GNG-based Clustering of Risk-aware Trajectories into Safe Corridors

Jakub Sláma*, Petr Váňa, and Jan Faigl

Department of Computer Science, Faculty of Electrical Engineering
Czech Technical University in Prague
Technická 2, 166 27, Prague 6, Czech Republic
*Corresponding author: Jakub Sláma (slamajak@fel.cvut.cz)
{slamajak,vanapet1,faiglj}@fel.cvut.cz,
WWW home page: https://comrob.fel.cvut.cz/

Abstract. Personal air transportation on short distances is a promising trend in modern aviation, raising new challenges as flying in low altitudes in highly populated environments induces additional risk to people and properties on the ground. Risk-aware planning can mitigate the risk by preferring flying above low-risk areas such as rivers or brownfields. Finding such trajectories is computationally demanding, but they can be precomputed for areas that are not changing rapidly and form a planning roadmap. The roadmap can be utilized for multi-query trajectory planning using graph-based search. However, a quality roadmap is required to provide a low-risk trajectory for an arbitrary query on a risk-aware trajectory from one location to another. Even though a dense roadmap can achieve the quality, it would be computationally demanding. Therefore, we propose to cluster the found trajectories and create a sparse roadmap of safe corridors that provide similar quality of risk-aware trajectories. In this paper, we report on applying Growing Neural Gas (GNG) in estimating the suitable number of clusters. Based on the empirical evaluation using a realistic urban scenario, the results suggest a significant reduction of the computational burden on risk-aware trajectory planning using the roadmap with the clustered safe corridors.

Keywords: Growing Neural Gas, Risk-aware planning, Urban Air Mobility

1 Introduction

Urban Air Mobility (UAM) is an emerging field in the aerospace industry aiming to provide personal air transportation on short distances [6] that would result in raising the number of small aircraft flying in urban areas [8], which will increase the risk of possible accidents. Any accident is not only a threat to the aircraft and people on board but also to people and properties on the ground. A high population density can characterize urban areas, and so any crash can have immense consequences, such as the number of casualties or the caused material damage. Hence, the UAM brings challenges in risk mitigation.

The risk can be mitigated by risk-aware trajectory planning [10] to find the least risky trajectory with the minimal induced risk in the case of a malfunction. Risk evaluation can be prohibitively demanding for real-time queries or simultaneous planning of multiple aircraft. Therefore, a roadmap of risk-aware trajectories can be determined to support graph-based planning methods. Furthermore, the found least risky trajectories tend to be over less populated areas such as rivers or less risky brownfields [7]. Such low-risk trajectory segments can be called safe corridors, and similar safe corridor segments are common among different least risky trajectories. Hence, the safe corridors can be determined and utilized to determine a roadmap that would be a relatively dense roadmap of safe (individual) trajectories otherwise. Then, existing graph-based planning techniques can be employed for fast near-optimal risk-aware trajectory planning.

In this paper, we propose a clustering method based on the *Growing Neural Gas* (GNG) algorithm [5] to determine safe corridors from a determined set of least risky trajectory samples. The problem is determining a relatively spare roadmap that would provide risk-aware trajectories with similar risk to the original set of trajectories. The roadmap of the safe corridors can then be employed in computationally efficient multi-query risk-aware trajectory planning. The approach is based on the unsupervised clustering of the GNG to determine the roadmap vertices in the most used locations in the set of input trajectories. The roadmap edges are then determined based on relations between the vertices encoded in the GNG structure.

2 Problem Specification

The addressed problem of determining safe corridors from the set of available risk-aware trajectories builds on the existing work to determine the least risky trajectories [11]. However, the formal vehicle model is needed to determine the final trajectory roadmap because it induces the proposed distances between two sampled trajectory segments required in the clustering. Therefore, a brief overview of the notion is presented to make the paper self-contained. For a detailed definition of the least risky trajectory planning problem, the reader is referred to [11].

