Curvature-Constrained Multi-Goal Trajectory Planning with Unmanned Aerial Vehicles

Jan Faigl

Associate Professor – Department of Computer Science
Head of Computational Robotics Laboratory
Artificial Intelligence Center (AIC)
Center for Robotics and Autonomous Systems (CRAS)
Faculty of Electrical Engineering
Czech Technical University in Prague

July 30, 2019
Computational Robotics Laboratory
Robotic Information Gathering

- **Robotic information gathering** – build phenomena model using measurements collected by mobile robots
- **Quality guarantee** of the found solution (tight lower bound) and **computationally efficient** solutions

**Computational Robotics Laboratory (CRL)**
https://comrob.fel.cvut.cz
- **Curvature-constrained multi-goal trajectory planning** with Dubins vehicles
  - Multi-goal planning problem formulation – Traveling Salesman Problem (**TSP**)
  - Dubins TSP and Dubins TSP with Neighborhoods – solutions and lower bounds
    - Decoupled and sampling-based approaches
    - Multi-vehicle multi-goal planning $m$-DTSPN
    - Cluster-First, Route-Second
    - Simple construction heuristic
    - Combined heuristics (VNS and GSOA)
- Examples of further generalization of the multi-goal planning with curvature-constrained trajectories
  - Extensions to 3D planning and trajectory parametrization using Bézier curves
  - Multi-goal planning with profits (limited travel budgets)
  - Multi-goal motion planning planning – Physical Orienteering Problem (**POP**)
- Having a set of locations to be visited, determine the cost-efficient path to visit them, and return to a starting location.
- The multi-goal planning problem is also called robotic task sequencing problem.

Traveling Salesman Problem (TSP)

- Let $P$ be a set of $n$ locations $P = \{p_1, \ldots, p_n\}$, $p_i \in \mathbb{R}^2$ and $c(p_i, p_j)$ is a cost of travel from $p_i$ to $p_j$.

- **Traveling Salesman Problem (TSP)** is a problem to determine a closed tour visiting each $p \in P$ such that the total tour length is minimal, i.e., determine a sequence of visits $\Sigma = (\sigma_1, \ldots, \sigma_n)$ such that

$$
\text{minimize } \Sigma \quad \sum_{i=1}^{n-1} c(p_{\sigma_i}, p_{\sigma_{i+1}}) + c(p_{\sigma_n}, p_{\sigma_1})
$$

subject to $\Sigma = (\sigma_1, \ldots, \sigma_n)$, $1 \leq \sigma_i \leq n$, $\sigma_i \neq \sigma_j$ for $i \neq j$.

The TSP can be considered on a graph $G(V, E)$ where the set of vertices $V$ represents sensor locations $P$ and $E$ are edges connecting the nodes with the cost $c(p_i, p_j)$.

$c(p_i, p_j)$ can be Euclidean distance; otherwise, we have to address the path/motion planning.

It is known, the TSP is NP-hard (its decision variant) and several algorithms exist.

Existing solvers to the TSP

- **Exact solutions**
  - Branch and Bound, Integer Linear Programming (ILP)
    
    E.g., Concorde solver – [http://www.tsp.gatech.edu/concorde.html](http://www.tsp.gatech.edu/concorde.html)
  
  - Approximation algorithms
    - Minimum Spanning Tree (MST) heuristic with $L \leq 2L_{opt}$
    - Christofides’s algorithm with $L \leq \frac{3}{2}L_{opt}$

- **Heuristic algorithms**
  - Constructive heuristic – Nearest Neighborhood (NN) algorithm
  - 2-Opt – local search algorithm proposed by Croes 1958
  - **Lin-Kernighan** (LK) heuristic (e.g., LKH)
    
    E.g., Helsgaun’s implementation of the LK heuristic – [http://www.akira.ruc.dk/~keld/research/LKH](http://www.akira.ruc.dk/~keld/research/LKH)

- **Soft-computing techniques, e.g.,**
  - **Variable Neighborhood Search** (VNS), Greedy Randomized Adaptive Search Procedures (GRASP), evolutionary approaches (genetic, memetic, etc.), ant colony optimization (aco), particle swarm optimization (pso), unsupervised learning – Self-Organizing Maps (SOM) and Growing Self-Organizing Array (GSOA)
Multi-Goal Planning with Dubins Vehicle
Dubins Traveling Salesman Problem (with Neighborhoods)
Provide curvature-constrained path to collect the most valuable measurements with shortest possible path/time or under limited travel budget.

Formulated as routing problems with Dubins vehicle
- Dubins Traveling Salesman Problem (with Neighborhoods) (DTSP(N))
- Dubins Orienteering Problem (with Neighborhoods) (DOP(N)) – routing problems with profits
Dubins vehicle can be used for planning curvature-constrained paths: constant forward velocity $v$ and limited minimal turning radius $\rho$.

Dubins maneuvers are straight segments and parts of circle.

State $q = (x, y, \theta)$ of Dubins vehicle consists of position $(x, y) \in \mathbb{R}^2$ and heading $\theta \in S^2$.

\[
\begin{bmatrix}
\dot{x} \\
\dot{y} \\
\dot{\theta}
\end{bmatrix} = v \begin{bmatrix}
\cos \theta \\
\sin \theta \\
\frac{u}{\rho}
\end{bmatrix}
\]

Optimal path connecting two states $q_1, q_2 \in SE(2)$ corresponds to the control input $u \in \{-1, 0, 1\}$.
Difficult of Dubins Vehicle in the Solution of the TSP

- For the minimal turning radius $\rho$, the optimal path connecting $q_1 \in SE(2)$ and $q_2 \in SE(2)$ can be found analytically.

- L. E. Dubins (1957) – American Journal of Mathematics

- Two types of optimal Dubins maneuvers: CSC and CCC
  - **CSC** type: LSL, LSR, RSL, RSR;
  - **CCC** type: LRL, RLR.

