# Motion and Path Planning
# – Sampling-based methods –
## CSC398 Autonomous Robots

Ubbo Visser

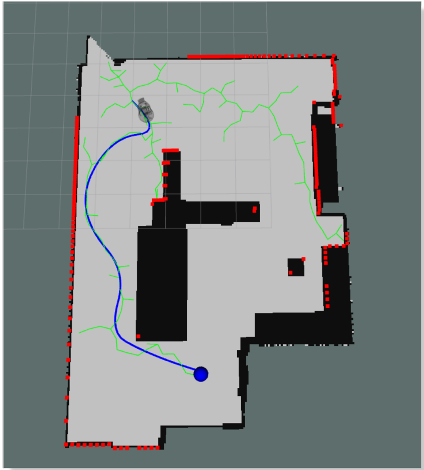
Department of Computer Science
University of Miami

October 31, 2024

UNIVERSITY
OF MIAMI

# Outline

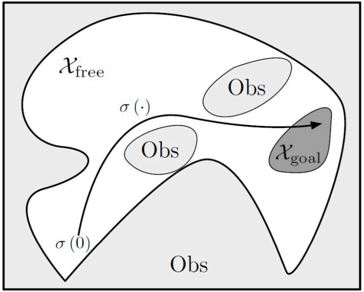




HSRB in the RoboCanes lab, Isaac-Sim simulator, RViz

# Motion and Path Planning

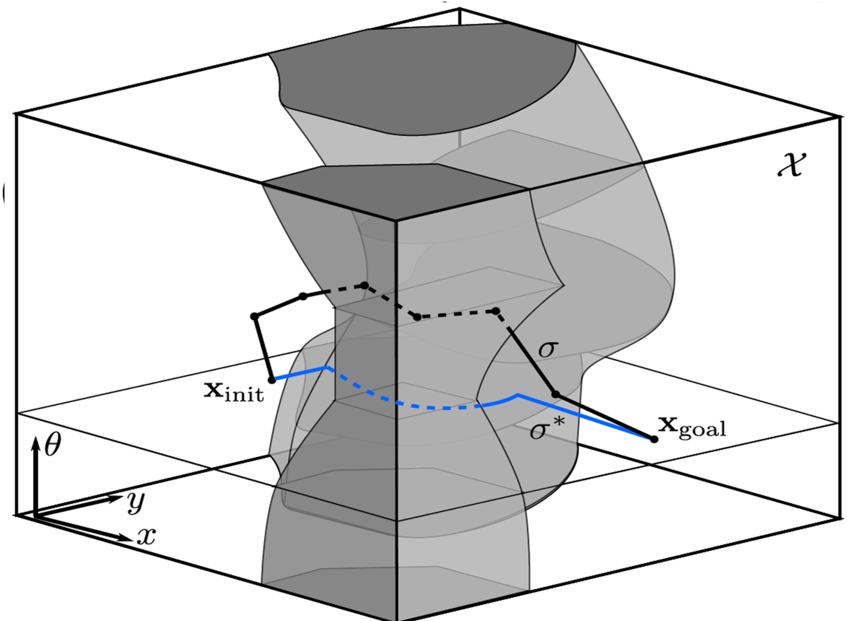
- **Definition:** Compute sequence of actions that drives a robot from an initial condition to a terminal condition while avoiding obstacles, respecting motion constraints, and possibly optimizing a cost function.
- **Aim:** Learn about sampling-based motion planning algorithms




**Suggested Readings:**

- *Planning Algorithms*, Chapter 5, Steven M. LaValle (2006), Cambridge University Press.

