Randomized Sampling-based Motion Planning Methods

Jan Faigl

Department of Computer Science
Faculty of Electrical Engineering
Czech Technical University in Prague

Lecture 09

B4M36UIR – Artificial Intelligence in Robotics

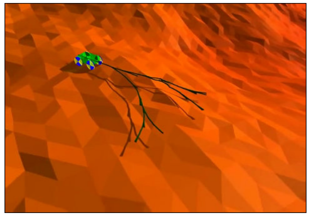
Overview of the Lecture

Part I

Part 1 – Sampling-based Motion Planning

(Randomized) Sampling-based Motion Planning

- It uses an explicit representation of the obstacles in $\mathcal{C}$*-space*.
- A "black-box" function is used to evaluate if a configuration q is a collision-free using geometrical models of the objects (robot and environment).
- 2D or 3D shapes of the robot and environment can be represented as sets of triangles – tesselated models.
- Collision test is then a test of for the intersection of the triangles.
- Collision free configurations **form a discrete representation of $\mathcal{C}_{free}$**.



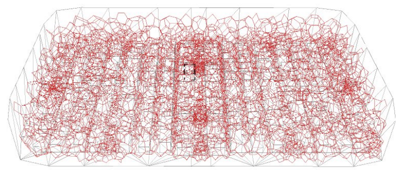
A collision test library RAPID http://gamma.cs.unc.edu/OBB/.

- Configurations in $\mathcal{C}_{free}$ can be sampled randomly and connected to a (**probabilistic**) **roadmap**.
- Rather than the full completeness they provide **probabilistic completeness** or resolution completeness. *It is probabilisticaly complete if for increasing number of samples, an admissible solution would be found (if exists).*

Probabilistic Roadmaps

A discrete representation of the continuous $\mathcal{C}$-space generated by randomly sampled configurations in $\mathcal{C}_{free}$ that are connected into a graph.

- **Nodes** of the graph represent admissible configurations of the robot.
- **Edges** represent a feasible path (trajectory) between the particular configurations.



Having the graph, the final path (trajectory) can be found by a graph search technique.

Incremental Sampling and Searching

- Single query sampling-based algorithms incrementally create a search graph (roadmap).
  1. **Initialization** – $G(V, E)$ an undirected search graph, V may contain q_{start}, q_{goal} and/or other points in $\mathcal{C}_{free}$.
  2. **Vertex selection method** – choose a vertex $q_{cur} \in V$ for the expansion.
  3. **Local planning method** – for some $q_{new} \in \mathcal{C}_{free}$, attempt to construct a path $\tau : [0, 1] \to \mathcal{C}_{free}$ such that $\tau(0) = q_{cur}$ and $\tau(1) = q_{new}$, τ must be checked to ensure it is collision free.
     - If τ is not a collision-free, go to Step 2.
  4. **Insert an edge in the graph** – Insert τ into E as an edge from q_{cur} to q_{new} and insert q_{new} to V if $q_{new} \notin V$. *How to test q_{new} is in V?*
  5. **Check for a solution** – Determine if G encodes a solution by using a single search tree or graph search technique.
  6. **Repeat Step 2** – iterate unless a solution has been found or a termination condition is satisfied.

LaValle, S. M.: Planning Algorithms (2006), Chapter 5.4.

Probabilistic Roadmap Strategies

Multi-Query strategy is to create a roadmap that can be used for several queries.

- Generate a single roadmap that is then used for repeated planning queries.
- An representative technique is **Probabilistic RoadMap (PRM)**.

Kavraki, L., Svestka, P., Latombe, J.-C., Overmars, M. H.B: *Probabilistic Roadmaps for Path Planning in High Dimensional Configuration Spaces*, **IEEE Transactions on Robotics**, 12(4):566–580, 1996.

Hsu, D., Latombe, J.-C., Kurniawati, H.: *On the Probabilistic Foundations of Probabilistic Roadmap Planning*. **The International Journal of Robotics Research**, 25(7):627–643, 2006.

Single-Query strategy is an incremental approach.

- For each planning problem, it constructs a new roadmap to characterize the subspace of $\mathcal{C}$-space that is relevant to the problem.
  - Rapidly-exploring Random Tree – RRT; *LaValle, 1998*
  - Expansive-Space Tree – EST; *Hsu et al., 1997*
  - Sampling-based Roadmap of Trees – SRT.

A combination of multiple–query and single–query approaches.
Plaku et al., 2005

Multi-Query Strategy

Build a roadmap (graph) representing the environment.

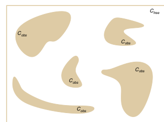
1. Learning phase
   1.1 Sample n points in $\mathcal{C}_{free}$.
   1.2 Connect the random configurations using a local planner.
2. Query phase
   2.1 Connect start and goal configurations with the PRM.
   t Using a local planner.
   2.2 Use the graph search to find the path.

Probabilistic Roadmaps for Path Planning in High Dimensional Configuration Spaces
Lydia E. Kavraki and Petr Svestka and Jean-Claude Latombe and Mark H. Overmars,
IEEE Transactions on Robotics and Automation, 12(4):566–580, 1996.

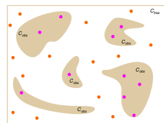
First planner that demonstrates ability to solve general planning problems in more than 4-5 dimensions.

PRM Construction

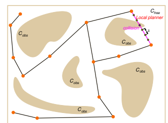
#1 Given problem domain



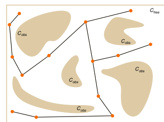
#2 Random configuration



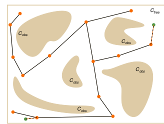
#3 Connecting samples



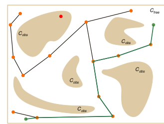
#4 Connected roadmap



#5 Query configurations

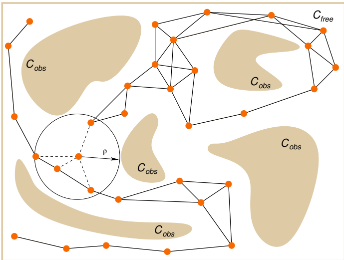


#6 Final found path



Practical PRM

- Incremental construction.
- Connect nodes in a radius r.
- Local planner tests collisions up to selected resolution δ.
- Path can be found by Dijkstra's algorithm.



What are the properties of the PRM algorithm?

We need a couple of more formalisms.