- Length of the optimal maneuver $L$
  - has a closed-form solution;
  - Can be computed in less than $0.5 \mu s$
  - is piecewise-continuous function;
  - (continuous for $\| (p_1, p_2) \| > 4 \rho$).
Dubins Traveling Salesman Problem (DTSP)

- Determine (closed) shortest Dubins path visiting each \( p_i \in \mathbb{R}^2 \) of the given set of \( n \) locations \( P = \{p_1, \ldots, p_n\} \).

1. Permutation \( \Sigma = (\sigma_1, \ldots, \sigma_n) \) of visits (sequencing).

2. Headings \( \Theta = \{\theta_{\sigma_1}, \theta_{\sigma_2}, \ldots, \theta_{\sigma_n}\} \), \( \theta_i \in [0, 2\pi) \), for \( p_{\sigma_i} \in P \).

**DTSP** is an optimization problem over all possible sequences \( \Sigma \) and headings \( \Theta \) at the states \( \{q_{\sigma_1}, q_{\sigma_2}, \ldots, q_{\sigma_n}\} \) such that

\[
q_{\sigma_i} = (p_{\sigma_i}, \theta_{\sigma_i}), \quad p_{\sigma_i} \in P
\]

minimize \( \Sigma, \Theta \)

\[
\sum_{i=1}^{n-1} L(q_{\sigma_i}, q_{\sigma_{i+1}}) + L(q_{\sigma_n}, q_{\sigma_1})
\]

subject to \( q_i = (p_i, \theta_i) \) \( i = 1, \ldots, n \),

where \( L(q_{\sigma_i}, q_{\sigma_j}) \) is the length of Dubins path between \( q_{\sigma_i} \) and \( q_{\sigma_j} \).
Existing Approaches to the DTSP(N)

Heuristics, Resolution Complete, and Sampling-based

- **Heuristic (decoupled & evolutionary) approaches**
  - Savla et al., 2005
  - Ma and Castanon, 2006
  - Macharet et al., 2011
  - Macharet et al., 2012
  - Ny et al., 2012
  - Yu and Hang, 2012
  - Macharet et al., 2013
  - Zhait et al., 2014
  - Macharet and Campost, 2014
  - Váňa and Faigl, 2015
  - Isaiah and Shima, 2015
  - ...

- **Sampling-based approaches**
  - Obermeyer, 2009
  - Oberlin et al., 2010
  - Macharet et al., 2016

- **Convex optimization**
  - (Only if the locations are far enough)
  - Goac et al., 2013

- **Lower bound for the DTSP**
  - Using Dubins Interval Problem (DIP)
  - Manyam et al., 2016
  - Using DIP to inform sampling in DTSP
  - Váňa and Faigl, 2017

- **Lower bound for the DTSPN**
  - Using Generalized DIP (GDIP)
  - Váňa and Faigl, 2018
**DTSP – Decoupled and Sampling-based Solution**

- **Decoupled** approach
  1. Find sequence of visits, e.g., as a solution of the **Euclidean TSP**.
  2. Determine the heading angles for the sequence of points – **Dubins multi-goal path** solved as the **Dubins Touring Problem (DTP)**.

- **Sampling-based** approach – discretize domains of the continuous variables into a finite set of heading values.
  - Solve the problem as a variant of the **Generalized TSP**.
  - Determine the optimal headings for the given sequence.

The first layer is duplicated layer to support the forward search method.

---

Jan Faigl - Curvature-Constrained Multi-Goal Planning

2019 IEEE RAS Summer School on Multi-Robot Systems
1. Sequencing part can be determined as a solution of the Euclidean TSP, e.g., using Lin-Kernighan or 2-opt heuristics.

2. Finding the optimal heading angles is continuous optimization problem that is called **Dubins Touring Problem (DTP)**.

- For a sequence of the $n$ waypoint locations $P = (p_1, \ldots, p_n)$, $p_i \in \mathbb{R}^2$, the **Dubins Touring Problem (DTP)** stands to determine the optimal headings $H = \{\theta_1, \ldots, \theta_n\}$ at the waypoints $q_i$ such that

$$
\begin{align*}
\text{minimize}_H & \quad \mathcal{L}(H, P) = \sum_{i=1}^{n-1} \mathcal{L}(q_i, q_{i+1}) + \mathcal{L}(q_n, q_1) \\
\text{subject to} & \quad q_i = (p_i, \theta_i), \quad \theta_i \in [0, 2\pi), \quad p_i \in P,
\end{align*}
$$

where $\mathcal{L}(q_i, q_j)$ is the length of the Dubins maneuver connecting $q_i$ with $q_j$.

- DTP can be solved by simple heuristic such as the Alternating Algorithm (AA), Local Iterative Optimization (LIO), or by sampling based methods.
Alternating Algorithm (AA)

1. Establish headings for even edges using straight line segments.
2. Determine optimal maneuvers for odd edges using the analytical form for Dubins maneuvers.

Local Iterative Optimization (LIO)

- Iteratively perform local optimization for each waypoint location.
- At each waypoint location $p_i$, the heading can be $\theta_i \in [0, 2\pi)$.

Algorithm 1: Local Iterative Optimization (LIO) for the DTSP

Data: Input sequence of the goal locations $P = (p_{\sigma_1}, \ldots, p_{\sigma_n})$, for the permutation $\Sigma$

Result: Waypoints $(q_{\sigma_1}, \ldots, q_n), q_i = (p_i, \theta_i)$

1. initialization() // random assignment of $q_i = (p_i, \theta_i)$, $\theta_i \in [0, 2\pi)$
2. while global solution is improving do
3. for every $p_i \in P$ do
4. $\theta_i := \text{optimizeHeadingLocally}(\theta_i)$;
5. $q_i := \text{checkLocalMinima}(\theta_i)$;
6. end
7. end
1. Create a set of $k$ discrete heading angles for each waypoint location, e.g., using uniform sampling, $h_i = \{\theta^1_i, \ldots, \theta^k_i\}$.

2. For a given sequence of waypoint locations, create a search graph, where each layer represents possible headings of the particular waypoint locations $p_i$.