# Configuration space



# Motion planning algorithms

- Key point: motion planning problem described in the real-world, but it really lives in an another space - the configuration (C-)space!
- Two main approaches to continuous motion planning:
  - *Combinatorial planning:* constructs structures in the C-space that discretely and completely capture all information needed to perform planning
  - *Sampling-based planning:* uses collision detection algorithms to probe and incrementally search the C-space for a solution, rather than completely characterizing all of the $C_{free}$ structure

# Sampling-based motion planning

Limitations of combinatorial approaches stimulated the development of sampling-based approaches

- Abandon the idea of explicitly characterizing $C_{free}$ and $C_{obs}$
- Instead, capture the structure of $C$ by random sampling
- Use a black-box component (collision checker) to determine which random configurations lie in $C_{free}$
- Use such a probing scheme to build a roadmap and then plan a path

Source: LaValle, S. M. Motion planning. 2011

# Sampling-based motion planning

Pros:

- Conceptually simple
- Relatively easy to implement
- Flexible: one algorithm applies to a variety of robots and problems
- Beyond the geometric case: can cope with complex differential constraints, uncertainty, etc.

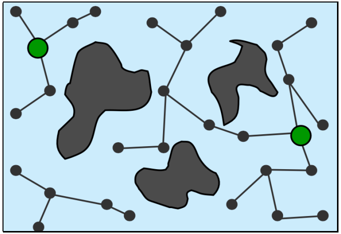
Cons:

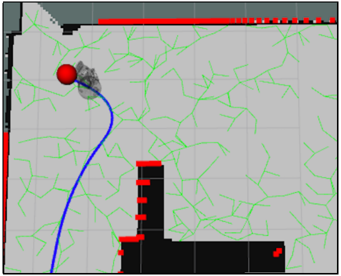
- Unclear how many samples should be generated to retrieve a solution
- Can not determine whether a solution does not exist

# Review of sampling-based methods

Traditionally, two major approaches:

- Probabilistic Roadmap (PRM): graph-based
  - Multi-query planner, i.e., designed to solve multiple path queries on the same scenario
  - Original version: [Kavraki et al., '96]
  - "Lazy" version: [Bohlin & Kavraki, '00]
  - Dynamic version: [Jaillet & T. Simeon, '04]
  - Asymptotically optimal version: [Karaman & Frazzoli, '11]
- Rapidly-exploring Random Trees (RRT): tree-based
  - Single-query planner
  - Original version: [LaValle & Kuner, '01]
  - RDT: [LaValle, '06]
  - SRT: [Plaku et al., '05]
  - Asymptotically optimal version [Karaman & Frazzoli, '11]





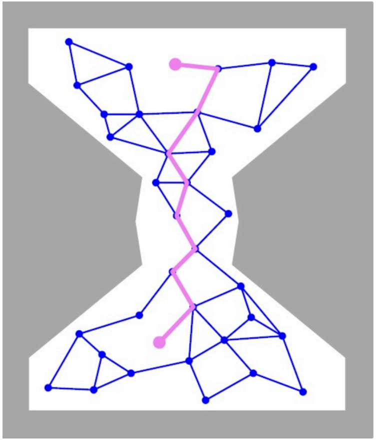
# Probabilistic roadmaps (PRM)

A multi-query planner, which generates a roadmap (graph) $G$, embedded in the free space

Preprocessing step:

1. Sample a collection of $n$ configurations $X_n$; discard configurations leading to collisions
2. Draw an edge between each pair of samples $x, x' \in X_n$ such that $||x - x'|| \leq r$ *and* straight-line path between $x$ and $x'$ is collision free

Given a query $s, t \in C_{free}$, connect them to $G$ and find a path on the roadmap



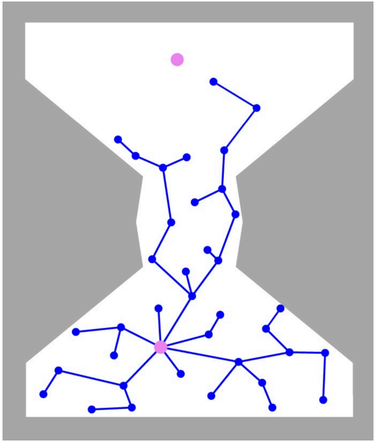
# Rapidly-exploring random trees (RRT)

A single-query planner, which grows a tree $T$, rooted at the start configuration $s$, embedded in $C_{free}$