The aircraft is modeled as Dubins Airplane [2] with the configuration $q = (x, y, z, \theta, \psi)$ with the position $(x, y, z) \in \mathbb{R}^3$, $\theta \in \mathbb{S}$ denoting the heading angle, and $\psi \in \mathbb{S}$ standing for the pitch angle. The trajectories are thus determined in the configuration space $\mathcal{C} = \mathbb{R}^3 \times \mathbb{S}^2$ and the aircraft state can be expressed as

$$\begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{z} \\ \dot{\theta} \end{bmatrix} = v \begin{bmatrix} \cos\theta\cos\psi \\ \sin\theta\cos\psi \\ \sin\psi \\ u_\theta \rho^{-1} \end{bmatrix}, \tag{1}$$

where v is the forward velocity of the aircraft, $u_\theta \in [-1, 1]$ denotes the control input controlling the heading angle θ, and ρ is the minimum turning radius of the aircraft. The pitch angle ψ is considered to change significantly faster than

θ, and thus abrupt changes in ψ are allowed, but $\psi \in [\psi_{\min}, \psi_{\max}]$ has to be satisfied. Trajectories are planned in the collision-free part of $\mathcal{C}$ denoted as $\mathcal{C}_{\text{free}}$.

For a trajectory $\Gamma : [0, T_\Gamma] \to \mathcal{C}_{\text{free}}$ going from an initial configuration $q_i = \Gamma(0)$ to the final configuration $q_f = \Gamma(T_\Gamma)$, the risk r induced by Γ is given as

$$r(\Gamma) = \int_0^{T_\Gamma} \mathcal{M}\left(\Gamma(t)\right) \, \mathrm{d}t \,, \tag{2}$$

where $\mathcal{M}$ is an aircraft specific risk map, and $\mathcal{M}(q)$ denotes the risk associated with any configuration q. Note that $\Gamma(t)$ denotes the aircraft configuration q on the trajectory Γ at the time t. Planning the least risky trajectory Γ^* between two configurations q_i and q_f stands to find a feasible trajectory from q_i to q_f with the minimal induced risk r that can be expressed as

$$\Gamma^* = \arg\min_{\Gamma} r(\Gamma) \,, \qquad \text{s.t.} \quad \Gamma^*(0) = q_i, \;\; \Gamma^*(T_{\Gamma^*}) = q_f \,, \tag{3}$$

with the associated risk

$$r^*(q_i, q_f) = r\left(\Gamma^*\right) \,. \tag{4}$$

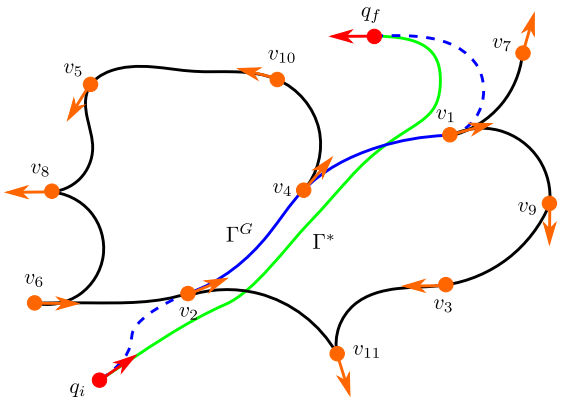


Fig. 1. An example of a safe corridors roadmap G with vertices $\boldsymbol{V}$ (in orange). The risk-aware trajectory Γ (in green) between configurations q_i and q_f (in red) can be approximated by the trajectory Γ^G (in blue) found through the roadmap G. The configurations q_i and q_f are inserted into the roadmap (shown by dashed lines), and the least risky trajectory in the graph is found. As Γ^G is only an approximation of Γ, its risk can be higher compared to the risk of Γ.

Denote $\Xi = \{\Gamma_1, \Gamma_2, \ldots, \Gamma_n\}$ as a set of least risky trajectories from various locations covering the operational area. Let $G(\boldsymbol{V}, \boldsymbol{E})$ denote a roadmap of trajectories connecting configurations associated to the vertices $\boldsymbol{V}$ connected by edges $\boldsymbol{E}$ representing the least risky trajectories. Since G created from Ξ would be very dense and thus graph-based searching for the least risky trajectory would

be demanding, the problem is to find G representing trajectories Ξ such that finding an approximation of the least-risky trajectory Γ^G using the roadmap G would be less demanding than determining Γ. The aim is to find G that minimizes the relative risk between trajectories $\Gamma_i \in \Xi$ and their corresponding approximations from G. An example of trajectories is depicted in Fig. 1.

