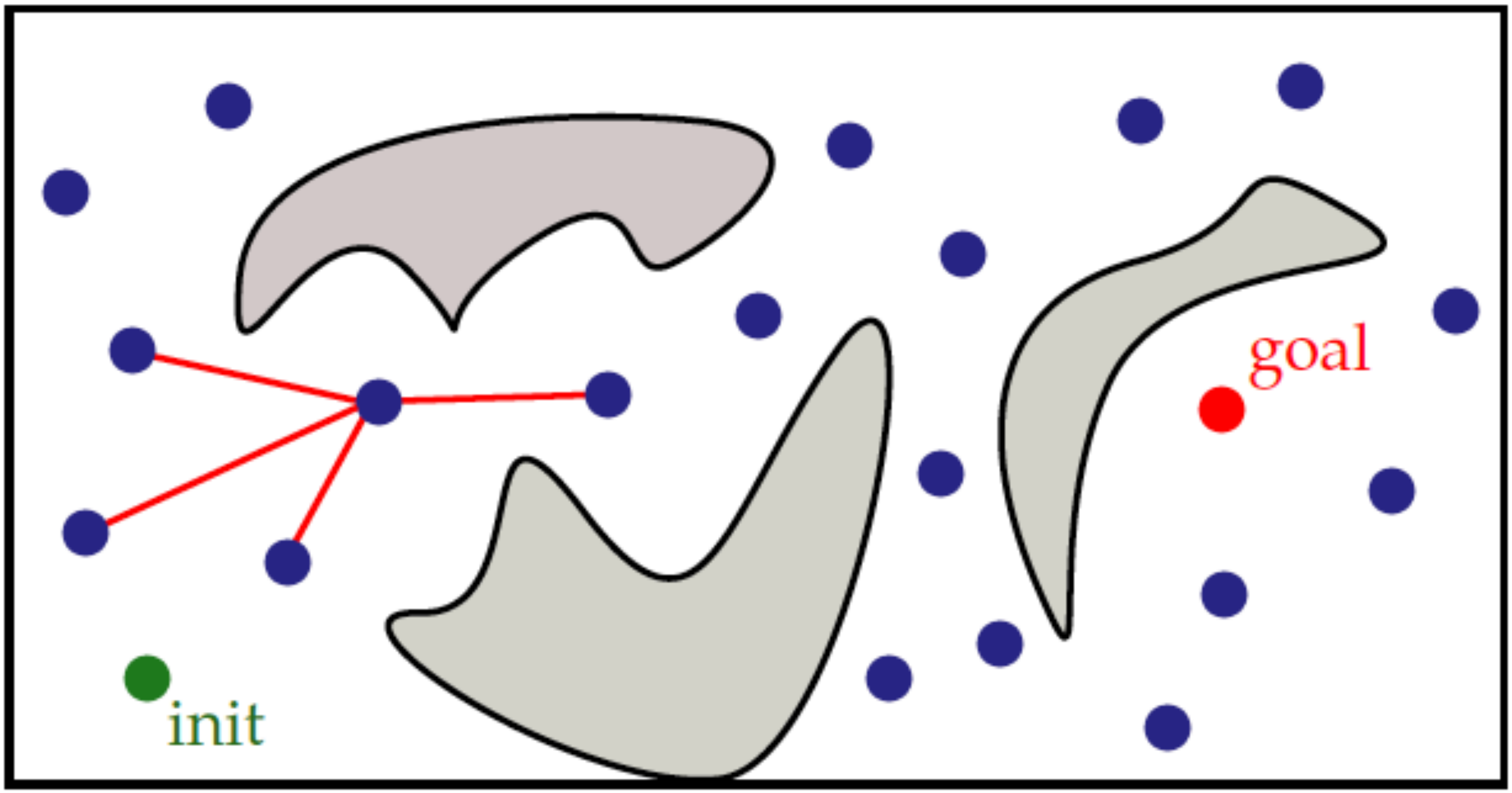
Probabilistic Roadmap (4/7)



[Fig from Erion Plaku]

Connect nearby configurations

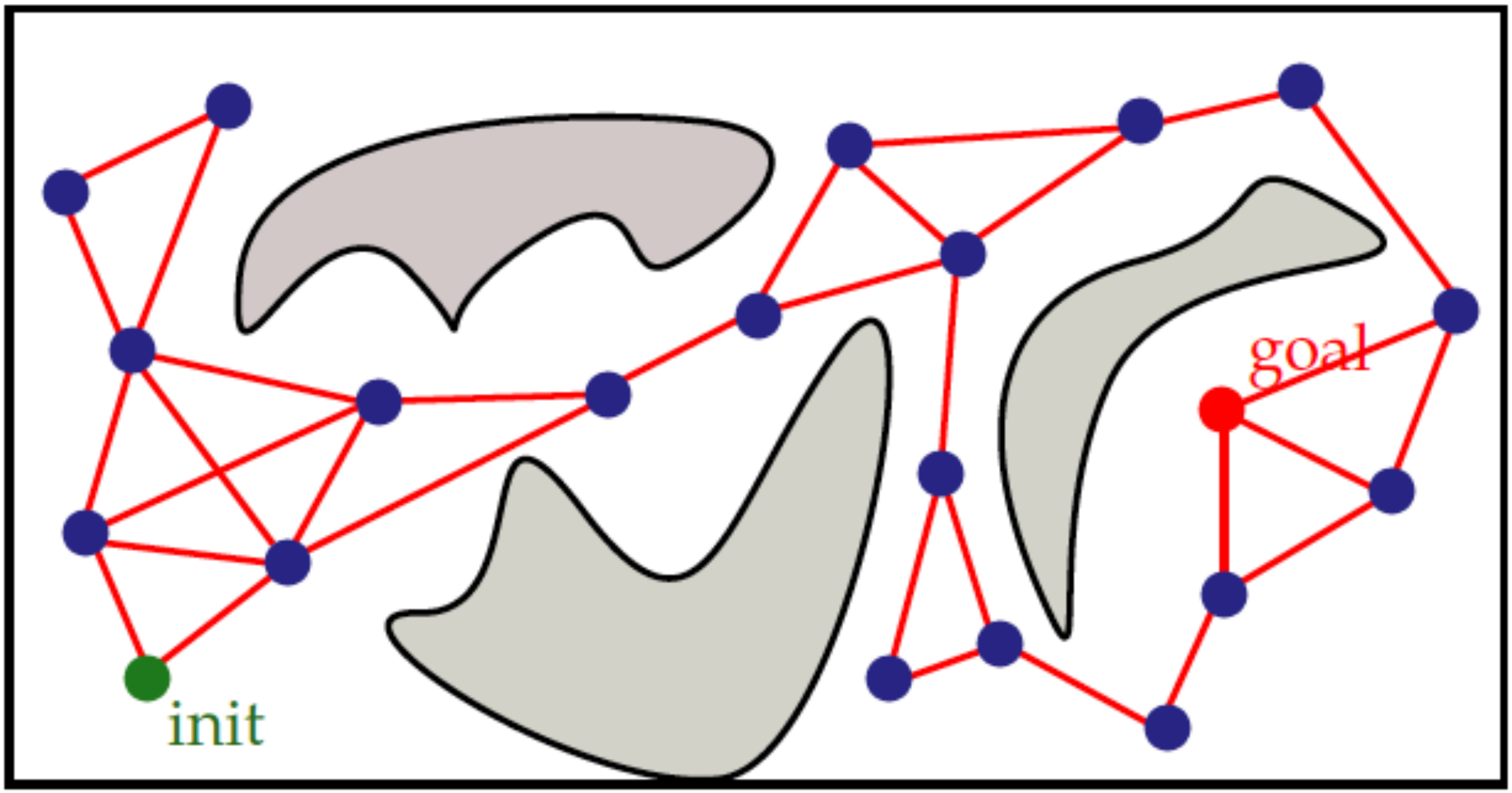
Probabilistic Roadmap (5/7)



[Fig from Erion Plaku]

Prune connections that collide with the obstacles

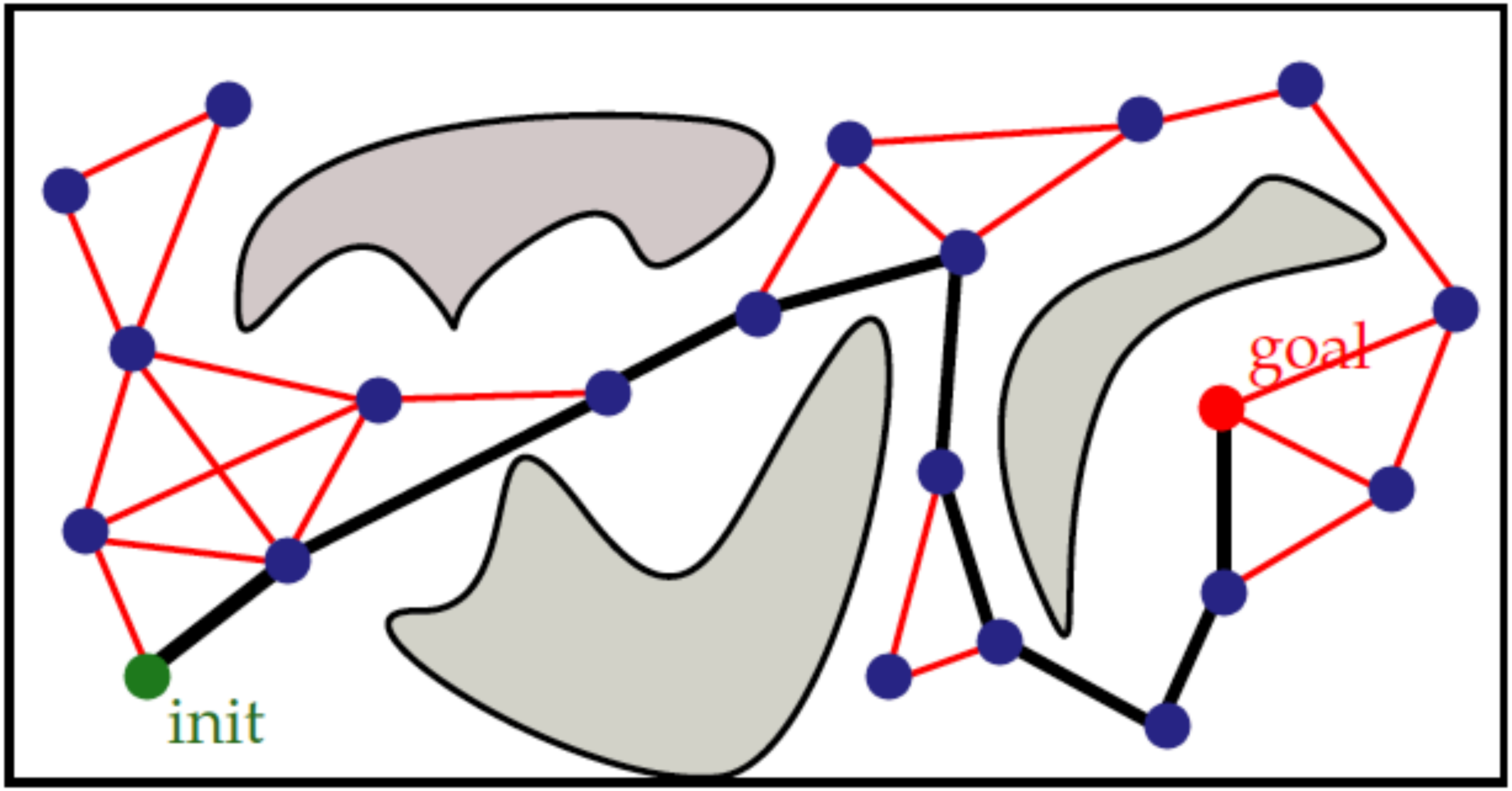
Probabilistic Roadmap (6/7)



[Fig from Erion Plaku]

The resulting structure is a finite roadmap (graph)

Probabilistic Roadmap (7/7)

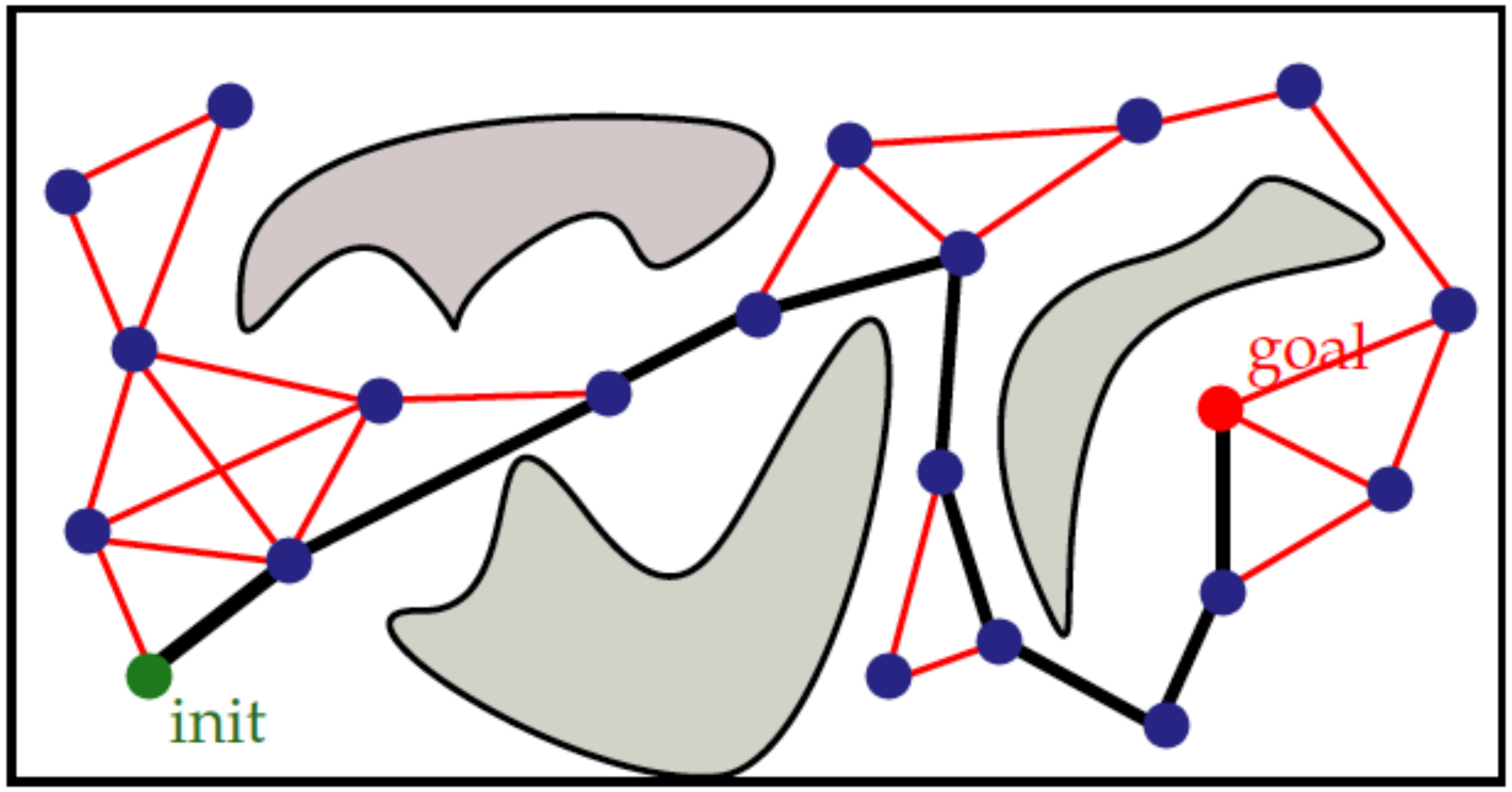


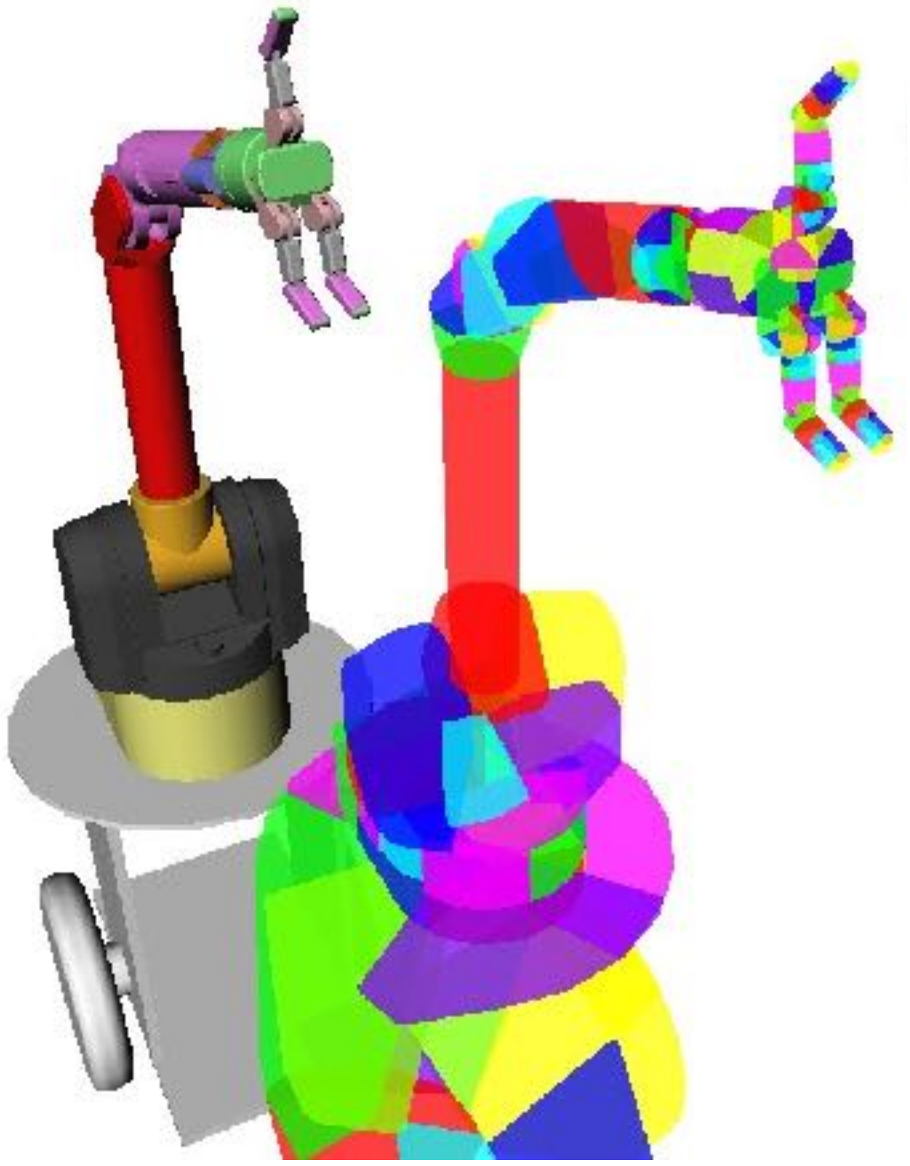
[Fig from Erion Plaku]

Search for the shortest-path on the roadmap

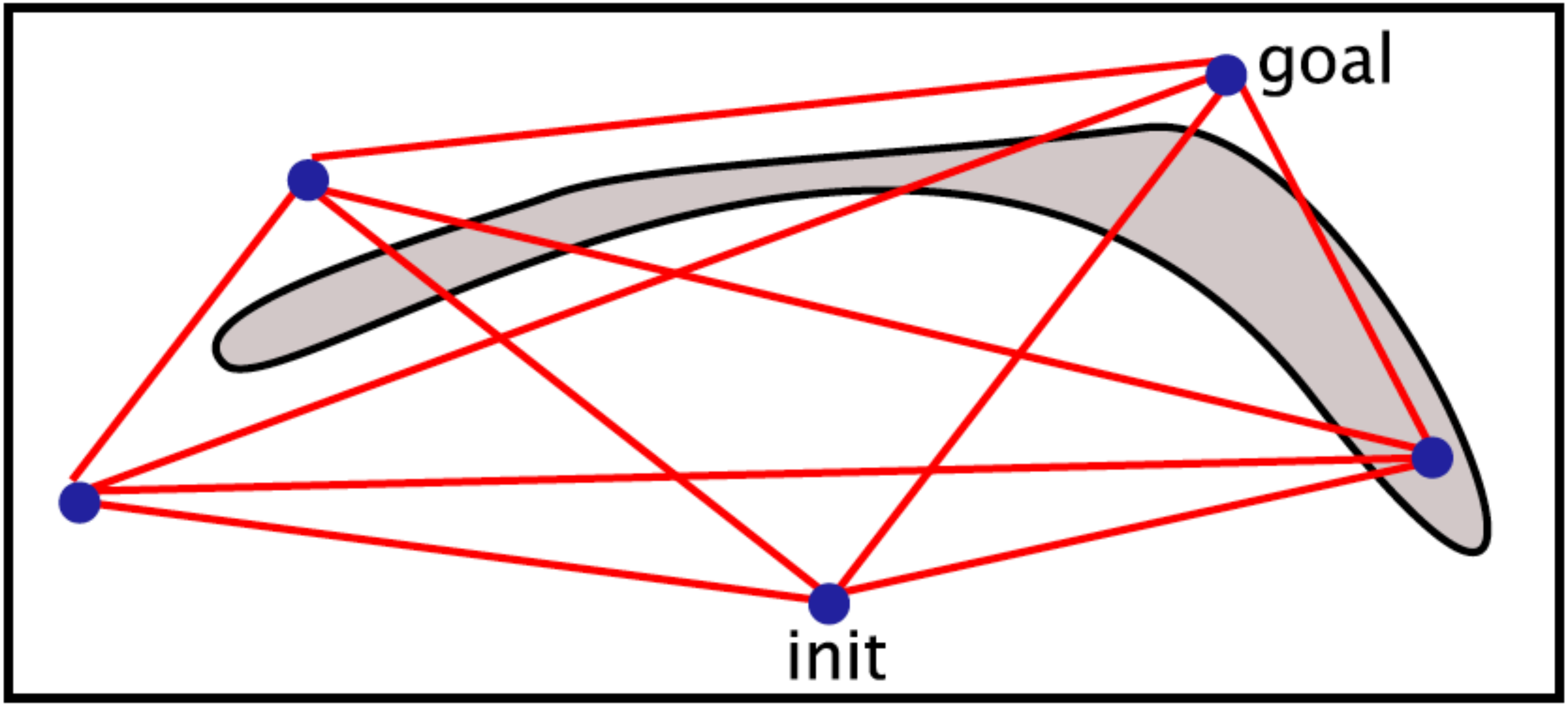
Collision Checking is Expensive

- Collision checking **dominates** runtime
  - **Complex geometries** & **fine resolutions** (for safety)
- Many edges clearly do not lie on a low-cost path
- **Optimistically** plan **without collisions**
- Check collisions **lazily** by evaluating only on **candidate plans**





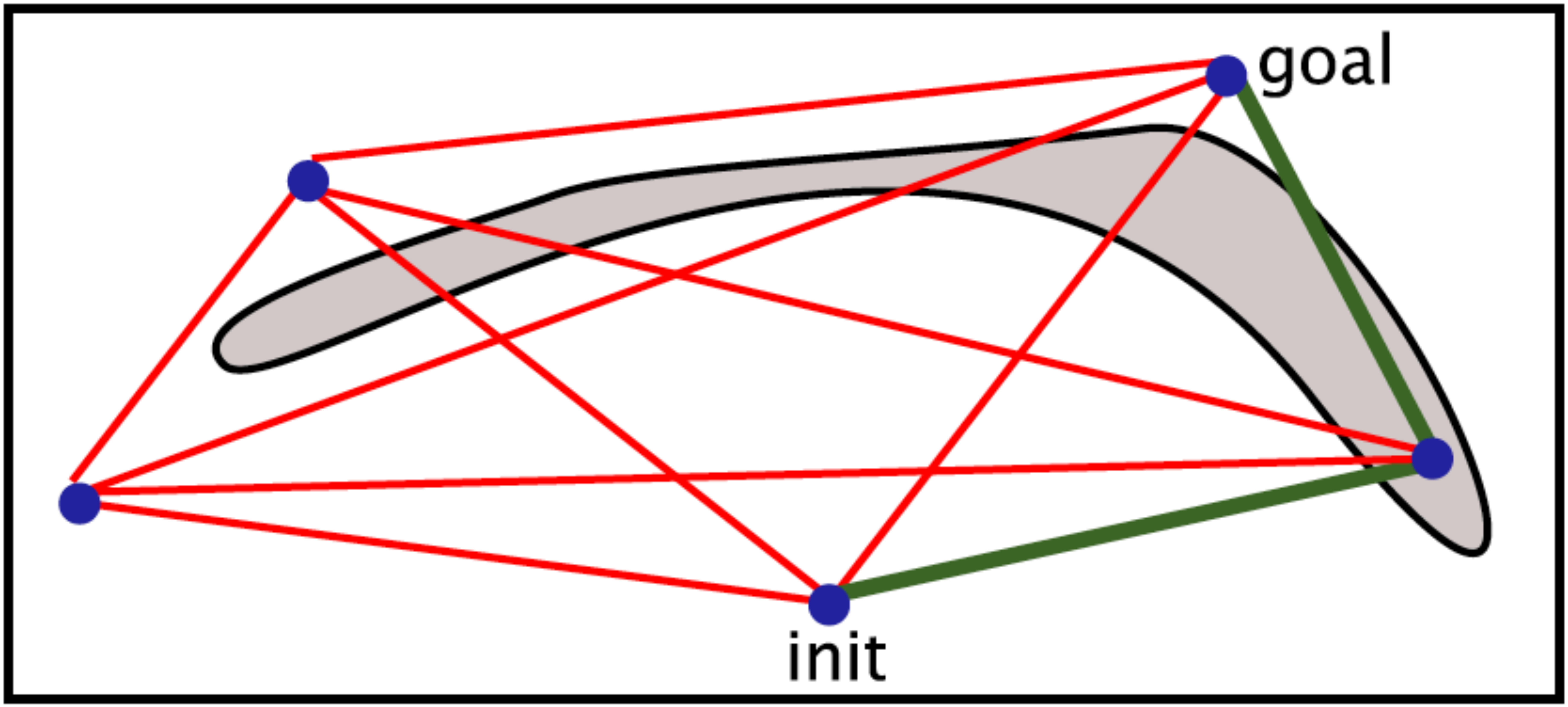
Lazy PRM (1/10)



[Fig from Erion Plaku]

Construct a PRM ignoring collisions

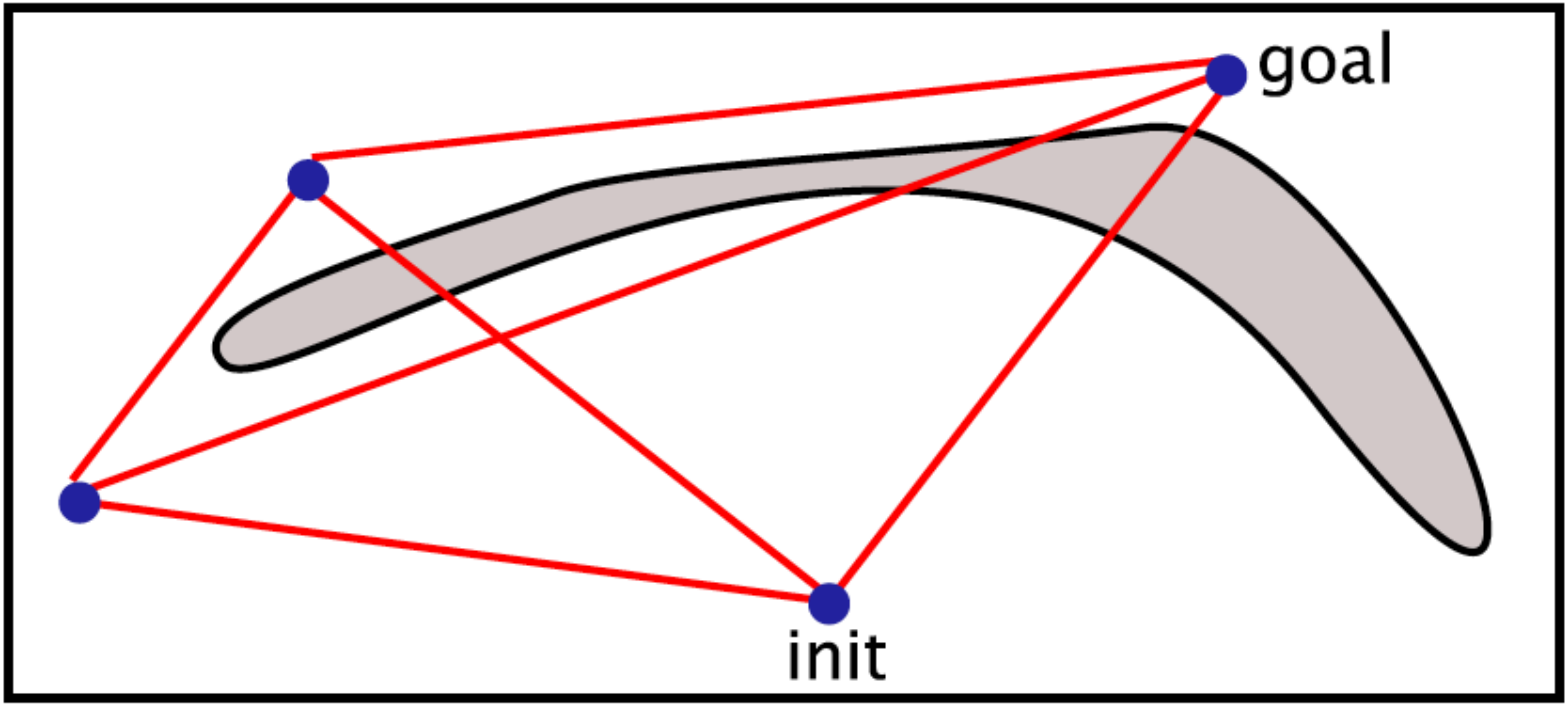
Lazy PRM (2/10)



[Fig from Erion Plaku]

Search for the shortest-path on the roadmap

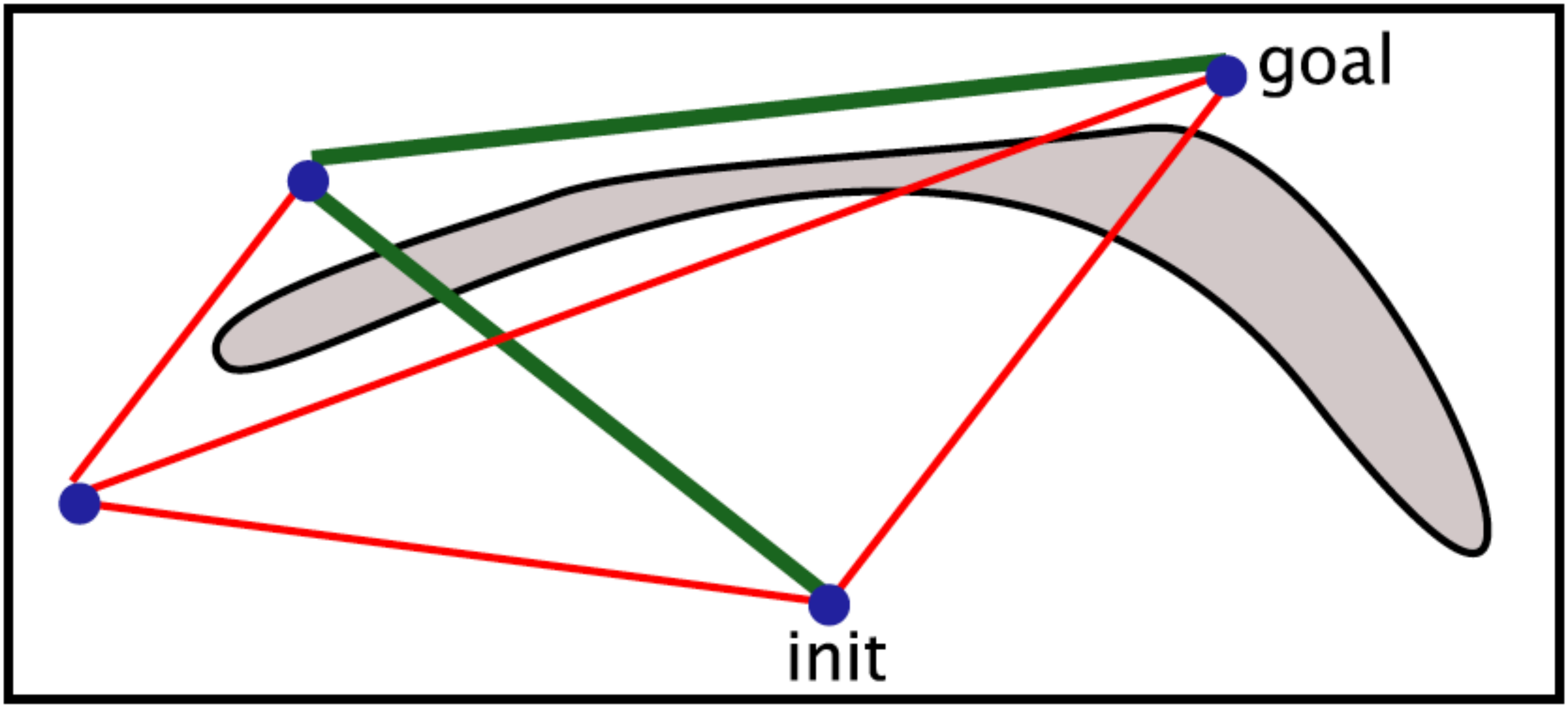
Lazy PRM (3/10)



[Fig from Erion Plaku]

Remove plan edges that collide with obstacles

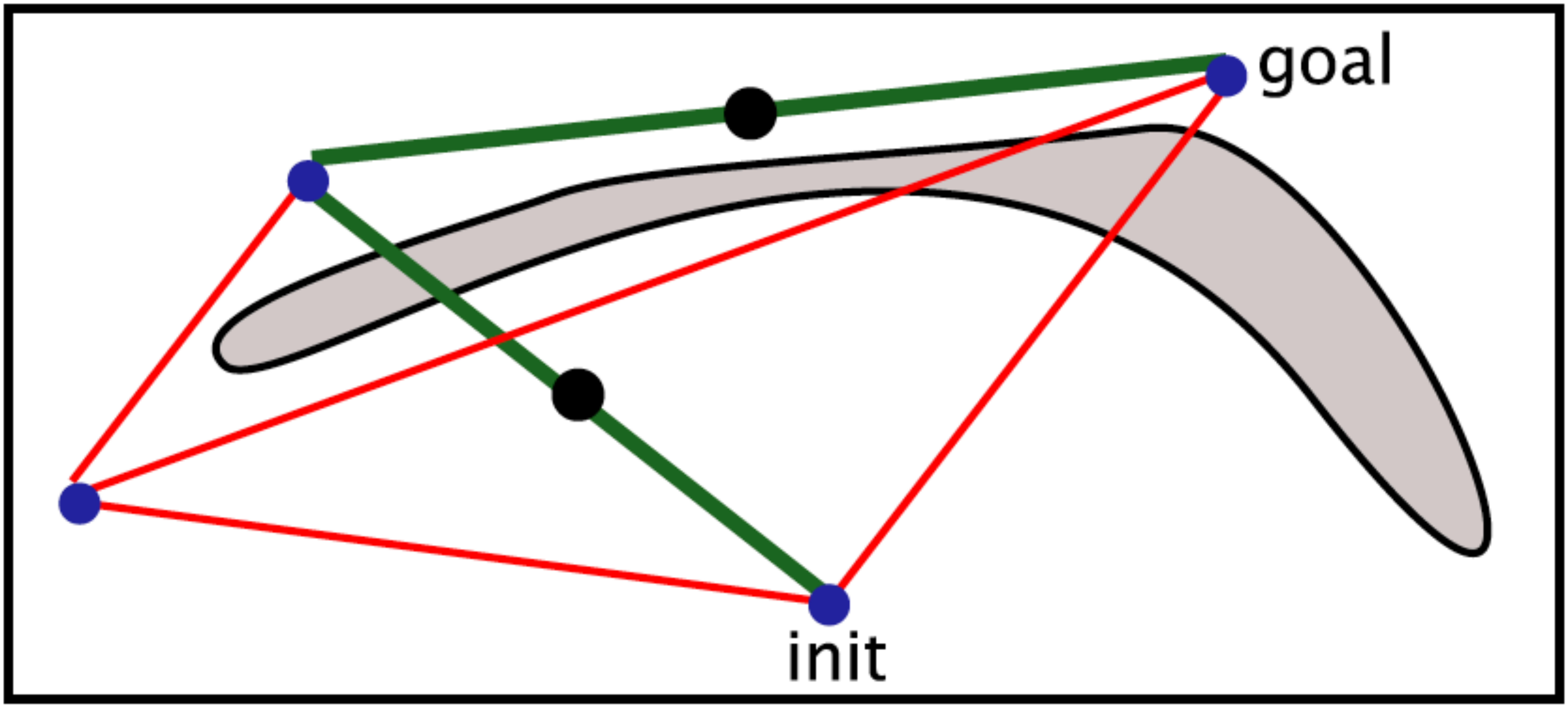
Lazy PRM (4/10)



[Fig from Erion Plaku]

Search for the new shortest-path on the roadmap

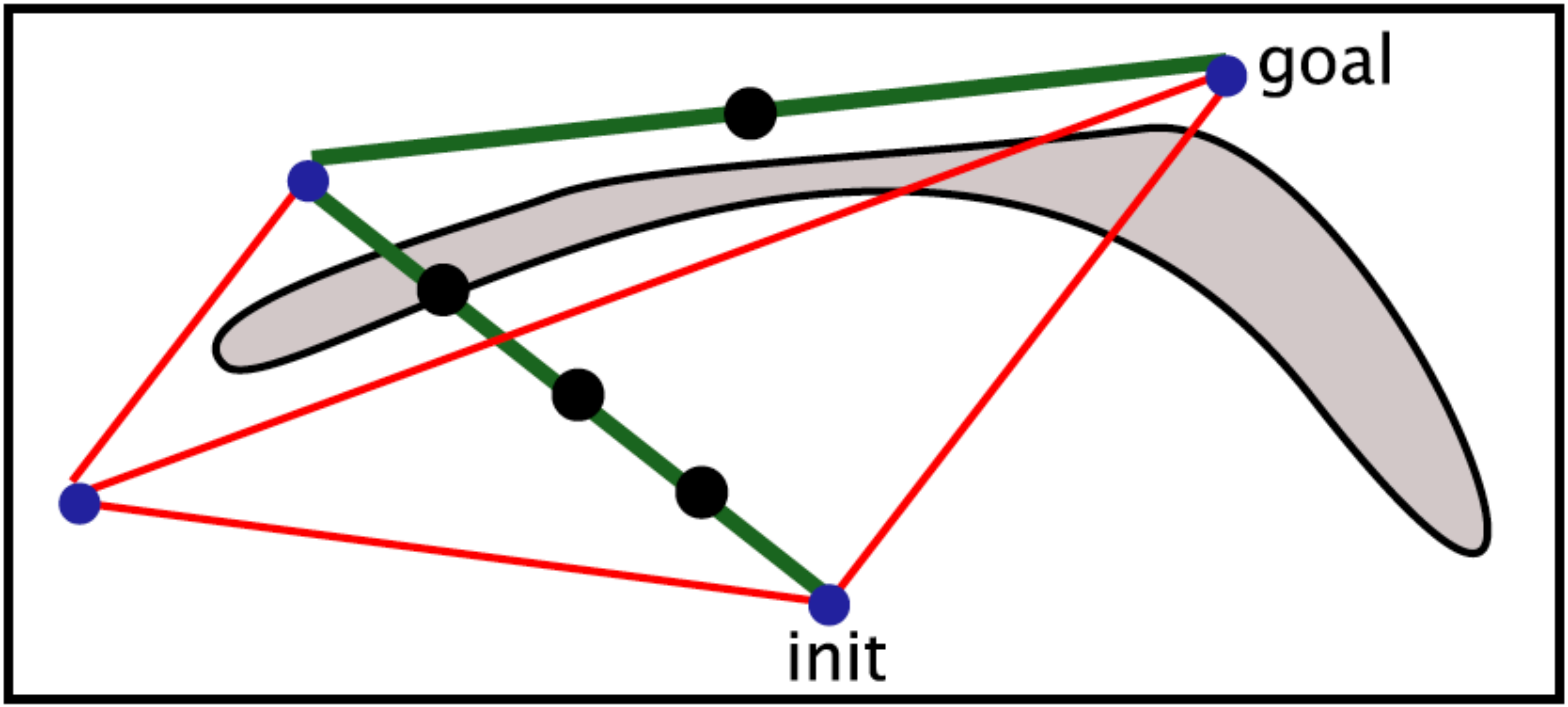
Lazy PRM (5/10)



[Fig from Erion Plaku]

Check the edges on the plan for collisions

Lazy PRM (6/10)

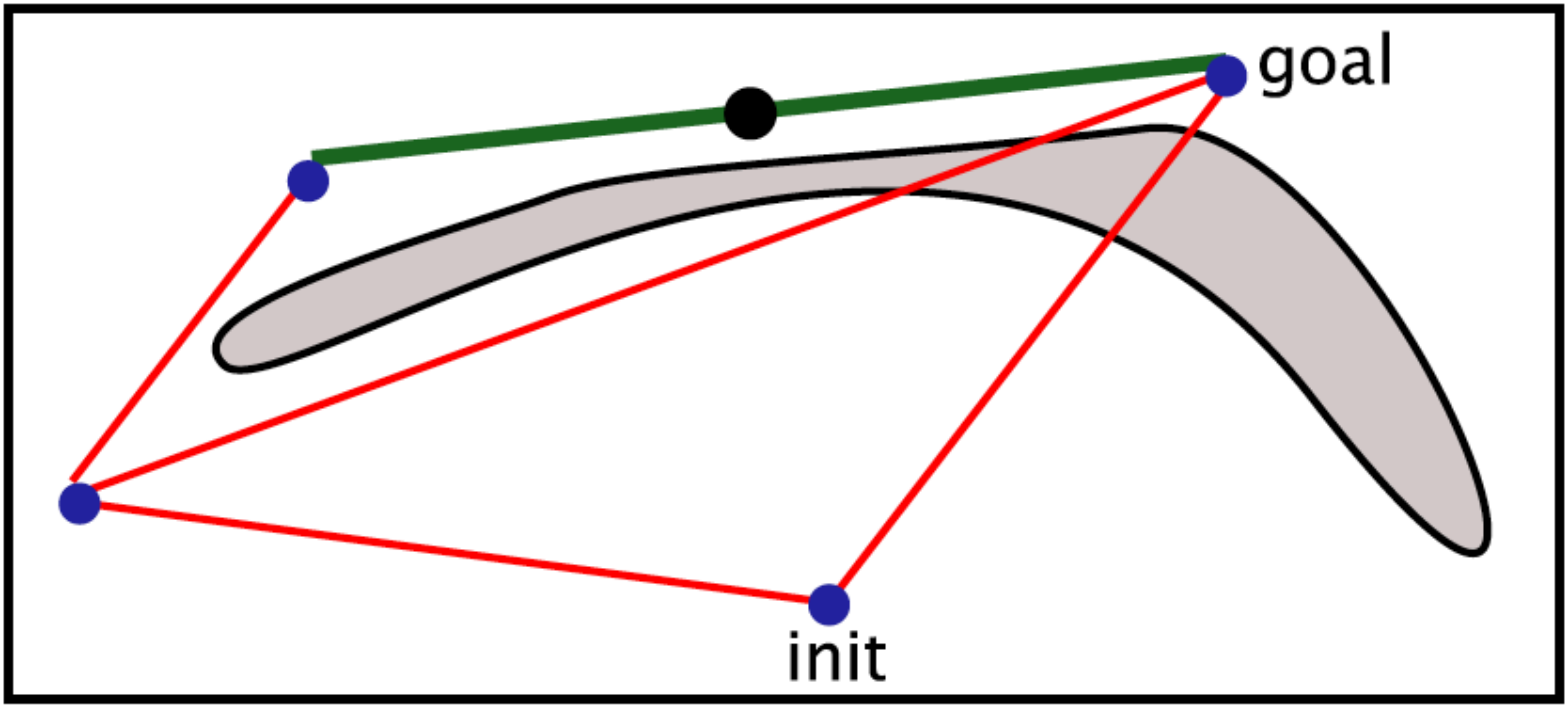


[Fig from Erion Plaku]

Check the edges on the plan for collisions

(with increased resolution)