The optimal headings can be determined as the shortest path in the graph connecting the layers in $O(nk^3)$. Optimal solution for the given discretization and given sequence of waypoint locations.
We can sample possible heading angles for each waypoint location and transform the DTSP into purely combinatorial the Generalized Traveling Salesman Problem (GTSP).

GTSP is also known as the Set TSP or Covering Salesman Problem, etc.

For a set of $n$ sets $S = \{S_1, \ldots, S_n\}$, each with particular set of sampled waypoints (nodes) $S_i = \{s^i_1, \ldots, s^i_{n_i}\}$; the problem is to determine the shortest tour visiting each set $S_i$, i.e., determining the order $\Sigma$ of visits to $S$ and a particular locations $s^i \in S_i$ for each $S_i \in S$

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

subject to

$\Sigma = (\sigma_1, \ldots, \sigma_n), 1 \leq \sigma_i \leq n, \sigma_i \neq \sigma_j$ for $i \neq j$

$s^{\sigma_i} \in S_{\sigma_i}, S_{\sigma_i} = \{s^{\sigma_i}_1, \ldots, s^{\sigma_i}_{n_{\sigma_i}}\}, S_{\sigma_i} \in S$. 

GTSP can be solved directly using ILP, heuristic algorithms or transformed to the Asymmetric TSP

- **GLKH** – [http://akira.ruc.dk/~keld/research/GLKH/](http://akira.ruc.dk/~keld/research/GLKH/)

- **GLNS** – [https://ece.uwaterloo.ca/~sl2smith/GLNS](https://ece.uwaterloo.ca/~sl2smith/GLNS) (in Julia)
The Generalized TSP can be transformed into the Asymmetric TSP that can be then solved, e.g., by LKH or exactly using Concorde with the further transformation of the problem to the TSP.

A transformation of the GTSP to the ATSP was proposed by Noon and Bean in 1993, and it is called Noon-Bean Transformation.

Noon-Bean transformation to transfer GTSP to ATSP

- Modify weight of the edges (arcs) such that the optimal ATSP tour visits all vertices of the same cluster before moving to the next cluster.
  - Adding a large constant $M$ to the weights of arcs connecting the clusters, e.g., a sum of the $n$ heaviest edges.
  - Ensure visiting all vertices of the cluster in the prescribed order, i.e., creating zero-length cycles within each cluster.

- The transformed ATSP can be further transformed to the TSP.
  - For each vertex of the ATSP create 3 vertices in the TSP, i.e., it increases the size of the problem three times.

1. Create a zero-length cycle in each set and set all other arcs to $\infty$ (or 2M).

   To ensure all vertices of the cluster are visited before leaving the cluster.

2. For each edge $(q_i^m, q_j^n)$ create an edge $(q_i^m, q_j^{n+1})$ with a value increased by sufficiently large $M$.

   To ensure visit of all vertices in a cluster before the next cluster.
1. Create a zero-length cycle in each set and set all other arcs to $\infty$ (or 2M).
   
   To ensure all vertices of the cluster are visited before leaving the cluster.

2. For each edge $(q^m_i, q^n_j)$ create an edge $(q^m_i, q^{n+1}_j)$ with a value increased by sufficiently large $M$.

   To ensure visit of all vertices in a cluster before the next cluster.
1. Create a zero-length cycle in each set and set all other arcs to $\infty$ (or 2M).
   To ensure all vertices of the cluster are visited before leaving the cluster.

2. For each edge $(q_i^m, q_j^n)$ create an edge $(q_i^m, q_j^{n+1})$ with a value increased by sufficiently large $M$.
   To ensure visit of all vertices in a cluster before the next cluster.
Sampling-based Solution of the DTSP
Example – Noon-Bean transformation (GATSP to ATSP)

1. Create a zero-length cycle in each set and set all other arcs to $\infty$ (or 2M).
   To ensure all vertices of the cluster are visited before leaving the cluster.

2. For each edge $(q^m_i, q^n_j)$ create an edge $(q^m_i, q^n_{j+1})$ with a value increased by sufficiently large $M$.
   To ensure visit of all vertices in a cluster before the next cluster.
Sampling-based Solution of the DTSP
Example – Noon-Bean transformation (GATSP to ATSP)

1. Create a zero-length cycle in each set and set all other arcs to $\infty$ (or 2M).
   To ensure all vertices of the cluster are visited before leaving the cluster.

2. For each edge $(q^m_i, q^n_j)$ create an edge $(q^m_i, q^{n+1}_j)$ with a value increased by sufficiently large $M$.
   To ensure visit of all vertices in a cluster before the next cluster.
1. Create a zero-length cycle in each set and set all other arcs to $\infty$ (or 2M). To ensure all vertices of the cluster are visited before leaving the cluster.

2. For each edge $(q_i^m, q_j^n)$ create an edge $(q_i^m, q_j^{n+1})$ with a value increased by sufficiently large $M$. To ensure visit of all vertices in a cluster before the next cluster.
1. Create a zero-length cycle in each set and set all other arcs to $\infty$ (or 2M).

To ensure all vertices of the cluster are visited before leaving the cluster.

2. For each edge $(q^m_i, q^n_j)$ create an edge $(q^m_i, q^n_{j+1})$ with a value increased by sufficiently large $M$.

To ensure visit of all vertices in a cluster before the next cluster.
Sampling-based Solution of the DTSP
Example – Noon-Bean transformation (GATSP to ATSP)

1. Create a zero-length cycle in each set and set all other arcs to $\infty$ (or 2M).
   
   To ensure all vertices of the cluster are visited before leaving the cluster.

2. For each edge $(q^m_i, q^n_j)$ create an edge $(q^m_i, q^n_{j+1})$ with a value increased by sufficiently large $M$.
   
   To ensure visit of all vertices in a cluster before the next cluster.
Sampling-based Solution of the DTSP
Example – Noon-Bean transformation (GATSP to ATSP)

1. Create a zero-length cycle in each set and set all other arcs to $\infty$ (or 2M).
   To ensure all vertices of the cluster are visited before leaving the cluster.

