engineering proceedings

MDPI

Proceeding Paper

Risk-Based UAV Flight Path Optimization in Accordance with SORA †

Jannik Heinze and Maarten Uijt de Haag *

Institute of Aeronautics and Astronautics, Technical University of Berlin (TU Berlin), 10587 Berlin, Germany ; j.heinze@campus.tu-berlin.de

* Correspondence: maarten.uijtdehaag@tu-berlin.de

† Presented at the European Navigation Conference 2023, Noordwijk, The Netherlands, 31 May–2 June 2023.

Abstract: With the EU regulation of drone operations varying based on the specific type of drone, path planning can be done to consider risk mitigation. This paper proposes a transition-based rapidly exploring random tree star (T-RRT*) path planning algorithm for fixed-wing drone operations over rural areas. Risk is decoded in the cost function, which mainly considers population density and special infrastructure types. It was found that the algorithm is capable of finding paths that minimize exposure of the general population and infrastructure. However, path and computational inefficiencies were found. Usage of another data structure or algorithm might improve performance.

Keywords: optimal path planning; complex cost-space planning; sampling-based path planning; T-RRT*; rural drone operations; (EU) VO 2019/947

1. Introduction

On 31 December 2020 a new regulation with regard to the operation of Unmanned Aircraft Systems (UASs) in the European Union (EU) was enacted [1]. This regulation categorizes drone operations into three categories (open, specific and certified) and imposes different restrictions [1]. Of particular interest is the specific category, since it covers operation types relevant to the current industry. One peculiar use case of UASs seen in many places is the delivery of rather small and light objects over distances up to tens of kilometers. These operations are necessarily executed beyond visual line of sight (BVLOS), which puts them into a specific category [1]. Examples of these kinds of operations are those of Zipline and Wingcopter. According to the EU regulation, the risk of the operation depends on the route that is actually flown. This leads to the problem of finding a route that minimizes the risks according to the EU regulations while also taking into account the flight performance of the used Unmanned Aerial Vehicle (UAV) and economic parameters.

Ortlieb and Adolf used the Special Operation Risk Assessment (SORA) and Acceptable Means of Compliance (AMC) as well as additional data to create risk maps for minimal risk planning [2]. Work regarding path planning in higher dimensions and complex cost spaces as well as easy access rules for the mentioned regulations are available [1]. Examples of path planning for aerial vehicles are [3,4]. Schopferer and Benders also worked on risk-based path planning using a sampling-based algorithm [5]. Sampling-based path planning algorithms have been proven useful for large configuration spaces [4]. One of those is the Rapidly exploring Random Tree (RRT) algorithm, which has been altered into many different sub-variants with different strengths [4,6–8].

This contribution introduces a way to decode the Special Operation Risk Assessment (SORA) for a Specific Assurance and Integrity Level (SAIL) three together with other operational parameters into a cost function. Additionally, practical considerations for the application of the T-RRT* algorithm are introduced. This type of a priori fight path planning also fits into the efforts of the U-spaces outlined in the EU regulation, where proposed flight path planning could be carried out with information about expected UAV traffic.

check for updates

Citation: Heinze, J.; Uijt de Haag, M. Risk-Based UAV Flight Path Optimization in Accordance with SORA. *Eng. Proc.* **2023**, *54*, 56. https://doi.org/10.3390/ENC2023-15473

Academic Editors: Tom Willems and Okko Bleeker

Published: 29 October 2023

The target vehicles for the algorithm have a characteristic dimension of one to three meters or an expected kinetic energy between 700 joules and 34 kilo-joules. The algorithm will not take into account most dynamic behaviors and assumes a static ground speed with no wind effects or other disturbances.

2. Materials and Methods

The path planner proposed in this work is based on the T-RRT* algorithm by Devaurs, Simeon and Cortés [4]. The pseudocode of this algorithm can be found in Algorithm 1. This algorithm combines the well-known RRT* algorithm with filtering of the randomly sampled configuration by the relative cost with respect to the nearest configuration. It was chosen due to the stated benefits in complex cost spaces in [4].

The configuration space is modeled as $\mathbb{R}^6$ with three spacial coordinates in the UTM frame and three Euler angles. The UTM frame is chosen to have a Cartesian coordinate system, which also supports the data sources described below. The roll and pitch angles are limited to account for the flight performance of the UAV. The course angle is defined in the mathematical positive direction from the x-axis, against the standard definition in aviation. It is also defined in the interval between $-\pi$ and π.