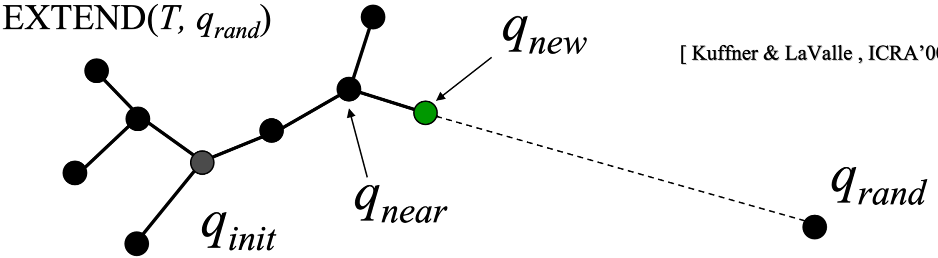
Algorithms works in $n$ iterations:

1. Sample configuration $x_{rand}$
2. Find nearest vertex $x_{near}$ in $T$ to $x_{rand}$
3. Generate configuration $x_{new}$ in direction of $x_{rand}$ from $x_{near}$, such that $\overline{x_{near}x_{new}} \subset C_{free}$
4. Update tree: $T = T \cup \{x_{new}, (x_{near}, x_{new})\}$

Every once in a while, set $x_{rand}$ to be the target vertex $t$; terminate when $X_{new} = T$



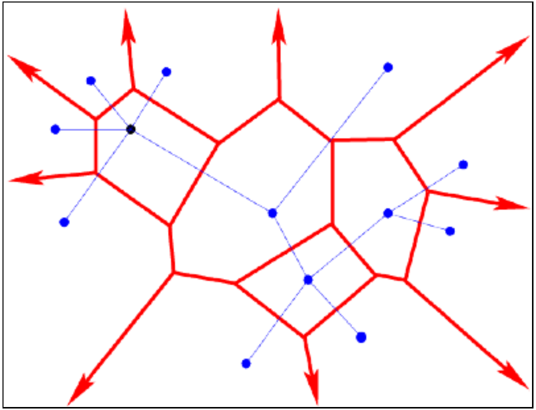
# Rapidly-exploring random trees (RRT)




[ Kuffner & LaValle , ICRA'00]

# Rapidly-exploring random trees (RRT)

- RRT is known to work quite well in practice
- Its performance can be attributed to its Voronoi bias:
  - Consider a Voronoi diagram with respect to the vertices of the tree
  - For each vertex, its Voronoi cell consists of all points that are closer to that vertex than to any other
  - Vertices on the frontier of the tree have larger Voronoi cells – hence sampling in those regions is more likely



Source: Arnold et al., 2013

# Theoretical guarantees: probabilistic completeness

Question: how large should the number of samples $n$ be? We can say something about the asymptotic behavior:

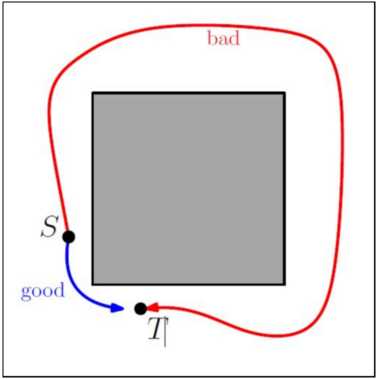
**Kavraki et al. '96**: PRM, with $r = \text{const}$, will eventually (as $n \to \infty$) find a solution if one exists

**LaValle, '98; Kleinbort et al., '18**: RRT will eventually (as $n \to \infty$) find a solution if one exists

Unless stated otherwise, the configuration space is assumed to be the $d$-dimensional Euclidean unit hypercube $[0, 1]^d$, with $2 \leq d \leq \infty$

# Theoretical guarantees: quality

Question: what can be said about the quality of the returned solution for PRM and RRT, in terms of length, energy, etc.?