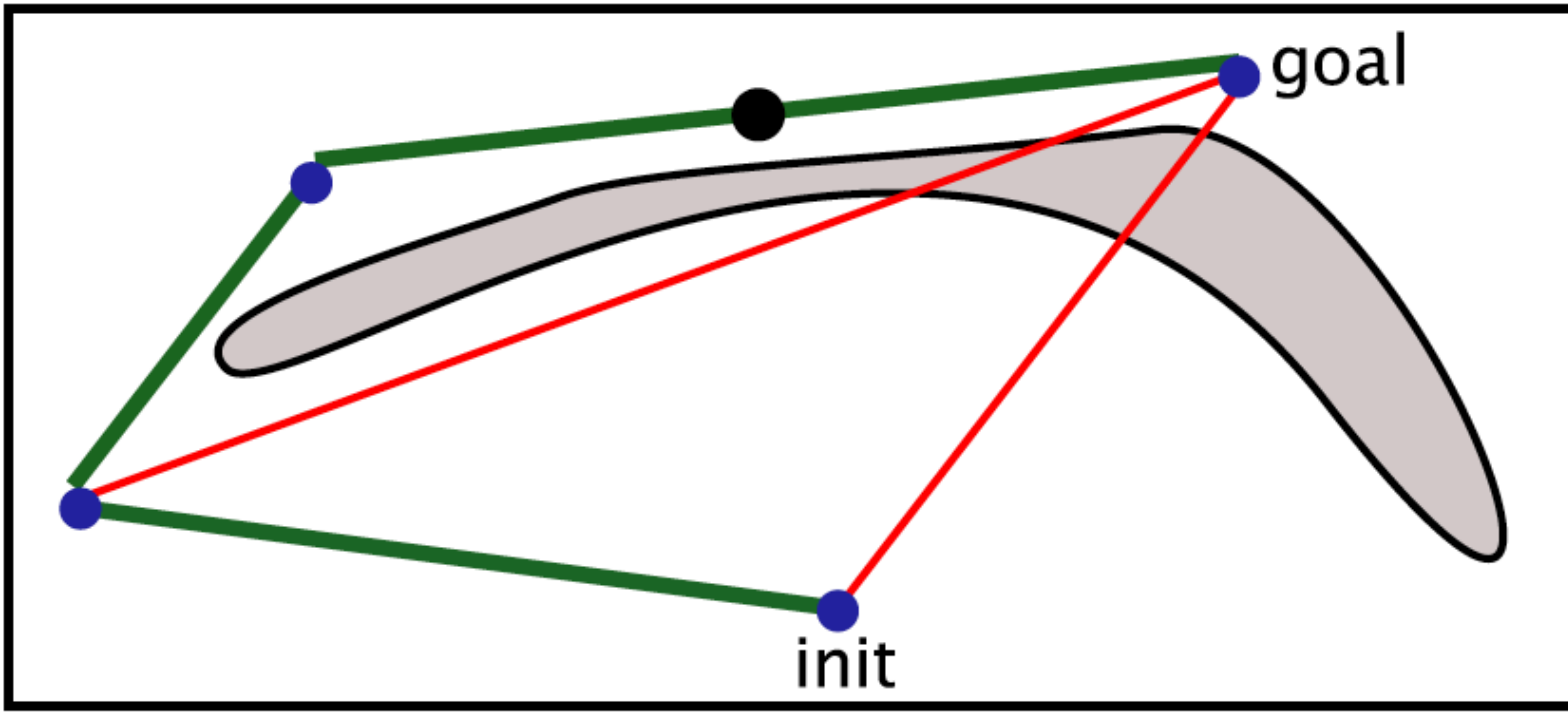
Lazy PRM (7/10)



[Fig from Erion Plaku]

Remove plan edges that collide with obstacles

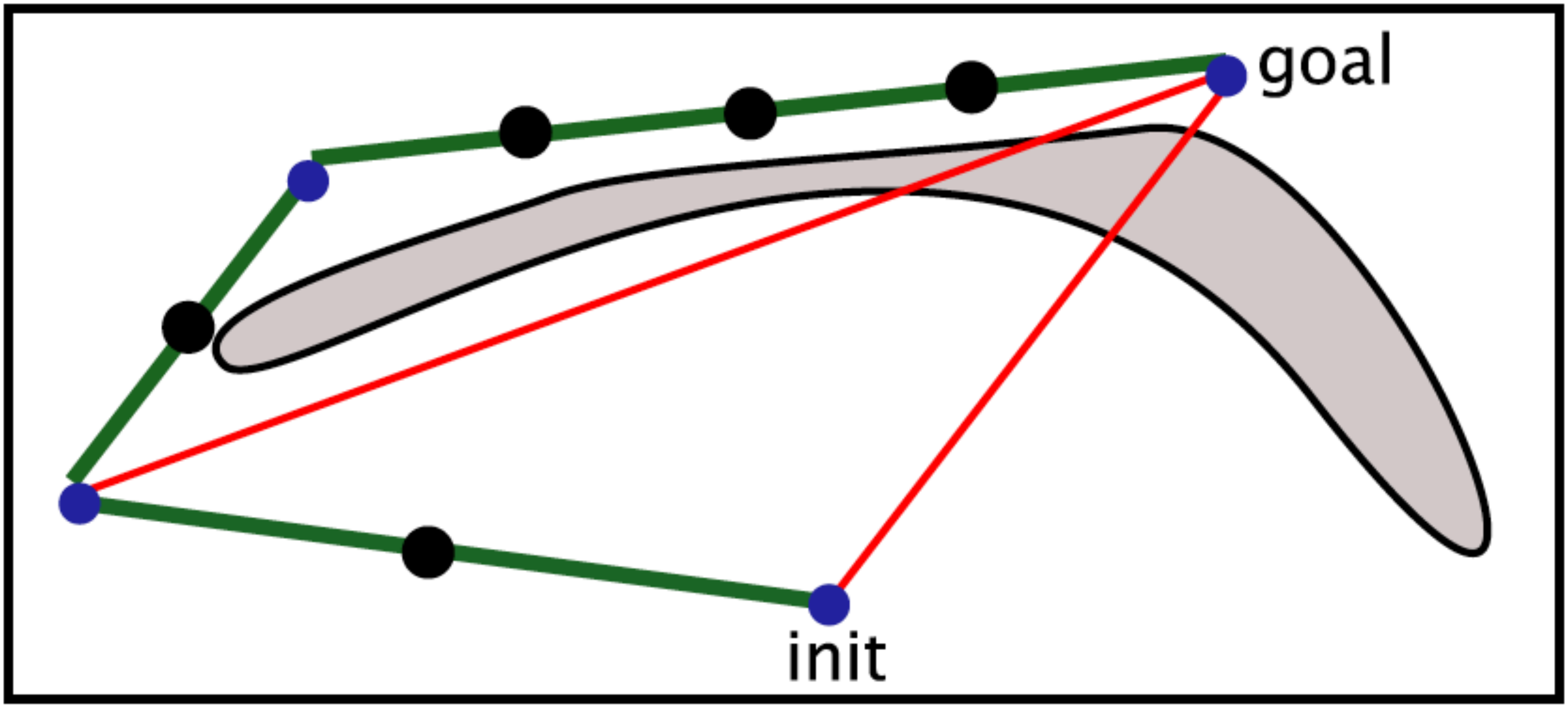
Lazy PRM (8/10)



[Fig from Erion Plaku]

Search for the new shortest-path on the roadmap

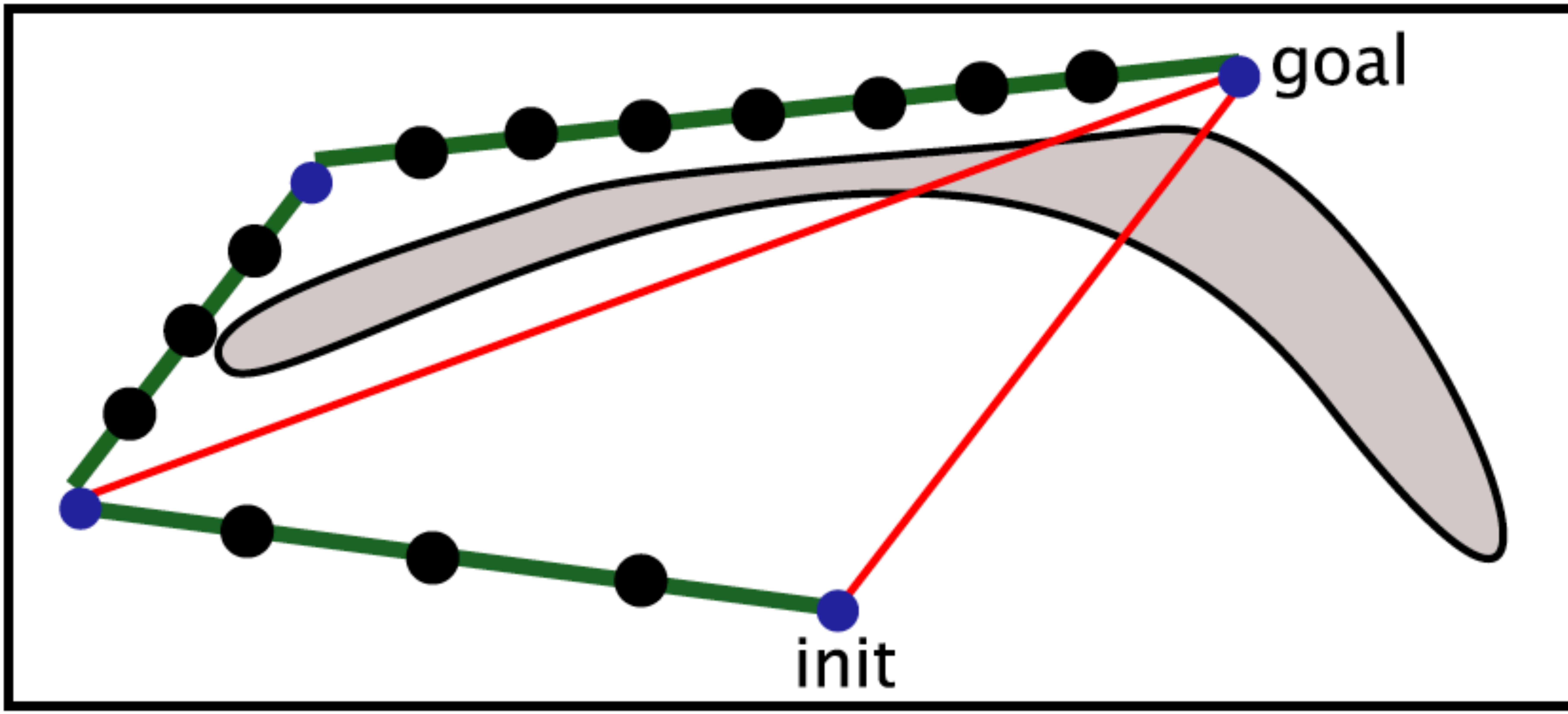
Lazy PRM (9/10)



[Fig from Erion Plaku]

Check the edges on the plan for collisions

Lazy PRM (10/10)



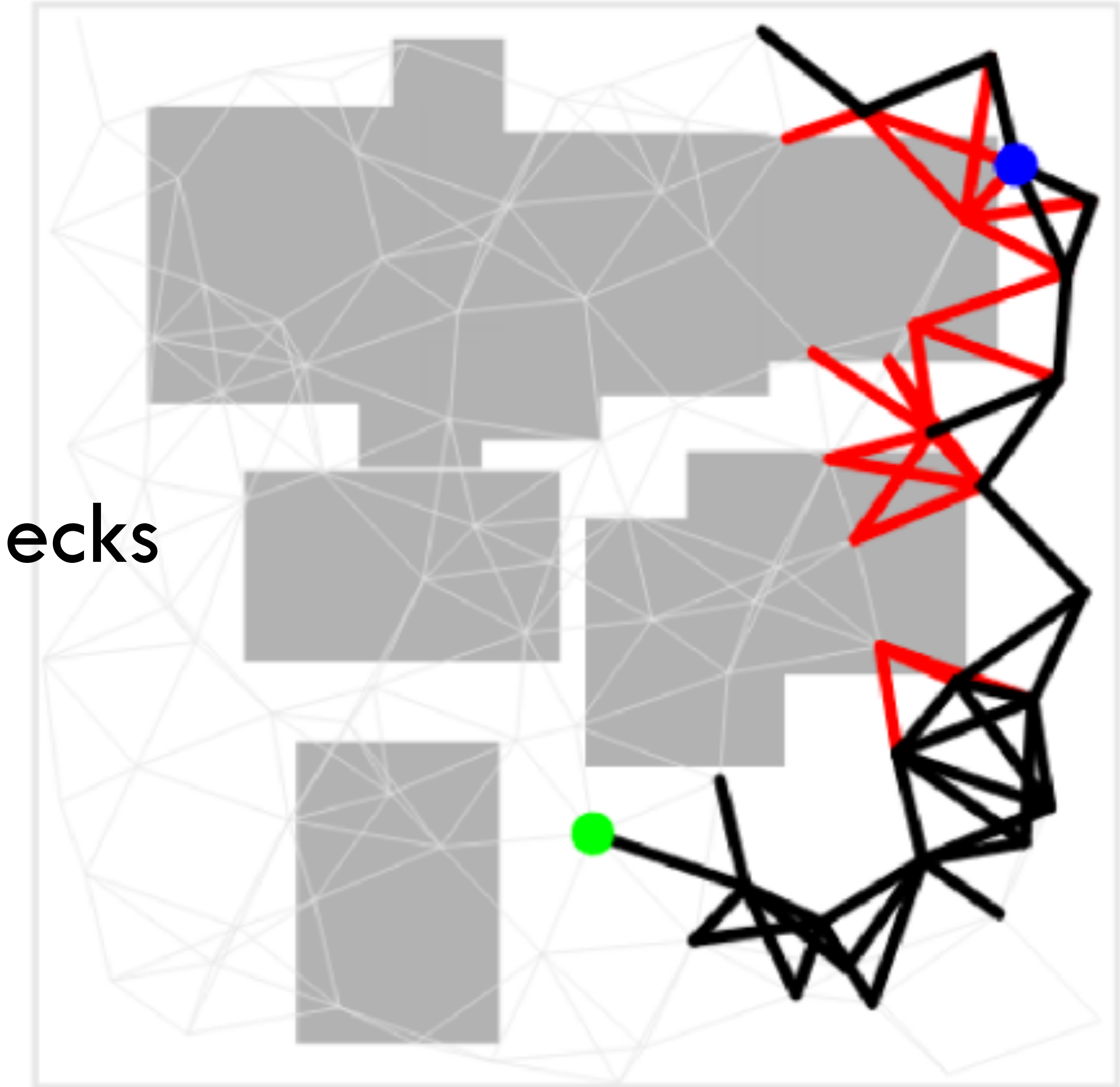
[Fig from Erion Plaku]

Return the current path as a solution

Lazy Motion Planning

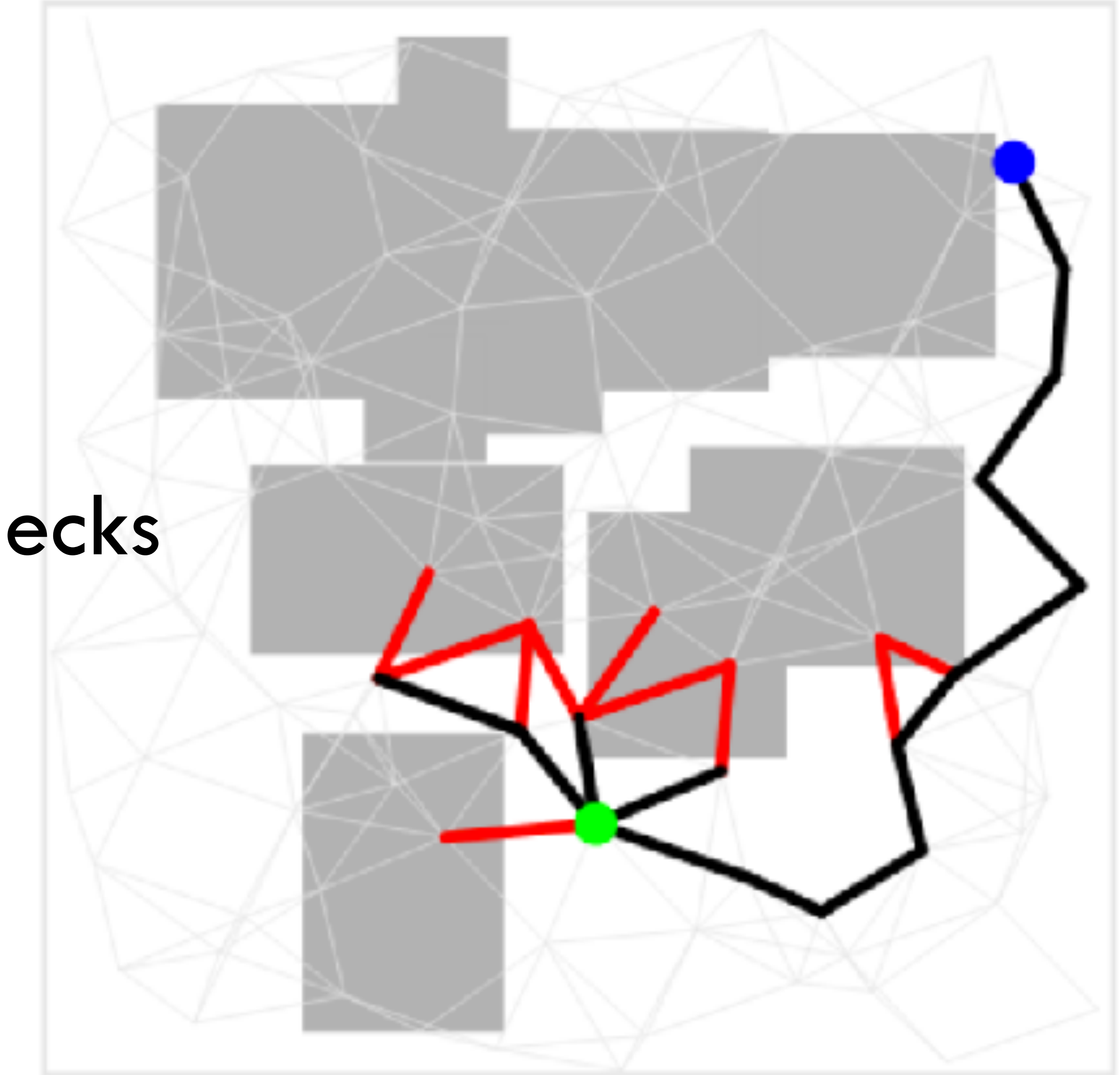
- **Defer** collision checking until a path is found
- **Remove** colliding edges path from the roadmap
- **Repeat** this process with a new path
- **Terminate** when a collision-fee path is found

77 checks

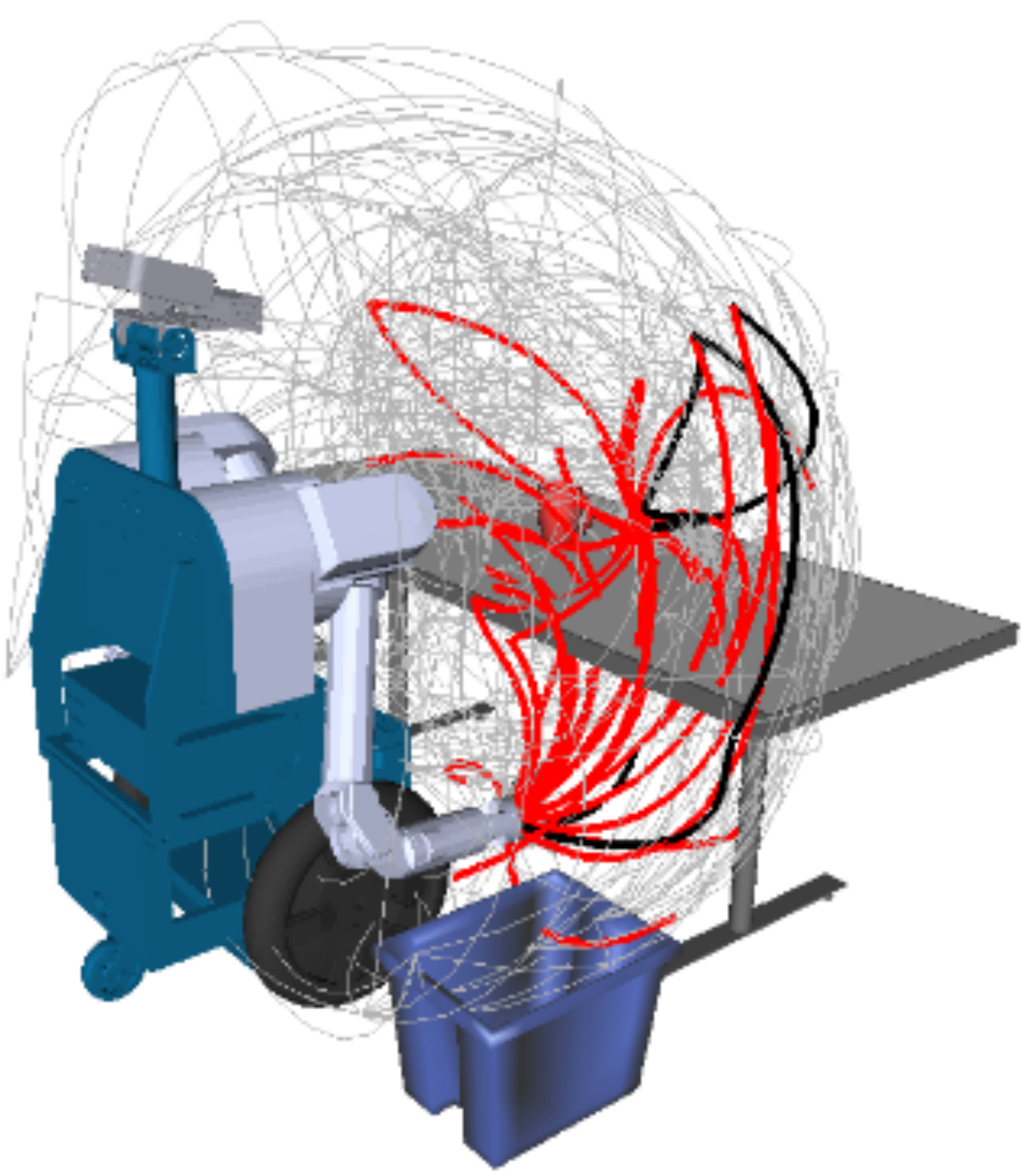


Eager (during search)

23 checks



Lazy



[Bohlin 2000][Dellin 2016]

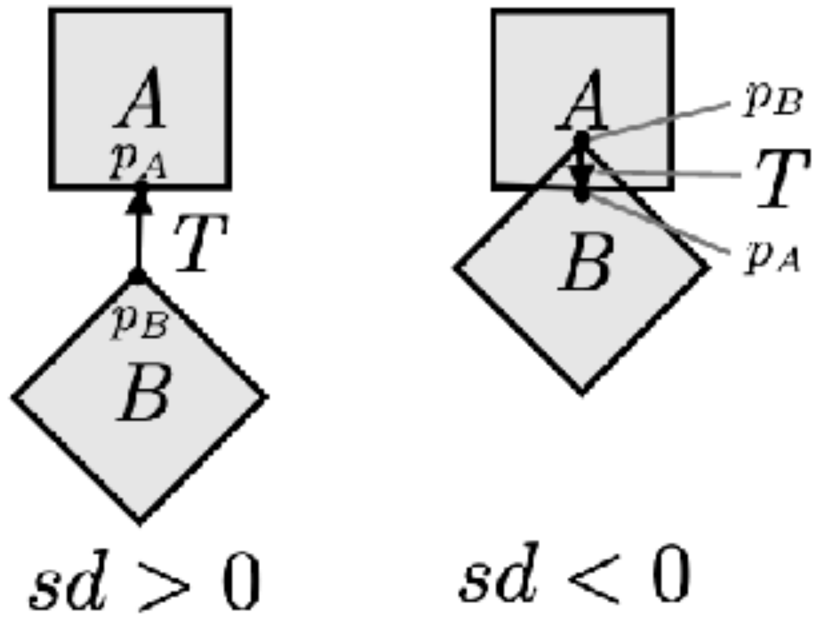
Trajectory Optimization

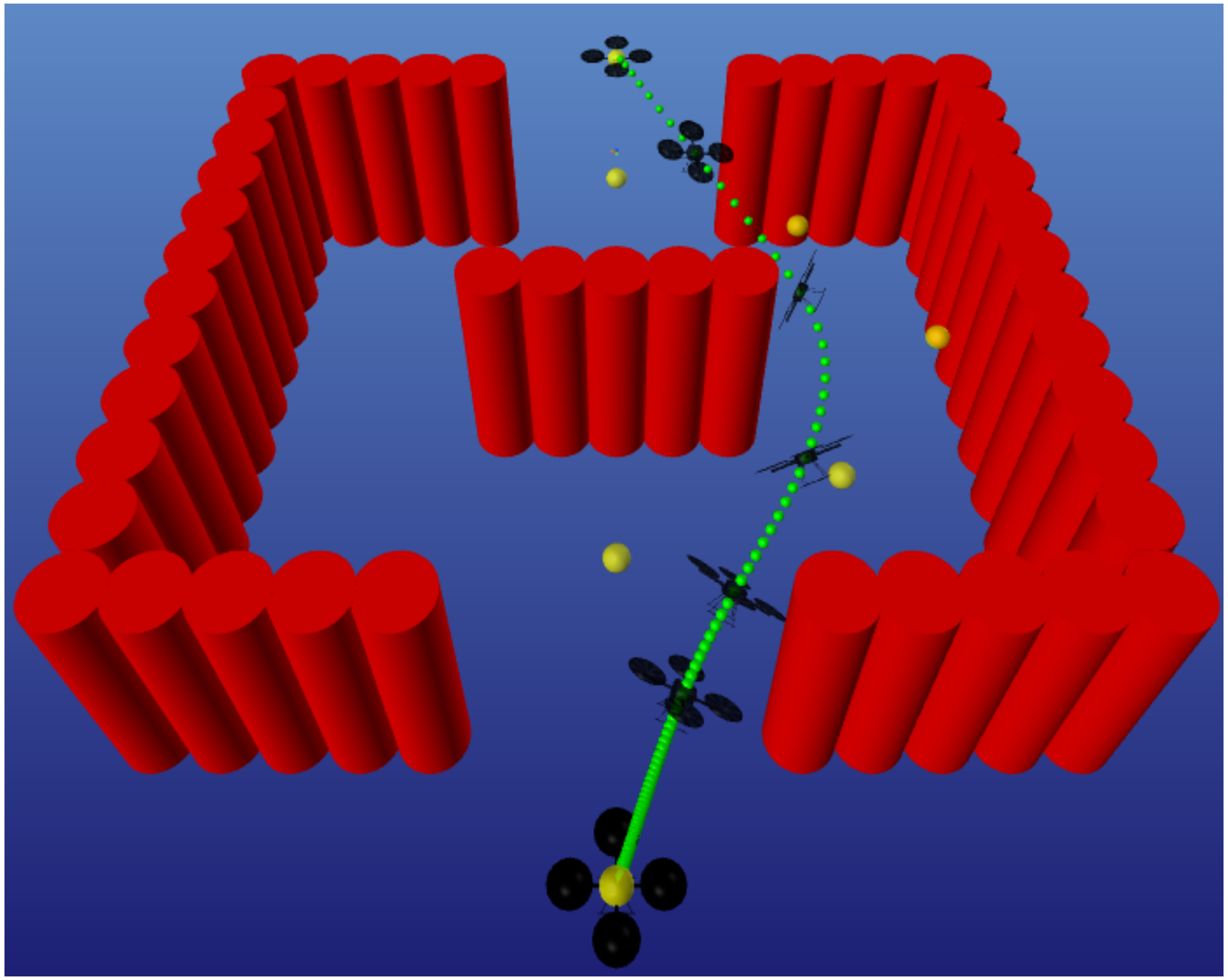
- Frame motion planning as a **non-convex constrained optimization** problem & converge to **local minima**

$$\begin{aligned}&\text{minimize } f(\mathbf{x})\\&\text{subject to}\\&\quad g_i(\mathbf{x}) \le 0, \quad i = 1, 2, \ldots, n_{ineq}\\&\quad h_i(\mathbf{x}) = 0, \quad i = 1, 2, \ldots, n_{eq}\end{aligned}$$

- Collision constraints enforced via **signed distance** (sd)

cuRobo
Local
TrajOpt





[Ratliff 2009][Schulman 2013][Sundaralingam 2022]

Task and Motion Planning (TAMP)

Shakey the Robot (1969)

- **First autonomous mobile manipulator** (via pushing)
  - Visibility graph, A* search, and STRIPS!
- **Decoupled** task and motion planning
  - Task planning **then** motion planning

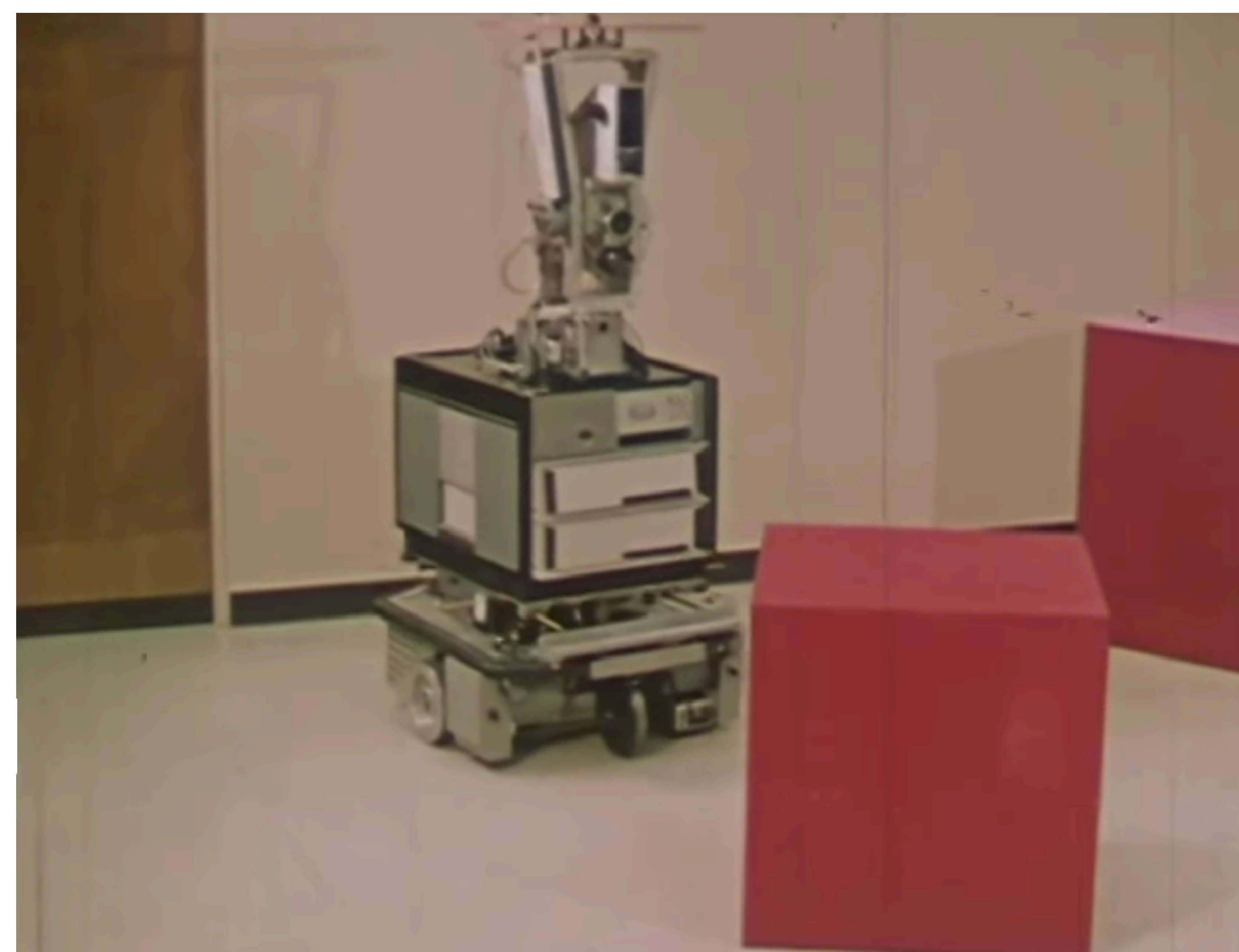
[Fikes 1971]

[Nilsson 1984]

```
type(robot robot)        type(o1 object)
name(robot shakey)       name(o1 box1)
at(robot 4.1 7.2)        at(o1 3.1 5.2)
theta(robot 90.1)        inroom(o1 r1)
                         shape(o1 wedge)
                         radius(o1 3.1)
```

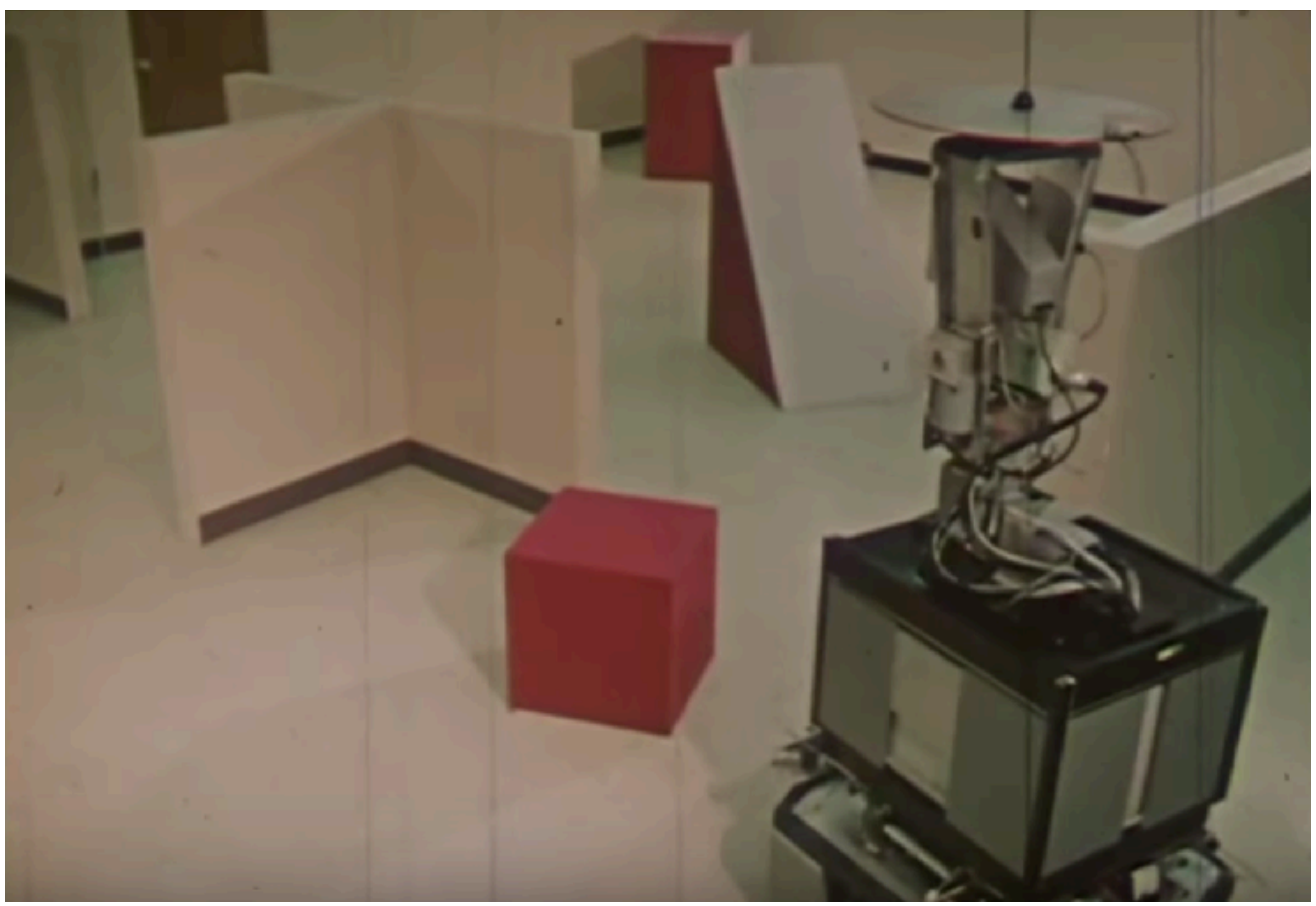
GOTHRU(d,r1,r2)

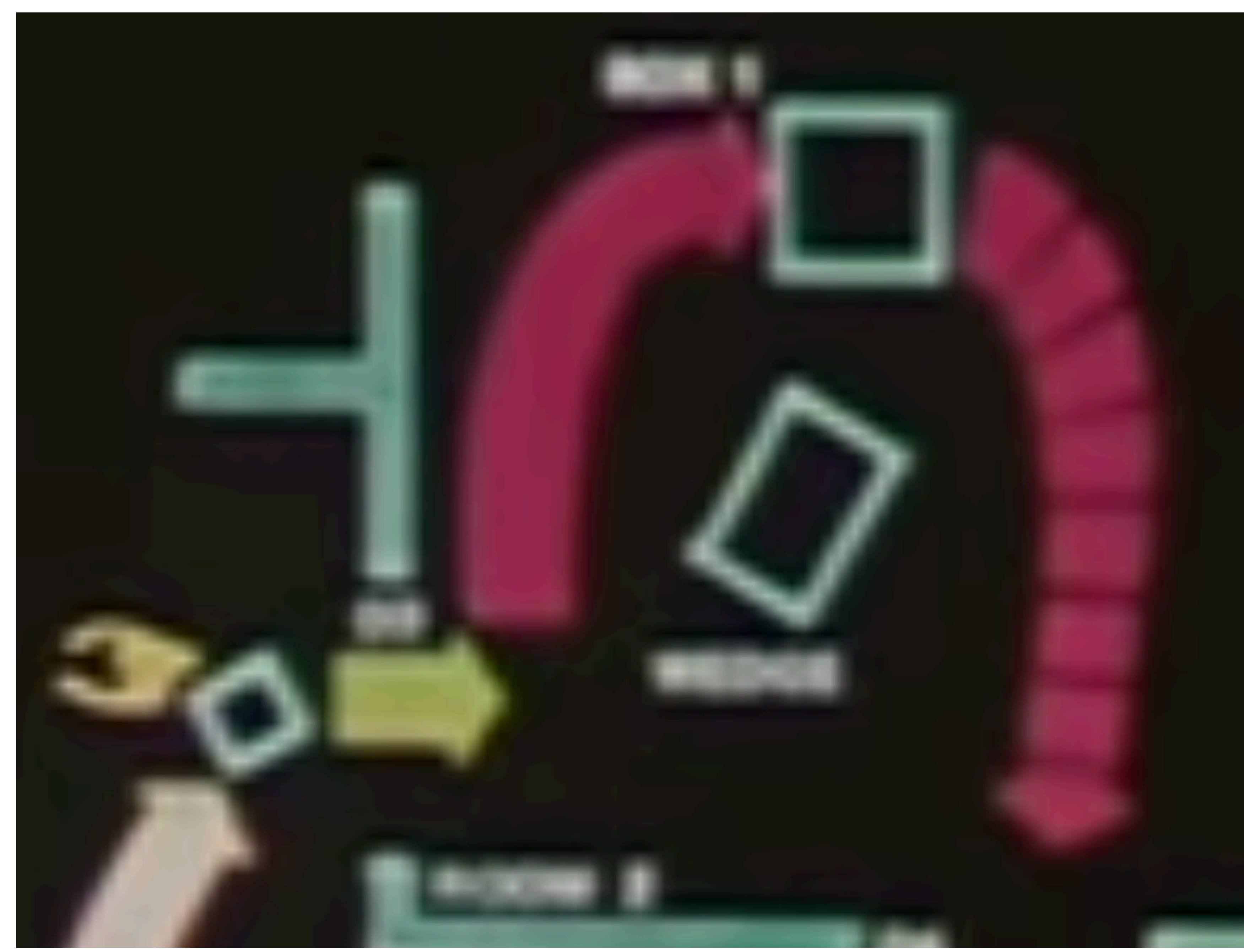
Precondition	INROOM(ROBOT,r1) ∧ CONNECTS(d,r1,r2)
Delete List	INROOM(ROBOT,$)
Add List	INROOM(ROBOT,r2)



Obstacle Blocks Shakey's Path

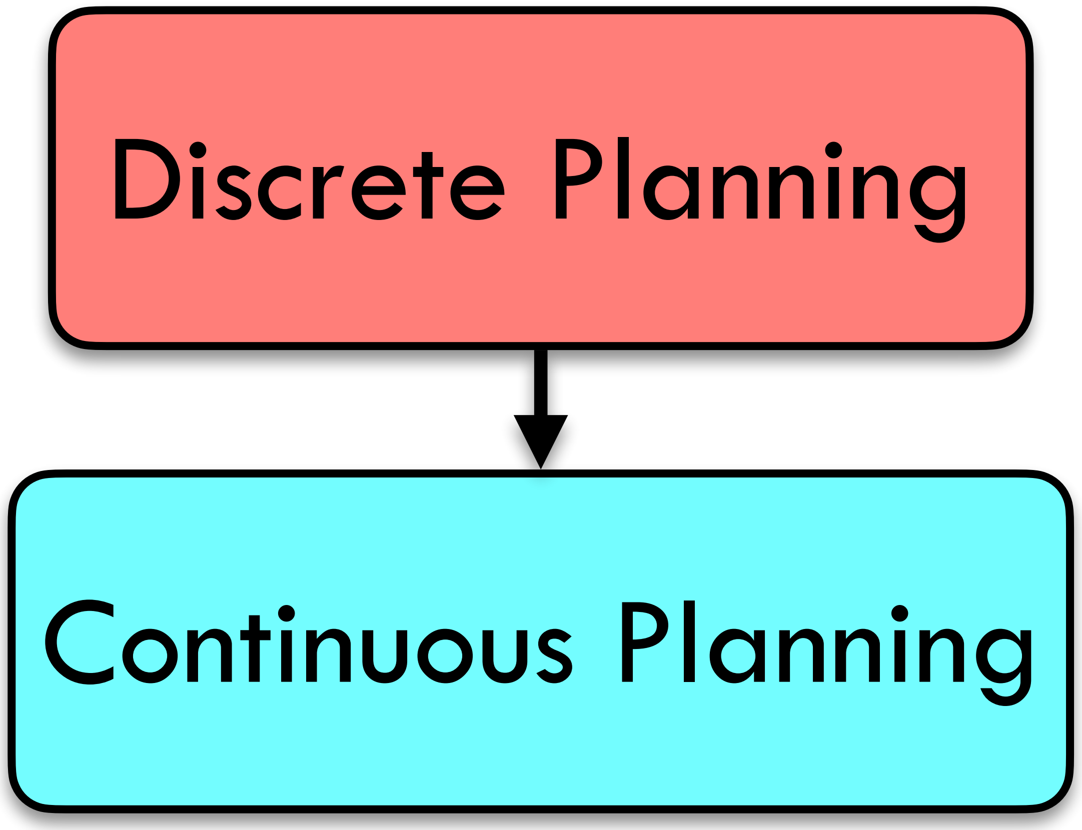
- What if a movable block **prevented** Shakey from safely moving into the adjacent room?
- Shakey could **push** it out of the way or **go around** it
  - What's more efficient? How to push it? …