For path planning, the UAV is considered to be a point of mass. Therefore, all obstacles have to be expanded in size to account for the Flight Geography (FG), Contingency Volume (CV) and Ground Risk Buffer (GRB). This is discussed in more detail during the data pre-processing.

Additionally, the UAV is modeled to be quasi-stationary. In order to abstract the algorithm proposed from any platform, the algorithm does not account for most dynamic behaviors. For example, it is assumed that the UAV has the same true airspeed, no matter the maneuver. In curved flights, it is assumed that the UAV instantly acquires the necessary bank angle. For changes to the path angle, a constant angular velocity $\dot{\gamma}$ with an instantaneous increase or decrease in the load factor n_z as a cause is assumed. These assumptions may not reflect the nature of the UAV perfectly but are considered useful for simple and vehicle-independent path planning.

According to the SORA process, SAIL 3 operations are only possible for a Ground-Risk Class (GRC) below or equal to 4 and residual Air-Risk Class (ARC) below or equal to b. Since ARC-a is not reasonable, the operation has to take place below 500 ft above ground level (AGL) in uncontrolled airspace over rural airspace. Depending on the encountered population density, ground risk mitigations (e.g., M1 and usage of a parachute for M2) need to be applied. The threshold for that is 300 inhabitants per square kilometer or a cluster of residential buildings. This, however, is just an interpretation of the regulation. Actual decisions of the competent authority may differ.

Another aspect considered to assess the risk imposed to the ground area is the ground infrastructure. While it may be considered relatively risk-free to fly over agricultural land, flights over power plants, prisons or biosafety level 4 laboratories have to be avoided. Aided by a list of different infrastructure types relevant to risk mitigation by the Federal Ministry for Digital and Transport (BMDV) of Germany [9], a set of areas deemed too risky for operations is set up. Collisions are determined based on the horizontal coordinates alone. Flying beneath or above airspaces is not supported in this work.

The cost function is set up as an integral of the cost, as in [4]. Four cost factors will be considered: height, number of overflown people, load on the propulsion system and flight over specific ground areas. This encourages paths at reasonable heights over sparsely populated regions with the least possible power draw, while also minimizing the flight over certain ground infrastructure types. The path cost function $c_p : \Pi \to \mathbb{R}^+$ and configuration cost function $c : \mathcal{C} \to \mathbb{R}^+$ are defined as follows (see [4]). Here, L represents the length of path π.

$$c_p(\pi) = \int_0^1 c(\pi(k))dk \approx \frac{L}{n} \sum_{k=1}^{n} c\left(\pi\left(\frac{k}{n}\right)\right) \tag{1}$$

$$c(q) = \lambda_h \cdot c_h(q) + \lambda_{pop} \cdot c_{pop}(q) + \lambda_{ENG} \cdot c_{ENG}(q) + \lambda_{infra} \cdot c_{infra}(q) \tag{2}$$

In order for the T-RRT* algorithm to work, the path cost function has to be linear on the path. Otherwise, adding the costs of two path segments would not represent the cost of the joined path. Furthermore, to effectively and comprehensibly weigh the different path factors, they should scale similarly with the path length and should be normalizable. The following configuration cost functions are chosen.

$$c_h(h(k)) = L\big(h(k) - \bar{h}\big)^2 \tag{3}$$

$$c_{pop}(x(k), y(k)) = L \cdot b_{OV} \cdot \rho_{pop}(x(k), y(k)) \tag{4}$$

$$c_{ENG}(\gamma(k)) = L \cdot (sin(\gamma(k)) + \epsilon \cdot cos(\gamma(k))) \tag{5}$$

$$c_{infra}(x(k), y(k)) = L \cdot infra(x(k), y(k)) \tag{6}$$

Here, $\bar{h}$ is the proposed flight height, b_{OV} is the width of the operational volume, ρ_{pop} is the population density, γ is the flight path angle, ϵ is the aerodynamic efficiency, and $infra(.,.)$ is 1 inside of penalized areas (e.g., motorways, railways, power lines, nature reserves) and otherwise 0. The propulsion system cost function is derived from linearized power consumption, yielding proportionality to the thrust required.