Nechushtan et al. (2010) and Karaman and Frazzoli (2011) proved that RRT can produce arbitrarily-bad paths with non-negligible probability: for example, RRT would prefer to take the long (red) way

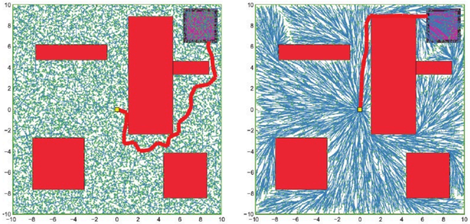
# Theoretical guarantees: quality

Karaman and Frazzoli in 2011 provided the first rigorous study of optimality in sampling-based planners:

**Theorem**

The cost of the solution returned by PRM converges, as $n \to \infty$, to the optimum, when $r_n = \gamma(\frac{\log n}{n})^{\frac{1}{d}}$, where $\gamma$ only depends on $d$

- KF11 also introduced an asymptotically optimal variant of RRT called RRT* (right)
- Result was later updated to [Solovey et al. '19]: $r_n = \gamma(\frac{\log n}{n})^{\frac{1}{d+1}}$
- Now back to 1/d [Lukyanenko & Soudbakhsh '23]



# Observations

PRM-like motion planning algorithms

- For a give number of nodes $n$, they find “good” paths
- ...however, require many costly collision checksRRT-like motion planning algorithms

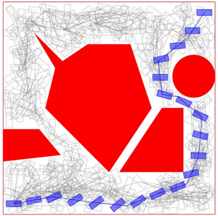
RRT-like motion planning algorithms

- Finds a feasible path quickly
- ...however the quality of that path is, in general, poor
- “traps” itself by disallowing new better paths to emerge - RRT* performs local label correction as samples are added to help remedy this

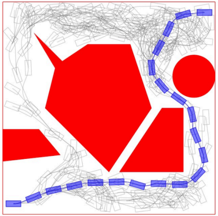
# Kinodynamic planning

Kinodynamic motion planning problem: in addition to obstacle avoidance, paths are subject to differential constraints

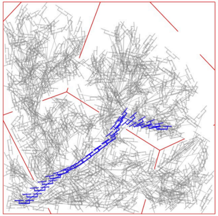
- The robot operates in the state space $X$
- To move the robot applies control $u \in U$
- Motion needs to satisfy the system's constraints: $\dot{x} = f(x, u)$ for $x \in X, u \in U$ [Schmerling & Pavone, '19]



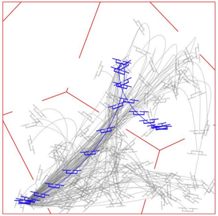
(a) Geometric Planning



(b) Planning with Dubins Car Dynamics



(c) Geometric Planning



(d) Planning with Simplified Quadrotor Dynamics

# Forward-propagation-based algorithms

RRT can be extended to kinodynamic case in a relatively easy way:

1. Draw a random state and find its nearest neighbor $x_{near}$
2. Sample a random control $u \in U$ and random duration $t$
3. Forward propagate the control $u$ for $t$ time from $x_{near}$

**Algorithm:** RRT with Control Input $(x_{\text{init}}, x_{\text{goal}}, k, T_{\text{prop}}, U)$

---

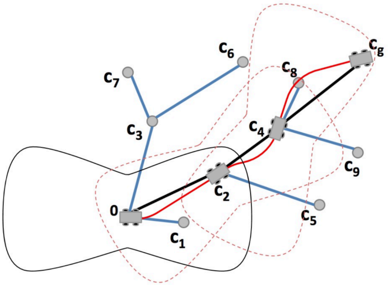
$T$.init($x_{\text{init}}$)
**for** $i = 1$ to $k$ **do**
    $x_{\text{rand}} \leftarrow$ RANDOM_STATE()
    $x_{\text{near}} \leftarrow$ NEAREST_NEIGHBOR($x_{\text{rand}}, T$)
    $t \leftarrow$ SAMPLE_DURATION($0, T_{\text{prop}}$)
    $u \leftarrow$ SAMPLE_CONTROL_INPUT($U$)
    $x_{\text{new}} \leftarrow$ PROPAGATE($x_{\text{near}}, u, t$)
    **if** COLLISION_FREE($x_{\text{near}}, x_{\text{new}}$) **then**
        $T$.add_vertex($x_{\text{new}}$)
        $T$.add_edge($x_{\text{near}}, x_{\text{new}}$)
**return** $T$

