Multi-goal Planning

Jan Faigl

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

Lecture 06

B4M36UIR – Artificial Intelligence in Robotics

Overview of the Lecture

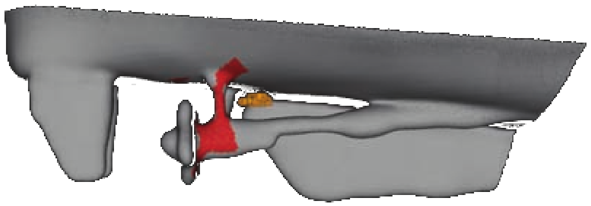
- Part 1 – Multi-goal Planning
  - Inspection Planning
  - Multi-goal Planning
- Part 2 – Unsupervised Learning for Multi-goal Planning
  - Unsupervised Learning for Multi-goal Planning
  - TSPN in Multi-goal Planning with Localization Uncertainty

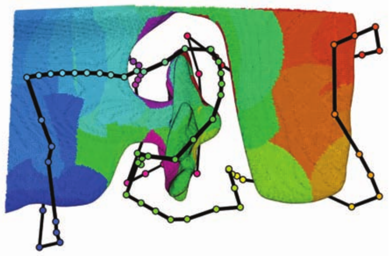
Part I

Part 1 – Multi-goal Planning

Robotic Information Gathering in Inspection of Vessel's Propeller

- The planning problem is to determine a shortest inspection path for an Autonomous Underwater Vehicle (AUV) to inspect the vessel's propeller.



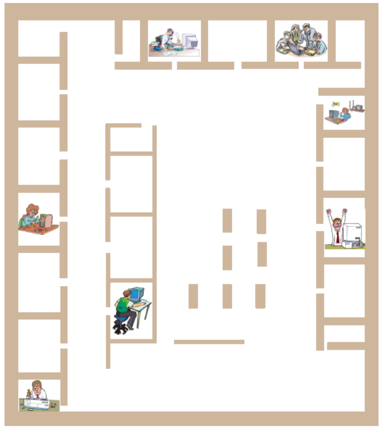


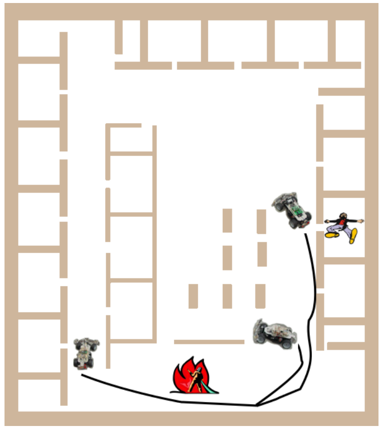
https://www.youtube.com/watch?v=8azP_9VnMtM

Englot, B., Hover, F.S.: *Three-dimensional coverage planning for an underwater inspection robot*, International Journal of Robotics Research, 32(9–10):1048–1073, 2013.

Example of Inspection Planning in Search Scenario

- Periodically visit particular locations of the environment and return to the starting locations.
- Use available floor plans to guide the search, e.g., finding victims in search-and-rescue scenario.

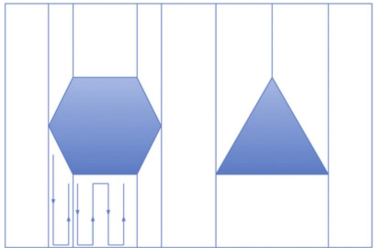




Inspection Planning

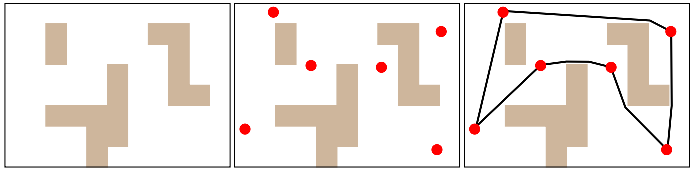
- **Inspection/coverage planning** stands to determine a plan (path) to inspect/cover the given areas or point of interest.
- We can directly find inspection/coverage plan using
  - predefined covering patterns such as *ox-plow* motion;
  - a "general" path satisfying coverage constraints.

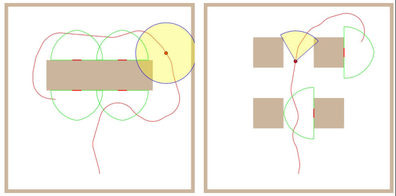
Galceran, E., Carreras, M.: *A survey on coverage path planning for robotics*, Robotics and Autonomous Systems, 61(12):1258–1276, 2013.



Trapezoidal decomposition method

- **Decoupled** approach – Locations to be visited are determined before path planning as the **sensor placement** problem.



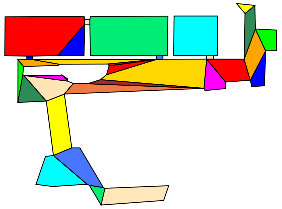


Kafka, Faigl, Váňa: ICRA 2016

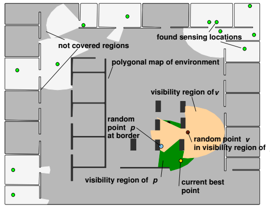
Inspection Planning – Decoupled Approach

1. Determine sensing locations such that the whole environment would be inspected (seen) by visiting them. It is **Sampling design problem**.

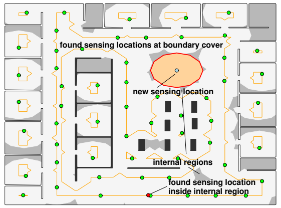
In the geometrical-based approach, a solution of the Art Gallery Problem.



Convex Partitioning (Kazazakis and Argyros, 2002)



Randomized Dual Sampling (González-Baños et al., 1998)



found sensing locations at boundary cover
new sensing location
internal regions
found sensing location inside internal region

Boundary Placement (Faigl et al., 2006)

The problem is related to the ***sensor placement*** *and* ***sampling design****.*

2. Create a roadmap connecting the sensing location.

Such as visibility graph or randomized sampling-based method.

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

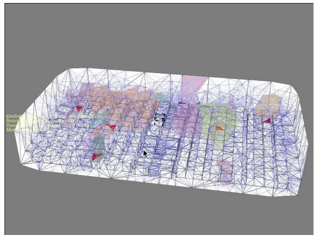
Planning to Capture Areas of Interest using UAV

- Determine a cost-efficient path 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 neighborhood.*
- **We search for the best sequence of visits to the regions.**

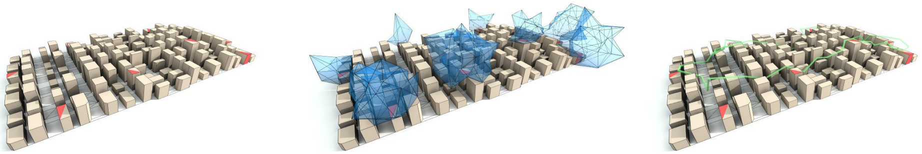
Combinatorial optimization

- The PRM is utilized to construct the planning roadmap (a graph).

