Advancements for A* and RRT in 3D path planning of UAVs

Zammit, Christian; van Kampen, Erik-jan

DOI
10.2514/6.2019-0920

Publication date
2019

Document Version
Final published version

Published in
AIAA Scitech 2019 Forum

AIAA SciTech Forum
7-11 January 2019, San Diego, California
AIAA Scitech 2019 Forum

10.2514/6.2019-0920

Advancements for A* and RRT in 3D path planning of UAVs

C. Zammit*† and E. van Kampen‡

Delft University of Technology, Delft, 2629HS, The Netherlands

Advancements in Unmanned Aerial Vehicles (UAVs) design, actuator and sensory systems and control are making such devices financially available to a wide spectrum of users with various demands and expectations. To mitigate with this ever increasing demand robust, efficient and application–specific path planning is important. This paper presents advancements over the A* and the smoothing algorithms presented in,[1] utilising the same test scenarios. Analysis of results in[1] showed a ripple in path length as the resolution changes for all scenarios considered and less than 0.1% path length improvements after certain amount of smoothing iterates. To attenuate the path length ripple, the A* ripple reduction algorithm was developed. Results show a reduction of more than 46% in terms of standard deviation with respect to the original A* algorithm without any increase in the mean path length for all scenarios. Secondly, the smoothing algorithm developed in[1] was improved to stop smoothing based on the rate of smoothing of previous iterates. Results show more than 10 multiple less path smoothing time maintaining a path length reduction especially for simple scenarios. These advancements further portray the discussed path planning algorithms as candidates to the realisation of online 3D UAV path planning.

I. Introduction

The utilisation of Unmanned Aerial Vehicles (UAVs) into a wide spectrum of civil and military applications requires robust, reliable, efficient and coordinated path planning. UAVs, or more commonly drones, vary in size, shape, weight, actuators, dynamics and are equipped with numerous and different sensory systems with different levels of accuracy, precision and reliability that provide a vast range of data required to define the UAV state and the environment in which it resides. Different control algorithms rely on this information to guide the UAVs to their destination in view of a predefined path or through a real-time path planning strategy. With advances in control algorithms and cost-reduction in UAV manufacturing have made general purpose UAVs financially available to a larger part of the public increasing the complexity of path planning algorithms that will ultimately create a bottleneck to the growth of the autonomous UAV market.

Path planning is the process of automatically generating feasible and optimal 2D[2,3] or 3D[4,5] paths to a predefined destination in view of weather,[7,8] sensory and model constraints[9,10] and uncertainties.[11,12] Different path planning algorithm were proposed each with their strongholds and drawbacks. An extensive overview of the state-of-the-art path planning algorithms was presented in.[1] The A*, a graph–based algorithm, and the Rapidly–Exploring Random Trees (RRT), a sampling–based algorithm are the two most utilised path planning algorithms and were selected as key candidates for the context of 3D UAV path planning.

The twofold aim of this paper is the attenuate and possibly eliminate the path length ripple effect with changes in resolution for the A* algorithm and the increase in path length reduction per computational time ratio of the smoothing algorithm since after certain iterates the path length reduction was less the 0.1%.

As the path planning algorithms are analysed in the context of 3D UAV path planning, the path length and the computational time will be considered as the performance parameters. The UAV fuel capacity and

*Ph. D. Candidate, Control & Simulation, TU Delft Aerospace Engineering, Kluyverweg 1, Delft, The Netherlands.

†Lecturer, Gozo Campus, Malta College of Arts Science and Technology (MCAST), J.F. De Chambray Street, Ghajnsielem, Gozo, Malta.

‡Assistant Professor, Control & Simulation, TU Delft Aerospace Engineering, Kluyverweg 1, Delft, The Netherlands, and AIAA Member.

efficiency which determines the flight time is one of the limitations of UAV autonomy. Therefore the shorter the path to reach a goal or a set of goals the longer the flight range. Moreover, computational power is very limited and a percentage of it is used for control. Therefore, the path planning algorithms shall limit the computational demand as low as possible.

The paper will be organised as follows. Section II will introduce and briefly explain the path planning and smoothing algorithms and the test scenarios. Section III will define, develop, test and assess the A* ripple reduction algorithm followed by Section IV which presents, tests and assesses an advancement to the smoothing algorithm developed in.[1] Section V concludes the paper by highlights the benefits of the proposed algorithm in view of online 3D UAV path planning.

II. A*, RRT, RRT without step size constraint, MRRT and Smoothing Algorithms and Test Environment

A. Introduction

This section will briefly explain the path planning and smoothing algorithms and the test scenarios utilised to assess the performance of the mentioned algorithms. The A* and RRT are the two most utilised algorithms for the graph–based and sampling–based methods, respectively.