2. For each edge $(q^m_i, q^n_j)$ create an edge $(q^m_i, q^{n+1}_j)$ with a value increased by sufficiently large $M$.
   To ensure visit of all vertices in a cluster before the next cluster.
1. Create a zero-length cycle in each set.

2. For each edge \((q^m_i, q^n_j)\) create an edge \((q^m_i, q^{n+1}_j)\) with a value increased by sufficiently large \(M\).

Original GATSP

<table>
<thead>
<tr>
<th></th>
<th>(q^1_1)</th>
<th>(q^2_1)</th>
<th>(q^3_1)</th>
<th>(q^1_2)</th>
<th>(q^2_2)</th>
<th>(q^3_2)</th>
</tr>
</thead>
<tbody>
<tr>
<td>(q^1_1)</td>
<td>(\infty)</td>
<td>(\infty)</td>
<td>(\infty)</td>
<td>(7)</td>
<td>(_)</td>
<td>(_)</td>
</tr>
<tr>
<td>(q^1_2)</td>
<td>(\infty)</td>
<td>(\infty)</td>
<td>(\infty)</td>
<td>(_)</td>
<td>(1)</td>
<td>(_)</td>
</tr>
<tr>
<td>(q^3_2)</td>
<td>(\infty)</td>
<td>(\infty)</td>
<td>(\infty)</td>
<td>(4)</td>
<td>(_)</td>
<td>(_)</td>
</tr>
<tr>
<td>(q^2_2)</td>
<td>(_)</td>
<td>(_)</td>
<td>(_)</td>
<td>(\infty)</td>
<td>(\infty)</td>
<td>(5)</td>
</tr>
<tr>
<td>(q^2_3)</td>
<td>(_)</td>
<td>(_)</td>
<td>(_)</td>
<td>(\infty)</td>
<td>(\infty)</td>
<td>(2)</td>
</tr>
<tr>
<td>(q^3_3)</td>
<td>(6)</td>
<td>(3)</td>
<td>(8)</td>
<td>(_)</td>
<td>(_)</td>
<td>(\infty)</td>
</tr>
</tbody>
</table>

Transformed ATSP

<table>
<thead>
<tr>
<th></th>
<th>(q^1_1)</th>
<th>(q^2_1)</th>
<th>(q^3_1)</th>
<th>(q^1_2)</th>
<th>(q^2_2)</th>
<th>(q^3_2)</th>
</tr>
</thead>
<tbody>
<tr>
<td>(q^1_1)</td>
<td>(\infty)</td>
<td>(0)</td>
<td>(\infty)</td>
<td>(_)</td>
<td>(7+M)</td>
<td>(_)</td>
</tr>
<tr>
<td>(q^1_2)</td>
<td>(\infty)</td>
<td>(\infty)</td>
<td>(0)</td>
<td>(1+M)</td>
<td>(_)</td>
<td>(_)</td>
</tr>
<tr>
<td>(q^3_3)</td>
<td>(0)</td>
<td>(\infty)</td>
<td>(\infty)</td>
<td>(_)</td>
<td>(4+M)</td>
<td>(_)</td>
</tr>
<tr>
<td>(q^2_2)</td>
<td>(_)</td>
<td>(_)</td>
<td>(_)</td>
<td>(\infty)</td>
<td>(0)</td>
<td>(5+M)</td>
</tr>
<tr>
<td>(q^2_3)</td>
<td>(_)</td>
<td>(_)</td>
<td>(_)</td>
<td>(_)</td>
<td>(\infty)</td>
<td>(2+M)</td>
</tr>
<tr>
<td>(q^3_3)</td>
<td>(8+M)</td>
<td>(6+M)</td>
<td>(3+M)</td>
<td>(_)</td>
<td>(_)</td>
<td>(0)</td>
</tr>
</tbody>
</table>
It transforms the GATSP into the ATSP which can be further solved by existing solvers, e.g., the Lin-Kernighan heuristic algorithm (LKH).

The ATSP can be further transformed into the TSP and solved optimally, e.g., by the Concorde solver.

It runs in $O(k^2n^2)$ time and uses $O(k^2n^2)$ memory, where $n$ is the number of sets (regions) each with up to $k$ samples.

The transformed ATSP problem contains $kn$ vertices.

### DTSP – Heuristic Solutions

Decoupled and Sampling-based Approaches

- **Heuristic solutions are fast and might provide relatively good (satisfiable) solutions.**
- **The optimal solution of the discretize problem (e.g., as the GTSP) does not guarantee optimal solution of the original continuous problem.**
- **We do not know how far the found solution is from the optimum.**
Heuristic solutions are fast and might provide relatively good (satisfiable) solutions.

- The optimal solution of the discretize problem (e.g., as the GTSP) does not guarantee optimal solution of the original continuous problem.

- We do not know how far the found solution is from the optimum.
Solution of the DTSP and Theoretical Guarantee – Lower Bound using solution of the Dubins Interval Problem (DIP)

- **Dubins Interval Problem (DIP)** – Determine the shortest Dubins maneuver connecting \( p_i \in \mathbb{R}^2 \) and \( p_j \in \mathbb{R}^2 \) for the given angle intervals \( \theta_i \in [\theta_{i \text{min}}, \theta_{i \text{max}}] \) and \( \theta_j \in [\theta_{j \text{min}}, \theta_{j \text{max}}] \).

- DIP has closed-form solution.

- For the intervals \( \Theta_i = \Theta_j = [0, 2\pi) \), the solution is the length of the straight line segment.

- It provides lower-bound of the length of the shortest Dubins maneuver connecting \( p_i \) and \( p_j \).

- DIP enables tight lower bound estimation.

- **Iteratively-Refined Informed Sampling (IRIS).**

Manyam, Rathinam, and Casbeer, 2016

Decoupled Solution of the DTSP
Iteratively-Refined Informed Sampling (IRIS) based on Lower Bound Estimation using DIP

- Refinement iteration 1, the angular resolution $2\pi/4$.