PRM – Probabilistic Roadmap Method – sampling-based motion planner, see lecture 8.



- The problem can be formulated as **the Traveling Salesman Problem with Neighborhoods**, as it is not necessary to visit exactly a single location to capture the area of interest.

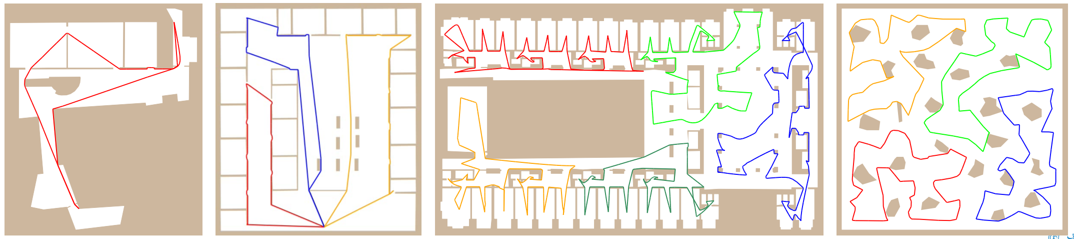


Janoušek and Faigl, ICRA 2013

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**.

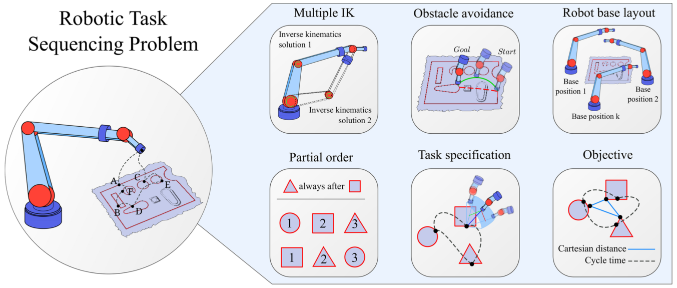
> Given a map of the environment $\mathcal{W}$ determine the shortest, closed, and collision-free path, from which the whole environment is covered by an omnidirectional sensor with the radius ρ.



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

Multi-Goal Planning

- Having a **set of locations** to be visited, determine the cost-efficient path to visit them.
  - Locations where a robotic arm or mobile robot performs some task. *The operation can be repeated–closed path.*
- The problem is called **robotic task sequencing problem** for robotic manipulators.



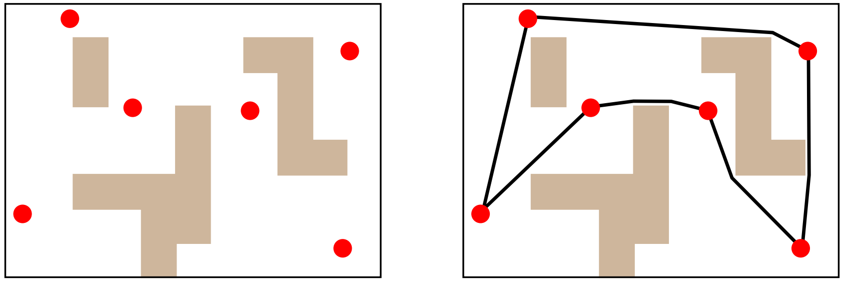
Alatartsev, S., Stellmacher, S., Ortmeier, F. (2015): Robotic Task Sequencing Problem: A Survey. Journal of Intelligent & Robotic Systems.

- The problem is also called **Multi-goal Path Planning** (MTP) problem or **Multi-goal Planning** (**MGP**).

Also studied in its Multi-goal Motion Planning (MGMP) variant.

Multi-Goal Path Planning (MTP)

- Multi-goal planning problem is a problem how to visit the given set of locations.
- It consists of **point-to-point path planning** on how to reach one location from another.
- The *challenge* is to determine the optimal sequence of the visits to the locations w.r.t. cost-efficient path to visit all the given locations.



- Determination the sequence of visits is a **combinatorial optimization problem** that can be formulated as the **Traveling Salesman Problem** (**TSP**).

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 $\boldsymbol{G}(V, E)$, 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).
- 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.

Traveling Salesman Problem (TSP)

- Let S be a set of n sensor locations $S = \{\boldsymbol{s}_1, \ldots, \boldsymbol{s}_n\}$, $\boldsymbol{s}_i \in \mathbb{R}^2$ and $c(\boldsymbol{s}_i, \boldsymbol{s}_j)$ is a cost of travel from $\boldsymbol{s}_i$ to $\boldsymbol{s}_j$.
- **Traveling Salesman Problem** (**TSP**) is a problem to determine a closed tour visiting each $\boldsymbol{s} \in S$ such that the total tour length is minimal.
  - We are searching for the optimal **sequence of visits** $\Sigma = (\sigma_1, \ldots, \sigma_n)$ such that

$$\begin{aligned} &\text{minimize}_{\Sigma} \quad && L = \left(\sum_{i=1}^{n-1} c(\boldsymbol{s}_{\sigma_i}, \boldsymbol{s}_{\sigma_{i+1}}) \right) + c(\boldsymbol{s}_{\sigma_n}, \boldsymbol{s}_{\sigma_1}) \\ &\text{subject to} && \Sigma = (\sigma_1, \ldots, \sigma_n), 1 \le \sigma_i \le n, \sigma_i \ne \sigma_j \text{ for } i \ne j. \end{aligned} \tag{1}$$

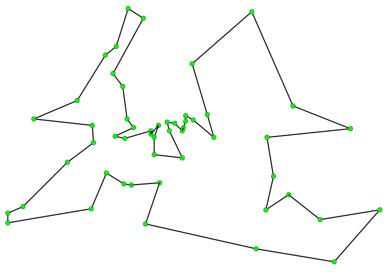
- The TSP can be considered on a graph $G(V, E)$ where the set of vertices V represents sensor locations S and E are edges connecting the nodes with the cost $c(\boldsymbol{s}_i, \boldsymbol{s}_j)$.
- For simplicity we can consider $c(\boldsymbol{s}_i, \boldsymbol{s}_j)$ to be Euclidean distance; otherwise, we also need to address the path/motion planning problem. **Euclidean TSP**
- If $c(\boldsymbol{s}_i, \boldsymbol{s}_j) \ne c(\boldsymbol{s}_j, \boldsymbol{s}_i)$ it is the **Asymmetric TSP**.
- The TSP is known to be NP-hard unless P=NP.