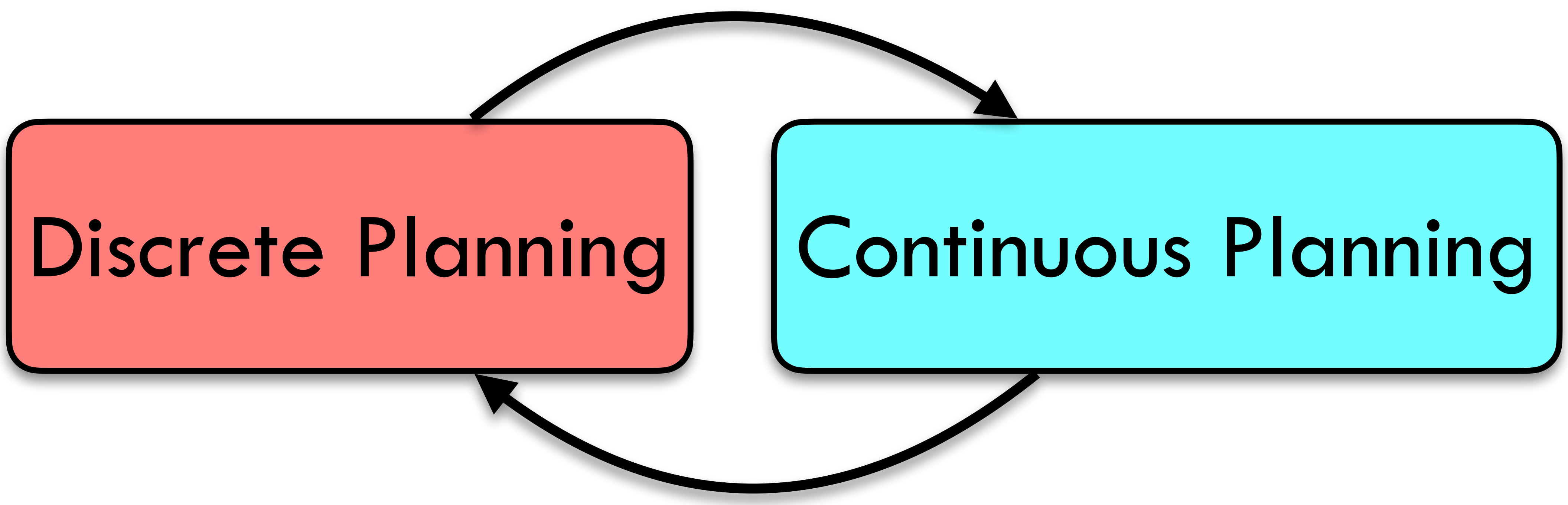


Decoupled vs Integrated TAMP

- **Decoupled:** discrete (task) planning **then** continuous (motion) planning
- Requires a strong **downward refinement** assumption
  - **<u>Every</u>** correct discrete plan can be **refined** into a correct continuous plan (from hierarchal planning)
- **Integrated:** **<u>simultaneous</u>** discrete & continuous planning



Decoupled



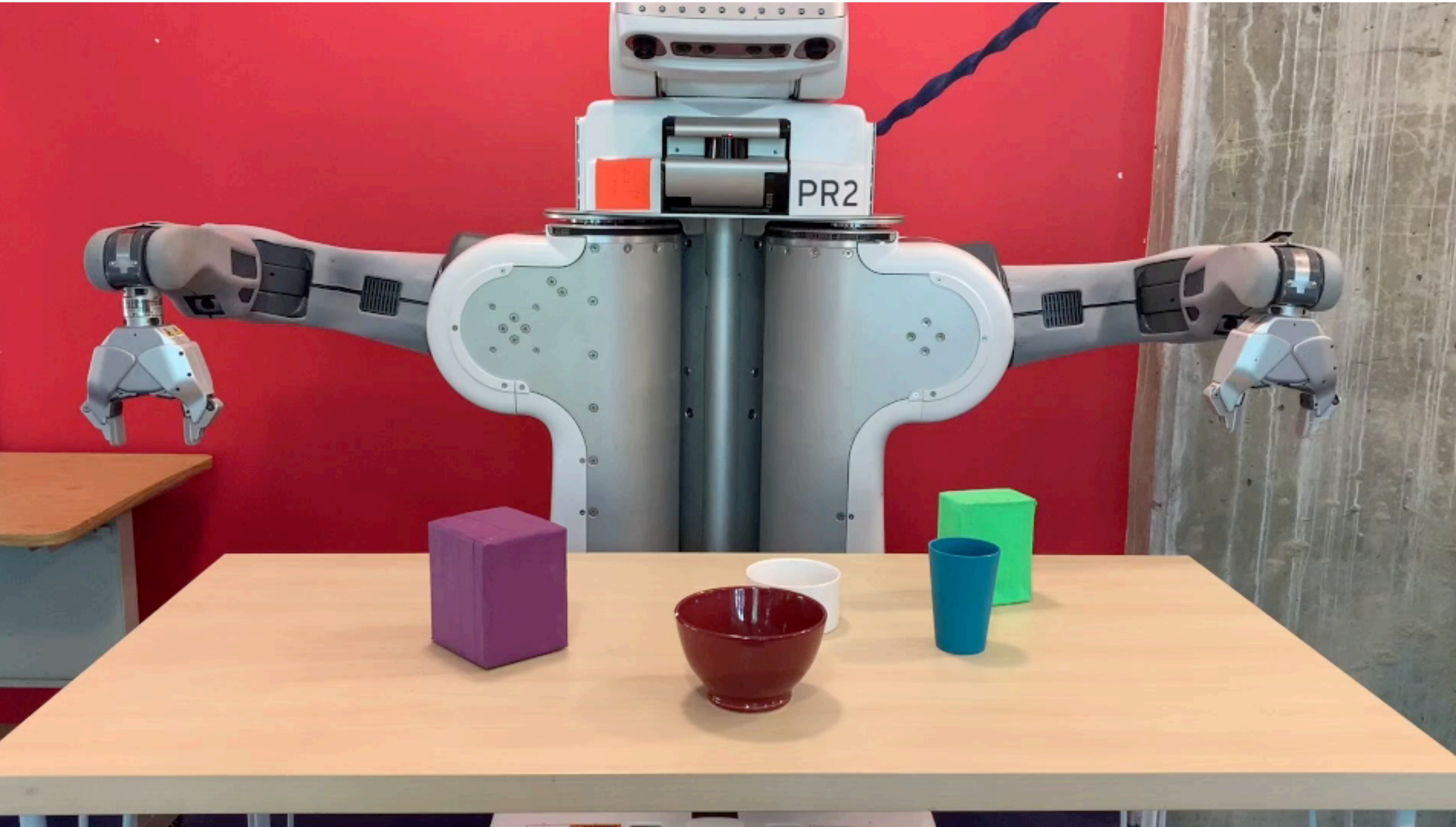
Integrated

Geometric Constraints Affect Plan

- **Inherits challenges** of both motion & classical planning
  - **High-dimensional, continuous** state-spaces
  - State-space **exponential** in number of variables
  - **Long horizons**

- Continuous constraints **limit high-level strategies**
  - Kinematics, reachability, joint limits, collisions grasp, visibility, stability, stiffness, torque limits, …

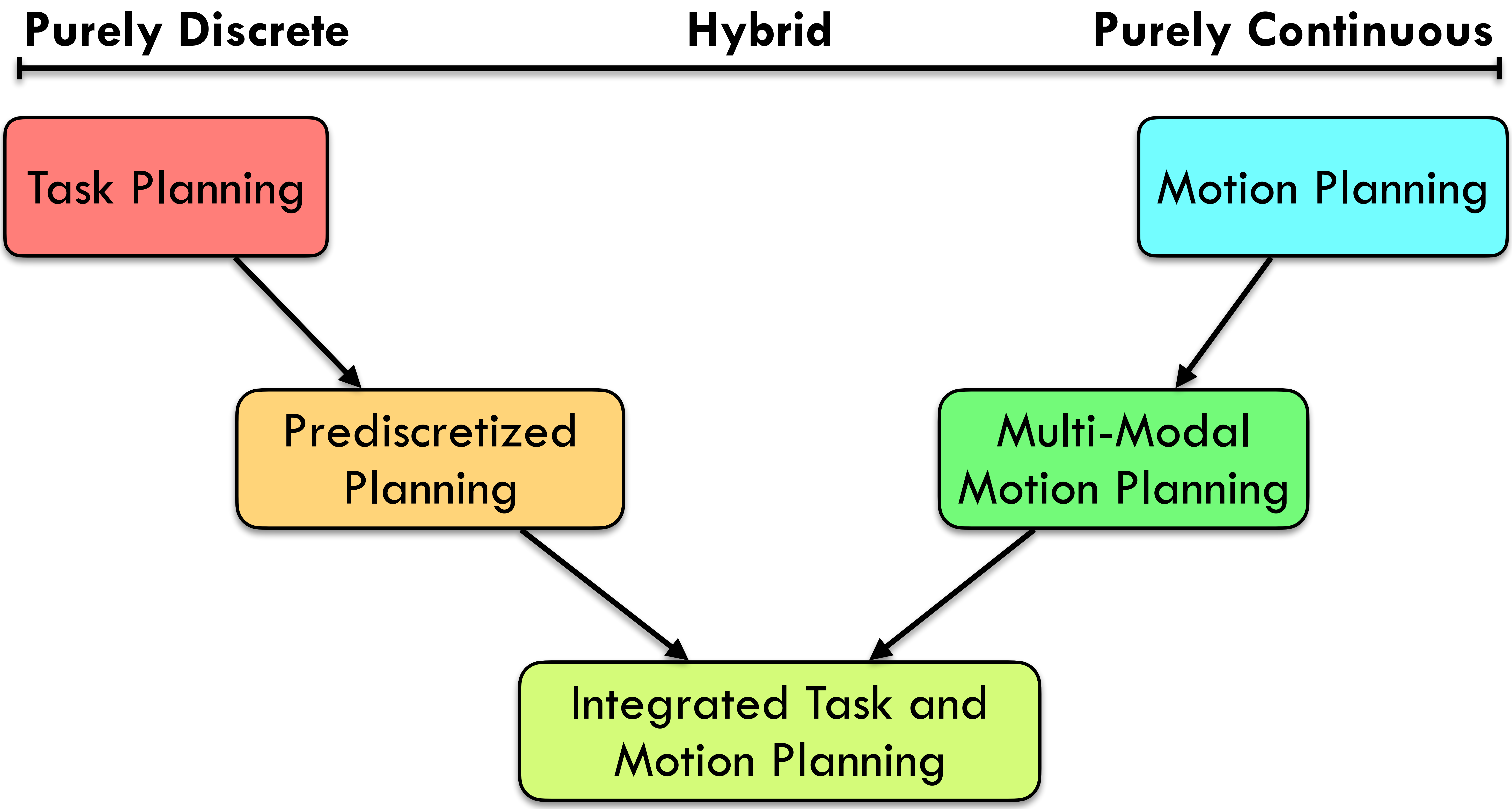
Pouring Among Obstacles



Block in Left Cabinet & Doors Closed

- Robot forced to **regrasp** the object
  - Change from a **top** grasp to a **side** grasp
- **Non-monotonic** problem
  - Plan must **undo goals** to solve
  - **Open** then **close** the cabinet door
- Physical constraints can be subtle!

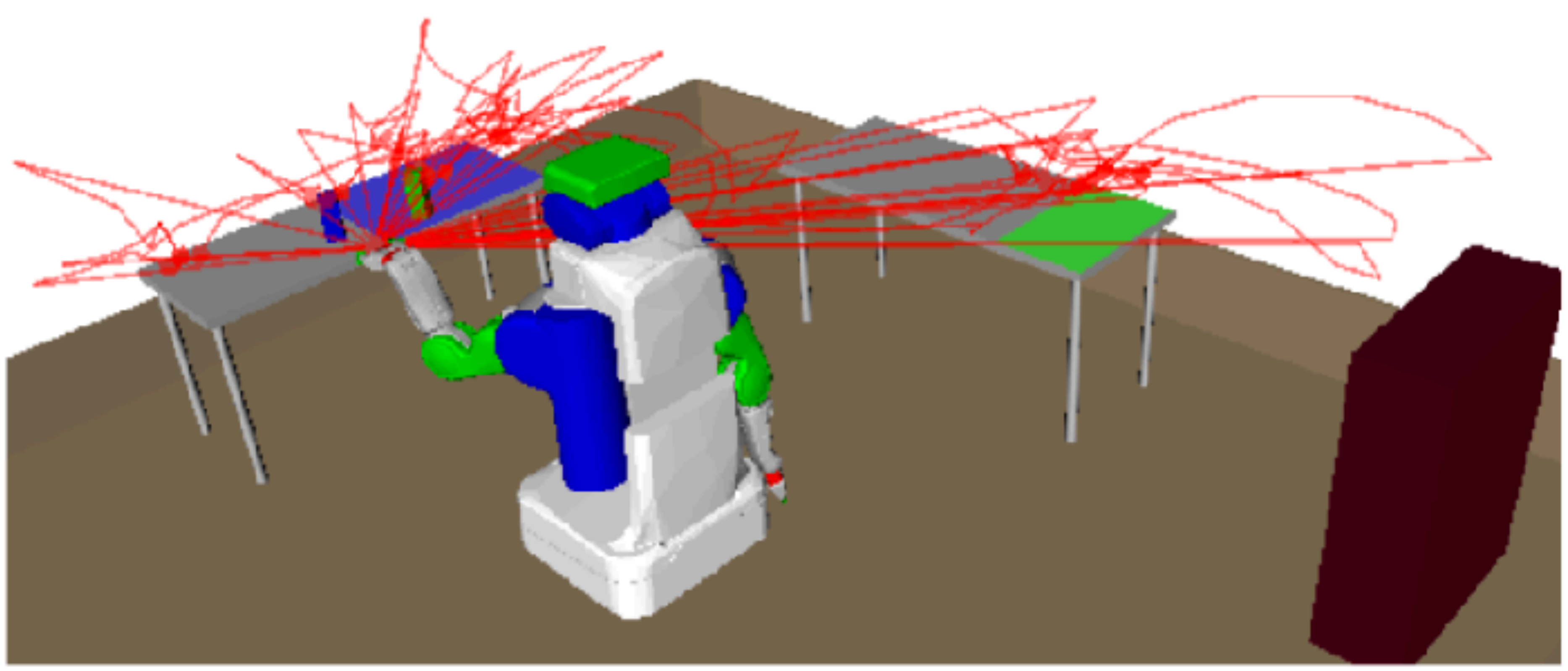
Hybrid Planning Spectrum

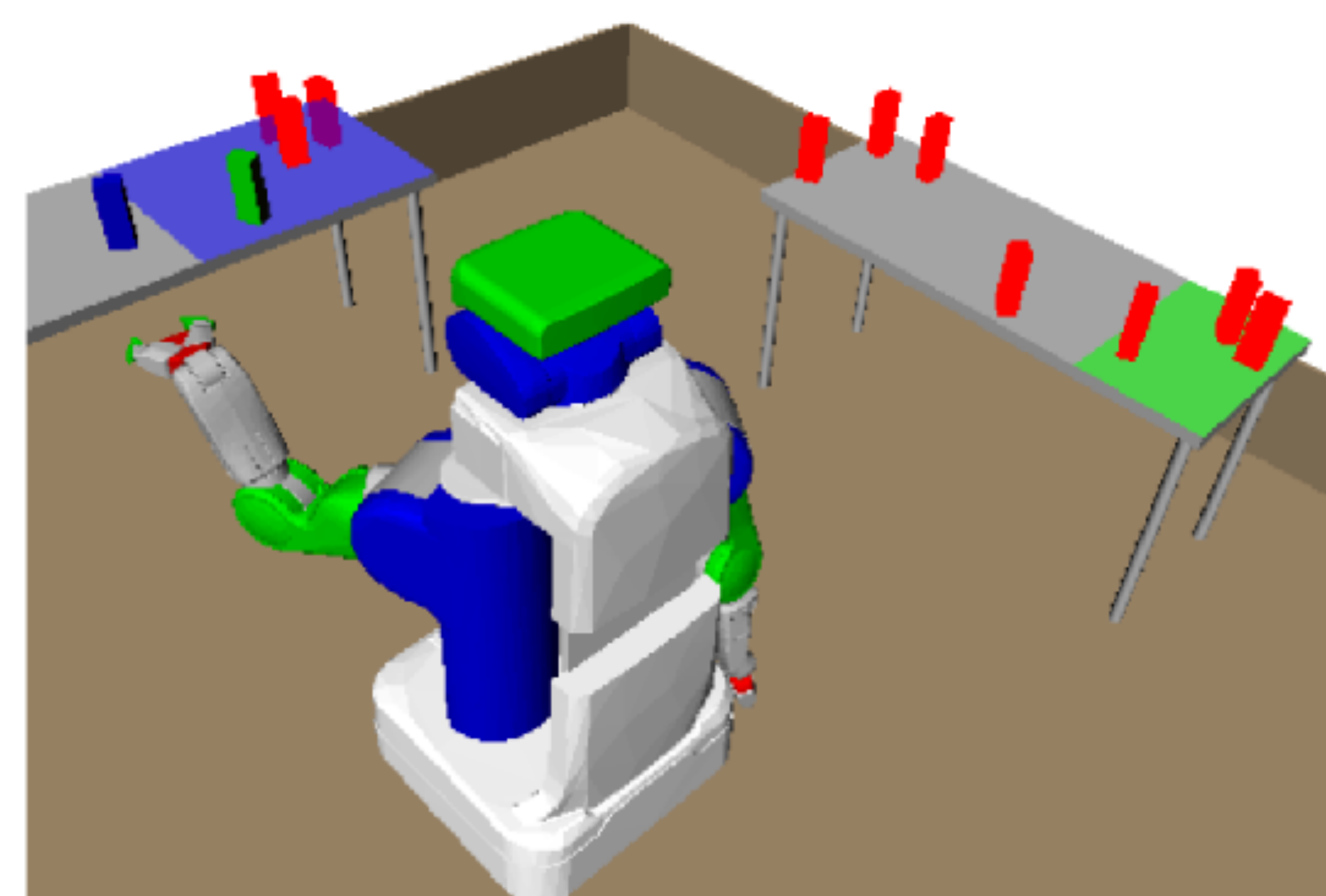


Prediscretized Planning

Prediscretized Continuous Variables

- Assume a **finite set** of object placements, object grasps, and (sometimes) robot configurations are **given**
- **Directly** perform discrete task planning
- Still need to evaluate **reachability**
  - Eagerly in **batch** [Lozano-Pérez 2014][Garrett 2017][Ferrer-Mestres 2017]
  - Eagerly during **search** [Dornhege 2009]
  - **Lazily** [Erdem 2011][Dantam 2018][Lo 2018]

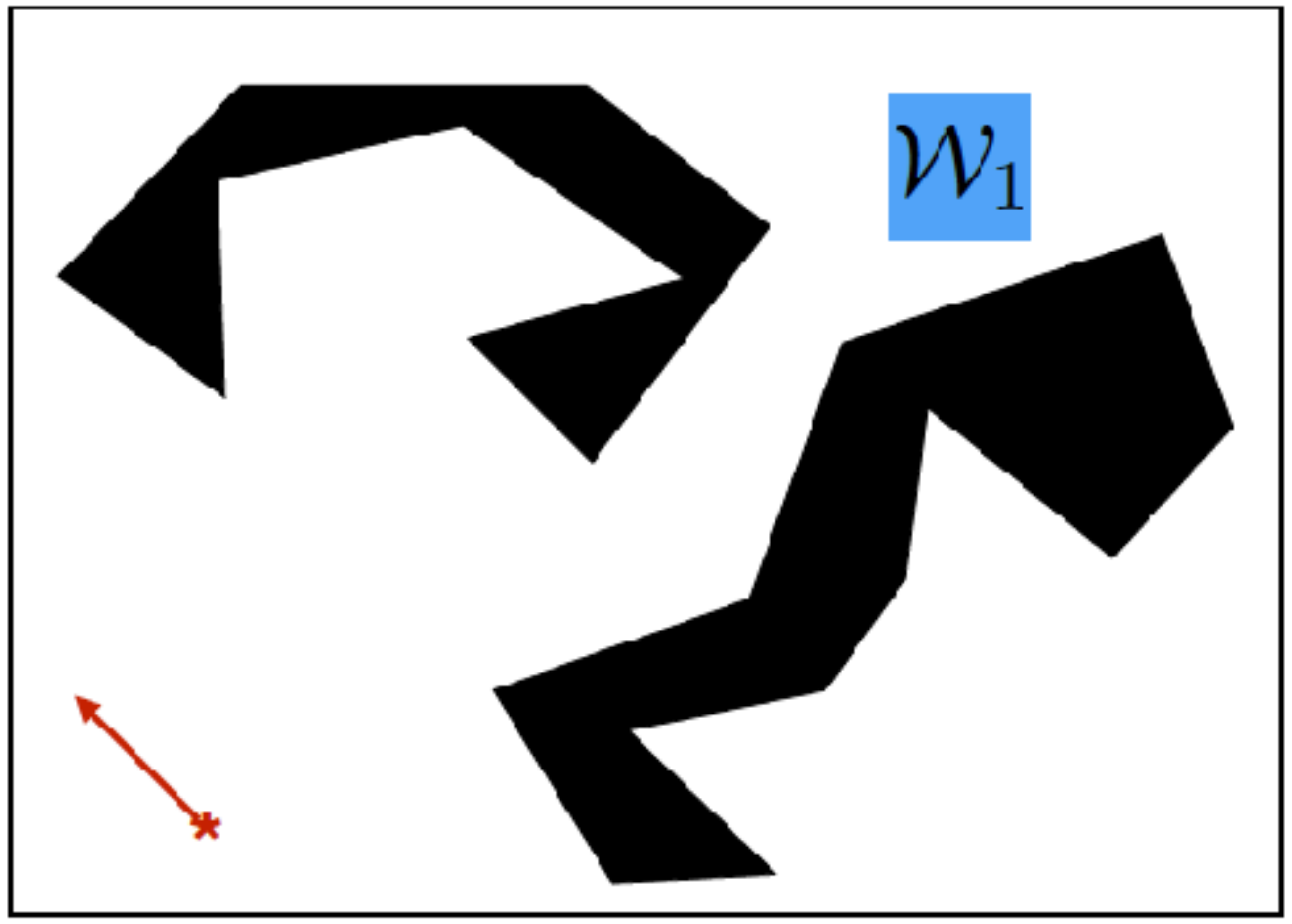


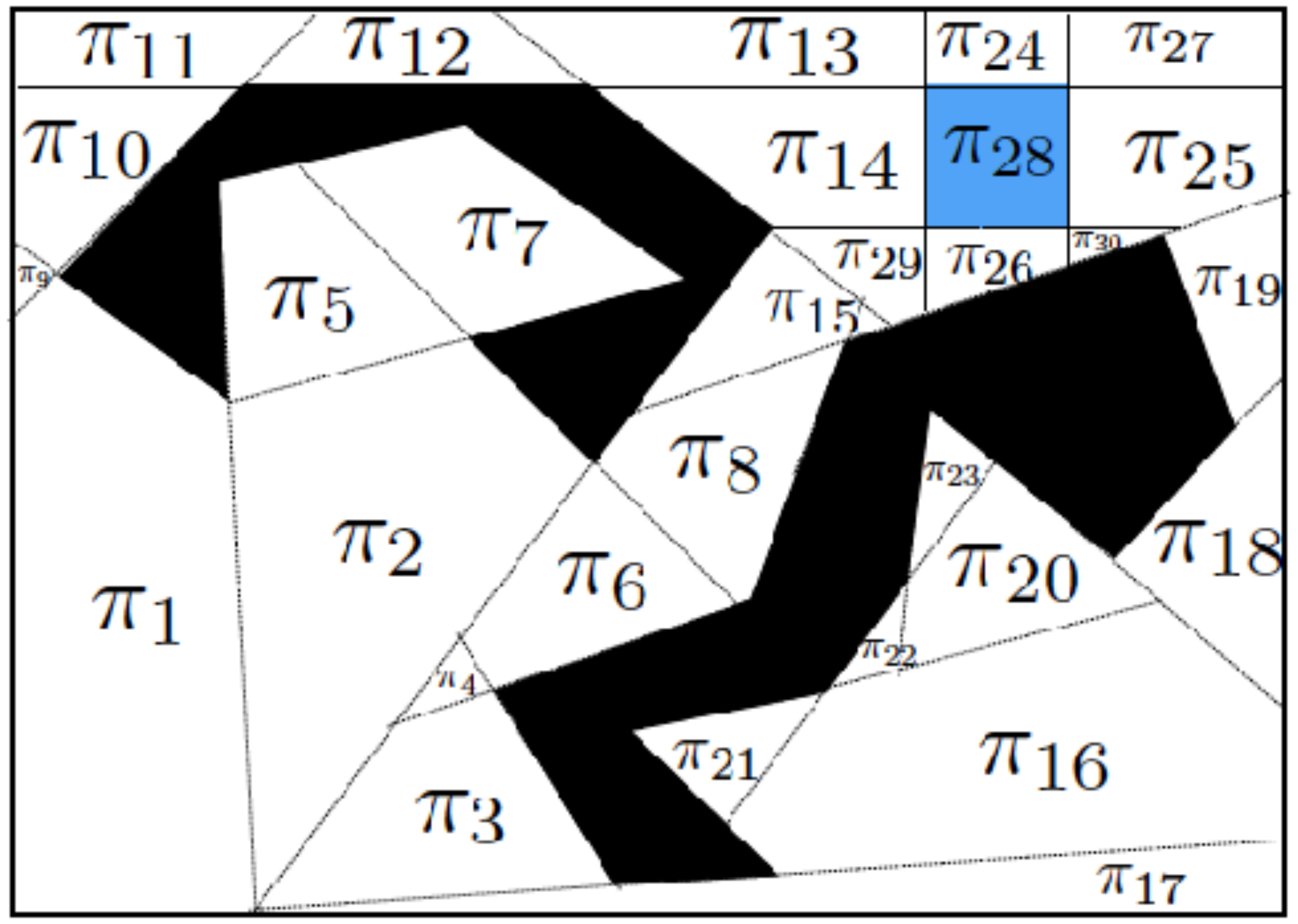


Prepartitioned Workspace

- Non-convexity handled by **partitioning** the workspace
- **Continuous control** parameters
- Tackle **convex dynamics** using **cone programming**

- **In contrast,** TAMP is often:
  - **High-dimensional**
  - **Non-convex**
    - 3D collision constraints
  - Less dynamically sophisticated



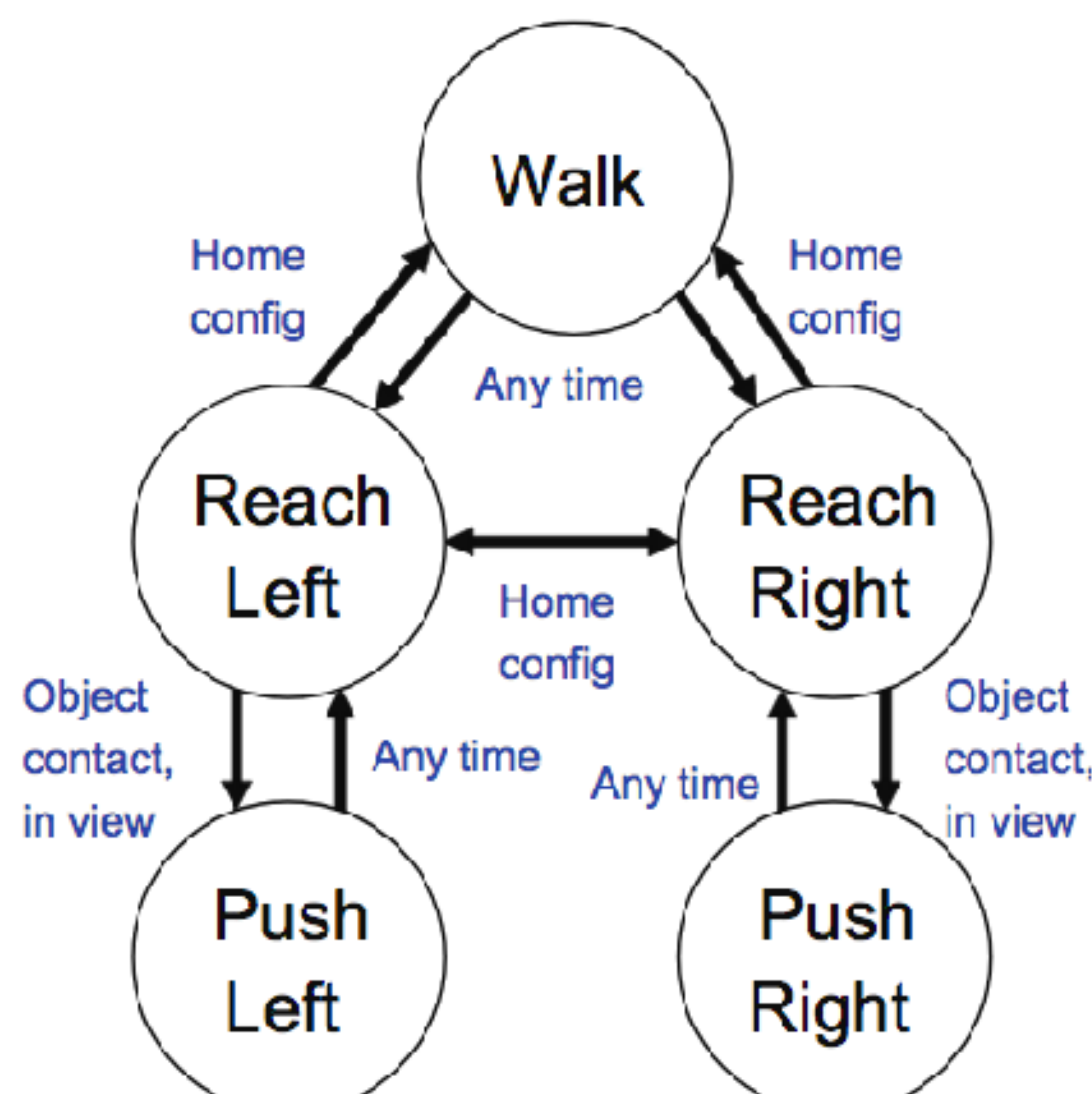


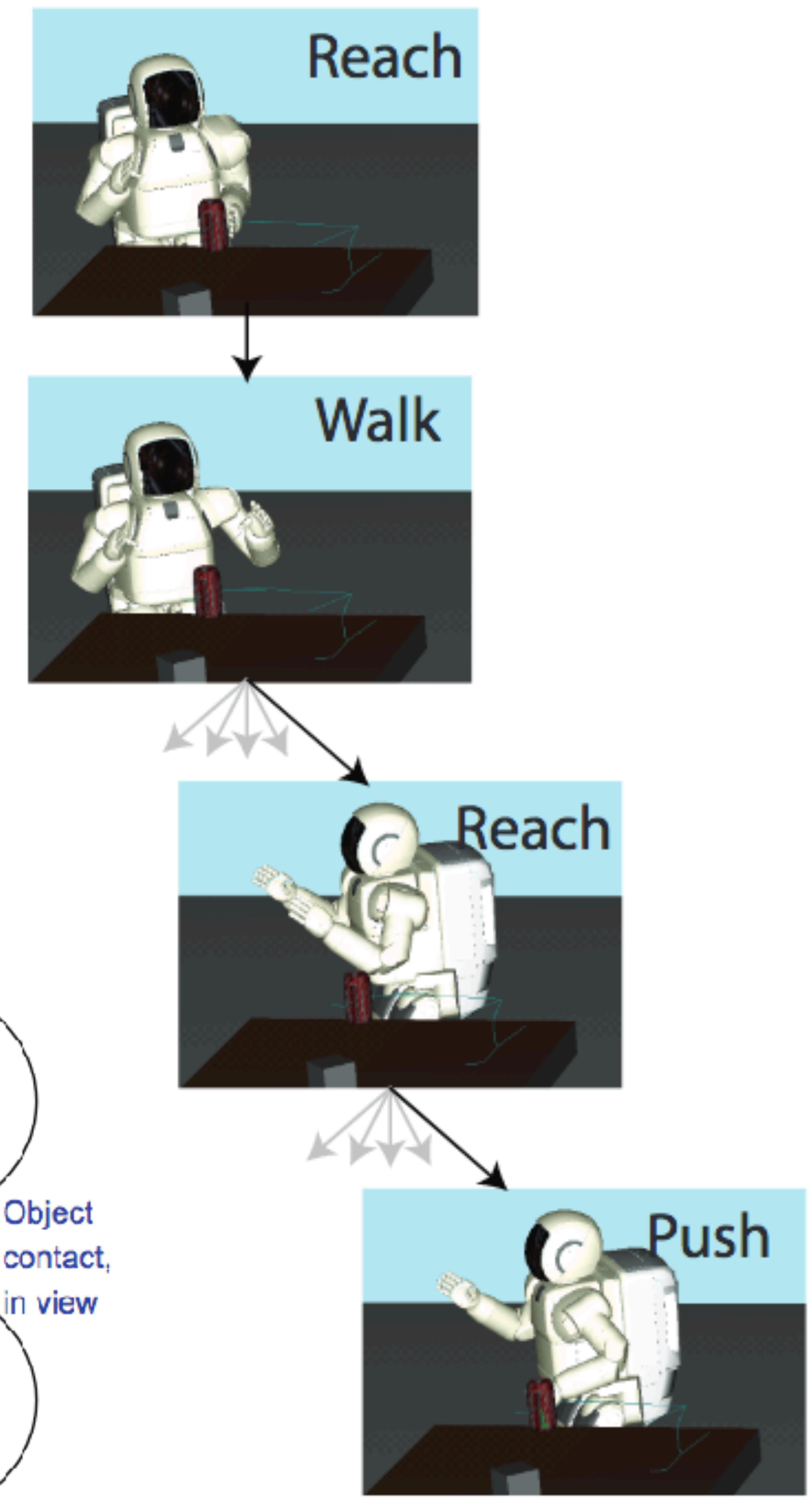
[Deits 2015][Shoukry 2016]
[Fernandez-Gonzalez 2018]

Multi-Modal Motion Planning

Multi-Modal Motion Planning

- Collision-free configuration space **changes** when objects are manipulated
- Use a **sequence** of motion planning problems each defined by a **mode**
- **Mode:** a set of motion constraints
  - Gripper is empty
  - Object pose remains **constant**

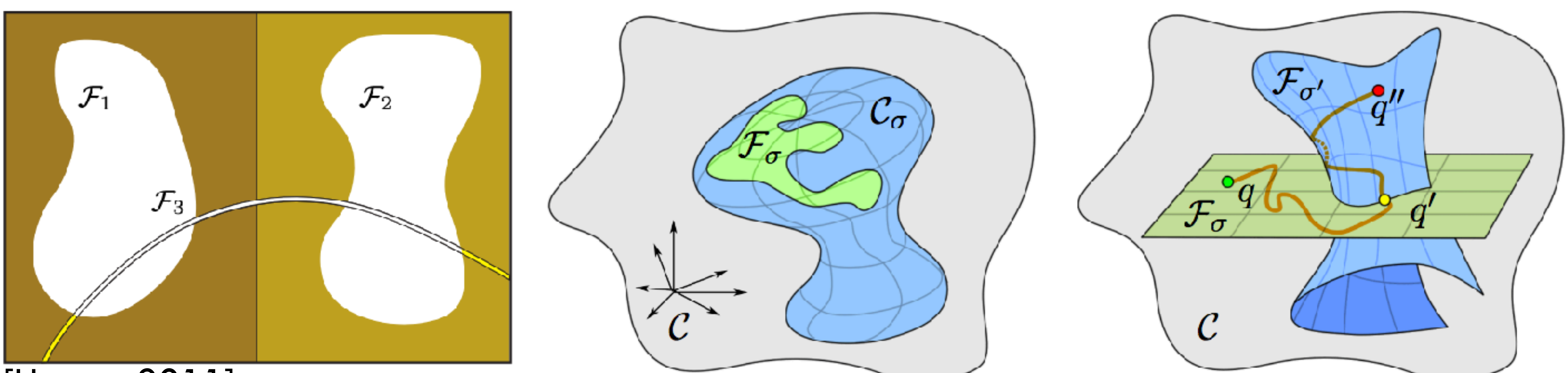




[Alami 1994][Siméon 2004][Hauser 2011]
[Barry 2013][Vega-Brown 2016]

Low-dimensional Intersections

- Need samples that **connect** adjacent modes
- Intersection of two modes is often **low-dimensional**
  - **Special-purpose** samplers are needed
- **Example:** transition from gripper **empty** to **holding**
- Configurations at the **intersection** obtained using **inverse kinematics** (IK)

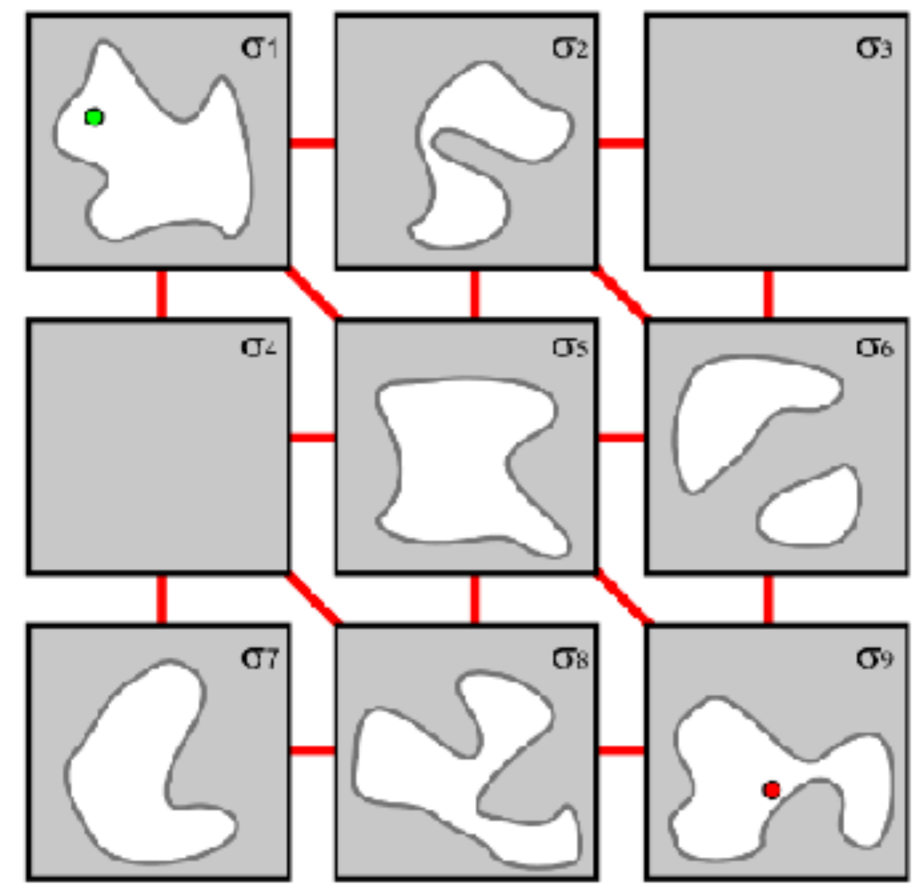


[Hauser 2011]

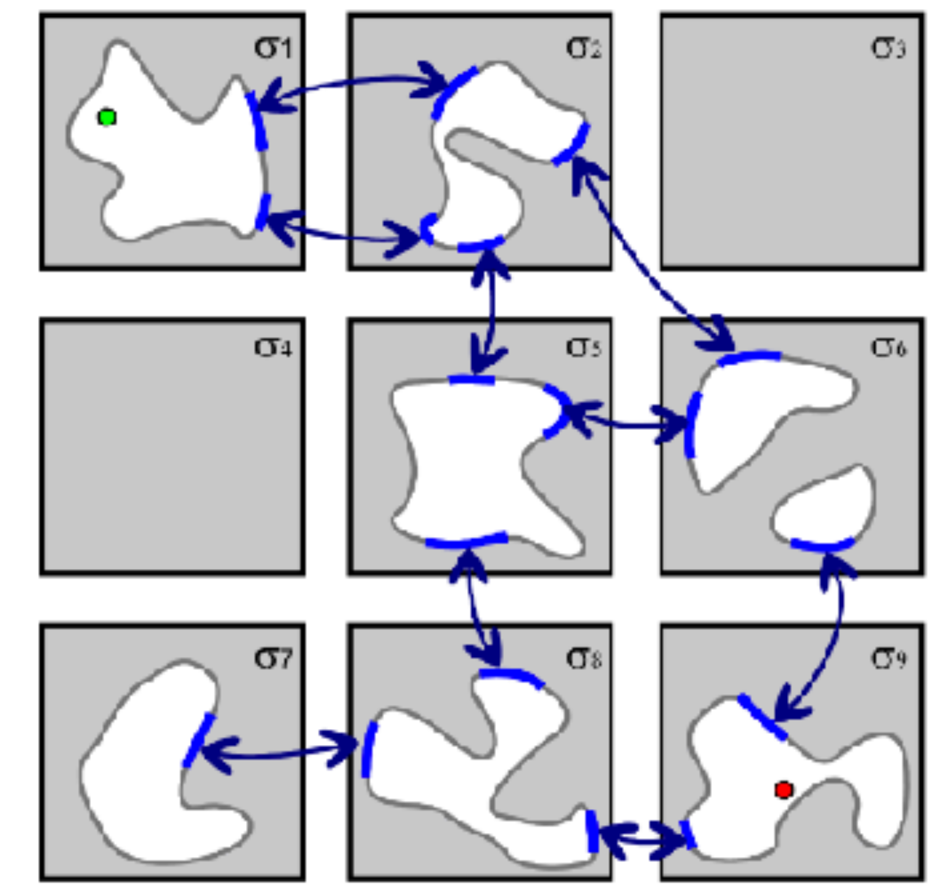
Sampling-Based Multi-Modal Planning

1. Sample from the set of **modes**
2. Sample at the **low-dimensional intersection** of adjacent modes
3. Sample a roadmap **within** each mode
4. Discrete search on the multi-modal **roadmap**

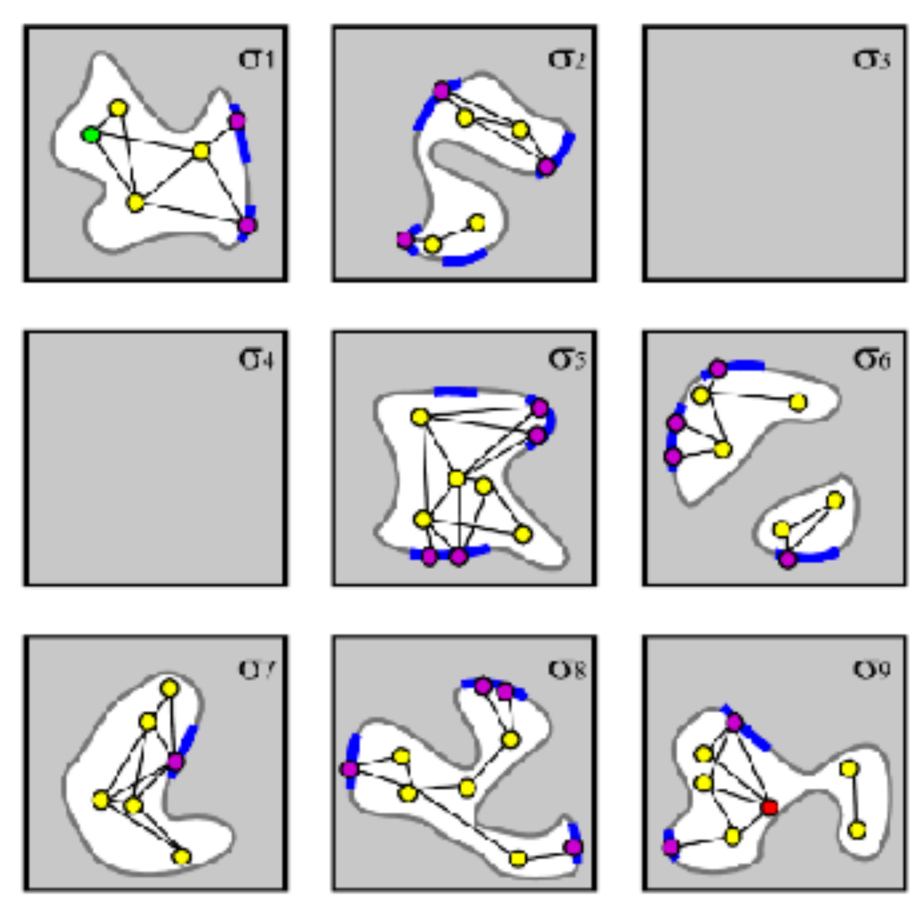
[Hauser 2011]



