First Summer School on Cognitive Robotics

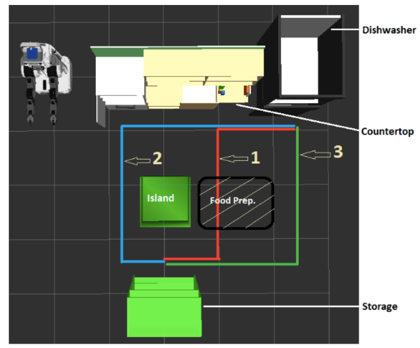
Sampling-Based Motion Planning

Mark Moll
Department of Computer Science
Rice University
Houston, TX

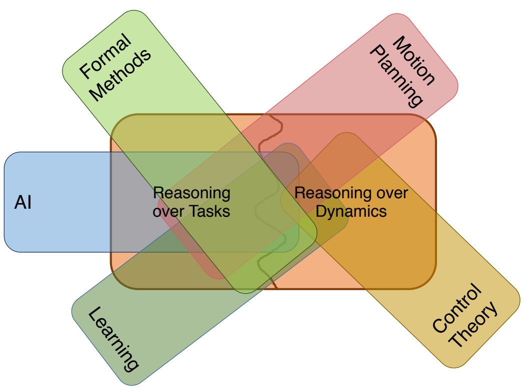
1

Combined task and motion planning

Motivating example: want all dirty dishes in the kitchen cleaned and stored

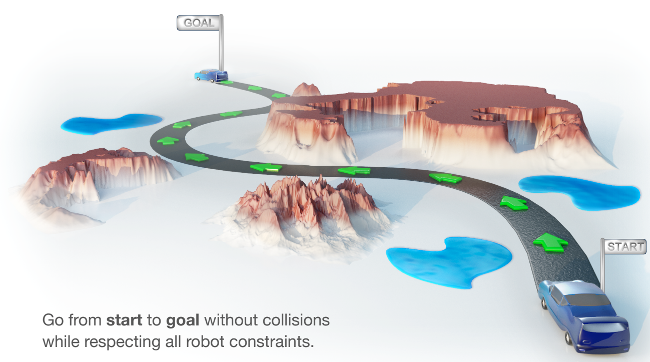


2



3

Motion planning: classical setting



Go from **start** to **goal** without collisions
while respecting all robot constraints.

4

Motion planning

- Is done in a continuous world and with constrained motions.
- Needs to know robot and world geometry.
- Needs to know robot and world physics.
- Must be accurate and predictive to work in practice.

Some notes:

- More powerful motion planning simplifies the task planner.
- More accurate motion planning simplifies motion execution.
- Motion planning is limited by model accuracy and complexity.

5

Motion planner is part of a replanning loop

Bekris et al.

6

Motion planning problems are hard

PROBLEM	COMPLEXITY
Geometric Constraints:	
Sofa Mover (3DOF)	$O(n^{2+\epsilon})$ - not implemented [HS96]
Piano Mover (6DOF)	Polynomial – no practical algorithm [SS83]
n Disks in the Plane	NP-Hard [SS83]
n Link Chain in 3D	PSPACE-Complete [HSS87]
Generalized Mover	PSPACE-Complete [Canny88]
Dynamics Constraints:	
Point with Newtonian Dynamics	NP-Hard [DXCR93]
Polygon Dubin's Car (Linear)	Decidable [CPK08]
Nonlinear	Unknown, probably undecidable
Discrete Transitions and Dynamics Constraints:	
Hybrid Systems	Undecidable [Alur et al. 95]

Exact, approximate, and probabilistically complete algorithms

Method	Advantage	Disadvantage
exact	theoretically insightful	impractical
cell decomposition	easy	does not scale easily
control-based	online, very robust	requires good trajectory
potential fields	online, easy	slow or fail
sampling-based	**fast and effective**	**cannot recognize impossible query**

Lecture outline

1. Overview of sampling-based robot motion planning
2. Integrated task and motion planning
3. Overview of the Open Motion Planning Library (OMPL)

Overview of sampling-based robot motion planning

Basic concepts and definitions

- Workspace
- Robot
- State space
- Path
- Planning Instance
- Query/Problem

Workspace

- The **workspace** is the environment that the robot operates in.
- The boundary of the workspace determines the **obstacles**.

Robot

A robot is defined by:

- Geometry
- Parameters or **Degrees of Freedom (DOF)**
  - Different settings for the parameters embed the geometry in different ways into the workspace.

State space

- The parameter space for the robot is called the **state space S**.
- A point in this space is a **state**.

Free state space

- A state is **free** if the corresponding embedding of the robot's geometry lies in the workspace.
- The subspace of free configurations is **free state space S_{free}**.
- **S_{free}** can be very complex even for seemingly simple systems.
- This complexity is the main difficulty in motion planning.