Traveling vs Travelling – http://www.math.uwaterloo.ca/tsp/history/travelling.html

Existing Approaches to the TSP

- Exact solutions
  - Branch&Bound, Branch&Cut, and Integer Linear Programming (ILP).

    Concorde-http://www.math.uwaterloo.ca/tsp/concorde.html
- Approximation algorithms
  - Minimum Spanning Tree (MST) heuristic with $L \le 2L_{opt}$.
  - Christofides's algorithm with $L \le \frac{3/2}{L}{}_{opt}$.
- Heuristic algorithms
  - Constructive heuristic – Nearest Neighborhood (NN) algorithm;
  - 2-Opt – local search algorithm proposed by Croes 1958;
  - **LKH** – K. Helsgaun efficient implementation of the Lin-Kernighan heuristic (1998). http://www.akira.ruc.dk/~keld/research/LKH/
- Combinatorial meta-heuristics
  - **Variable Neighborhood Search** (**VNS**);
  - **Greedy Randomized Adaptive Search Procedure** (**GRASP**).
- Soft-computing techniques, evolutionary methods, and unsupervised learning.

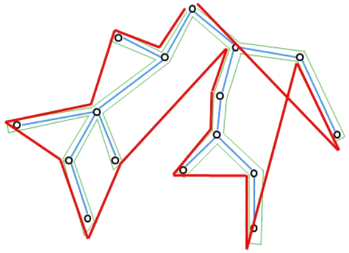


Problem Berlin52 from the TSPLIB

MST-based Approximation Algorithm to the TSP

- Minimum Spanning Tree heuristic

1. Compute the MST (denoted T) of the input graph G.
2. Construct a graph H by doubling every edge of T.
3. Shortcut repeated occurrences of a vertex in the tour.



- For the triangle inequality, the length of such a tour L is

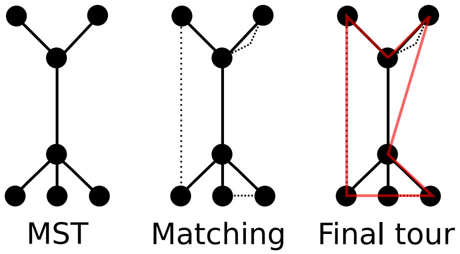
$$L \le 2L_{optimal},$$

where $L_{optimal}$ is the cost of the optimal solution of the TSP.

Christofides's Algorithm to the TSP

- Christofides's algorithm

1. Compute the MST of the input graph G.
2. Compute the minimal matching on the odd-degree vertices.
3. Shortcut a traversal of the resulting Eulerian graph.



- For the triangle inequality, the length of such a tour L is

$$L \le \frac{3}{2} L_{optimal},$$

where $L_{optimal}$ is the cost of the optimal solution of the TSP.

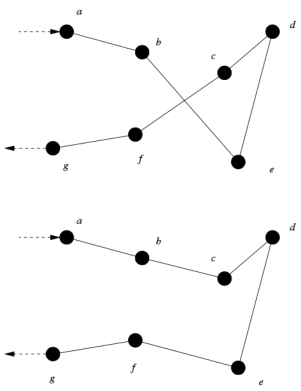
Length of the MST is $\le L_{optimal}$

Sum of lengths of the edges in the matching $\le \frac{1}{2} L_{optimal}$

2-Opt Heuristic

1. Use a construction heuristic to create an initial route
   - NN algorithm, cheapest insertion, farther insertion
2. Repeat until no improvement is made

   2.1 Determine swapping that can shorten the tour (i, j) for $1 \leq i \leq n$ and $i + 1 \leq j \leq n$
   - route[0] to route[i-1];
   - route[i] to route[j] in reverse order;
   - route[j] to route[end];
   - Determine length of the route;
   - Update the current route if the length is shorter than the existing solution.



Croes, G.A.: *A method for solving traveling salesman problems*, Operations Research 6:791–812, 1958.

Overview of the Variable Neighborhood Search (VNS) for the TSP

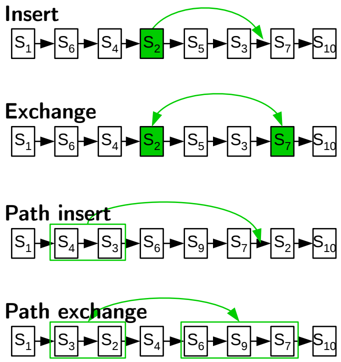
- **Variable Neighborhood Search** (**VNS**) is a metaheuristic for solving combinatorial optimization and global optimization problems by searching distant neighborhoods of the current **incumbent solution** using **shake** and **local search**.

Mladenović and Hansen, 1997

1. **Shake** procedure explores the neighborhood of the current solution to escape from a local minima using operators:
   - **Insert** – moves one element;
   - **Exchange** – exchanges two elements.
2. **Local search** procedure improves the solution by
   - **Path insert** – moves a subsequence;
   - **Path exchange** – exchanges two subsequences.

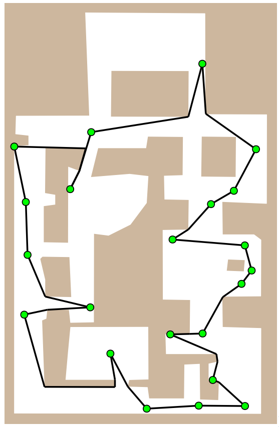
Algorithm 1: VNS-based Solver to the TSP

```
Input: $S$ – Set of the target locations to be visited.
Output: $\Sigma$ – Found sequence of visits to locations $S$.
$\Sigma^* \leftarrow$ Initial sequence found by cheapest insertion
while terminal condition is not met do
    $\Sigma' \leftarrow$ shake($\Sigma^*$)
    for $n^2$-times do
        $\Sigma'' \leftarrow$ localSearch($\Sigma'$)
        if $\Sigma''$ is "better" than $\Sigma'$ then
            $\Sigma' \leftarrow \Sigma''$    // Select $\Sigma''$ instead of $\Sigma'$
    if $\Sigma'$ is "better" than $\Sigma^*$ then
        $\Sigma^* \leftarrow \Sigma'$    // Replace the incumbent sequence.
return $\Sigma^*$
```


Insert
S_1 S_6 S_4 S_2 S_5 S_3 S_7 S_{10}
Exchange
S_1 S_6 S_4 S_2 S_5 S_3 S_7 S_{10}
Path insert
S_1 S_4 S_3 S_6 S_9 S_7 S_2 S_{10}
Path exchange
S_1 S_3 S_2 S_4 S_6 S_9 S_7 S_{10}

