Learning in Search-based Planning for Robotics

Maxim Likhachev

Associate Professor

Carnegie Mellon University and Waymo

Search-based Planning Lab (SBPL)

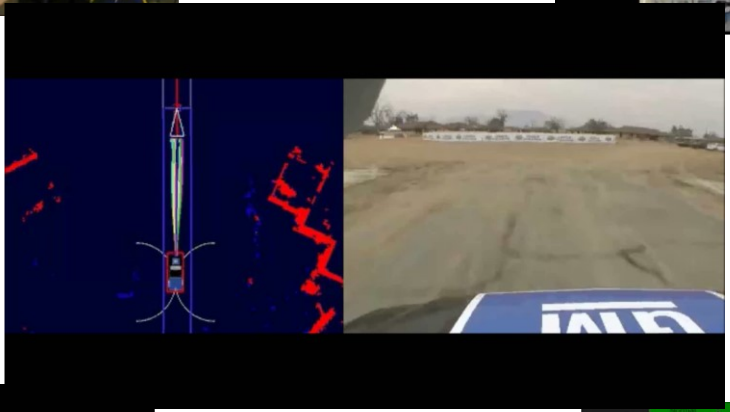
Joint work with

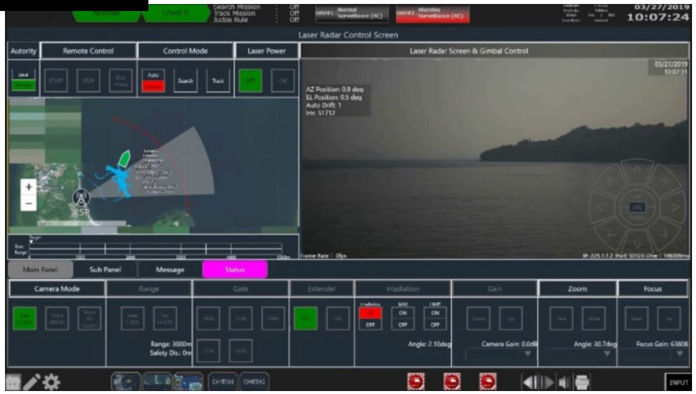
F. Islam, O. Salzman, A. Vemula *(and others)*

Research Focus of My Group

- Planning, Decision-making and Learning in robotic systems
- General algorithmic methods with rigorous theoretical guarantees
- Applications to real-world robotic problems/systems

Research Focus of My Group

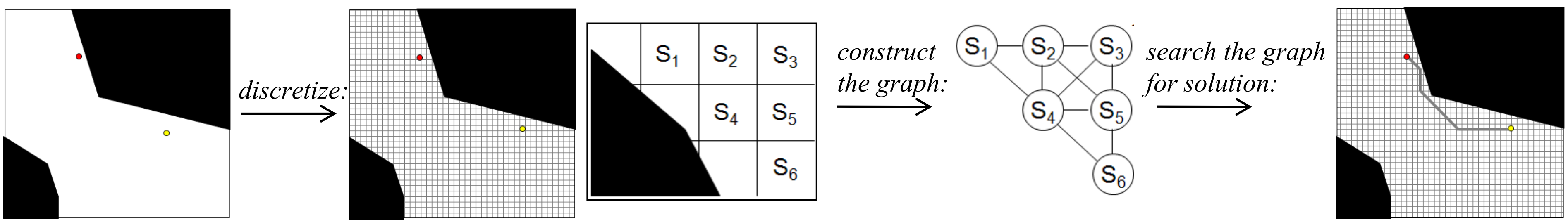




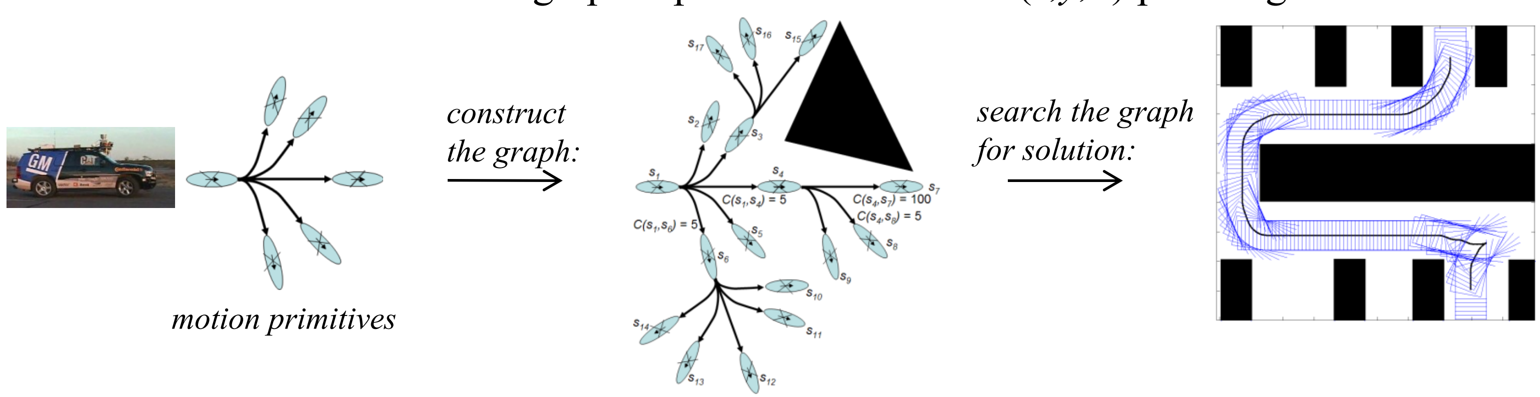
What is Search-based Planning

- Generate a systematic graph representation of the planning problem
- Search the graph for a solution with a heuristic search (e.g., A* search)
- Can interleave the construction of the representation with the search (i.e., construct only what is necessary)

2D grid-based graph representation for 2D (*x,y*) search-based planning:



Lattice-based graph representation for 3D (*x,y,θ*) planning:



What is Search-based Planning

- Generate a systematic grap … problem
- S
-

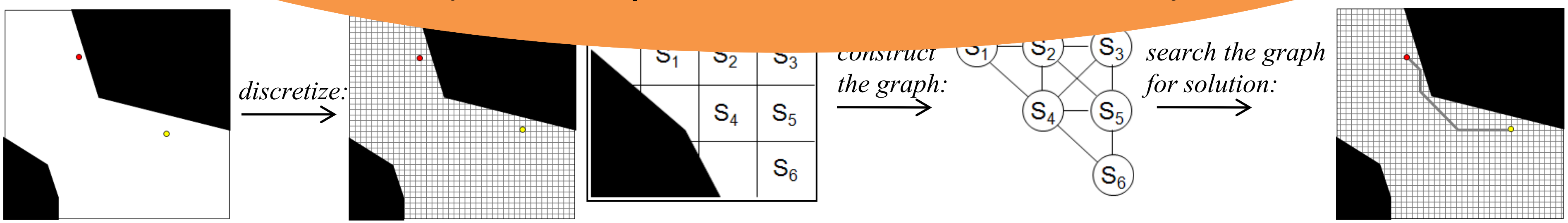
Pros:

- rigorous guarantees on solution quality

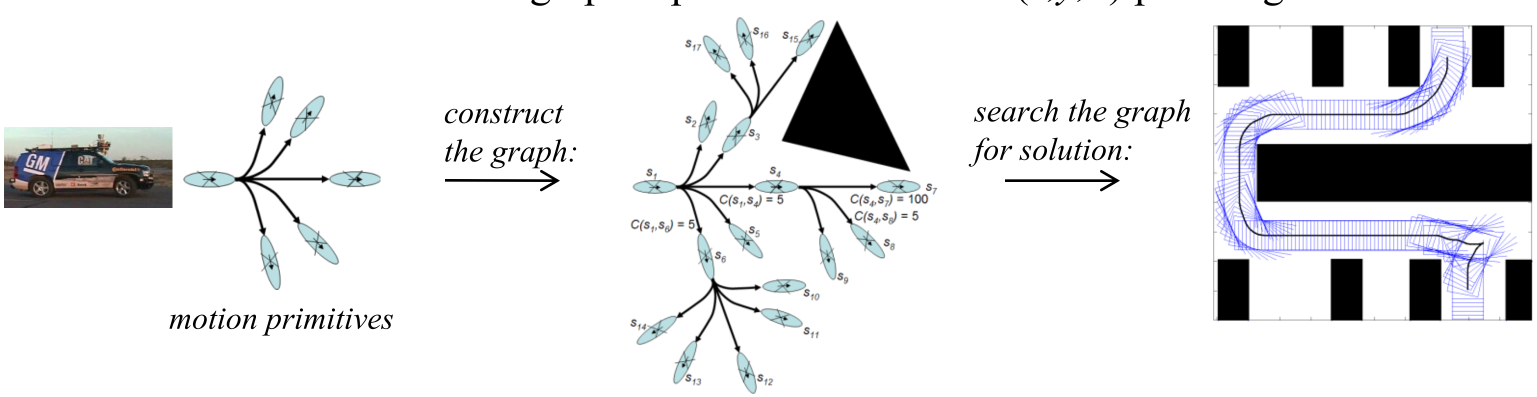
- good cost minimization

- consistency in solution

(similar queries = similar solutions)



Lattice-based graph representation for 3D (x,y,θ) planning:



What is Search-based Planning

- Generate a systematic grap … planning problem
- S

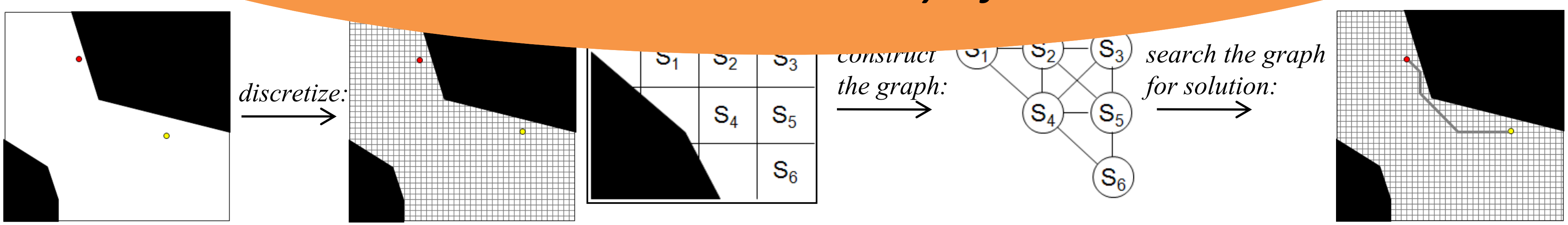
Challenges:

- high-dimensionality/graph size

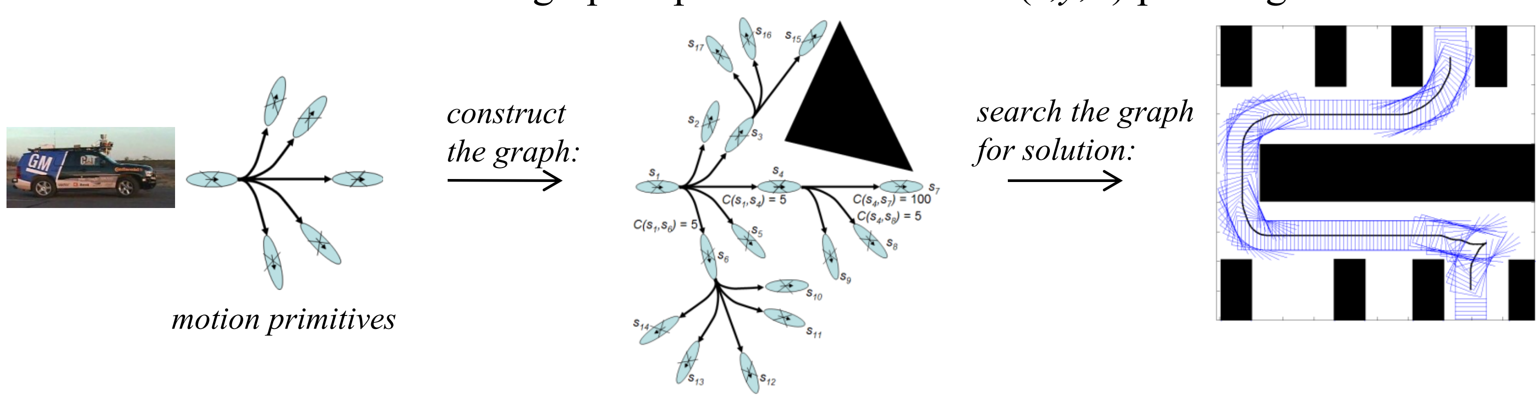
- expensive edge cost evaluation

- edge construction for dynamic systems

- reliance on the accuracy of the model



Lattice-based graph representation for 3D (x,y,θ) planning:



Agenda

- Challenges in Search-based Planning
- Constant-time Motion Planning (CTMP) – offline learning for online planning
- CMAX/CMAX++ for handling inaccurate models
- Summary and thoughts on research directions

Agenda

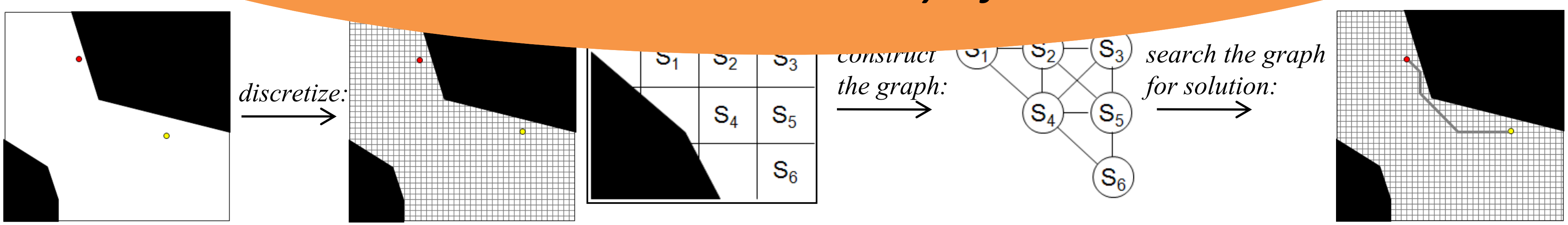
- Challenges in Search-based Planning
- Constant-time Motion Planning (CTMP) – offline learning for online planning
- CMAX/CMAX++ for handling inaccurate models
- Summary and thoughts on research directions