Paths

- A **path** is continuous mapping in S

$$\pi : [0, L] \rightarrow S_{free}$$

- L is the **length** of the path.
- The path is **collision-free** if for all t

$$\pi(t) \in S_{free}$$

Planning instance

A planning instance consists of:

- Robot (S-space and embedding).
- Workspace.
- Constraints.

Query / problem definition

- A problem or **query** is:
  - Given two states, q_0 and q_f, determine if there is a collision-free path between q_0 and q_f.

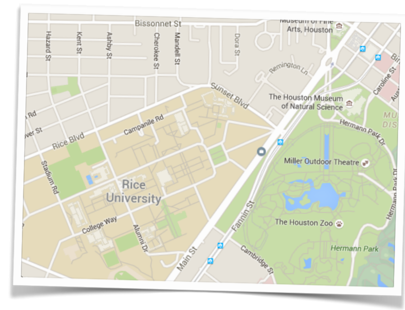
Probabilistic Roadmap Planner (PRM)

Kavraki, Svestka, Overmars and Latombe, 1996

19

PRM

- Uses random sampling.
- Uses simple local planner.
- Builds a roadmap of the state space.



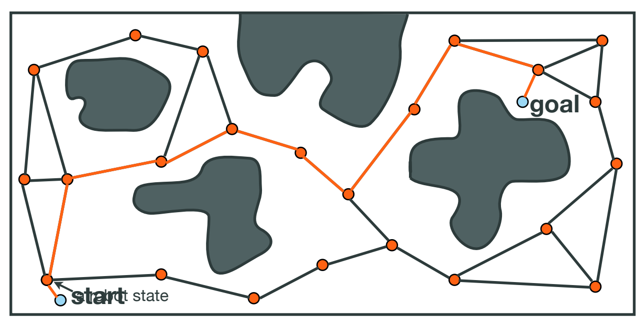
20

PRM

- Illustrate with an easy planning instance/problem set up.
  - Robot is a point in 2D.
  - Robot moves freely.
  - Simple example used for illustration only.
- Isolate primitive techniques.
- Generalize.

21

Point robot in 2-D



: nodes: random states
: edges: paths computed by local planner
plan a path: 1. connect start & goal to roadmap
2. perform graph search

22

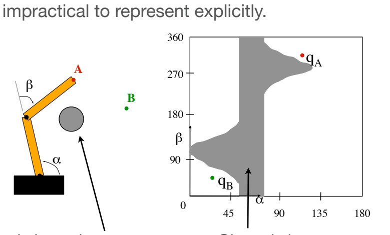
Primitive techniques

- **Select Sample:** (in the example) Uniform sampling to get milestones.
- **Connect:** (in the example) Local planner uses "straight lines."
- **Store in some data structure:** (in the example) A graph.
  - A **roadmap** is finite graph G=(V,E)
  - V is a subset of S_{free}.
  - (s_1,s_2) in E implies that the local planner found a path.

23

Why use sampling?

- S_{free} is impractical to represent explicitly.

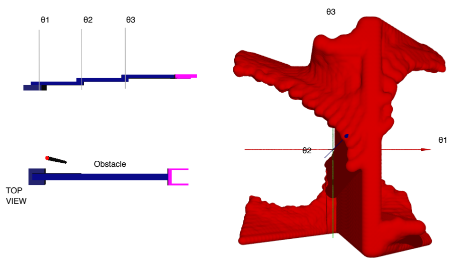


Obstacle in workspace
Obstacle in state space

24

Why use sampling?

- S_{free} is impractical to represent explicitly.

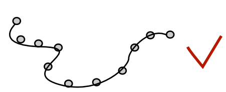


Why use sampling?

- S_{free} is impractical to represent explicitly.
- Sampling can be very efficient.
- Resulting data structure can be very compact.

Connecting samples

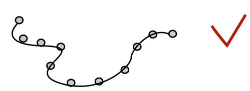
- An example of a simple planner:
  - Computes the straight line path between q1, q2.
  - Checks to see if it is valid.
  - If so, returns SUCCESS and the path.
  - Otherwise, returns FAIL.



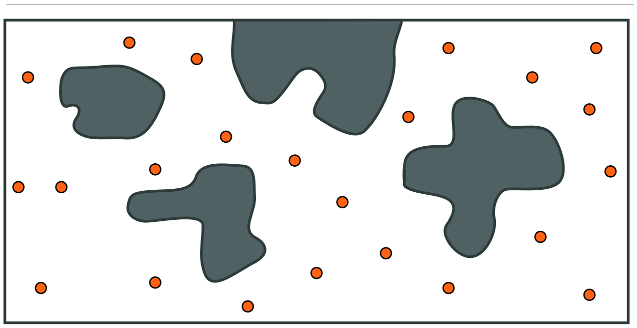
May fail often

State validity checker

- For states
  - Use e.g., collision checking, check any bounds
- For paths
  - State validation along a path is done by recursive refinement.
  - Bounds on clearance are combined with bounds on motion to cover the path with open balls or find a collision.