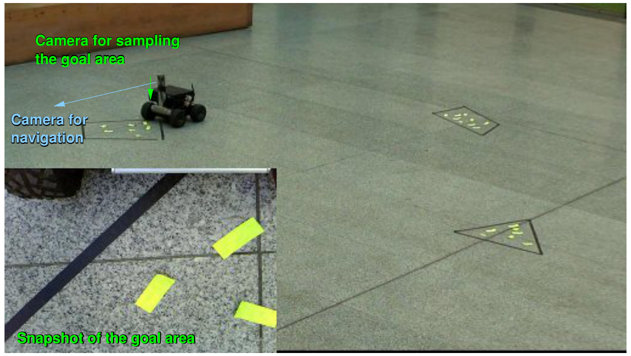
Multi-Goal Path Planning (MTP) Problem

- **MTP problem** is a **robotic variant of the TSP** with the edge costs as the length of the *shortest* path connecting the locations.
- Variants of the **robotic TSP** includes additional constraints arising from limitations of real robotic systems such as
  - obstacles, curvature-constraints, sensing range, location precision.
- For n locations, we need to compute up to n^2 shortest paths.
- Having a **roadmap** (graph) representing $\mathcal{C}_{free}$, the paths can be found in the graph (roadmap), from which the $G(V, E)$ for the TSP can be constructed.
  Visibility graph as a roadmap for a point robot provides a straight forward solution, but such a shortest path may not be necessarily feasible for more complex robots.
- We can determine the roadmap using randomized sampling-based motion planning techniques. See lecture 8.



Multi-goal Path Planning with Goal Regions

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



*Not only a **sequence** of goals visit has to be determined, but also an **appropriate location** at each region has 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.

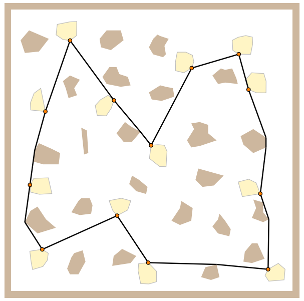
- The problem is NP-hard and APX-hard, it cannot be approximated to within factor $2-\epsilon$, where $\epsilon > 0$. *Safra and Schwartz (2006) – Computational Complexity*
- Approximate algorithms exist for particular problem variants such as disjoint unit disk neighborhoods.
- **TSPN provides a suitable problem formulation for planning various inspection and data collection missions.**
- It enables to exploit non-zero sensing range, and thus find shortest (more cost-efficient) **data collection plans**.

Traveling Salesman Problem with Neighborhoods (TSPN)

- Instead visiting a particular location $\boldsymbol{s} \in S$, $\boldsymbol{s} \in \mathbb{R}^2$ as in the TSP, we request to visit a set of regions $R = \{r_1, \ldots, r_n\}$, $r_i \subset \mathbb{R}^2$ to save travel cost.
- The TSP becomes the **TSP with Neighborhoods** (**TSPN**) where, in addition to the determination of the **sequence** Σ, we determine a suitable locations of visits $P = \{\boldsymbol{p}_1, \ldots, \boldsymbol{p}_n\}$, $\boldsymbol{p}_i \in r_i$.
- The problem is a combination of combinatorial optimization to determine Σ with **continuous optimization** to determine P.

$$\text{minimize}_{\Sigma,P} \quad L = \left(\sum_{i=1}^{n-1} c(\boldsymbol{p}_{\sigma_i}, \boldsymbol{p}_{\sigma_{i+1}})\right) + c(\boldsymbol{p}_{\sigma_n}, \boldsymbol{p}_{\sigma_1})$$

$$\begin{aligned}\text{subject to} \quad & R = \{r_1, \ldots, r_n\}, r_i \subset \mathbb{R}^2 \\ & P = \{\boldsymbol{p}_1, \ldots, \boldsymbol{p}_n\}, \boldsymbol{p}_i \in r_i \\ & \Sigma = (\sigma_1, \ldots, \sigma_n), 1 \le \sigma_i \le n, \\ & \sigma_i \ne \sigma_j \text{ for } i \ne j \\ & \text{Foreach } r_i \in R \text{ there is } \boldsymbol{p}_i \in r_i.\end{aligned}$$



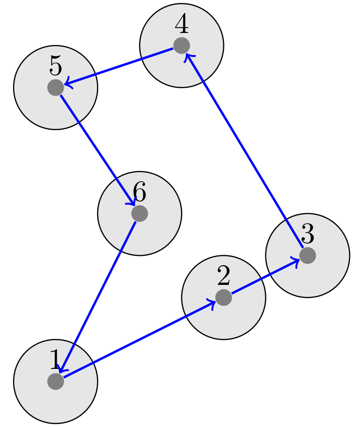
Approaches to the TSPN

- A direct solution of the TSPN – approximation algorithms and heuristics
  E.g., using evolutionary techniques or unsupervised learning
- Euclidean TSPN with, disk-shaped δ neighborhoods is called **Closed Enough TSP** (**CETSP**).
  - Simplified variant with regions as disks with radius δ – remote sensing with the δ communication range.
- **Decoupled approach**
  1. Determine sequence of visits Σ independently on the locations P, e.g., as a solution of the TSP using centroids of the (convex) regions R.
  2. For the sequence Σ determine the locations P to minimize the total tour length using
     - Touring polygon problem (TPP);
     - Sampling possible locations and use a forward search for finding the best locations;
     - Continuous optimization such as hill-climbing.

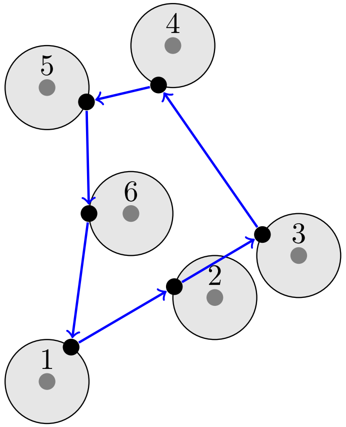
     E.g., Local Iterative Optimization (LIO), Váňa & Faigl (IROS 2015)
- **Sampling-based** approaches
  - For each region, sample possible locations of visits into a discrete set of locations for each region.
  - The problem can be then formulated as the **Generalized Traveling Salesman Problem** (**GTSP**).

Close Enough Traveling Salesman Problem (CETSP)

- **Close Enough TSP** (**CETSP**) is a variant of the TSPN with disk shaped δ-neighborhoods.