Path Planning Problem Formulation

- Path planning problem can be defined by a triplet
  $\mathcal{P} = (\mathcal{C}_{free}, q_{init}, \mathcal{Q}_{goal})$, where
  - $\mathcal{C}_{free} = \mathrm{cl}(\mathcal{C} \setminus \mathcal{C}_{obs})$, $\mathcal{C} = (0,1)^d$, for $d \in \mathbb{N}$, $d \geq 2$; (scaling)
  - $q_{init} \in \mathcal{C}_{free}$ is the initial configuration (condition);
  - $\mathcal{Q}_{goal}$ is the goal region defined as an open subspace of $\mathcal{C}_{free}$.
- Function $\pi : [0,1] \to \mathbb{R}^d$ of *bounded variation* is called:
  - **path** – if it is continuous;
  - **collision-free path** – if it is a path and $\pi(\tau) \in \mathcal{C}_{free}$ for $\tau \in [0,1]$;
  - **feasible** – if it is a collision-free path, and $\pi(0) = q_{init}$ and $\pi(1) \in \mathrm{cl}(\mathcal{Q}_{goal})$.
- A function π with total variation $\mathrm{TV}(\pi) < \infty$ is said to have bounded variation, where $\mathrm{TV}(\pi)$ is the total variation
  $$\mathrm{TV}(\pi) = \sup_{\{n \in \mathbb{N}, 0=\tau_0<\tau_1<\ldots<\tau_n=s\}} \sum_{i=1}^{n} |\pi(\tau_i) - \pi(\tau_{i-1})|.$$
- Total variation $\mathrm{TV}(\pi)$ is de facto a path length.

Path Planning Problem

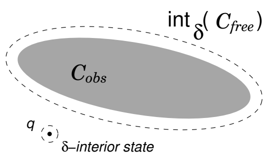
- **Feasible path planning**
  For a path planning problem $(\mathcal{C}_{free}, q_{init}, \mathcal{Q}_{goal})$:
  - Find a feasible path $\pi : [0,1] \to \mathcal{C}_{free}$ such that $\pi(0) = q_{init}$ and $\pi(1) \in \mathrm{cl}(\mathcal{Q}_{goal})$, if such path exists;
  - Report failure if no such path exists.
- **Optimal path planning**
  The optimality problem asks for a feasible path with the minimum cost.
  For $(\mathcal{C}_{free}, q_{init}, \mathcal{Q}_{goal})$ and a cost function $c : \Sigma \to \mathbb{R}_{\geq 0}$:
  - Find a feasible path π^* such that $c(\pi^*) = \min\{c(\pi) : \pi \text{ is feasible}\}$;
  - Report failure if no such path exists.

The cost function is assumed to be monotonic and bounded. There exists k_c such that $c(\pi) \leq k_c\,\mathrm{TV}(\pi)$.

Probabilistic Completeness 1/2

First, we need **robustly feasible path planning problem** $(\mathcal{C}_{free}, q_{init}, \mathcal{Q}_{goal})$.

- $q \in \mathcal{C}_{free}$ is *δ-interior state* of $\mathcal{C}_{free}$ if the closed ball of radius δ centered at q lies entirely inside $\mathcal{C}_{free}$.
- *δ-interior* of $\mathcal{C}_{free}$ is $\mathrm{int}_\delta(\mathcal{C}_{free}) = \{q \in \mathcal{C}_{free} | \mathcal{B}_{/,\delta} \subseteq \mathcal{C}_{free}\}$. *A collection of all δ-interior states.*
- A collision free path π has **strong δ-clearance**, if π lies entirely inside $\mathrm{int}_\delta(\mathcal{C}_{free})$.
- $(\mathcal{C}_{free}, q_{init}, \mathcal{Q}_{goal})$ is **robustly feasible** if a solution exists and it is a feasible path with **strong δ-clearance**, for $\delta > 0$.

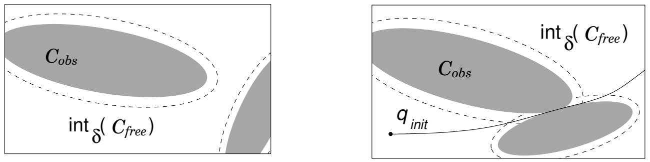


Probabilistic Completeness 2/2

An algorithm $\mathcal{ALG}$ is **probabilistically complete** if, for any *robustly feasible path planning problem* $\mathcal{P} = (\mathcal{C}_{free}, q_{init}, \mathcal{Q}_{goal})$,

$$\lim_{n\to\infty} Pr(\mathcal{ALG} \text{ returns a solution to } \mathcal{P}) = 1.$$

- It is a *"relaxed"* notion of the completeness.
- Applicable only to problems with a **robust solution**.



We need some space where random configurations can be sampled.

Asymptotic Optimality 1/4 – Homotopy

Asymptotic optimality relies on a notion of **weak δ-clearance**.

Notice, we use strong δ-clearance for probabilistic completeness.

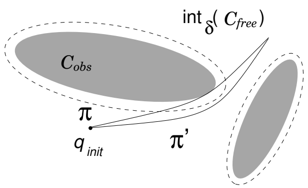
- We need to describe possibly improving paths (during the planning).
- Function $\psi : [0,1] \to \mathcal{C}_{free}$ is called **homotopy**, if $\psi(0) = \pi_1$ and $\psi(1) = \pi_2$ and $\psi(\tau)$ is collision-free path for all $\tau \in [0,1]$.
- A collision-free path π_1 is **homotopic** to π_2 if there exists homotopy function ψ.

A path homotopic to π can be continuously transformed to π through $\mathcal{C}_{free}$.

Asymptotic Optimality 2/4 – Weak δ-clearance

- A collision-free path $\pi : [0,s] \to \mathcal{C}_{free}$ has **weak δ-clearance** if there exists a path π' that has **strong δ-clearance** and homotopy ψ with $\psi(0) = \pi$, $\psi(1) = \pi'$, and for all $\alpha \in (0,1]$ there exists $\delta_\alpha > 0$ such that $\psi(\alpha)$ has strong δ-clearance.

Weak δ-clearance does not require points along a path to be at least a distance δ away from obstacles.



- A path π with a weak δ-clearance.
- π' lies in $\mathrm{int}_\delta(\mathcal{C}_{free})$ and it is the same homotopy class as π.

We need the strong δ-clearance to find π' (by randomized sampling). Then, such a path can be (localy) improved (shorten) towards the shortest π. π' must be within the same homotopy class (passing obstacles at the same way as the optimal path π) to guarantee such a path π can be the optimal path.

Asymptotic Optimality 3/4 – Robust Optimal Solution

- *Asymptotic optimality* is applicable with a **robust optimal solution** that can be obtained as a limit of robust (non-optimal) solutions.
- A collision-free path π^* is **robust optimal solution** if it has *weak δ-clearance* and for any sequence of collision free paths $\{\pi_n\}_{n\in\mathbb{N}}$, $\pi_n \in \mathcal{C}_{free}$ such that $\lim_{n\to\infty} \pi_n = \pi^*$,
  $$\lim_{n\to\infty} c(\pi_n) = c(\pi^*).$$
  *There exists a path with strong δ-clearance, and π^* is homotopic to such path and π^* is of **the lowest cost**.*
- Weak δ-clearance implies an existence of the strong δ-clearance path within the some homotopy, and thus robustly feasible solution problem.

Thus, it implies the probabilistic completeness.