Uniform sampling

Informed sampling (IRIS)

$\epsilon = 2\pi/4$, $N = 28$, $T_{CPU} = 8 \text{ ms}$
$L = 27.9$, $L_U = 13.2$

$\epsilon = 2\pi/4$, $N = 21$, $T_{CPU} = 8 \text{ ms}$
$L = 29.9$, $L_U = 13.2$
Decoupled Solution of the DTSP
Iteratively-Refined Informed Sampling (IRIS) based on Lower Bound Esti-

- Refinement iteration 2, the angular resolution $2\pi/8$.

Uniform sampling

Informed sampling (IRIS)

$\epsilon = 2\pi/8, \ N = 56, \ T_{CPU} = 16 \text{ ms}$
$L = 20.8, \ L_U = 13.2$

$\epsilon = 2\pi/8, \ N = 28, \ T_{CPU} = 20 \text{ ms}$
$L = 21.0, \ L_U = 13.2$
Decoupled Solution of the DTSP
Iteratively-Refined Informed Sampling (IRIS) based on Lower Bound Esti-

- Refinement iteration 3, the angular resolution $2\pi/16$.

**Uniform sampling**

$\epsilon = 2\pi/16$, $N = 112$, $T_{CPU} = 40$ ms

$L = 20.3$, $L_U = 13.5$

**Informed sampling (IRIS)**

$\epsilon = 2\pi/16$, $N = 35$, $T_{CPU} = 24$ ms

$L = 20.1$, $L_U = 13.5$
Decoupled Solution of the DTSP
Iteratively-Refined Informed Sampling (IRIS) based on Lower Bound Esti-

- Refinement iteration 4, the angular resolution $2\pi/32$.

**Uniform sampling**

- $\epsilon = 2\pi/32$, $N = 224$, $T_{CPU} = 140$ ms
- $L = 19.8$, $L_U = 13.8$

**Informed sampling (IRIS)**

- $\epsilon = 2\pi/32$, $N = 44$, $T_{CPU} = 32$ ms
- $L = 19.9$, $L_U = 13.8$
Decoupled Solution of the DTSP
Iteratively-Refined Informed Sampling (IRIS) based on Lower Bound Esti-

- Refinement iteration 5, the angular resolution $2\pi/64$.

**Uniform sampling**

$\epsilon = 2\pi/64, \ N = 448, \ T_{CPU} = 456 \text{ ms}$

$L = 14.5, \ L_U = 14.5$

**Informed sampling (IRIS)**

$\epsilon = 2\pi/64, \ N = 51, \ T_{CPU} = 48 \text{ ms}$

$L = 19.9, \ L_U = 13.9$
Decoupled Solution of the DTSP

Iteratively-Refined Informed Sampling (IRIS) based on Lower Bound Estimation using DIP

- Refinement iteration 6, the angular resolution $2\pi/128$.

### Uniform sampling

- $\epsilon = 2\pi/128$, $N = 896$, $T_{\text{CPU}} = 1620$ ms
- $L = 14.5$, $L_U = 14.5$

### Informed sampling (IRIS)

- $\epsilon = 2\pi/128$, $N = 70$, $T_{\text{CPU}} = 60$ ms
- $L = 14.8$, $L_U = 14.1$
- Refinement iteration 7 (final), the angular resolution $2\pi/256$.

**Uniform sampling**

$\epsilon = 2\pi/256$, $N = 1792$, $T_{CPU} = 6784$ ms

$L = 14.4$, $L_U = 14.3$

**Informed sampling (IRIS)**

$\epsilon = 2\pi/256$, $N = 100$, $T_{CPU} = 88$ ms

$L = 14.4$, $L_U = 14.3$
Comparison with the AA, LIO, IRIS (with Lower bound) and Memetic algorithms.


In the decoupled approaches (AA, LIO, IRIS), a sequence of the waypoint locations is determined as the Euclidean TSP (ETSP).

In Memetic algorithm, the best sequence of visits is determined during the solution.

Similarly to the sampling-based approaches that solve the Generalized TSP.
Surveillance Missions with Non-Zero Sensing Radius
Solution of the DTSP with Neighborhoods (DTSPN)
Dubins Traveling Salesman Problem with Neighborhoods (DTSPN)
Problem Formulation

- In surveillance planning, it may be required to visit a set of target regions \( G = \{R_1, \ldots, R_n\} \) by Dubins vehicle.

- For each target region \( R_i \), we have to determine a particular point of the visit \( p_i \in R_i \) and the DTSP becomes the **Dubins Traveling Salesman Problem with Neighborhoods (DTSPN)**.

- In addition to \( \Sigma \) and headings \( \Theta \), waypoint locations \( P \) have to be determined.

DTSPN is an optimization problem over all permutations \( \Sigma \), headings \( \Theta = \{\theta_{\sigma_1}, \ldots, \theta_{\sigma_n}\} \) and points \( P = (p_{\sigma_1}, \ldots, p_{\sigma_n}) \) for the states \( q_{\sigma_1}, \ldots, q_{\sigma_n} \) such that \( q_{\sigma_i} = (p_{\sigma_i}, \theta_{\sigma_i}) \) and \( p_{\sigma_i} \in R_{\sigma_i} \).

\[
\begin{align*}
\text{minimize}_{\Sigma, \Theta, P} & \quad \sum_{i=1}^{n-1} \mathcal{L}(q_{\sigma_i}, q_{\sigma_{i+1}}) + \mathcal{L}(q_{\sigma_n}, q_{\sigma_1}) \\
\text{subject to} & \quad q_i = (p_i, \theta_i), \; p_i \in R_i \; i = 1, \ldots, n.
\end{align*}
\] (1)

\( \mathcal{L}(q_{\sigma_i}, q_{\sigma_j}) \) is the length of the shortest possible Dubins maneuver connecting the states \( q_{\sigma_i} \) and \( q_{\sigma_j} \).
Similarly to the DTSP, also the DTSPN can be addressed by

- **Decoupled approaches** for which a sequence of visits to the regions can be found as a solution of the ETSP(N).
- **Sampling-based approaches** and formulation as the Generalized TSP (GTSP).
  - Clusters of sampled waypoint locations each with sampled possible heading values.
- **Combinatorial heuristics** such as Variable Neighborhood Search (VNS).
- **Soft-computing** techniques such as memetic algorithms.
- **Unsupervised learning** techniques.