What is Search-based Planning

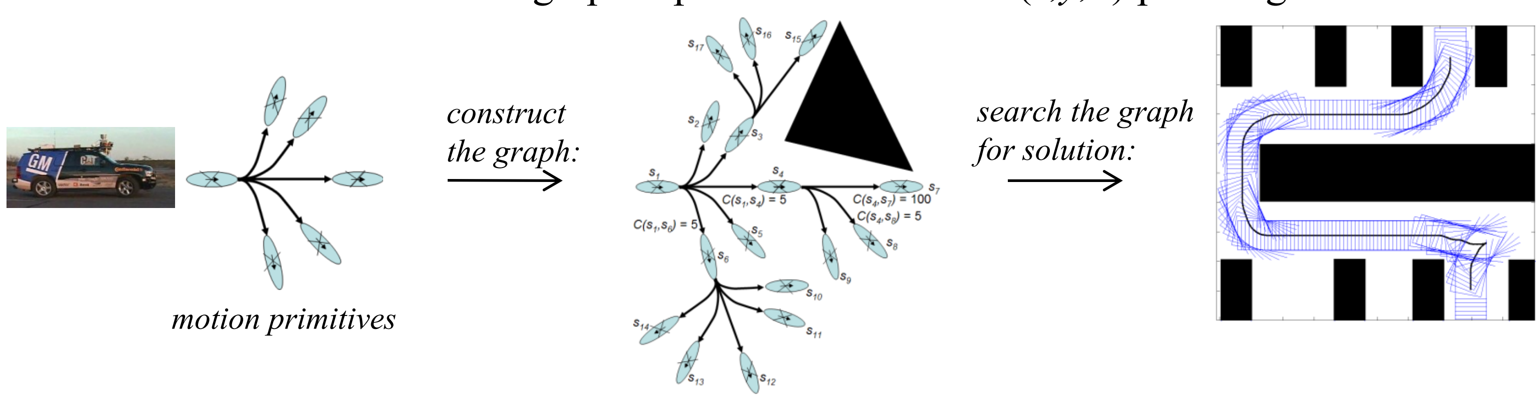
- Generate a systematic grap… problem
- S…

Challenges:

- *high-dimensionality/graph size*
- *expensive edge cost evaluation*
- *edge construction for dynamic systems*
- *reliance on the accuracy of the model*



Lattice-based graph representation for 3D (x,y,θ) planning:



High-dimensionality/Graph Size

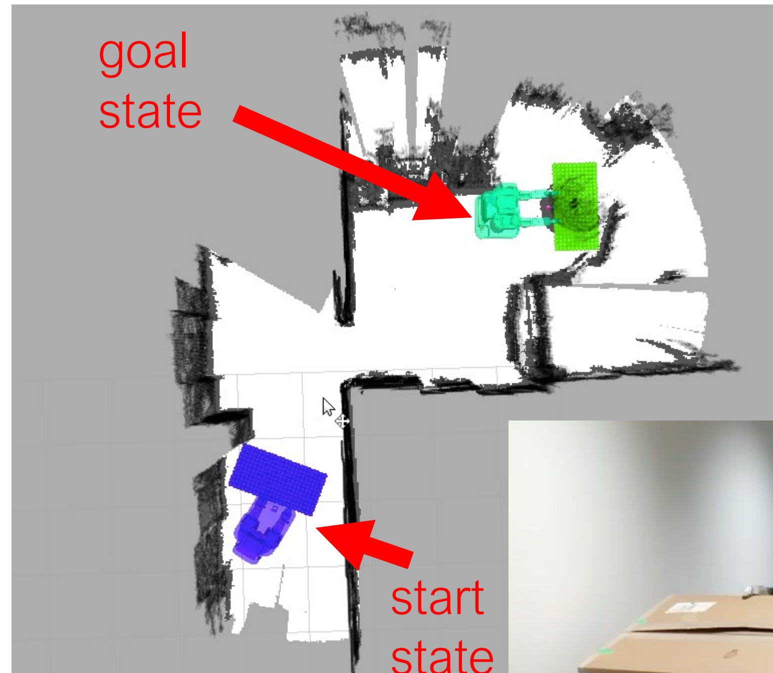
5-D trajectory planning

(x,y,z,Ɵ,v)

12-D full-body planning

(3D base pose, 1D torso height,

6DOF object pose, 2 redundant DOFs in arms)



High-dimensionality/Graph Size

5-D trajectory planning

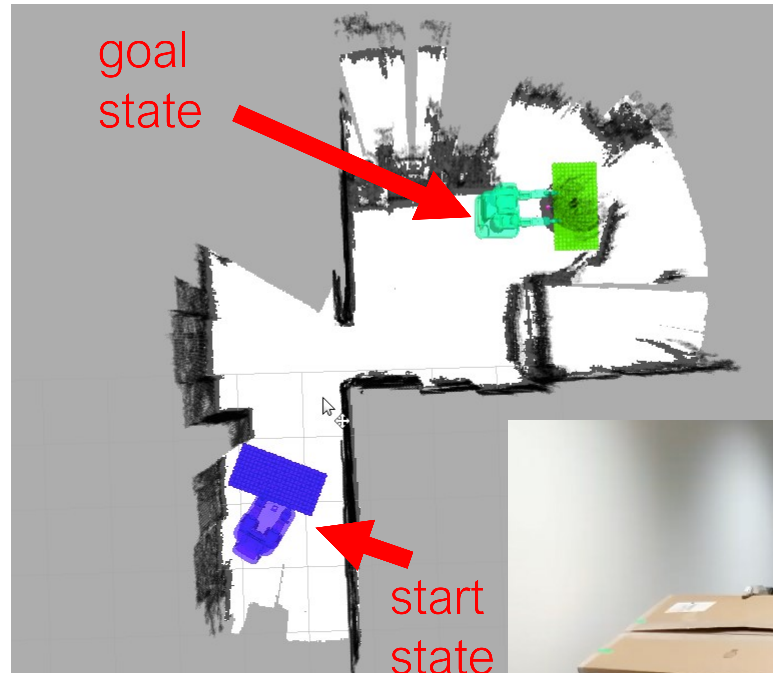
(x,y,z,Θ,v)

over 500M states!

(for 10km by 10km area discretized into 25m cubes, 32 yaw angles, 5 velocities)

12-D full-body planning

(3D base pose, 1D torso height, 6DOF object pose, 2 redundant DOFs in arms)



over 10^{19} states!

(for small indoor space, joint angle resolution = 10 degrees)

High-dimensionality/Graph Size

5-D trajecto ... y planning

... eight,

... arms)

Methodologies to address it:

- *implicit graphs*
- *compact graph representations including adaptive dimensionality [Gochev et al. 11]*
- *sub-optimal/anytime search [Pearl 84, Likhachev et al. 04, Hansen & Zhou 07, Thayer & Ruml 08,…]*
- *incremental planning, especially within sub-optimal/anytime search [Koenig & Likhachev 04, Likhachev et al. 08, …]*
- *deriving multiple heuristics, each corresponding to a low-dimensional version of the problem, and using these via Multi-Heuristic A* [Aine et al. 15]*

over 500M states!

(for 10km by 10km area discretized into 25m cubes, 32 yaw angles, 5 velocities)

over 10^{19} states!

(for small indoor space, joint angle resolution = 10 degrees)

What is Search-based Planning

- Generate a systematic graph … planning problem
- S…

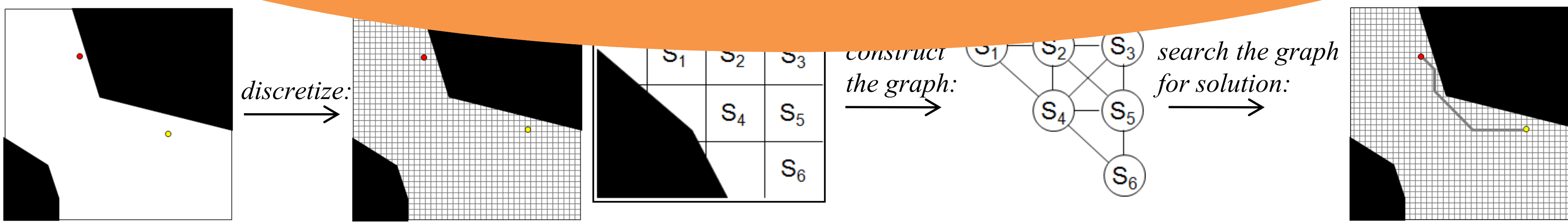
Challenges:

- high-dimensionality/graph size

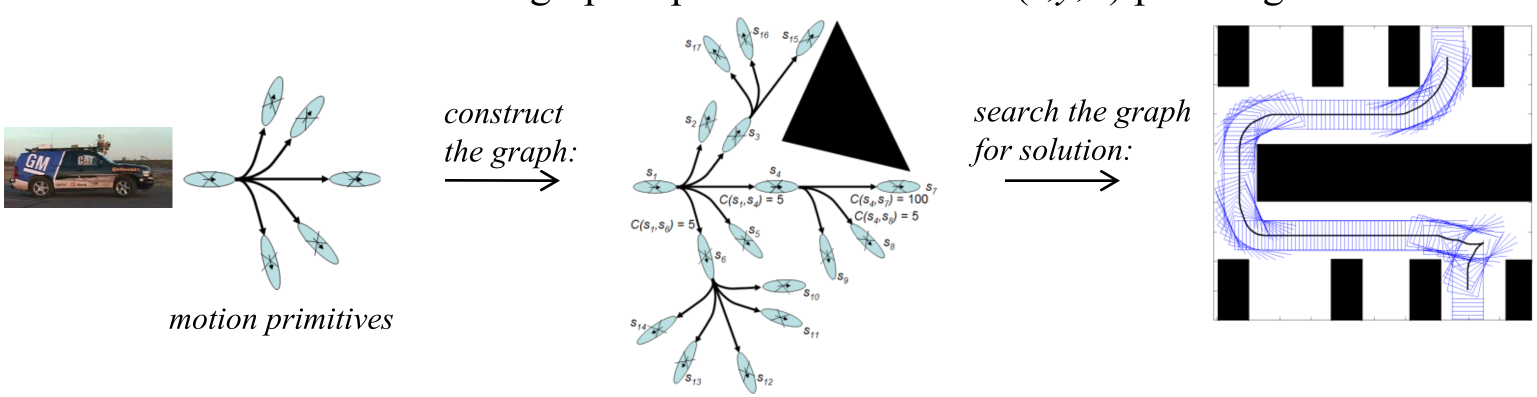
- expensive edge cost evaluation

- edge construction for dynamic systems

- reliance on the accuracy of the model

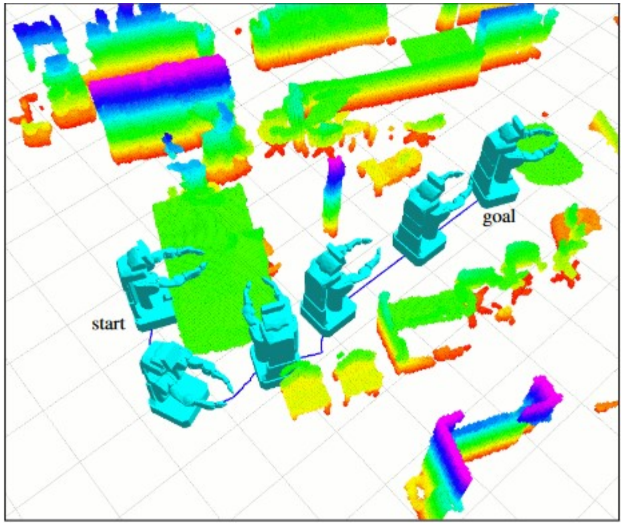


Lattice-based graph representation for 3D (x,y,θ) planning:



Expensive Edge Cost Evaluation

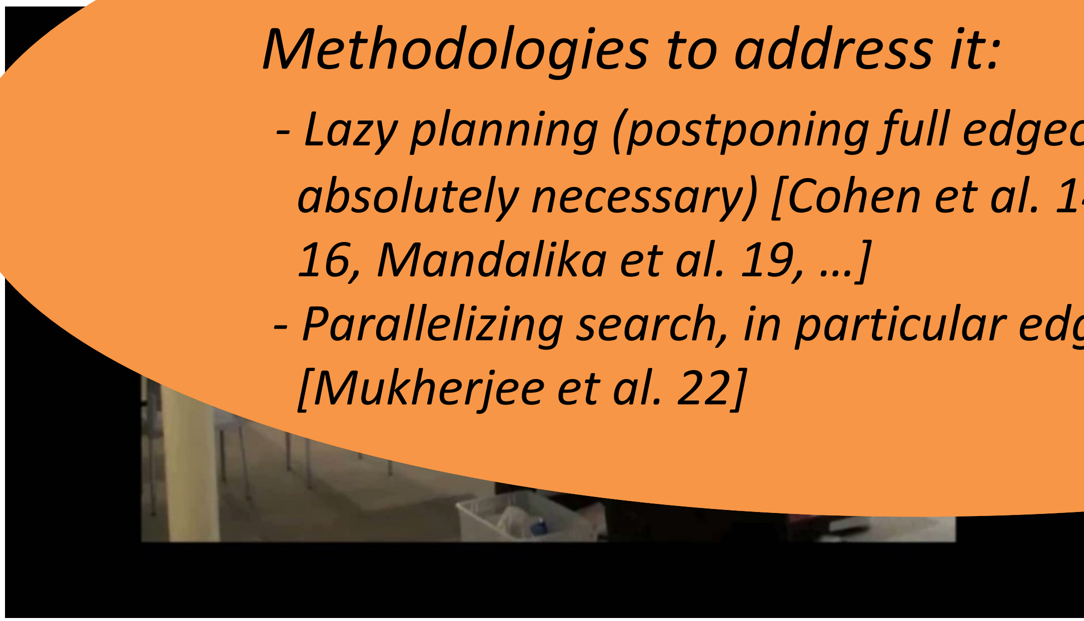
*3-D (x,y,Θ) planning with **full-body** collision checking*