Asymptotic Optimality 4/4 – Asymptotically Optimal Algorithm

An algorithm $\mathcal{ALG}$ is **asymptotically optimal** if, for any path planning problem $\mathcal{P} = (\mathcal{C}_{free}, q_{init}, \mathcal{Q}_{goal})$ and cost function c that admits **a robust optimal solution** with the finite cost c^* such that

$$Pr\left(\left\{\lim_{i\to\infty} Y_i^{\mathcal{ALG}} = c^*\right\}\right) = 1.$$

- $Y_i^{\mathcal{ALG}}$ is the extended random variable corresponding to the minimum-cost solution included in the graph returned by $\mathcal{ALG}$ at the end of the iteration i.

Properties of the PRM Algorithm

- Completeness for the standard PRM has not been provided when it was introduced.
- A simplified version of the PRM (called sPRM) has been most studied.
- sPRM is probabilistically complete.

What are the differences between PRM and sPRM?

PRM vs. simplified PRM (sPRM)

Algorithm 1: PRM

Input: q_{init}, the number of samples n, and radius r
Output: PRM – $G = (V, E)$

$V \leftarrow \emptyset; E \leftarrow \emptyset$;
for $i = 0, \dots, n$ **do**
  $q_{rand} \leftarrow$ SampleFree;
  $U \leftarrow$ Near$(G = (V, E), q_{rand}, r)$;
  $V \leftarrow V \cup \{q_{rand}\}$;
  foreach $u \in U$ *with increasing* $||u - q_r||$ **do**
    if q_{rand} *and* u *are not in the same connected component of* $G = (V, E)$ **then**
      if CollisionFree(q_{rand}, u) **then**
        $E \leftarrow E \cup \{(q_{rand}, u), (u, q_{rand})\}$;
return $G = (V, E)$;

Algorithm 2: sPRM

Input: q_{init}, the number of samples n, and radius r
Output: PRM – $G = (V, E)$

$V \leftarrow \{q_{init}\} \cup \{\text{SampleFree}_i\}_{i=1,\dots,n-1}; E \leftarrow \emptyset$;
foreach $v \in V$ **do**
  $U \leftarrow$ Near$(G = (V, E), v, r) \setminus \{v\}$;
  foreach $u \in U$ **do**
    if CollisionFree(v, u) **then**
      $E \leftarrow E \cup \{(v, u), (u, v)\}$;
return $G = (V, E)$;

- Connections between vertices in the same connected component are allowed.
- The radius r is fixed and can be relatively long; thus sPRM can be very demanding.

Several ways for the set U of vertices to connect them can improved the performance, such as k-nearest neighbors to v; or variable connection radius r as a function of n at the cost of lost of asymptotical optimality or even probabilistic completeness.

PRM – Properties

- sPRM (simplified PRM):
  - **Probabilistically complete and asymptotically optimal.**
  - Processing complexity can be bounded by $O(n^2)$.
  - Query complexity can be bounded by $O(n^2)$.
  - Space complexity can be bounded by $O(n^2)$.
- *Heuristics practically used* are not necessarily probabilistic complete and asymptotically optimal.
  - k-nearest sPRM is not probabilistically complete for $k = 1$.
  - Variable radius sPRM is not probabilistically complete; with the radius $r(n) = \gamma n^{-\frac{1}{d}}$.

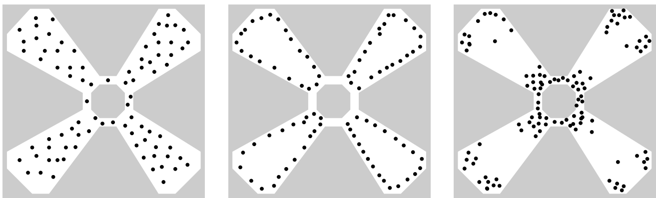
See Karaman and Frazzoli: Sampling-based Algorithms for Optimal Motion Planning, IJRR 2011.

PRM algorithm

\+ It has very simple implementation.
\+ It provides completeness (for sPRM).
– Differential constraints (car-like vehicles) are not straightforward (but possible).

Comments about Random Sampling 1/2

- Different sampling strategies (distributions) may be applied.



- Notice, one of the main issues of the randomized sampling-based approaches is the narrow passage.
- Several modifications of sampling-based strategies have been proposed in the last decades.

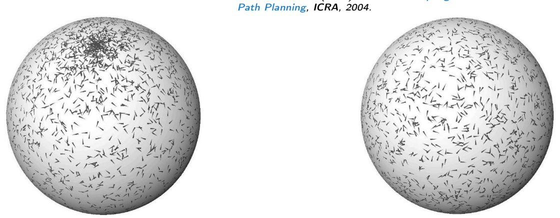
Comments about Random Sampling 2/2

- A solution can be found using only a few samples.

Using the Oraculum.

- Sampling strategies are important: Near obstacles; Narrow passages; Grid-based; Uniform sampling must be carefully considered.

James J. Kuffner (2004): Effective Sampling and Distance Metrics for 3D Rigid Body Path Planning, ICRA, 2004.



Naïve sampling | Uniform sampling of SO(3) using Euler angles

Rapidly Exploring Random Tree (RRT)

Single–Query algorithm.

- It incrementally builds a graph (tree) towards the goal area.

It does not guarantee precise path to the goal configuration.

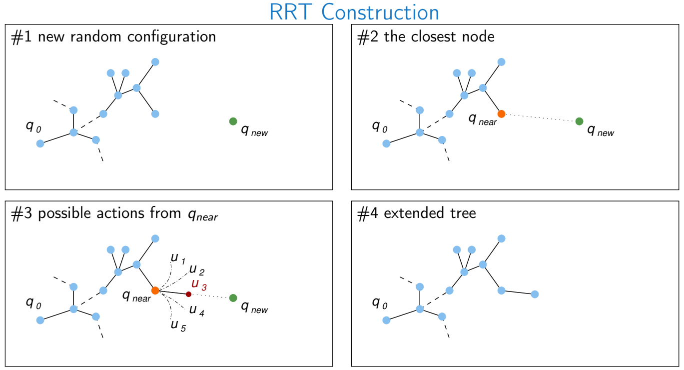
1. Start with the initial configuration q_0, which is a root of the constructed graph (tree).
2. Generate a new random configuration q_{new} in $\mathcal{C}_{free}$.
3. Find the closest node q_{near} to q_{new} in the tree.

   KD-tree implementation like ANN or FLANN libraries can be utilized.
4. Extend q_{near} towards q_{new}.

   Extend the tree by a small step or using a direct control $u \in \mathcal{U}$ that will move the robot to the position closest to q_{new} applied for δt.
5. Go to Step 2 until the tree is within a sufficient distance from the goal configuration.

   Or terminates after dedicated running time.

RRT Construction



RRT Algorithm

- Motivation is a single query and *control-based* path finding.
- It incrementally builds a graph (tree) towards the goal area.