Let Γ^G be the least risky trajectory from q_i to q_f through G. Γ^G visits the intermediate configurations $v_i \in \boldsymbol{V}$ that can be expressed as a sequence of indices $\Sigma = \{\sigma_1, \sigma_2, \ldots, \sigma_{|\Sigma|}\}$. The corresponding risk r^G of the roadmap-based trajectory Γ^G can be expressed as

$$r^G(q_i, q_f) = \min_{\Sigma} \left[r(q_i, v_{\sigma_1}) + \sum_{k=1}^{|\Sigma|-1} r(v_{\sigma_k}, v_{\sigma_{k+1}}) + r(v_{\sigma_{|\Sigma|}}, q_f) \right]. \tag{5}$$

The addressed creation of the safe corridors roadmap G is a problem determining G with a suitable number of vertices k from the trajectories Ξ such that the mean relative risk between the original trajectory Γ and its approximation Γ^G is minimal. The optimization problem can be formally defined as Problem 1.

Problem 1 (Determining Safe Corridors Roadmap G).

$$\underset{G}{\text{minimize}} \ \frac{1}{n} \sum_{j=1}^{n} \frac{r^G(q_i^j, q_f^j)}{r(\Gamma_j)} \tag{6}$$

3 Proposed GNG-based Safe Corridors Roadmap

The proposed safe corridors roadmap determination is based on clustering the given set of trajectories Ξ using GNG to learn the best locations for the roadmap nodes. The edges are then created using the learned topology encoded in the GNG. The proposed roadmap creation is summarized in Algorithm 1, and it works as follows.

The algorithm starts by sampling the trajectories Ξ with the sampling step Δ (Lines 3 to 5, Algorithm 1); the set of all samples is denoted as $\mathcal{Q}$. The samples are from the *Special Euclidean* group $SE(2) = \mathbb{R}^2 \times \mathbb{S}$, where a sample is given by its position and also its orientation. The nodes $\boldsymbol{V}$ of the safe corridors roadmap G are found by the GNG [5] modified for the $SE(2)$ configuration space by the proposed `LearnGNG` method detailed in Section 3.1. Once the GNG is terminated, the found nodes serve as the nodes of the safe corridors roadmap G (Line 7, Algorithm 1). Then, the edges of G are created using GNG edges that are non-directional and only denote the relations between nodes, not the actual corridors. Therefore a direction with the shorter feasible maneuver is preferred as the directed edge of the roadmap G (Lines 8 to 12, Algorithm 1).

3.1 GNG-based Safe Corridors Topology Learning

The original GNG algorithm [5] is adapted for $SE(2)$ to respect a distance between two nodes, which depends on the Euclidean distance and also on the

Algorithm 1: Safe Corridors Roadmap Creation

Input: $\mathcal{M}$ – Risk map; Ξ – Set of trajectories.
Parameters: Δ – Sampling step; $\mathcal{A}$ – Aircraft model.
Output: G – Roadmap of safe corridors.

Function `CreateCorridors`$(\mathcal{M}, \Xi)$**:**
  $\mathcal{Q} \leftarrow \emptyset$
  forall $\Gamma_j \in \Xi$ **do**
    $\{q_1^j, q_2^j, \ldots, q_{m_j}^j\} \leftarrow$ `SamplePath`(Γ_j, Δ)
    $\mathcal{Q} \leftarrow \mathcal{Q} \cup \{q_1^j, q_2^j, \ldots, q_{m_j}^j\}$
  $G_{\text{GNG}} \leftarrow$ `LearnGNG`$(\mathcal{Q})$
  $G \leftarrow \{\boldsymbol{V} \leftarrow \boldsymbol{V}_{\text{GNG}}, \boldsymbol{E} \leftarrow \emptyset\}$
  forall $(s_1, s_2) \in \boldsymbol{E}_{\text{GNG}}$ **do**
    if $\mathcal{L}\left(\texttt{Dubins3D}(s_1, s_2, \mathcal{A})\right) < \mathcal{L}\left(\texttt{Dubins3D}(s_2, s_1, \mathcal{A})\right)$ **then**
      $\boldsymbol{E} \leftarrow \boldsymbol{E} \cup \{(s_1, s_2)\}$
    else
      $\boldsymbol{E} \leftarrow \boldsymbol{E} \cup \{(s_2, s_1)\}$
  return G

vehicle's heading angle. The algorithm is summarized in Algorithm 2, and it works as follows.