Váňa and Faigl (IROS 2015), Faigl and Váňa (ICANN 2016, IJCNN 2017)

Similarly to the lower bound of the DTSP based on the Dubins Interval Problem (DIP) a lower bound for the DTSPN can be computed using Generalized DIP (GDIP).
1. Determine a sequence of visits to the \( n \) target regions as the solution of the ETSP(N).
2. Sample possible waypoint locations and for each such a location sample possible heading values, e.g., \( s \) locations per each region and \( h \) headings per each location.
3. Construct a search graph and determine a solution in \( O(n(sh)^3) \).

An example of the search graph for \( n = 6, s = 6, \) and \( h = 6 \).
As in the solution of the DTSP, we can perform local optimization to determine (locally optimal) heading angle $\theta_i$.

In a similar way, we can locally optimize the waypoint location $p_i$.

The waypoint location $p_i$ can be parametrized as a point at the boundary of the respective region $R_i$ using a parameter $\alpha \in [0, 1)$ measuring a normalized distance on the boundary of $R_i$, i.e., $p_i \in \delta R_i$.

**Algorithm 2:** Local Iterative Optimization (LIO) for the DTSP

Data: Input sequence of the regions $G = (R_{\sigma_1}, \ldots, R_{\sigma_n})$, for the permutation $\Sigma$

Result: Waypoints $(q_{\sigma_1}, \ldots, q_n)$, $q_i = (p_i, \theta_i)$, $p_i$ is at the border of $R_i$, i.e., $p_i \in \delta R_i$. 

1. initialization() // random assignment of $q_i = (p_i, \theta_i)$, $\theta_i \in [0, 2\pi)$, $p_i \in \delta R_i$;

2. while global solution is improving do

3.   for every $R_i \in G$ do

4.     $\theta_i := \text{optimizeHeadingLocally}(\theta_i)$;

5.     $\alpha_i := \text{optimizePositionLocally}(\alpha_i)$;

6.     $q_i := \text{checkLocalMinima}(\theta_i, \alpha_i)$;

7.   end

8. end
- In the DTSPN, we need to determine not only the **headings** but the waypoint locations themselves.

- Dubins Interval Problem (DIP) is not sufficient to provide tight lower bound.

- **Generalized Dubins Interval Problem (GDIP)** can be utilized for the DTSPN similarly as the DIP for the DTSP.

Generalized Dubins Interval Problem (GDIP) and its Optimal Solution

- Determine the shortest Dubins maneuver connecting $P_i$ and $P_j$ given the angle intervals $\theta_i \in [\theta_{i \text{min}}, \theta_{i \text{max}}]$ and $\theta_j \in [\theta_{j \text{min}}, \theta_{j \text{max}}]$.

**Full problem (GDIP)**

**One-side version (OS-GDIP)**

- **Optimal solution** – Closed-form expressions for (1–6) and convex optimization (7).

<table>
<thead>
<tr>
<th>Average computational time</th>
<th>Problem</th>
<th>Time [μs]</th>
<th>Ratio</th>
</tr>
</thead>
<tbody>
<tr>
<td>Dubins maneuver</td>
<td>0.37</td>
<td>1.0</td>
<td></td>
</tr>
<tr>
<td>DIP</td>
<td>1.11</td>
<td>3.0</td>
<td></td>
</tr>
<tr>
<td>GDIP</td>
<td>5.42</td>
<td>14.5</td>
<td></td>
</tr>
</tbody>
</table>

https://github.com/comrob/gdip


Jan Faigl - Curvature-Constrained Multi-Goal Planning 2019 IEEE RAS Summer School on Multi-Robot Systems
DTSPN – Decoupled Solution based on Informed Sampling using Lower Bound Estimation based on the GDIP

- Iteratively-Refined Informed Sampling (IRIS) of the neighborhood intervals and heading angles.

Resolution: 4
Gap: 69.3 %
Time: 0.079 s
DTSPN – Decoupled Solution based on Informed Sampling using Lower Bound Estimation based on the GDIP

- Iteratively-Refined Informed Sampling (IRIS) of the neighborhood intervals and heading angles.

Resolution: 8  
Gap: 39.4 %  
Time: 0.211 s
Iteratively-Refined Informed Sampling (IRIS) of the neighborhood intervals and heading angles.

Resolution: 16  
Gap: 19.9 %  
Time: 0.552 s
Iteratively-Refined Informed Sampling (IRIS) of the neighborhood intervals and heading angles.

Resolution: 32  
Gap: 10.7 %  
Time: 1.292 s
DTSPN – Decoupled Solution based on Informed Sampling using Lower Bound Estimation based on the GDIP

- Iteratively-Refined Informed Sampling (IRIS) of the neighborhood intervals and heading angles.

Resolution: 64  
Gap: 5.3 %  
Time: 3.183 s
ICP-TRM – Decoupled Solution based on Informed Sampling using Lower Bound Estimation based on the GDIP

- Iteratively-Refined Informed Sampling (IRIS) of the neighborhood intervals and heading angles.

Resolution: 128  
Gap: 2.6%  
Time: 8.994 s
Iteratively-Refined Informed Sampling (IRIS) of the neighborhood intervals and heading angles.
Algorithm 3: Informed sampling-based solution of the DTRP using optimal solution the GDIP.