Algorithm 3: Rapidly Exploring Random Tree (RRT)

Input: q_{init}, number of samples n
Output: Roadmap $G = (V, E)$

$V \leftarrow \{q_{init}\}; E \leftarrow \emptyset$;
for $i = 1, \dots, n$ **do**
  $q_{rand} \leftarrow$ SampleFree;
  $q_{nearest} \leftarrow$ Nearest$(G = (V, E), q_{rand})$;
  $q_{new} \leftarrow$ Steer$(q_{nearest}, q_{rand})$;
  if CollisionFree$(q_{nearest}, q_{new})$ **then**
    $V \leftarrow V \cup \{x_{new}\}$; $E \leftarrow E \cup \{(x_{nearest}, x_{new})\}$;
return $G = (V, E)$;

Rapidly-exploring random trees: A new tool for path planning
S. M. LaValle,
Technical Report 98-11, Computer Science Dept., Iowa State University, 1998.

Properties of RRT Algorithms

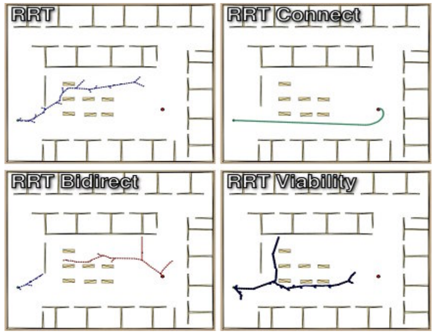
- The RRT algorithm rapidly explores the space.

  q_{new} will more likely be generated in large, not yet covered parts (voroni bias).
- Allows considering kinodynamic/dynamic constraints (during the expansion).
- Can provide trajectory or a sequence of direct control commands for robot controllers.
- A collision detection test is usually used as a "black-box."

  RAPID, Bullet libraries.
- Similarly to PRM, RRT algorithms have poor performance in narrow passage problems.
- RRT algorithms provide feasible paths.

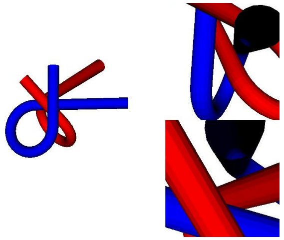
  It can be relatively far from an optimal solution; according to the length of the path.
- Many variants of the RRT have been proposed in the literature.

Examples 1/4 – Variants of RRT algorithms

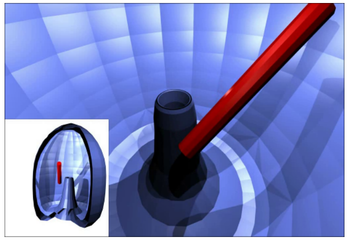


Courtesy of P. Vaněk.

Examples 2/4 – Motion Planning Benchmarks



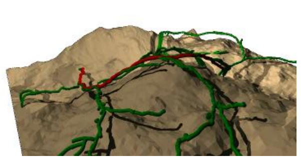
Alpha puzzle benchmark



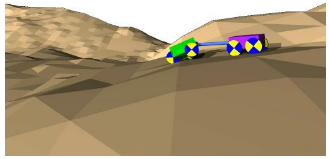
Bugtrap benchmark

Courtesy of V. Vonásek.

Examples 3/4 – Planning on Terrain Considering Frictions



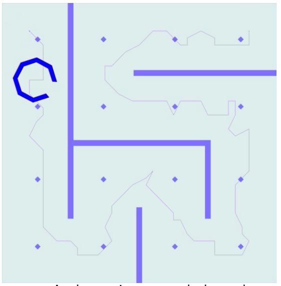
Planning on a 3D surface



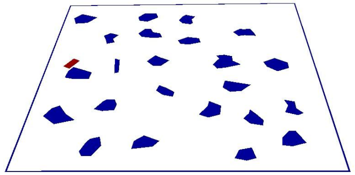
Planning with dynamics (friction forces)

Courtesy of V. Vonásek.

Examples 4/4 – Motion Planning for Complex Shape and Car-like Robot



Apply rotations to reach the goal



Planning for a car-like robot

Courtesy of V. Vonásek and P. Vaněk.

Car-like Robot

- Configuration

$$\vec{x} = \begin{pmatrix} x \\ y \\ \phi \end{pmatrix}$$

position and orientation.

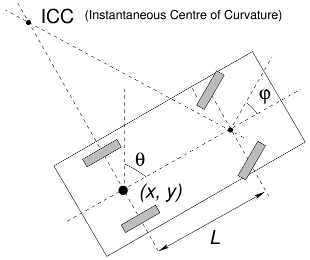
- Controls

$$\vec{u} = \begin{pmatrix} v \\ \varphi \end{pmatrix}$$

forward velocity, steering angle.

- System equation

$$\begin{aligned} \dot{x} &= v\cos\phi \\ \dot{y} &= v\sin\phi \\ \dot{\phi} &= \frac{v}{L}\tan\varphi \end{aligned} .$$



Kinematic constraints $\dim(\vec{u}) < \dim(\vec{x})$.

Differential constraints on possible $\dot{q}$:

$\dot{x}\sin(\phi) - \dot{y}\cos(\phi) = 0$.

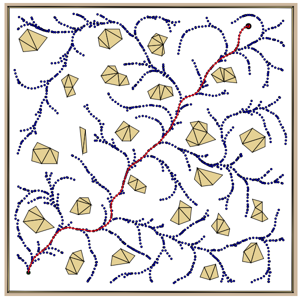
Control-Based Sampling

- Select a configuration q from the tree T of the current configurations.
- Pick a control input $\vec{u} = (v, \varphi)$ and the integrate system (motion) equation over a short period Δt:

$$\begin{pmatrix} \Delta x \\ \Delta y \\ \Delta\phi \end{pmatrix} = \int_t^{t+\Delta t} \begin{pmatrix} v\cos\phi \\ v\sin\phi \\ \frac{v}{L}\tan\varphi \end{pmatrix} dt.$$

- If the motion is collision-free, add the endpoint to the tree.

Considering k configurations for $k\delta t = dt$.



Part II

Part 2 – Optimal Sampling-based Motion Planning Methods

Sampling-Based Motion Planning

- PRM and RRT are theoretically probabilistic complete.
- They provide a feasible solution without quality guarantee.

However, they are successfully used in many practical applications.

- In 2011, a systematical study of the asymptotic behavior of randomized sampling-based planners has been published. *It shows, that in some cases, they converge to a non-optimal value with a probability 1.*

It builds on properties of Random Geometric Graphs (RGG) introduced by Gilbert (1961) and further studied by Penrose (1999).

- Based on the study, new algorithms have been proposed: RRG and optimal RRT (RRT*).

Karaman, S., Frazzoli, E.: *Sampling-based algorithms for optimal motion planning*, IJRR, 30(7):846–894, 2011.

http://sertac.scripts.mit.edu/rrtstar

RRT and Quality of Solution 1/2

- Let Y_i^{RRT} be the cost of the best path in the RRT at the end of the iteration i.
- Y_i^{RRT} converges to a random variable

$$\lim_{i\to\infty} Y_i^{RRT} = Y_\infty^{RRT}.$$

- The random variable Y_∞^{RRT} is sampled from a distribution with zero mass at the optimum, and

$$Pr[Y_\infty^{RRT} > c^*] = 1.$$

Karaman and Frazzoli, 2011

- The best path in the RRT converges to a sub-optimal solution almost surely.