Graph–based methods segment the space into an occupancy grid with obstacles defined as inaccessible grid points.[14, 15] Sampling–based methods unevenly selects a set of points from the configuration space, creating a path by connecting these points.[15, 16] Oppositely to graph–based methods which offer no guarantee of solution,[17] sampling–based methods are probabilistically-complete.[15]

B. The A* Algorithm

The A* algorithm constructs an optimal path based on an evaluation function $f(n)$ that calculates the actual cost of an optimal path constrained to pass through n, from a point x_{init} to the goal node of n, $x_{goal} \in \mathbb{R}^M$.[18, 19] n is any node and x_{init} is the starting node in the M–Dimensional available space such that n, $x_{init} \in \mathbb{R}^M$. This evaluation function $f(n)$ is the summation of the actual cost from x_{init} to a node, n $(g(n))$, and the actual cost from n to the goal point of n, $(h(n))$, where f, g, h: $\in \mathbb{R}^M \to \mathbb{R}$:

$$f(n) = g(n) + h(n) \tag{1}$$

C. The RRT Algorithm

The RRT algorithm grows trees of feasible trajectories by randomly planting seeds ultimately growing a tree that interconnects the start and goal points.[20–22] Seeds are only considered if they lie on an obstacle free point. A point a predefined distance from the nearest node is selected if the direct path to the latter does not collide with an obstacle. Paths generated by RRT are not optimal.[23, 24]

D. The RRT Algorithm without step size constraint

In this variant the step size limitation is eliminated so that a tree branch is connected directly between the nearest node and the randomly generated node if an unobstructed path exists between the two.

E. The MRRT Algorithm

In MRRT, an arbitrary number of RRTs are simultaneous expanded, with two starting from the start and end nodes and the remaining starting from predefined or randomly–selected nodes.[25] As for the RRT the algorithm stops when the start and goal nodes are interconnected by a single tree.

F. The Smoothing Algorithm

A basic smoothing algorithm is utilised by all these algorithms. The algorithm randomly selects two path points and consequently selects two points on the lines connecting these path points with their respective next path point. If an interconnection is possible without collision with obstacles then intermediate points between these two path points are eliminated. Further details of this algorithm are provided in Section IV.

G. The Experimental Scenarios

The experimental space is defined as a generic cube of 1x1x1 with centre (0,0,0) and limits [-0.5 $\rightarrow$ 0.5, -0.5 $\rightarrow$ 0.5, -0.5 $\rightarrow$ 0.5]. The generic environmental space is normalised to arbitrary units so that the test scenarios it can be exported to any environmental space. These test scenarios were originally developed by Clifton *et. al.*[26] and made available online in.[27] For all tests the start and goal nodes are defined at [0,-0.5,0] and [0,0.5,0], respectively. Three different obstacle scenarios, illustrated in Figure 1 are considered. Scenario 1 consists of two Y-Z obstacle planes with 0.2x0.2 square opening later referred to as windows. Scenario 2 consists of three Y-Z with 0.2x0.2 windows and two X-Y planes without windows while Scenario 3 is similar to Scenario 2 but with five Y-Z planes instead of three.

Reference is made to our previous work[1] for further understanding of the working principle of these five algorithms (Subsections B to F) and the experimental scenarios.