Normalization is achieved with the following values, restricting the normalized cost functions to $[0, L]$.

$$\lambda_{n_h} = \max_{h=\{h_{min}, h_{max}\}} (h - \bar{h})^2 \qquad \lambda_{n_{pop}} = b_{OV} \cdot 300 \left[\frac{1}{km^2}\right]$$

$$\lambda_{n_{ENG}} = 1 \qquad \lambda_{n_{infra}} = 1$$

Without hyper-parameter optimization the following weights are chosen:

$$\lambda_{w_h} = \frac{1.5}{\lambda_{n_h}} \qquad \lambda_{w_{pop}} = \frac{1}{\lambda_{n_{pop}}}$$

$$\lambda_{w_{ENG}} = \frac{1}{\lambda_{n_{ENG}}} \qquad \lambda_{w_{infra}} = \frac{10}{\lambda_{n_{infra}}}$$

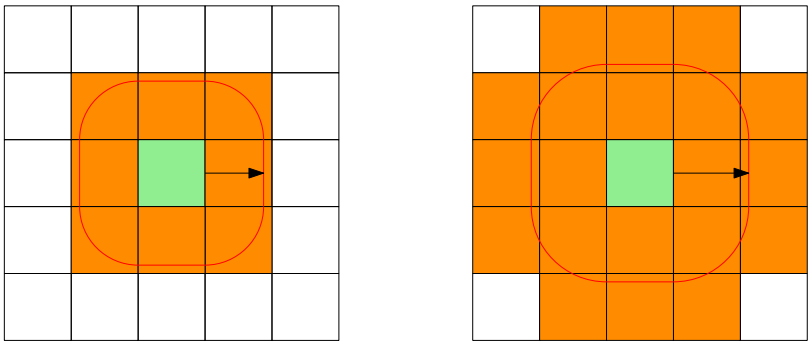
The definition of the configuration cost function is analogous to [5], especially with the normalization carried out.

As shown above, three different types of data are required: terrestrial surface data, population density data and infrastructure data. The data were gathered specifically for the study case explained in Section 3.

For the surface data as ground elevation measurements, the digital surface model ('Digitales Oberflächenmodell' (DOM)) is used. It is provided by each state separately [10,11]. The DOM data for Thuringia and Saxony–Anhalt gives grid-based information of surface elevation (elevation of vegetation, buildings, terrain, etc.) for a certain area in UTM coordinates (zone UTM32U). Since the UAV may deviate from the planned path to the extent of operational volume, the ground elevation of that area has to be considered for ground collision avoidance. In order to do so with reasonable computation time, a pre-processing step with respect to the width of the operational volume b_{OV} is executed. For a given set of coordinates, the processed elevation is equal to the maximal elevation in a circle with radius of $\frac{b_{OV}}{2}$ around that coordinate. This ensures staying well-clear of the ground even when deviating from the path inside of the operational volume. In order to reduce the size of the elevation data, the grid resolution of processed data is set to 20 m.

For the population density data, the 2019 European Commission global human settlement layer is used [12]. Here, data from 2015 in a grid of 250 m are used. For each square, the number of inhabitants in that 0.125 square kilometer area is given and has to be converted to inhabitants per square kilometer. Since the georeference is given in latitude and longitude, it has to be converted into UTM32U coordinates. To account for the width of the FG, CV and GRB in the obstacle calculation, the processed population number for each 250 m by 250 m square is set to be the highest of the neighboring squares reachable

with half of the width of the FG, CV and GRB, as shown in Figure 1. The orange squares are used together with the green square to calculate the processed population density of the green square. This results in an over-estimate of the overflown population and thus is conservative.

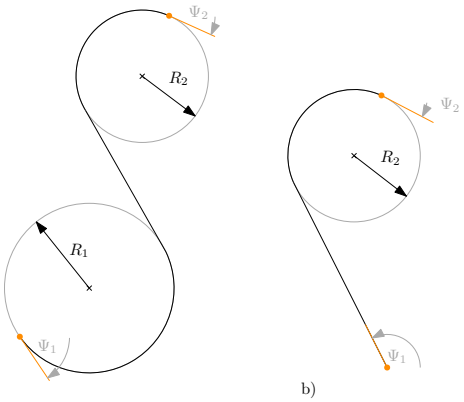


a) b)

Figure 1. Selection of used grid cells for GHSL processing depending on the width of the FG, CV and GRB. (**a**) FG, CV and GRB values are causing the selection of a square grid pattern for population density calculation. (**b**) FG, CV and GRB values lead to a non-square pattern since the outer most corners cannot be reached with those values

