

Multi-Goal Path and Motion Planning

Jan Faigl

Department of Computer Science

Faculty of Electrical Engineering
Czech Technical University in Prague

Lecture 07

B4M36UIR – Artificial Intelligence in Robotics

Part I

Part 1 – Improved Sampling-based Motion Planning

Overview of the Lecture

- Part 1 – Improved Sampling-based Motion Planning
 - Improved Sampling-based Motion Planners
- Part 2 – Multi-Goal Path and Motion Planning
 - Multi-Goal Path Planning
 - Multi-Goal Motion Planning
 - Multi-Goal Planning in Robotic Missions

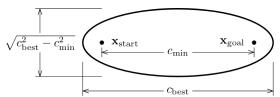
Improved Sampling-based Motion Planners

- Although Asymptotically optimal sampling-based motion planners such RRT* or RRG may provide high-quality or even optimal solutions of complex problem, their performance in simple, e.g., 2D scenarios, is relatively poor In a comparison to the previous approaches
- They are computationally demanding and performance can be improved similarly as RRT, e.g.,
 - Goal biasing, supporting sampling in narrow passages, multi-tree grows (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***)

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}

$$X_f = \{x \in X \mid \|x_{start} - x\|_2 + \|x - x_{goal}\|_2 \leq c_{best}\}$$



- Directly Based on the RRT*
- Having a feasible solution
- Sampling inside the ellipse

```

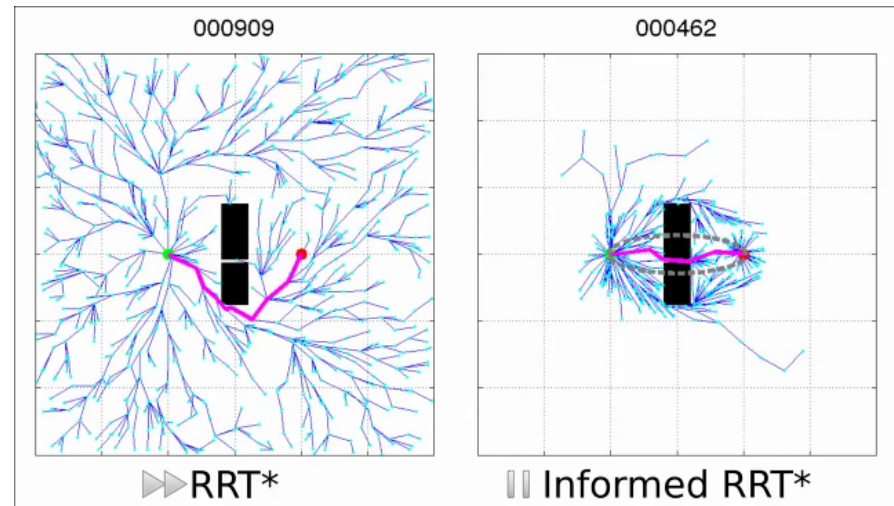
Algorithm 2: Sample (X_start, X_goal, c_max)
1 if c_max < ∞ then
2   c_min ← ||X_start - X_goal||_2;
3   X_near ← (X_start + X_goal) / 2;
4   C ← RotationToWorldFrame (X_start, X_goal);
5   r_j ← c_max / 2;
6   (r_1)_min ← ((sqrt(c_max^2 - c_min^2)) / 2);
7   L ← dirg (r_1, r_2, ..., r_n);
8   S_half ← SampleUnitBall;
9   X_half ← (CL(S_half + X_near)) ∩ X;
10 else
11   X_half ← U(X);
12 return X_half;
    
```

```

Algorithm 1: Informed RRT*(X_start, X_goal)
1 V ← {X_start};
2 E ← ∅;
3 X_soln ← ∅;
4 T ← (V, E);
5 for iteration = 1...N do
6   c_best ← min_{X_soln ∈ X_soln} {Cost (X_soln)};
7   X_rand ← Sample (X_start, X_goal, c_best);
8   X_nearest ← Nearest (T, X_rand);
9   X_new ← Steer (X_nearest, X_rand);
10  if CollisionFree (X_nearest, X_new) then
11    V ← V ∪ {X_new};
12    X_near ← Near (T, X_new, r_RRT*);
13    X_min ← X_nearest;
14    c_min ← Cost (X_min) + c · Line (X_nearest, X_new);
15    for X_near ∈ X_near do
16      c_new ← Cost (X_near) + c · Line (X_near, X_new);
17      if c_new < c_min then
18        if CollisionFree (X_near, X_new) then
19          X_min ← X_near;
20          c_min ← c_new;
21
22  E ← E ∪ {(X_min, X_new)};
23  for X_near ∈ X_near do
24    c_near ← Cost (X_near);
25    c_new ← Cost (X_new) + c · Line (X_new, X_near);
26    if c_new < c_near then
27      if CollisionFree (X_new, X_near) then
28        X_parent ← Parent (X_new);
29        E ← E ∪ {(X_parent, X_new)};
30        E ← E ∪ {(X_new, X_near)};
31
32  if InGoalRegion (X_new) then
33    X_soln ← X_soln ∪ {X_new};
34
35 return T;
    
```