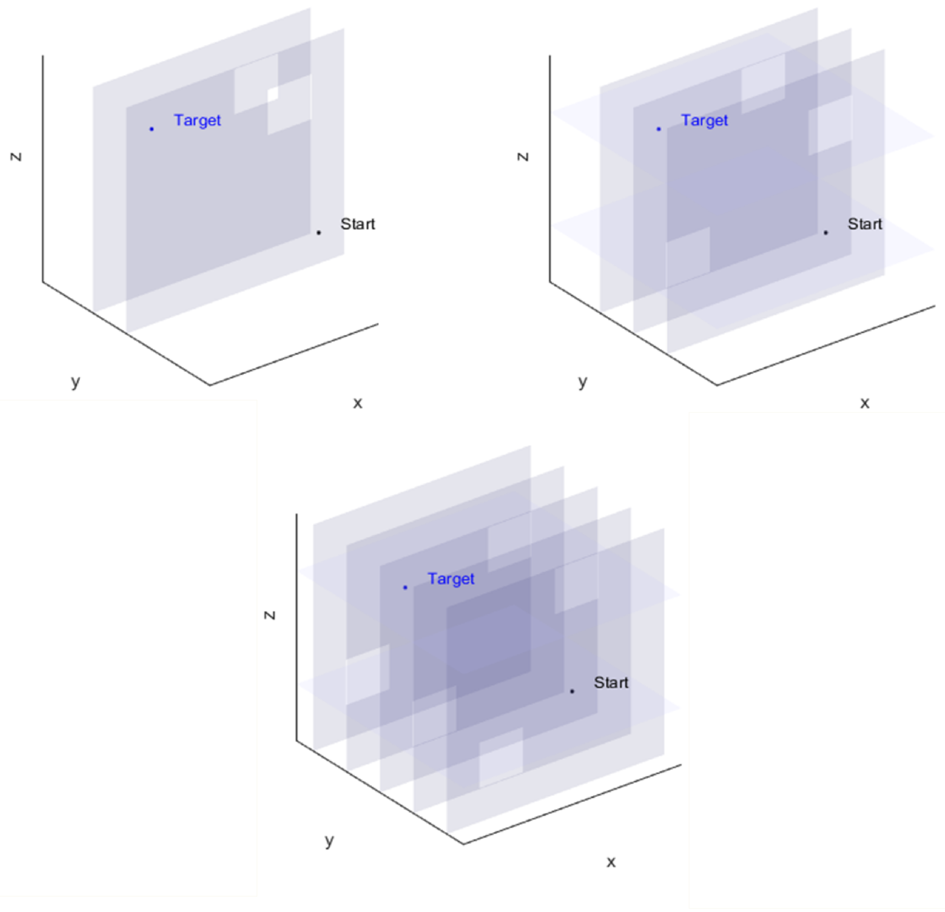

Figure 1. Obstacle scenarios: (a) First Scenario (b) Second Scenario and (c) Third Scenario modified from,[1] consisting of obstacle planes in the Y-Z with windows as openings and X-Y planes for scenarios 2 and 3 with no openings.

III. A* Ripple Reduction Algorithm

A. Introduction

In this section the cause of the ripple in the path length for the A* algorithm will be identified in Subsection B. This will lead to the definition of the A* ripple reduction algorithm in Subsection C. Results will be presented in Subsection D. Following the ripple reduction and the mean path length and computational time for the A^*_R will be analysed with respect to the A* algorithm in Subsections E and F. A conclusion summing up the results will finish this section.

B. The cause of the ripple in path length for the A* algorithm

The A*'s graph–based nature creates situations in which a minute increase in resolution which theoretically shall slightly decrease the path length effectively generates significantly longer or significantly shorter unsmoothed paths and vice versa. This characteristic is envisioned in Figure 2(a), cited from our previous work.[1] The illustrated results are the mean of 100 runs for each particular resolution and scenario.

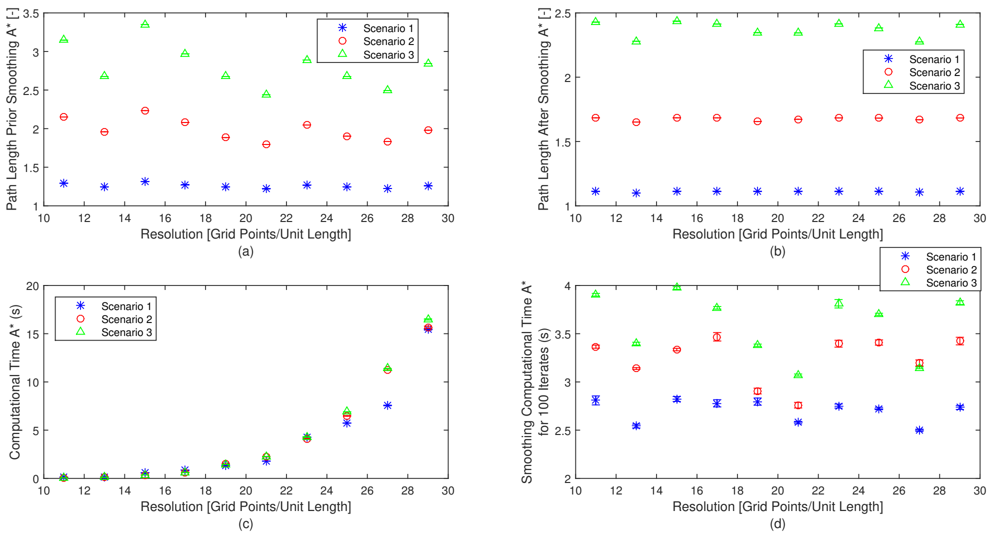

Figure 2. A* Results with 95% Confidence Interval: (a) Path Length Prior Smoothing (Generic length with respect to a unity square box) (b) Path Length After Smoothing (c) Computational Time and (d) Smoothing Computational Time for 100 Iterates[1]

In A*, the environment space is discretised into a finite number of evenly–spaced points based on the selected resolution. As the resolution increases or decreases slightly it can result that a whole plane of points which was previously available for the path planner to utilise for path construction, will now reside exactly on one of the obstacle planes, leaving available only the points in the windows. This reduces computational time as fewer points are computed and also creates longer or shorter paths than the previous resolution test. This effect creates a ripple in the path length versus resolution plot, when resolution is increased gradually from 11 to 29 in all 3 dimensions.