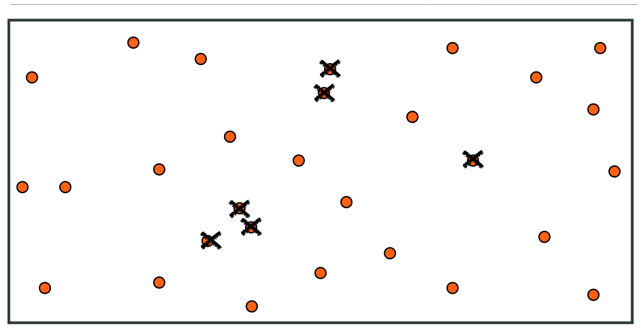


Operation of PRM



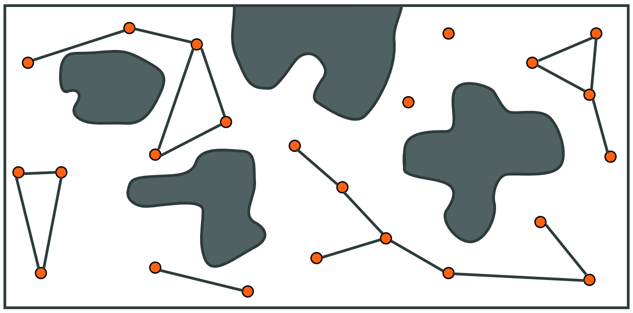
: nodes, random states

Operation of PRM



: states

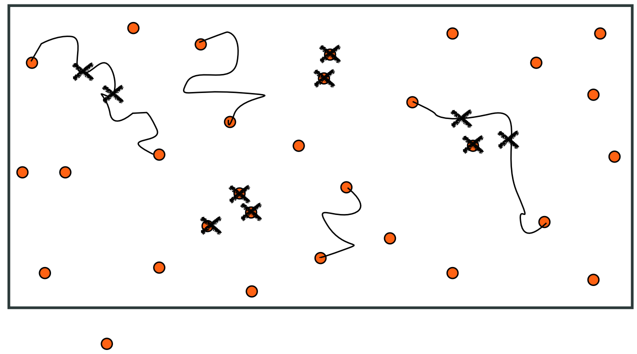
Operation of PRM



:**edges,** paths computed by local planner

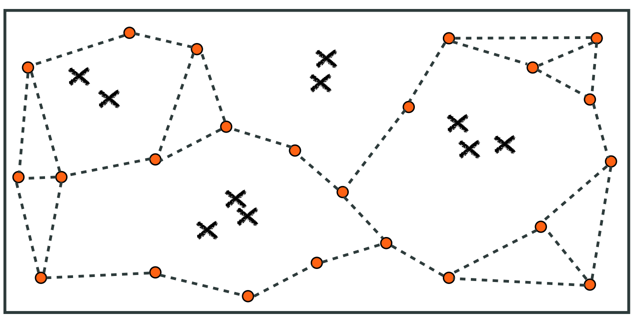
31

Operation of PRM



32

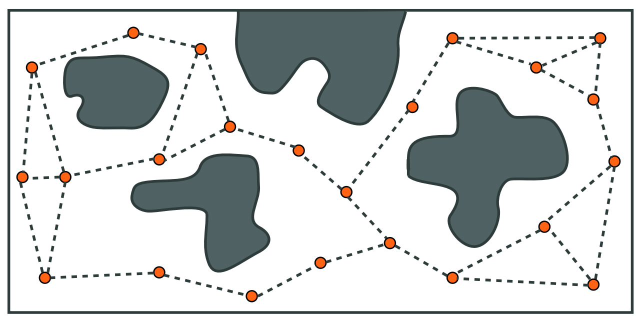
Operation of PRM



feasible path computed by local planner

33

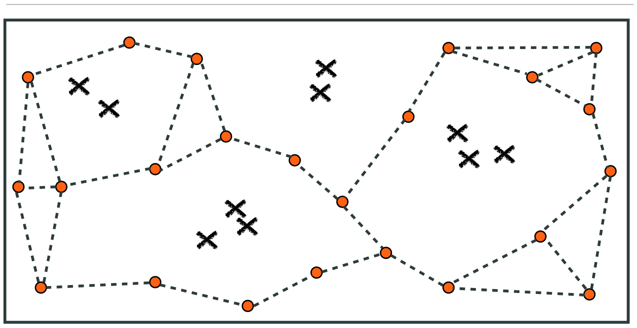
Operation of PRM



feasible path computed by local planner

34

Operation of PRM



feasible path computed by local planner

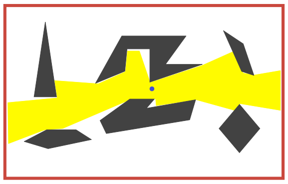
35

Completeness of PRM

- If no path exists, then PRM cannot find the path.
- But… if a path exists, it is possible PRM fails to find it.
- PRM is not complete but instead is **probabilistically complete**.