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

Informed RRT* – Demo



<https://www.youtube.com/watch?v=d7dX5MvDYtC>

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

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

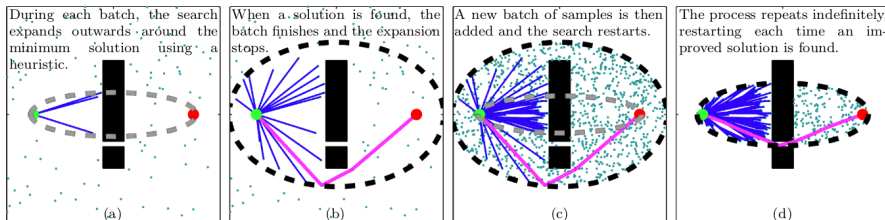
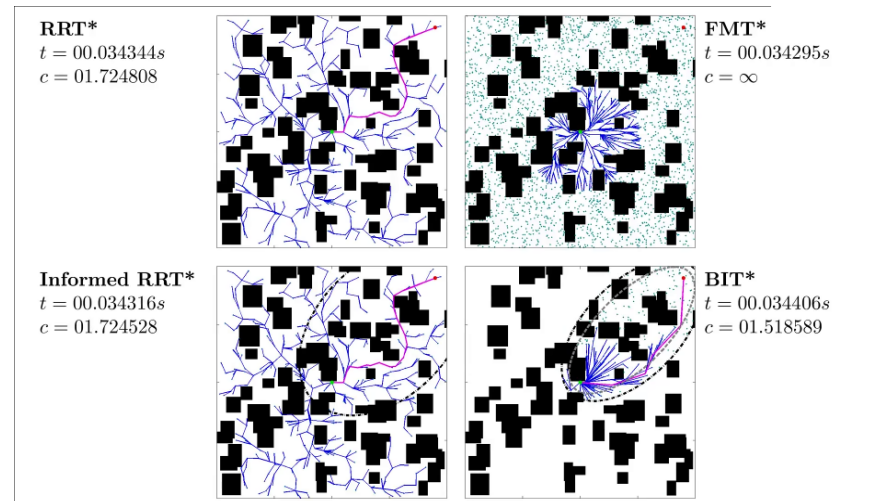


Fig. 3. An illustration of the informed search procedure used by BIT*. The start and goal states are shown as green and red, respectively. The current solution is highlighted in magenta. The subproblem that contains any better solutions is shown as a black dashed line, while the progress of the current 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. (2015): **Batch Informed Trees (BIT*)**: Sampling-based optimal planning via the heuristically guided search of implicit random geometric graphs. ICRA.

Batch Informed Trees (BIT*) – Demo



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

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

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.