From the standard deviations of the A* algorithm in Table 1, it can be concluded that the largest peak-to-peak path length ripple is experienced at the lower resolutions. This confirms the hypothesis that as the resolution decreases fewer possible path point locations are available and therefore a slight increase or decrease in resolution will significantly change the position of possible path point locations leading to large variants in path length. Furthermore, the percentage reduction in open nodes resulting when a whole obstacle plane coincides with a plane of open nodes is significantly larger for lower resolutions than for higher ones, leading to a higher peak–to–peak ripple for low resolutions.

The ripple in path length is slightly attenuated by the path smoothing algorithm as illustrated in 2(b). The standard deviation in length after path smoothing, tabulated in Table 1, shows a significant reduction for all scenarios. This reduction is possible since during path smoothing the selected random points on the line connecting the randomly selected nodes and their respective next node are not discretised and reside anywhere in the environmental space. On the other hand, this behaviour is not exhibited in the considered sampling–based algorithms since branch and tree seeds are randomly generated without any discretisation. This further confirms that the generated ripple is a counter effect of discretisation. Therefore, only a ripple reduction algorithm for A* is considered.

Although the cause of the path length ripple was theoretical explained it is the scope of this paper to attenuate it so as to be negligible. In our previous work,[1] the peak-to-peak path length ripple was reduced by considering a solid cube as an obstacle. In this situation, changes in available nodes due to changes in resolution, will only marginally effect the path length. But in 3D path planning scenarios of UAVs the planner has no control over the obstacle shape, size and position and therefore the option of re-modelling obstacles by cuboids instead of planes is not realistic.

C. Definition and explanation of the A* ripple reduction algorithm

To mitigate this situation, the graph points, including the start and goal points, are randomly shifted in all 3 dimensions by a distance varying between 0 and half the distance between adjacent graph points, leaving unaltered the obstacle planes. Through this algorithm the difference in graph point positions that ultimately leads to a longer or smaller path length by slight changes in resolution is averaged out. This is because the 100 different random shifts for each situation (same scenario and resolution) will lead to different path lengths. The mean path length of these 100 instances for each particular situation will reduce the path length ripple as a function of the resolution. Previously only one graph of points was considered for 100 instances for each situation that yielded the same path.

D. Results

The test scenarios defined in Section II, Subsection G were set–up. The considered resolution started from 11 to 29 in all 3 dimensions in steps of 2. Each run is repeated for 100 times for each particular case (resolution and obstacle scenario) each time randomly changing the offset shifting value.

Figure 2 shows that the limits of the 95% confidence interval is 0 for the unsmoothed path length. This is because the A* is run for 100 times with the same starting and goal nodes and the same open and closed nodes. Therefore the generated path will be the same always. The confidence interval is almost negligible but not 0 in the smoothing length since the interconnecting points between path segments selected to eliminate intermediate points by a single line are defined by a random function. Similarly the interval of the path generation time is less than 1% in all the situations. The interval increases exponentially in view of the exponential increase in path planning time. As the resolution increases more open nodes in the same space are available, increasing computational time. The 95% confidence interval is also small for smoothing time varying randomly from 0.3% to 1.2% and independent of resolution. This random behaviour is a direct consequence of the random selection process of points in randomly selected unsmoothed path point segments.

Figure 3 illustrates the simulation results of the A* ripple reduction algorithm denoted by A_R^*. From the figure it is clear that the path length reduces asymptotically as the resolution increases linearly. This confirms the theoretical analysis that path length is inversely proportional with resolution. From the results presented in[1] and illustrated in Figure 2 this path length vs. resolution relationship could not be directly concluded but was only partially confirmed with the single basic cube obstacle in the centre of the environment space. To further confirm this hypothesis the resolution was increased to 49 in steps of 2 and the same asymptotic relationship was visualised.

Table 1. Standard Deviation in path length for the A* with and without the ripple reduction algorithm

Standard Deviation in path length	Scenario 1	Scenario 2	Scenario 3
Prior Smoothing for A*	0.0294	0.1410	0.2844
Prior Smoothing for A_R^*	0.0158	0.0759	0.1476
After Smoothing for A*	0.0037	0.0126	0.0594
After Smoothing for A_R^*	0.0053	0.0181	0.0348

E. Ripple Analysis

In terms of confidence interval, Figure 3 and Figure 4 show asymptotic relationship with increase in resolution for both the smoothed and unsmoothed path lengths, excluding resolution 11. As the resolution increases the difference in path length between the shortest, second and third shortest paths decreases. By slight shifts in the environment the previous second or third shortest path option can become the shortest. Therefore, as the resolution increases the difference in the range of options decreases as illustrated by the confidence interval.