Source: Kleinbort et al., 2019

# Steering-based algorithms

When efficient online steering subroutines exist, kinodynamic planning algorithms may take advantage of this domain knowledge

- Connect samples by using an optimal trajectory (steering problem)
- Use reachable sets to find nearest neighbors




Source: Schmerling et al., 2015

# Should probabilistic planners be probabilistic?

Key question: would theoretical guarantees and practical performance still hold if these algorithms were to be derandomized, i.e., run on deterministic samples?

Important question as derandomization would:

- Ease certification process
- Ease use of offline computation
- Potentially simplify a number of operations (e.g., NN search)

# Designing "good" sequences

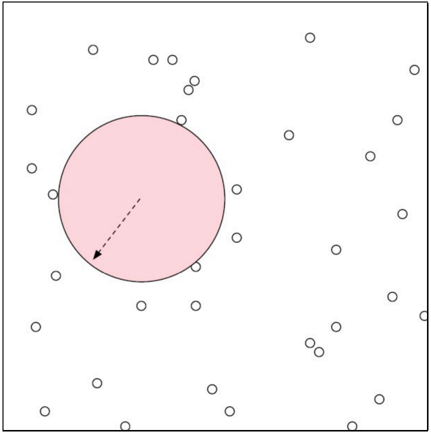
## $l_2$-dispersion

For a finite set $S$ of points contained in $x \subset \mathbb{R}^d$, its $l_2$-dispersion $D(S)$ is defined as

$$D(S) = \sup_{x \in X} \min_{s \in S} ||s - x||_2$$

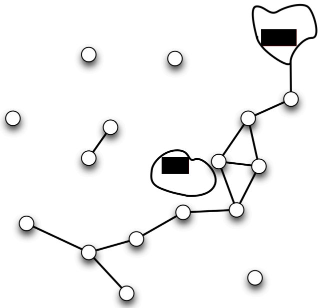
Key facts:

- There exist deterministic sequences with $D(S)$ of order $O(n^{-1/d})$, referred to as **low-dispersion** sequences
- Sequences minimizing $l_2$-dispersion only known for $d = 2$



# Optimality of deterministic planning

```
V ← {x_init} ∪ SampleFree(n);    E ← ∅
for all v ∈ V do
    X_near ← Near(V \ {v}, v, r_n)
    for all x ∈ X_near do
        if CollisionFree(v, x) then
            E ← E ∪ {(v, x)} ∪ {(x, v)}
return ShortestPath(x_init, V, E)
```



**Optimality:** Let $c'$ denote the arc length of the path returned with $n$ samples. Then if

1. Samples set $S$ has dispersion $D(S) \leq \gamma n^{-1/d}$ for some $\gamma > 0$,
2. $n^{1/d} r_n \to \infty$,

then $\lim_{n\to\infty} c_n = c*$, where $c*$ is the cost of the optimal path

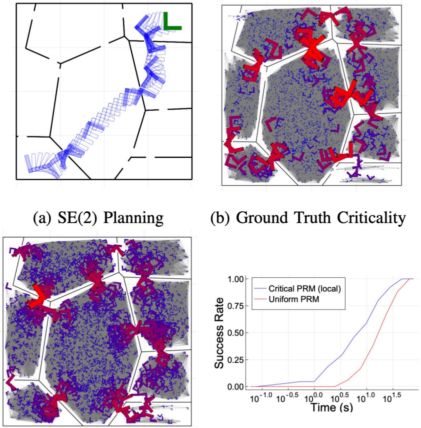
# Deterministic sampling-based motion planning