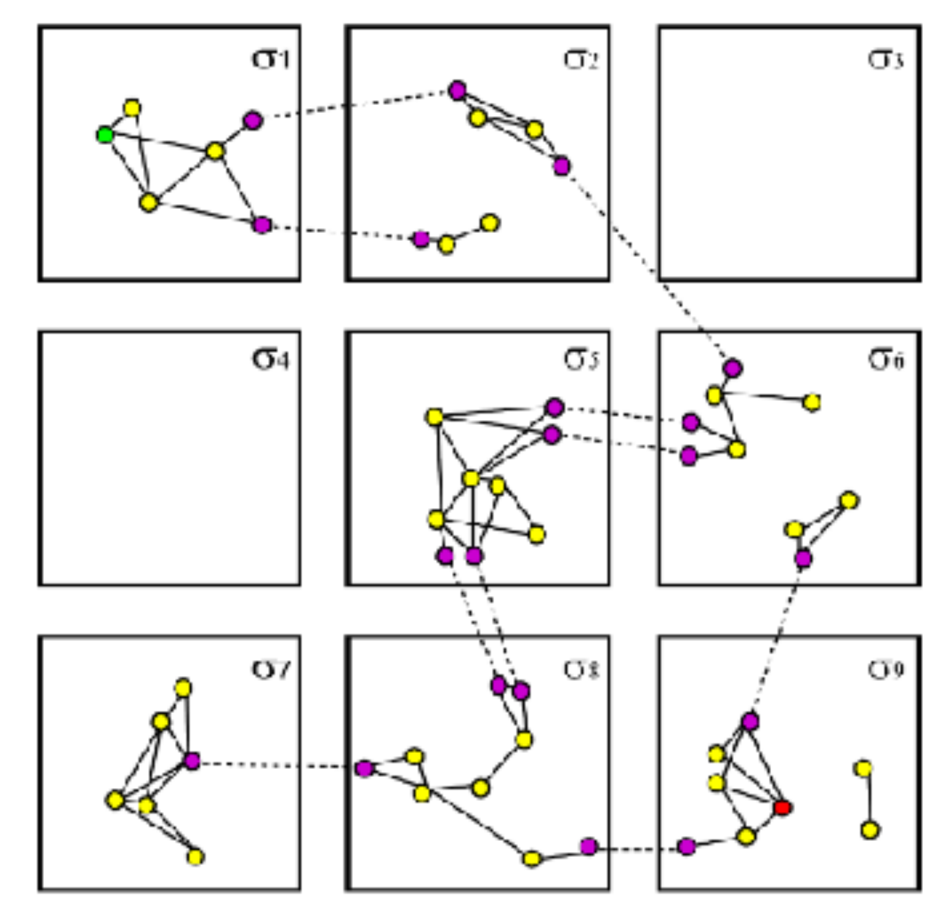
Adjacent modes



Intersections



Individual mode roadmaps

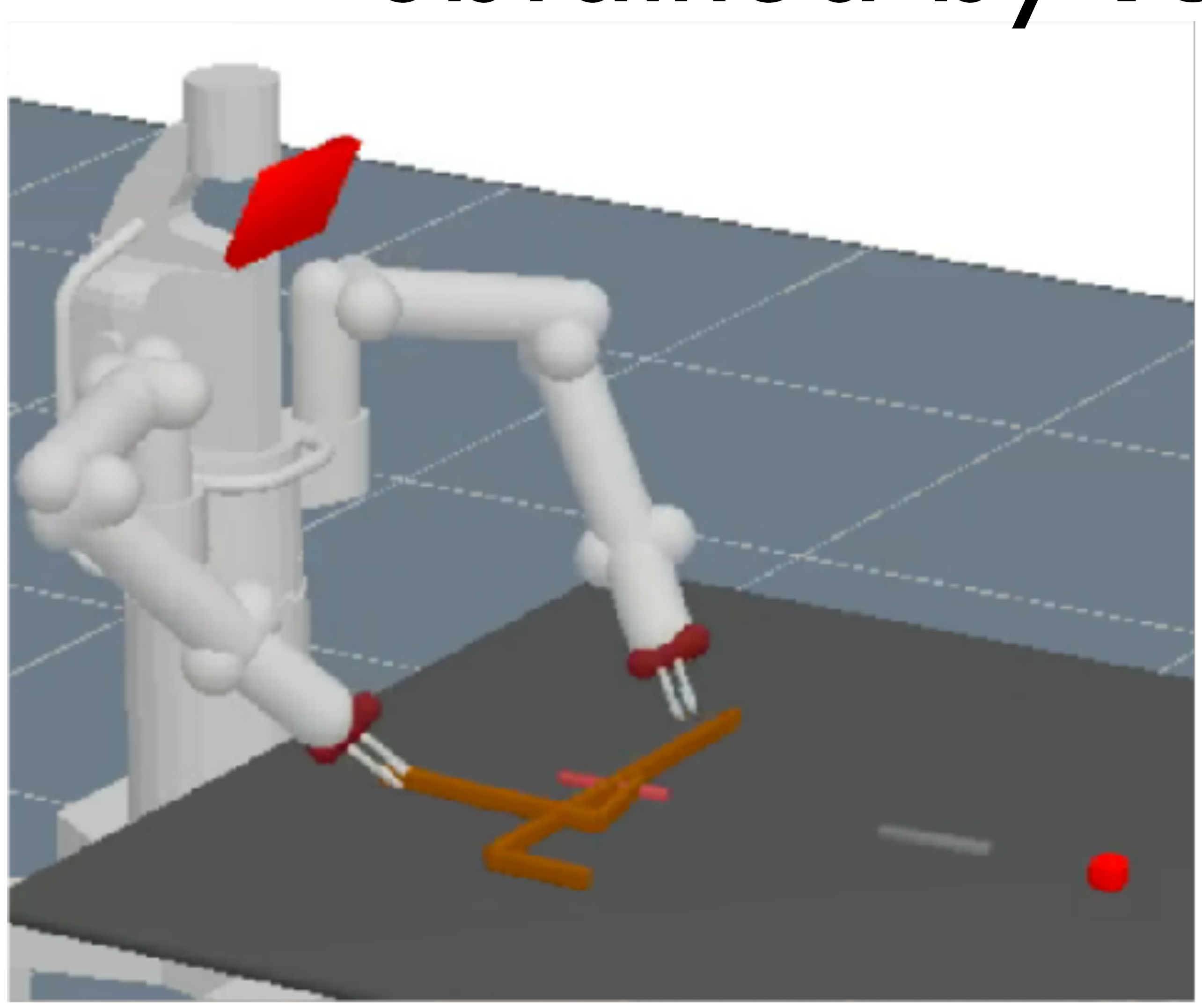


Combined Roadmap

Optimization-Based Multi-Modal Motion Planning

- Discrete search over sequences of **mode switches**
  - Sequences have **varying length**
  - Each sequence induces a **non-convex constrained optimization problem**
- Sequences can be pruned using **lower bounds** obtained by **relaxing** some constraints

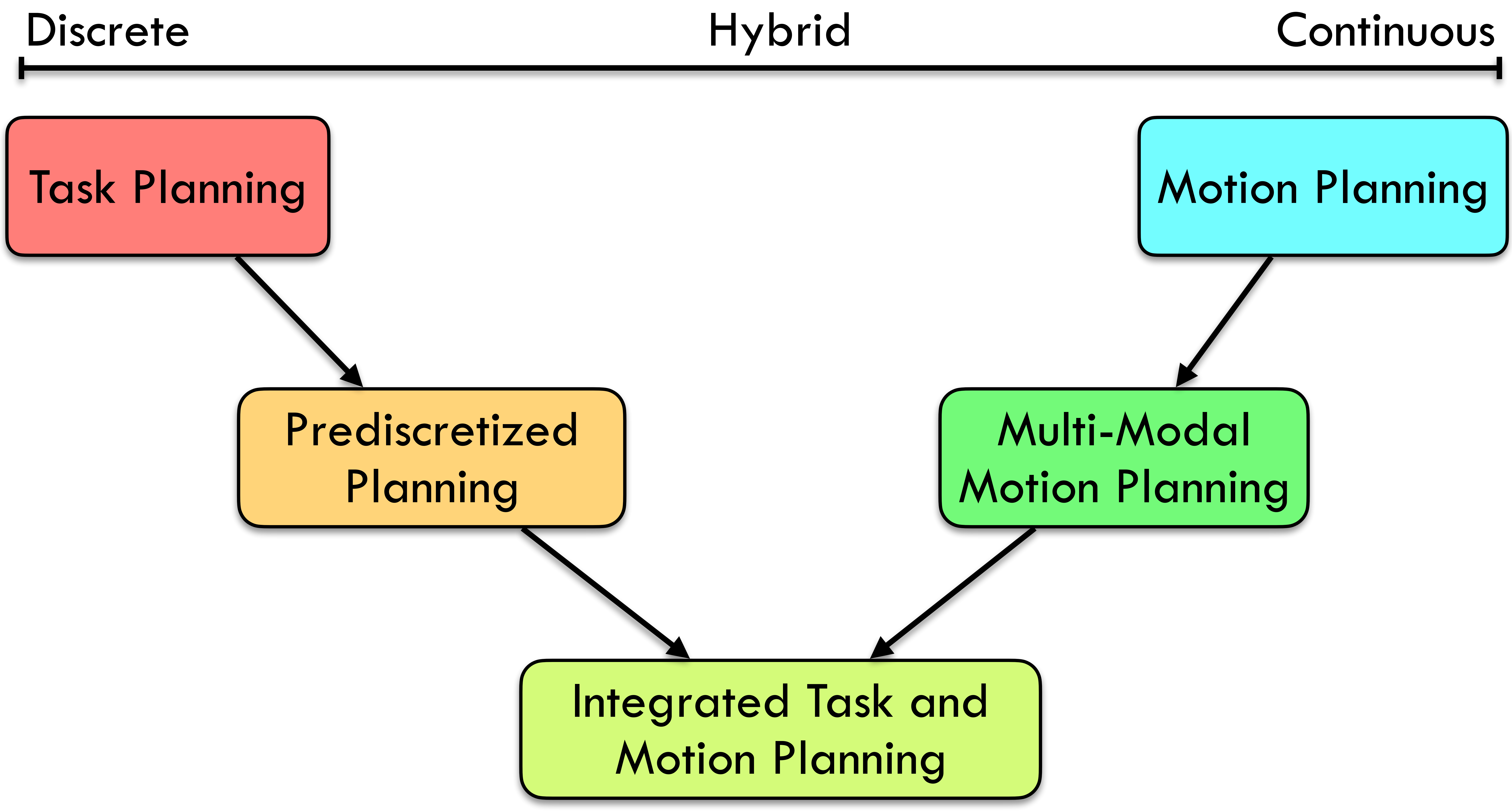
[Lagriffoul 2014]



[Toussaint 2015]
[Toussaint 2018]

$$\begin{aligned}
\min_{x, a_{1:K}, s_{1:K}} \quad & \int_0^T f_{\text{path}}(\bar{x}(t))\, dt + f_{\text{goal}}(x(T)) \\
\text{s.t.} \quad & x(0) = x_0,\ h_{\text{goal}}(x(T)) = 0,\ g_{\text{goal}}(x(T)) \le 0, \\
& \forall t \in [0, T]:\ h_{\text{path}}(\bar{x}(t), s_{k(t)}) = 0, \\
& \qquad\qquad\quad g_{\text{path}}(\bar{x}(t), s_{k(t)}) \le 0 \\
& \forall k \in \{1, .., K\}:\ h_{\text{switch}}(\hat{x}(t_k), a_k) = 0, \\
& \qquad\qquad\qquad g_{\text{switch}}(\hat{x}(t_k), a_k) \le 0, \\
& \qquad\qquad\qquad s_k \in \text{succ}(s_{k\text{-}1}, a_k)\ .
\end{aligned}$$

Hybrid Planning Spectrum Revisited



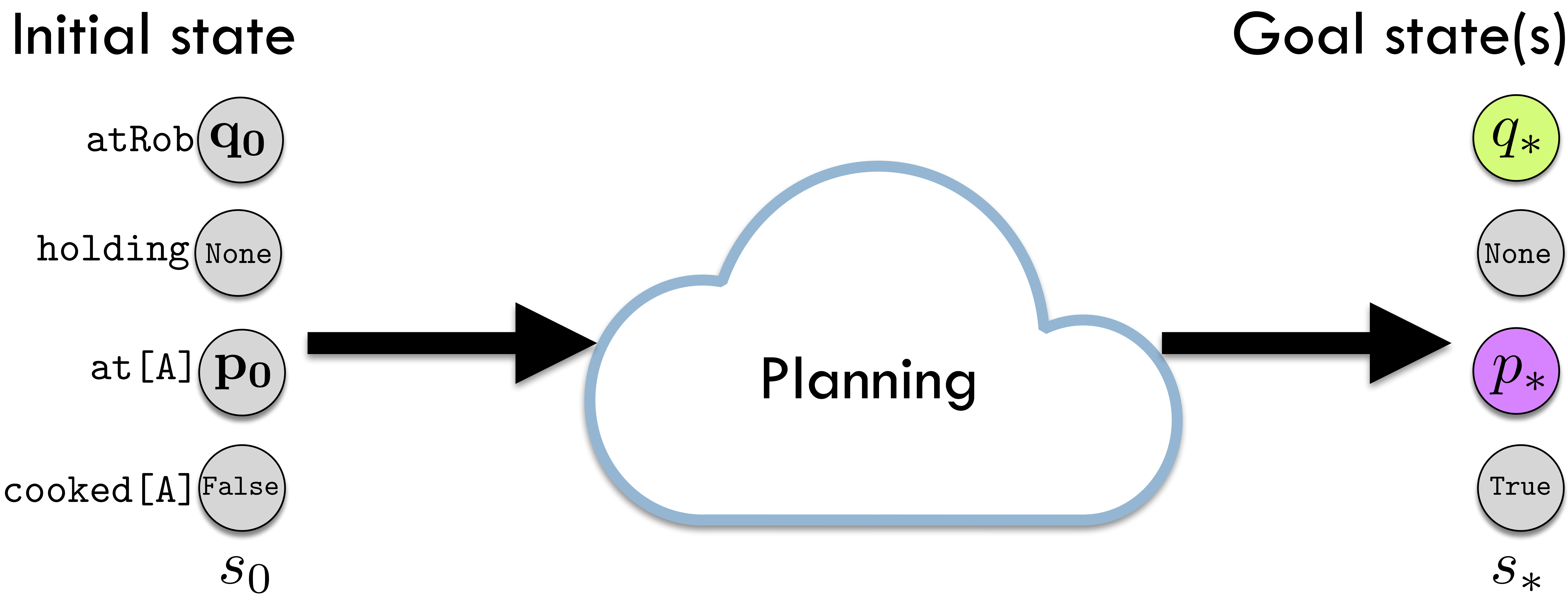
TAMP Example

Integrated Task and Motion Planning. Caelan Reed Garrett, Rohan Chitnis, Rachel Holladay, Beomjoon Kim, Tom Silver, Leslie Pack Kaelbling, Tomás Lozano-Pérez. *Annual Review of Control, Robotics, and Autonomous Systems*, 2021.

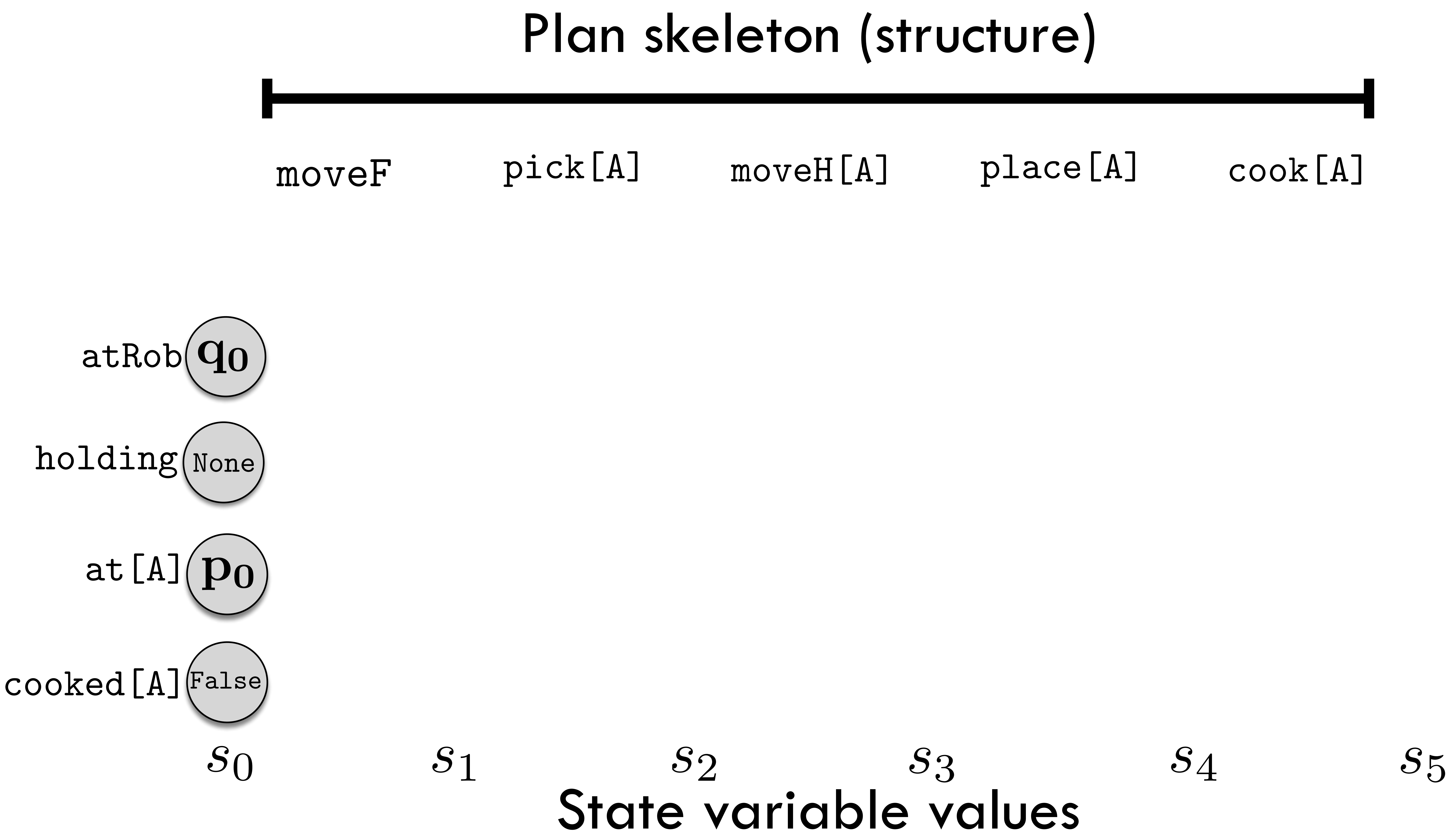
TAMP Example: Cook Object A

s_0 = {atRob=$\mathbf{q}_0$, at[A]=$\mathbf{p}_0$, holding=None, cooked[A]=False}

Goal conditions: cooked[A]=True

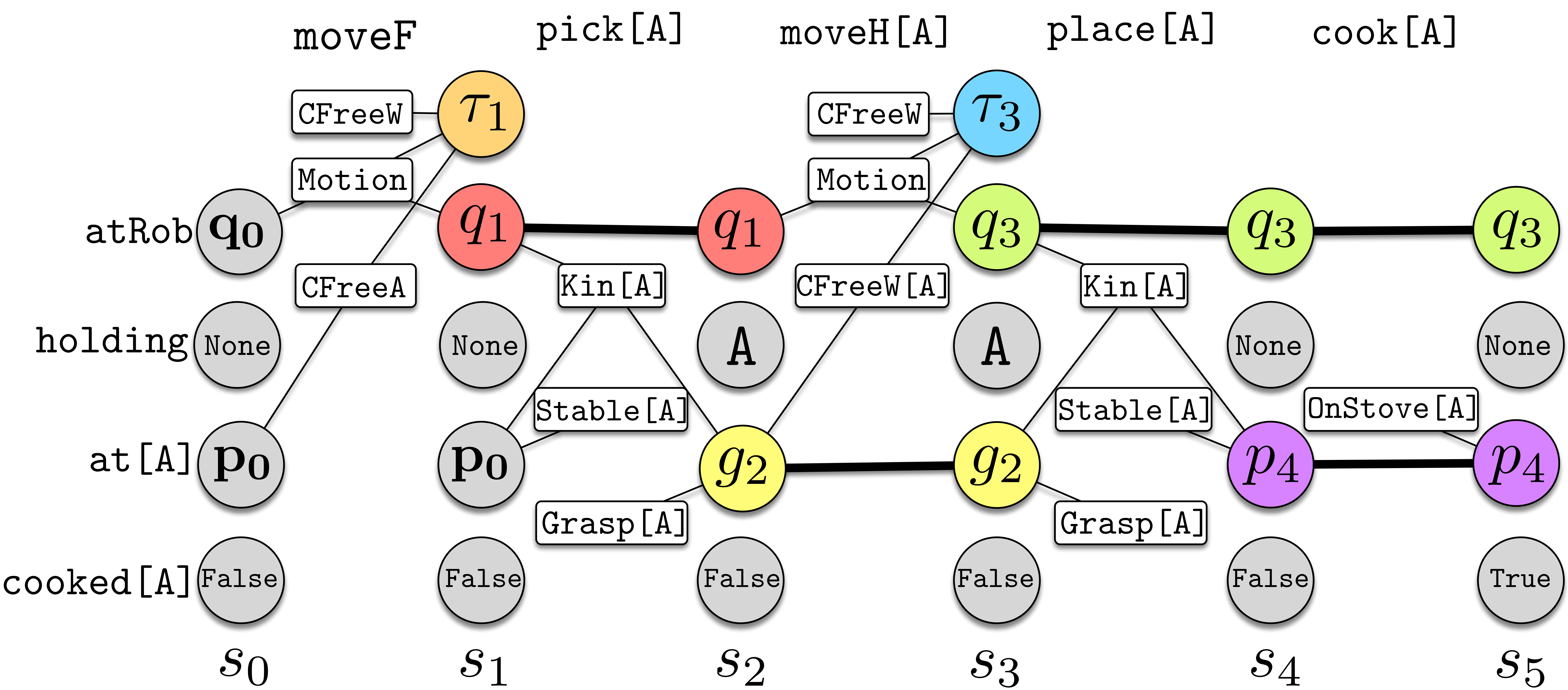


Plan Skeleton & Action Parameters



Plan Constraints & Parameter Values

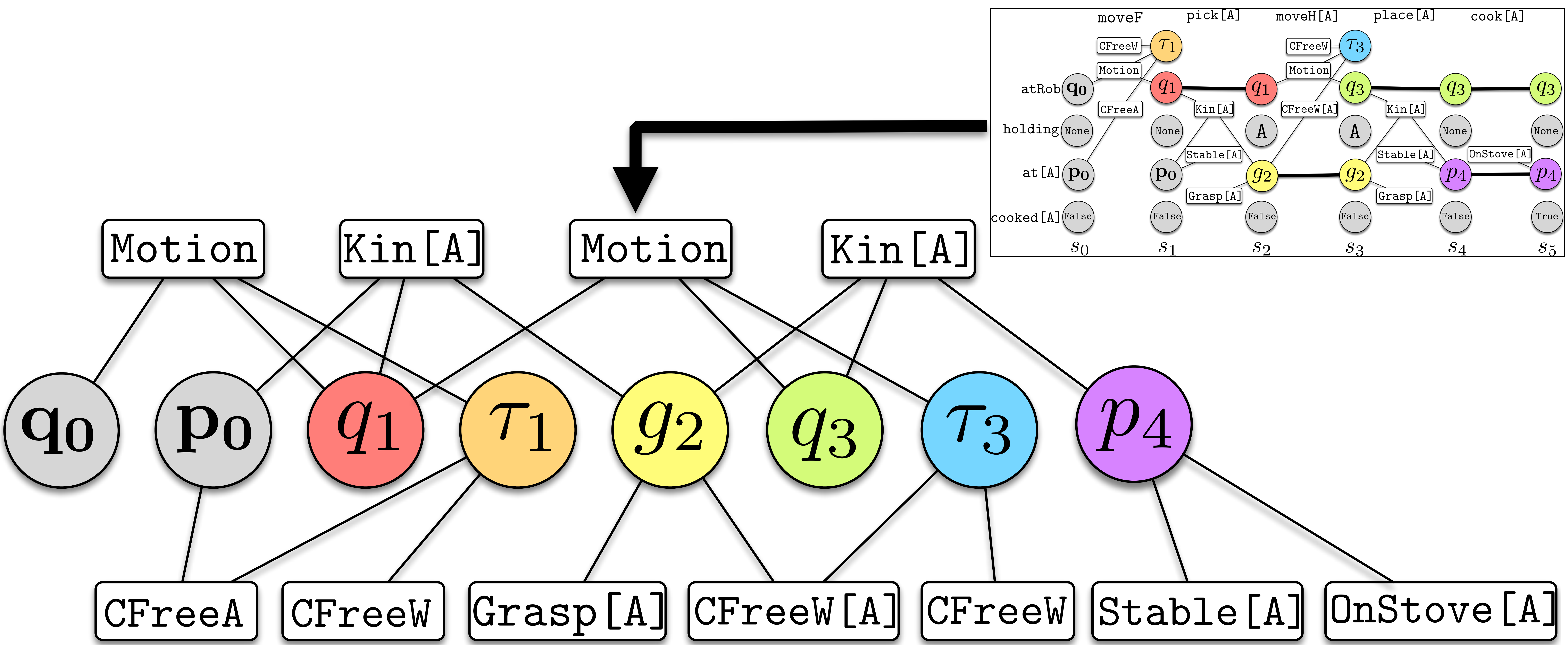
Example plan: $\pi = [\texttt{moveF}(\mathbf{q_0}, \tau_1, q_1), \texttt{pick[A]}(q_1, \mathbf{p_0}, g_2), \texttt{moveH[A]}(q_1, \tau_3, q_3), \texttt{place[A]}(q_3, p_4, g_2), \texttt{cook[A]}(p_4)]$



Constraints

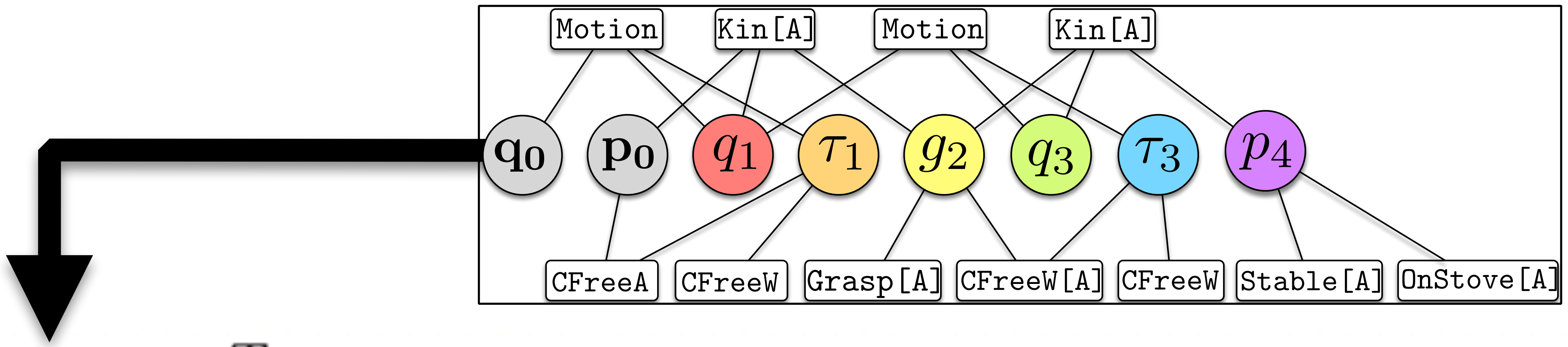
Constraint Network (Factor Graph)

- Compress plan skeleton into a constraint network
- Undirected bipartite graph of **variables & constraints**
- Can address with **optimization** and/or **sampling**



Joint Optimization

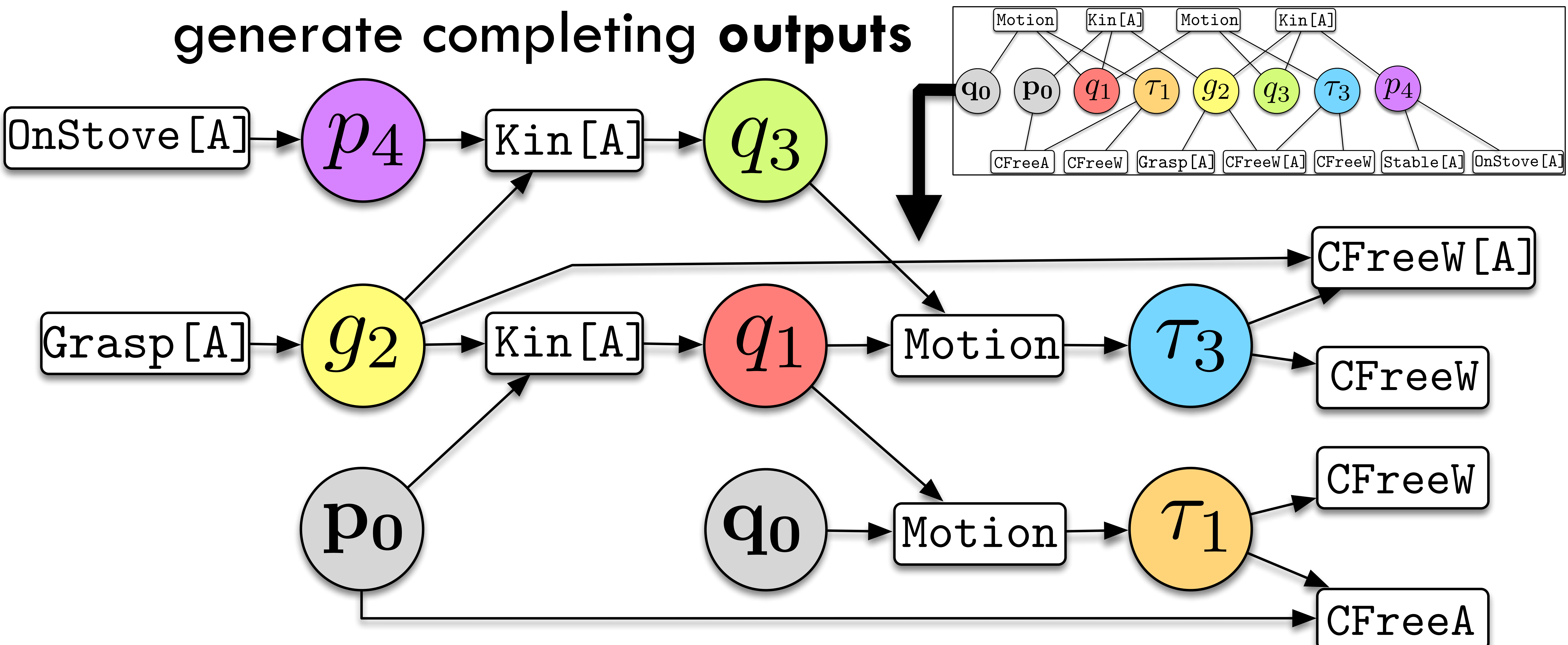
- Constraint network is a **mathematical program**
- **Hard to solve:** non-convex, constrained, mixed integer
- Often reduce to a sequence of simpler (unconstrained, quadratic) programs
- First- (**gradient**) vs second-order (**Hessian**) methods



$$\begin{aligned}
\underset{q_1,\tau_1,g_2,\tau_3,q_3,p_4}{\text{minimize}} \quad & \textstyle\sum_{t=1}^{T} f_{\texttt{moveF}}(\tau_1[t], \tau_1[t-1]) + \sum_{t=1}^{T} f_{\texttt{moveH[A]}}(g_1, \tau_3[t], \tau_3[t-1]) \\
\text{subject to} \quad & g_{\texttt{Grasp[A]}}(g_1) = 0,\ g_{\texttt{Stable[A]}}(p_4) = 0,\ h_{\texttt{OnStove[A]}}(p_4) \le 0 \\
& g_{\texttt{Kin[A]}}(q_1, \mathbf{p_0}, g_2) = 0,\ g_{\texttt{Kin[A]}}(q_3, p_4, g_1) = 0 \\
& h_{\texttt{Motion}}(\tau_1[t], \tau_1[t-1]) \le 0,\ h_{\texttt{Motion}}(\tau_3[t], \tau_3[t-1]) \le 0 \qquad \text{for } t \in [T] \\
& h_{\texttt{CFreeW}}(\tau_1[t]) \le 0,\ h_{\texttt{CFreeA}}(\mathbf{p_0}, \tau_1[t]) \le 0 \qquad \text{for } t \in [T] \\
& h_{\texttt{CFreeW}}(\tau_3[t]) \le 0,\ h_{\texttt{CFreeW[A]}}(g_1, \tau_3[t]) \le 0 \qquad \text{for } t \in [T]
\end{aligned}$$

Sampling Network

- Satisfy constraint network **compositionally**
- Directed Acyclic Graph (DAG) **computation graph**
  - **Conditional** samplers consume **inputs** and generate completing **outputs**



Taxonomy of TAMP Approaches

	Pre-discretized	Sampling	Optimization
Satisfaction first	Ferrer-Mestres et al. (84, 85)[b]	Siméon et al. (22)[a] Hauser et al. (13, 14, 29)[a] Garrett et al. (21, 86)[b] Krontiris & Bekris (87, 88)[a] Akbari & Rosell (89)[b] Vega-Brown & Roy (90)[a]	
Interleaved	Dornhege et al. (62, 63, 91)[b] Gaschler et al. (92–94)[b] Colledanchise et al. (95)[b]	Gravot et al. (96, 97)[b] Stilman et al. (23, 98, 99)[a] Plaku & Hager (100)[a] Kaelbling & Lozano-Pérez (101, 102)[b] Barry et al. (30, 103, 104)[a] Garrett et al. (70, 71)[b] Thomason & Knepper (105)[b] Kim et al. (106, 107)[b] Kingston et al. (108)[a]	Fernández-González et al. (109)[b]
Sequencing first	Nilsson (3)[b] Erdem et al. (74, 75)[b] Lagriffoul et al. (65–67)[b] Pandey et al. (110, 111)[b] Lozano-Pérez & Kaelbling (112)[b] Dantam et al. (77–79)[b] Lo et al. (113)[b]	Wolfe et al. (114)[b] Srivastava et al. (60, 76)[b] Garrett et al. (55, 73)[b]	Toussaint et al. (61, 68, 69)[b] Shoukry et al. (81–83)[b] Hadfield-Menell et al. (115)[b]

[a]Approaches for MMMP.
[b]Approaches for TAMP.

Garrett et al., 2021. *"Integrated Task and Motion Planning", Annual Review of Control, Robotics, and Autonomous Systems.*

My Approach: PDDLStream

- **No general-purpose, flexible framework** for planning in a variety of TAMP domains
- Extends **PDDL** to incorporate **sampling procedures**
  - Can model domains with **infinitely-many** actions

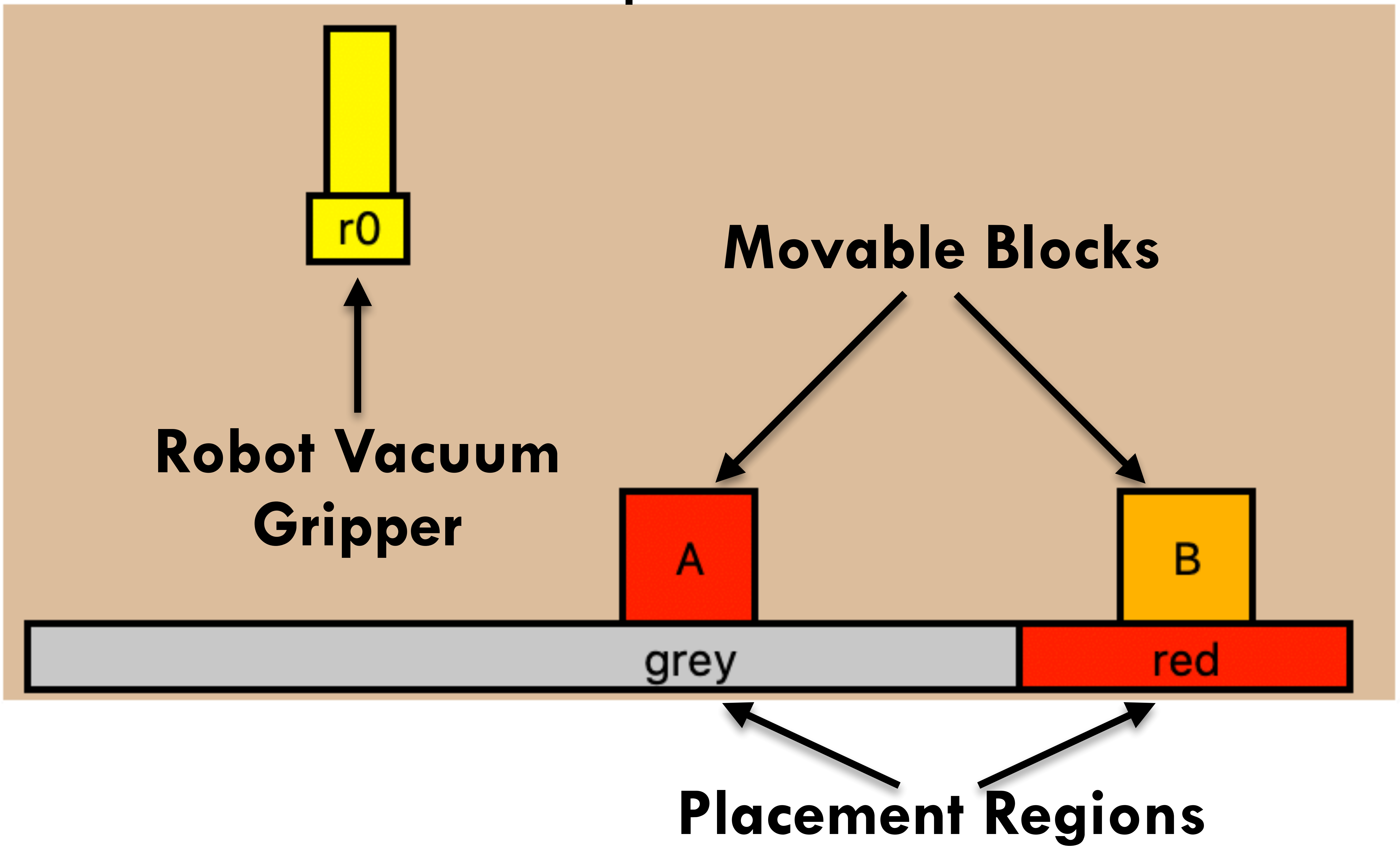
- Develop **domain-independent** algorithms that treat the samplers as **blackbox inputs**
- Algorithms solve a **sequence of finite** PDDL problems
  - Leverage existing **classical planners** as subroutines
- Algorithms are particularly **fast when downward refinement holds** while remaining **complete**

PDDLStream Language

PDDLStream: Integrating Symbolic Planners and Blackbox Samplers via Optimistic Adaptive Planning. Caelan Reed Garrett, Tomás Lozano-Pérez, Leslie Pack Kaelbling. *International Conference on Automated Planning and Scheduling (ICAPS)*, 2020.