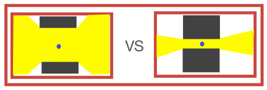
36

Theoretical analysis of PRM (1/2)



[Kavraki et al 96, 98, 00, 03, 07]

ε-goodness property



- Tradeoff: planner may fail with probability α
- Number of nodes/states:

$$N \approx \frac{1}{\epsilon}[log(\frac{1}{\epsilon}) + log(\frac{4}{\alpha})]$$

- Important: Performance related to properties of the space

37

Theoretical analysis of PRM (2/2)

- We sacrifice completeness for speed
- **Probabilistic completeness**
- Novel analysis and performance guarantees

$$Pr(failure) = f(e^{-cN})$$

- How much can the assumptions be relaxed?

38

Primitive techniques

39

Primitives

- **Select Sample:** Uniform sampling is general but not the most efficient.
  - Optimal selection remains elusive.
- **Connect:** Connect all to all is general but not efficient.
  - Neighbors
  - Notion of “straight line” or other local plan needs to be adapted.
- **Store efficiently**

40

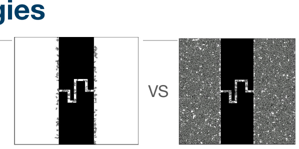
Primitives

- **Select Sample:** Uniform sampling is general but not the most efficient.
  - Optimal selection remains elusive.
- **Connect:** Connect all to all is general but not efficient.
  - Neighbors
  - Notion of “straight line” or other local plan needs to be adapted.
- **Store efficiently**

41

Several sampling strategies

- Gaussian sampling [Overmars et al.]:
  - Places samples close to objects.
  - Distribution is Gaussian around the obstacle boundary.



- Medical-axis sampling [Amato et al.]
- Bridge Test sampling for narrow corridors [Hsu et al.]
- Quasi-Random sampling [LaValle et al.]
- Selective sampling [Kavraki et al.]

One of the most critical parts of the planner [Hsu, Latombe 1998].

42

Primitives

- **Select Sample:** Uniform sampling is general but not the most efficient.
  - Optimal selection remains elusive.
- **Connect:** Connect all to all is general but not efficient.
  - Neighbors
  - Notion of "straight line" or other local plan needs to be adapted.
- **Store efficiently**

43

Several connection strategies

- **Nearness:** Try to connect each configuration to a constant number of "nearby" configurations.
  - nearest neighbors by kd-trees, k-NN, k-ANN
  - random neighbors may be helpful
- **Component technique:** Only test edges which reduce the number of connected components in the roadmap.

[Svestka, Overmars, 1996]

44

Primitives

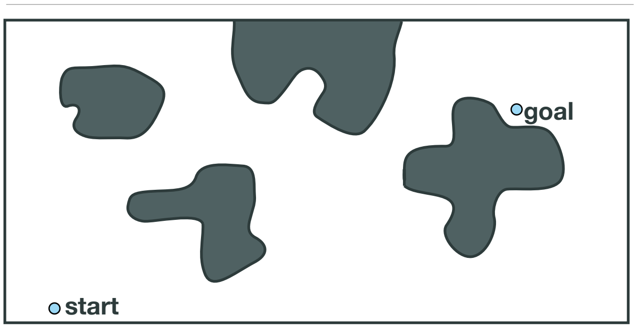
- **Select Sample:** Uniform sampling is general but not the most efficient.
  - Optimal selection remains elusive.
- **Connect:** Connect all to all is general but not efficient.
  - Neighbors
  - Notion of "straight line" or other local plan needs to be adapted.
- **Store efficiently**