A solution of the TSP for the centers of the disks

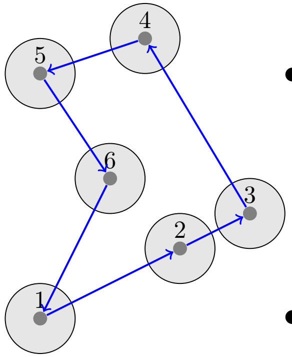


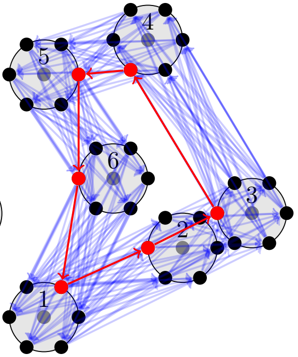
A solution of the CETSP

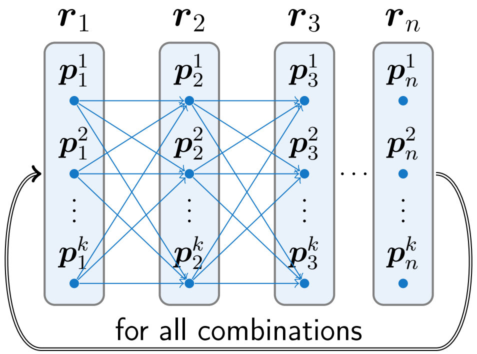
Decoupled Sampling-based Solution of the TSPN / CETSP

- **Decoupled** – Determine sequence of visits as a solution of the Euclidean TSP for the representatives of the regions R, e.g., using centroids.
- Sample each region (neighborhood) with k samples, e.g., $k = 6$.
- Construct graph and find the shortest tour in by graph search in $\mathcal{O}(nk^3)$ for n regions and nk^2 edges in the sequence.

For the closed path, we need to examine all k possible starting locations.







Iterative Refinement in the Multi-goal Planning Problem with Regions

- Let the sequence of n polygon regions be $R = (r_1, \ldots, r_n)$.

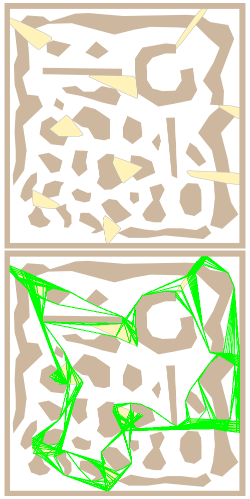
Li, F., Klette, R.: Approximate algorithms for touring a sequence of polygons. 2008

1. Sampling regions into a discrete set of points and determine all shortest paths between each sampled points in the sequence of visits to the regions.

E.g., using visibility graph

2. *Initialization:* Construct an initial touring polygons path using a sampled point of each region. Let the path be defined by $P = (\boldsymbol{p}_1, \boldsymbol{p}_2, \ldots, \boldsymbol{p}_n)$, where $\boldsymbol{p}_i \in r_i$ and $L(P)$ be the length of the shortest path induced by P.
3. *Refinement:* **For** $i = 1, 2, \ldots, n$:
   - Find $\boldsymbol{p}_i^* \in r_i$ minimizing the length of the path $d(\boldsymbol{p}_{i-1}, \boldsymbol{p}_i^*) + d(\boldsymbol{p}_i^*, \boldsymbol{p}_{i+1})$, where $d(\boldsymbol{p}_k, \boldsymbol{p}_l)$ is the path length from $\boldsymbol{p}_k$ to $\boldsymbol{p}_l$, $\boldsymbol{p}_0 = \boldsymbol{p}_n$, and $\boldsymbol{p}_{n+1} = \boldsymbol{p}_1$.
   - If the total length of the current path over point $\boldsymbol{p}_i^*$ is shorter than over $\boldsymbol{p}_i$, replace the point $\boldsymbol{p}_i$ by $\boldsymbol{p}_i^*$.
4. Compute the path length L_{new} using the refined points.
5. *Termination condition:* If $L_{new} - L < \epsilon$ Stop the refinement. Otherwise $L \leftarrow L_{new}$ and go to Step 3.
6. *Final path construction:* Use the last points and construct the path using the shortest paths among obstacles between two consecutive points.

On-line sampling during the iterations – Local Iterative Optimization (LIO), Váňa & Faigl (IROS 2015).



Part II

Part 2 – Unsupervised Learning for Multi-goal Planning

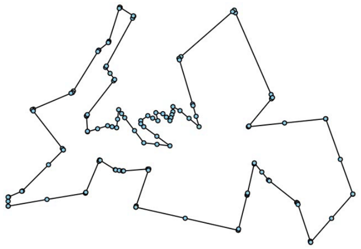
Unsupervised Learning based Solution of the TSP

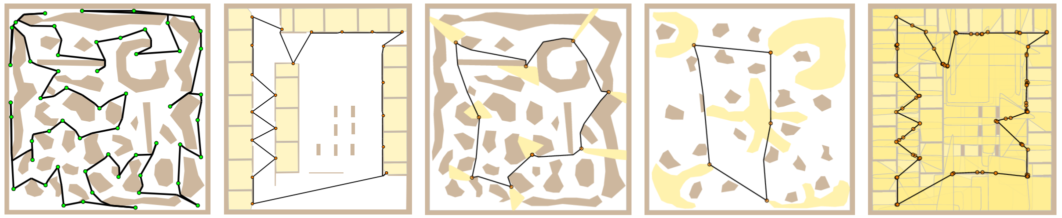
- Iterative learning procedure where neurons (nodes) adapt to the target locations.
- Based on self-organizing map by T. Kohonen.

Somhom, S., Modares, A., Enkawa, T. (1999)

- Deployed in robotic problems such as inspection and search-and-rescue planning. Faigl, J. et al. (2011)
  - Generalized to polygonal domain with (overlapping) regions.
- Evolved to **Growing Self-Organizing Array** (**GSOA**).

A general heuristic for various routing problems with neighborhoods; including routing problems with profit aka the orienteering problem.





Unsupervised Learning based Solution of the TSP

Kohonen's type of **unsupervised** two-layered neural network (**Self-Organizing Map**)

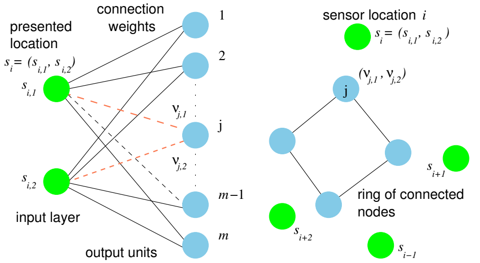
- Neurons' weights represent nodes $\mathcal{N} = \{\boldsymbol{\nu}_1, \ldots, \boldsymbol{\nu}_m\}$ in a plane (input space $\mathbb{R}^2$).
- Nodes are organized into a ring that evolved in the output space $\mathbb{R}^2$).
- Target locations $\boldsymbol{S} = \{\boldsymbol{s}_1, \ldots \boldsymbol{s}_n\}$ are presented to the network in a random order.
- Nodes compete to be winner according to their distance to the presented goal $\boldsymbol{s}$

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

- The winner and its neighbouring nodes are adapted (moved) towards the target according to the neighbouring function $\boldsymbol{\nu}' \leftarrow \mu f(\sigma, d)(\boldsymbol{\nu} - \boldsymbol{s})$

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

- Best matching unit $\boldsymbol{\nu}$ to the presented prototype s is determined according to the distance function $|\mathcal{D}(\boldsymbol{\nu}, \boldsymbol{s})|$.