2D Pick-and-Place Domain

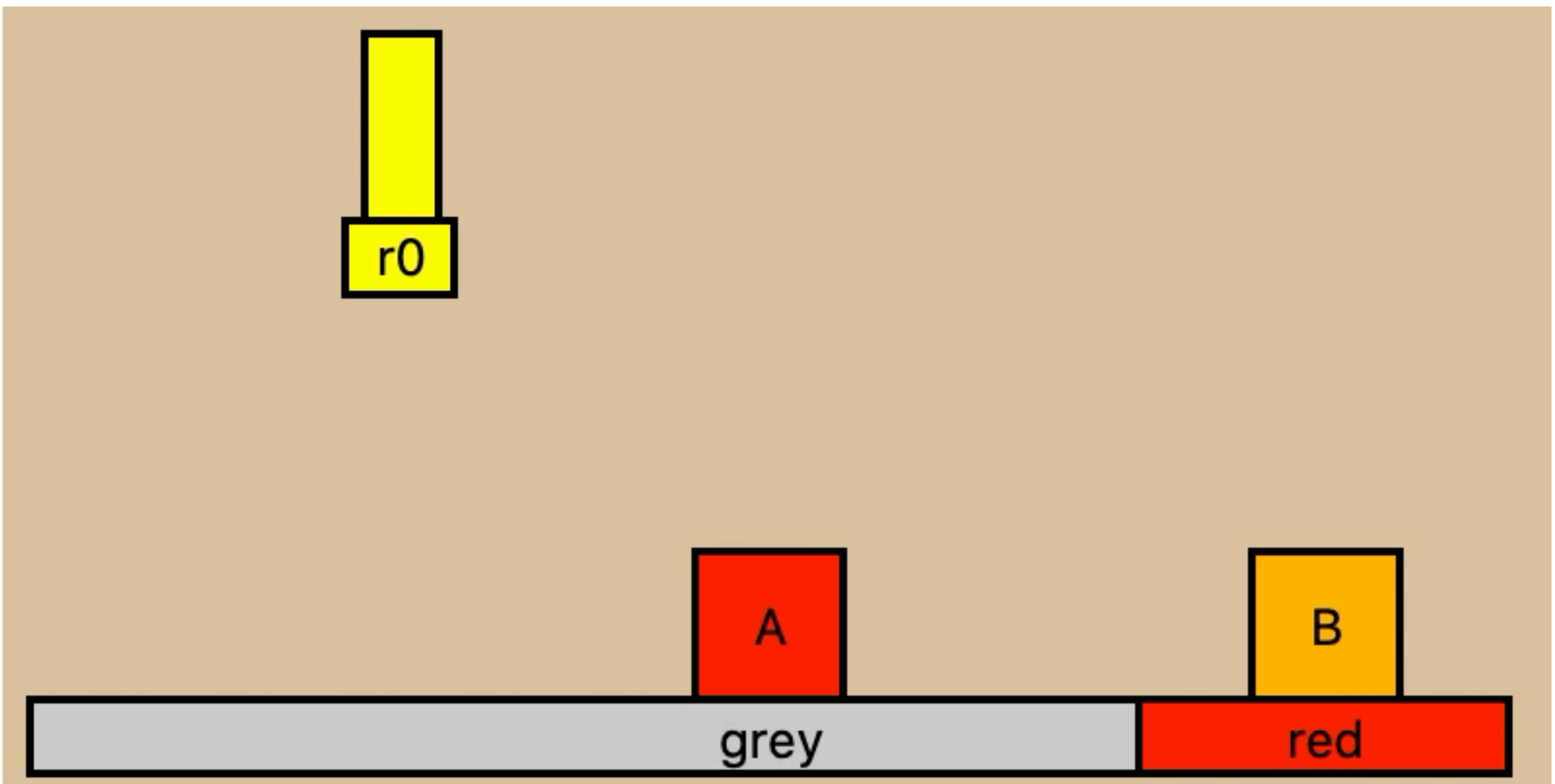
- Robot and block poses are continuous `[x y]` pairs
- **Goal:** block **`A`** within the **`red`** region
  - Block **`B`** obstructs the placement of **`A`**



2D Pick-and-Place Solution

- One (of **infinitely many**) possible solutions

```
[move(…), pick(B,…), move(…), place(B,…),
 move(…), pick(A,…), move(…), place(A,…)]
```



2D Pick-and-Place Initial & Goal

- Not all values are discrete, some are **continuous**
- **Static** (constant) initial **facts - satisfied constraints**

$F =$ `{Block(`**`A`**`), Block(`**`B`**`), Region(`**`red`**`), Region(`**`grey`**`), Conf([-7.5, 5]), Pose(`**`A`**`, [0. 0.]), Pose(`**`B`**`, [7.5 0.])}`

- **Fluent** (changing) initial facts - **state variables**

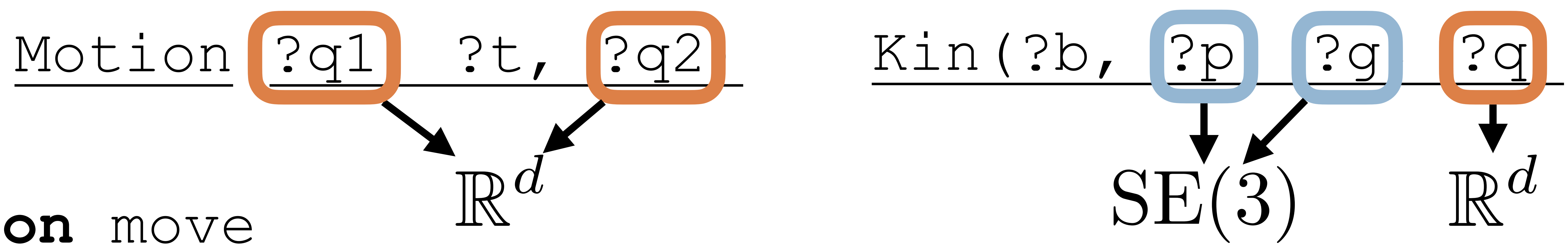
$s_0 =$ `{AtConf([-7.5 5.]), HandEmpty(), AtPose(`**`A`**`, [0. 0.]), AtPose(B, [7.5 0.])}`

- **Goal** logical formula - set of **goal states**

$S_* =$ **`exists`**`(?p){Contained(`**`A`**`,?p,`**`red`**`),AtPose(`**`A`**`,?p)}`

Pick-and-Place Actions

- Typical PDDL action description except that arguments are **high-dimensional & continuous**!
- To use, must **satisfy** static facts **(constraints)**




```
(:action move
 :parameters (?q1, ?t, ?q2)
 :precondition {Motion(?q1, ?t, ?q2), AtConf(?q1)}
 :effect {AtConf(?q2), ¬AtConf(?q1)})
```

$A_{\texttt{move}}$

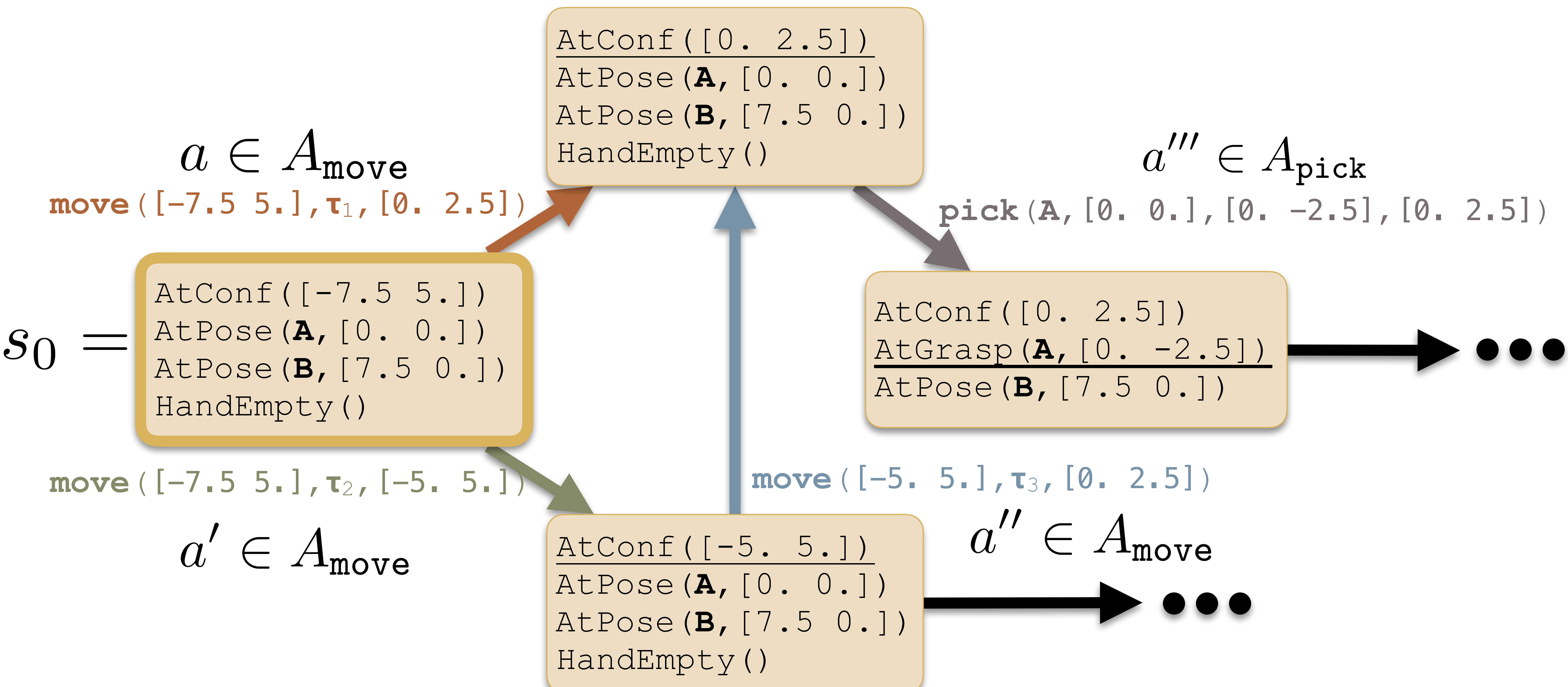
```
(:action pick
 :parameters (?b, ?p, ?g, ?q)
 :precondition {Kin(?b, ?p, ?g, ?q), AtConf(?q),
                AtPose(?b, ?p), HandEmpty()}
 :effect {AtGrasp(?b,?g), ¬AtPose(?b,?p), ¬HandEmpty()})
```

$A_{\texttt{pick}}$

Search in Discretized State-Space

Suppose an **oracle** gave use the following values and facts:

$F =$ {**Motion**([-7.5 5.],τ1,[0. 2.5]), **Motion**([-7.5 5.],τ2,[-5. 5.]), **Motion**([-5. 5.],τ3,[0. 2.5]), **Kin**(**A**,[0. 0.],[0. -2.5],[0. 2.5]), ...}

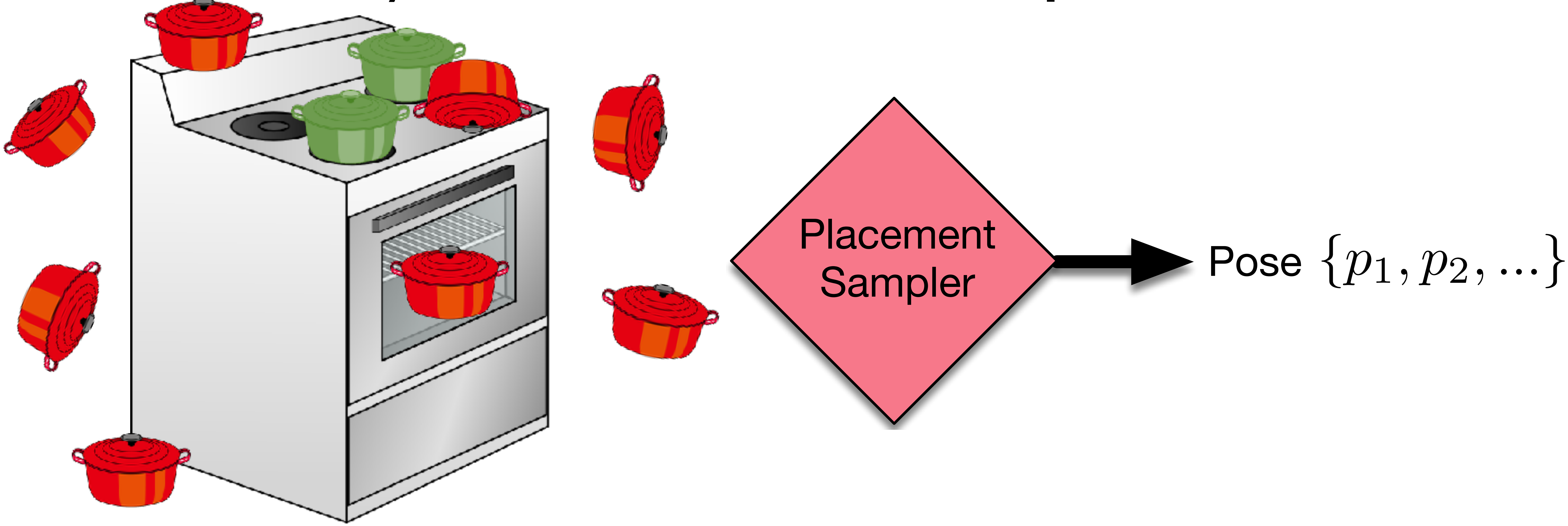


No a Priori Discretization

- Values **given** at start:
  - 1 initial configuration: `Conf([-7.5 5.])`
  - 2 initial poses: `Pose(A, [0. 0.])` `Pose(B, [7.5 0.])`
- **Planner** needs to **find:**
  - 1 pose for **A** within **red**: `Contain(A, ?p, red)`
  - 1 collision-free pose for **B**: `CFree(A, ?p, B, ?p2)`
  - 1 grasp for **A** and **B**: `Grasp(A,?g),Grasp(B,?g)`
  - 4 grasping configurations: `Kin(?b, ?p, ?g, ?q)`
  - 4 robot trajectories: `Motion(?q1, ?t, ?q2)`

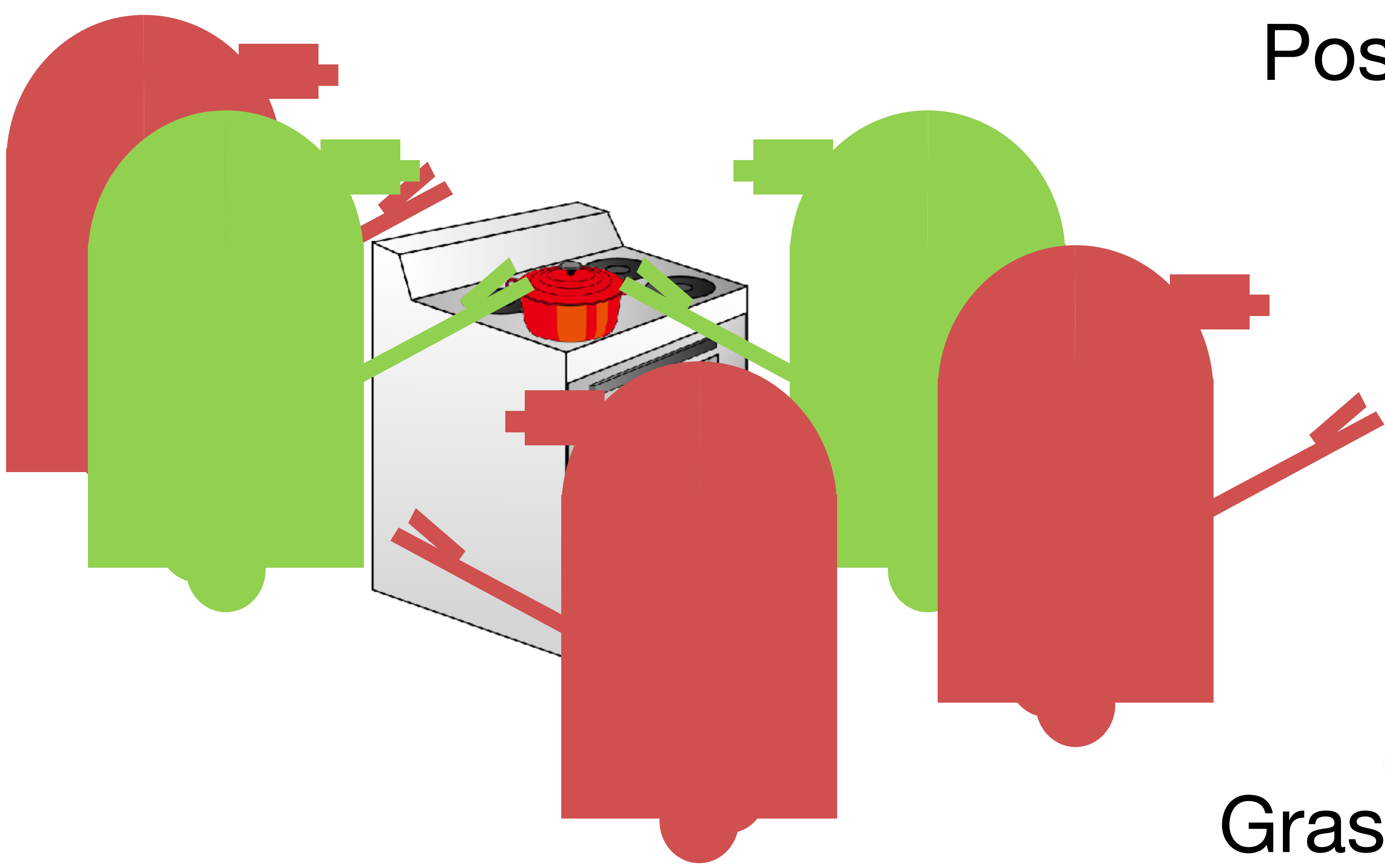
What Samplers Do We Need?

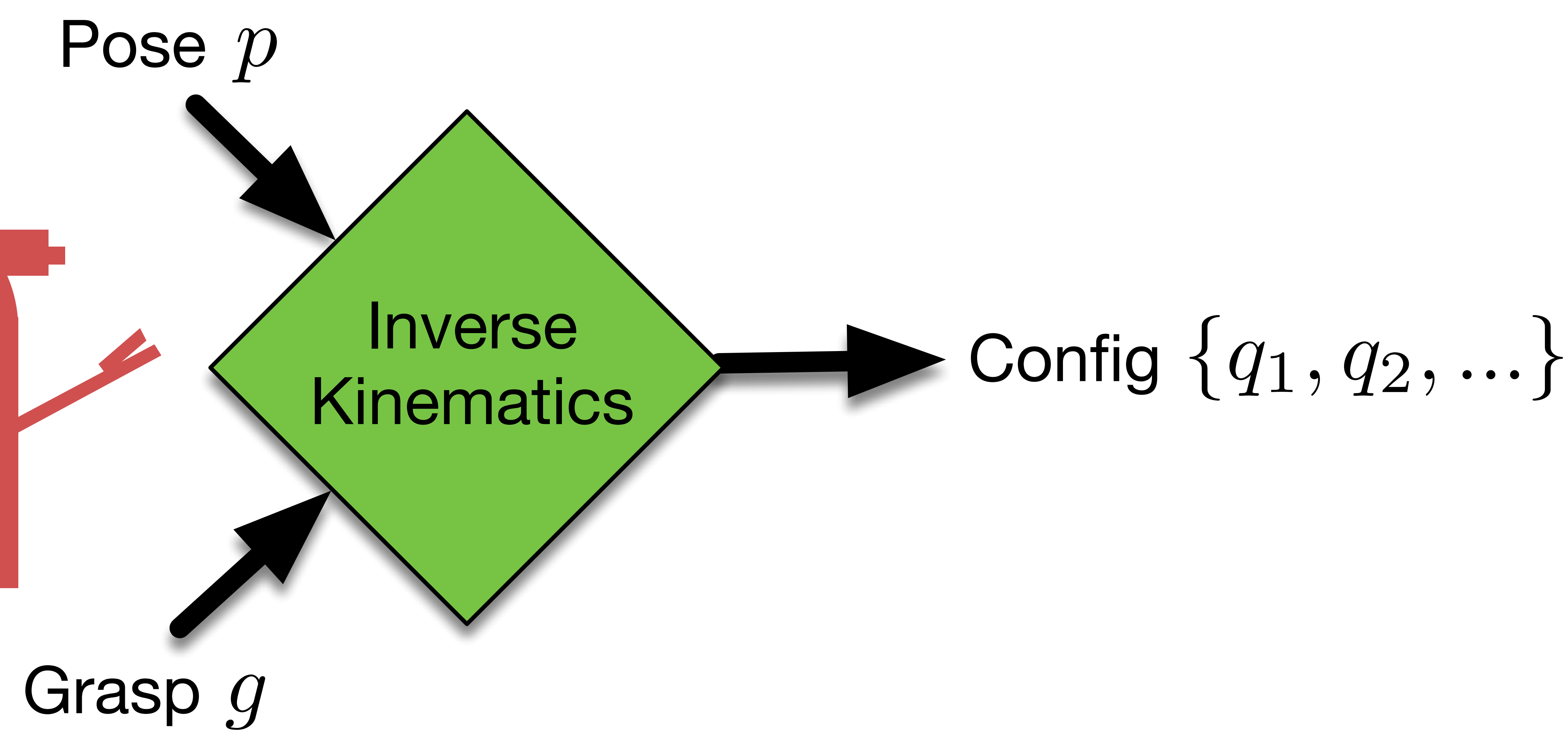
- **Low-dimensional** placement stability constraint (`Contain`)
  - e.g. 1D line embedded in 2D placement space
- Directly **sample values that satisfy the constraint**
- May need **arbitrarily many** samples
  - Gradually enumerate an **infinite sequence**



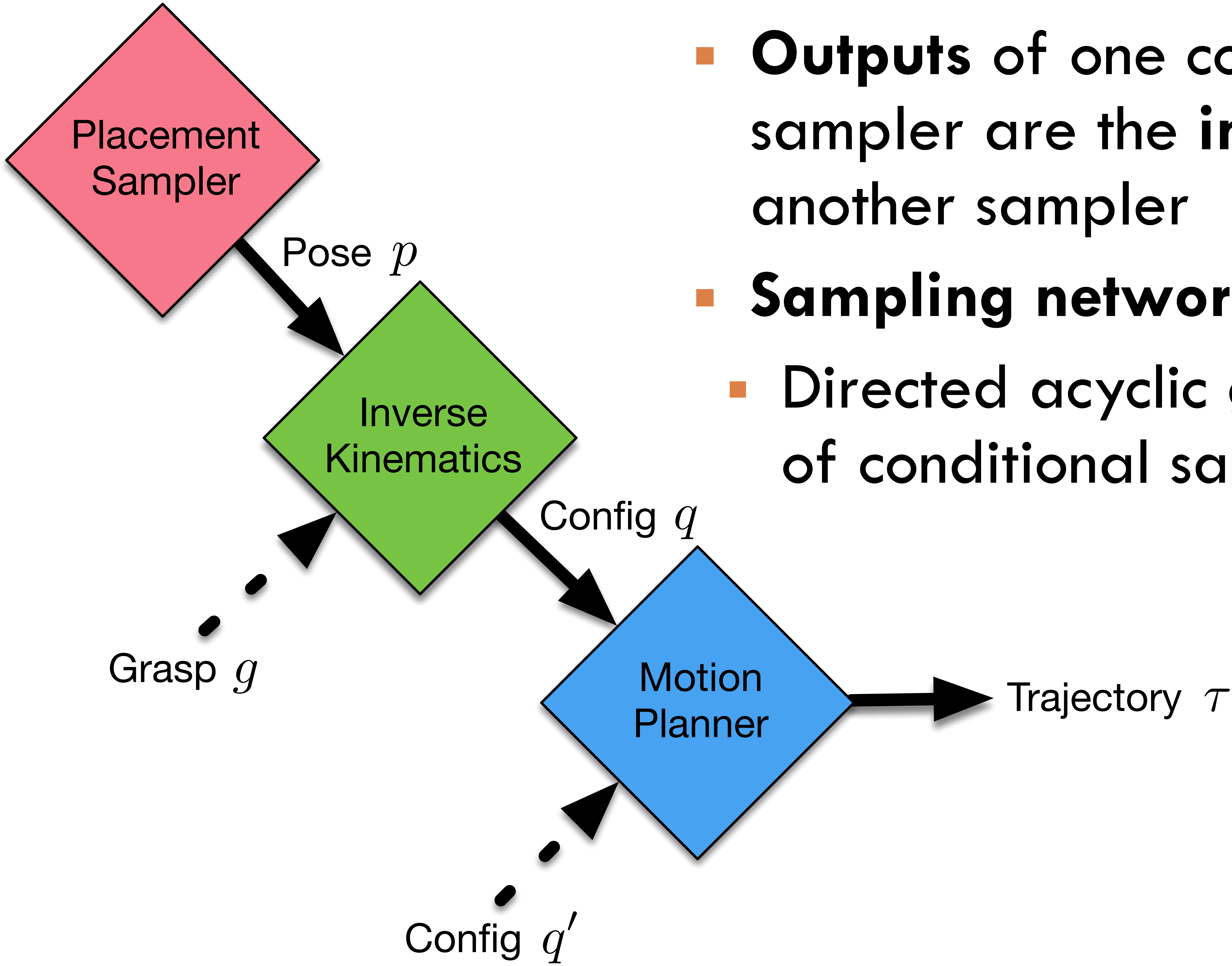
Intersection of Constraints

- **Kinematic constraint** (`Kin`) involves poses, grasps, and configurations
- **Conditional samplers** - function from **input values** to a sampler that generates **output values**





Composing Conditional Samplers



- **Outputs** of one conditional sampler are the **inputs** to another sampler
- **Sampling network**
  - Directed acyclic graph (DAG) of conditional samplers

Stream: Specification for a Sampler

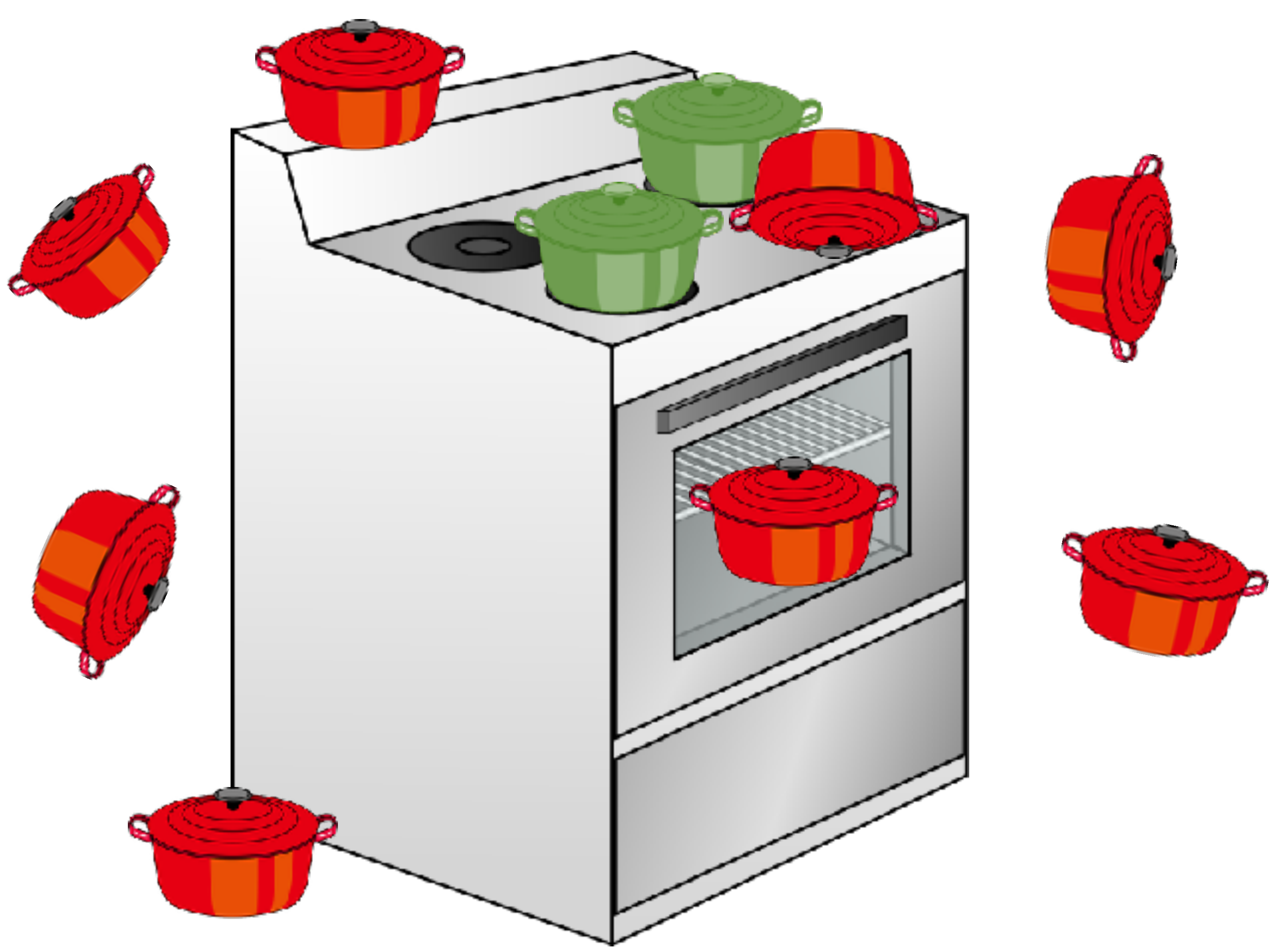
- What do **inputs** & **outputs** represent?
  - Communicate semantics using **predicates** (constraints)

- Declarative stream specification:
  - **Domain facts** - static facts declaring legal **inputs**
    - e.g. only configurations can be motion planner inputs

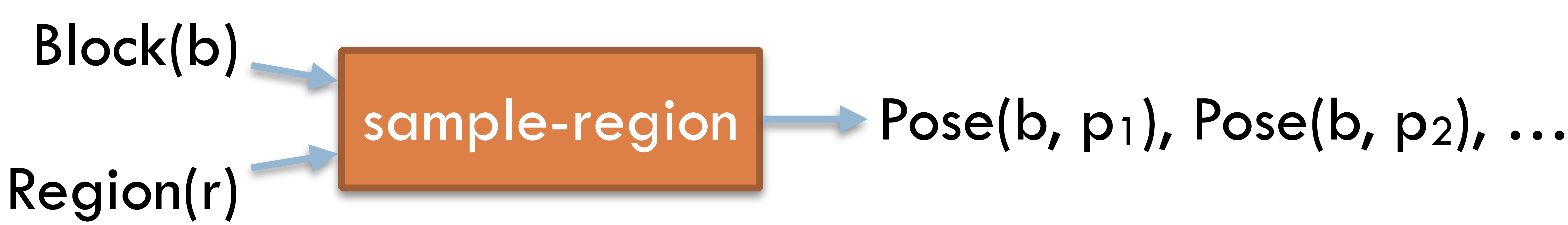
  - **Certified facts** - static facts that all **outputs** are **asserted** to satisfy with their corresponding **inputs**
    - e.g. poses sampled from a region are within it

Sampling Placements in a Region


```
(:stream sample-region
 :inputs (?b, ?r)
 :domain {Block(?b), Region(?r)}
 :outputs (?p)
 :certified {Pose(?b, ?p), Contain(?b, ?p, ?r)})
```



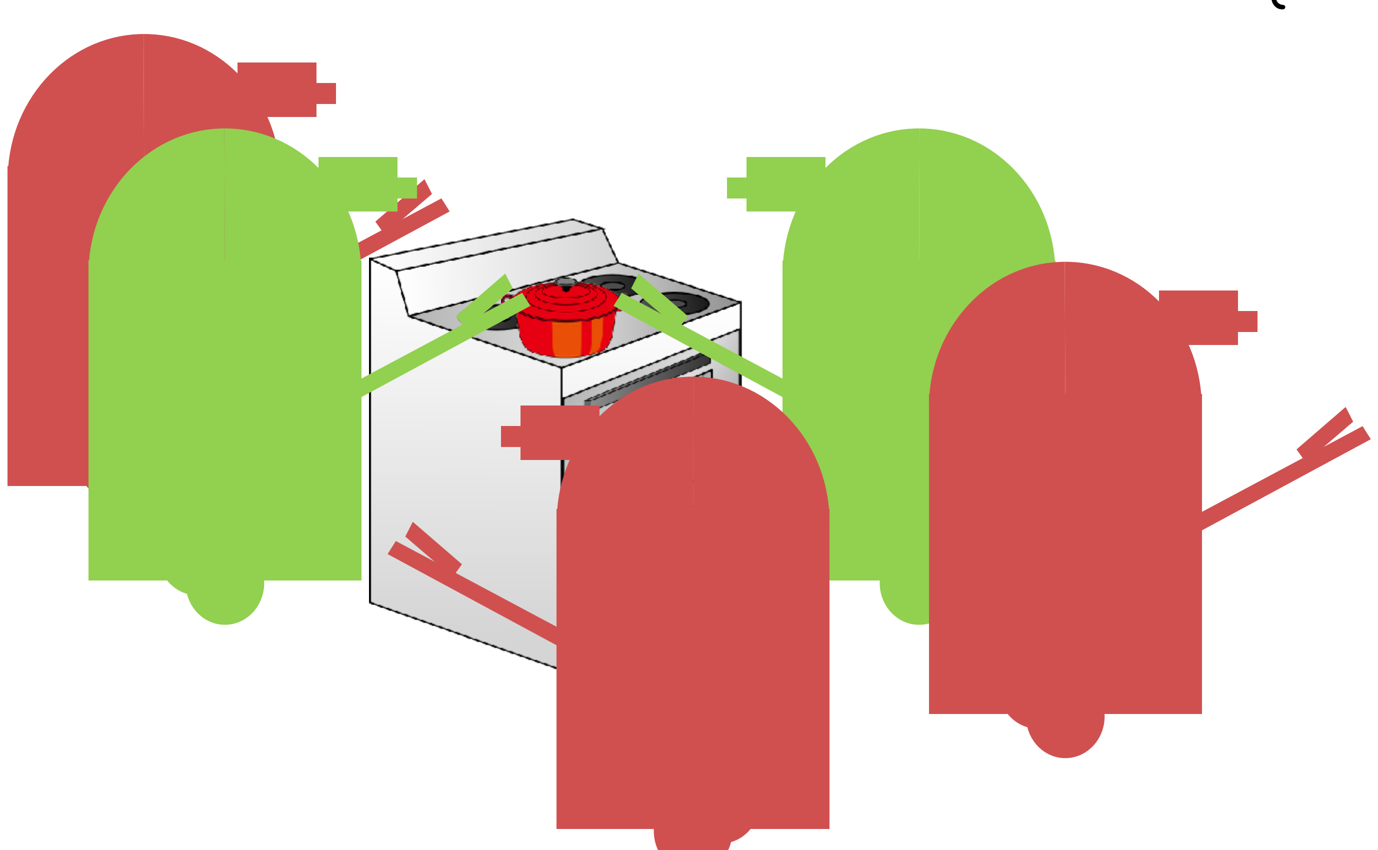
```
def sample_region(b, r):
  x_min, x_max = REGIONS[r]
  w = BLOCKS[b].width
  while True:
      x = random.uniform(x_min + w/2,
                         x_max - w/2)
      p = np.array([x, 0.])
      yield (p,)
```

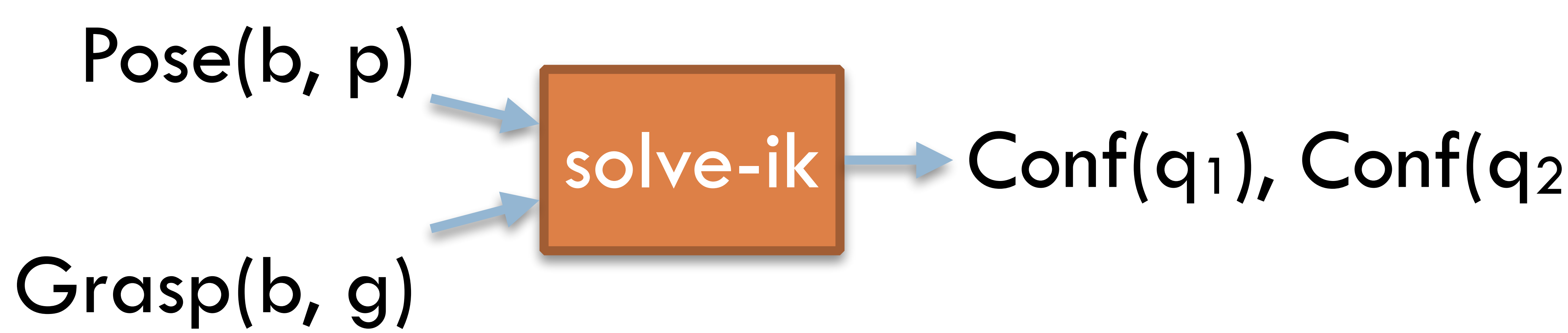


Sampling IK Solutions

- **Inverse kinematics** (IK) to produce robot grasping configurations
- Trivial in 2D, non-trivial in general (e.g. 7-DOF arm)

```
(:stream solve-ik
 :inputs (?b, ?p, ?g)
 :domain {Pose(?b, ?p), Grasp(?b, ?g)}
 :outputs (?q)
 :certified {Conf(?q), Kin(?b, ?p, ?g, ?q)})
```

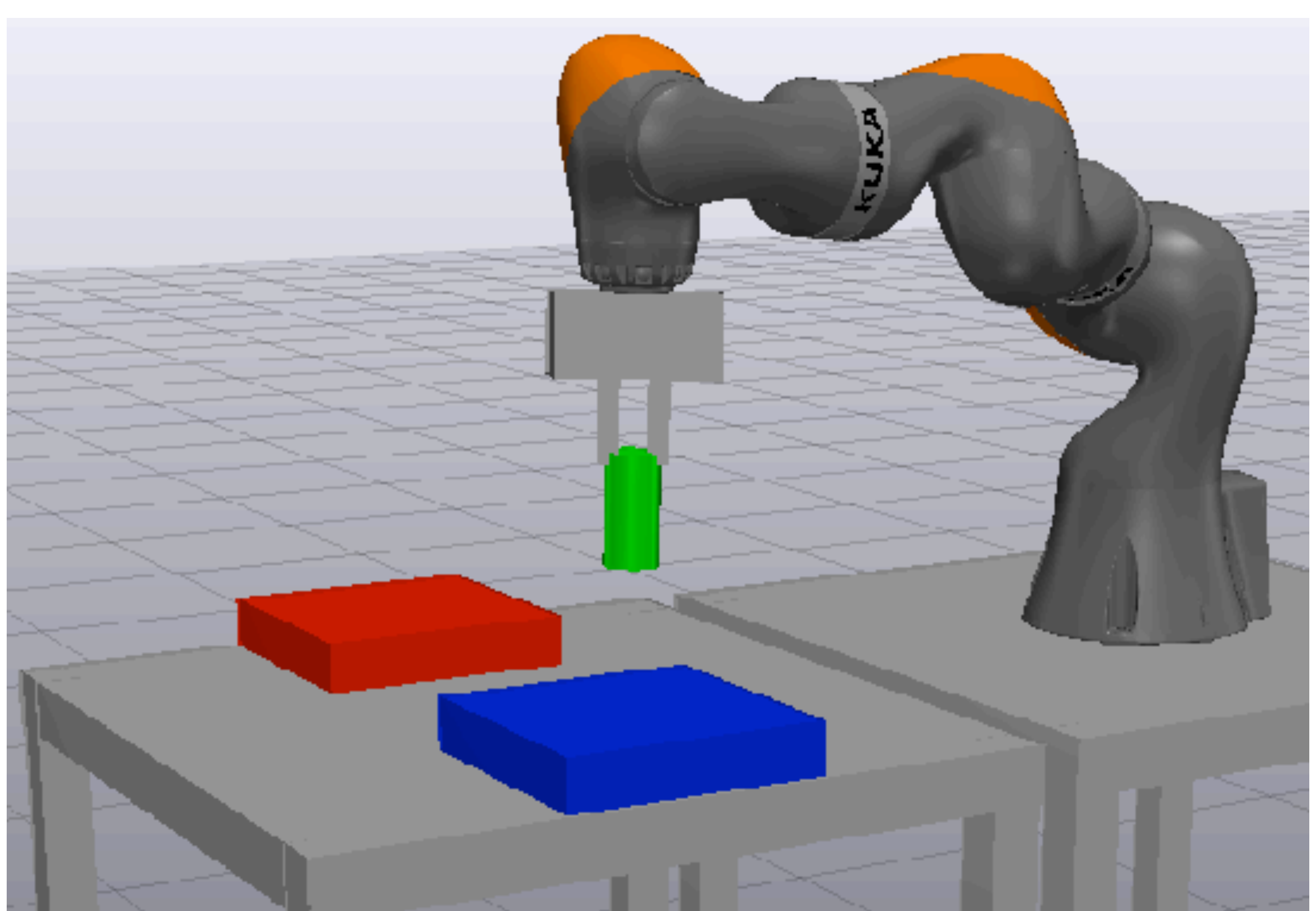


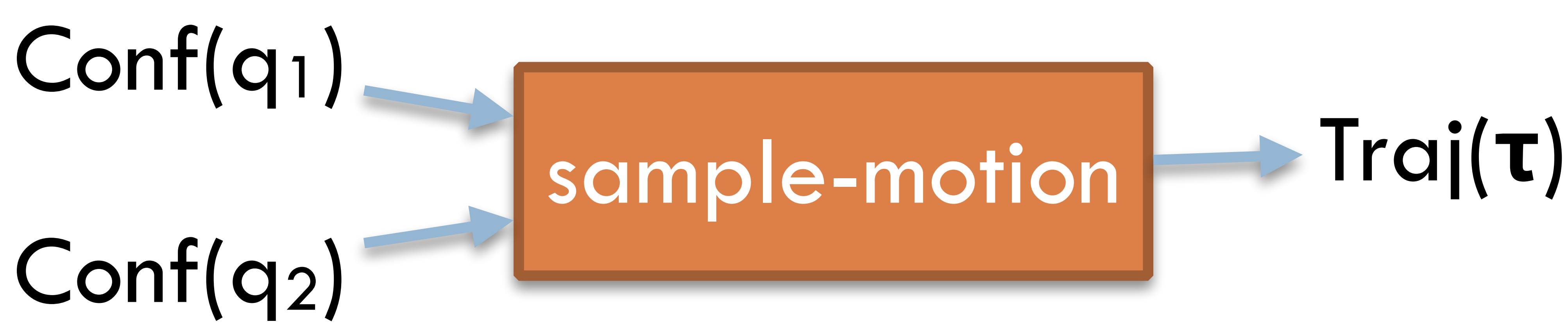


Invoking a Motion Planner

- "Sample" **multi-waypoint** robot **trajectories**
- Use off-the-shelf motion planner (e.g. RRT)

```
(:stream sample-motion
 :inputs (?q1, ?q2)
 :domain {Conf(?q1), Conf(?q2)}
 :outputs (?t)
 :certified {Traj(?t), Motion(?q1, ?t, ?q2)})
```





PDDLStream Algorithms

PDDLStream: Integrating Symbolic Planners and Blackbox Samplers via Optimistic Adaptive Planning. Caelan Reed Garrett, Tomás Lozano-Pérez, Leslie Pack Kaelbling. *International Conference on Automated Planning and Scheduling (ICAPS)*, 2020.

Two PDDLStream Algorithms

- PDDLStream **algorithms decide** which streams to use
- **Reduce** planning to a sequence of PDDL problems
  1. **Search** a finite PDDL problem for plan
  2. **Modify** the PDDL problem (depending on the plan)

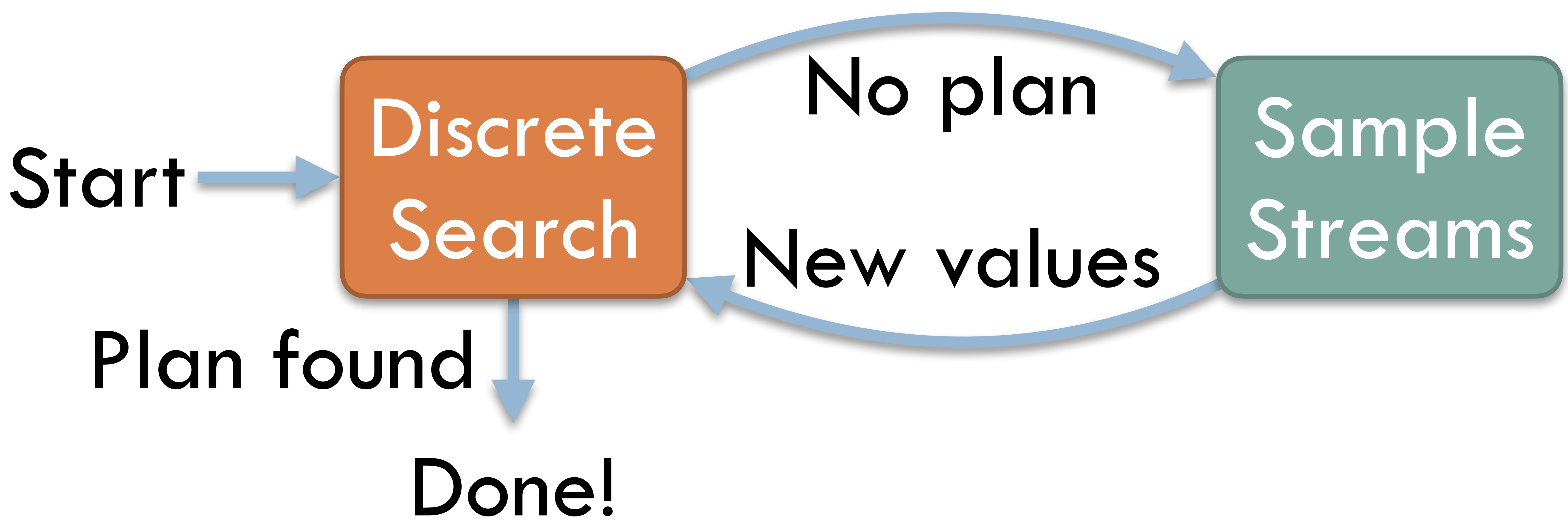
[Garrett 2018]
[Garrett 2020a]

- Implement search using off-the-shelf domain-independent **PDDL planners** (e.g. FastDownward)
  - **Greedy** best-first heuristic search
  - Exploit **factoring** in PDDL for heuristics (e.g. h_{FF})

Incremental Algorithm

- **Incrementally** grow the set of values and facts
- Repeat:

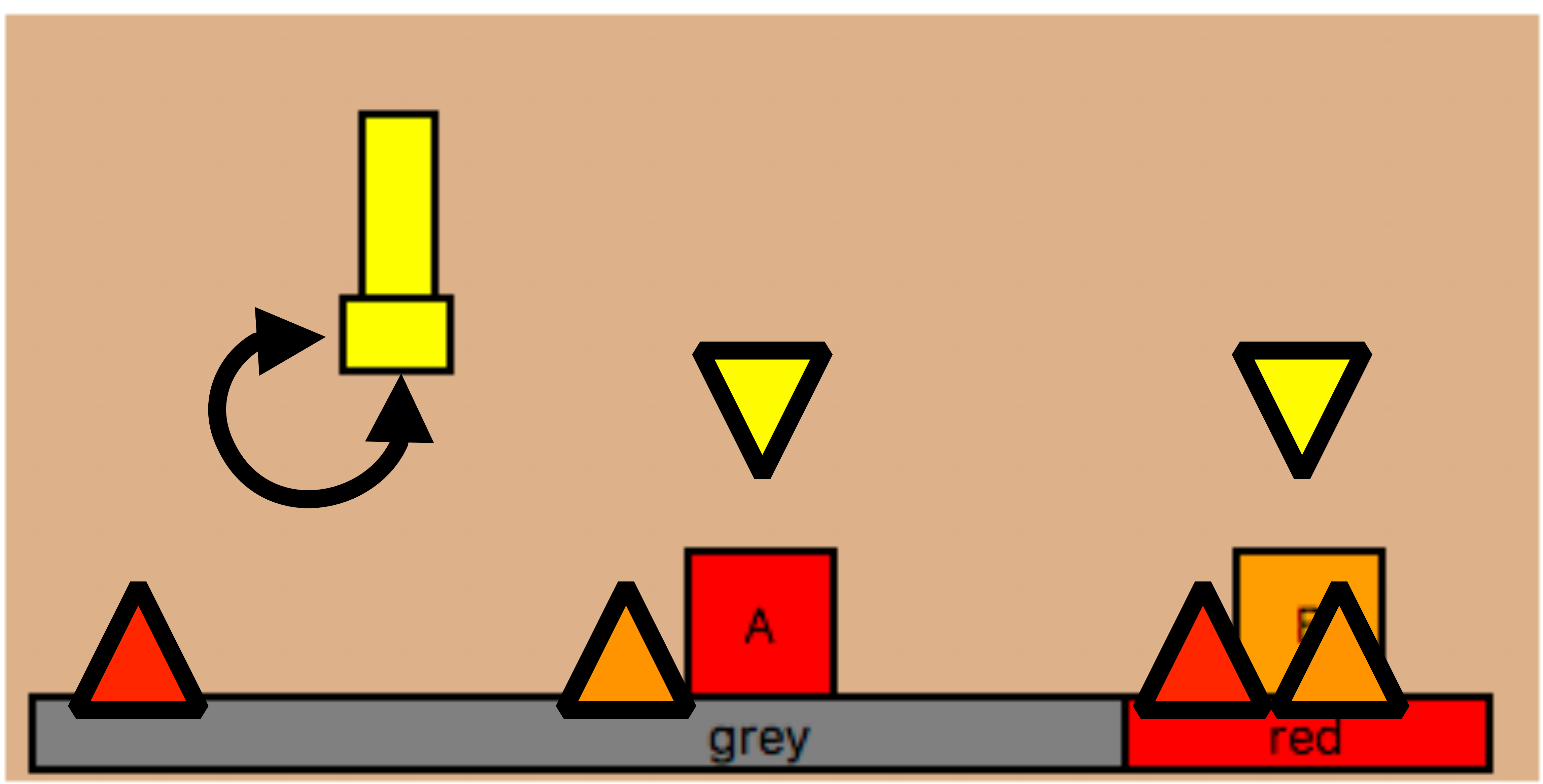
1. **Instantiate** and **sample** streams to generate new values and prove new facts
2. **Search** for a plan using the current values
3. **Return** when a plan is found