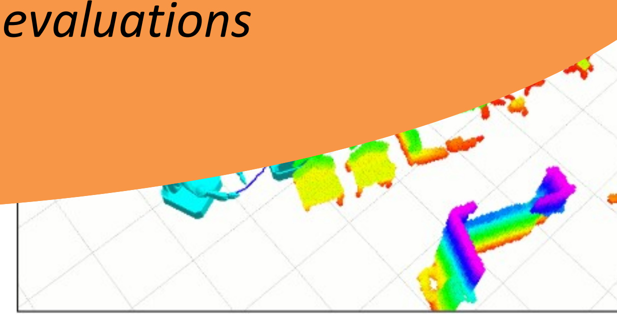


Work done in collaboration with Willow Garage

[Hornung et al. 12]

Expensive Edge Cost Evaluation





Methodologies to address it:

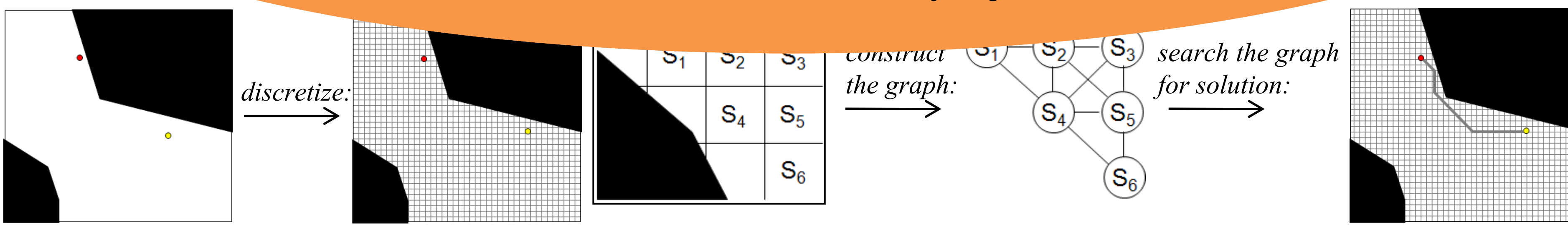
- *Lazy planning (postponing full edgecost evaluation until absolutely necessary) [Cohen et al. 14, Dellin & Srinivasa 16, Mandalika et al. 19, …]*
- *Parallelizing search, in particular edge evaluations [Mukherjee et al. 22]*

What is Search-based Planning

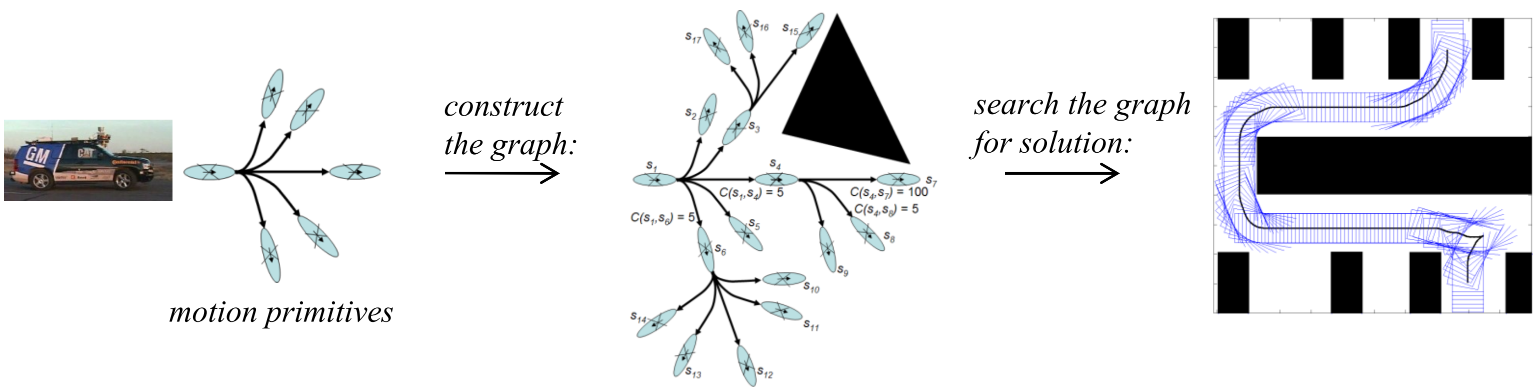
- Generate a systematic grap[…]ing problem
- S[…]

Challenges:

- *high-dimensionality/graph size*
- *expensive edge cost evaluation*
- *edge construction for dynamic systems*
- *reliance on the accuracy of the model*

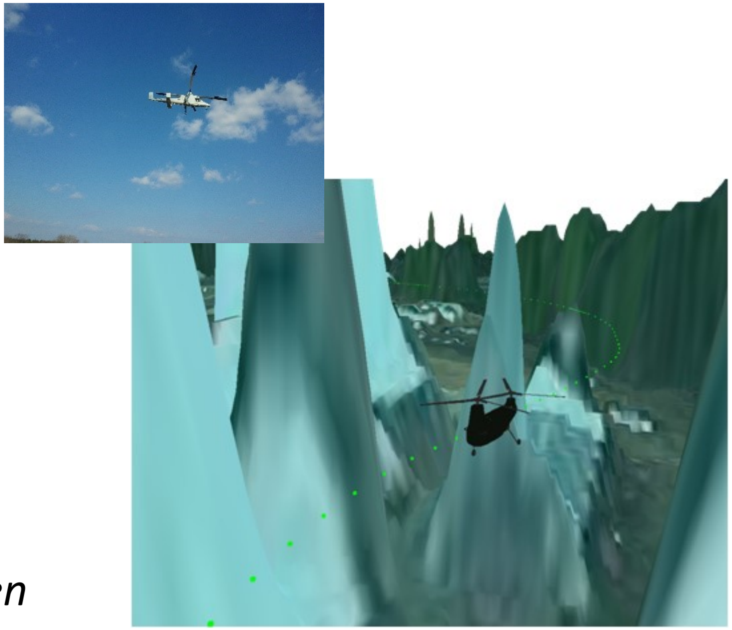


Lattice-based graph representation for 3D (x,y,θ) planning:

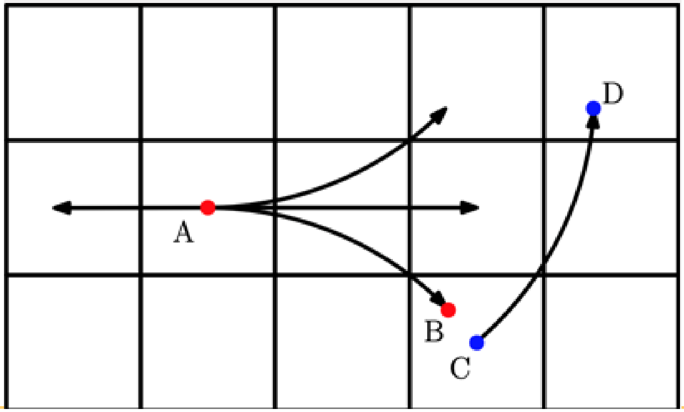


Edge Construction for Dynamic Systems

Planning for a highly dynamic driving (collaboration between Thyssenkrupp and RobotWits, now part of Waymo)



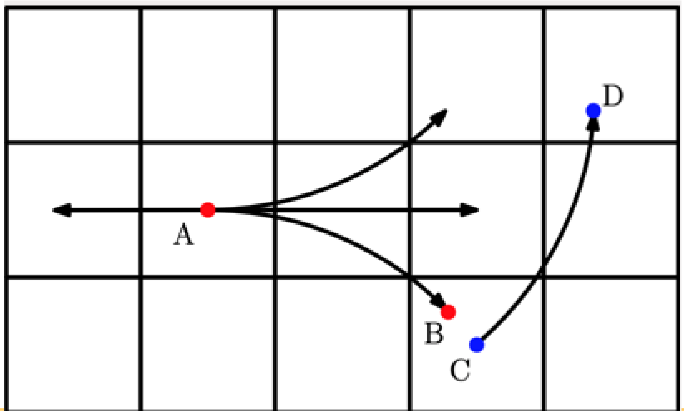
Hard/impossible to construct transitions whose endpoints land at the centers of high-d cells

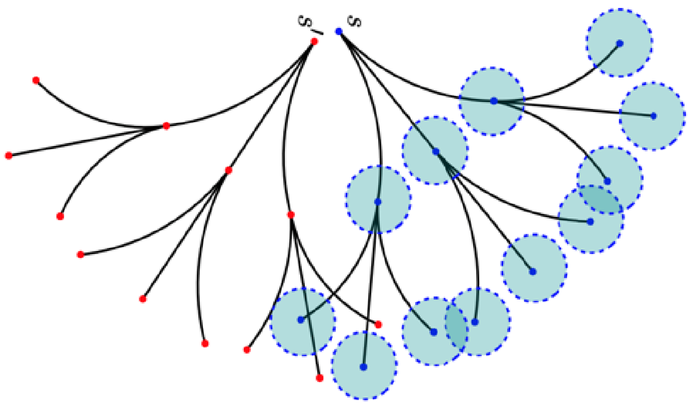


Methodologies to address it:

- Soft duplicate detection for search without state discretization but de-prioritizing states that are similar to previously expanded states (i.e., are soft duplicates) [Du et al. 19, Maray et al. 22]

Hard/impossible to construct transitions whose endpoints land at the centers of high-d cells



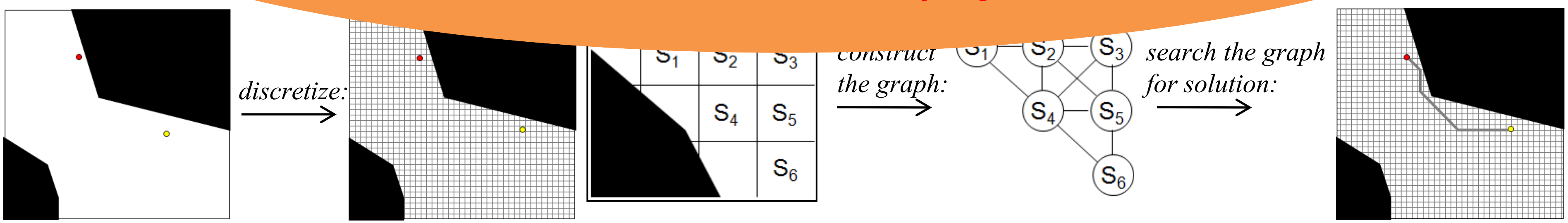


What is Search-based Planning

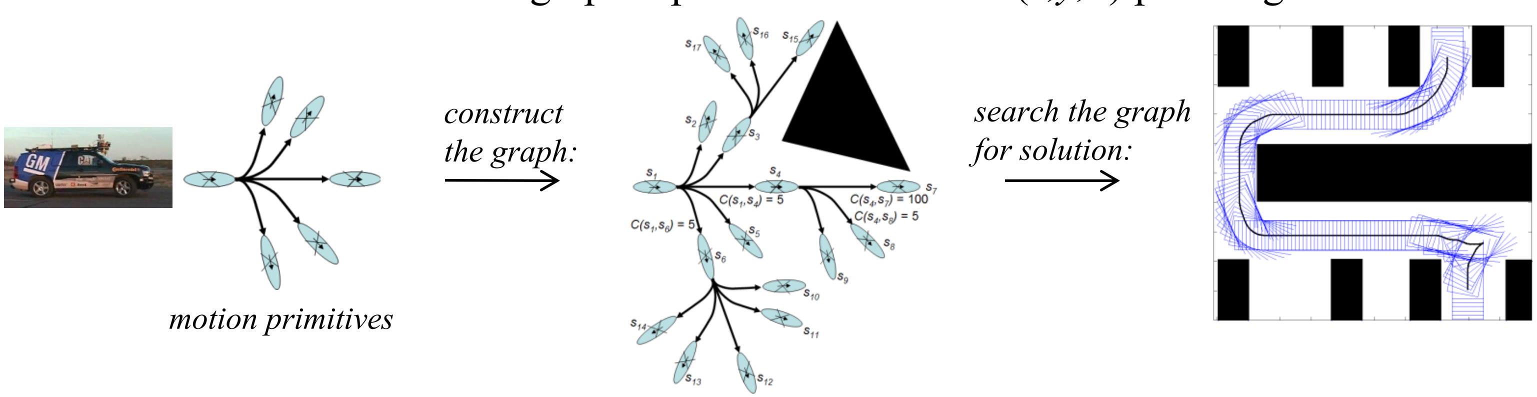
- Generate a systematic graph … problem
- S…

Challenges:

- *high-dimensionality/graph size*
- *expensive edge cost evaluation*
- *edge construction for dynamic systems*
- *reliance on the accuracy of the model*



Lattice-based graph representation for 3D (x,y,θ) planning:

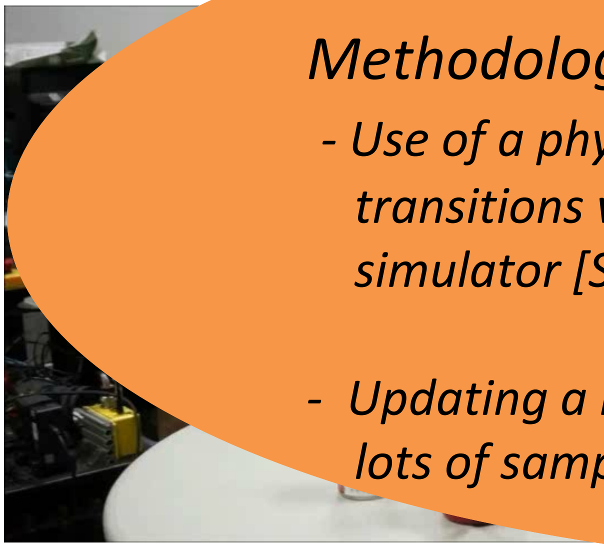


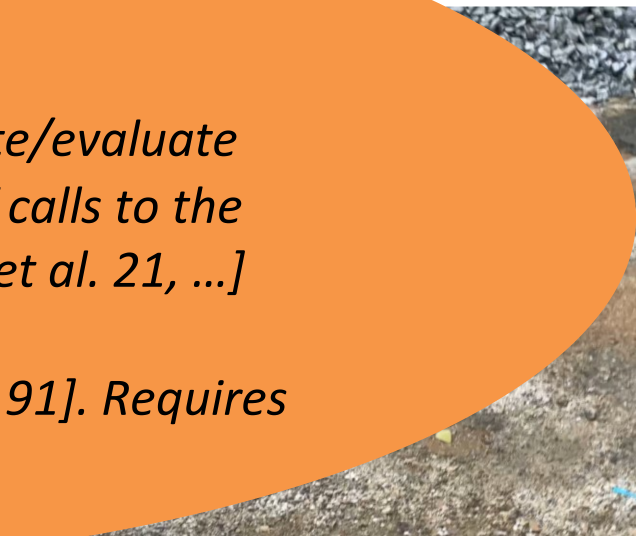
Reliance on the Accuracy of the Model

Planning for tasks requiring heavy interaction with the world

[Saleem & Likhachev 20]

Reliance on the Accuracy of the Model





Methodologies to address it:

- *Use of a physics-based simulator to compute/evaluate transitions while minimizing the number of calls to the simulator [Saleem & Likhachev 20, Saxena et al. 21, …]*

- *Updating a model during execution [Sutton 91]. Requires lots of samples and an updateable model.*

[Saleem & Likhachev 20]

What is Search-based Planning

- Generate a systematic grap ... problem

Challenges:

- high-dimensionality/graph size

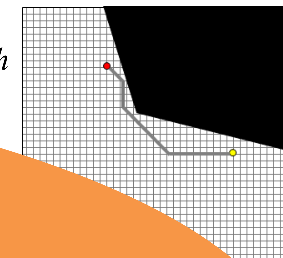
- expensive edge cost evaluation

- edge construction for dynamic systems

- reliance on the accuracy of the model

discretize:

search the graph for solution:



Lots of work on learning to address these challenges:

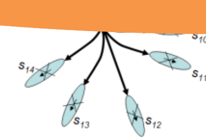
- learning heuristics [Bhardwaj et al. 17, …]

- learning collision detection estimator [Das & Yip 19, Huh & Lee 16, …]

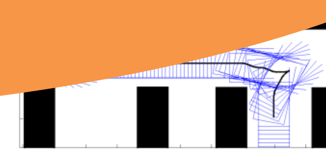
- learning soft duplicate measure [Maray et al. 22]

- learning residual models [Nagabandi et al. 19, …]

- ….



motion primitives



What is Search-based Planning

- Generate a systematic graph … problem

Challenges:

- high-dimensionality/graph size

- expensive edge cost evaluation

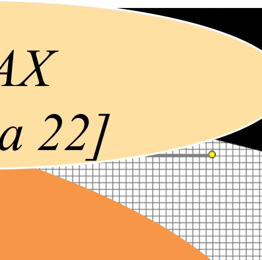
- edge construction for dynamic systems

- reliance on the accuracy of the model

CTMP algorithms [Islam et al. 21]

CMAX [Vemula 22]

discretize:



Lots of work on learning to address these challenges:

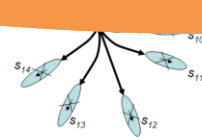
- learning heuristics [Bhardwaj et al. 17, …]

- learning collision detection estimator [Das & Yip 19, Huh & Lee 16, …]

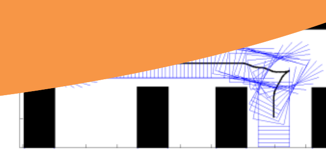
- learning soft duplicate measure [Maray et al. 22]

- learning residual models [Nagabandi et al. 19, …]

- ….



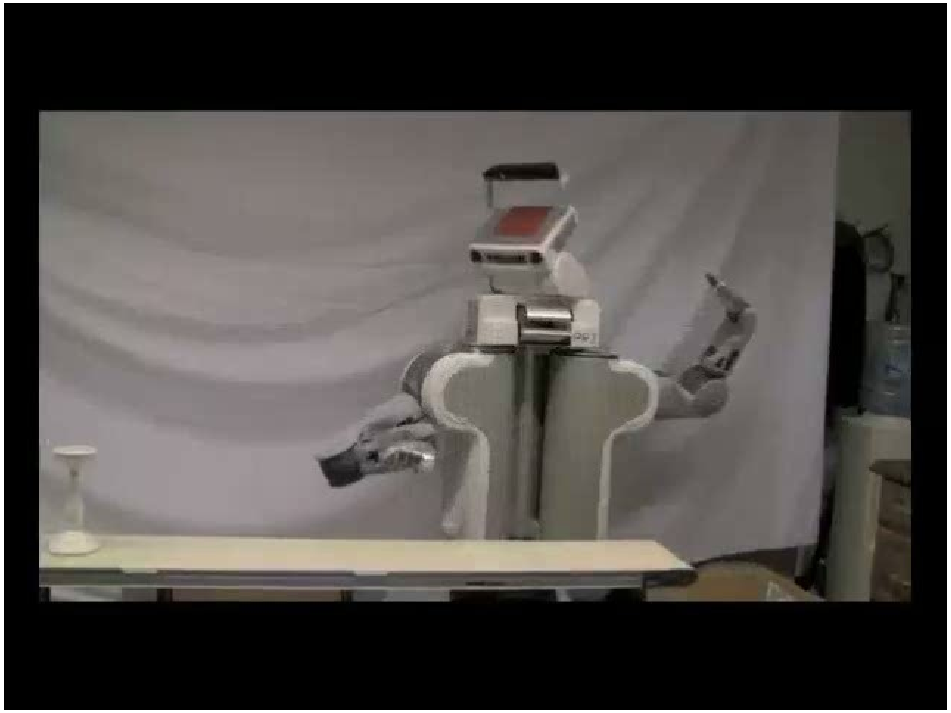
motion primitives



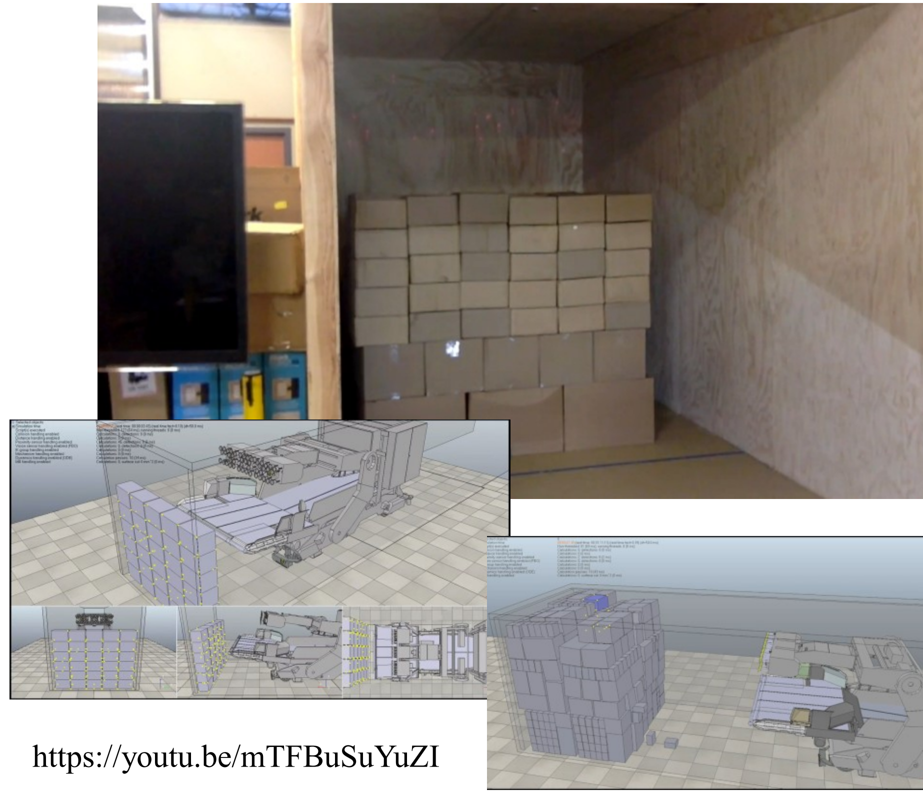
Agenda

- Challenges in Search-based Planning
- Constant-time Motion Planning (CTMP) – offline learning for online planning
- CMAX/CMAX++ for handling inaccurate models
- Summary and thoughts on research directions

Motivation for Constant-time Motion Planning



[Cowley et al., 2013], joint work with CJ Taylor

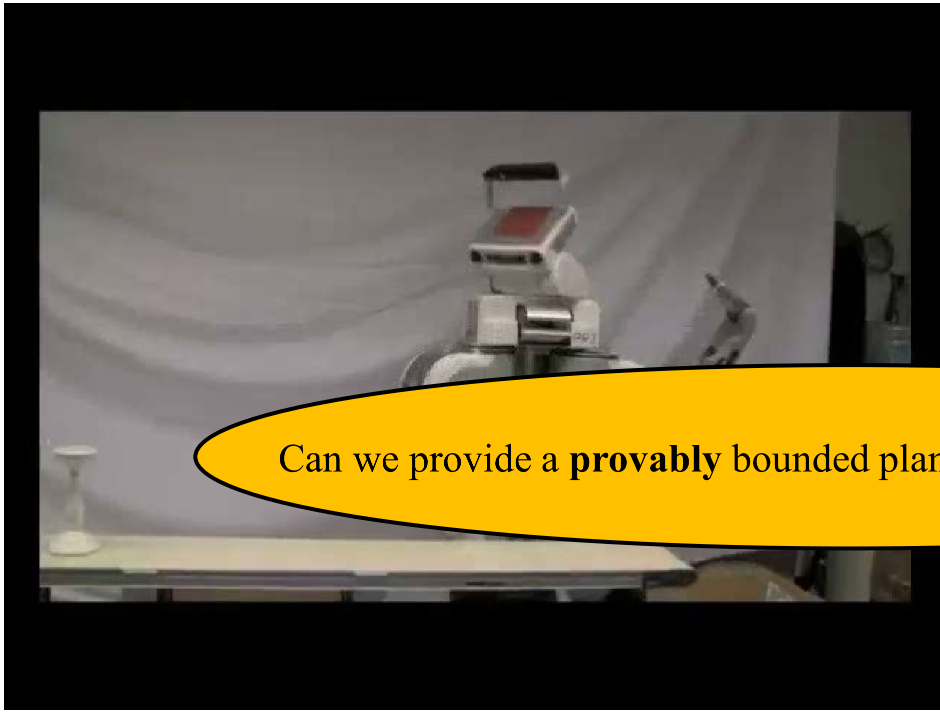


https://youtu.be/mTFBuSuYuZI

Autonomous truck unloading
(joint work with Honeywell & NREC - Herman, Pires, etc.)

- Planning often needs to be fast and “constant-time”
- while tasks are often repetitive

Motivation for Constant-time Motion Planning



Can we provide a **provably** bounded planning time by pre-processing?

https://youtu.be/mTFBuSuYuZI

[Cowley et al., 2013], joint work with CJ Taylor

Autonomous truck unloading
(joint work with Honeywell & NREC - Herman, Pires, etc.)