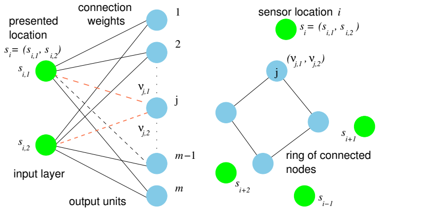
- For the Euclidean TSP, $\mathcal{D}$ is the Euclidean distance
- However, for problems with obstacles, the multi-goal path planning, $\mathcal{D}$ should correspond to the length of the shortest, collision-free path.

Fort, J.C. (1988), Angéniol, B. et al. (1988), Somhom, S. et al. (1997), and further improvements.

Unsupervised Learning based Solution of the TSP - Detail

- Target (sensor) locations $S = \{\boldsymbol{s}_1, \ldots, \boldsymbol{s}_n\}$, $\boldsymbol{s}_i \in \mathbb{R}^2$; Neurons $\mathcal{N} = (\boldsymbol{\nu}_1, \ldots, \boldsymbol{\nu}_m)$, $\boldsymbol{\nu}_i \in \mathbb{R}^2$, $m = 2.5n$.
- Learning gain σ; epoch counter i; gain decreasing rate $\alpha = 0.1$; learning rate $\mu = 0.6$.

```
1. $\mathcal{N} \leftarrow$ init ring of neurons as a small ring around some $\boldsymbol{s}_i \in S$, e.g., a circle with radius 0.5.
2. $i \leftarrow 0$; $\sigma \leftarrow 12.41n + 0.06$;
3. $I \leftarrow \emptyset$                                   //clear inhibited neurons
4. foreach $\boldsymbol{s} \in \Pi(S)$    (a permutation of $S$)
   4.1 $\boldsymbol{\nu}^* \leftarrow \mathrm{argmin}_{\boldsymbol{\nu} \in \mathcal{N} \setminus I} \|(\boldsymbol{\nu}, \boldsymbol{s})\|$
   4.2 foreach $\boldsymbol{\nu}$ in $d$ neighborhood of $\boldsymbol{\nu}^*$
           $\boldsymbol{\nu} \leftarrow \boldsymbol{\nu} + \mu f(\sigma, d)(\boldsymbol{s} - \boldsymbol{\nu})$
           $f(\sigma, d) = \begin{cases} e^{-\frac{d^2}{\sigma^2}} & \text{for } d < 0.2m, \\ 0 & \text{otherwise,} \end{cases}$
   4.3 $I \leftarrow I \bigcup \{\boldsymbol{\nu}^*\}$                    // inhibit the winner
5. $\sigma \leftarrow (1 - \alpha)\sigma$; $i \leftarrow i + 1$;
6. If (termination condition is not satisfied) Goto Step 3;
   Otherwise retrieve solution.
```

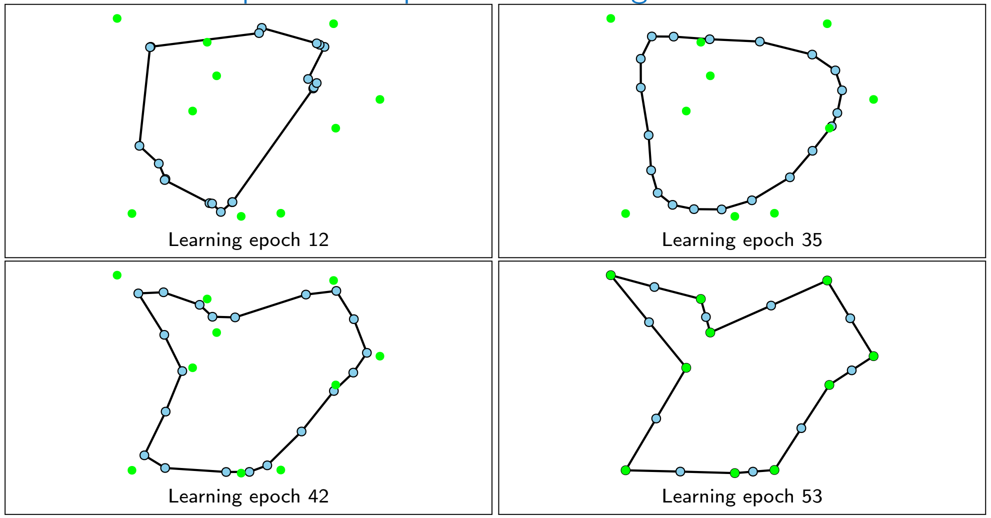


Termination condition can be

- Maximal number of learning epochs $i \leq i_{max}$, e.g., $i_{max} = 120$.
- Winner neurons are negligibly close to sensor locations, e.g., less than 0.001.

Somhom, S., Modares, A., Enkawa, T. (1999): Competition-based neural network for the multiple travelling salesmen problem with minmax objective. Computers & Operations Research.
Faigl, J. et al. (2011): An application of the self-organizing map in the non-Euclidean Traveling Salesman Problem. Neurocomputing.

Example of Unsupervised Learning for the TSP



Learning epoch 12

Learning epoch 35

Learning epoch 42

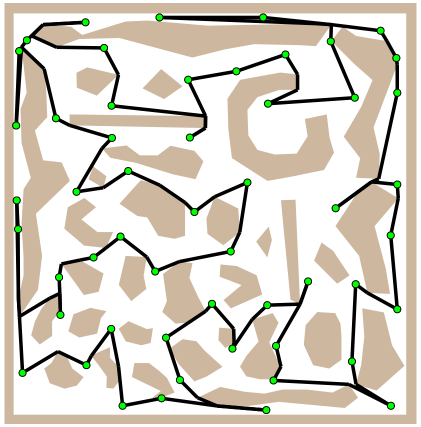
Learning epoch 53

Unsupervised Learning for the Multi-Goal Path Planning

- Unsupervised learning procedure for the Multi-goal Path Planning (**MTP**) problem a robotic variant of the Traveling Salesman Problem (TSP).