The ground infrastructure and type of ground area is derived from the Web Map Service (WMS) of the 'Digitale Plattform Unbemannte Luftfahrt (dipul)' by the Federal Ministry of Digital and Transport of Germany. The WMS enables generation of the image of a map showing the areas covered by the relevant infrastructure and ground areas. For both sets of infrastructure, an image is generated and is interpreted as grid data referenced in latitude and longitude. These images are converted to 2D Boolean arrays indicating the presence of an infrastructure element at the given grid point based on a color comparison with the background color. After converting the georeference to UTM32U, similar processing like the height processing is executed. Here, a grid point is considered covered if any grid point in the vicinity is covered. The vicinity for the obstacles includes both the operational volume and the GRB; for the risk calculation, only the operational volume is considered.

The connector function is set up such that the path between two configurations consists of three parts: an arc with constant bank angle and constant change in the path angle, a straight line in 3D space, and another arc like the first. The boundary conditions are the path and the bank and course angles of the start and end configurations. Since the initial and goal configuration of the planning problem are set to have zero bank, the first arc is not possible (either just not defined or because the problem would become underdefined). Figure 2 shows the horizontal path components of both cases.



a) b)

Figure 2. Distinction of two cases in the horizontal path calculation. (**a**) arc-straight-arc path segment (**b**) straight-arc path segment

In contrast to other RRT or RRT* algorithms, the goal configuration is a specific configuration rather than an arbitrary configuration inside of a goal region. Since the configuration space may be rather big, this ensures that a goal configuration at a given location is found. Relying on chance may result in no configuration being sampled in the vicinity (horizontal position error smaller than 50 m) of the specified goal configuration.

Collision detection relies on checking configurations on a path piece-by-piece since the underlying data are given in a grid format rather than a shape file. Cost calculation is done in the same fashion. To reduce the computation time, both cost calculation of a path segment and collision detection are executed at once. Therefore, in contrast to the suggestion in [4], no additional collision check function is needed.

To accelerate the collision check and path cost calculation, grid data are reduced to the segment needed based on a rectangular boundary around the query points. Therefore, the nearest neighbor interpolation only uses data points relevant to the calculation at hand.

The path planning problem is considered to be a mostly horizontal problem. This is due to the size of each dimension of the configuration space. Because of the stated problem of long-distance flights of multiple kilometers and the small altitude spectrum, horizontal position and course are the main variables of the path planning problem, and all other variables either describe dynamics of the path or are used to optimize the horizontal path in the vertical domain. Therefore, the nearest neighbor (see Algorithm 1) is calculated regarding the horizontal position and the course angle. For calculation of the set of near configurations (see Algorithm 1), the three-dimensional position and the course angle are used.

Algorithm 1 Transition-based RRT* (T-RRT*) [4]

$\mathcal{G} \leftarrow initGraph(q_{init})$

while not $stoppingCriteria(\mathcal{G})$ **do**
  $q_{rand} \leftarrow sampleRandomConfiguration(\mathcal{C})$
  $q_{near} \leftarrow findNearestNeighbour(\mathcal{G}, q_{rand})$
  $q_{new} \leftarrow extend(q_{near}, q_{rand})$

  if $q_{new} \neq null$ **and** $transitionTest(\mathcal{G}, c(q_{near}), c(q_{new})$ **then**
    $addNewNode(\mathcal{G}, q_{new})$
    $n \leftarrow numberOfNodes(\mathcal{G})$
    $Q_{near} \leftarrow nodesInBall(\mathcal{G}, q_{new}, \gamma(log(n)/n)^{1/d}$
    $q_{par} \leftarrow node_m inCost_{init}(q_{new}, q_{near}, Q_{near}, c_p)$
    $addNewNode(\mathcal{G}, q_{par}, q_{new})$

    for each $q_n \in Q_{near}$ **do**
      $\pi \leftarrow pathInSpace(q_{new}, q_n)$
      if $cost_{init}(q_{new}) + c_p(\pi) < cost_{init}(q_n)$ **and** $isCollosionFree(\pi)$ **then**
        $removeEdge(\mathcal{G}, parent(q_n), q_n)$
        $addNewEdge(\mathcal{G}, q_{new}, q_n)$
      end if
    end for
  end if
end while
return $\mathcal{G}$