RRT and Quality of Solution 2/2

- RRT does not satisfy a necessary condition for the asymptotic optimality.
  - For $0 < R < \inf_{q \in \mathcal{Q}_{goal}} ||q - q_{init}||$, the event $\{\lim_{n\to\infty} Y_n^{RTT} = c^*\}$ occurs only if the k-th branch of the RRT contains vertices outside the R-ball centered at q_{init} for infinitely many k.

See Appendix B in Karaman and Frazzoli, 2011.

- It is required the root node will have infinitely many subtrees that extend at least a distance ϵ away from q_{init}.

The sub-optimality is caused by disallowing new better paths to be discovered.

Rapidly-exploring Random Graph (RRG)

Algorithm 4: Rapidly-exploring Random Graph (RRG)

Input: q_{init}, the number of samples n
Output: $G = (V, E)$

```
V ← ∅; E ← ∅
for i = 0, ..., n do
    q_rand ← SampleFree
    q_nearest ← Nearest(G = (V, E), q_rand)
    q_new ← Steer(q_nearest, q_rand)
    if CollisionFree(q_nearest, q_new) then
        Q_near ← Near(G = (V, E), q_new, min{γ_RRG(log(card(V))/ card(V))^{1/d}, η})
        V ← V ∪ {q_new}; E ← E ∪ {(q_nearest, q_new), (q_new, q_nearest)}
        foreach q_near ∈ Q_near do
            if CollisionFree(q_near, q_new) then
                E ← E ∪ {(q_near, q_new), (q_new, q_near)}     // Connect Q_near with q_new.
return G = (V, E)
```

Proposed by Karaman and Frazzoli (2011). Theoretical results are related to properties of Random Geometric Graphs (RGG) introduced by Gilbert (1961) and further studied by Penrose (1999).

RRG Expansions

- At each iteration, RRG tries to connect new sample to all vertices in the r_n ball centered at it.
- The ball of radius

$$r(\mathrm{card}(V)) = \min\left\{\gamma_{RRG}\left(\frac{\log(\mathrm{card}(V))}{\mathrm{card}(V)}\right)^{1/d}, \eta\right\},$$

where
  - η is the constant of the local steering function;
  - $\gamma_{RRG} > \gamma^*_{RRG} = 2(1+1/d)^{1/d}(\mu(\mathcal{C}_{free})/\zeta_d)^{1/d}$;
    - d – dimension of the space;
    - $\mu(\mathcal{C}_{free})$ – Lebesgue measure of the obstacle–free space;
    - ζ_d – volume of the unit ball in d-dimensional Euclidean space.
- The connection radius decreases with n.
- The rate of decay $\approx$ the average number of connections attempted is proportional to $\log(n)$.

RRG Properties

- Probabilistically complete;
- Asymptotically optimal;
- Complexity is $O(\log n)$.

(per one sample)

- Computational efficiency and optimality:
  - It attempts a connection to $\Theta(\log n)$ nodes at each iteration;

In average

    - Reduce volume of the "connection" ball as $\log(n)/n$;
    - Increase the number of connections as $\log(n)$.

Other Variants of the Optimal Motion Planning

- **PRM*** follows the standard PRM algorithm where connections are attempted between roadmap vertices that are the within connection radius r as the function of n:

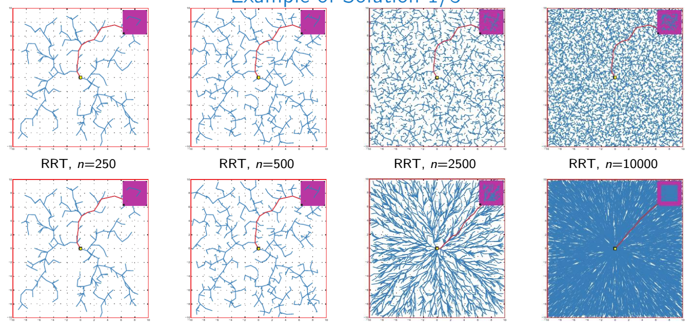
$$r(n) = \gamma_{PRM}(\log(n)/n)^{1/d}.$$

- **RRT*** is a modification of the RRG, where cycles are avoided.

It is a tree version of the RRG.

  - A tree roadmap allows considering non-holonomic dynamics and kinodynamic constraints.
  - It is basically the RRG with "rerouting" the tree when a better path is discovered.

Example of Solution 1/3

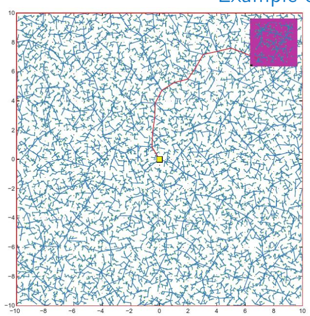


RRT, n=250 | RRT, n=500 | RRT, n=2500 | RRT, n=10000

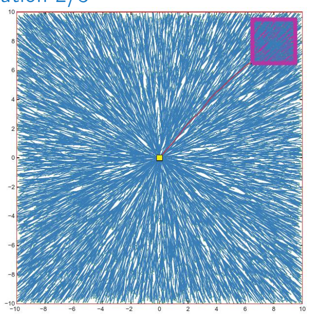
RRT*, n=250 | RRT*, n=500 | RRT*, n=2500 | RRT*, n=10000

Karaman & Frazzoli, 2011

Example of Solution 2/3

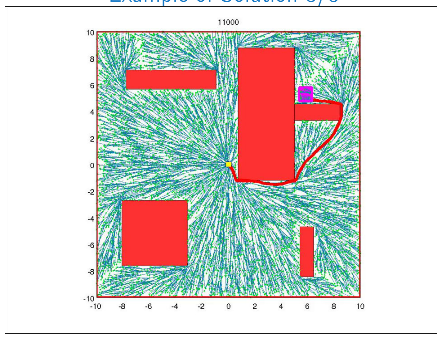


RRT, n=20000



RRT*, n=20000

Example of Solution 3/3



https://www.youtube.com/watch?v=YKiQTJpPFkA

Overview of Randomized Sampling-based Algorithms

Algorithm	Probabilistic Completeness	Asymptotic Optimality
PRM	✔	✘
sPRM	✔	✔
k-nearest sPRM	✘	✘
RRT	✔	✘
RRG	✔	✔
PRM*	✔	✔
RRT*	✔	✔

sPRM with connection radius r as a function of n; $r(n) = \gamma_{PRM}(\log(n)/n)^{1/d}$ with $\gamma_{PRM} > \gamma^*_{PRM} = 2(1+1/d)^{1/d}(\mu(\mathcal{C}_{free})/\zeta_d)^{1/d}$.