45

A generic sampling-based tree planner

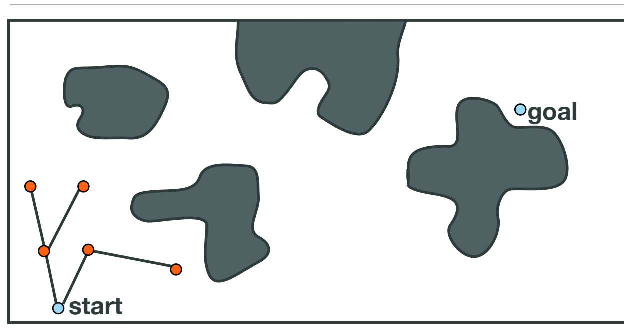
46

Sampling-based tree planner operation



47

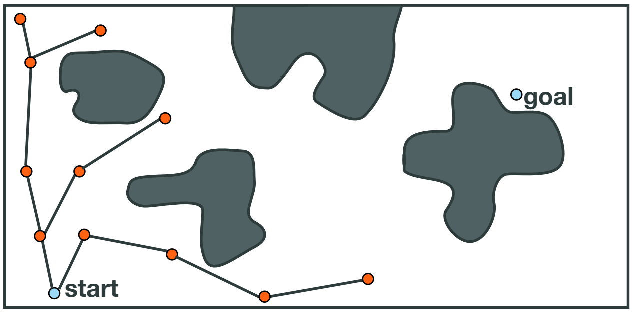
Sampling-based tree planner operation



grow random tree from **start**

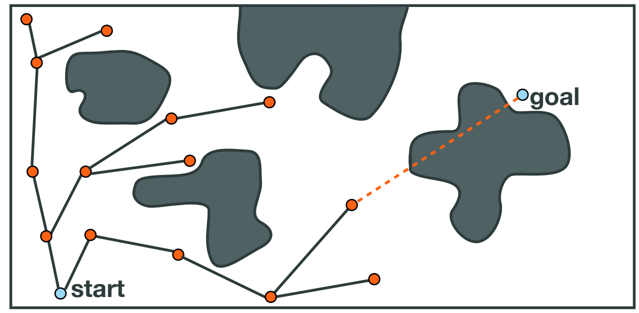
48

Sampling-based tree planner operation



49

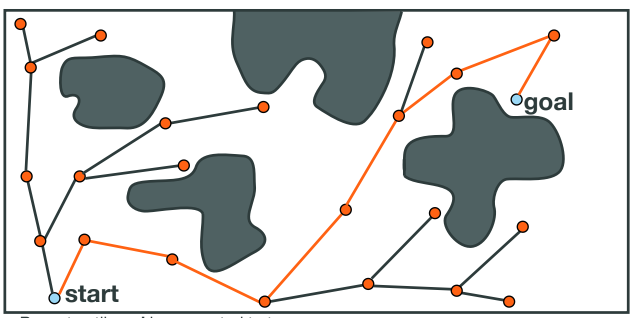
Sampling-based tree planner operation



: occasionally attempt to connect tree to **goal**

50

Sampling-based tree planner operation



- Repeat until **goal** is connected to tree.
- Bi-directional trees are possible when considering only geometric constraints.

51

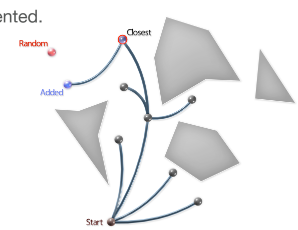
Primitives

- **Select Sample**
- **Expand from the sample**
- **Store efficiently**

52

Rapidly Exploring Random Trees (RRT)

- Uses proximity query to guide construction (Voronoi Bias).
- Uses propagation instead of connection.
- Powerful heuristic for single-query planning.
- Bi-directional search can be implemented.

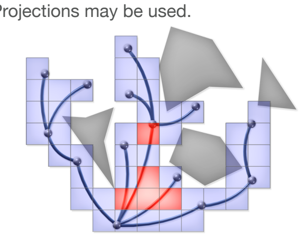


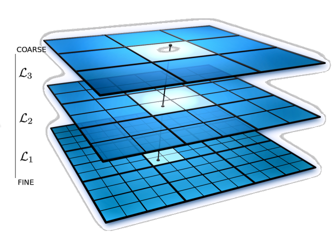
[Lavalle, Kuffer 1999, 2000]

53

KPIECE

- Keeps track of coverage by using discretization and by distinguishing the boundary from the covered space.
- Keeping track of coverage can be done in a hierarchical fashion.
- Projections may be used.





[Şucan, Kavraki 2008]

54

Performance improvements for trees

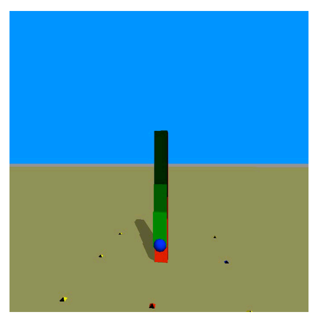
- Bi-directional search.
- Lazy collision checking.
- Goal biasing.
- Accounting for constraint manifolds.
- Employing motion primitives.
- and many others.

Optimal Paths

- Most sampling-based planning algorithms produce **feasible** rather than **optimal** paths.
- Two approaches to achieving optimality:
  - **Local** optimization of path in post-processing step (short-cutting, smoothing, etc.)
  - **Global** optimization: connect to “enough” neighbors, “rewire” tree as nodes are added.
    → provably asymptotic (near-)optimal paths!