The GNG constructs a group roadmap G_{GNG} initialized by two nodes s_a and s_b at randomized locations q_a and q_b (Lines 2 to 4, Algorithm 2). Then, the learning epoch is repeated until a termination condition is met. In each iteration, a random configuration (signal) is generated (Line 6, Algorithm 2) with the probability distribution that should correspond to the probability distribution of the underlying data; in our case, the underlying topology of safe corridors. Since the trajectories Ξ are sampled uniformly, the condition is satisfied by selecting random samples from $\mathcal{Q}$.

In the second step, the closest node s_1 and the second-closest node s_2 to ξ are found by the `GetTwoClosestNodes` method. The distance function `Dist` between two configurations q_1 and q_2 is computed as

$$\texttt{Dist}(q_1, q_2) = \max\left(\left\| q_1^{\text{2D}} - q_2^{\text{2D}} \right\|, \frac{d_{\max}}{\Delta_{\max}^{\theta}} \left| \angle(q_1^{\theta}, q_2^{\theta}) \right| \right), \tag{7}$$

where $\left\| q_1^{\text{2D}} - q_2^{\text{2D}} \right\|$ is the Euclidean distance between the samples, $\angle(q_1^{\theta}, q_2^{\theta})$ is the angular difference between their headings, and $\Delta_{\max}^{\theta}$ and $d_{\max}$ are the maximum heading difference and Euclidean distance between two samples to be considered as near, respectively. Thus, the closest node s_1 and the second closest node s_2 are given as

$$s_1 = \underset{v_i \in \boldsymbol{V}}{\arg\min}\ \texttt{Dist}(\texttt{q}(v_i), \xi), \tag{8}$$

$$s_2 = \underset{v_i \in \boldsymbol{V} \setminus s_1}{\arg\min}\ \texttt{Dist}(\texttt{q}(v_i), \xi), \tag{9}$$

where $\texttt{q}(\cdot)$ provides the configuration q associated with the given node.

Algorithm 2: Learning of GNG (adopted from [5])

Input: $\mathcal{Q}$ – Set of samples.

Parameters: ϵ_b and ϵ_n – Attraction parameters for the closest node and neighbors of the closest node, respectively; $a_{\max}$ – Maximum endge age; λ – Number of generated signals before inserting new node; α and d – Error reduction parameters when inserting new node and for all nodes, respectively.

Output: G_{GNG} – GNG Roadmap.