- Planning often needs to be fast and "constant-time"
- while tasks are often repetitive

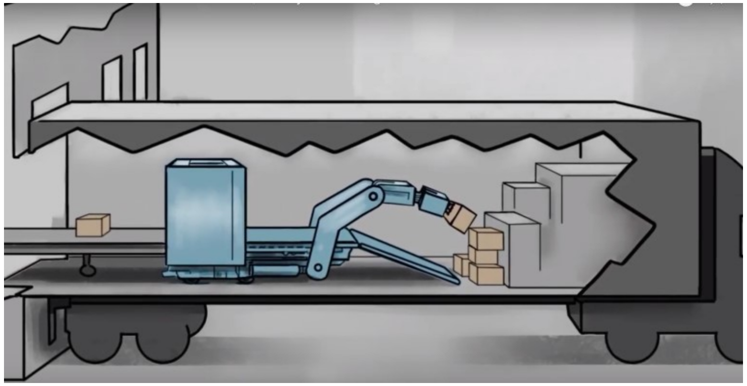
Constant-time Motion Planning (CTMP) class of algorithms

[Islam et al., ICAPS'19], [Islam et al., RSS'20], [Islam et al., ICRA'21]

Algorithms that learn offline data structures which enable online planners to guarantee to find a solution (if one exists) within a (small) user-defined time

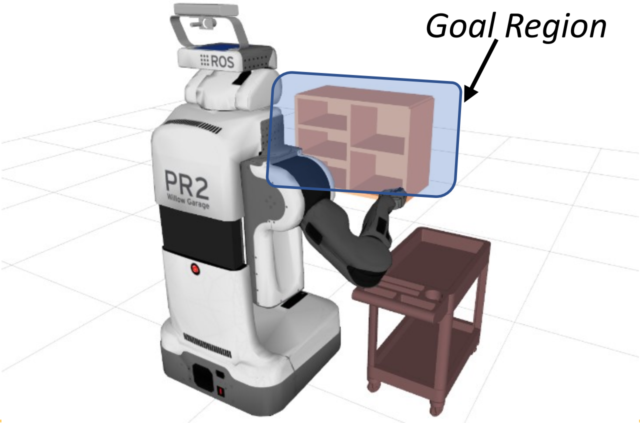
Assumptions

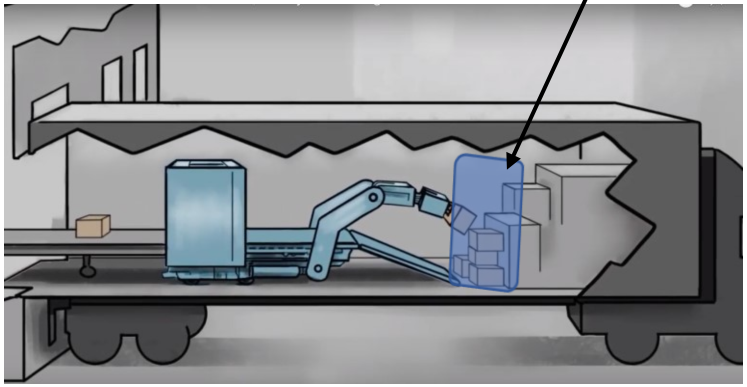
- Given (known) environment
- Given set of potential start configurations
- Given set of corresponding goal regions (e.g., we need to handle goal perturbations)



Assumptions

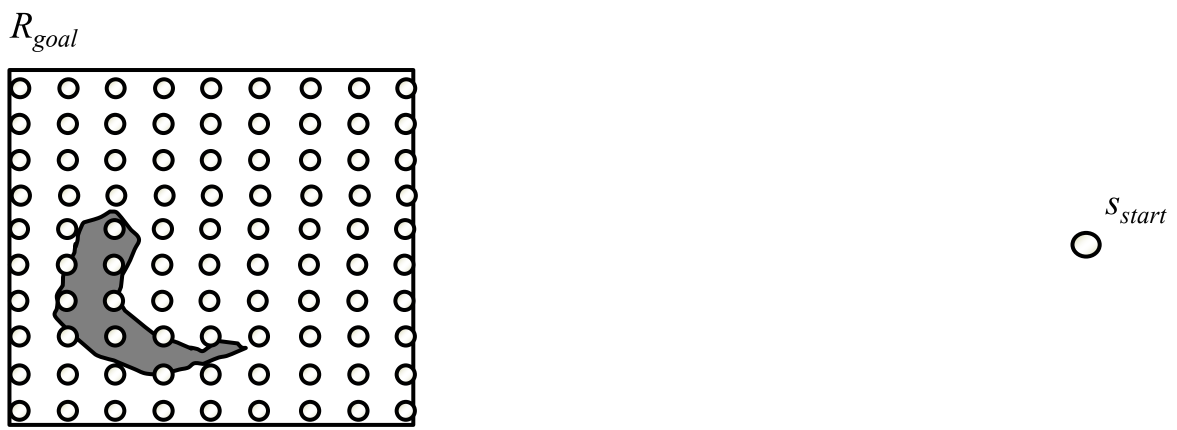
- Given (a mostly known) environment
- Given set of potential start configurations
- Given set of corresponding goal regions (e.g., we need to handle goal perturbations)





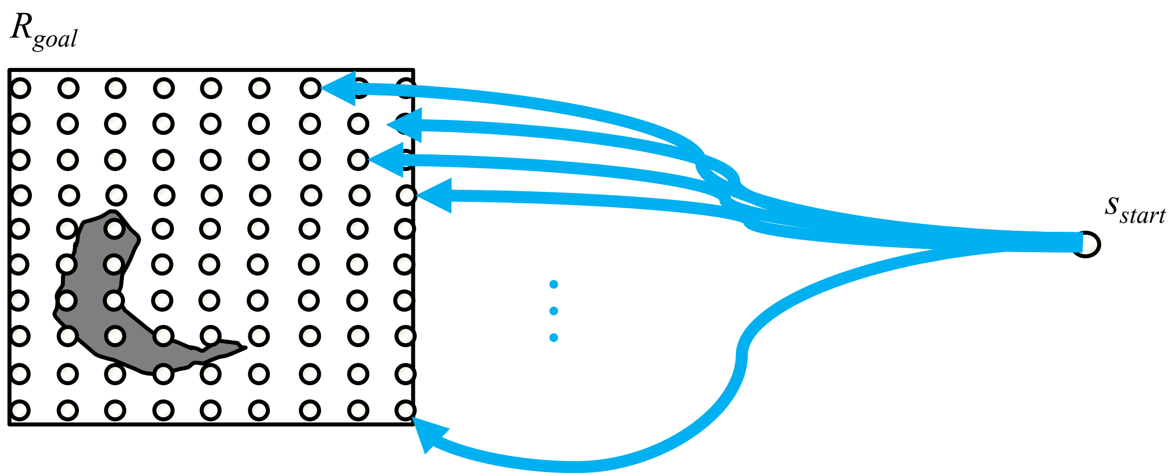
- Given a start state and a goal region

- Given a start state and a goal region
- Discretize the goal region



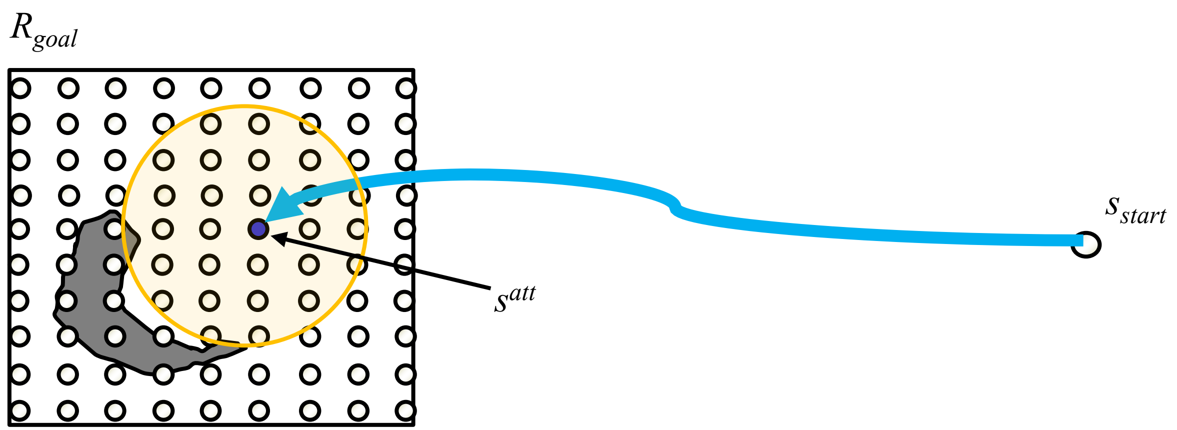
Constant-time Motion Planning (CTMP)

- Given a start state and a goal region
- Discretize the goal region
- ~~We could pre-compute all paths but too long, too much memory… and not~~ interesting ☺.



- Given a start state and a goal region
- Discretize the goal region

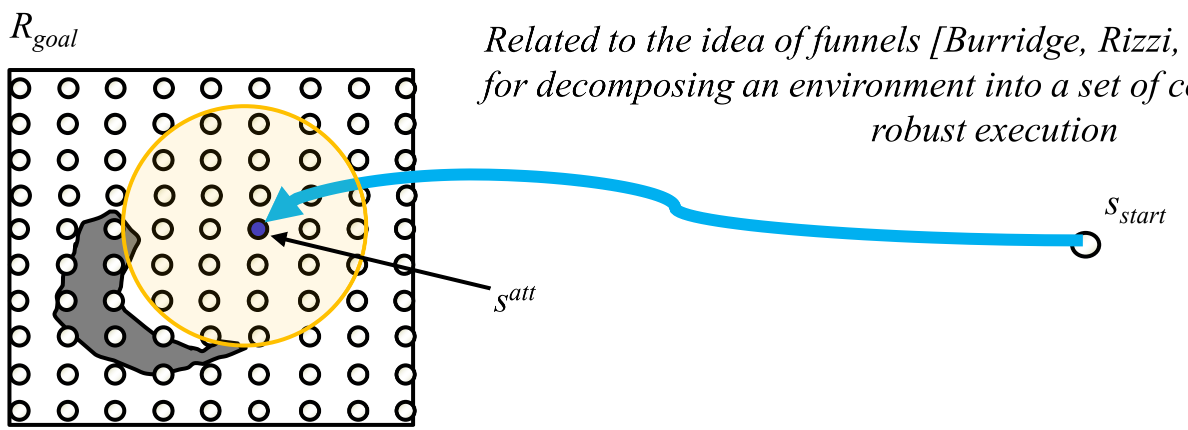
<u>Key idea:</u>
Given a potential function and any "attractor" state S^{att} , there is typically a large region of states that can reach the attractor state following the potential function



- Given a start state and a goal region
- Discretize the goal region

<u>Key idea:</u>

Given a potential function and any "attractor" state S^{att} , there is typically a large region of states that can reach the attractor state following the potential function



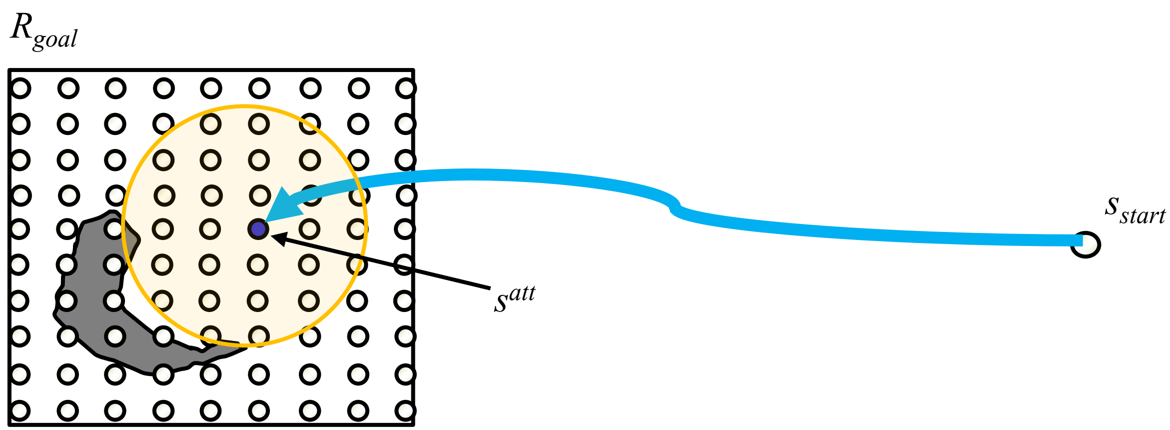
Related to the idea of funnels [Burridge, Rizzi, and Koditschek, '99] for decomposing an environment into a set of controller domains for robust execution

Constant-time Motion Planning (CTMP)

- Given a start state and a goal region
- Discretize the goal region

Key idea:

Given a potential function and any "attractor" state S^{att}, there is typically a large region of states that can reach the attractor state following the potential function



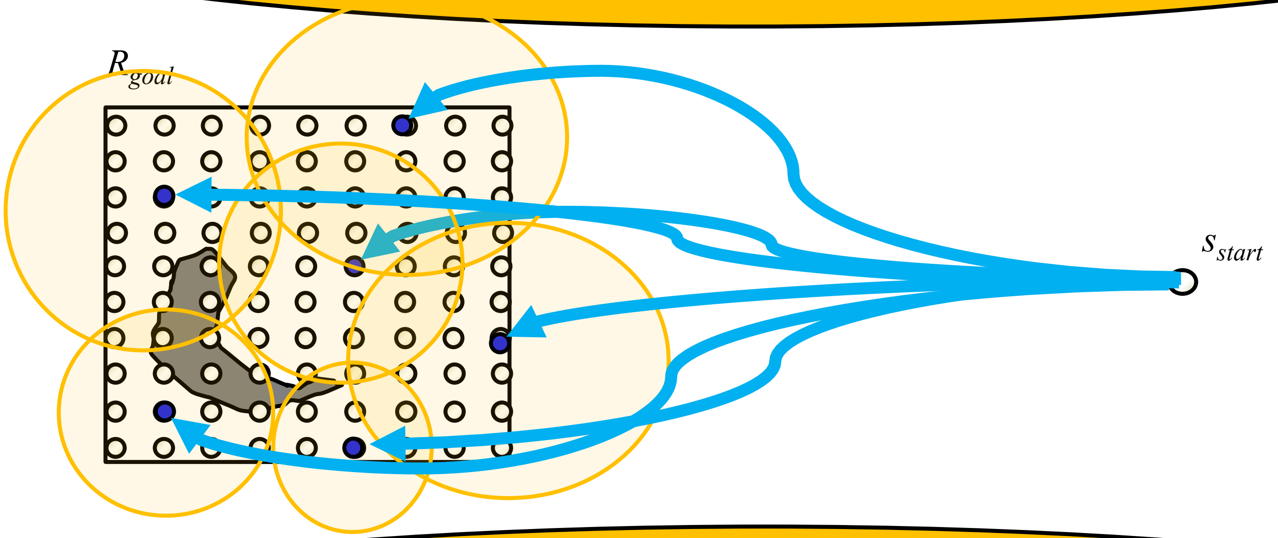
Pre-processing algorithm:

Decompose R_{goal} into a set of subregions $R_{1\ldots K}$, each defined by $\{S_i^{att}, r_i\}$, s. t. $U_i^K R_i$ completely covers R_{goal} where r_i – radius of subregion R_i

Constant-time Motion Planning (CTMP)

- Given a start state and a goal region
- Discretize the goal region

<u>Key idea:</u>
Given a potential function and any "attractor" state S^{att}, there is typically a large region of states that can reach the attractor state following the potential function



<u>Pre-processing algorithm:</u>
Decompose R_{goal} into a set of subregions $R_{1\ldots K}$, each defined by $\{S_i^{att}, r_i\}$,
s. t. $U_i^K R_i$ completely covers R_{goal}
where r_i – radius of subregion R_i