Planning with dynamics: Trees offer an advantage

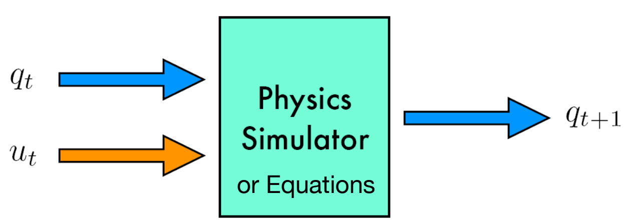
Planning with dynamics



Ladd et al.

Bekris et al.

Physical system planning



$$F(q_t, u_t) = q_{t+1}$$

Space of controls is defined

Physical system planning

Given

1. an initial state $q_0 \in Q$
2. a goal set $G \subset Q$

The discrete physical systems planning problem is to compute a sequence $u_0, \ldots, u_N$ such that:

$$F(q_i, u_i) = q_{i+1}$$

and $q_{N+1} \in G$ is contained in the goal set.

Planning with dynamics

- Adding dynamics is essential to increase physical realism.
- Techniques from control theory can be used to create better paths or reduce differential equation integrations.
- Metrics tend to work poorly.
- Efficient planning for systems with dynamics is still fairly open: sampling-based tree planners offer an advantage.

Primitives

- **Select Sample**
- **Expand from the sample**
- **Store efficiently**

These primitives are combined with various optimizations.

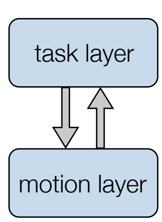
Lecture outline

1. Overview of sampling-based robot motion planning
2. **Integrated task and motion planning**
3. Overview of the Open Motion Planning Library (OMPL)

Integrated task and motion planning

Integrated task and motion planning (TMP)

- **Task Planning**
  - **Discrete** abstraction of the robot/environment
  - **PRO:** efficiently plans in abstractions
  - **CON:** continuous domains hard to abstract
- **Motion Planning**
  - Collision free path in **continuous** space
  - **PRO:** practical planning algorithms
  - **CON:** computational limits on dimensionality / plan length
- **Integrated Task and Motion Planning**
  - Task Planning reduces search space for Motion Planner
  - Motion Planning can help guide abstraction for Task Planner



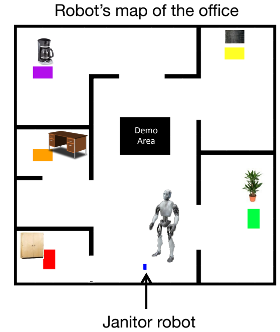
Moving beyond "reach the destination" paradigm

Mission:

"Do the following in any order and always avoid the black demo area

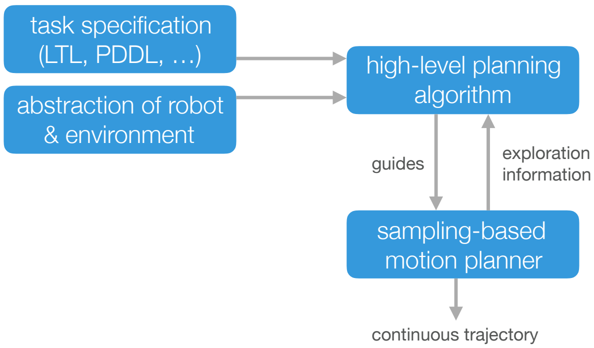
- water the plant
- clean the blackboard
- turn off the coffee maker
- pick up vacuum cleaner from the supply room, then vacuum the orange room and dust its table."

Robot's map of the office



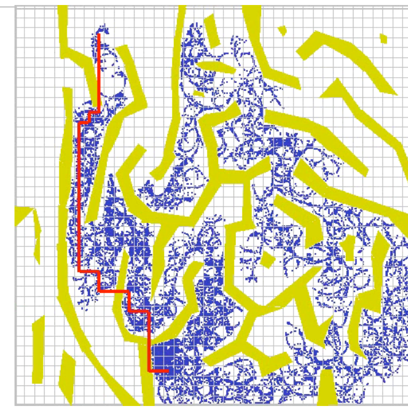
iRobot image: http://i.imgur.com/Y0SITTQ.png

Overview of our approach:
A synergistic framework for task & motion planning



67

Basic example of synergistic framework:



68

Synergistic task & motion planning

Our group has explored two different approaches:

- **Linear Temporal Logic task specifications + complex dynamics:**
  - Use state space abstraction + automaton to construct guides
  - Sampling-based planning around guides, exploration progress feedback
  - ➡ *See LTLPlanner in OMPL*
- **PDDL + Incremental SMT Solvers:**
  - Generate task plans of fixed length
  - Attempt to find corresponding motion plans
  - If no solution found, increase task plan length
  - ➡ *See TMKit at* http://tmkit.kavrakilab.org

69

Moving beyond "reach the destination" paradigm

Mission:

"Do the following in any order and always avoid the black demo area

- water the plant
- clean the blackboard
- turn off the coffee maker
- pick up vacuum cleaner from the supply room, then vacuum the orange room and dust its table."