**Input:** \( R \) – Sequence of the regions to be visited

**Input:** \( \omega_{\text{max}} \) – Maximal requested resolution

**Output:** \( Q \) – Visiting configurations of the final tour

**Output:** \( L_L \) – Lower bound (unfeasible)

**Output:** \( L_U \) – Upper bound (feasible)

1. \( \omega \leftarrow 1 \) // initial resolution
2. \( S \leftarrow \text{sampleIntervals}(R) \) // initial intervals
3. \( L_L \leftarrow 0 \) // init lower bound
4. \( L_U \leftarrow \infty \) // init upper bound
5. while \( \omega \leq \omega_{\text{max}} \) do
6. \( (S, L_L) \leftarrow \text{refineLowerBound}(R, \omega, S) \)
7. \( (Q, L_U) \leftarrow \text{findFeasibleSolution}(R, S) \)
8. \( \omega \leftarrow 2 \omega \)
9. end
10. return \( Q, L_L, L_U \)

- Maximal resolution of the position \( w_p \).
- Maximal resolution of the heading \( w_h \).
Convergence of the Solution to Lower Bound

Optimality Gap of the Lower and Feasible Solution

- Convergence of the solution to lower bound for 20 random instances with $n = 10$ target regions.

- Optimality Gap $\Delta$ of the feasible solution to the lower bound solution.

Váňa and Faigl: *Optimal Solution of the Generalized Dubins Interval Problem*, RSS 2018

Multi-Vehicle Multi-Goal Planning with Dubins Vehicle – \( m \)-DTSPN
Having $m$ vehicles and $n$ target regions, we aim to determine $m$ Dubins tours to visit the regions as quickly as possible. *For each vehicle, we consider a separate starting (ending) location.*

- We need to allocate $n$ regions to $m$ vehicles such that each region is visited by one vehicle and the longest tour is minimized (MinMax).

Existing approaches can be categorized into

- **Cluster-First, Route-Second** – regions are firstly assigned (clustered) to the vehicles, and then the problem is addressed as a set of instances of the single-vehicle DTSPN. E.g., using K-Means (or Hungarian algorithm) with the Euclidean distance of their centers.

- **Constructive Heuristic** – Incremental construction of the tours.

- **Compound optimization** of regions allocation, sequences, headings and waypoint locations.
Sample possible waypoint locations ($s$ samples) and headings angles ($h$ samples) into set of waypoint locations for each region $R_i$.

Sort all regions according to its minimal distance $d_{min}^i$ from its center to any of $m$ starting locations (using Euclidean distance).

For each vehicle $r$, create a small tour using the initial location $q^r_0$ and $q^r_1$ from the nearby region such that Dubins tour $Q^r = (q^r_0, q^r_1, q^r_0)$ has the minimal length – minimize $L(Q^r_{init}) = L(q^r_0, q^r_1) + L(q^r_1, q^r_0)$.

Not assigned regions are sequentioanal (with increasing $d_{min}^i$) and the waypoint $q(R_i)$ is inserted into the tour $r^*$ at the position $j^*$ according to

$$ r^*, j^* = \arg\min_{1 \leq r \leq m, 1 \leq j \leq n_r} (L(q^r_j, q(o_i)) + L(q(o_i), q^r_{j+1})) \mathcal{W}(m, r), \quad (3) $$

where $q(R_i)$ is the most suitable waypoint from $s \cdot h$ samples of $R_i$. The term $\mathcal{W}(m, r)$ is the competitive weight to address the minmax variant of the $m$-TSP.

$$ \mathcal{W}(m, r) = 1 + \frac{L(Q^r) + L_{avg}^m}{L_{avg}^m}, \quad (4) $$

where $L_{avg}^m$ is the average length of the Dubins tours $Q^1, \ldots, Q^m$

$$ L_{avg}^m = \frac{1}{m} \sum_{r=1}^{m} L(Q^r). \quad (5) $$
Provide curvature-constrained paths for a team of autonomous unmanned aerial vehicles to verify expected objects of interest.

\[ v = 5 \text{ m.s}^{-1}, \, 80 \text{ m} \times 60 \text{ m} \] testing site for experimental verification of our system for the Mohamed Bin Zayed International Robotics Challenge (MBZIRC)

- Sampling-based methods are relatively slow.
- Desired properties of the requested surveillance mission planner are: fast trajectories and low computational time (\( \leq 1 \text{ s} \)).
**Fast Heuristics for $m$-DTSPN**

- **Fast heuristic solutions** to quickly (in less than 1 second) find a solution for the MBZIRC 2017 scenario.

- Comparison with Memetic (Zhang et al., 2014) and VNS-based algorithms restricted to the maximal computational time $T_{\text{max}} \in \{1, 5, 10, 60\}$ seconds and $k$ vehicles.

- Quick unsupervised learning (SOM) based heuristic and greedy constructive heuristic used as VNS Init.

<table>
<thead>
<tr>
<th>Problem</th>
<th>$m = 1$</th>
<th>$m = 2$</th>
<th>$m = 3$</th>
<th>$m = 1$</th>
<th>$m = 2$</th>
<th>$m = 3$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$mbzirc22, \delta = 0.0$</td>
<td>224.2</td>
<td>195.5</td>
<td>173.9</td>
<td>94.5</td>
<td>50.5</td>
<td>39.9</td>
</tr>
<tr>
<td>$mbzirc22, \delta = 0.5$</td>
<td>165.9</td>
<td>157.5</td>
<td>159.7</td>
<td>1 108.8</td>
<td>1 232.8</td>
<td>1 206.8</td>
</tr>
<tr>
<td>$mbzirc22, \delta = 1.0$</td>
<td>152.7</td>
<td>151.1</td>
<td>151.6</td>
<td>1 312.2</td>
<td>1 160.1</td>
<td>1 100.0</td>
</tr>
<tr>
<td>$mbzirc22, \delta = 2.0$</td>
<td>134.4</td>
<td>135.2</td>
<td>142.0</td>
<td>1 978.8</td>
<td>1 085.9</td>
<td>983.7</td>
</tr>
</tbody>
</table>

Examples of Generalized Multi-Goal Planning Problems with Curvature-Constrained Vehicles
Dubins Airplane model describes the vehicle state $q = (p, \theta, \psi)$, $p \in \mathbb{R}^3$ and $\theta, \psi \in S^1$.