As explained by Devaurs, Simeon and Cortès, a spatial search radius in the form of a ball is assumed for determining the set of near nodes. Due to its shrinking, problems for the expansion of the tree arise. From a certain number of generated nodes, the ball used has a smaller radius than the distance d_{max} used for the expansion. Therefore, a new node at a distance of d_{max} to the nearest node cannot be connected to the tree. To address this issue, the parameter d_{max} is adjusted over the run time as follows, with the initial maximal

step size $d_{max,0}$, a parameter γ as defined in [4], the number of generated nodes n, and the number of dimensions used for the ball d (here: 3).

$$d_{max} = min\{d_{max,0}, r\}$$
$$r = \gamma \cdot \left(\frac{log(n)}{n}\right)^{\frac{1}{d}}$$

Another difference from the original algorithm is the reduction of inspected nearby nodes for the rewiring. If no feasible path from a near configuration $q_i \in Q_{near}$ to a new configuration q_{new} exists, it is assumed that the path from q_{new} to q_i is also not feasible. This is meant to improve the execution time of the algorithm.

Due to the random nature of the algorithm, three inefficiencies emerge from the algorithm, which we try to remove. The first two are depicted in Figure 3 and deal with the combination of the sampled course and bank angle. In Figure 3a, one can see that the bank angle is in the wrong direction for the most efficient turn. In Figure 3b, the turn can be made without flying a full circle first. Both inefficiencies are solved by placing a new node at the red point and using this instead if the new path is valid. Invalidity can arise from violations of the maximum path angle or changes to the path angle.

The third inefficiency can be seen in Figure 4, where the path angle is suboptimal. This optimization is only applied as post-processing of the final path.

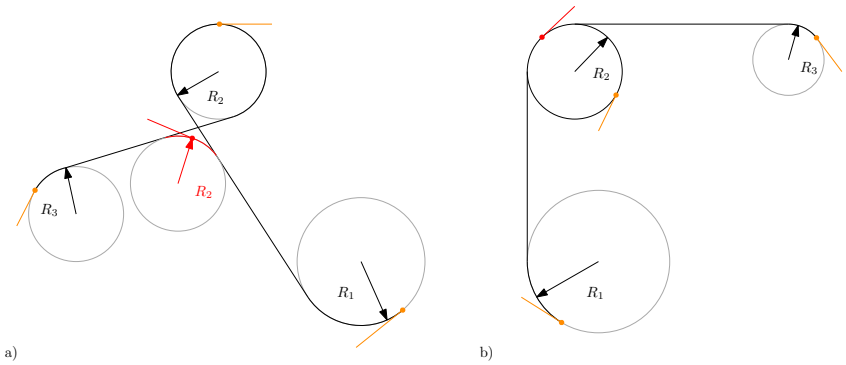


Figure 3. Horizontal path inefficiencies due to random path buildup. (**a**) wrong bank angle direction (**b**) flight includes an additional full circle

Figure 4. Vertical path inefficiency due to random path buildup.

3. Results

The test case of this work is to connect three hospitals in eastern Germany by fixed-wing UAV routes. The area is mostly rural and lies at the border region between Saxony, Saxony–Anhalt and Thuringia. For evaluation, we try to find paths from Altenburg to Zeitz and Naumburg.

The implementation of the algorithm results in two types of information: the final path (if found) together with all nodes of the tree, and information regarding the performance of the algorithm itself. In the following, both will be assessed, starting with the path information.

3.1. Analysis of the Path-Related Results

Shown in Figure 5 is the base map used for the evaluation of the horizontal flight path. Areas in black indicate no-fly zones due to high populations or restricted areas drawn from the dipul data set. Areas outlined in grey are areas associated with high risks (e.g., power lines). The green and orange areas are part of the collision-free configuration space. The color decodes population density as shown with the color bar at the right.

As one can see, the region of interest for the case study contains many small and a few large obstacles. Therefore, the path planning algorithm has to find paths around many obstacles.

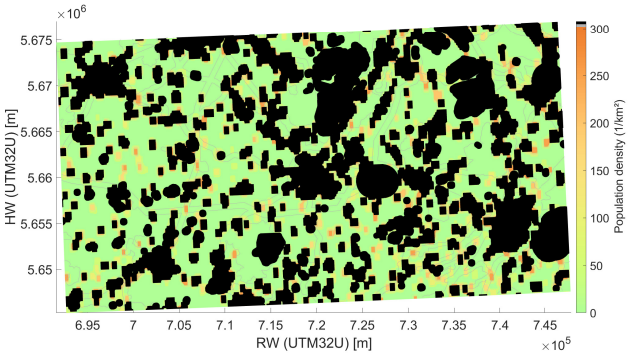


Figure 5. Horizontal obstacle map and cost function representation for $h_{max} = 100$ m and $V = 80\ \frac{\text{km}}{\text{h}}$.