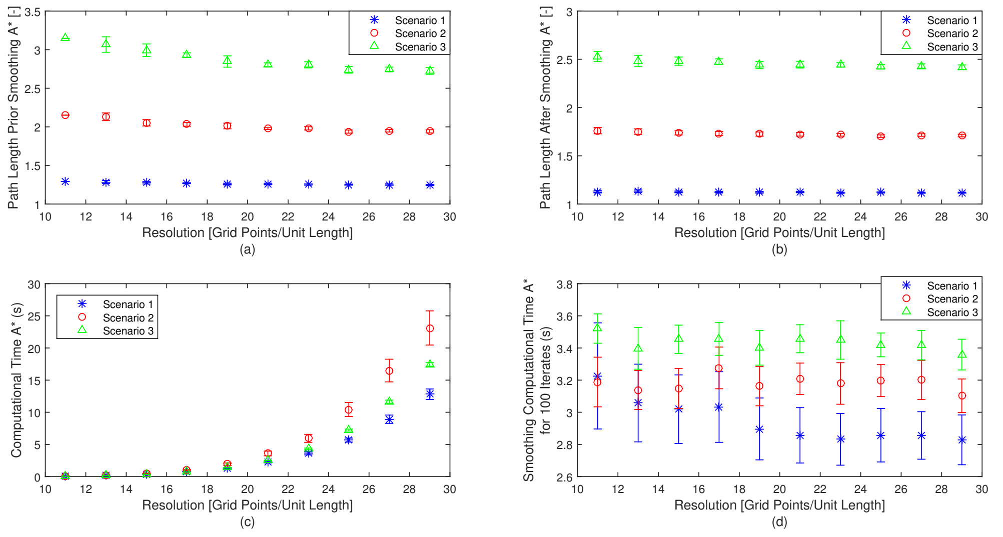

Figure 3. A* Ripple Reduction Algorithm (A^*_R) Results with 95% Confidence Interval: (a) Path Length Prior Smoothing (Generic length with respect to a unity square box) (b) Path Length After Smoothing (c) Computational Time and (d) Smoothing Computational Time for 100 Iterates

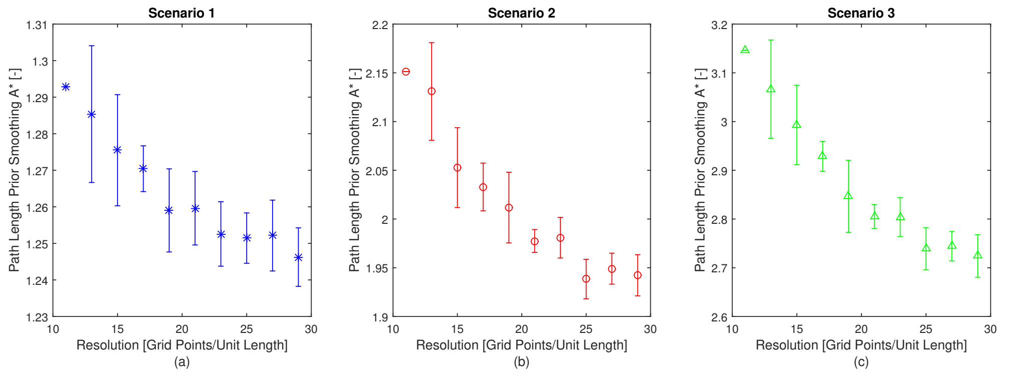

Figure 4. Zoomed Path Length Prior Smoothing for the A* Ripple Reduction Algorithm (A^*_R) with 95% Confidence Interval

Table 2. Mean values for all resolutions for the A* and A^*_R algorithms

Mean values	Scenario 1	Scenario 2	Scenario 3
Unsmoothed path length for A*	1.2586	1.9864	2.8154
Unsmoothed path length for A^*_R	1.2645	2.0167	2.8798
Smoothed path length for A*	1.1092	1.6749	2.3730
Smoothed path length for A^*_R	1.1227	1.7275	2.4570
Generating time for A* (s)	3.7783	4.2427	4.3869
Generating time for A^*_R (s)	3.6067	6.3297	4.5796
Smoothing time for A* (ms)	27.036	32.397	35.975
Smoothing time for A^*_R (ms)	29.463	31.802	34.332