Algorithm 2: SOM-based MTP solver

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



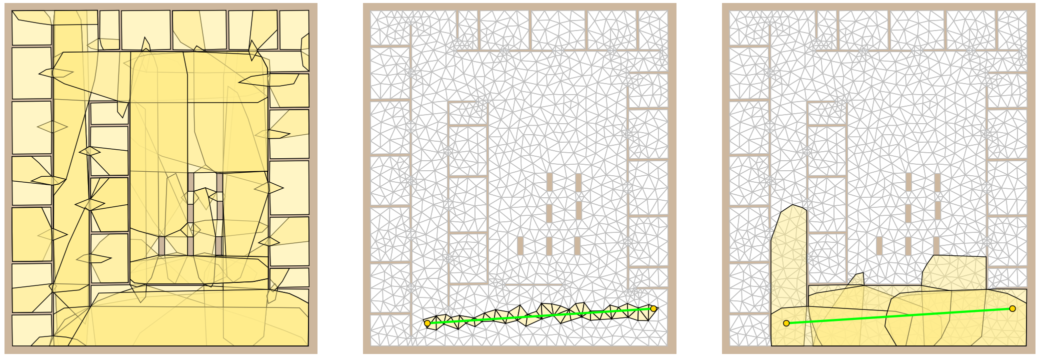
- For multi-goal path planning – the **selectWinner** and **adapt** procedures are based on the solution of the **path planning problem.**

Faigl, J., Kulich, M., Vonásek, V., Přeučil, L.: *An Application of Self-Organizing Map in the non-Euclidean Traveling Salesman Problem*, Neurocomputing, 74(5):671-679, 2011.

SOM for the TSP in the Watchman Route Problem – *Inspection Planning*

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.

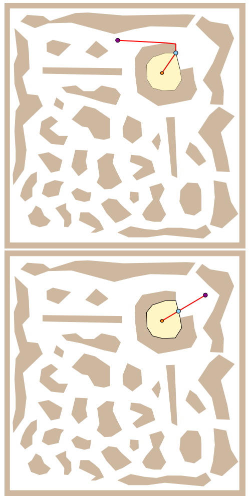


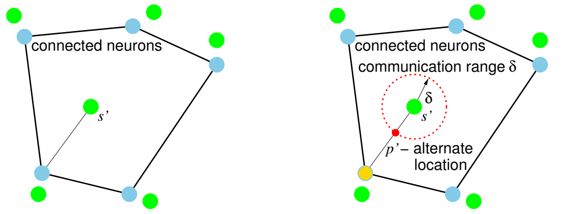
Faigl, J.: *Approximate solution of the multiple watchman routes problem with restricted visibility range*, **IEEE Transactions on Neural Networks**, 21(10):1668-1679, 2010.

Unsupervised Learning for the TSPN

- A suitable location of the region can be sampled during the winner selection.
- We can use the centroid of the region for the shortest path computation from ν to the region r presented to the network.
- Then, an intersection point of the path with the region can be used as an alternate location.

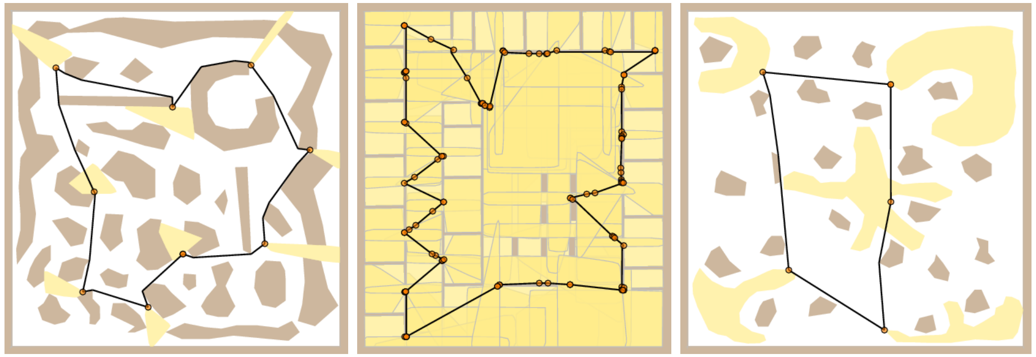
  Faigl, J. et al. (2013): Visiting convex regions in a polygonal map. Robotics and Autonomous Systems.
- For the Euclidean TSPN with disk-shaped δ neighborhoods, we can compute the alternate location directly from the Euclidean distance.





SOM for the Traveling Salesman Problem with Neighborhoods (TSPN)

- Unsupervised learning of the SOM for the TSP allows to generalize the adaptation procedure to the TSPN.
- It also provides solutions for non-convex regions, overlapping regions, and coverage problems.



Polygonal Goals n=9, T= 0.32 s

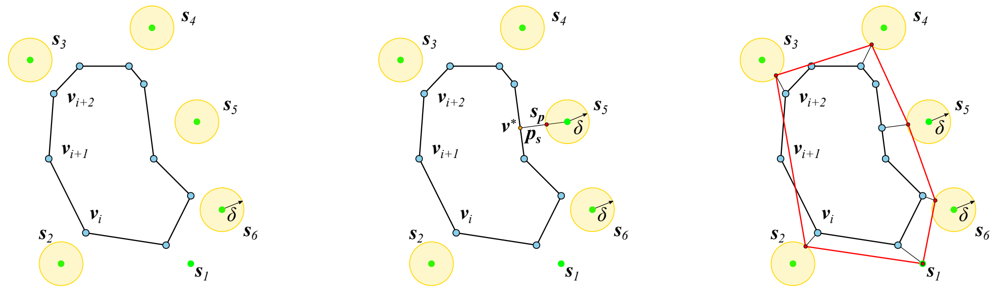
Convex Cover Set n=106, T=5.1 s

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

Faigl, J., Vonásek, V., Přeučil, L.: *Visiting Convex Regions in a Polygonal Map*, **Robotics and Autonomous Systems**, 61(10):1070–1083, 2013.

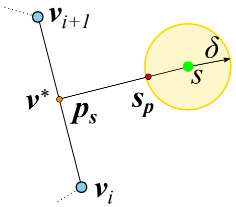
Growing Self-Organizing Array (GSOA)

- **Growing Self-Organizing Array** (**GSOA**) is generalization of the unsupervised learning to routing problems motivated by data collection planning, i.e., routing with neighborhoods such as the **Close Enough TSP**.
- The GSOA is an array of nodes $\mathcal{N} = \{\nu_1, \ldots, \nu_M\}$ that evolves in the problem space using unsupervised learning.
- The array adapts to each $s \in S$ (in a random order) and for each s a **new winner node** $\boldsymbol{\nu}^*$ is determined together with the corresponding $\boldsymbol{s}_p$, such that $\|(\boldsymbol{s}_p, \boldsymbol{s})\| \leq \delta(s)$. It adaptively adjusts the number of nodes.
- The winner and its neighborhoods are adapted (moved) towards $\boldsymbol{s}_p$.
- After the adaptation to all $\boldsymbol{s} \in S$, each $\boldsymbol{s}$ has its $\boldsymbol{\nu}$ and $\boldsymbol{s}_p$, and the array defines the sequence Σ and the requested waypoints P.