Constant-time Motion Planning (CTMP)

- Given a start state and a goal region
- Discretize the goal region

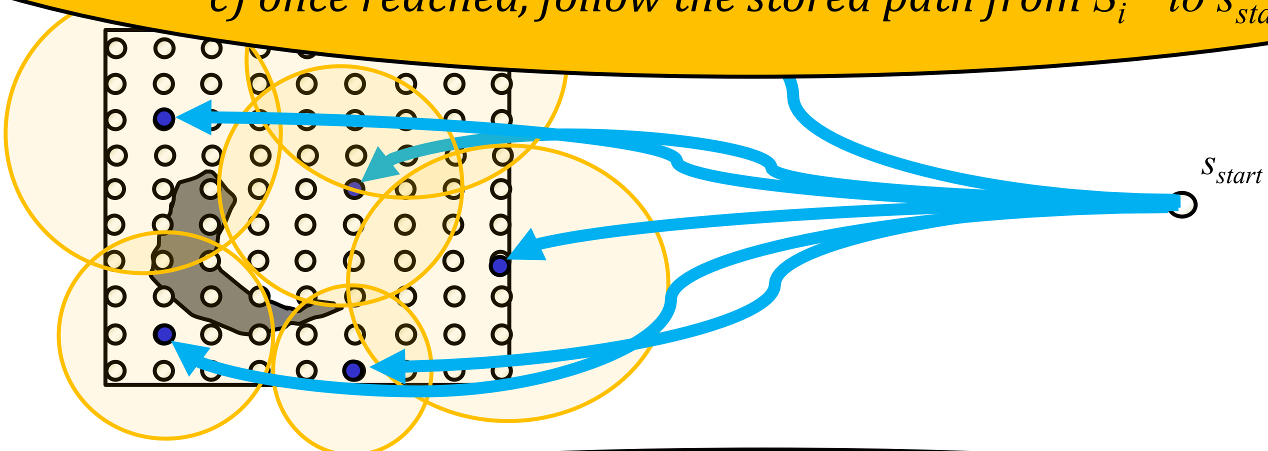
Online Planning:

Given an s_{goal} in R_{goal}:

a) find which R_i contains s_{goal},

b) follow the potential function towards S_i^{att},

c) once reached, follow the stored path from S_i^{att} to s_{start}



Pre-processing algorithm:

Decompose R_{goal} into a set of subregions $R_{1\ldots K}$, each defined by $\{S_i^{att}, r_i\}$,

s. t. $U_i^K R_i$ completely covers R_{goal}

where r_i – radius of subregion R_i

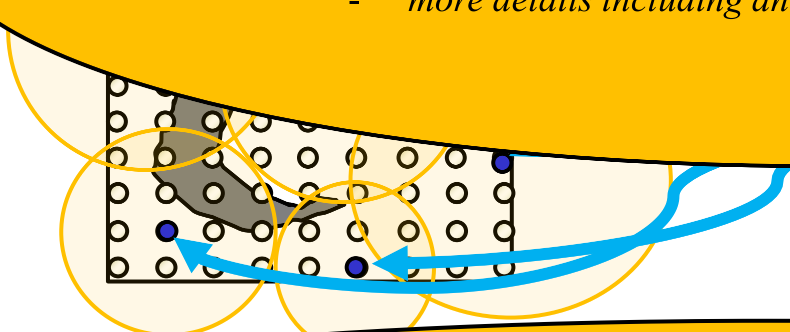
Constant-time Motion Planning (CTMP)

- Given a start state and a goal region
- Discretize the goal region

Online Planning:

Details on pre-processing:

- *each subregion R_i can be computed via single backward search from S_i^{att}*
- *S_{i+1}^{att} can be any state that lies in the portion of R_{goal} that isn't yet covered*
- *more details including analysis in [Islam et al., ICAPS'19]*



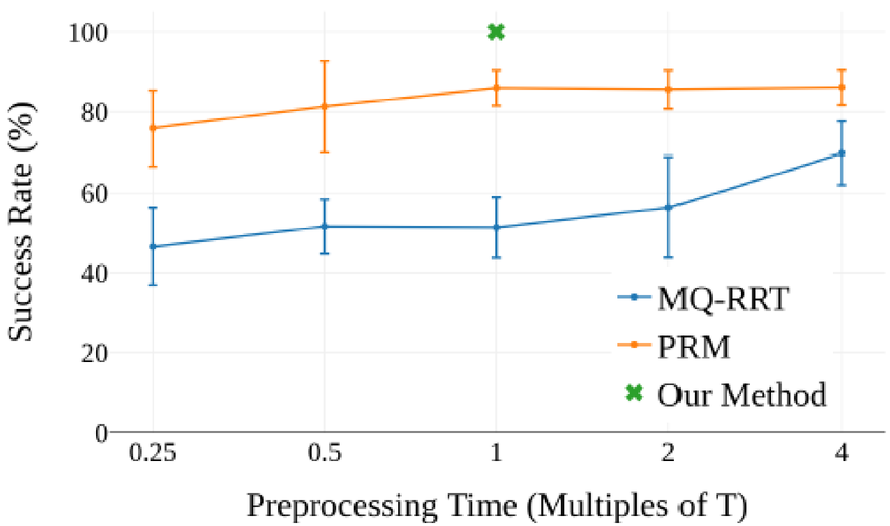
Provides provable guarantee on finding a plan to the goal (if one exists) within a given user-defined time

where r_i – radius of subregion R_i

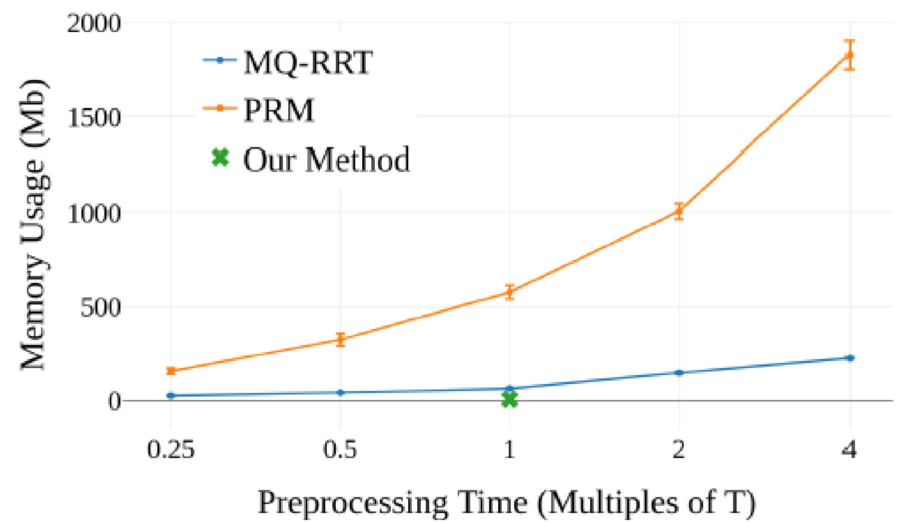
Constant-time Motion Planning (CTMP)

- On bin picking 7DOF arm planning

	PRM (4T)	MQ-RRT (4T)	E-graph	RRT-Connect	**Our method**
Planning time [ms]	21.7 (59.6)	21.2 (35.5)	497.8 (9678.5)	1960 (9652)	**1.0 (1.6)**
Success rate [%]	86	69.75	76.5	83.8	**100**
Memory usage [Mb]	1,828	225.75	**2.0**	-	7.8



(a)



(b)

Constant-time Motion Planning (CTMP) class of algorithms

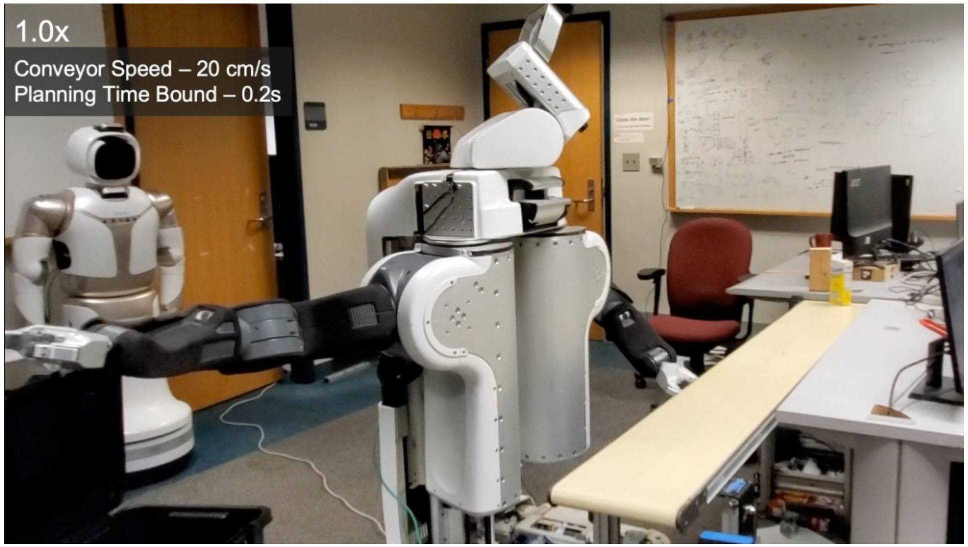
[Islam et al., ICAPS'19], [Islam et al., RSS'20], [Islam et al., ICRA'21]

Algorithms that learn offline data structures which enable online planners to guarantee to find a solution (if one exists) within a (small) user-defined time

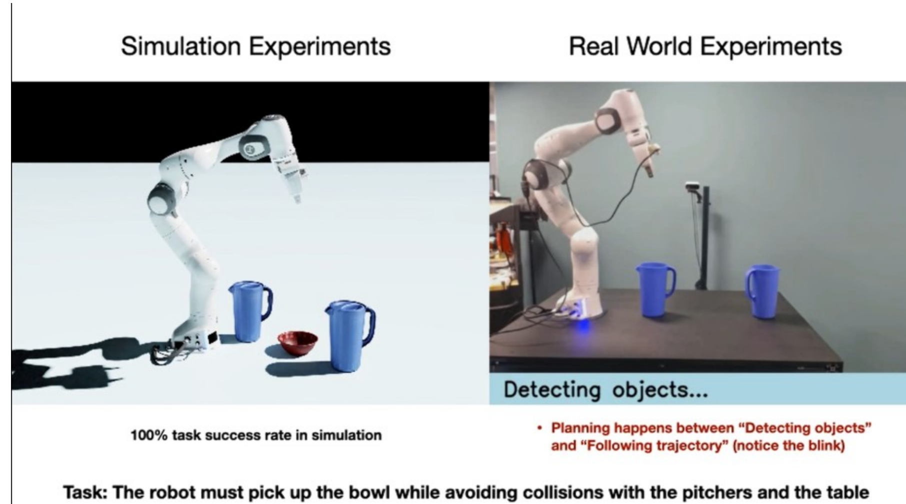
Examples of observations exploited by these algorithms:

- *Goal region can be decomposed into subsets within which one can get to its center by following a potential function [Islam et al., ICAPS'19]*
- *Paths can be precomputed so as to guide Experience-based planner to find a solution within X expansions [Islam et al., RSS'20]*
- *Disjoint paths guarantee that at least one is feasible given a potential for an obstacle blocking one [Islam et al., ICRA'21]*

CTMP in Action



CTMP for picking dynamic objects off a conveyor with imperfect perception [Islam et al., RSS'20]



CTMP for picking up objects in partially-known environments [Islam, Paxton, Eppner, Peele, Likhachev, Fox et al., ICRA'21] (collaboration with Nvidia)

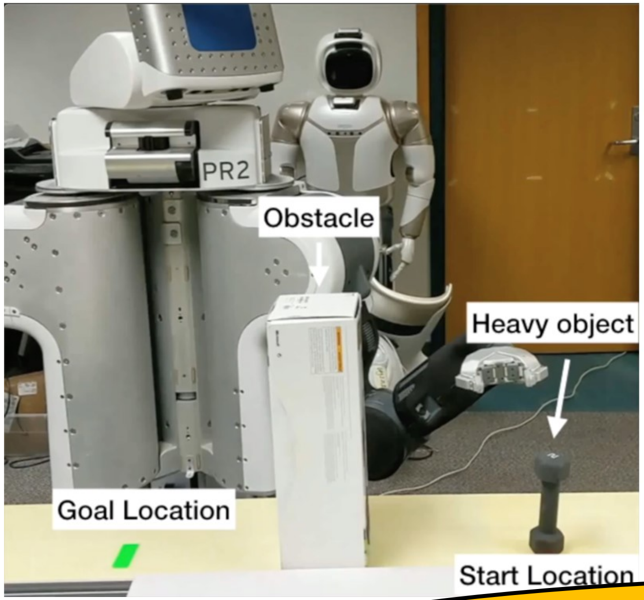
CTMP for Shield-based Protection project (work in progress)

Agenda

- Challenges in Search-based Planning
- Constant-time Motion Planning (CTMP) – offline learning for online planning
- CMAX/CMAX++ for handling inaccurate models
- Summary and thoughts on research directions

Motivation

- Planning models for real world tasks are often complex (e.g., physics-based simulators, analytically computed motion primitives, etc.), yet often imperfect
- Learning a model on-the-fly requires too many samples for goal-oriented execution



How to interleave planning and execution to **guarantee task achievement** despite the inaccuracies in the model?