Improved Sampling-based Motion Planners

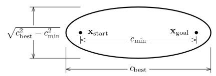
- Although asymptotically optimal sampling-based motion planners such as RRT* or RRG may provide high-quality or even optimal solutions to the complex problem, their performance in simple scenarios (such as 2D) is relatively poor.
  In a comparison to the ordinary approaches such as visibility graph.
- The computational performance can be improved similarly as for the RRT.
  - Using goal biasing, supporting sampling in narrow passages, multi-tree growing (Bidirectional RRT).
- The general idea of improvements is based on **informing** the sampling process.
- Many modifications of the algorithms exists, selected representative modifications are
  - **Informed RRT***;
  - Batch Informed Trees (**BIT***);
  - Regionally Accelerated BIT* (**RABIT***).

It shows an evolution of the improvements.

Informed RRT*

- Focused RRT* search to increase the convergence rate.
- Use Euclidean distance as an admissible heuristic.
- Ellipsoidal informed subset – the current best solution c_{best}

$$X_{\hat{f}} = \{x \in X \,|\, ||x_{start} - x||_2 + ||x - x_{goal}||_2 \le c_{best}\}.$$



Algorithm 2: Sample $(x_{start}, x_{goal}, c_{max})$

```
if c_max < ∞ then
    c_min ← ||x_goal − x_start||_2;
    x_centre ← (x_start + x_goal)/2;
    C ← RotationToWorldFrame(x_start, x_goal);
    r_1 ← c_max/2;
    {r_i}_{i=2,...,n} ← (sqrt(c_max^2 − c_min^2))/2;
    L ← diag{r_1, r_2, ..., r_n};
    x_ball ← SampleUnitNBall;
    x_rand ← (C L x_ball + x_centre) ∩ X;
else
    x_rand ~ U(X);
return x_rand;
```

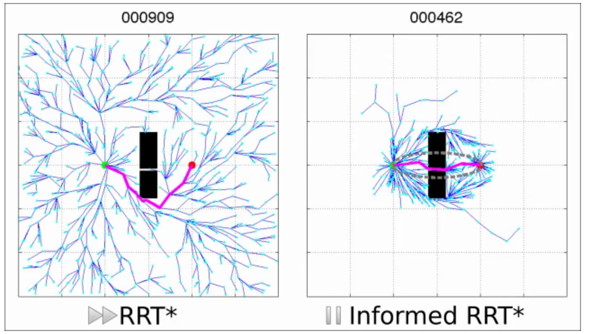
- Directly based on the RRT*.
- Having a feasible solution; item sample inside the ellipse.

Algorithm 1: Informed RRT*(x_{start}, x_{goal})

```
V ← {x_start};
E ← ∅;
X_soln ← ∅;
T = (V, E);
for iteration = 1 ... N do
    c_best ← min_{x_soln ∈ X_soln} {Cost(x_soln)};
    x_rand ← Sample(x_start, x_goal, c_best);
    x_nearest ← Nearest(T, x_rand);
    x_new ← Steer(x_nearest, x_rand);
    if CollisionFree(x_nearest, x_new) then
        V ← ∪ {x_new};
        X_near ← Near(T, x_new, r_RRT*);
        x_min ← x_nearest;
        c_min ← Cost(x_min) + c · Line(x_nearest, x_new);
        for ∀x_near ∈ X_near do
            c_new ← Cost(x_near) + c · Line(x_near, x_new);
            if c_new < c_min then
                if CollisionFree(x_near, x_new) then
                    x_min ← x_near;
                    c_min ← c_new;

        E ← E ∪ {(x_min, x_new)};
        for ∀x_near ∈ X_near do
            c_near ← Cost(x_near);
            c_new ← Cost(x_new) + c · Line(x_new, x_near);
            if c_new < c_near then
                if CollisionFree(x_new, x_near) then
                    x_parent ← Parent(x_near);
                    E ← E \ {(x_parent, x_near)};
                    E ← E ∪ {(x_new, x_near)};

        if InGoalRegion(x_new) then
            X_soln ← X_soln ∪ {x_new};

return T;
```

Gammell, J. B., Srinivasa, S. S., Barfoot, T. D.: **Informed RRT*:** *Optimal Sampling-based Path Planning Focused via Direct Sampling of an Admissible Ellipsoidal Heuristic*. IROS, 2014.

Informed RRT* – Demo



https://www.youtube.com/watch?v=d7dX5MvDYTc

Batch Informed Trees (BIT*)

- Combining RGG (Random Geometric Graph) with the heuristic in incremental graph search technique, e.g., Lifelong Planning A* (LPA*). *The properties of the RGG are used in the RRG and RRT*.*
- Batches of samples – a new batch starts with denser implicit RGG.
- The search tree is updated using LPA* like incremental search to reuse existing information.

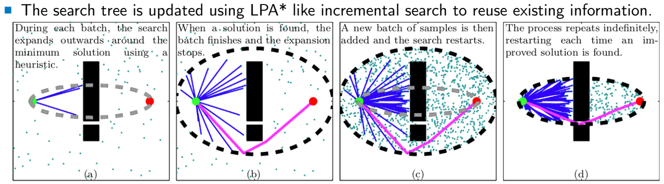
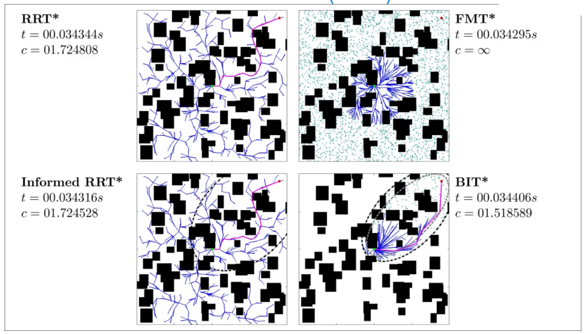


Fig. 3. An illustration of the informed search procedure used by BIT*. The start and goal states are shown as green and red, respectively. The current solution is highlighted in magenta. The subproblem that contains any better solutions is shown as a black dashed line, while the progress of the current batch is shown as a grey dashed line. Fig. (a) shows the growing search of the first batch of samples, and (b) shows the first search ending when a solution is found. After pruning and adding a second batch of samples, Fig. (c) shows the search restarting on a denser graph while (d) shows the second search ending when an improved solution is found. An animated illustration is available in the attached video.

Gammell, J. B., Srinivasa, S. S., Barfoot, T. D.: **Batch Informed Trees (BIT*)**: *Sampling-based optimal planning via the heuristically guided search of implicit random geometric graphs*, ICRA, 2015.

Batch Informed Trees (BIT*) – Demo



https://www.youtube.com/watch?v=TQIoCC48gp4

Regionally Accelerated BIT* (RABIT*)

- Use local optimizer with the BIT* to improve the convergence speed.
- Local search Covariant Hamiltonian Optimization for Motion Planning (CHOMP) is utilized to connect edges in the search graphs using local information about the obstacles.