1 **Function** `LearnGNG(`$\mathcal{Q}$`)`**:**
  `/* GNG Initialization */`
2 $q_a, q_b \leftarrow$ `RandomConfigurations(`$\mathcal{M}$`)`
3 $s_a, s_b \leftarrow$ `CreateNodes(`q_a, q_b`)`
4 $G_{\text{GNG}} \leftarrow \{\boldsymbol{V}_{\text{GNG}} \leftarrow \{s_a, s_b\}, \boldsymbol{E}_{\text{GNG}} \leftarrow \{(s_a, s_b)\}\}$
  `/* Learning phase of GNG */`
5 **while** *Terminal condition is not met* **do**
6 $\xi \leftarrow$ `Random(`$\mathcal{Q}$`)`; $n_\xi \leftarrow n_\xi + 1$ `// Step 1: signal generation`
7 $s_1, s_2 \leftarrow$ `GetTwoClosestNodes(`$\boldsymbol{V}_{\text{GNG}}, \xi$`)` `// Step 2: nearest nodes`
8 **forall** $e \in$ `GetConnections(`G_{GNG}, s_1`)` **do** `// Step 3: age update`
9 `Age(`e`)` $\leftarrow$ `Age(`e`)` $+ 1$
10 `Error(`s_1`)` $\leftarrow$ `Error(`s_1`)` $+$ `Dist(`s_1, ξ`)` `// Step 4: error update`
11 $\mathtt{q}(s_1) \leftarrow \mathtt{q}(s_1) + \epsilon_b\,(\xi - \mathtt{q}(s_1)))$ `// Step 5: position update`
12 **forall** $n \in$ `GetNeighbors(`G_{GNG}, s_1`)` **do**
13 $\mathtt{q}(n) \leftarrow \mathtt{q}(n) + \epsilon_n\,(\xi - \mathtt{q}(s_1))$
14 **if** $(s_1, s_2) \in \boldsymbol{E}_{\text{GNG}}$ **then** `// Step 6: edge update`
15 `Age(`(s_1, s_2)`)` $\leftarrow 0$
16 **else**
17 $\boldsymbol{E}_{\text{GNG}} \leftarrow \boldsymbol{E}_{\text{GNG}} \cup (s_1, s_2)$
18 **forall** $e \in \boldsymbol{E}_{\text{GNG}}$: `Age(`e`)` $> a_{\max}$ **do** `// Step 7: edge removal`
19 $\boldsymbol{E}_{\text{GNG}} \leftarrow \boldsymbol{E}_{\text{GNG}} \setminus e$
20 **if** $n_\xi > \lambda$ **then** `// Step 8: node insertion`
21 $n_\xi \leftarrow 0$
22 `InsertNewNode(`G_{GNG}`)`
23 **forall** $n \in \boldsymbol{V}$ **do** `// Step 9: error reduction`
24 `Error(`n`)` $\leftarrow d$ `Error(`n`)`
25 **return** G_{GNG}

For the position update (Line 11, Algorithm 2) a vector from s_1 to ξ is calculated, where we exploit features of Euler angles and the heading difference is calculated as

$$\theta_\xi - \theta_{s_1} = ((\theta_\xi - \theta_{s_1} + \pi) \bmod 2\pi) + \pi \tag{10}$$

to get a correctly oriented angle difference. The configuration of s_1 is adapted towards ξ with the power of ϵ_b (Line 11, Algorithm 2), and all direct neighbors of s_1 are adapted towards ξ with the power of ϵ_n (Lines 12 and 13, Algorithm 2).

The further important modification of the GNG is the node insertion. If λ random configurations (signals) ξ have been generated since the last node inser-

tion, a new node is generated by the `InsertNewNode` method and inserted into G_{GNG} (Lines 20 to 22, Algorithm 2). The insertion is summarized in Algorithm 3.

Algorithm 3: Insertion of a new node into the GNG roadmap

Input: $G_{\mathrm{GNG}}(\boldsymbol{V}, \boldsymbol{E})$ – GNG Roadmap.
Output: Updated roadmap $G_{\mathrm{GNG}}(\boldsymbol{V}, \boldsymbol{E})$.

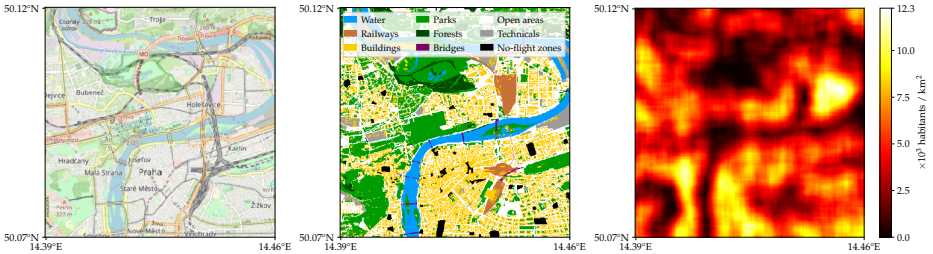
1 **Function** `InsertNewNode`(G_{GNG}):
2 $\quad q \leftarrow \underset{v_i \in \boldsymbol{V}}{\arg\min}\ \mathtt{Error}(v_i)$
3 $\quad f \leftarrow \underset{v_i \in \mathtt{GetNeighbors}(G_{\mathrm{GNG}}, q)}{\arg\min}\ \mathtt{Error}(v_i)$
4 $\quad r \leftarrow \mathtt{NewNode}\left(\frac{\mathtt{q}(q)+\mathtt{q}(f)}{2}\right)$
5 $\quad \boldsymbol{E} \leftarrow \boldsymbol{E} \setminus (q, f) \cup (q, r) \cup (r, f)$
6 $\quad \mathtt{Error}(q) \leftarrow \alpha\, \mathtt{Error}(q)$
7 $\quad \mathtt{Error}(f) \leftarrow \alpha\, \mathtt{Error}(f)$
8 $\quad \mathtt{Error}(r) \leftarrow \mathtt{Error}(q)$