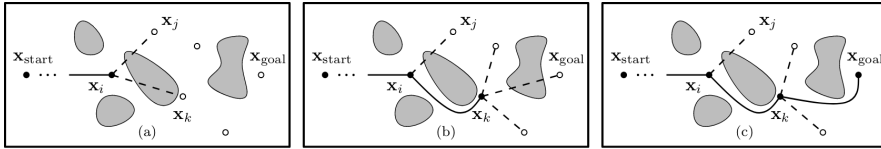
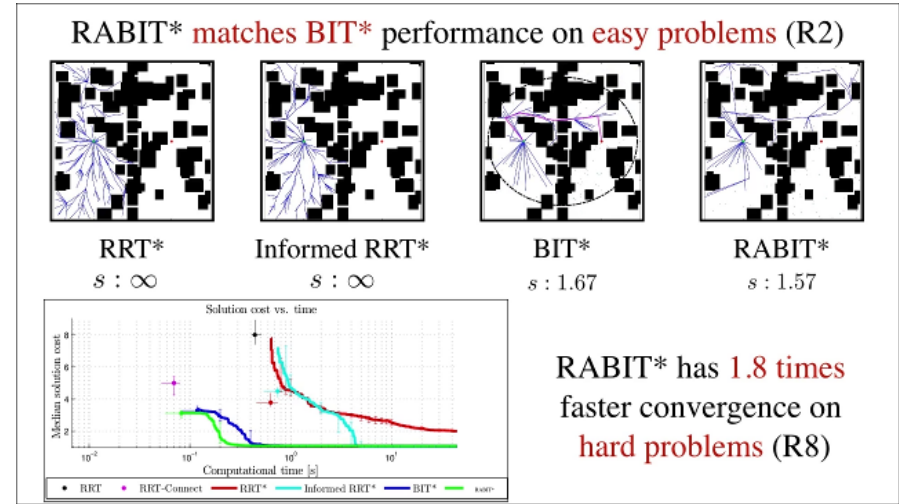


Fig. 2. An illustration of how the RABIT* algorithm uses a local optimizer to exploit obstacle information and improve a global search. The global search is performed, as in BIT*, by incrementally processing an edge queue (dashed lines) into a tree (a). Using heuristics, the potential edge from x_i to x_j is processed first as it could provide a better solution than an edge from x_i to x_k . 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. (2016): **Regionally Accelerated Batch Informed Trees (RABIT*)**: A Framework to Integrate Local Information into Optimal Path Planning. ICRA.

Regionally Accelerated BIT* (RABIT*) – Demo



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

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

Overview of Improved Algorithm

- Optimal 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. (2016): Optimal path planning using RRT* based approaches: a survey and future directions. IJACSA.

Part II

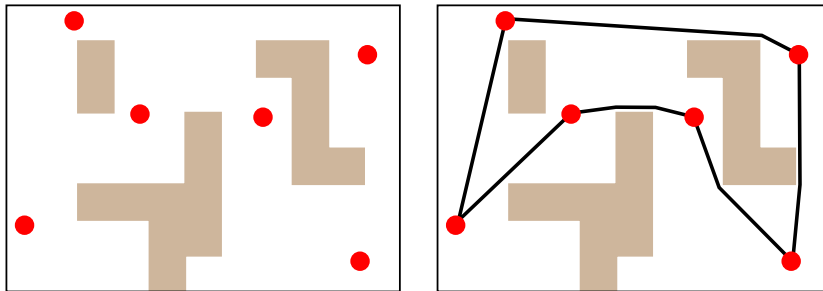
Part 2 – Multi-Goal Path and Motion Planning

Multi-Goal Path Planning

Motivation

Having a set of locations (goals) to be visited, determine the cost-efficient path to visit them and return to a starting location.

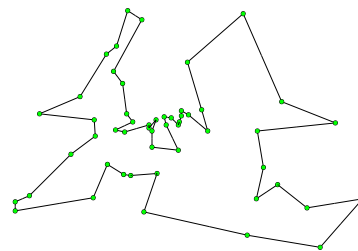
- Locations where a robotic arm performs some task
- Locations where a mobile robot has to be navigated
To perform measurements such as scan the environment or read data from sensors.



Alatartsev et al. (2015) – Robotic Task Sequencing Problem: A Survey

Solutions of the TSP

- Efficient heuristics from the Operational Research have been proposed
- LKH – K. Helsgaun efficient implementation of the Lin-Kernighan heuristic (1998)
<http://www.akira.ruc.dk/~keld/research/LKH/>
- Concorde – Solver with several heuristics and also optimal solver
<http://www.math.uwaterloo.ca/tsp/concorde.html>



Problem Berlin52 from the TSPLIB

Beside the heuristic and approximations algorithms (such as Christofides 3/2-approximation algorithm), other („soft-computing”) approaches have been proposed, e.g., based on genetic algorithms, and memetic approaches, ant colony optimization (ACO), and **neural networks**.

Traveling Salesman Problem (TSP)

Given a set of cities and the distances between each pair of cities, what is the shortest possible route that visits each city exactly once and returns to the origin city.