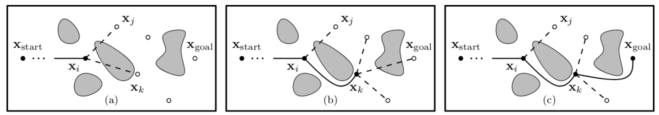
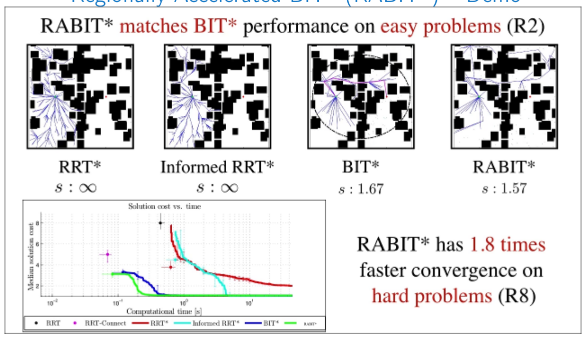


Fig. 2. An illustration of how the RABIT* algorithm uses a local optimizer to exploit obstacle information and improve a global search. The global search is performed, as in BIT*, by incrementally processing an edge queue (dashed lines) into a tree (a). Using heuristics, the potential edge from x_i to x_k is processed first as it could provide a better solution than an edge from x_i to x_j. The initial straight-line edge is given to a local optimizer which uses information about obstacles to find a local optima between the specified states (b). If this edge is collision free, it is added to the tree and its potential outgoing edges are added to the queue. The next-best edge in the queue is then processed in the same fashion, using the local optimizer to once again propose a better edge than a straight-line (c).

Choudhury, S., Gammell, J. D., Barfoot, T. D., Srinivasa, S. S., Scherer, S.: **Regionally Accelerated Batch Informed Trees (RABIT*)**: *A Framework to Integrate Local Information into Optimal Path Planning*. ICRA, 2016.

Regionally Accelerated BIT* (RABIT*) – Demo



https://www.youtube.com/watch?v=mgq-DW36jSo

Covariant Hamiltonian Optimization for Motion Planning (CHOMP)

- Trajectory optimization based on functional gradient techniques to improve the trajectory with trade-off between trajectory *smoothness* and *obstacle avoidance*.
- Trajectory function $\pi : [0, T] \rightarrow \mathcal{C}$ with a cost function $\mathcal{U} : \Pi \rightarrow \mathbb{R}^+$.
- The trajectory optimization $\pi^* = \mathrm{argmin}_{\pi \in \Pi} \mathcal{U}(\pi)$, s.t. $\pi(0) = q_{\mathrm{init}}$ and $\pi(T) = q_{\mathrm{goal}}$.
  Function gradient descent $\pi_{i+1} \leftarrow \frac{1}{\alpha} \nabla_\pi \mathcal{U}(\pi_i)$.
- CHOMP instantiates functional gradient descent for the cost

$$\mathcal{U}(\pi) = \mathcal{U}_{\mathrm{smooth}}(\pi) + \lambda \mathcal{U}_{\mathrm{obs}}(\pi). \quad (1)$$

- Smoothness cost can be defined as $\mathcal{U}_{\mathrm{smooth}}(\pi) = \frac{1}{2} \int_0^T \|\pi'(t)\|^2 \, dt$.
- Obstacle cost

$$\mathcal{U}_{\mathrm{obs}}(\pi) = \int_t \int_{\mathcal{A}} c(\psi_{\mathcal{A}}(\pi(t))) \cdot \left\| \frac{d}{dt} \psi_{\mathcal{A}}(\pi(t)) \right\| da dt. \quad (2)$$

  - The cost function in $\mathcal{W}$, $c : \mathcal{W} \rightarrow \mathbb{R}$ that uses signed distance field to computed distance to the closes obstacle.
    Return higher cost the closer the point is to an obstacle.
  - Computing the cost for each point of the trajectory, thus integral over time.
  - Integral over body points a using forward kinematics mapping $\psi_{\mathcal{A}}$ to get robot's points for $\pi(t)$.

Zucker, M., Ratliff, N., Dragan, A. D., Pivtoraiko, M., Klingensmith, M., Dellin, C. M., Bagnell, J. A., and Srinivasa, S. S.: *CHOMP: Covariant Hamiltonian optimization for motion planning*. The International Journal of Robotics Research. 32(9–10):1164–1193, 2013.

Overview of Improved Algorithm

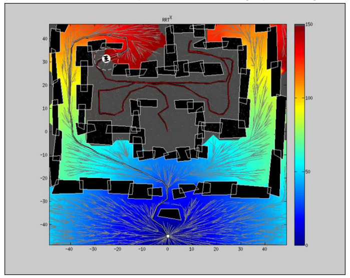
- Optimal path/motion planning is an active research field.

Approaches	Constraints	Planning Mode	Kinematic Model	Sampling Strategy	Metric
1. RRT* [7]	Holonomic	Offline	Point	Uniform	Euclidean
2. Anytime RRT* [4]	Non-holonomic	Online	Dubin Car	Uniform	Euclidean + Velocity
3. B-RRT* [58]	Holonomic	Offline	Rigid Body	Local bias	Goal biased
4. RRT*FN [33]	Holonomic	Offline	Robotic Arm	Uniform	Cumulative Euclidean
5. RRT*-Smart [35]	Holonomic	Offline	Point	Intelligent	Euclidean
6. Optimal B-RRT* [36]	Holonomic	Offline	Point	Uniform	Euclidean
7. RRT# [50]	Holonomic	Offline	Point	Uniform	Euclidean
8. Adapted RRT* [64], [49]	Non-holonomic	Offline	Car-like and UAV	Uniform	A* Heuristic
9. SRRT* [44]	Non-holonomic	Offline	UAV	Uniform	Geometric + dynamic constraint
10. Informed RRT* [34]	Holonomic	Offline	Point	Direct Sampling	Euclidean
11. IB-RRT* [37]	Holonomic	Offline	Point	Intelligent	Greedy + Euclidean
12. DT-RRT [39]	Non-holonomic	Offline	Car-like	Hybrid	Angular + Euclidean
13. RRT*i [3]	Non-holonomic	Online	UAV	Local Sampling	A* Heuristic
14. RTR+CS* [43]	Non-holonomic	Offline	Car-like	Uniform + Local Planning	Angular + Euclidean
15. Mitsubishi RRT* [2]	Non-holonomic	Online	Autonomous Car	Two-stage sampling	Weighted Euclidean
16. CARRT* [65]	Non-holonomic	Online	Humanoid	Uniform	MW Energy Cost
17. PRRT* [48]	Non-holonomic	Offline	P3-DX	Uniform	Euclidean

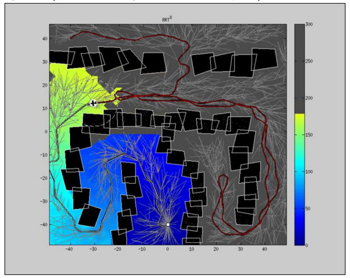
Noreen, I., Khan, A., Habib, Z.: *Optimal path planning using RRT* based approaches: a survey and future directions*. IJACSA, 2016.

Motion Planning for Dynamic Environments – RRT^{X}

- Refinement and repair of the search graph during the navigation (quick rewiring of the shortest path).