Similar to path generation time results of A*, shown in Figure 2, the confidence interval increases exponentially as explained earlier. The A* path generation time for Scenario 2 is slightly smaller than Scenario 3, although the number of open nodes in the latter is smaller than in the former due to more obstacle planes. Oppositely, the path generation time for A_R^* in Scenario 2 is larger than that of Scenario 3. Theoretically, the generation time is inversely proportional to the number of open nodes and directly proportional to the path length. Due to the arbitrary placement of obstacles in the working space, obstacle planes are modelled using more than one grid plane, reducing the number of nodes and computations in A*. Oppositely, in A_R^* this advantage is neutralised and this explains the longer time for Scenario 2. This hypothesis was confirmed by the comparative analysis of each run for both A* and A_R^*. Only some instances A_R^* matched A* drifting significantly in the majority of runs for Scenario 2, therefore increasing the mean and confidence interval.

The confidence interval in the smoothing time is almost independent of the resolution. This is in line with theory since the smoothing algorithm neglects the graph based nature of the path generating algorithm. The confidence interval in smoothing time reduces as the obstacle occupying space increases (Scenario 1 to 3). This is because with fewer available smoothing options, as complexity increases, the amount of prospective smoothing improvements is limited even if the environment is shifted as in A_R^*. Although the A_R^* algorithm increases the confidence interval by the shifting effect as opposed to A*, A_R^* reduces the difference in smoothing time between Scenarios 1 to 3.

From Table 1 it can be deduced that the A_R^* algorithm presents a 46.3%, 46.2% and 48.1% path length ripple reduction in terms of standard deviation with respect to the A* algorithm. This significant reduction in unsmoothed path length ripple is re–confirmed with a different random seed generator. This improvement confirms that the ripple presented in[1] was a consequence of the possibility that one obstacle plane could reside exactly on one one whole plane of available points, leaving only available points in the windows, as explained in more detail earlier. Through the A_R^* algorithm these one–off situations are averaged out, providing a more realistic situation of the A* performance characteristics.

The ripple cannot be totally eliminated with the A_R^* algorithm, even with the smoothing algorithm, although the ripple is reduced by 3, 4 and 4 times for scenarios 1 to 3, respectively. From the results it is clear that the peak–to–peak ripple with and without the ripple reduction algorithm produces the same results, implying that smoothing is more effective on the higher peak–to–peak ripple situation. Also, this highlights the saturation of the smoothing algorithm that can smoothen and shorten the path up to a certain degree. Therefore, primarily in the case of the A_R^* algorithm, time is wasted in trying the smoothen a path which is already optimised for the smoothing algorithm considered. In this light, the number of smoothing iterates shall be accurately selected to make optimal use of time. This will be discussed in Section IV. From the test repetition, using different random seeds and from saturated reduction of the smoothing algorithm, it can be concluded that the peak–to–peak ripple will approach zero as the number of tests per test case scenario approaches infinity.

F. Mean path length and time analysis

From Table 2 it can be deduced that the A_R^* algorithm marginally increases the mean unsmoothed path length by 0.5%, 1.5% and 2.3% for scenarios 1 to 3, respectively with respect to the A* algorithm. This small mean increase is not exhibited throughout all resolutions considered. In fact the mean unsmoothed path length was smaller in 12 different resolutions and equal in another 3 instances for A_R^* algorithm when compared to the A* algorithm. In A* all 100 tests produced the same unsmoothed path length for each resolution since all parameters remains the same. Oppositely, different lengths resulted for the same resolution in the A_R^* algorithm due to the shifting of the environmental space and consequently resulting in different positions of obstacle planes with respect to the start and goal nodes. As in A*, obstacle planes were arbitrary defined at values sub–dividable by 10 of the environment space length. This created situations where the obstacle plane is located exactly on the available node planes as opposed to the A_R^* algorithm that considers a mean of 100 different situations per scenario.

In fact, the marginal mean increase can be explained in view of obstacle modelling in the A* algorithm. If an obstacle is a plane, as in all scenarios considered, a singular value defines the plane in one particular dimension. In such situations the obstacle discretising algorithm considers all points a predefined distance, in this case half the distance between nodes, from the obstacle plane in both directions as also being an obstacle. This criterion was introduced since obstacle planes that didn't reside exactly on available open nodes were neglected during the discretisation process. This creates a situation in which the majority of the obstacle planes were neglected as the probability of exactly residing on defined graph point was small.

The buffer width of the obstacle plane was selected as half the distance between the nodes since otherwise in the worst case scenario an obstacle plane just in the middle between two available nodes would not have been considered. But this quantisation process created a situation where one obstacle plane created two obstacle planes as illustrated in Figure 5. From an application's point of view, this is not a drawback but an additional required safety feature since, if only one obstacle plane is considered, the node interconnection path can overlap with the actual obstacle plane creating potential collisions. Furthermore, obstacle position and shape, UAV kinematics and dynamics are only know with a certain degree of uncertainty therefore this buffer feature is strongly encouraged or even increased.