However, changing the parameters of the operation (e.g., the velocity) will increase the CV and GRB, resulting in larger obstacles.

Shown in Figure 6 is the resulting horizontal path after sampling 1200 nodes for a velocity of 80 km/h and a maximum altitude of 100 m. Figure 6 only shows the area between Altenburg and Zeitz.

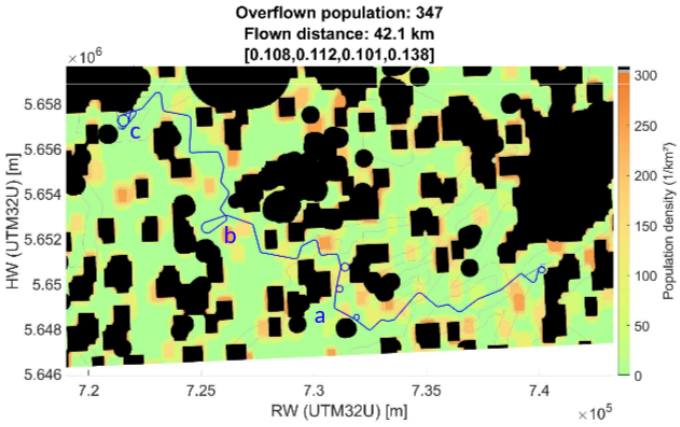


Figure 6. Found path between Altenburg and Zeitz for 1200 nodes and $d_{max,0}$ of 2.5 km.

The path found mitigates highly populated areas as well as infrastructure areas. This includes crossing, e.g., highways and power lines mostly perpendicularly. However, one can see a few inefficiencies that reveal issues with the proposed algorithm.

3.1.1. Residual Loops in the Flight Path

As one can see in Figure 6, at (a), there are still loops remaining in the path. The optimization introduced above is, therefore, not able to remove all inefficiencies. This is assumed to be due to the feasibility constraint of the optimized path. By reducing the distance flown in a horizontal arc, the change of path angle has to higher than before to match the boundary constraints of the connector function. This, then, increases the necessary load factor into an unfeasible regime. Multi-node loops such as the one at (b) of Figure 6 will be considered in Section 3.1.2.

3.1.2. Graph Structure and Re-Wiring

The issues resulting from the graph structure and re-wiring process can be seen in the unnecessary loops or jiggly lines as well as detours: for example, around the end in the northeast of Figure 6. The loop is constructed from tree nodes (two intermediate path segments), whereas a direct connection to the subsequent path would be better suited. Even

though the true reason for this behavior cannot be stated with certainty, a contributing circumstance is that re-wiring only considers a certain node but nothing downstream. To connect the nodes after the loop at (b) to a better-suited parent node, this node would need to be newly sampled.

A related issue arises at the goal configuration (seen at (c)): Since the goal node is a single node, it can only have a single parent. Once a path is found and a node is sampled, connecting to the goal in a more efficient way, the first connection is removed and lost. Together with the fact that sampling a node able to connect to the goal configuration is hard, few chances arise to optimize the last section of the path.

The issue of inefficient optimization of the path due to only local re-wiring and forgetting former possible connections is further intensified by shrinking of the search radius for re-wiring. With that, newer configurations have to be even closer to an optimizable region to connect to the relevant nodes. This can be seen best at the goal configuration, as newer samples have to be closer and closer to it to connect.

Another problem arising from the spatial search volume and step size is the trade-off between fast expansion and the ability to explore more restricted areas.

3.1.3. Difficulties with the Cost Function

Figure 7 shows the path profile (blue), processed ground level (brown) and minimum and maximum heights (red). Having set $\bar{h}$ of the cost function to the mean of the extreme heights above the ground, the algorithm is capable of finding a path reasonably close to the chosen desired height. However, one can also see that the height changes quite a lot without a comprehensive reason (see, for example, around 19 km). The propulsion cost parameter was intended to minimize climb and decent maneuvers. But the linearized definition introduces quasi path-independence in the sense that this parameter is only dependent on the height difference between a given configuration and the initial configuration. As a result, the algorithm prefers nodes with low altitudes with regard to the propulsion cost, and for the path between the initial and goal configuration, this cost parameter does not contribute in an meaningful way. This demonstrates that finding a good cost function is rather hard.