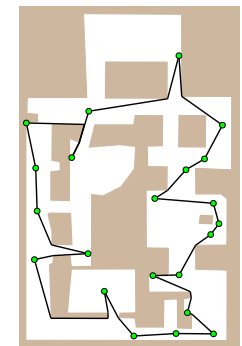
- The TSP can be formulated for a graph $G(V, E)$, where V denotes a set of locations (cities) and E represents edges connecting two cities with the associated travel cost c (distance), i.e., for each $v_i, v_j \in V$ there is an edge $e_{ij} \in E$, $e_{ij} = (v_i, v_j)$ with the cost c_{ij} .
- If the associated cost of the edge (v_i, v_j) is the Euclidean distance $c_{ij} = |(v_i, v_j)|$, the problem is called the **Euclidean TSP** (ETSP).
In our case, $v \in V$ represents a point in \mathbb{R}^2 and solution of the ETSP is a path in the plane.
- It is known, the TSP is NP-hard (its decision variant) and several algorithms can be found in literature.

William J. Cook (2012) – In Pursuit of the Traveling Salesman: Mathematics at the Limits of Computation

Multi-Goal Path Planning (MTP) Problem

Given a map of the environment \mathcal{W} , mobile robot \mathcal{R} , and a set of locations, what is the shortest possible **collision free path** that visits each location exactly once and returns to the origin location.

- MTP problem is de facto the TSP with the cost associated to the edges as the length of the *shortest* path connecting the locations
- For n locations, we need to compute up to n^2 shortest paths (solve n^2 motion planning problems)
- The paths can be found as the shortest path in a graph (roadmap), from which the $G(V, E)$ for the TSP can be constructed



Visibility graph as the roadmap for a point robot provides a straight forward solution, but such a shortest path may not be necessarily feasible for more complex robots

Multi-Goal Motion Planning

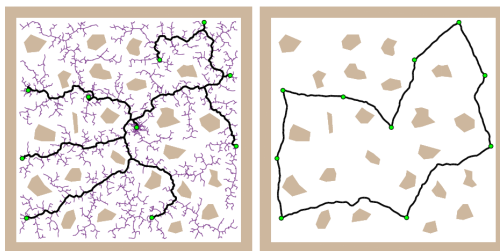
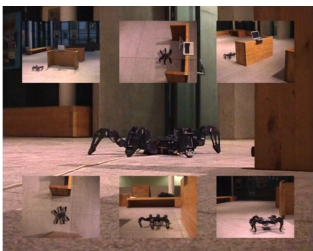
- In the previous cases, we consider existing roadmap or relatively “simple” collision free (shortest) paths in the polygonal domain
- However, determination of the collision-free path in a high dimensional configuration space (\mathcal{C} -space) can be a challenging problem itself
- Therefore, we can generalize the MTP to multi-goal **motion** planning (MGMP) considering motion (trajectory) planners in \mathcal{C} -space.
- An example of MGMP can be

Plan a cost efficient trajectory for hexapod walking robot to visit a set of target locations.



MGMP – Examples of Solutions

- Determination of all paths connecting any two locations $g_i, g_j \in \mathcal{G}$ is usually very computationally demanding
- Several approaches can be found in literature, e.g.,
 - Considering Euclidean distance as approximation in 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. (2006): Planning Tours of Robotic Arms among Partitioned Goals. IJRR.
 - **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. T-RO.
 - Steering RRG roadmap expansion by unsupervised learning for the TSP



Faigl (2016), WSOM

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 $\mathcal{G} = (g_1, \dots, g_n)$, $g_i \in \mathcal{C}_{free}$
- Collision free path from q_{start} to q_{goal} is $\kappa : [0, 1] \rightarrow \mathcal{C}_{free}$ with $\kappa(0) = q_{start}$ and $d(\kappa(1), q_{end}) < \epsilon$, for an admissible distance ϵ
- **Multi-goal path τ is admissible** if $\tau : [0, 1] \rightarrow \mathcal{C}_{free}$, $\tau(0) = \tau(1)$ and there are n points such that $0 \leq t_1 \leq t_2 \leq \dots \leq t_n$, $d(\tau(t_i), v_i) < \epsilon$, and $\bigcup_{1 < i \leq n} v_i = \mathcal{G}$
- **The problem is to find path τ^* for a cost function c such that $c(\tau^*) = \min\{c(\tau) \mid \tau \text{ is admissible multi-goal path}\}$**