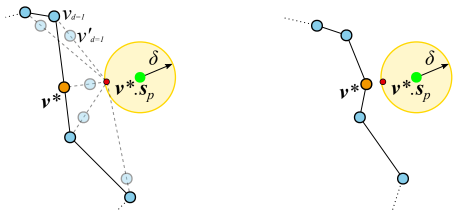


GSOA – Winner Selection and Its Adaptation

- Selecting winner node $\boldsymbol{\nu}^*$ for $\boldsymbol{s}$ and its waypoint $\boldsymbol{s}_p$



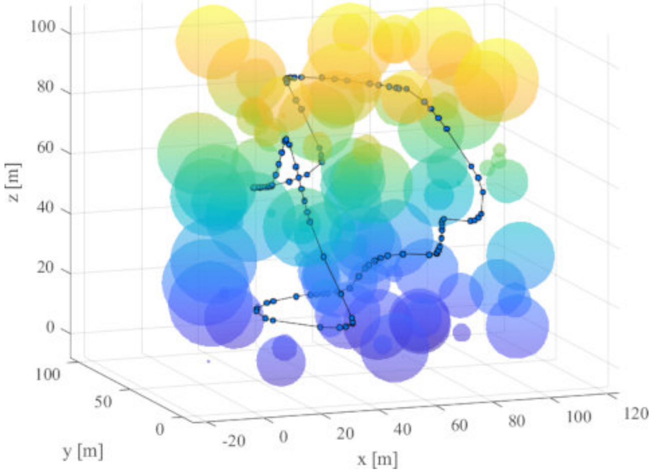
- Winner adaptation



- For each $s \in S$, we create new node ν^*, and therefore, all not winning nodes are removed after processing all locations in S (one learning epoch) to balance the number of nodes in the GSOA.
- After **each learning epoch**, the **GSOA encodes a feasible solution of the CETSP**.
- The power of adaptation is decreasing using a cooling schedule after each learning epoch.
- The GSOA converges to a stable solution in tens of epochs. Number of epochs can be set.

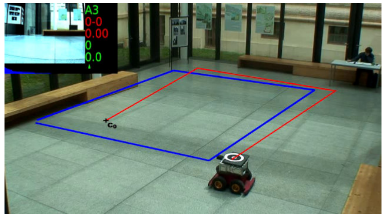
Faigl, J. (2018): GSOA: Growing Self-Organizing Array - Unsupervised learning for the Close-Enough Traveling Salesman Problem and other routing problems. Neurocomputing 312: 120-134 (2018).

GSOA Evolution in solving the 3D CETSP



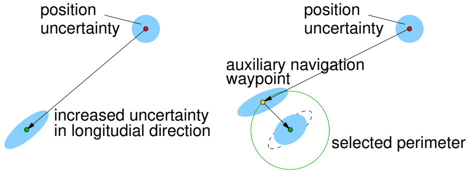
Example – TSPN for Planning with Localization Uncertainty

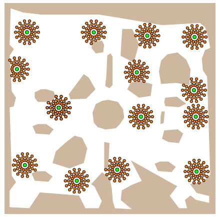
- Teach-and-repeat autonomous navigation using vision-based bearing corrections that are more precise than estimation of the traveled distance based on odometry measurements.

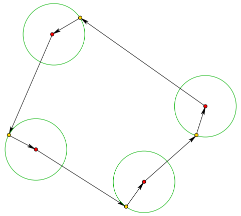


Krajník, T., Faigl, J., Vonásek, V., Košnar, K., Kulich, M., and Přeučil, L.: *Simple yet stable bearing-only navigation*, **Journal of Field Robotics**, 27(5):511-533, 2010.

- The localization uncertainty can be decreased by visiting auxiliary navigation waypoints prior the target locations.
- It can be formulated as a variant of the TSPN with auxiliary navigation waypoints.


position uncertainty
increased uncertainty in longitudinal direction
position uncertainty
auxiliary navigation waypoint
selected perimeter



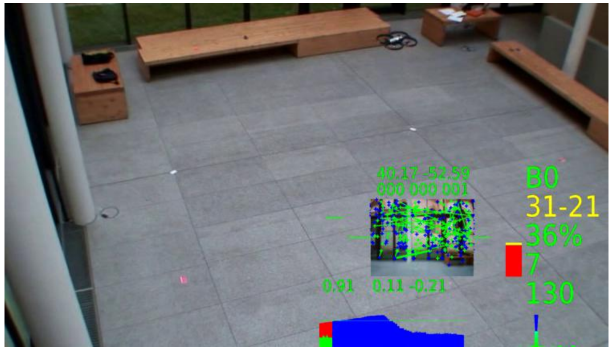


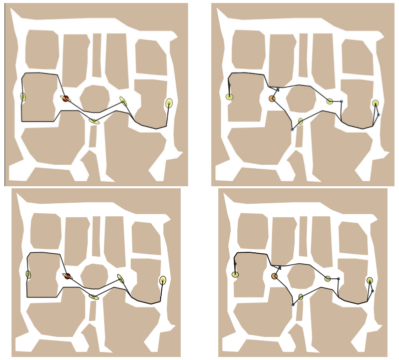
- The adaptation procedure is modified to select the auxiliary navigation waypoint to decrease the expected localization error at the target locations.

Faigl, J., Krajník, T., Vonásek, V., and Přeučil, L.: *On localization uncertainty in an autonomous inspection*, **IEEE International Conference on Robotics and Automation** (ICRA), 2012, pp. 1119-1124.

Example – Results on the TSPN for Planning with Localization Uncertainty

- Deployment in indoor and outdoor environment with ground mobile robots and aerial vehicle in indoor environment.
- For the MMP5 robot, the error decreased from 16.6 cm → 12.8 cm in indoor.
- For the P3AT robot, the real overall error at the goals decreased from 0.89 m → 0.58 m (about 35 %) in outdoor.
- For a small aerial vehicle, the Parrot AR.Drone, the success of the locations' visits improved from 83 % to 95 %.





TSP: L=184 m, E_{avg}=0.57 m TSPN: L=202 m, E_{avg}=0.35 m

Summary of the Lecture

Topics Discussed

- Robotic information gathering in inspection missions
- Inspection planning and multi-goal path planning - coverage planning
- **Multi-goal path planning** (**MTP)**
  - Robotic Traveling Salesman Problem (TSP)
  - Traveling Salesman Problem with Neighborhoods (TSPN) and Close Enough Traveling Salesman Problem (CETSP)
    - **Decoupled** and **Sampling-based** approaches
    - TSP can be solved by efficient heuristics such as **LKH**
    - Optimal, approximation, and heuristics solutions
    - **Generalized TSP** (**GTSP**)
- Next: Data collection planning