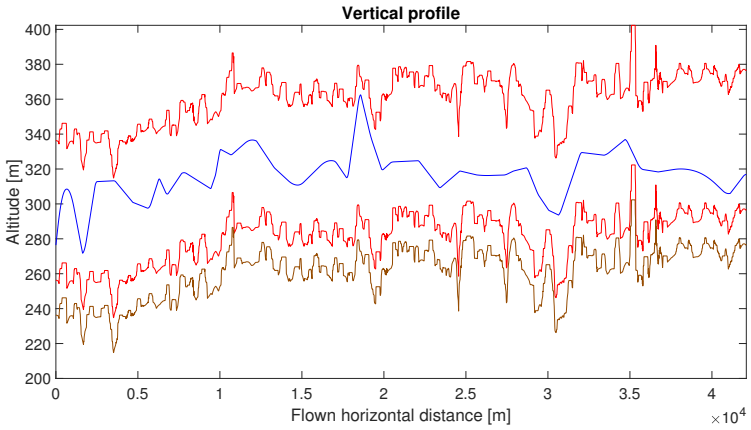


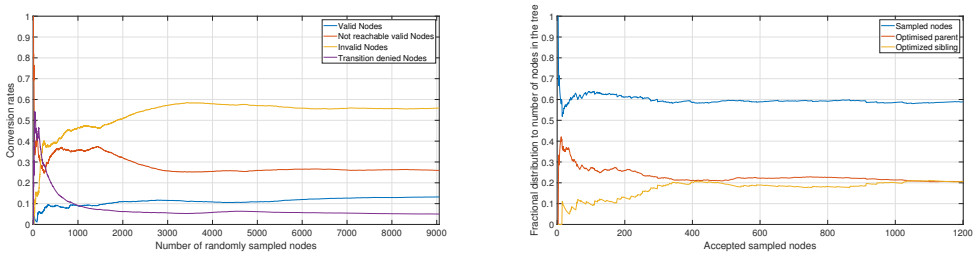
Figure 7. Vertical profile (blue) of the path shown in Figure 6. Brown shows the ground (elevation), both red lines are the limits of the specified height band.

3.2. Analysis of the Performance-Related Outputs

Figure 8a shows the percentage distribution of the sampled configurations regarding their acceptance. One can see that the distribution converges after around 4000 sampled nodes. The majority of the samples are discarded upon sampling (55%, yellow), followed by non-reachable nodes from known ones (25%, red), valid and accepted samples (15%, blue) and samples failing the transition test (5%, purple). Discarding samples sampled in obstacles does not waste too much computational expense, but the 25% of samples not able to be connected to any existing node due to blocking obstacles does. This is the

issue of the described trade-off between exploration speed and the ability to navigate obstacles. Not only does the inability to navigate obstacles reduce exploration, it also wastes computational effort. Overall, the conversion rate of 15% is rather small.

Figure 8b shows the percentage distribution of the origins of the nodes in the tree. Due to the optimization, nodes that are not sampled directly are also added into the tree. After convergence, 60% of the nodes in the tree are actually sampled, and 20% each are additional nodes from optimizations in path finding and re-wiring. The number of nodes in the tree is higher than the number of sampled ones by a factor of 1.66, which lets the search radius of the T-RRT* algorithm shrink even faster without contributing too much to the exploration of the area.



(**a**) (**b**)

Figure 8. (**a**) Percentage distribution of the validity and reasons for discarding sampled nodes. (**b**) Percentage distribution of the origins of nodes in the tree.

4. Discussion

In conclusion, the presented algorithm is capable of finding an optimized path for at least a goal configuration in the middle of the configuration space. However, it was shown that the implementation still has deficiencies regarding its performance. The computation of path costs is expensive, while only one calculated path segment at a time really contributes to the solution. Additionally, the computational expense of the cost calculation barely depends on the number of query points. With more efficient data storage and access as well as another data structure for the nodes, this efficiency could be increased. The performance of the algorithm might be improved by usage of another programming language and a more efficient cost calculation. More nodes in the tree would reduce the randomness of the solutions found, which then results in more reliable and repeatable results. Sampling more configurations would also result in finer paths able to follow straighter lines, which otherwise are only producible by logic-based optimization.