Multi-Goal Path Planning in Robotic Missions

Multi-goal path planning

- It builds on a simple path and trajectory planning
- It is a **combinatorial optimization problem** to **determine the sequence** to visit the given locations
- It allows to solve (or improve performance of) more complex problems such as
 - **Inspection planning** - Find the shortest tour to see (inspect) the whole environment
 - **Data collection planning** – Determine a cost efficient path to collect data from the sensor stations (locations)
 - **Robotic exploration** - Create a map of unknown environment as quickly as possible

Inspection Planning

Motivations (examples)

- Periodically visit particular locations of the environment to check, e.g., for intruders, and return to the starting locations
- Based on available plans, provide a guideline how to search a building to find possible victims as quickly as possible (search and rescue scenario)



Inspection Planning – Decoupled Approach

1. Determine sensing locations such that the whole environment would be inspected (seen) by visiting them

A solution of the Art Gallery Problem

2. Create a roadmap connecting the sensing location

E.g., using visibility graph or randomized sampling based approaches

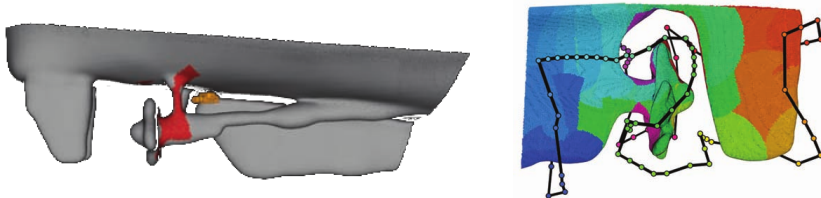
3. Find the inspection path visiting all the sensing locations as a solution of the multi-goal path planning

De facto solution of the TSP

Inspection planning is also called coverage path planning in literature.

Example – Inspection Planning with AUV

- Determine shortest inspection path for Autonomous Underwater Vehicle (AUV) to inspect a propeller of the vessel

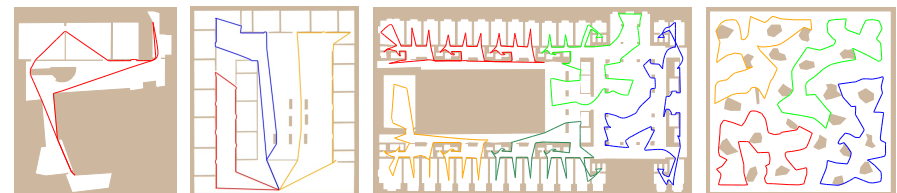


Three-dimensional coverage planning for an underwater inspection robot
 Brendan Englot and Franz S. Hover
 International Journal of Robotic Research, 32(9-10):1048–1073, 2013.

Inspection Planning – “Continuous Sensing”

- If we do not prescribe a discrete set of sensing locations, we can formulate the problem as the **Watchman route problem**

Given a map of the environment \mathcal{W} determine the shortest, closed, and collision-free path, from which the whole environment is covered by an omnidirectional sensor with the radius ρ .



Approximate Solution of the Multiple Watchman Routes Problem with Restricted Visibility Range
 Jan Faigl
 IEEE Transactions on Neural Networks, 21(10):1668–1679, 2010.

Self-Organizing Maps based Solution of the TSP

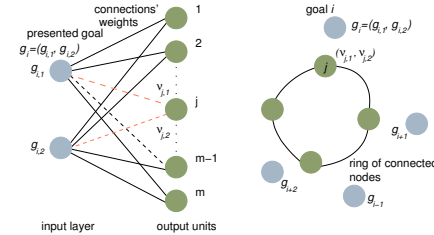
Kohonen's type of unsupervised two-layered neural network

- Neurons' **weights** represent **nodes** $\mathcal{N} = \{\nu_1, \dots, \nu_m\}$ in a **plane**.
- Nodes are organized into a **ring**.
- Sensing locations $\mathbf{S} = \{s_1, \dots, s_n\}$ are presented to the network in a **random** order.
- Nodes **compete** to be winner according to their distance to the presented goal s

$$\nu^* = \operatorname{argmin}_{\nu \in \mathcal{N}} |\mathcal{D}(\nu, s)|$$

- The **winner** and its **neighbouring** nodes are adapted (**moved**) towards the city according to the neighbouring function

$$f(\sigma, d) = \begin{cases} e^{-\frac{d^2}{\sigma^2}} & \text{for } d < m/n_f, \\ 0 & \text{otherwise,} \end{cases}$$