During the insertion, the node q with the maximal error and its neighbor f with the maximal error are determined. A new node r with the configuration corresponding to the midpoint between q and f is created and connected to q and f while the edge between q and f is removed. Finally, the errors of q and f are reduced by the factor α, and the error of r is assumed to be identical to the error of q.

4 Results

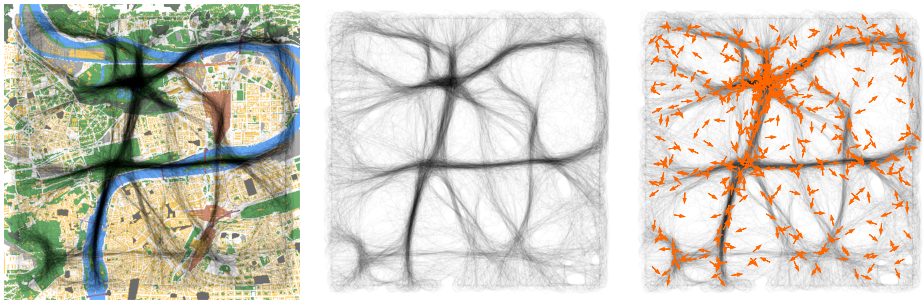
The proposed method has been empirically evaluated in a realistic urban scenario to test its performance and behavior. The test scenario consists of $5\,\mathrm{km} \times 5\,\mathrm{km}$ large urban area of the Prague city center adopted from [11]; the scenario is visualized in Fig. 2. The herein considered risk is measured as on-ground casualties in the case of a crash, and so the risk map is mainly given by the population density and shelter provided by the environment. The utilized aircraft model corresponds to Cessna 172 with the minimum turning radius $\rho = 65.7\,\mathrm{m}$ adopted from [12]. The set Ξ of 3 678 trajectories between randomly generated pairs of start and goal configurations has been found by the RRT*-based risk-aware trajectory algorithm [11]. The algorithm has been run for 750 s for each trajectory. Trajectories are visualized in Figs. 3a and 3b.

The maximum Euclidean distance $d_{\max} = 250\,\mathrm{m}$ and maximum angular difference $\Delta^{\theta}_{\max} = 30°$ have been used in the distance function (7). The used GNG learning parameters are $\epsilon_b = 0.3$, $\epsilon_n = 0.01$, $\alpha = 0.75$, $d = 0.995$, the maximum allowed edge age $a_{\max} = 300$, and a new node is inserted after generating $\lambda = 300$ signals. The utilized terminal condition is the number of vertices k selected as $k \in \{25, 50, 100, 250, 500, 1\,500\}$. The risk-based planning algorithm from [11] has been used for the evaluation of risk associated with the determined



(a) Map (b) Area type (c) Population density

Fig. 2. Selected layers of the utilized urban scenario based on the Prague city center adopted from [11]. The map layer is obtained from the OpenStreetMap [9] and real population density layer is based on [4].



(a) Trajectories over the city map (b) Heatmap of the least risky trajectories (c) Roadmap vertices

Fig. 3. A heatmap of the least risky trajectories obtained by [11] over (**a**) the city map, (**b**) the resulting heatmap of 3 678 trajectories, and (**c**) 500 vertices of the GNG-based roadmap (orange). Darker areas mean more trajectories passing through them.

safe corridors. The method has been implemented in Julia ver. 1.6.2 [1] and executed on a single core of the Intel© Core™ i7-9700 CPU. An example of the found safe corridor roadmap's nodes for $k = 500$ is depicted in Fig. 3c.