Furthermore, the algorithm proposed has trouble navigating an environment filled with obstacles. More-urban spaces with more no-fly zones or operations with a higher velocity and/or height might not be compatible with this algorithm. Finding narrow passages between obstacles is not a strength of the proposed algorithm. It might be worth exploring the usage of RRV [7].

However, this work together with Ortlieb and Adolf can be seen as a first step towards the usage of path planning algorithms in the risk management procedure for the application process of operation permits.

Author Contributions: Conceptualization, J.H.; methodology, J.H.; software, J.H.; validation, J.H.; formal analysis, J.H.; investigation, J.H.; writing—original draft preparation, J.H.; writing—review and editing, M.U.d.H.; visualization, J.H.; supervision, M.U.d.H. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Data sharing is not applicable to this article.

Acknowledgments: Special thanks to Globe UAV GmbH for information regarding the application process of operation permits and possible mission goals of UAV operations.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. EASA, Easy Access Rules for Unmanned Aircraft Systems. September 2021. Available online: https://www.easa.europa.eu/downloads/110913/en (accessed on 12 July 2022).
2. Ortlieb, M.; Adolf, F.M. Modular Modelling of Ground and Air Risks for Unmanned Aircraft Operations Over Congested Areas. In Proceedings of the Digital Avionics Systems Conference, San Antonio, TX, USA, 11–15 October 2020 . [CrossRef]
3. Benders, S.; Schopferer, S. A Line-Graph Path Planner for Performance Constrained Fixed-Wing UAVs in Wind Fields. In Proceedings of the 2017 International Conference on Unmanned Aircraft Systems (ICUAS), Miami, FL, USA, 13–16 June 2017; pp. 79–86. [CrossRef]
4. Devaurs, D.; Simeon, T.; Cortés, J. Optimal Path Planning in Complex Cost Spaces With Sampling-Based Algorithms. *IEEE Trans. Autom. Sci. Eng. Inst. Electr. Electron. Eng.* **2016**, *13*, 415–424. [CrossRef]
5. Schopferer, S.; Benders, S. Minimum-Risk Path Planning for Long-Range and Low-Altitude Flights of Autonomous Unmanned Aircraft. In Proceedings of the AIAA Scitech 2020 Forum, Orlando, FL, USA, 6–10 January 2020. [CrossRef]
6. Gammell, J.D.; Srinivasa, S.S.; Barfoot, T.D. Informed RRT*: Optimal Sampling-based Path Planning Focused via Direct Sampling of an Admissible Ellipsoidal Heuristic. In Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems, Chicago, IL, USA, 14–18 September 2014. [CrossRef]
7. Tahirovic, A.; Ferizbegovic, M. Rapidly-Exploring Random Vines (RRV) for Motion Planning in Configuration Spaces with Narrow Passages. In Proceedings of the IEEE International Conference on Robotics and Automation (ICRA), Brisbane, Australia, 21–25 May 2018. [CrossRef]
8. Luders, B.D.; Kothari, M.; How, J.P. Chance Constrained RRT for Probabilistic Robustness to Environmental Uncertainty. In Proceedings of the AIAA Guidance, Navigation, and Control Conference, Toronto, ON, Canada, 2–5 August 2010.
9. Federal Ministry for Digital and Transport of Germany. Anleitung für den Web Map Service (WMS). Available online: https://www.dipul.de/homepage/de/hilfe/anleitung-web-map-service-wms/ (accessed on 13 July 2022).
10. Thüringen, F. Download Höhendaten. Available online: https://www.geoportal-th.de/de-de/Downloadbereiche/Download-Offene-Geodaten-Th%C3%BCringen/Download-H%C3%B6hendaten (accessed on 1 June 2022).
11. Anhalt, S. Kostenfreies Digitales Oberflächenmodell mit einer Gitterweite von 2 m (DOM2)—Landesweit. Available online: https://www.lvermgeo.sachsen-anhalt.de/de/dom2-landesweit.html (accessed on 1 June 2022).
12. Schiavina, M.; Freire, S.; MacManus, K. GHS Population Grid Multitemporal (1975, 1990, 2000, 2015) R2019A. European Commission, Joint Research Centre (JRC). 2019. Available online: http://data.europa.eu/89h/0c6b9751-a71f-4062-830b-43c9f432370f (accessed on 1 June 2022).