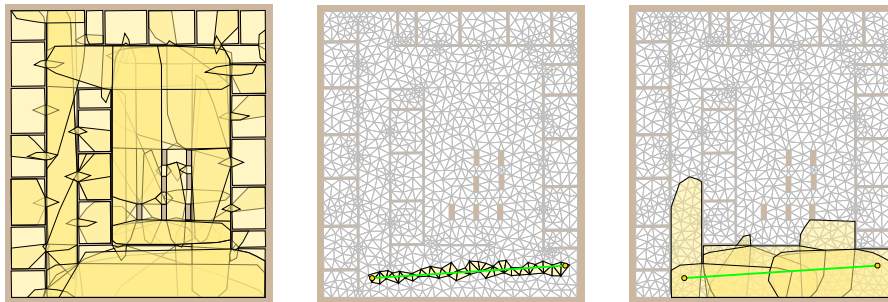


- Best matching unit ν to the presented prototype s is determined according to distance function $|\mathcal{D}(\nu, s)|$
- For the Euclidean TSP, \mathcal{D} is the Euclidean distance
- However, for problems with obstacles, the multi-goal path planning, \mathcal{D} should correspond to the length of the shortest, collision free path.

SOM for the TSP in the Watchman Route Problem

During the unsupervised learning, we can compute **coverage** of \mathcal{W} from the current **ring** (solution represented by the neurons) and **adapt** the network **towards uncovered parts** of \mathcal{W}

- Convex cover set of \mathcal{W} created on top of a triangular mesh
- Incident convex polygons with a straight line segment are found by walking in a triangular mesh technique



Jan Faigl (2010), TNN

The

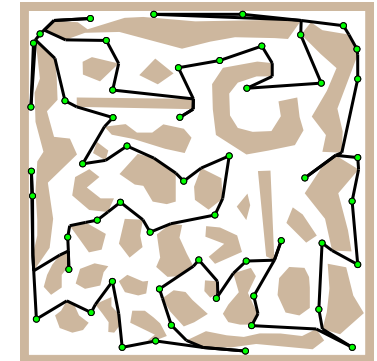
SOM for the Multi-Goal Path Planning

Unsupervised learning procedure

Algorithm 1: SOM-based MTP solver

```

 $\mathcal{N} \leftarrow$  initialization( $\nu_1, \dots, \nu_m$ );
repeat
  error  $\leftarrow$  0;
  foreach  $g \in \Pi(\mathbf{S})$  do
     $\nu^* \leftarrow$ 
    selectWinner argmin $_{\nu \in \mathcal{N}} |S(g, \nu)|$ ;
    adapt( $S(g, \nu)$ ,  $\mu f(\sigma, l) |S(g, \nu)|$ );
    error  $\leftarrow$  max{error,  $|S(g, \nu^*)|$ };
   $\sigma \leftarrow (1 - \alpha) \cdot \sigma$ ;
until error  $\leq$   $\delta$ ;
```



- For multi-goal path planning – the **selectWinner** and **adapt** procedures are based on the solution of the path planning problem



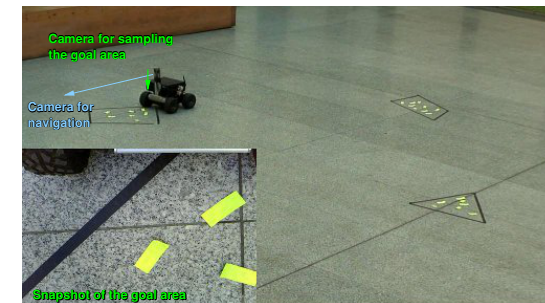
An Application of Self-Organizing Map in the non-Euclidean Traveling Salesman Problem

Jan Faigl, Miroslav Kulich, Vojtěch Vonásek and Libor Přeučil
Neurocomputing, 74(5):671–679, 2011.

Multi-Goal Path Planning with Goal Areas

- It may be sufficient to visit a goal region instead of the particular point location

E.g., to take a sample measurement at each goal



Not only a sequence of goals visit has to be determined, but also an appropriate sensing location for each goal need to be found.