[Garrett 2018]
[Garrett 2020a]

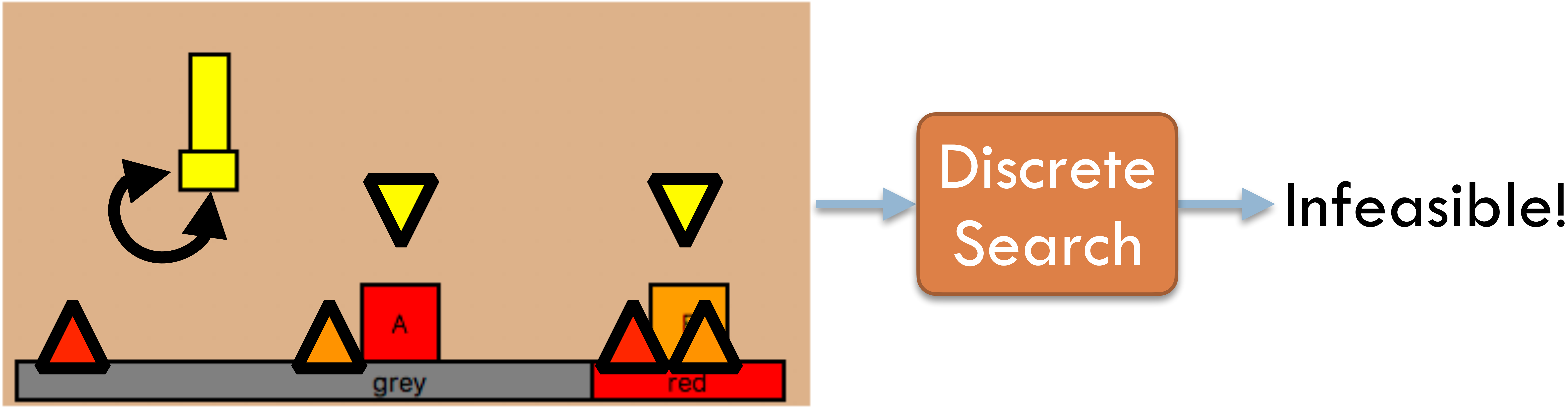
Incremental: Iteration 1 - Sampling

- **Iteration 1** - evaluated 14 streams
- **Sampled:**
  - 4 new block poses:
  - 2 new robot configurations:
  - 2 new trajectories:



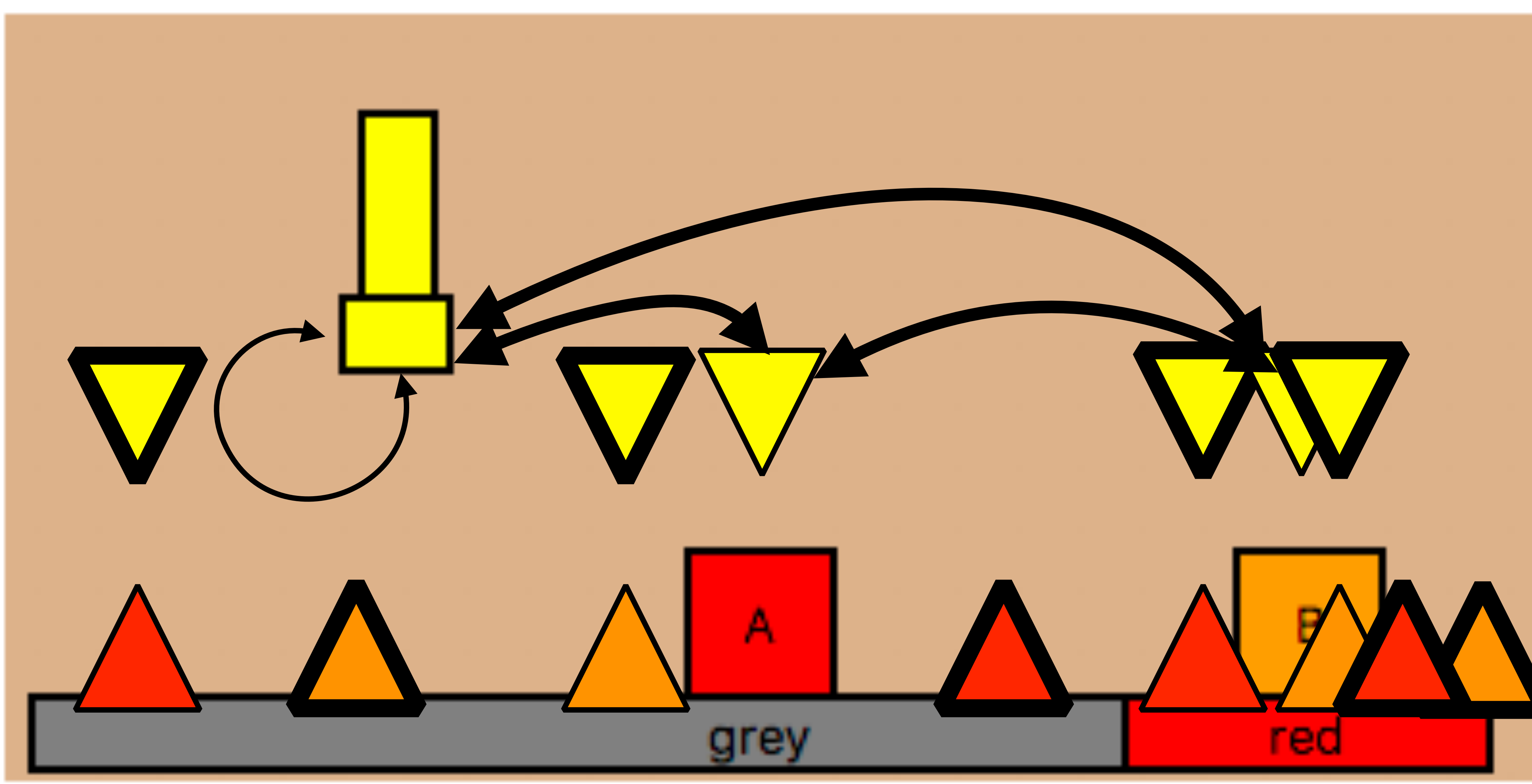
Incremental: Iteration 1 - Search

- Pass current discretization to FastDownward
- If **infeasible**, the current set of samples is insufficient



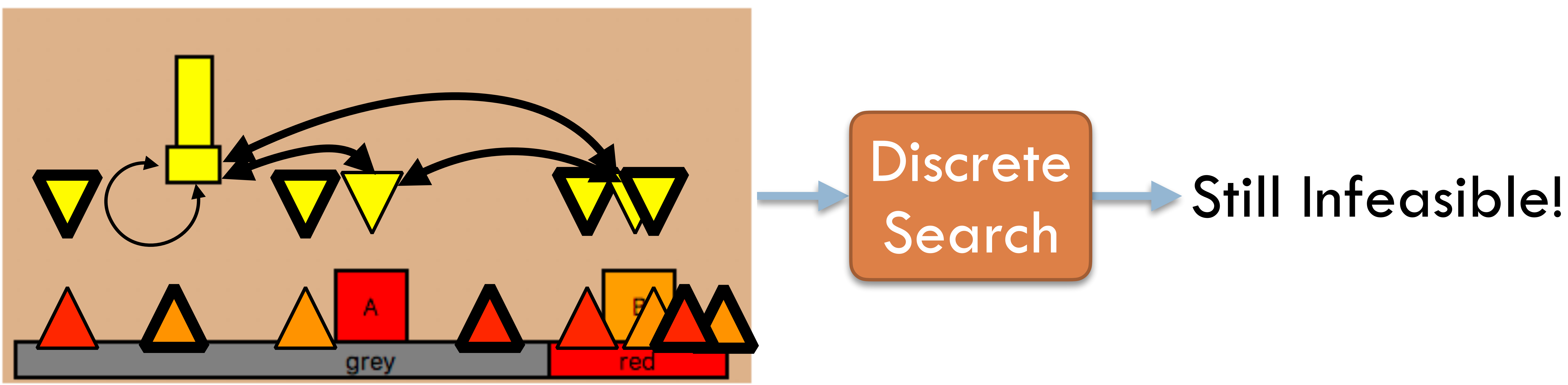
Incremental: Iteration 2 - Sampling

- **Iteration 2** - evaluated 54 streams
- **Sampled:**
  - 4 new block poses:
  - 4 new robot configurations:
  - 10 new trajectories:



Incremental: Iteration 2 - Search

- Pass current discretization to FastDownward
- If **infeasible**, the current set of samples is insufficient



Incremental Example: Iterations 3-4

Iteration 3 - 118 queried streams - infeasible

Iteration 4 - 182 queried streams - **solved!**

Solution:

1. move ([-7.5 5.], τ_1, [7.5 2.5])
2. pick (**B**, [7.5 0.], [0. -2.5], [7.5 2.5])
3. move ([7.5 2.5], τ_2, [10.97 2.5])
4. place(**B**, [10.97 0.], [0. -2.5], [10.97 2.5])
5. move ([10.97 2.5], τ_3, [0. 2.5])
6. pick (**A**, [0. 0.], [0. -2.5], [0. 2.5])
7. move ([0. 2.5], τ_4, [7.65 2.5])
8. place(**A**, [7.65 0.], [0. -2.5], [7.65 2.5])

- **Planner generated** all but the underlined values
- **Drawback** - many unnecessary samples produced

Optimistic Stream Evaluation

- Many TAMP streams are computationally **expensive**
  - Inverse kinematics, collision checking, motion planning
- Only query streams after they are **identified** as useful
  - Plan with **optimistic hypothetical** outputs

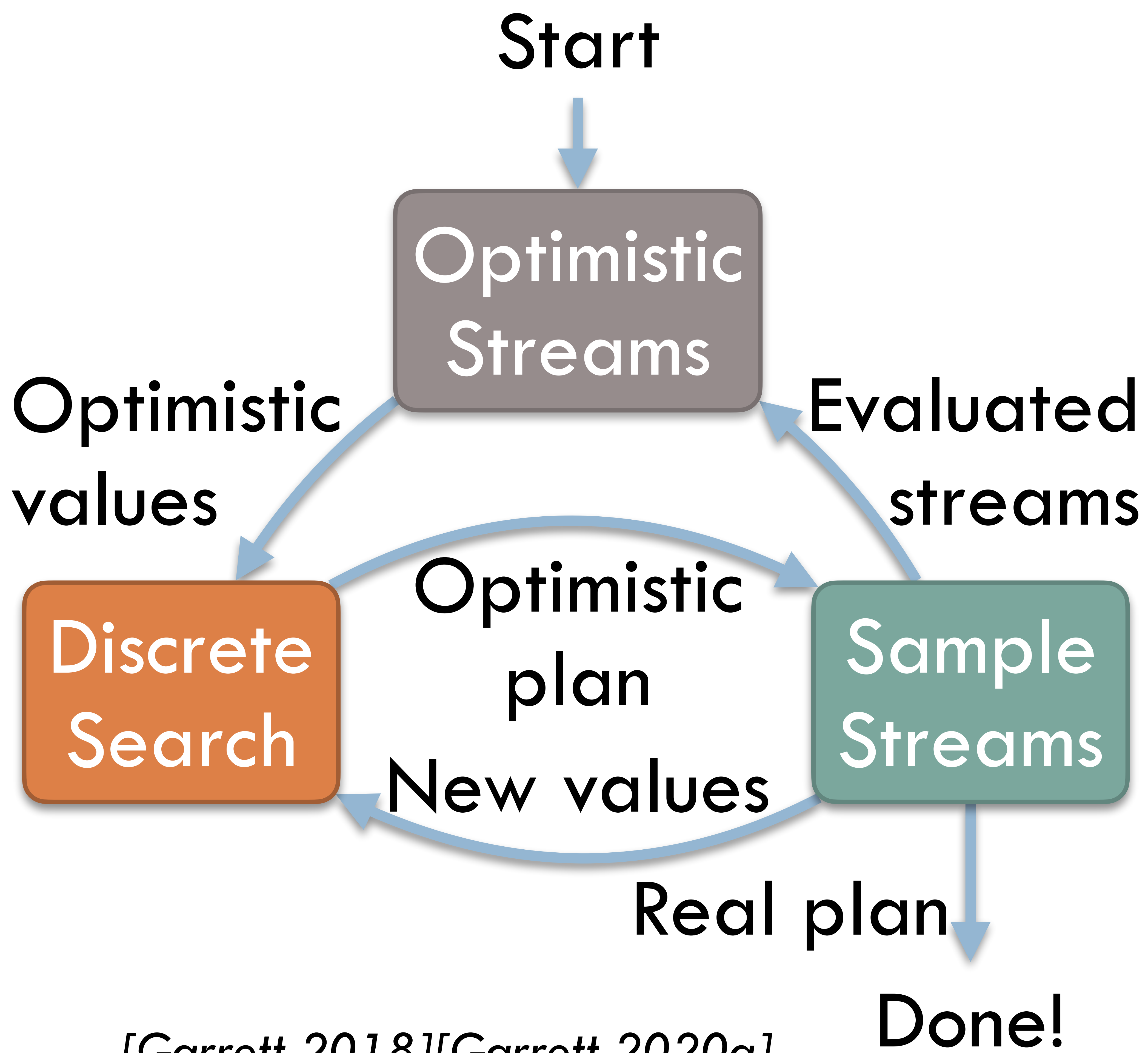
- Inductively create unique **optimistic placeholder values** for each stream output (denoted by prefix #)

1. `s-region(`**`A`**`, `**`red`**`)→#p0`
2. `s-ik(`**`A`**`, [0. 0.], [0. -2.5])→#q0,`
3. `s-ik(`**`A`**`, #p0, [0. -2.5])→#q2,`
4. `s-motion(`**`A`**`, #q0, #q2)→#t0, …`

[Garrett 2018]
[Garrett 2020a]

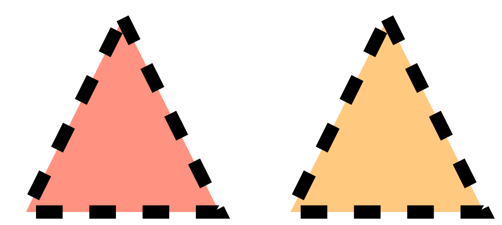
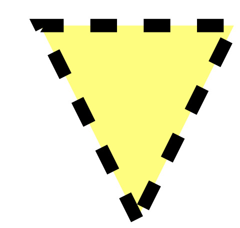
Focused Algorithm

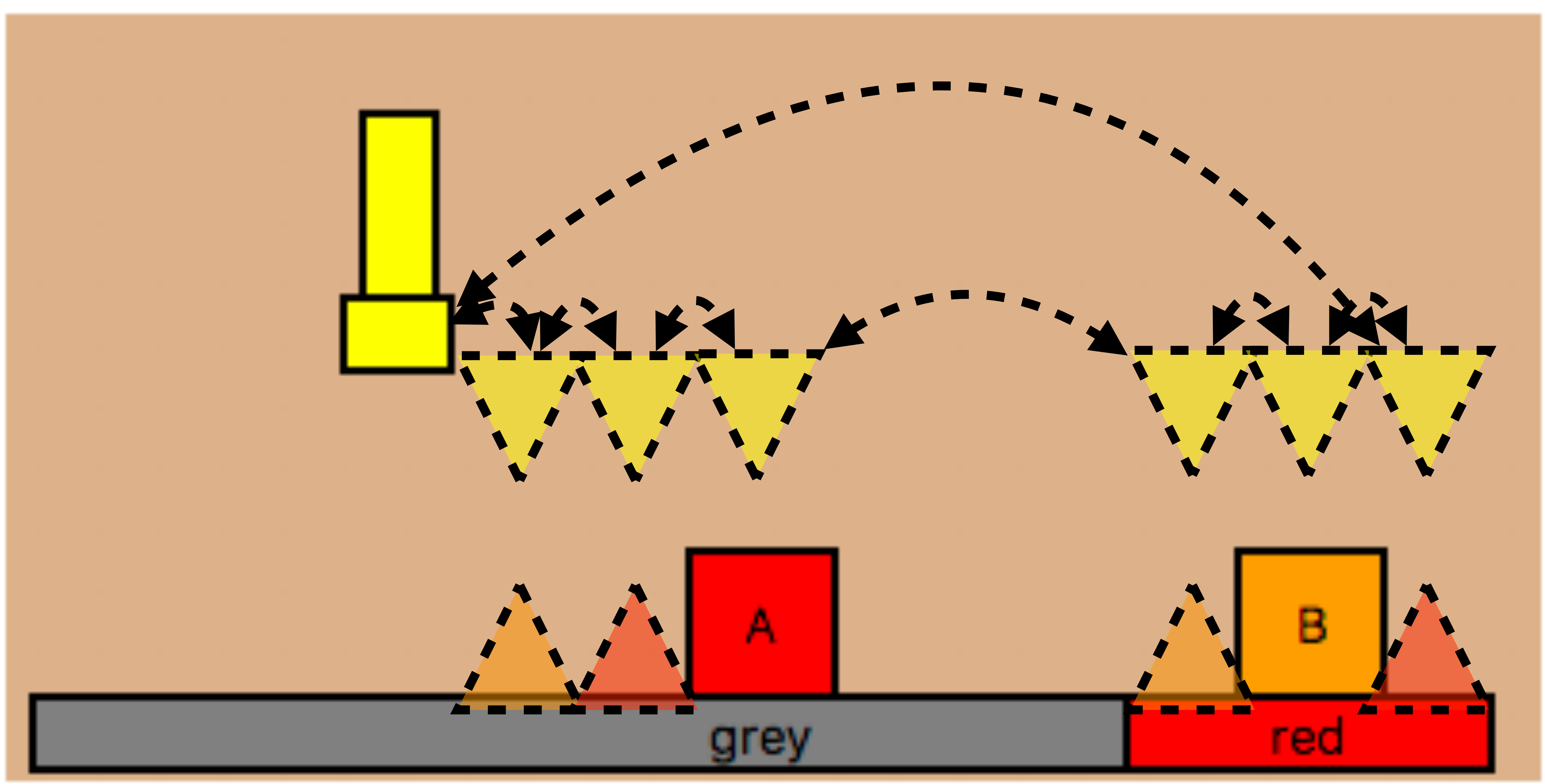
- **Lazily** plan using optimistic values **before** real values
- Repeat:
  1. **Construct** optimistic stream outputs
  2. **Search** with real & optimistic values
  3. **Retrace** and **evaluate** streams
  4. **Replace** optimistic with real if they exist
  5. **Return** if all succeed



[Garrett 2018][Garrett 2020a]

Focused: Iteration 1

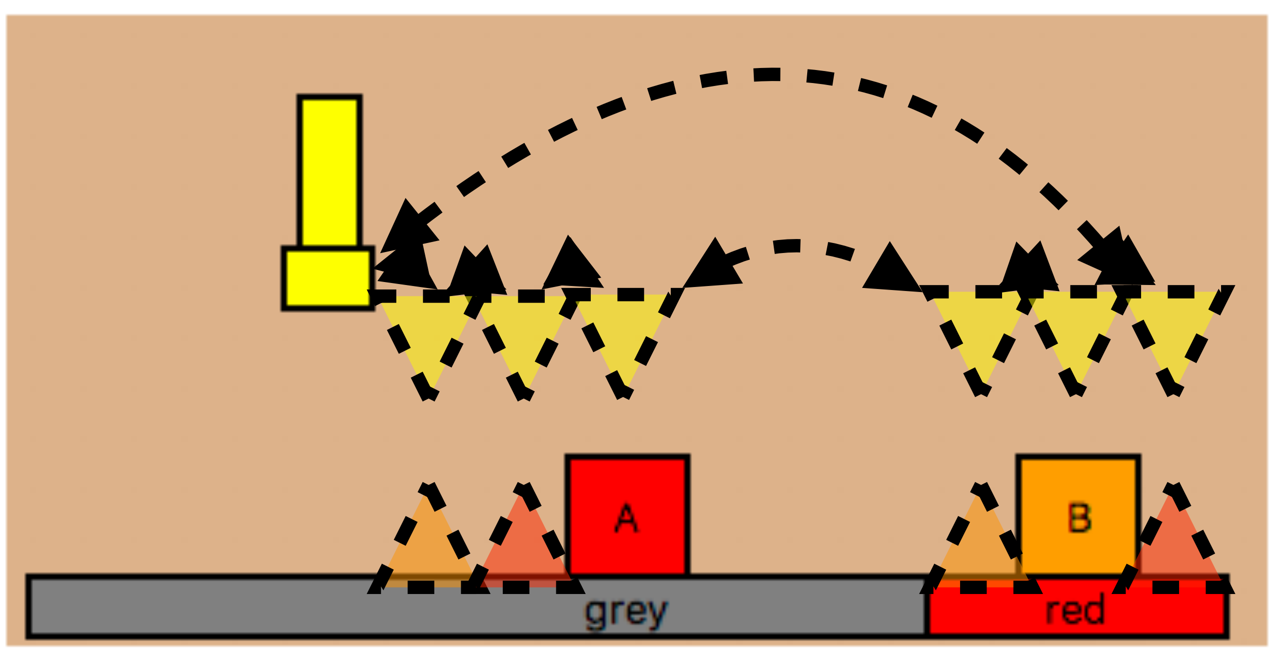
- **Iteration 1** - optimistically evaluated 46 streams
- **Created:**
  - 4 optimistic block poses: 
  - 6 optimistic robot configurations: 
  - 36 optimistic trajectories: 

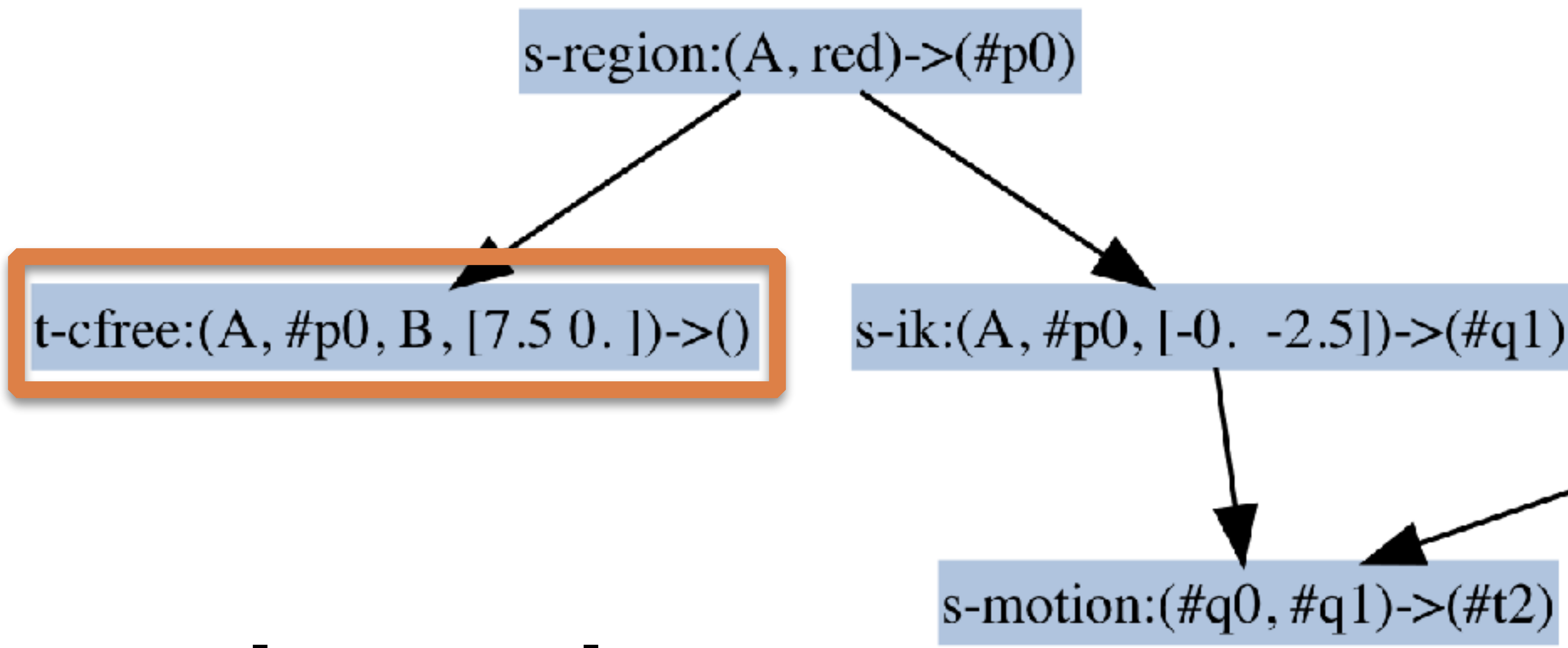


Focused: Iteration 1 - Sampling

Optimistic plan:

[move([-5. 5.],#t0,#q0), pick(**A**,[0. 0.],[0. -2.5],#q0), move(#q0, #t2, #q1), place(**A**,#p0,[0. -2.5],#q1)]





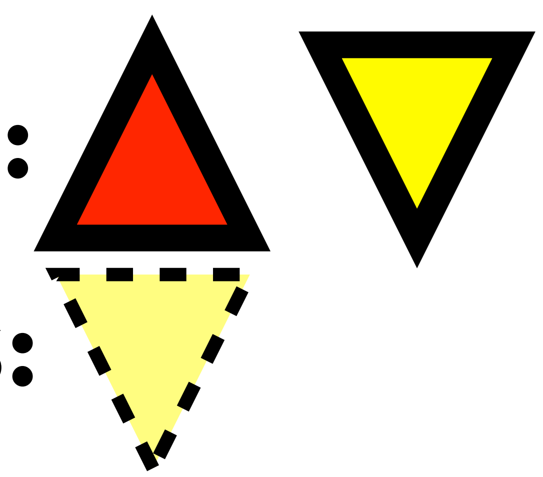
Queried streams:

1. s-region(**A**, **red**)→<u>[8.21 0.]</u>
2. s-ik(**A**, [0. 0.], [0. -2.5])→<u>[0. 2.5]</u>
3. t-cfree(**A**, [8.21 0.], **B**, [7.5 0.])→<u>False</u>

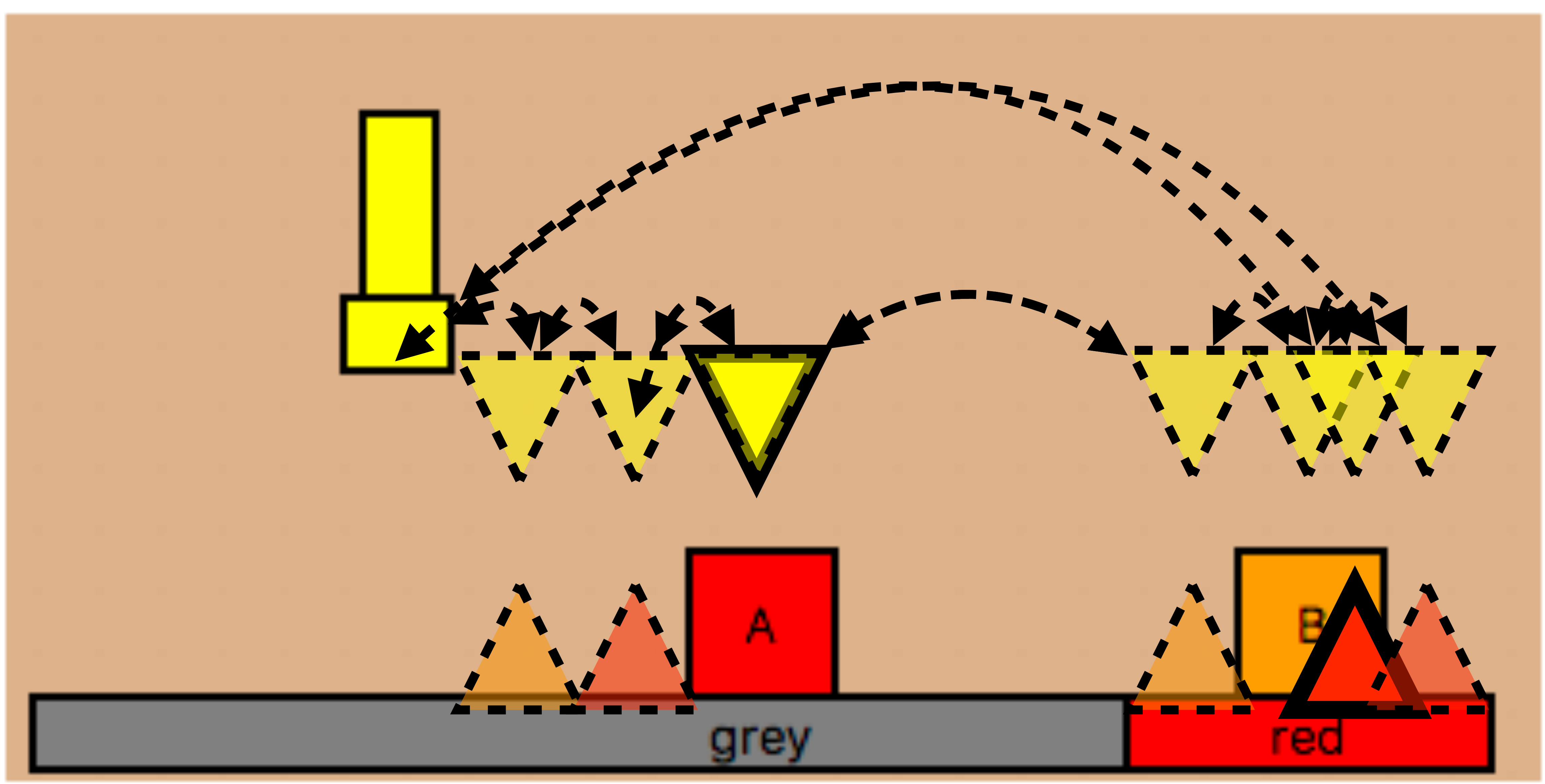
Temporarily **remove** these streams from the next search

Focused: Iteration 2

- **Iteration 2** - optimistically evaluated 42 streams
- Removed **optimistic** pose and configuration
- Added **sampled** pose and configuration:
- Added 1 **optimistic** robot configurations:



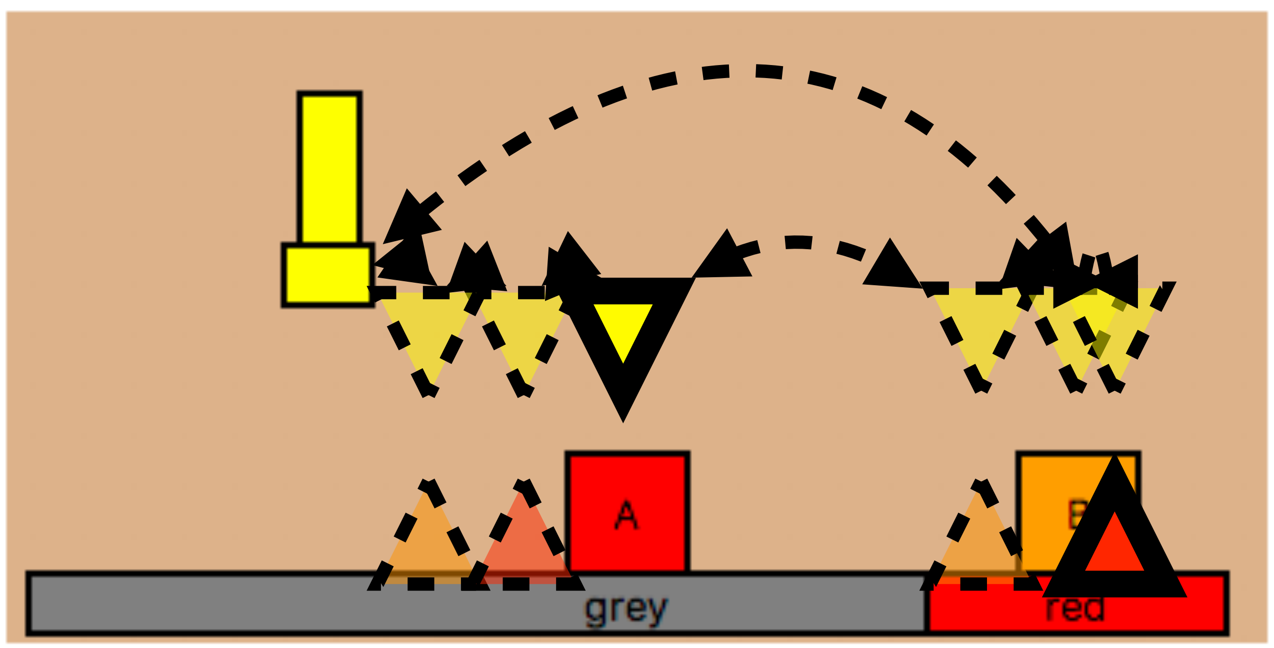
- Added 14 **optimistic** trajectories:



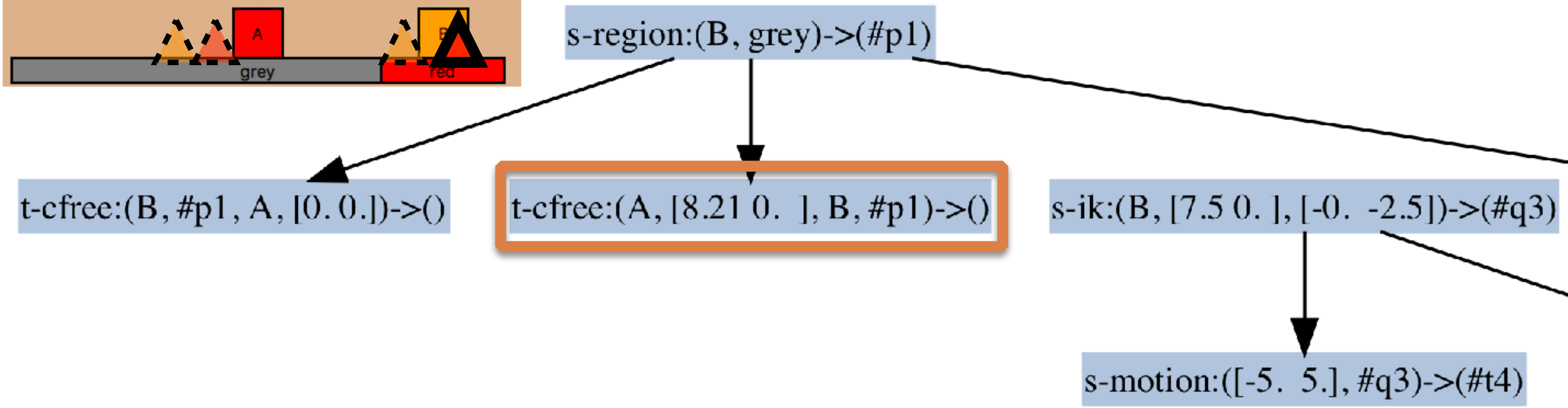
Focused: Iteration 2 - Sampling

New optimistic plan:

```
[move([-5. 5.],#t4,#q2), pick(B,[7.5 0.],[0. -2.5],#q2),
 move(#q2, #t9, #q3),    place(B, #p1, [0. -2.5], #q3),
 move(#q3,#t6,[0. 2.5]), pick(A,[0. 0.],[0. -2.5],[0. 2.5]),
 move([0. 2.5],#t8,#q4), place(A,[8.21 0.],[0. -2.5],#q4)]
```



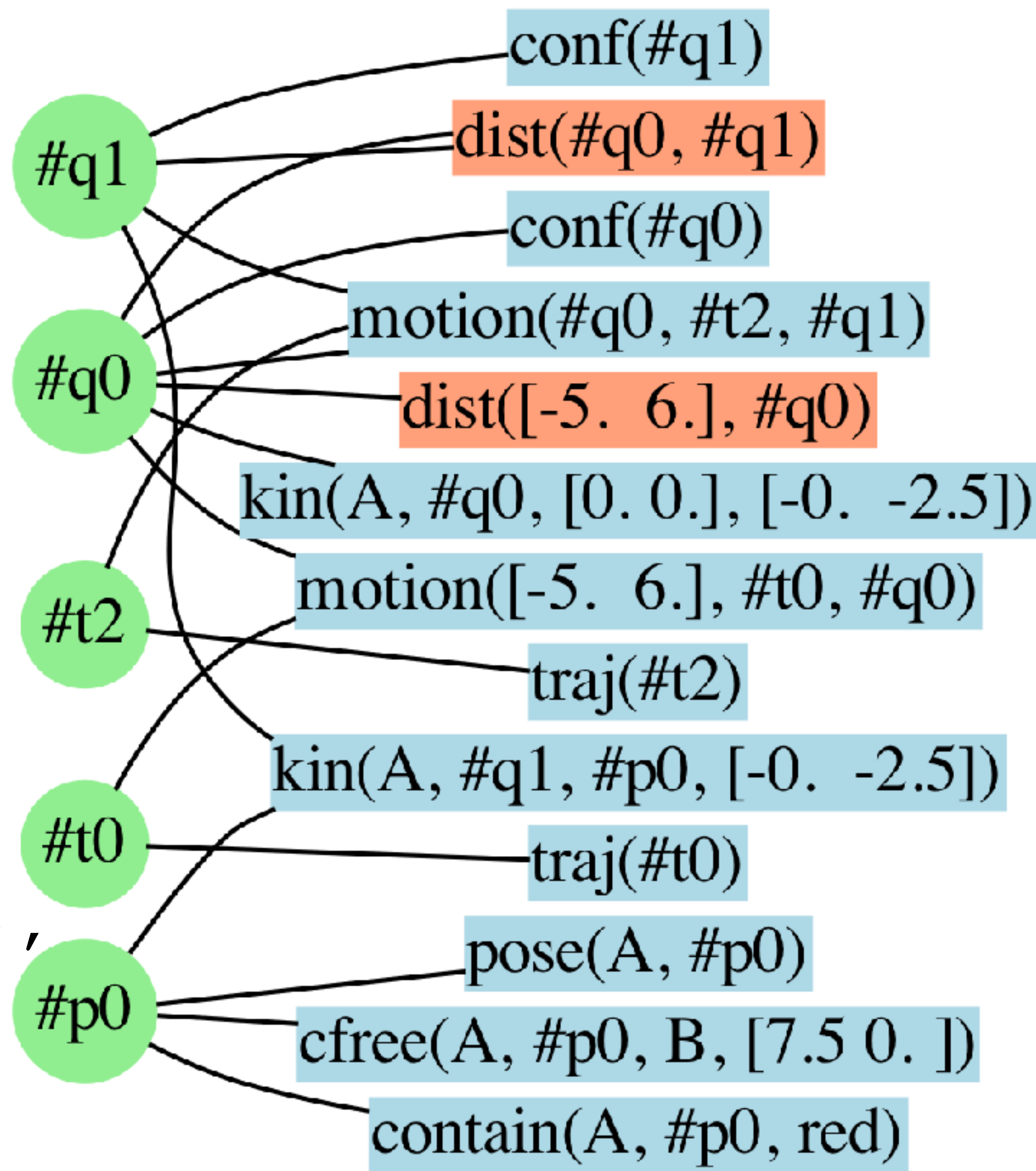
Different stream plan might succeed!