70

The framework is powerful

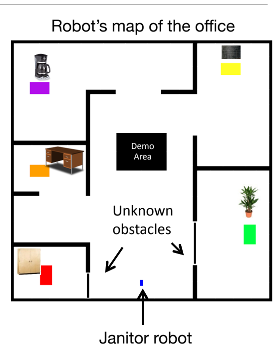
Safe LTL ✓

Hybrid systems ✓

Changes in the environment ✓
(partial satisfaction)

Manipulation ✓

Motion uncertainty ✓



71

Lecture outline

1. Overview of sampling-based robot motion planning
2. Integrated task and motion planning
3. **Overview of the Open Motion Planning Library (OMPL)**

72

The Open Motion Planning Library

73

OMPL in a nutshell

- Common core for sampling-based motion planners
- Includes commonly-used heuristics
- Takes care of many low-level details often skipped in corresponding papers
- Intended for use in:
  - Education
  - Research
  - Industry

74

Other related robotics software

ROS MoveIt!

OpenRAVE

v-rep virtual robot experimentation platform

Robotics Library

MORSE the Modular OpenRobots Simulation Engine

VEROSIM SOLUTIONS

All of them support OMPL!

75

Abstract interface to core sampling-based motion planning concepts

- state space / control space
- state validator (e.g., collision checker)
- sampler
- goal (problem definition)
- planner
- ...

except robot & workspace...