Chitsaz, H., LaValle, S.M. (2017)

Constant forward velocity $v$, the minimal turning radius $\rho$, and limited pitch angle, i.e., $\psi \in [\psi_{\text{min}}, \psi_{\text{max}}]$. 
3D Multi-Goal Planning with Dubins Airplane Model
Solutions of the 3D-DTSPN

- Parametrization of 3D regions to be visited.

- Solutions based on LIO (ETSP+LIO), TSP with the travel cost according to Dubins Airplane Model (DAM-TSP+LIO), and sampling-based approach with the transformation of the GTSP to the ATSP solved by LKH.

Algorithm 4: LIO-based Solver for 3D-DTSPN

Data: Regions \( R \)
Result: Solution represented by \( Q \) and \( \Sigma \)
1. \( \Sigma \leftarrow \text{getInitialSequence}(R); \)
2. \( Q \leftarrow \text{getInitialSolution}(R, \Sigma); \)
3. while terminal condition do
   4. \( Q \leftarrow \text{optimizeHeadings}(Q, R, \Sigma); \)
   5. \( Q \leftarrow \text{optimizeAlpha}(Q, R, \Sigma); \)
   6. \( Q \leftarrow \text{optimizeBeta}(Q, R, \Sigma); \)
4. end
5. return \( Q, \Sigma; \)

Multi-rotor aerial vehicles can generally move in arbitrary direction.
- It is not limited by the minimal turning radius $\rho$.
- They can accelerate on straight segments and decelerate before turning.

Find a 3D smooth trajectory visiting a given set of 3D regions.

Minimizes the Travel Time Estimation (TTE).

Satisfies limited velocity and acceleration of the vehicle.

Benefits of Bézier curves

- Flexible and easy to use.
- Start/end direction is given by the first/last two control points.

Example of a cubic Bézier curve

\[ X(\tau) = B_0(1 - \tau)^3 + 3B_1\tau(1 - \tau)^2 + 3B_2\tau^2(1 - \tau) + B_3\tau^3 \]

Multi-Goal Planning with Bézier Curves
Real Experimental Results


Jan Faigl - Curvature-Constrained Multi-Goal Planning
2019 IEEE RAS Summer School on Multi-Robot Systems
Visit the most important targets because of limited travel budget $T_{\text{max}}$.

The problem can be formulated as a variant of the Orienteering Problem.

Let the given set of $n$ locations $P = \{p_1, \ldots, P_n\}$, be located in $\mathbb{R}^2$, $p_i \in \mathbb{R}^2$.

Each $p_i$ has an associated score $\zeta_i$ characterizing the reward if $p_i$ is visited.

Determine a subset of $k$ locations $P_k \subseteq P$ that maximizes the sum of the collected rewards while the travel cost to visit them is not exceeding $T_{\text{max}}$.

\[
\text{maximize}_{k,P_k,\Sigma} \quad R = \sum_{i=1}^{k} \zeta_{\sigma_i} \\
\text{subject to} \quad \sum_{i=2}^{k} |(p_{\sigma_{i-1}}, p_{\sigma_i})| \leq T_{\text{max}} \quad \text{and} \quad p_{\sigma_1} = p_1, p_{\sigma_k} = p_n.
\]

The Orienteering Problem (OP) combines two NP-hard problems:

- Knapsack problem in determining the most valuable locations $P_k \subseteq P$.
- Travel Salesman Problem (TSP) in determining the shortest tour.
Visit the most important targets with the limited travel budget.

Formulated as the **Dubins Orienteering Problem** (DOP).

It can be solved using sampling-based methods, e.g., sampling possible headings into a finite discretize set of headings values.

**Variable Neighborhood Search** (VNS) combinatorial meta-heuristic can be then employed.


Similarly, the **Dubins Orienteering Problem with Neighborhoods** (DOPN) can be formulated and solved.

We need to sample the waypoint locations and headings as in the DTSPN.


We might also consider the discretized problem as the **Set OP**.

Operational time of multi-rotor aerial vehicles is limited and only a subset of locations can be visited.

Planning multi-goal trajectories as a sequence of Bézier curves.

Targets are mised in a case of colliding trajectories, because of local collision avoidance and optimal trajectory following.

There is a practical need to include coordination in multi-vehicle multi-goal trajectory planning.

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** – motion planning is addressed by PRM* and VNS is utilized for routing problem.
  - 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 $T_{\text{max}}$.

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

Planning using the proposed VNS-PRM*
Conclusion
Multi-goal planning with Dubins vehicle

- **Dubins Traveling Salesman Problem (DTSP)** and **DTSP with Neighborhoods (DTSPN)**
- Decoupled approaches based on ETSP and heading optimization
  - Alternating Algorithm (AA); Local iterative optimization (LIO); **Sampling and graph search**
  - Iteratively-Refined Informed Sampling (IRIS) using **lower bound** based on the DIP and GDIP
- Sampling-based transformation methods based on the **Generalized TSP** and Noon-Bean transformation to the **Asymmetric TSP**.
- Multi-Vehicle Planning – \( m \)-DTSPN
  - Cluster-First, Route-Second; **Greedy Constructive Heuristic**
  - Variable Neighborhood Search (VNS) and Unsupervised Learning based methods
- Generalization to 3D planning (Dubins airplane model, Bézier curves, motion planning)
- Multi-goal planning with limited travel budget – variants of the **Orienteering Problem**
People Behind the Scene

- The presented work are mostly results of my colleagues from the Computational Robotics Laboratory and also Multi-Robot Systems Group.

- Work with us within the
  - Computational Robotics Laboratory – https://comrob.fel.cvut.cz
  - Artificial Intelligence Center (AIC) – http://aic.cvut.cz
  - Research Center for Informatics – http://rci.cvut.cz

- Open positions: phd, post-doc, and senior tenure-track