The quality of the roadmap G is evaluated using the risk induced by the trajectories between the initial and final configurations of the trajectories Ξ. The configurations have been inserted into the found roadmap, and the least risky path has been extracted using Dijkstra's algorithm [3]. The quality of G is characterized by the ratio of the sum of risks induced by all the trajectories Ξ.

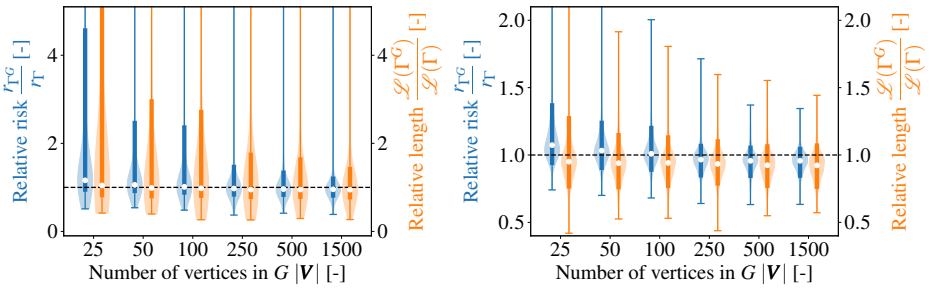
Table 1. Roadmap quality given by (6) based on 3 678 trajectories

Num. of vertices k	25	50	100	250	500	1 500
Roadmap quality [-]	1.80	1.33	1.27	1.08	1.08	1.01

Table 2. Computational demands of the proposed method

Num. of vertices	25	50	100	250	500	1 500
GNG Creation [s]	2	5	19	117	353	3 452
Roadmap edges creation [s]	6	11	18	33	47	103
Avg. query time [s]	2.3	1.9	1.8	1.3	1.0	0.8

The risk of the original trajectories and trajectories found using the safe corridors roadmap G has been evaluated using [11] and the results are depicted in Fig. 4. The quality of the roadmap is measured according to (6) with lower values meaning the higher quality of the roadmap. The roadmap quality for the evaluated number of vertices k is listed in Table 1, and the computational requirements are in Table 2.

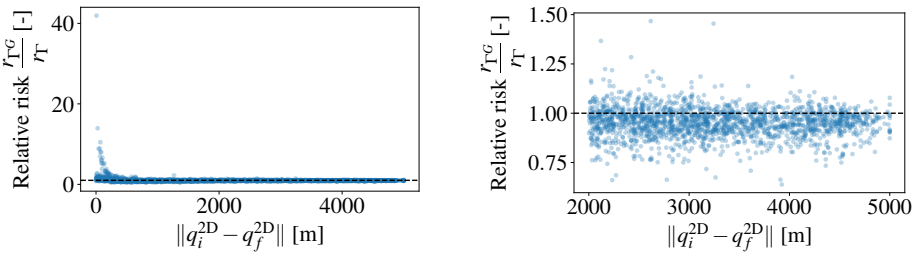


(a) All scenarios (b) Scenarios where $\|q_i^{\mathrm{2D}} - q_f^{\mathrm{2D}}\| > 2000\,\mathrm{m}$

Fig. 4. Influence of the number of roadmap vertices k on the trajectory relative risk and relative length compared to the original trajectories Ξ: (**a**) all solved scenarios, (**b**) scenarios where q_i and q_f are further apart. Medians (white dots) and 90 % non-parametric intervals (thick vertical lines) are shown. The relative risk and relative length reduce with the increasing number of the roadmap vertices as the found safe corridors provide better coverage.

The results support the idea that the safe corridors roadmap can provide trajectories with similar induced risk to the original trajectories Ξ. Increasing k increases the roadmap density that better covers the area. Notice the query times are decreased with increasing k. It is because the risk of maneuvers inserting initial and final configurations q_i and q_f into the roadmap needs to be evaluated, and the maneuvers are generally shorter with increasing roadmap density; the utilized risk evaluation [11] scales with the maneuver length. The best roadmap with $k = 1\,500$ vertices is further analyzed.