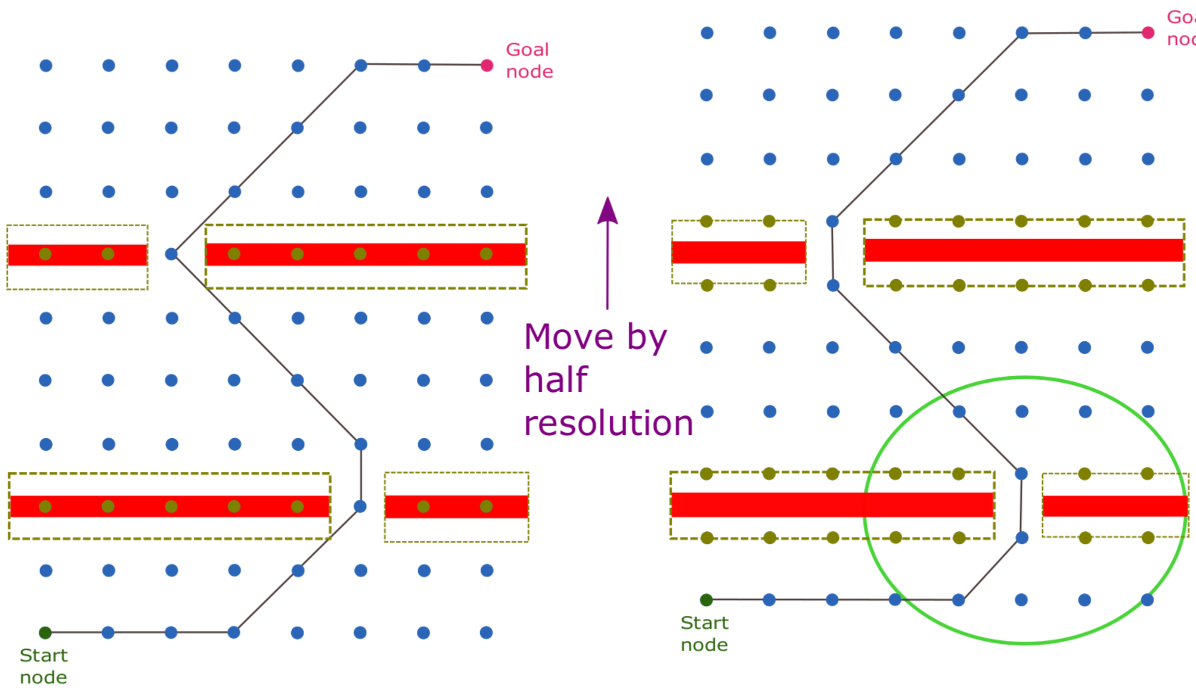

Figure 5. An example illustrating how the A* ripple reduction algorithm could effectively increase the path length prior smoothing due to quantisation in obstacle definition

Furthermore, obstacle plane windows effectively vary in size depending on the considered resolution and grid positioning. This effect is visually explained in Figure 6. Figure 6 (a) shows a typical example of an obstacle plane window with only four nodes available for the path planner. The area from which the UAV can pass (violet square) is much smaller than the real area (olive green square). By increasing the resolution we effectively increased the traversible area (violet square) (b). This is not always the case as it depends on where the new nodes will reside. In fact, by keeping the same resolution but offsetting the graph points by half the distance between available nodes the traversabile area reduced to less than the low resolution situation (a).

The situations explained by Figures 5 and 6 highlight the randomness of the path planning process in the A^*_R algorithm. Therefore a band of possible paths can be created. From an application's point of view, this implies that the path planner will not be able to generate the same path for the same scenario and resolution unless each test is repeated for infinite times each time changing the offset. In such scenario all paths shall converge to the same path points. In online systems were computational power and time are the bottleneck, each test scenario cannot be repeated for multiple times unless the UAV is stationary and waiting for the re–location of a new or updated goal point or the passing of a moving obstacle. Therefore, this introduces an element of uncertainty in the path point locations for the same test situation.