- Asymptotic optimality can be achieved with deterministic sequences and with a smaller connection radius
- Deterministic convergence rates: instrumental to the certification of sampling-based planners
- Computational and space complexity: under some assumptions, arbitrarily close to theoretical lower bound
- Deterministic sequences appear to provide superior performance

Source: Janson et al. Deterministic Sampling-Based Motion Planning: Optimality, Complexity, and Performance. 2018

# Biased sampling for SBMP

- Potential issue with uniform sampling: narrow corridors in C-space require many samples to identify/traverse
- Key idea: bias sampling towards suspected such challenging regions of C-space
- Biased sampling distributions can be hand- constructed and/or adapt online (e.g., Hybrid Sampling PRM), or learned from prior experience solving similar planning problems




(a) SE(2) Planning (b) Ground Truth Criticality

(c) Learned Criticality (d) Time (s) vs. Success

Sources: Hsu et al. Hybrid PRM sampling with a cost-sensitive adaptive strategy. 2005.
Ichter et al. Learned Critical Probabilistic Roadmaps for Robotic Motion Planning. 2020

# Acknowledgements

**Acknowledgements**

This slide deck is based on material from the Stanford ASL and CMU's Robotics Institute

# References


S. M. LaValle, *Planning Algorithms*. Cambridge University Press, 2006. [Online]. Available: http://planning.cs.uiuc.edu/

——, "Motion planning," *IEEE Robotics Autom. Mag.*, vol. 18, no. 2, pp. 108–118, 2011. [Online]. Available: https://doi.org/10.1109/MRA.2011.941635

L. E. Kavraki, P. Svestka, J. Latombe, and M. H. Overmars, "Probabilistic roadmaps for path planning in high-dimensional configuration spaces," *IEEE Trans. Robotics Autom.*, vol. 12, no. 4, pp. 566–580, 1996. [Online]. Available: https://doi.org/10.1109/70.508439

R. Bohlin and L. E. Kavraki, "Path planning using lazy PRM," in *Proceedings of the 2000 IEEE International Conference on Robotics and Automation, ICRA 2000, April 24-28, 2000, San Francisco, CA, USA*. IEEE, 2000, pp. 521–528. [Online]. Available: https://doi.org/10.1109/ROBOT.2000.844107

L. Jaillet and T. Siméon, "A prm-based motion planner for dynamically changing environments," in *2004 IEEE/RSJ International Conference on Intelligent Robots and Systems, Sendai, Japan, September 28 - October 2, 2004*. IEEE, 2004, pp. 1606–1611. [Online]. Available: https://doi.org/10.1109/IROS.2004.1389625

S. Karaman and E. Frazzoli, "Sampling-based algorithms for optimal motion planning," *CoRR*, vol. abs/1105.1186, 2011. [Online]. Available: http://arxiv.org/abs/1105.1186

S. M. LaValle and J. J. K. Jr., "Randomized kinodynamic planning," *Int. J. Robotics Res.*, vol. 20, no. 5, pp. 378–400, 2001. [Online]. Available: https://doi.org/10.1177/02783640122067453

E. Plaku, K. E. Bekris, B. Y. Chen, A. M. Ladd, and L. E. Kavraki, "Sampling-based roadmap of trees for parallel motion planning," *IEEE Trans. Robotics*, vol. 21, no. 4, pp. 597–608, 2005. [Online]. Available: https://doi.org/10.1109/TRO.2005.847599

M. Arnold, Y. Baryshnikov, and S. Lavalle, "Convex hull asymptotic shape evolution," 01 2013.

M. Kleinbort, K. Solovey, Z. Littlefield, K. E. Bekris, and D. Halperin, "Probabilistic completeness of RRT for geometric and kinodynamic planning with forward propagation," *CoRR*, vol. abs/1809.07051, 2018. [Online]. Available: http://arxiv.org/abs/1809.07051

K. Solovey, L. Janson, E. Schmerling, E. Frazzoli, and M. Pavone, "Revisiting the asymptotic optimality of RRT," *CoRR*, vol. abs/1909.00688, 2019. [Online]. Available: http://arxiv.org/abs/1909.00688