RRT^{X} – Robot in 2D
https://www.youtube.com/watch?v=S9pguCPUo3M



RRT^{X} – Robot in 2D
https://www.youtube.com/watch?v=KxFivNgTV4o

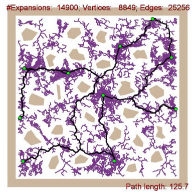
Otte, M., & Frazzoli, E. (2016). RRT^{X}: *Asymptotically optimal single-query sampling-based motion planning with quick replanning.* International Journal of Robotics Research, 35(7), 797–822.

Part III

Part 3 – Multi-goal Motion Planning (MGMP)

Multi-Goal Motion Planning

- In the previous cases, we consider existing roadmap or relatively "simple" collision free (shortest) paths in the polygonal domain.
- However, determination of the collision-free path in high dimensional configuration space ($\mathcal{C}$-space) can be a challenging problem itself.
- Therefore, we can generalize the MTP to multi-goal **motion** planning (MGMP) considering motion planners using the notion of $\mathcal{C}$-space for avoiding collisions.
- *An example of MGMP can be to plan a cost efficient trajectory for hexapod walking robot to visit a set of target locations.*

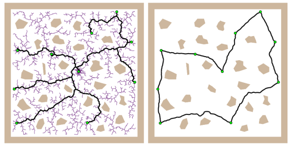


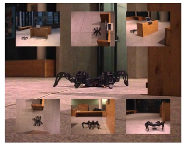
Problem Statement – MGMP Problem

- The working environment $\mathcal{W} \subset \mathbb{R}^3$ is represented as a set of obstacles $\mathcal{O} \subset \mathcal{W}$ and the robot configuration space $\mathcal{C}$ describes all possible configurations of the robot in $\mathcal{W}$.
- For $q \in \mathcal{C}$, the robot body $\mathcal{A}(q)$ at q is collision free if $\mathcal{A}(q) \cap \mathcal{O} = \emptyset$ and all collision free configurations are denoted as $\mathcal{C}_{free}$.
- Set of n goal locations is $\mathcal{G} = (g_1, \ldots, g_n)$, $g_i \in \mathcal{C}_{free}$.
- Collision free path from q_{start} to q_{goal} is $\kappa : [0,1] \to \mathcal{C}_{free}$ with $\kappa(0) = q_{start}$ and $d(\kappa(1), q_{end}) < \epsilon$, for an admissible distance ϵ.
- Multi–goal path τ is **admissible** if $\tau : [0,1] \to \mathcal{C}_{free}$, $\tau(0) = \tau(1)$ and there are n points such that $0 \le t_1 \le t_2 \le \ldots \le t_n$, $d(\tau(t_i), v_i) < \epsilon$, and $\bigcup_{1<i\le n} v_i = \mathcal{G}$.
- **The problem is to find the path τ^* for a cost function c such that $c(\tau^*) = \min\{c(\tau) \mid \tau$ *is admissible multi–goal path*$\}$.**

MGMP – Existing Approches

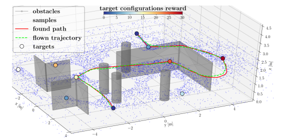
- Determining all paths connecting any two locations $g_i, g_j \in \mathcal{G}$ is usually very computationally demanding.
- Considering Euclidean distance as an approximation in the solution of the TSP as the Minimum Spanning Tree (MST) – Edges in the MST are iteratively refined using optimal motion planner until all edges represent a feasible solution. Saha, M., Roughgarden, T., Latombe, J.-C., Sánchez-Ante, G.: *Planning Tours of Robotic Arms among Partitioned Goals.*, International Journal of Robotics Research, 5(3):207–223, 2006
- **Synergistic Combination of Layers of Planning (SyCLoP)** – A combination of route and trajectory planning. Plaku, E., Kavraki, L.E., Vardi, M.Y. (2010): *Motion Planning With Dynamics by a Synergistic Combination of Layers of Planning*, IEEE Transactions on Robotics, 26(3):469–482, 2010.
- Steering RRG roadmap expansion by unsupervised learning for the TSP.
- Steering PRM* expansion using VNS-based routing planning in the **Physical Orienteering Problem** (POP).

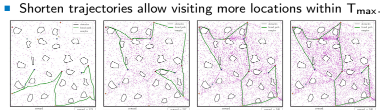


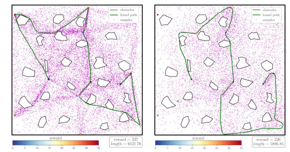


Multi-Goal Trajectory Planning with Limited Travel Budget Physical Orienteering Problem (POP)

- **Orienteering Problem (OP)** in an environment with obstacles and motion constraints of the data collecting vehicle.
- A combination of motion planning and routing problem with profits.
- **VNS-PRM*** – **VNS**-based routing and motion planning is addressed by **PRM***.
- An initial low-dense roadmap is continuously expanded during the VNS-based POP optimization to shorten paths of promising solutions.
- Shorten trajectories allow visiting more locations within T_{max}.

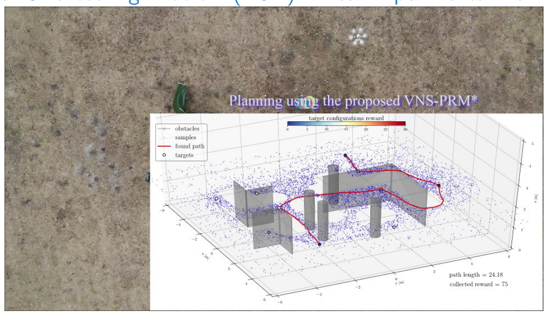






- Pěnička, Faigl and Saska: *Physical Orienteering Problem for Unmanned Aerial Vehicle Data Collection Planning in Environments with Obstacles.* IEEE Robotics and Automation Letters 4(3):3005–3012, 2019.

Multi-Goal Trajectory Planning with Limited Travel Budget Physical Orienteering Problem (POP) – Real Experimental Verification



Summary of the Lecture

Topics Discussed – Randomized Sampling-based Methods

- Single and multi-query approaches
  Probabilistic Roadmap Method (**PRM**); Rapidly Exploring Random Tree (**RRT**).
- Optimal sampling-based planning – Rapidly-exploring Random Graph (**RRG**).
- Properties of the sampling-based motion planning algorithms:
  - **Path**, **collision-free path**, **feasible path**;
  - **Feasible path planning** and **optimal path planning**;
  - **Probabilistic completeness**, **strong δ-clearance**, **robustly feasible** path planning problem;
  - **Asymptotic optimality**, **homotopy**, **weak δ-clearance**, **robust optimal solution**;
  - **PRM**, **RRT**, **RRG**, **PRM***, **RRT***.
- Improved randomized sampling-based methods
  - Informed sampling – Informed RRT*; Improving by batches of samples and reusing previous searches using Lifelong Planning A* (LPA*).
  - Improving local search strategy to improve convergence speed.
  - Planning in dynamic environments – RRT^{X}.
- Multi-goal motion planning (**MGMP**) problems are further variants of the robotic TSP.
- Next: Semestral project assignment, and Game Theory in Robotics (12th and 13rd week).