The problem with goal regions can be considered as a variant of the **Traveling Salesman Problem with Neighborhoods (TSPN)**.

Traveling Salesman Problem with Neighborhoods

Given a set of n regions (neighbourhoods), what is the shortest closed path that visits each region.

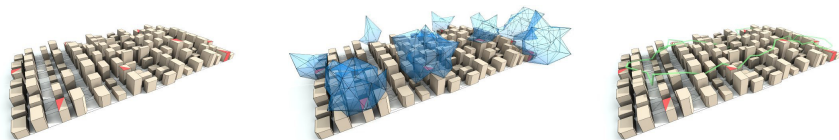
- The problem is NP-hard and APX-hard, it cannot be approximated to within factor $2 - \epsilon$, where $\epsilon > 0$

Safra and Schwartz (2006) – Computational Complexity

- Approximate algorithms exist for particular problem variants
E.g., Disjoint unit disk neighborhoods
- Flexibility of SOM for the TSP allows generalizing the unsupervised learning procedure to address the TSPN
- TSPN provides a suitable problem formulation for planning various inspection and data collection missions**

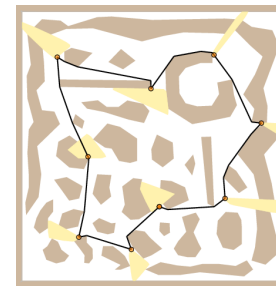
Example – TSPN for Inspection Planning with UAV

- Determine a cost-efficient trajectory from which a given set of target regions is covered
- For each target region a subspace $S \subset \mathbb{R}^3$ from which the target can be covered is determined
S represents the neighbourhood
- The PRM motion planning algorithm is utilized to construct a motion planning roadmap (a graph)
- SOM based solution of the TSP with a graph input is generalized to the TSPN



Janoušek and Faigl, (2013) – ICRA

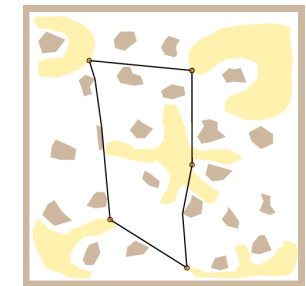
SOM-based Solution of the Traveling Salesman Problem with Neighborhoods (TSPN)



Polygonal Goals
 $n=9, T=0.32$ s



Convex Cover Set
 $n=106, T=5.1$ s



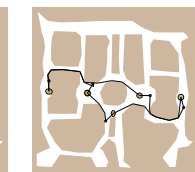
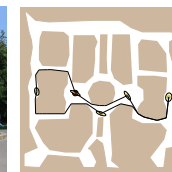
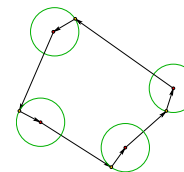
Non-Convex Goals
 $n=5, T=0.1$ s



Visiting Convex Regions in a Polygonal Map,
Jan Faigl, Vojtěch Vonásek and Libor Přeučil
Robotics and Autonomous Systems, 61(10):1070–1083, 2013.

Example – TSPN for Planning with Localization Uncertainty

- Selection of waypoints from the neighborhood of each location
- P3AT ground mobile robot in an outdoor environment



TSP: $L=184$ m,
 $E_{avg}=0.57$ m

TSPN: $L=202$ m,
 $E_{avg}=0.35$ m

Real overall error at the goals decreased from 0.89 m → 0.58 m (about 35%)

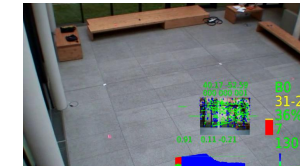
- Decrease localization error at the target locations (indoor)

Small UGV - MMP5



Error decreased from 16.6 cm → 12.8 cm

Small UAV - Parrot AR.Drone



Improved success of the locations' visits 83% → 95%

Faigl et al., (2012) – ICRA

Summary of the Lecture

Topics Discussed

- Improved sampling-based motion planners
- Multi-goal planning
 - Robotic variant of the Traveling Salesman Problem (TSP)
 - Multi-Goal Path Planning (MTP) problem
 - Multi-Goal Motion Planning (MGMP) problem
- Multi-goal planning in robotic missions
 - Traveling Salesman Problem with Neighborhoods (TSPN)
 - Inspection planning

- Next: Data collection planning