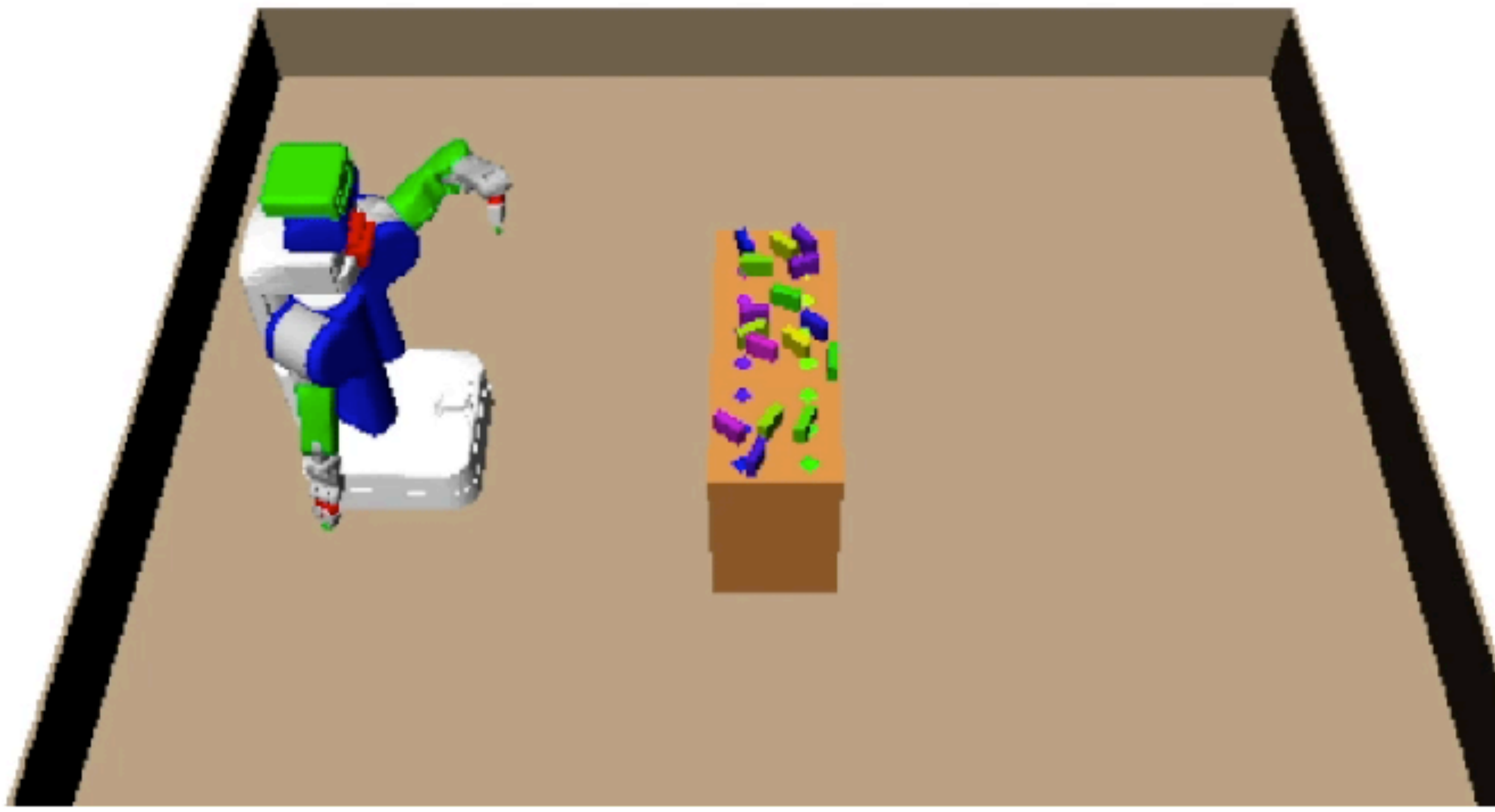
Optimistic Planning with Optimization

- Instead of sampling, directly **optimize** the constraint network
- Non-convex constrained mathematical program **solver** as a **stream**
- Additional PDDLStream algorithms…

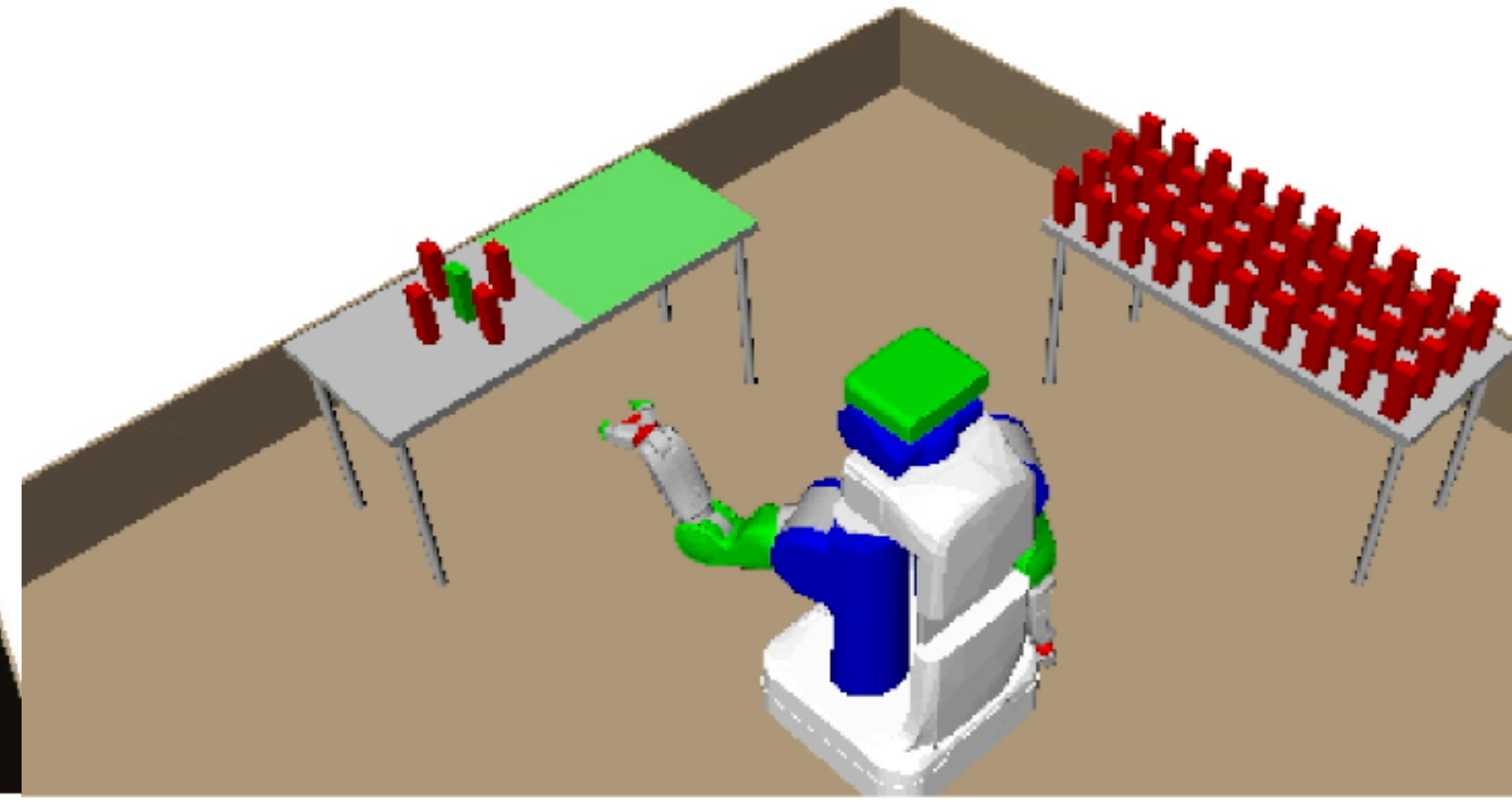
```
[move([-5. 6.], #t0, #q0),
 pick(A,[0. 0.],[0. -2.5],#q0),
 move(#q0, #t2, #q1),
 place(A,#p0,[0. -2.5],#q1)]
```



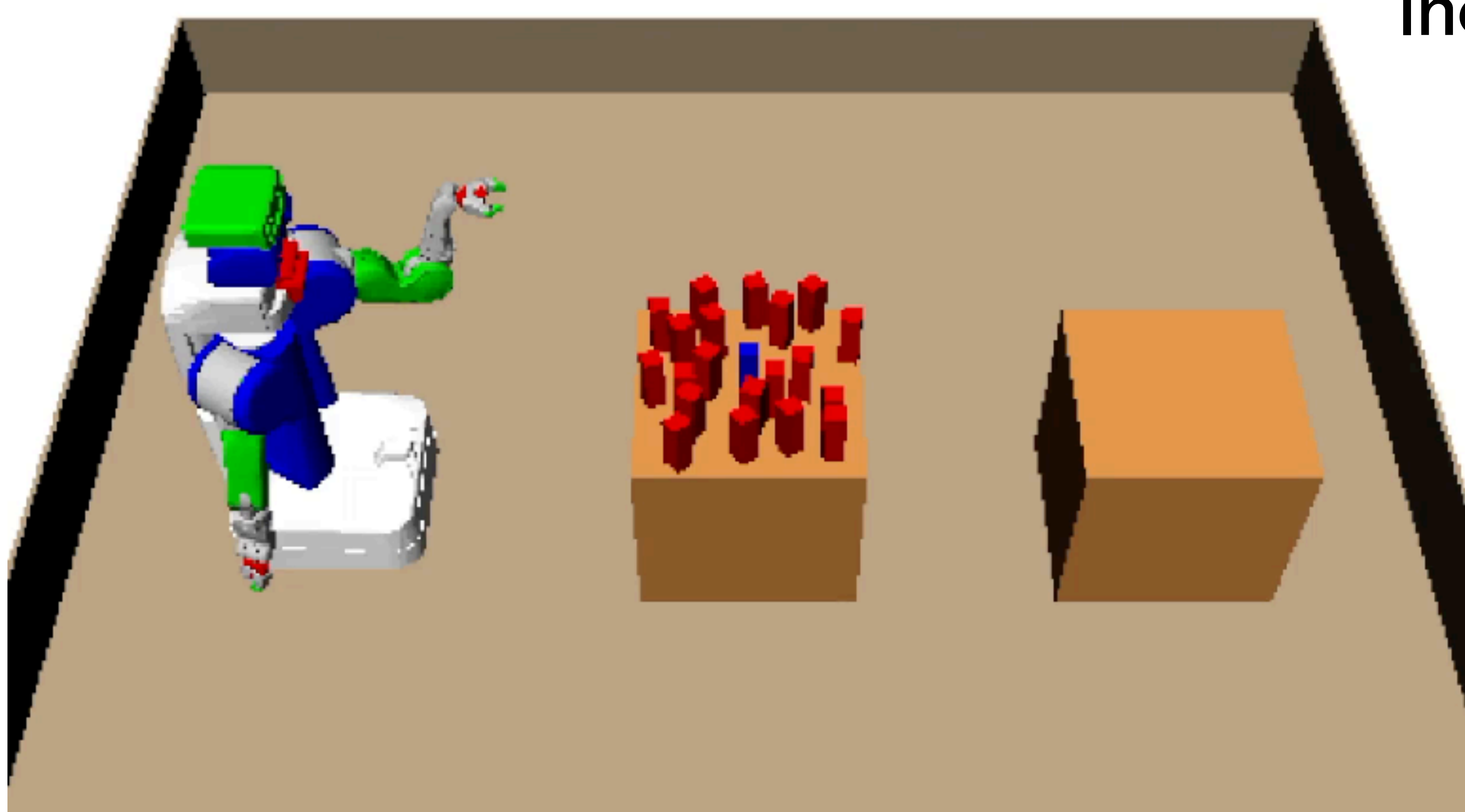
Scaling Experiments



Incremental ~20s
Focused ~10s



Incremental 120+
Focused ~25s

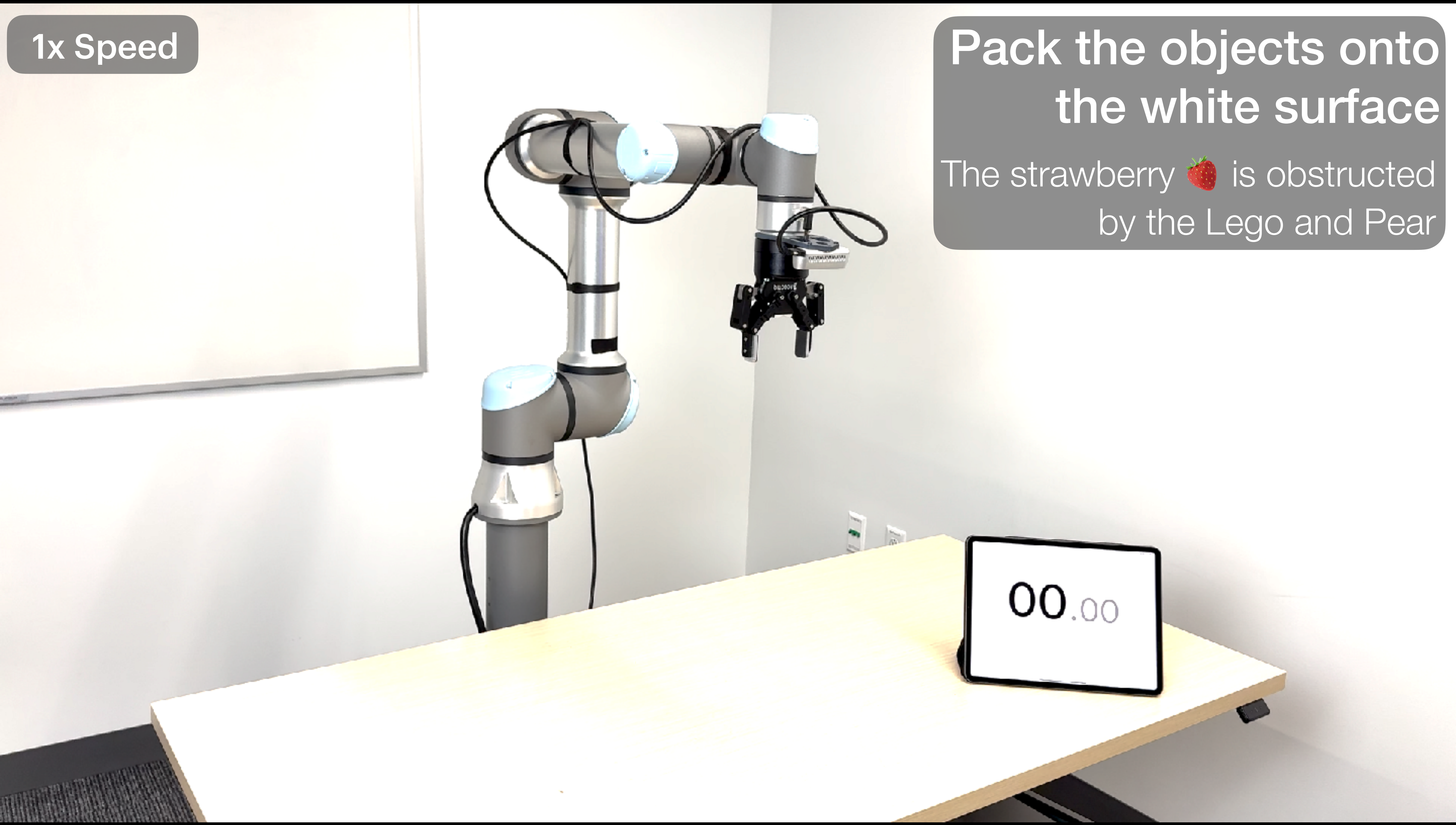


Incremental 120+
Focused ~20s

[Garrett 2018]

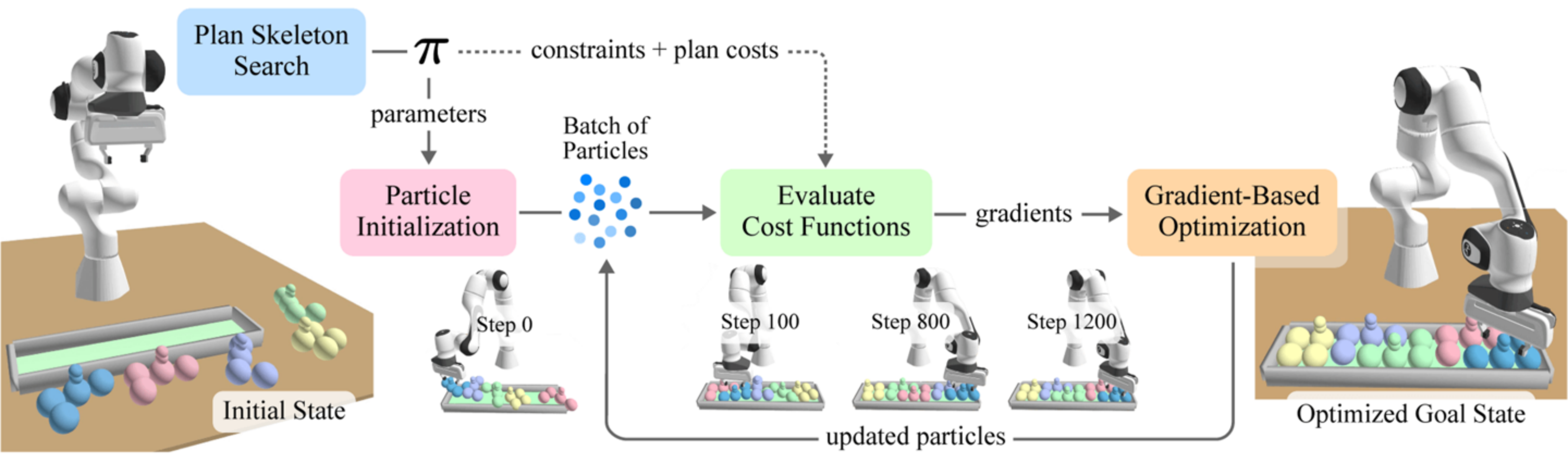
GPU-Parallelized TAMP

Differentiable GPU-Parallelized Task and Motion Planning. William Shen, Caelan Garrett, Nishanth Kumar, Ankit Goyal, Tucker Hermans, Leslie Pack Kaelbling, Tomás Lozano-Pérez, Fabio Ramos. *Robotics: Science and Systems (RSS)*, 2025.


1x Speed
Pack the objects onto the white surface
The strawberry 🍓 is obstructed by the Lego and Pear
00.00

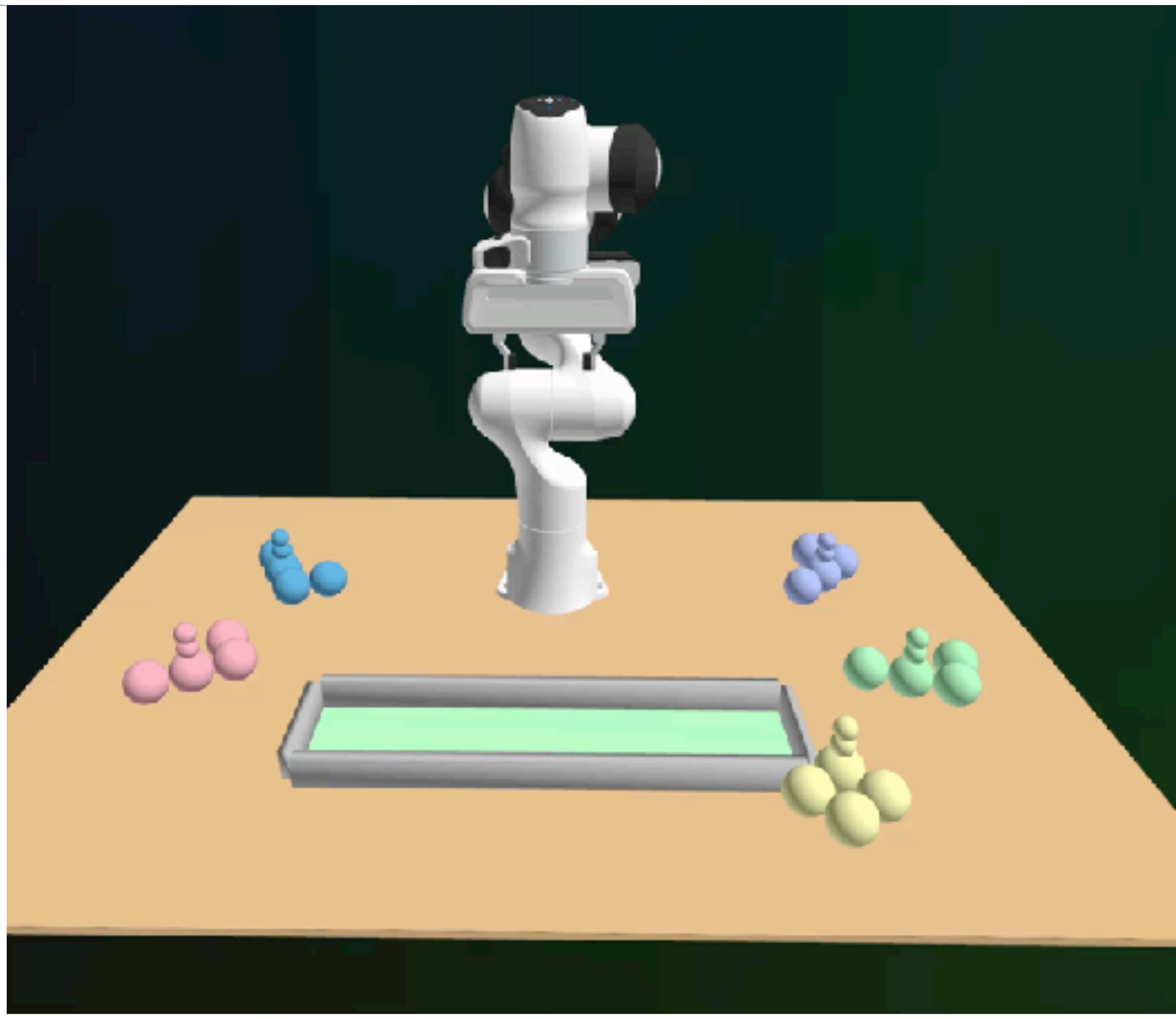
cuTAMP: GPU-Parallelized TAMP

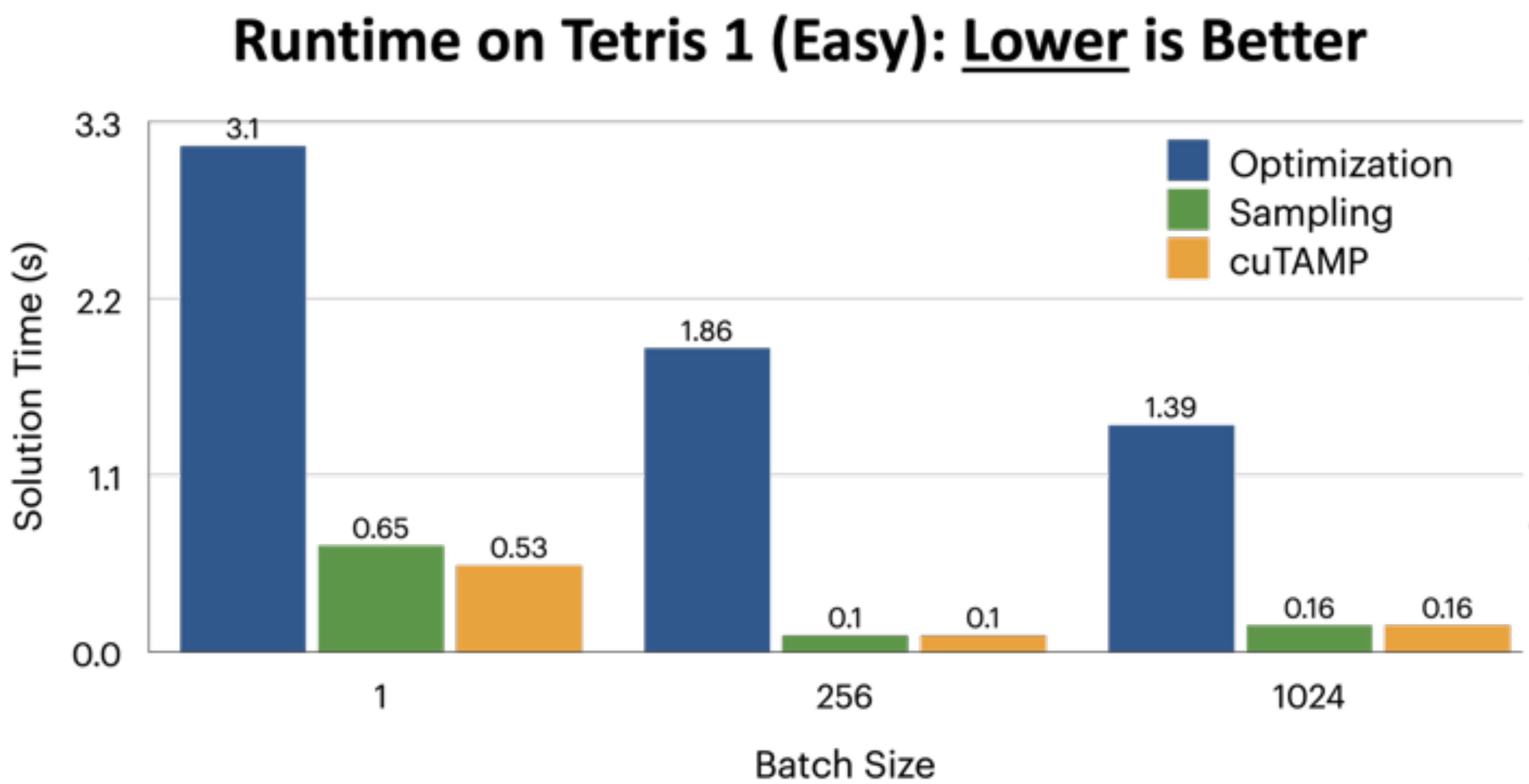
- Like Focused but combine **sampling** & **optimization**
  - Sample ~1000 candidates and optimize in batch
- Leverage **GPU acceleration** during both phases
- Custom CUDA kernels and PyTorch auto differentiation
  - Generalizes **cuRobo** motion planning to TAMP

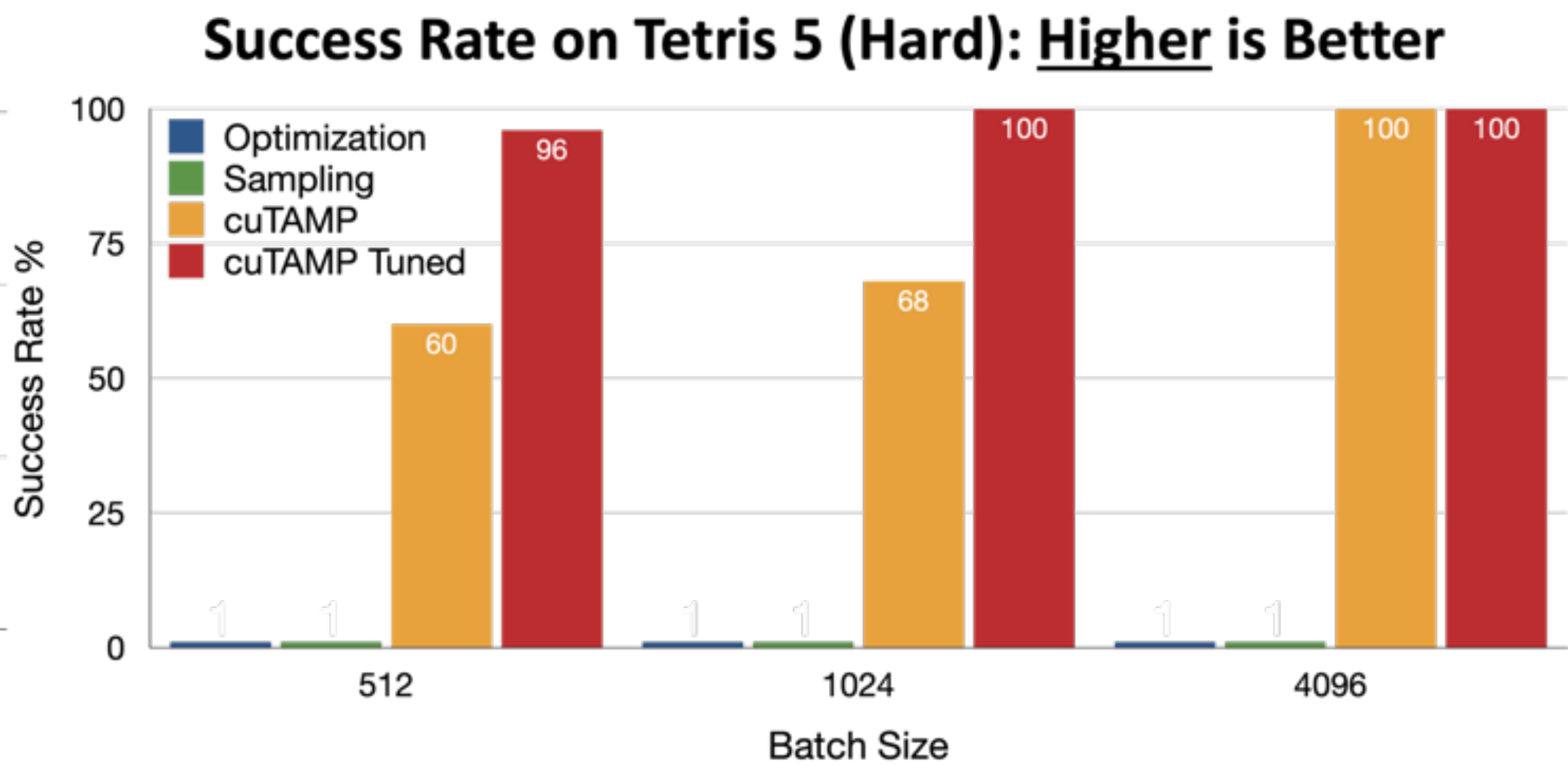


cuTAMP Experimental Results

- cuTAMP's combines **sampling** & **optimization** to outperform each individually
- Larger **GPU batch sizes** increase success rates and decrease runtime

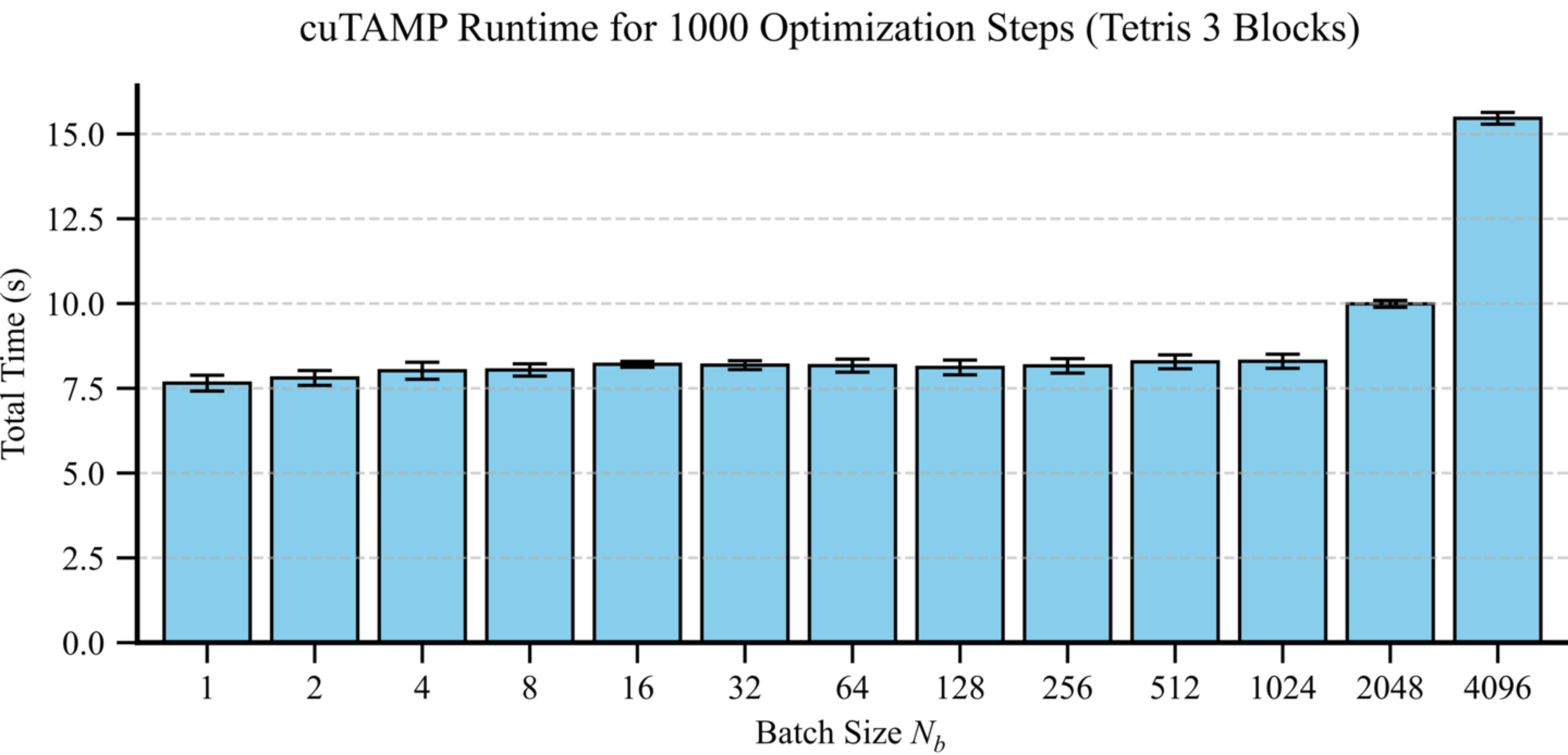


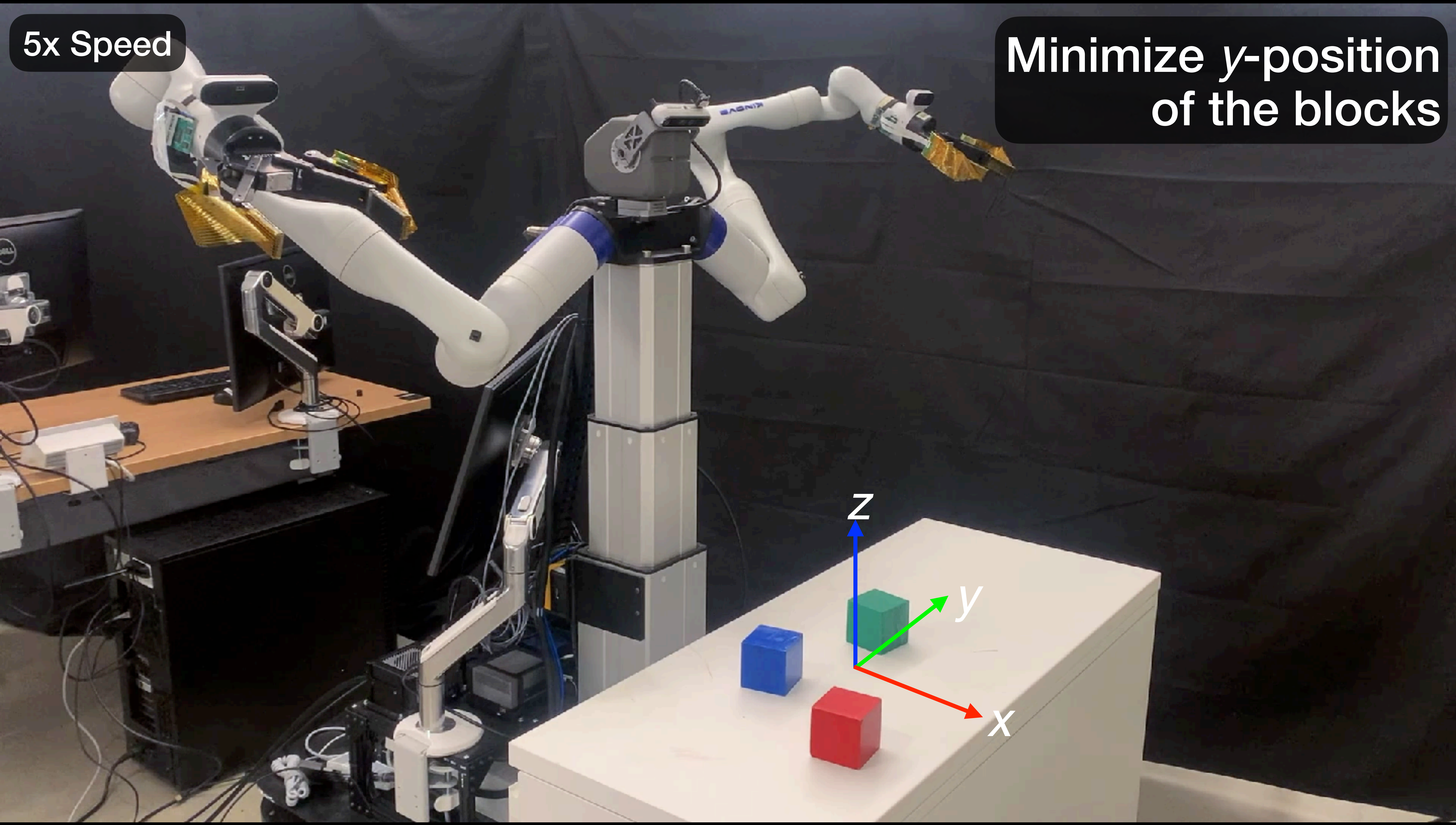




cuTAMP Scales Sublinearly*

- *For up to ~1000 Particles (GPU memory limit)




5x Speed
Minimize y-position
of the blocks
z
y
x

TAMP Under Uncertainty

Online Replanning in Belief Space for Partially Observable Task and Motion Problems. Caelan Reed Garrett, Chris Paxton, Tomás Lozano-Pérez, Leslie Pack Kaelbling, Dieter Fox. *IEEE International Conference on Robotics and Automation (ICRA)*, 2020.

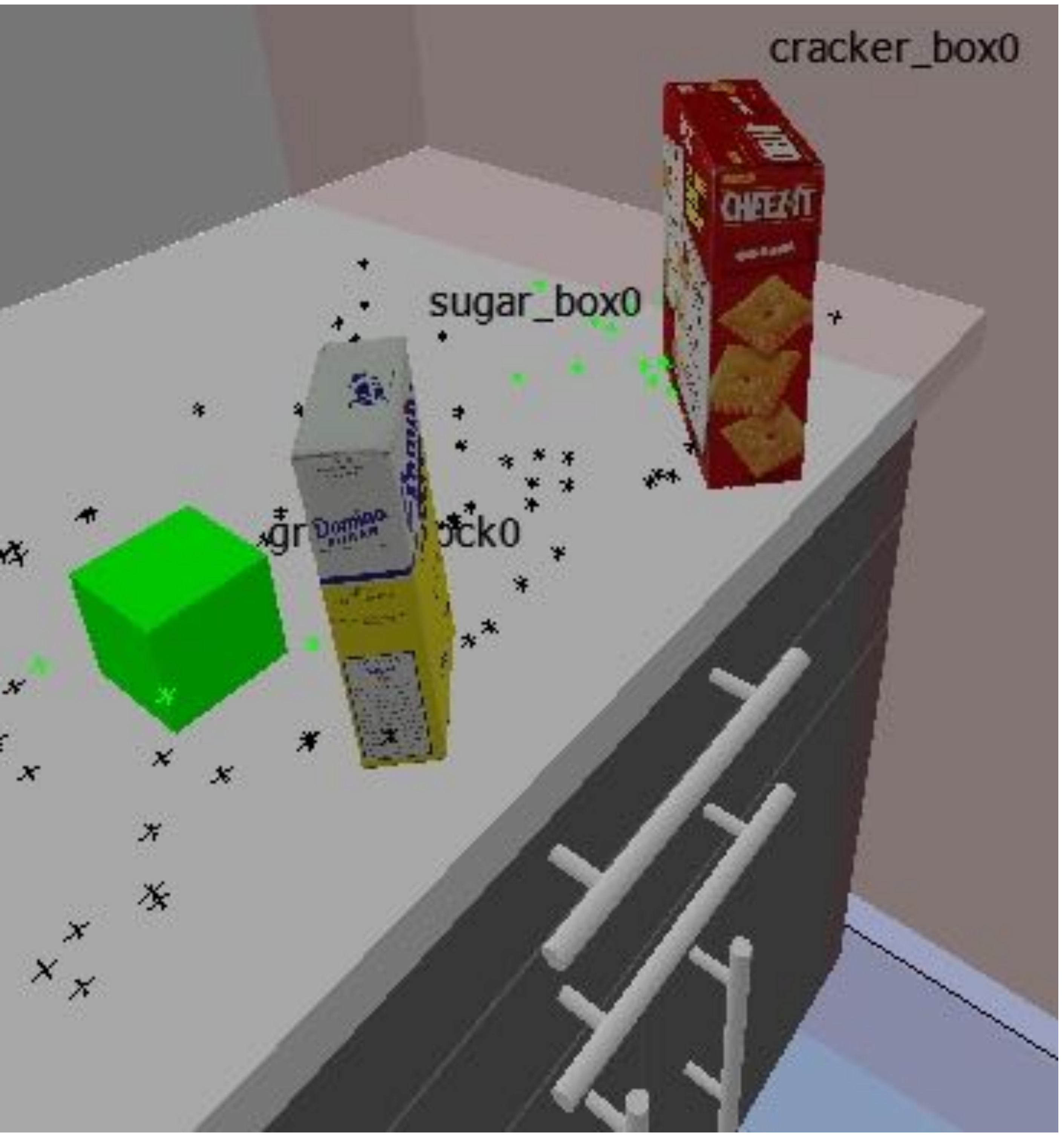
Long-Horizon Manipulation of Unknown Objects via Task and Motion Planning with Estimated Affordances. Aidan Curtis*, Xiaolin Fang*, Leslie Pack Kaelbling, Tomás Lozano-Pérez, Caelan Reed Garrett. *IEEE International Conference on Robotics and Automation (ICRA)*, 2022.

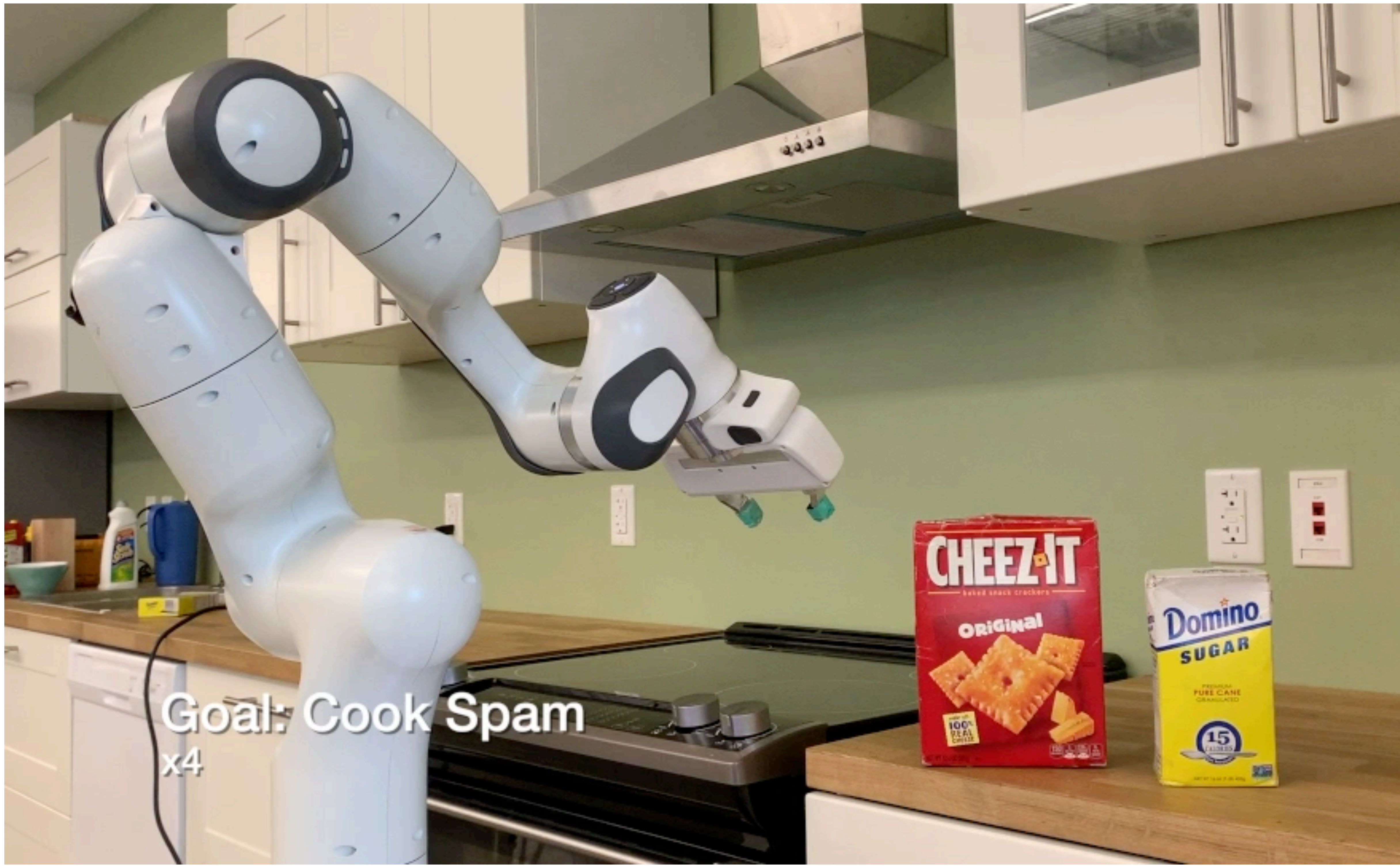
MDP: Stochastic Action Effects

- Approximate as cost-sensitive **deterministic** problem
- **Policy** evaluated online via **replanning**

POMDP: Partially Observable State

- Update a **belief** (probability distribution) over states
- Plan in **belief space** using **inference** streams & actions
- **Observation** actions reduce state variable **uncertainty**