For $k = 1\,500$, the median risk reduction is 4 % compared to the reference method represented by the trajectories Ξ. The 90 % non-parametric interval of the relative risk is [0.77, 1.25]. Furthermore, the median length reduction is



(a) All scenarios (b) Scenarios where $\|q_i^{2D} - q_f^{2D}\| > 2000\,\text{m}$

Fig. 5. The relative risk based on the distance between q_i and q_f: (**a**) all solved trajectories, (**b**) trajectories where q_i and q_f are further apart.

5 % compared to the reference, and the 90 % non-parametric interval of relative length is [0.73, 1.47]. The analysis of the relative risk based on the Euclidean distance between the start and final configurations is depicted in Fig. 5. Notice that the risk is increased mainly for the short scenarios, in which the maneuvers connecting q_i and q_f into the roadmap can significantly prolong the solution, and so the risk. On the contrary, the risk is reduced compared to the reference for the longer maneuvers.

5 Conclusion

A safe corridors roadmap generation for risk-aware trajectory planning is presented in the paper. The addressed problem is motivated by reducing the computational demands of the single-query risk-aware trajectory planning methods by determining a set of trajectories from which a safe corridors roadmap is created. The proposed method employs the GNG-based clustering to detect common low-risk trajectory segments, so-called safe corridors, among provided risk-aware trajectories. The resulting roadmap allows using graph-based planning for fast on-demand queries. The proposed method needs tens to hundreds of seconds to build the roadmap. However, the queries can then be solved within seconds, compared to hundreds of seconds required by the existing single-query risk-aware trajectory planners.

Acknowledgments

The presented work has been supported by the Ministry of Education Youth and Sports (MEYS) of the Czech Republic under project No. LTAIZ19013 The support of the Czech Science Foundation (GAČR) under research project No. 19-20238S is also acknowledged.

References

1. Bezanson, J., Edelman, A., Karpinski, S., Shah, V.B.: Julia: A fresh approach to numerical computing. SIAM review 59(1), 65–98 (2017)
2. Chitsaz, H., LaValle, S.M.: Time-optimal paths for a dubins airplane. In: IEEE Conference on Decision and Control (CDC). pp. 2379–2384 (2007)
3. Dijkstra, E.W., et al.: A note on two problems in connexion with graphs. Numerische mathematik 1(1), 269–271 (1959)
4. Facebook Connectivity Lab and Center for International Earth Science Information Network - CIESIN - Columbia University: High Resolution Settlement Layer (HRSL). Source imagery for HRSL © 2016 DigitalGlobe (2016), accessed on: 10 Feb 2021
5. Fritzke, B.: A growing neural gas network learns topologies. Advances in neural information processing systems 7 (1994)
6. Hasan, S.: Urban air mobility (uam) market study. Tech. rep., National Aeronautics and Space Administration (NASA) (2019)
7. Melnyk, R., Schrage, D., Volovoi, V., Jimenez, H.: A third-party casualty risk model for unmanned aircraft system operations. Reliability Engineering & System Safety 124, 105–116 (2014)
8. Moore, M.: 21st century personal air vehicle research. In: AIAA International Air and Space Symposium and Exposition: The Next 100 Years. p. 2646 (2003)
9. OpenStreetMap contributors: Planet dump retrieved from https://planet.osm.org. `https://www.openstreetmap.org` (2021), accessed on: 10 Feb 2021
10. Primatesta, S., Rizzo, A., la Cour-Harbo, A.: Ground risk map for unmanned aircraft in urban environments. Journal of Intelligent & Robotic Systems 97(3), 489–509 (2020)
11. Sláma, J., Váňa, P., Faigl, J.: Risk-aware trajectory planning in urban environments with safe emergency landing guarantee. In: 2021 IEEE 17th International Conference on Automation Science and Engineering (CASE). pp. 1606–1612. IEEE (2021)
12. Váňa, P., Sláma, J., Faigl, J., Pačes, P.: Any-time trajectory planning for safe emergency landing. In: IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). pp. 5691–5696 (2018)