76

Many planners available in OMPL

Planner

geometric planning

KPIECE, BKPIECE, LBKPIECE
PRM, LazyPRM
RRT, RRTConnect, LazyRRT
EST, SBL
PDST
STRIDE

Optimizing planners:
PRM*
RRT*, FMT*, BIT*
T-RRT
SPARS, SPARS-2

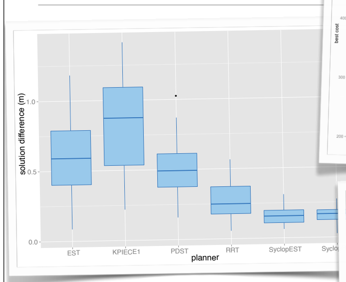
planning with controls

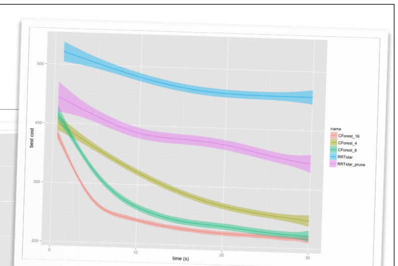
KPIECE
RRT
EST
Syclop
PDST
LTLPlanner

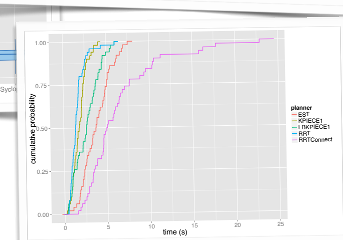
Optimizing planner:
SST

77

Benchmarking



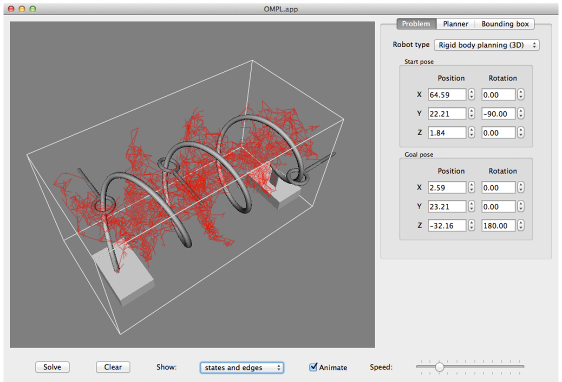




http://plannerarena.org

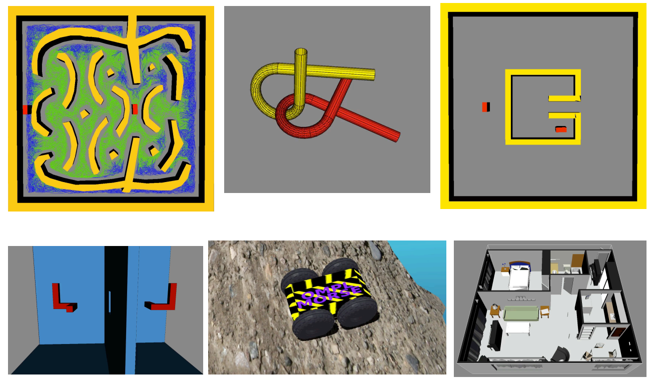
78

OMPL.app

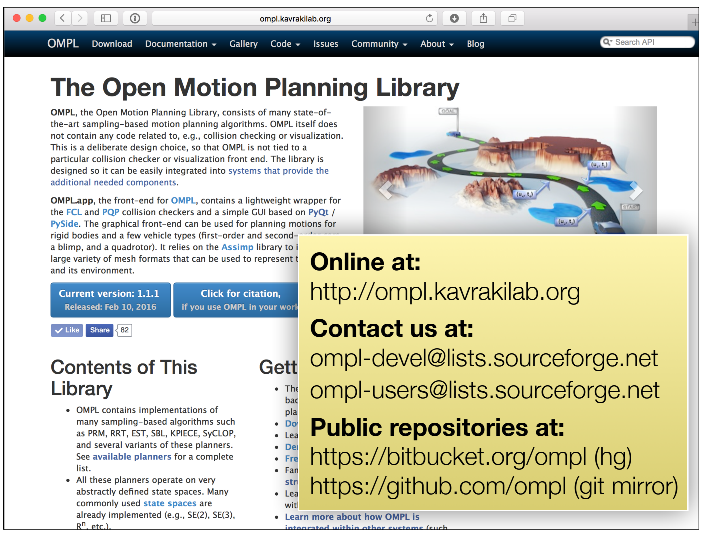


79

Sample OMPL.app problems



80



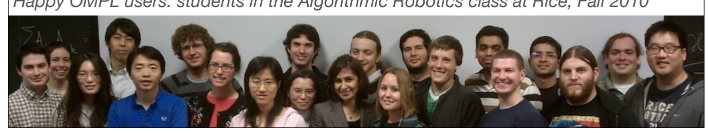
Online at:
http://ompl.kavrakilab.org
Contact us at:
ompl-devel@lists.sourceforge.net
ompl-users@lists.sourceforge.net
Public repositories at:
https://bitbucket.org/ompl (hg)
https://github.com/ompl (git mirror)

81

OMPL for education

- Programming assignments centered around OMPL, available upon request.
- Educational assessment.
  Moll et al., *Comp. Sci. Education* 23(4):332–348, 2013
- Already in use in several robotics / motion planning classes.

Happy OMPL users: students in the Algorithmic Robotics class at Rice, Fall 2010



82

ROS MoveIt!

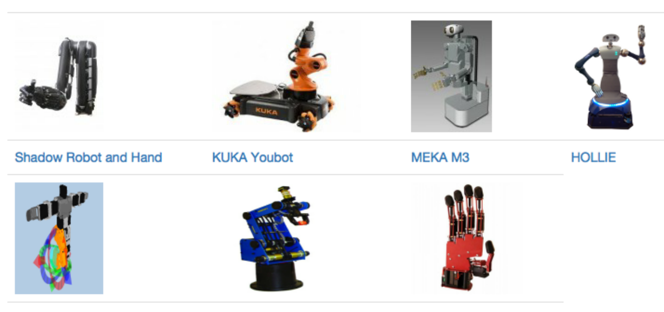
- motion planning (using OMPL)
- kinematics
- collision checking integrated with perception
- grasping
- control and navigation for mobile manipulation

83

ROS MoveIt!

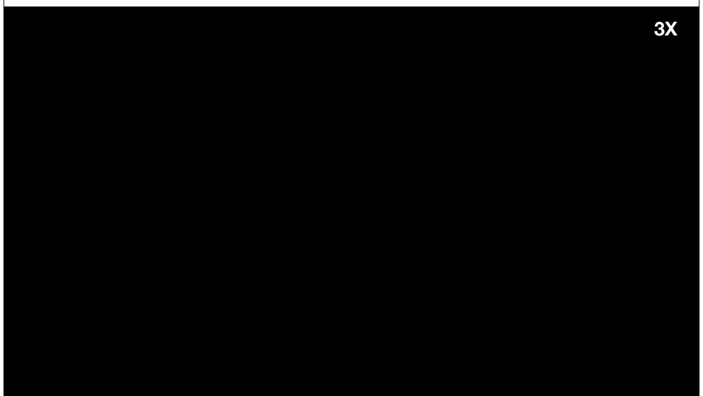
http://moveit.ros.org/robots

Clopema Robot DLR-Hit Hand iCub REEM-C



84

Robonaut 2:
34 degrees of freedom and many constraints



Summary

- Sampling-based motion planning provides effective approach to search high-dimensional, continuous state space of robots.
- Task planning can help guide motion planning,
  motion planning can help refine abstractions used in task planning.
- OMPL provides generic implementations of sampling-based algorithms,
  MoveIt! provides the “glue” to run them on real robots.

THANK YOU!

Mark Moll
mmoll@rice.edu
http://mmoll.rice.edu

Acknowledgements: this work has been supported by the National Science Foundation