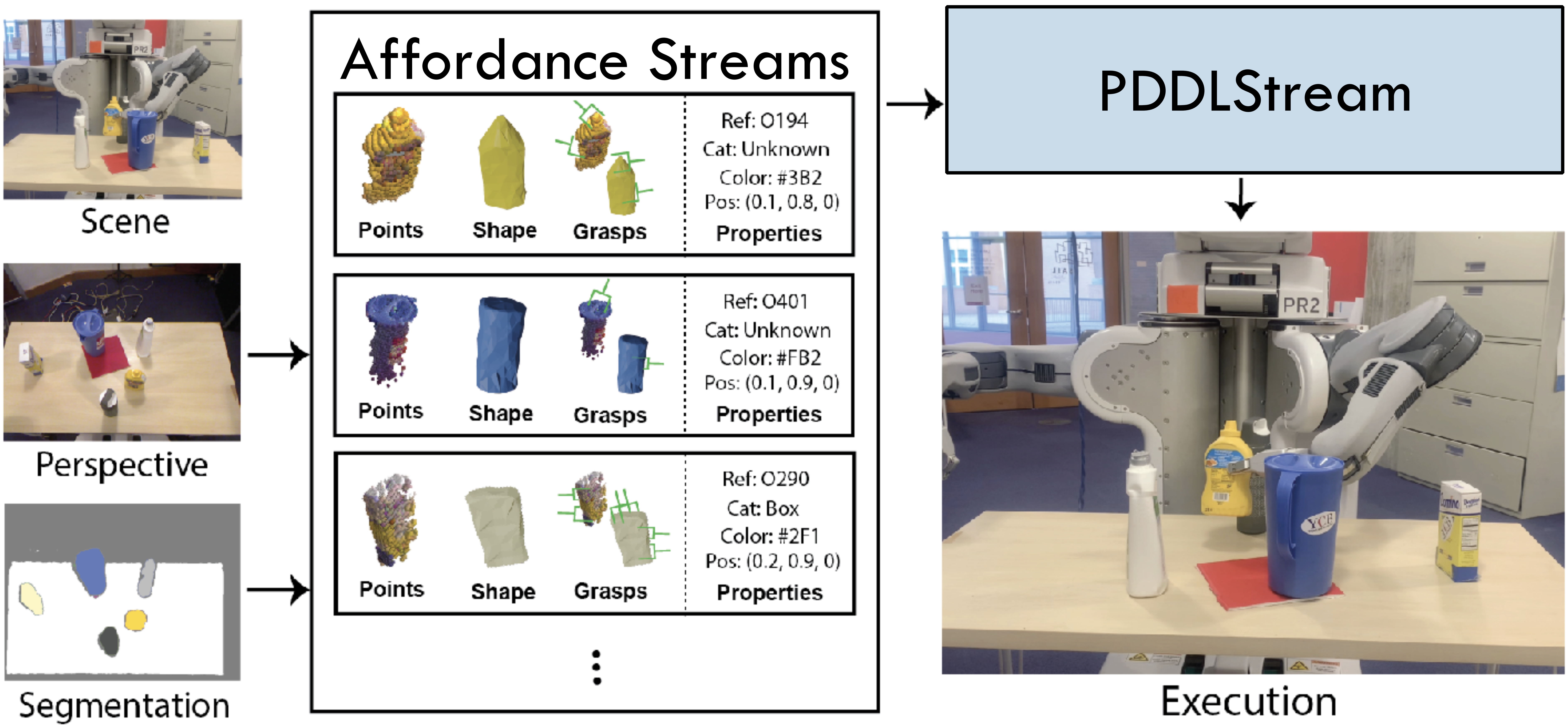


Probabilistic & Geometric Constraints

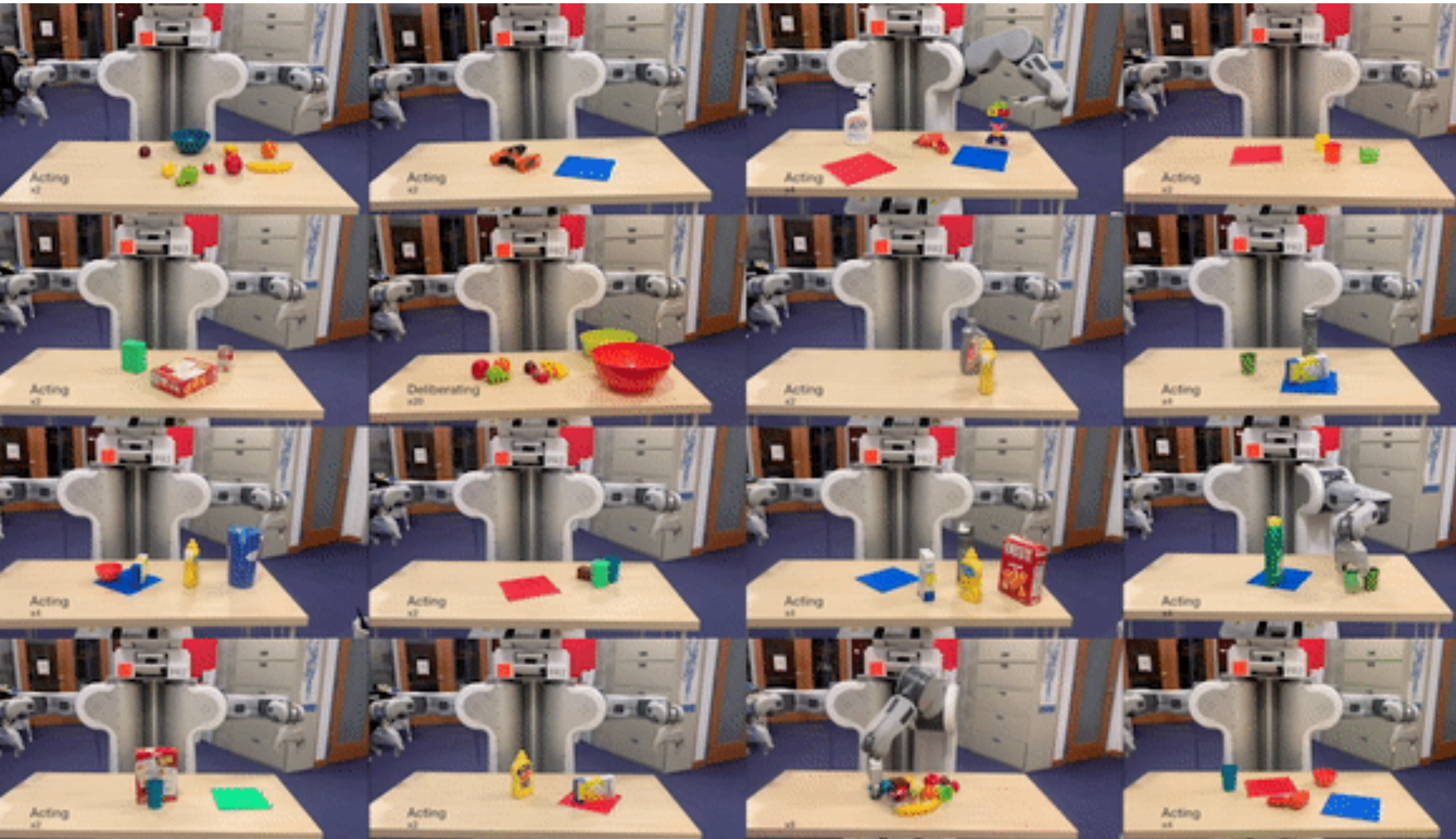
Unknown Objects via Learned Streams

Plan using Estimated Affordances

- **Learned** segmentation, grasp prediction, collision checking
- Streams call perceptual modules using object **point clouds**



Single System Generalizes Across Novel Objects, Initial States, & Goals

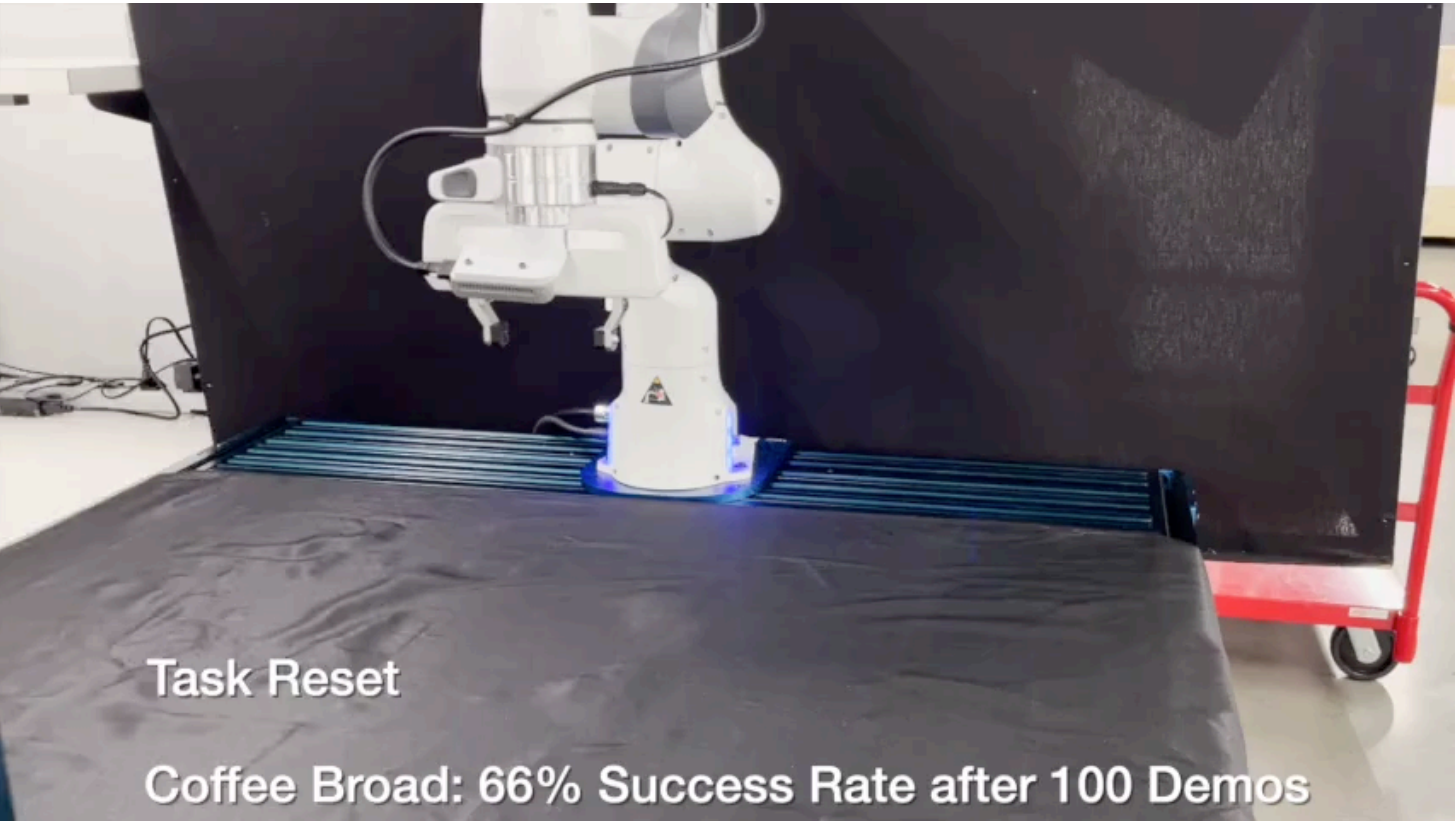


TAMP + Imitation Learning

Human-In-The-Loop Task and Motion Planning for Imitation Learning. Ajay Mandlekar*, Caelan Garrett*, Danfei Xu, Dieter Fox. *Conference on Robot Learning (CoRL)*, 2023.

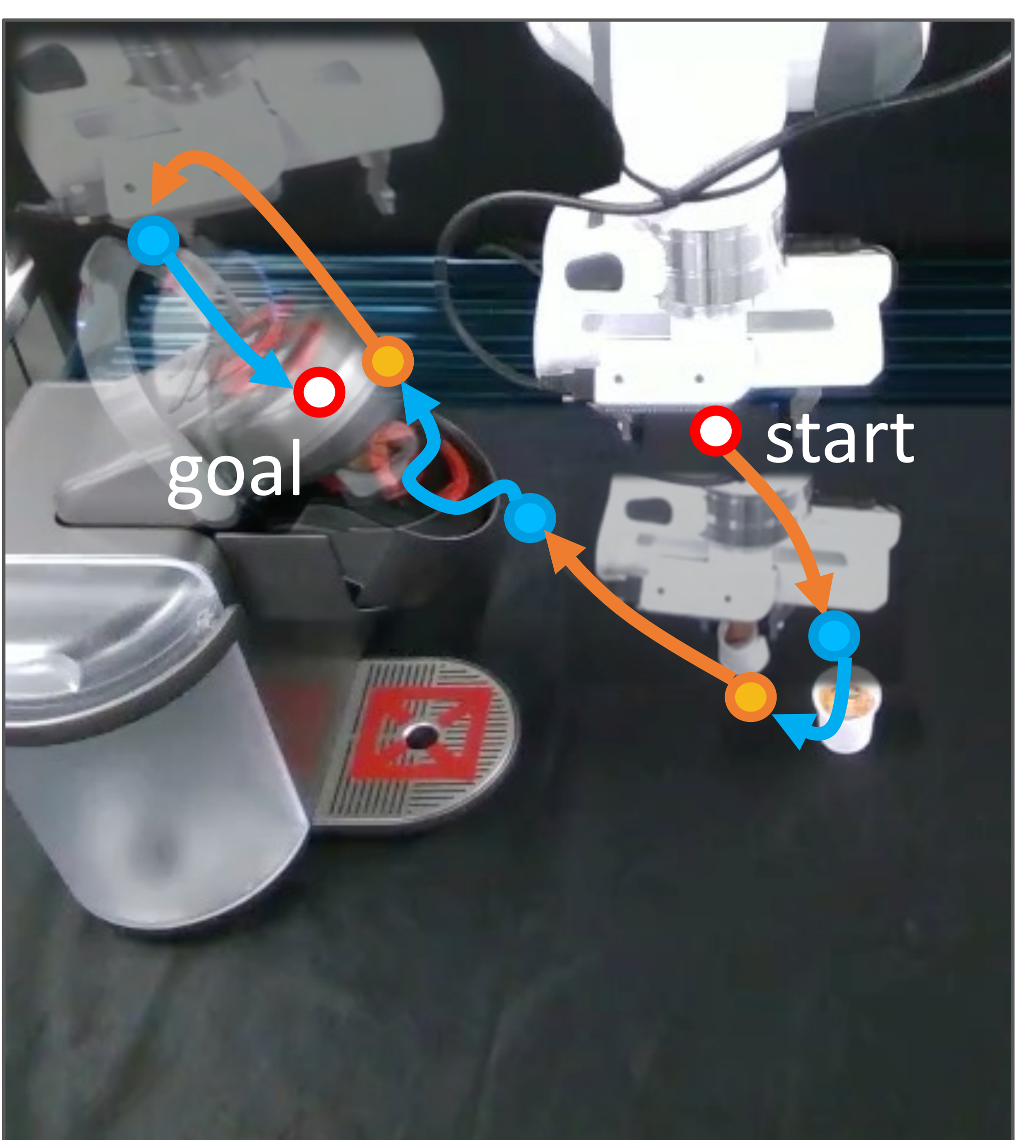
Imitating Task and Motion Planning with Visuomotor Transformers. Murtaza Dalal, Ajay Mandlekar*, Caelan Garrett*, Ankur Handa, Ruslan Salakhutdinov, Dieter Fox. *Conference on Robot Learning (CoRL)*, 2023.

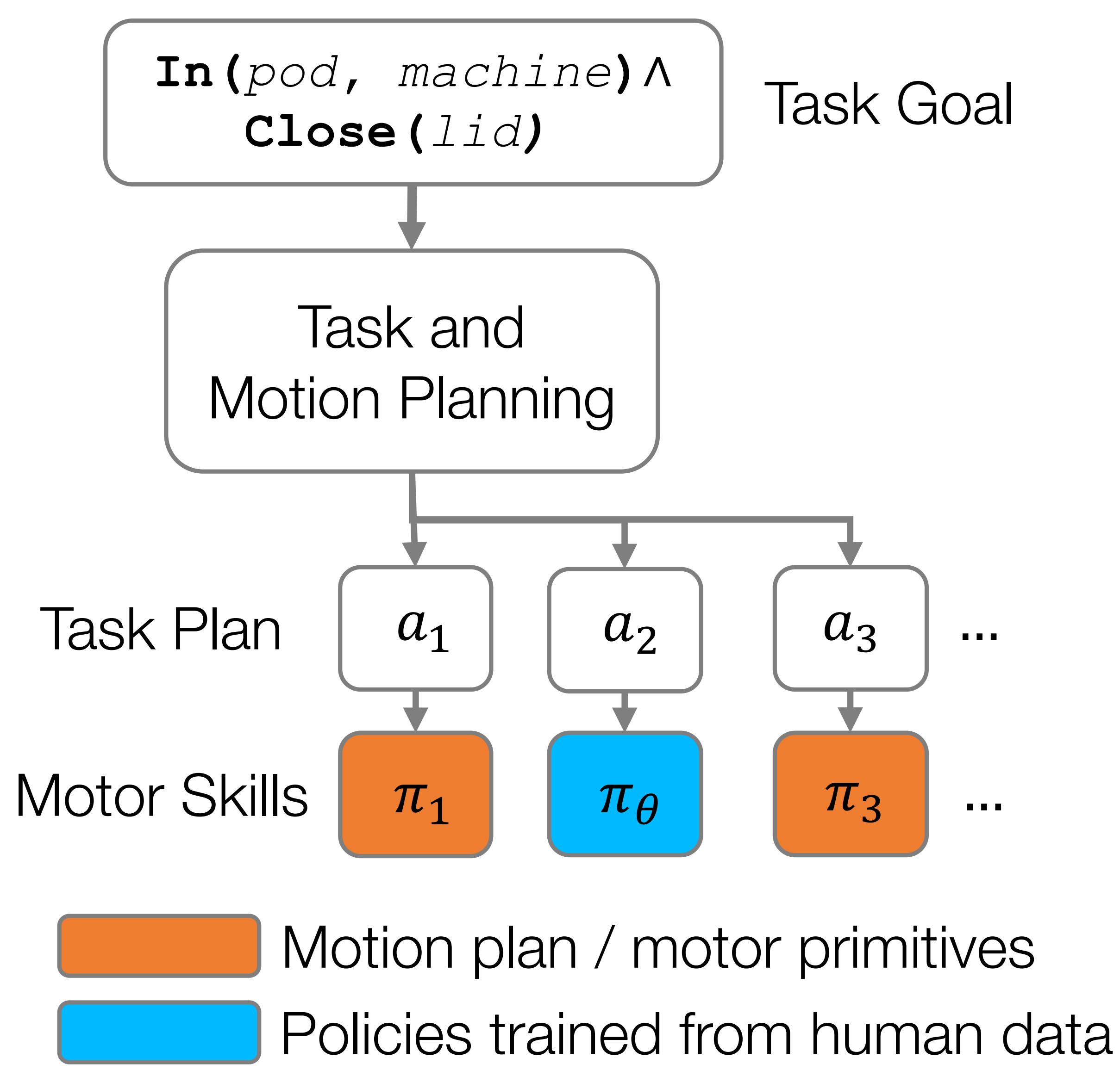
Planning with Contact-Rich Actions



Actions Learned from Human Demos

- Assume human **teleoperated** skill demonstrations
- Train image => control policy using **behavior cloning** (5/22)
- **Stochastic** actions within TAMP

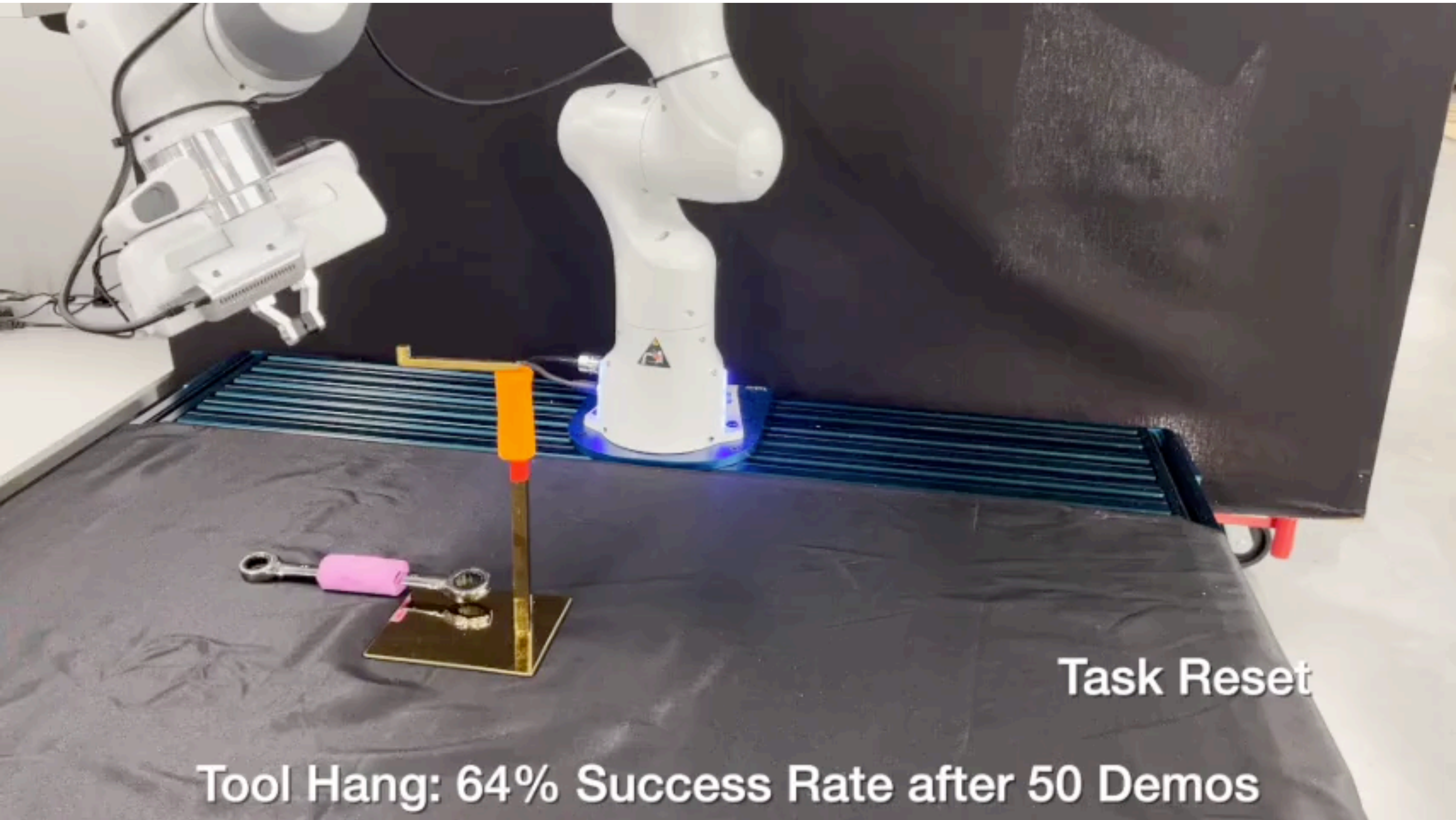




Alternating TAMP & Learned Control

- TAMP plans when to **deploy** which learned policies
- Can also use planning for **data generation** (5/22)

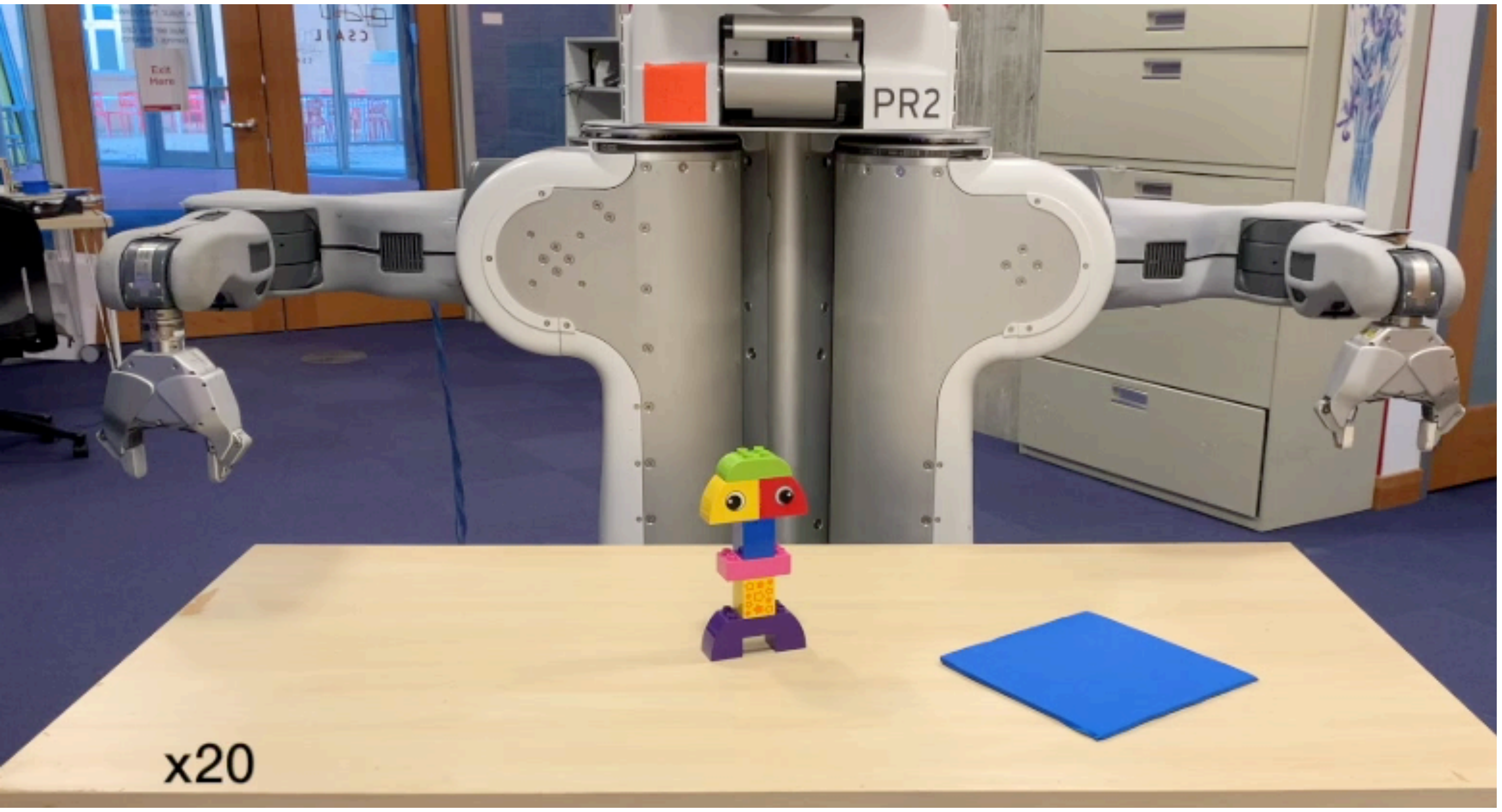
TAMP as a Learning Inductive Bias

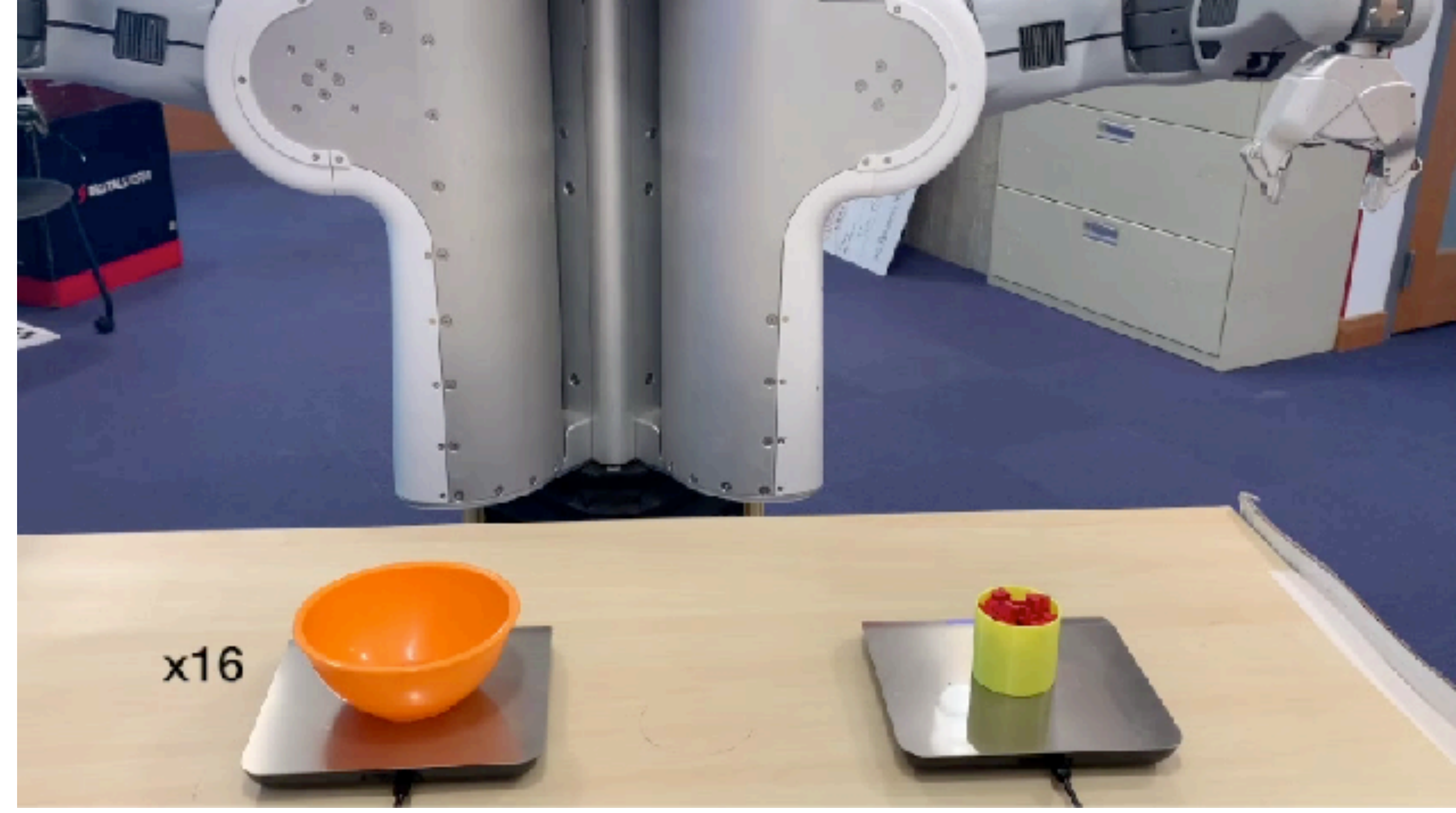


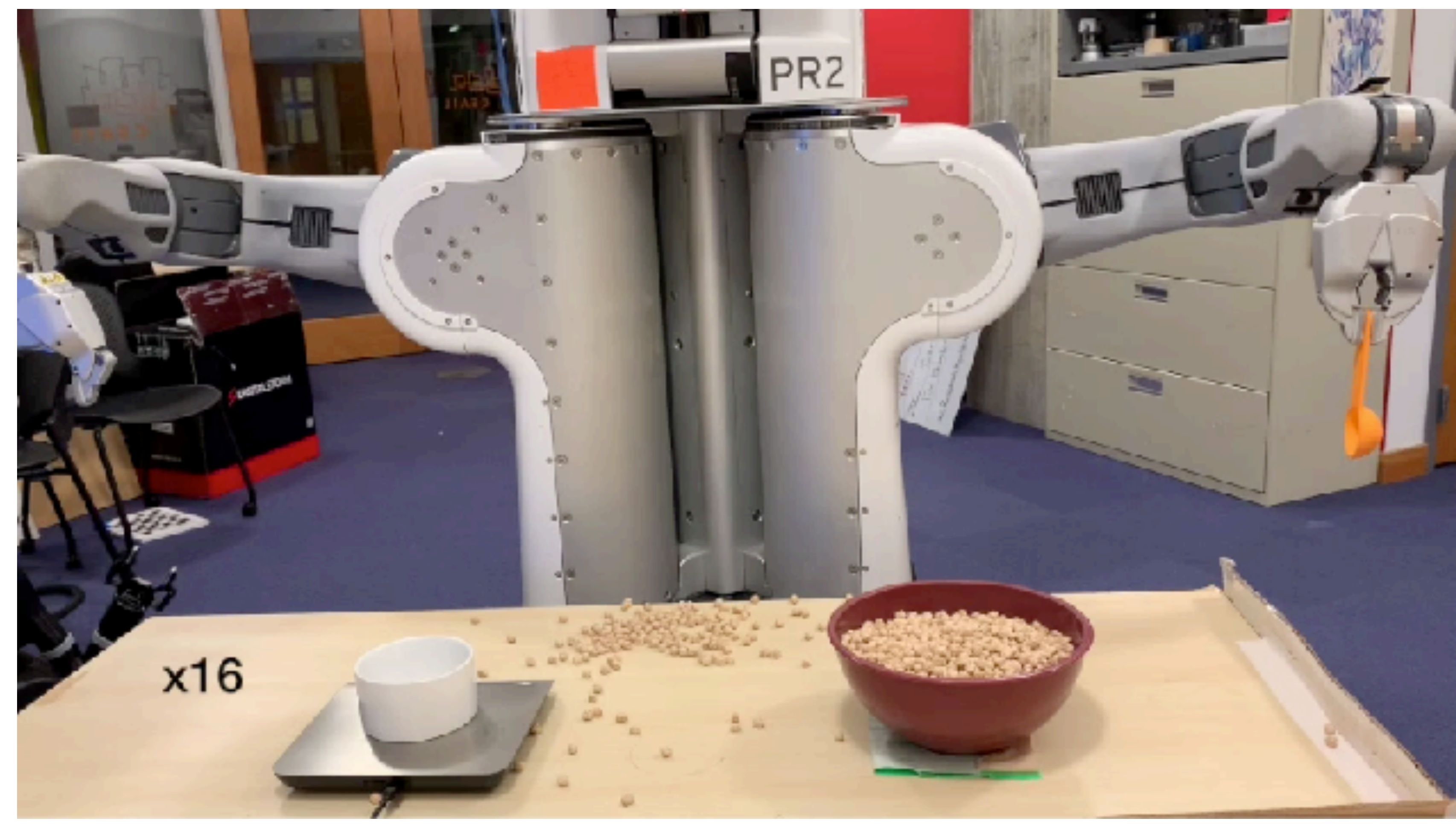
Takeaways

- **Task and Motion Planning (TAMP):** hybrid planning where continuous constraints affect discrete decisions
- **Sampling** & **optimization** for continuous satisfaction
- **PDDLStream:** planning language that supports **sampling procedures** as blackbox streams
  - **Domain-independent** algorithms
  - Efficient **lazy/optimistic** planning (focused algorithm)
- **Ongoing work:** GPU-accelerated, probabilistic & partially observable, learning-assisted TAMP

Questions?







References

Task Planning

- **[Fikes 1971]** Fikes, R.E. and Nilsson, N.J., 1971. STRIPS: A new approach to the application of theorem proving to problem solving. *Artificial intelligence*, *2*(3-4), pp.189-208.
- **[Nilsson 1984]** Nilsson, N.J., 1984. *Shakey the robot*. SRI INTERNATIONAL MENLO PARK CA.
- **[Penberthy 1992]** Penberthy, J.S. and Weld, D.S., 1992. UCPOP: A Sound, Complete, Partial Order Planner for ADL. *Kr*, *92*, pp.103-114.
- **[Aeronautiques 1998]** Aeronautiques, C., Howe, A., Knoblock, C., McDermott, I.D., Ram, A., Veloso, M., Weld, D., SRI, D.W., Barrett, A., Christianson, D. and Friedman, M., 1998. PDDL| The Planning Domain Definition Language.
- **[Kautz 1999]** Kautz, H. and Selman, B., 1999, June. Unifying SAT-based and graph-based planning. In *IJCAI* (Vol. 99, pp. 318-325).
- **[Bonet 2001]** Bonet, B. and Geffner, H., 2001. Planning as heuristic search. *Artificial Intelligence*, *129*(1-2), pp.5-33.
- **[Hoffman 2001]** Hoffmann, J. and Nebel, B., 2001. The FF planning system: Fast plan generation through heuristic search. *Journal of Artificial Intelligence Research*, *14*, pp.253-302.
- **[Ghallab 2004]** Ghallab, M., Nau, D. and Traverso, P., 2004. *Automated Planning: theory and practice*. Elsevier.
- **[Thiébaux 2005]** Thiébaux, S., Hoffmann, J. and Nebel, B., 2005. In defense of PDDL axioms. *Artificial Intelligence*, *168*(1-2), pp.38-69.
- **[Helmert 2006]** Helmert, M., 2006. The fast downward planning system. *Journal of Artificial Intelligence Research*, *26*, pp.191-246.

Motion Planning

- **[Lozano-Pérez 1979]** Lozano-Pérez, T. and Wesley, M.A., 1979. An algorithm for planning collision-free paths among polyhedral obstacles. *Communications of the ACM*, *22*(10), pp.560-570.
- **[Kavraki 1994]** Kavraki, L., Svestka, P. and Overmars, M.H., 1994. Probabilistic roadmaps for path planning in high-dimensional configuration spaces (Vol. 1994).
- **[Bohlin 2000]** Bohlin, R. and Kavraki, L.E., 2000, April. Path planning using lazy PRM. In Proceedings 2000 ICRA. Millennium Conference. IEEE International Conference on Robotics and Automation. Symposia Proceedings (Cat. No. 00CH37065) (Vol. 1, pp. 521-528). IEEE.
- **[Kuffner 2000]** Kuffner Jr, J.J. and LaValle, S.M., 2000, April. RRT-connect: An efficient approach to single-query path planning. In *ICRA* (Vol. 2).
- **[Kuffner 2001]** LaValle, S.M. and Kuffner Jr, J.J., 2001. Randomized kinodynamic planning. *The International Journal of Robotics Research*, *20*(5), pp.378-400.
- **[LaValle 2006]** LaValle, S.M., 2006. *Planning algorithms*. Cambridge university press.
- **[Ratliff 2009]** Ratliff, N., Zucker, M., Bagnell, J.A. and Srinivasa, S., 2009. CHOMP: Gradient optimization techniques for efficient motion planning.
- **[Schulman 2013]** Schulman, J., Ho, J., Lee, A.X., Awwal, I., Bradlow, H. and Abbeel, P., 2013, June. Finding Locally Optimal, Collision-Free Trajectories with Sequential Convex Optimization. In *Robotics: science and systems* (Vol. 9, No. 1, pp. 1-10).
- **[Dellin 2016]** Dellin, C.M. and Srinivasa, S.S., 2016, March. A unifying formalism for shortest path problems with expensive edge evaluations via lazy best-first search over paths with edge selectors. In *Twenty-Sixth International Conference on Automated Planning and Scheduling*.

Prediscretized Planning

- **[Dornhege 2009]** Dornhege, C., Eyerich, P., Keller, T., Trüg, S., Brenner, M. and Nebel, B., 2009, October. Semantic attachments for domain-independent planning systems. In *Nineteenth International Conference on Automated Planning and Scheduling*.
- **[Erdem 2011]** Erdem, E., Haspalamutgil, K., Palaz, C., Patoglu, V. and Uras, T., 2011, May. Combining high-level causal reasoning with low-level geometric reasoning and motion planning for robotic manipulation. In *2011 IEEE International Conference on Robotics and Automation* (pp. 4575-4581). IEEE.
- **[Lagriffoul 2014]** Lagriffoul, F., Dimitrov, D., Bidot, J., Saffiotti, A. and Karlsson, L., 2014. Efficiently combining task and motion planning using geometric constraints. *The International Journal of Robotics Research*, *33*(14), pp.1726-1747.
- **[Lozano-Pérez 2014]** Lozano-Pérez, T. and Kaelbling, L.P., 2014, September. A constraint-based method for solving sequential manipulation planning problems. In *2014 IEEE/RSJ International Conference on Intelligent Robots and Systems* (pp. 3684-3691). IEEE.
- **[Garrett 2017]** Garrett, C.R., Lozano-Perez, T. and Kaelbling, L.P., 2017. FFRob: Leveraging symbolic planning for efficient task and motion planning. *The International Journal of Robotics Research*, *37*(1), pp.104-136.
- **[Ferrer-Mestres 2017]** Ferrer-Mestres, J., Frances, G. and Geffner, H., 2017. Combined task and motion planning as classical AI planning. *arXiv preprint arXiv:1706.06927*.
- **[Dantam 2018]** Dantam, N.T., Kingston, Z.K., Chaudhuri, S. and Kavraki, L.E., 2018. An incremental constraint-based framework for task and motion planning. *The International Journal of Robotics Research*, *37*(10), pp.1134-1151.
- **[Lo 2018]** Lo, S.Y., Zhang, S. and Stone, P., 2018, July. PETLON: Planning Efficiently for Task-Level-Optimal Navigation. In *Proceedings of the 17th International Conference on Autonomous Agents and MultiAgent Systems* (pp. 220-228). International Foundation for Autonomous Agents and Multiagent Systems.
- **[Huang 2018]** Huang, Y., Garrett, C.R. and Mueller, C.T., 2018. Automated sequence and motion planning for robotic spatial extrusion of 3D trusses. *Construction Robotics*, *2*(1-4), pp.15-39.

Numeric Planning

- **[Fox 2003]** Fox, M. and Long, D., 2003. PDDL2. 1: An extension to PDDL for expressing temporal planning domains. *Journal of artificial intelligence research*, *20*, pp.61-124.
- **[Hoffmann 2003]** Hoffmann, J., 2003. The Metric-FF Planning System: Translating``Ignoring Delete Lists''to Numeric State Variables. *Journal of artificial intelligence research*, *20*, pp.291-341.
- **[Eyerich 2009]** Eyerich, P., Mattmüller, R. and Röger, G., 2009, October. Using the context-enhanced additive heuristic for temporal and numeric planning. In *Nineteenth International Conference on Automated Planning and Scheduling*.
- **[Deits 2015]** Deits, R. and Tedrake, R., 2015, May. Efficient mixed-integer planning for UAVs in cluttered environments. In *2015 IEEE international conference on robotics and automation (ICRA)* (pp. 42-49). IEEE.
- **[Shoukry 2016]** Shoukry, Y., Nuzzo, P., Saha, I., Sangiovanni-Vincentelli, A.L., Seshia, S.A., Pappas, G.J. and Tabuada, P., 2016, December. Scalable lazy SMT-based motion planning. In *2016 IEEE 55th Conference on Decision and Control (CDC)* (pp. 6683-6688). IEEE.
- **[Fernandez-Gonzalez 2018]** Fernandez-Gonzalez, E., Williams, B. and Karpas, E., 2018. ScottyActivity: Mixed Discrete-Continuous Planning with Convex Optimization. *Journal of Artificial Intelligence Research*, *62*, pp.579-664.

Multi-Modal Motion Planning

- **[Alami 1994]** Alami, R., Laumond, J.P. and Siméon, T., 1994. Two manipulation planning algorithms. In *WAFR Proceedings of the workshop on Algorithmic foundations of robotics* (pp. 109-125). AK Peters, Ltd. Natick, MA, USA.
- **[Siméon 2004]** Siméon, T., Laumond, J.P., Cortés, J. and Sahbani, A., 2004. Manipulation planning with probabilistic roadmaps. *The International Journal of Robotics Research*, *23*(7-8), pp.729-746.
- **[Hauser 2011]** Hauser, K. and Ng-Thow-Hing, V., 2011. Randomized multi-modal motion planning for a humanoid robot manipulation task. *The International Journal of Robotics Research*, *30*(6), pp.678-698.
- **[Barry 2013]** Barry, J., Kaelbling, L.P. and Lozano-Pérez, T., 2013, May. A hierarchical approach to manipulation with diverse actions. In *2013 IEEE International Conference on Robotics and Automation* (pp. 1799-1806). IEEE.
- **[Toussaint 2015]** Toussaint, M., 2015, June. Logic-geometric programming: An optimization-based approach to combined task and motion planning. In *Twenty-Fourth International Joint Conference on Artificial Intelligence*.
- **[Vega-Brown 2016]** Vega-Brown, W. and Roy, N., 2016, December. Asymptotically optimal planning under piecewise-analytic constraints. In *Workshop on the Algorithmic Foundations of Robotics*.
- **[Toussaint 2018]** Toussaint, M., Allen, K., Smith, K.A. and Tenenbaum, J.B., 2018. Differentiable Physics and Stable Modes for Tool-Use and Manipulation Planning. In *Robotics: Science and Systems*.

Task and Motion Planning

- **[Gravot 2005]** Gravot, F., Cambon, S. and Alami, R., 2005. aSyMov: a planner that deals with intricate symbolic and geometric problems. In *Robotics Research. The Eleventh International Symposium* (pp. 100-110). Springer, Berlin, Heidelberg.
- **[Plaku 2010]** Plaku, E. and Hager, G.D., 2010, May. Sampling-based motion and symbolic action planning with geometric and differential constraints. In *2010 IEEE International Conference on Robotics and Automation* (pp. 5002-5008). IEEE.
- **[Kaelbling 2011]** Kaelbling, L. P. and Lozano-Pérez, T. Hierarchical task and motion planning in the now. *2011 IEEE International Conference on Robotics and Automation*, Shanghai, 2011, pp. 1470-1477.
- **[De Silva 2013]** De Silva, L., Pandey, A.K., Gharbi, M. and Alami, R., 2013. Towards combining HTN planning and geometric task planning. *arXiv preprint arXiv:1307.1482*.
- **[Srivastava 2014]** Srivastava, S., Fang, E., Riano, L., Chitnis, R., Russell, S. and Abbeel, P., 2014, May. Combined task and motion planning through an extensible planner-independent interface layer. In *2014 IEEE international conference on robotics and automation (ICRA)* (pp. 639-646). IEEE.
- **[Garrett 2018a]** Garrett, C.R., Lozano-Pérez, T. and Kaelbling, L.P., 2018. Sampling-based methods for factored task and motion planning. *The International Journal of Robotics Research*, *37*(13-14), pp.1796-1825.
- **[Garrett 2018b]** Garrett, C.R., Lozano-Pérez, T. and Kaelbling, L.P., 2018. STRIPStream: Integrating Symbolic Planners and Blackbox Samplers. *arXiv preprint arXiv:1802.08705*.

Probabilistic & Partially-Observable

- **[Kaelbling 1998]** Kaelbling, L.P., Littman, M.L. and Cassandra, A.R., 1998. Planning and acting in partially observable stochastic domains. *Artificial intelligence*, *101*(1-2), pp.99-134.
- **[Kocsis 2006]** Kocsis, L. and Szepesvári, C., 2006, September. Bandit based monte-carlo planning. In *European conference on machine learning* (pp. 282-293). Springer, Berlin, Heidelberg.
- **[Yoon 2007]** Yoon, S.W., Fern, A. and Givan, R., 2007, September. FF-Replan: A Baseline for Probabilistic Planning. In *ICAPS* (Vol. *7*, pp. 352-359).
- **[Silver 2010]** Silver, D. and Veness, J., 2010. Monte-Carlo planning in large POMDPs. In *Advances in neural information processing systems* (pp. 2164-2172).
- **[Platt 2010]** Platt Jr, R., Tedrake, R., Kaelbling, L. and Lozano-Perez, T., 2010. Belief space planning assuming maximum likelihood observations.
- **[Kaelbling 2013]** Kaelbling, L.P. and Lozano-Pérez, T., 2013. Integrated task and motion planning in belief space. *The International Journal of Robotics Research*, *32*(9-10), pp.1194-1227.
- **[Hadfield-Menell 2015]** Hadfield-Menell, D., Groshev, E., Chitnis, R. and Abbeel, P., 2015, September. Modular task and motion planning in belief space. In *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)* (pp. 4991-4998). IEEE.