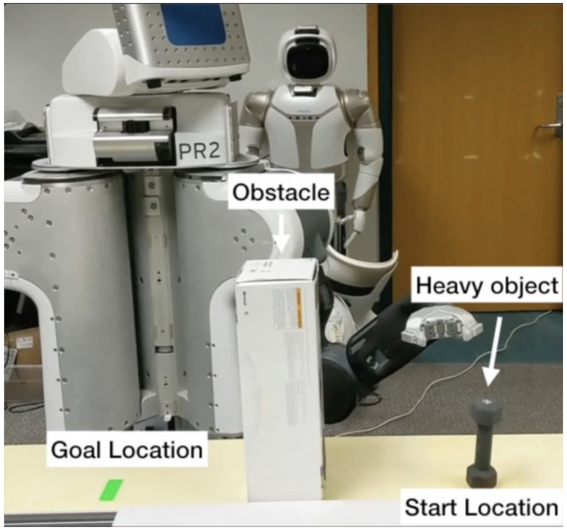
CMAX

Related to
"cost poisoning" [Zucker et al. 2011]

- Main points behind CMAX [Vemula, Oza, Bagnell & Likhachev, RSS'20]
  - instead of updating dynamics, inflates the cost of transitions discovered to be incorrect
  - does not require updates to the dynamics of the model
  - uses limited expansion search as planner to bound computation
  - uses function approximation to scale to large state spaces

- Guarantees task achievement under certain conditions

CMAX



- Objective:

Provably reach the goal online, despite having an inaccurate dynamical model, ***without any resets***

[1] *Resets allow the robot to "reset" to a state, usually a previously visited state*

- The problem is formulated as a shortest path problem $M=(\mathbb{S},\mathbb{A},\mathbb{G},f,c)$

$\mathbb{S}$: State space, $\mathbb{A}$: Discrete action space, $\mathbb{G}$: Goal space

Cost function: c: $\mathbb{S}\times\mathbb{A}\rightarrow[0,1]$

Unknown Deterministic True Dynamics: f: $\mathbb{S}\times\mathbb{A}\rightarrow\mathbb{S}$

Access to Approximate Dynamics: f': $\mathbb{S}\times\mathbb{A}\rightarrow\mathbb{S}$

State is fully observable

- Incorrect transitions:

Transitions where true and approximate dynamics differ

for example, $f(s,a) \neq f'(s,a)$ or $\| f(s,a) - f'(s,a) \| > \xi$

ξ - smallest discrepancy handled by low-level feedback controller

$\mathcal{X} \subseteq \mathbb{S} \times \mathbb{A}$ = set of "incorrect" transitions

$\mathcal{X}$ is not known beforehand, and is only discovered through online executions

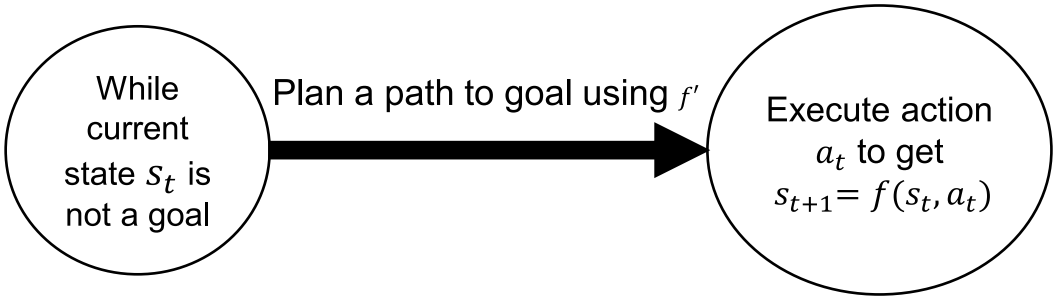
Key Idea:

Instead of learning the true dynamics, CMAX maintains

a running estimate of the set of incorrect transitions $\mathcal{X}$ and

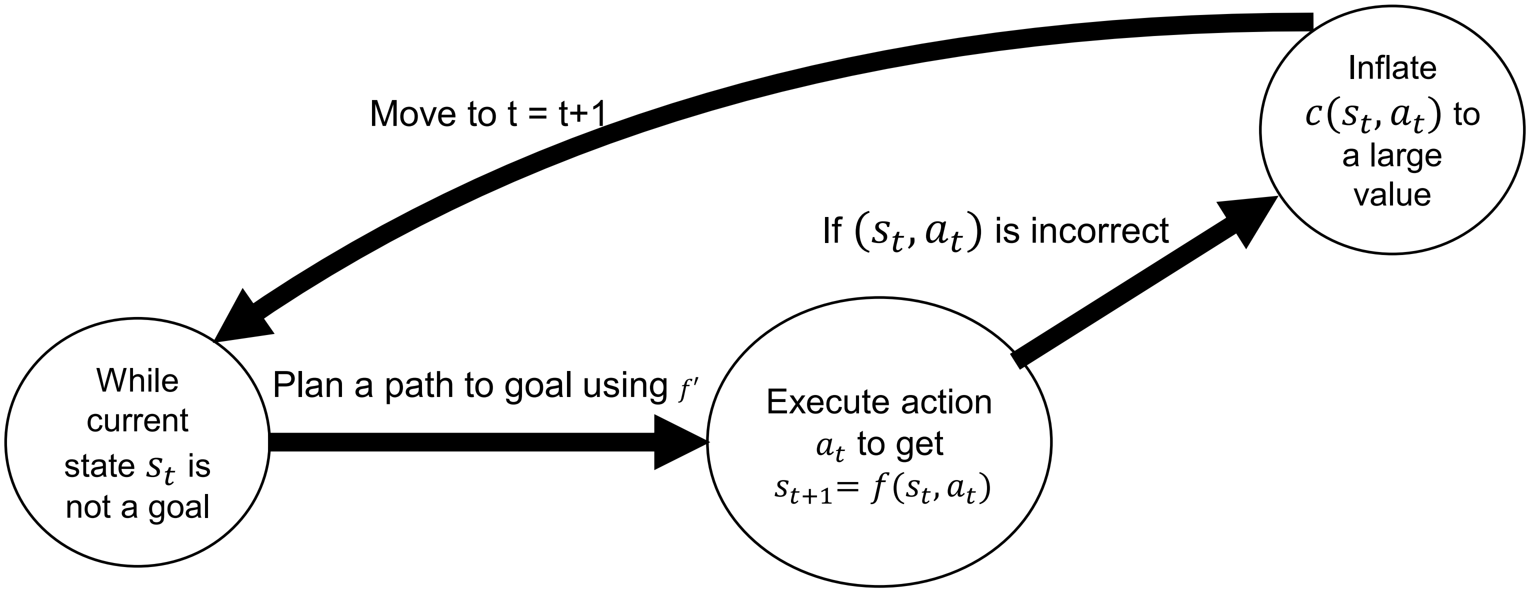
biases the planner to avoid using transitions known to be incorrect

- Key idea:

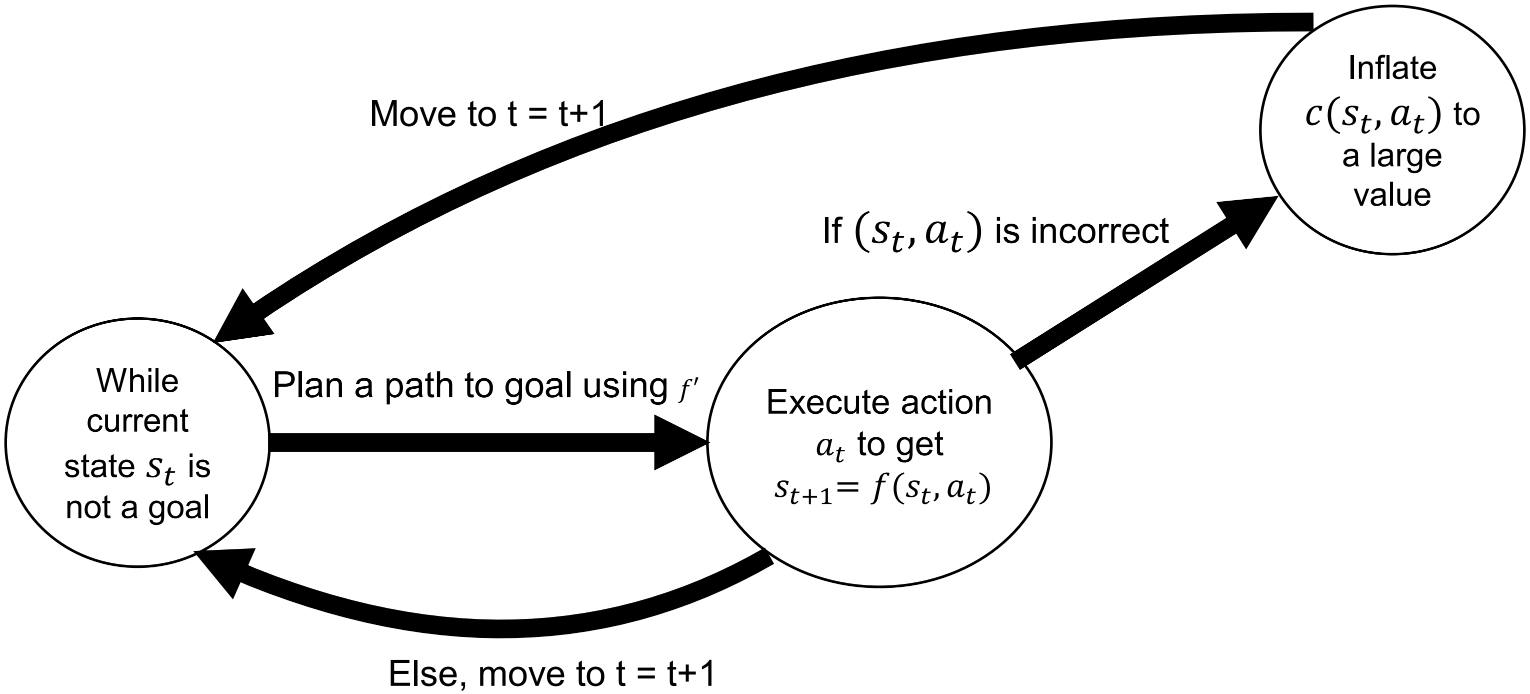

While current state s_t is not a goal
Plan a path to goal using f'
Execute action a_t to get $s_{t+1} = f(s_t, a_t)$

CMAX

- Key idea:



- Key idea:



Does not update approximate dynamics f' !

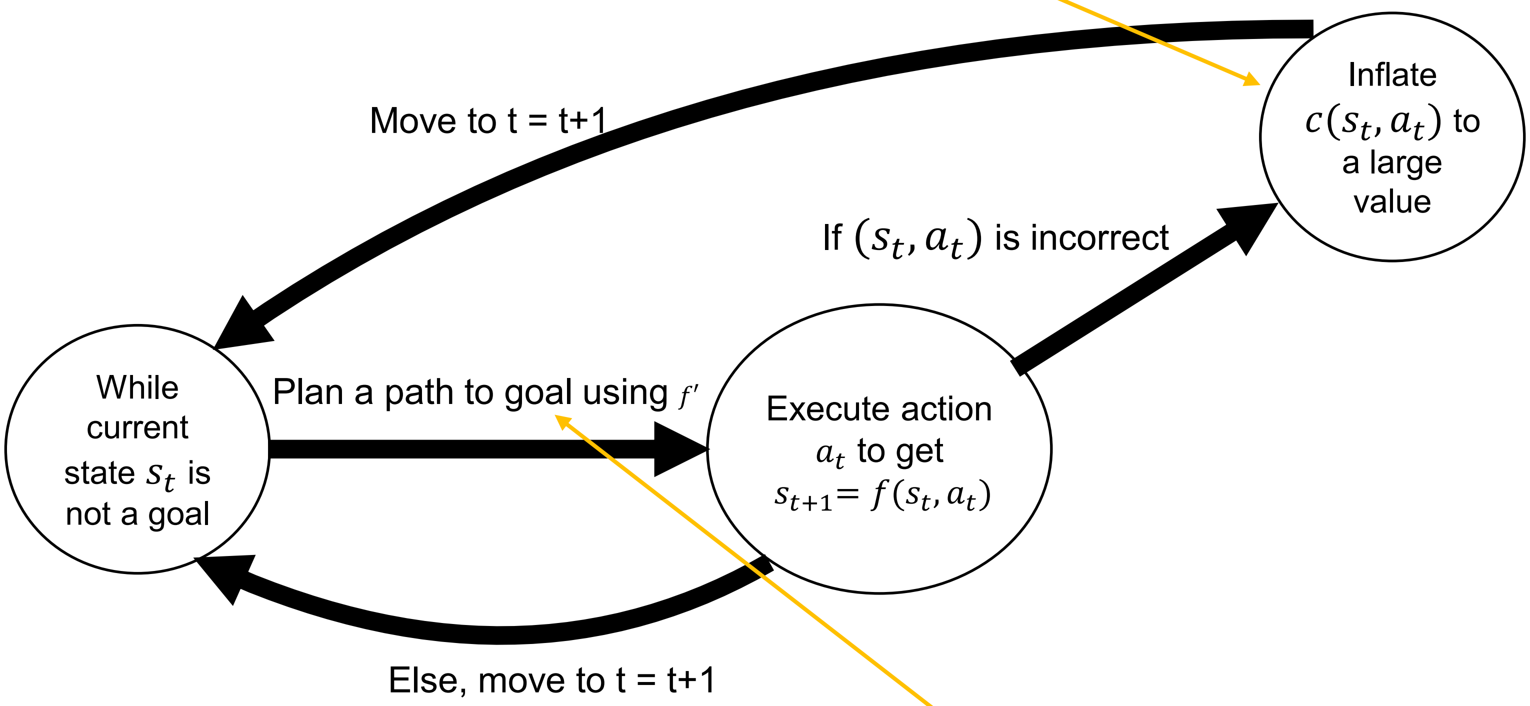
***Theorem**. The robot is guaranteed to reach a goal (accomplish its task), i.e. CMAX is task-complete, under the assumption that there always exists a path from s_t to a goal that does not contain any transitions (s, a) known to be incorrect, i.e. (s, a) ϵ $\mathcal{X}_t$*

CMAX

- Key idea:

To handle large state-spaces:

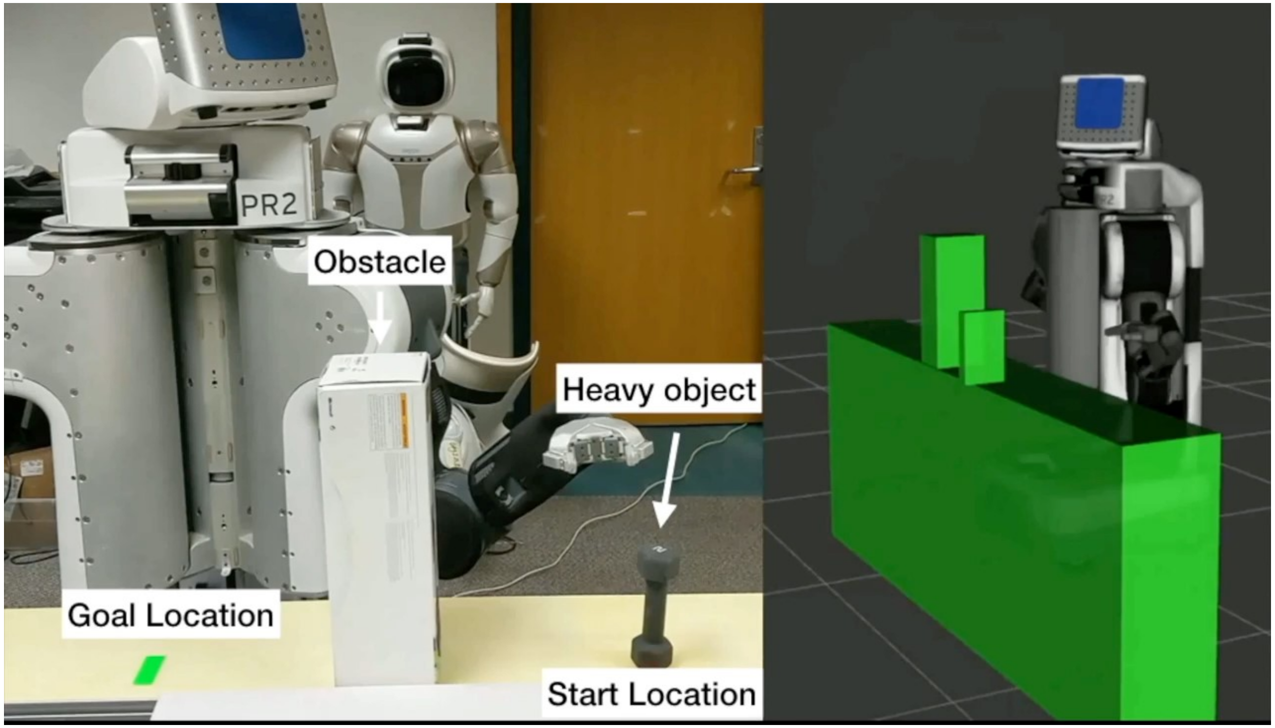
uses function approximators to learn where discrepancies are



To handle large state-spaces:

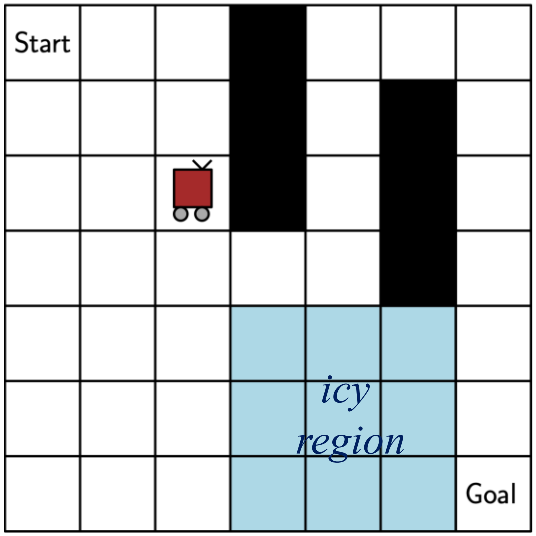
extended to support limited-horizon search (e.g., LRTA* [Korf 90])

- CMAX in action:

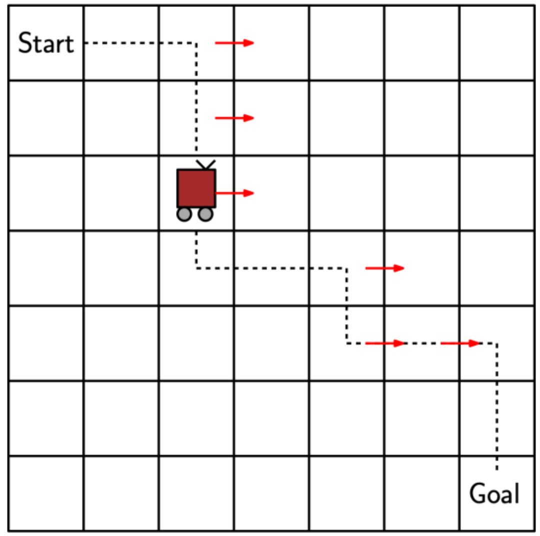


CMAX++

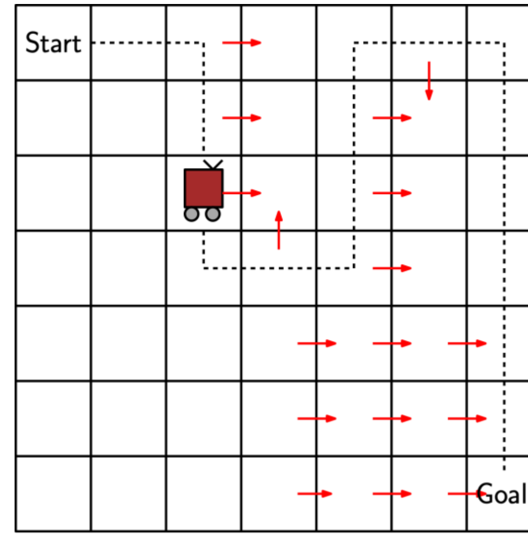
- Task achievement vs. optimality for repeated tasks



CMAX
after 1st iteration



CMAX
after nth iteration



potentially highly sub-optimal path

CMAX++

- Task achievement vs. optimality for repeated tasks

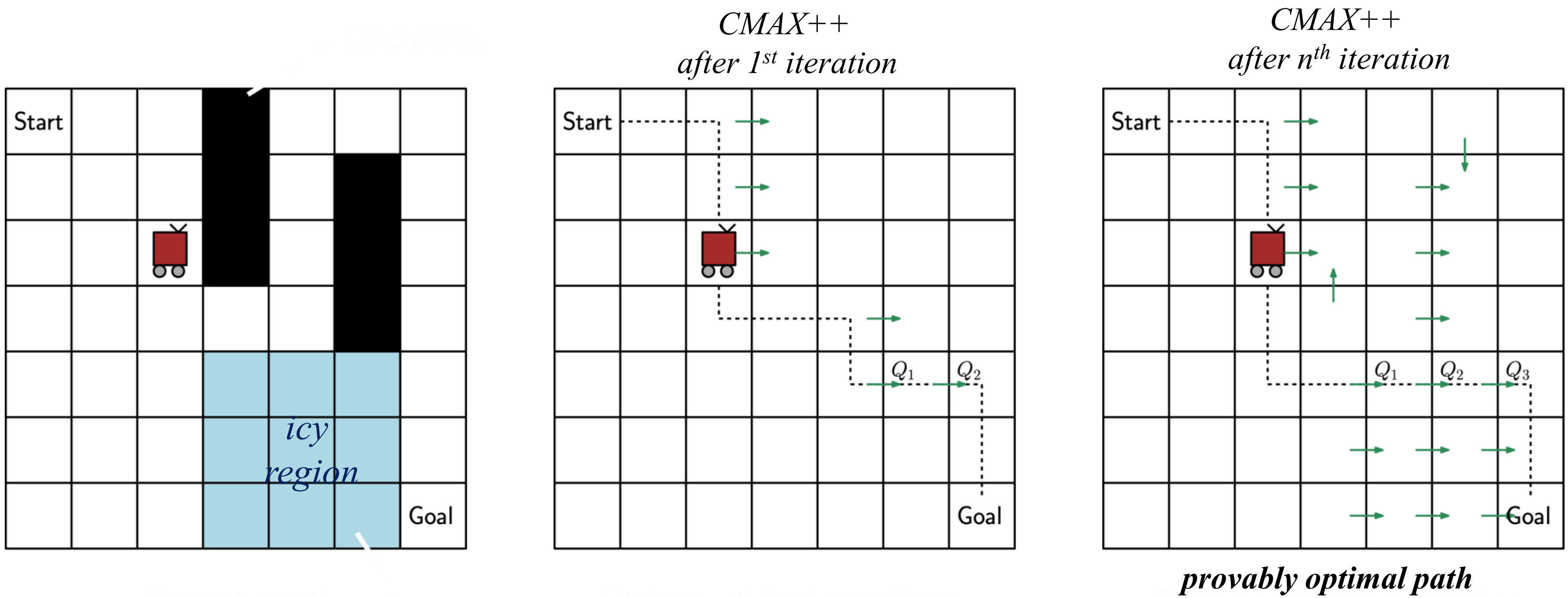
CMAX++ [Vemula, Bagnell & Likhachev, AAAI'21]

- *combines CMAX with model-free Q-learning*
- *learns optimal Q-values of incorrect transitions over time and slowly switches to using those during planning*

- ***Guarantees task achievement under certain conditions AND convergence to optimal execution of repeated tasks***

CMAX++

- Task achievement vs. optimality for repeated tasks



CMAX++ [Vemula, Bagnell & Likhachev, AAAI'21]

- *combines CMAX with model-free Q-learning*
- *learns optimal Q-values of incorrect transitions over time and slowly switches to using those during planning*

- ***Guarantees task achievement under certain conditions AND convergence to optimal execution of repeated tasks***

Agenda

- Challenges in Search-based Planning
- Constant-time Motion Planning (CTMP) – offline learning for online planning
- CMAX/CMAX++ for handling inaccurate models
- Summary and thoughts on research directions

Summary and Thoughts

Challenges in Search-based Planning:

- *high-dimensionality/graph size*
- *expensive edge cost evaluation*
- *edge construction for dynamic systems*
- *reliance on the accuracy of the model*

CTMP algorithms [Islam et al. 21]

CMAX [Vemula 22]

Summary and Thoughts

Challenges in Search-based Planning:

Challenges in learning to reduce reliance on the accuracy of the model:

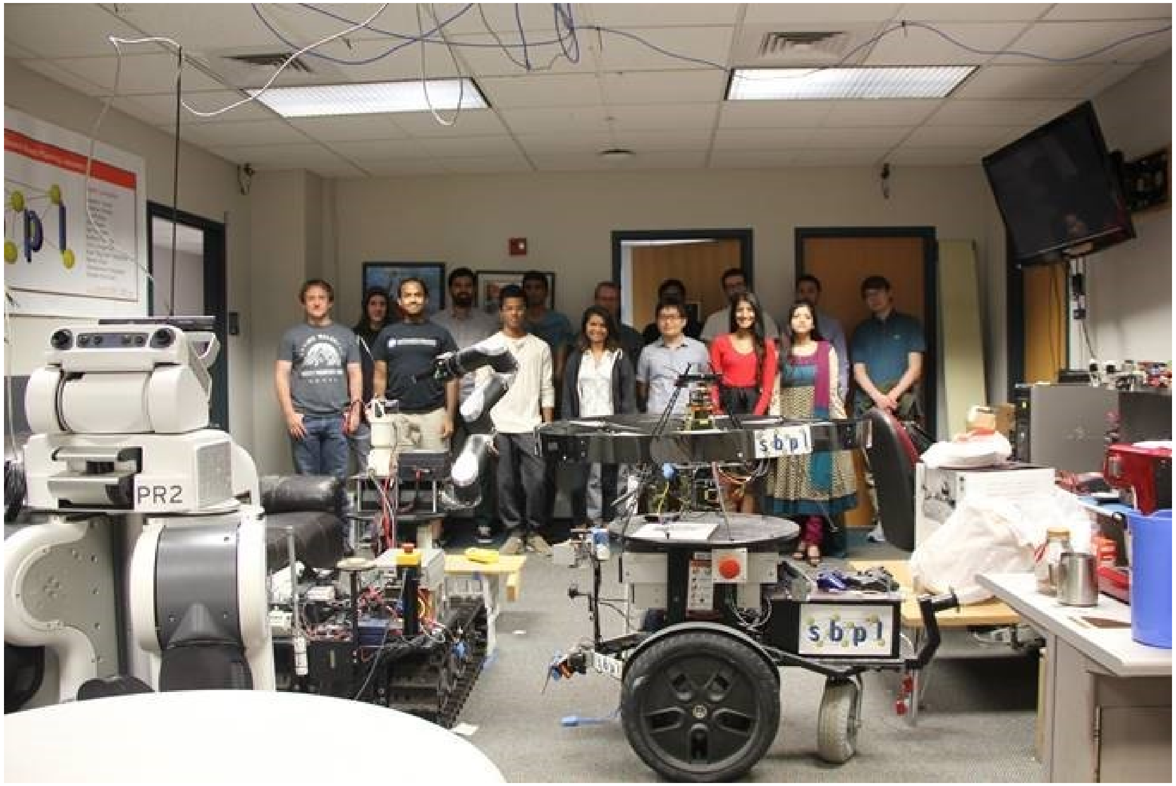
1) *How to do it while providing goal-directed behavior and task achievement guarantees?*
2) *How to combine learning from demos with model-based planning while guaranteeing safety?*
3) *How to get good confidence estimates in learned policies and how to incorporate those into planning?*

Two main goals:

1) *Learning with the aim of speeding up planning*
2) *Learning with the aim of reducing dependency on the accuracy of the model*

Thanks!

- All the students, postdocs and collaborators!



- Funding sources for this research:
  - ONR
  - ARL
  - NSF
  - DARPA
  - Mitsubishi
  - Honeywell