The path length reduction by the smoothing algorithm is 11.9%, 15.7% and 15.7% for A* and 11.2%, 14.3% and 14.7% for the A^*_R algorithm for scenarios 1 to 3, respectively. The original smoothing algorithm, explained in II Subsection F, makes use of points on the path segments interconnecting the unsmoothed path points. The performance of the smoothing algorithm is limited by the open nodes available during path construction. The availability of open nodes is a function of the resolution and obstacle definition algorithm. Therefore, the performance of the smoothing algorithm depends on how far the path segments are away from

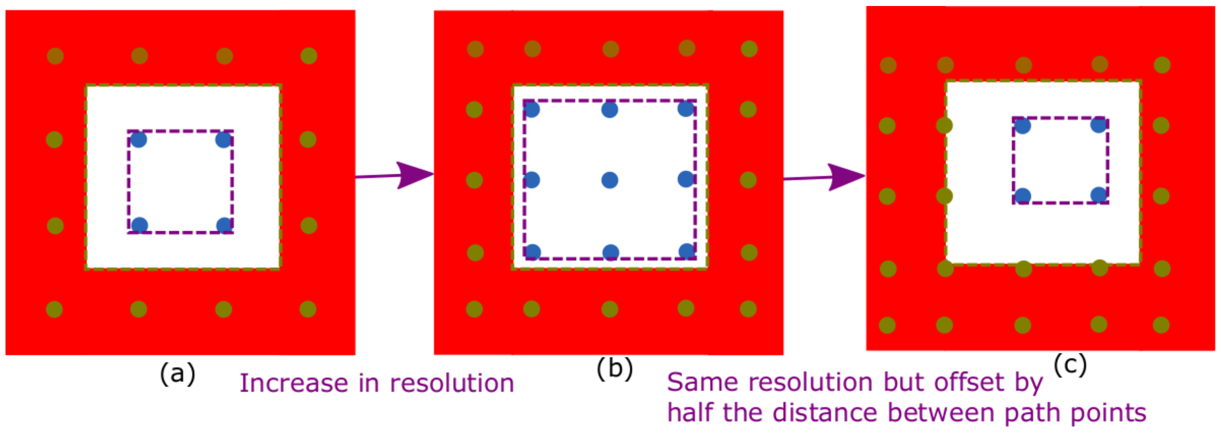
(a)
Increase in resolution
(b)
Same resolution but offset by half the distance between path points
(c)

Figure 6. An example illustrating how by increasing the resolution (a) to (b) the effective available window increased. By offsetting the graph nodes keeping the same resolution (b) to (c) a smaller available than (a) resulted

the actual obstacle plane. In other words, the uncertainty buffer that is considered by the obstacle definition algorithm will effect the path length reduction potential of the smoothing algorithm.

Furthermore, the mean smoothed path length reduces with resolution with the A^*_R algorithm as opposed to the original A*, which oscillates as a consequence of the ripple in the unsmoothed length. This results as a consequence of the inverse relationship between path length and resolution described earlier.

Interestingly enough, the path length after smoothing in the A* outperforms the A^*_R algorithm in all resolutions considered for the same number of smoothing iterates, as opposed to the path length prior smoothing which sees both options outperforming one another in different resolutions. This implies that the smoothing algorithm was more effective on the A* than A^*_R algorithm, reducing path length to less than that of the A^*_R algorithm after smoothing, even in situation when the unsmoothed path is longer than that of the A^*_R algorithm.

Since the path generated by an A* algorithm is almost optimal[18] and consequently few smoothing iterates are required (less than 100 in the considered scenarios as will be explained in Section IV), this implies that by shifting by equal distances only the start and goal positions, a longer optimal path will be created after smoothing. Therefore the reason for this longer optimal depends upon the start and goal node positions, with respect to the obstacle plane positions although the distance between the start and goal nodes remains the same. In the original A*, the goal and start nodes are symmetrical with respect to the obstacle plane window, but with shifting this symmetry is lost.

The path generation time was 4.5% shorter and 49.2% and 4.4% longer for the A^*_R algorithm with respect to the original A*. This shows that, apart from Scenario 2, the path generating time remains the same. The increase in path generation time for Scenario 2 is a consequence of the obstacle grid point estimation as explained earlier.

The path smoothing time is 9% longer, 1.8% shorter and 4.6% shorter for the A^*_R algorithm compared to the A* algorithm. This marginal difference is primarily due to the shifting of the environment and the creation of different paths as a consequence.

G. Conclusion

It can be concluded that the A^*_R algorithm is able to reduce the ripple in path length introduced by the original A*. The results based on different complexity 3D scenarios confirm that, for the A^*_R algorithm, the length and computational time will not be negatively compromised. Furthermore, the A^*_R algorithm's shifting nature provides a more robust test platform as each run will create a situation different from the previous, even for the same resolution, scenario and start and goal nodes as opposed to A* where for the same resolution, obstacle scenario and start and goal nodes, the same test would be repeated for 100 times.

IV. Smoothing Algorithm

A. Introduction

This section will initiate with the scope of improving the original smoothing algorithm after analysis of its drawbacks (Subsection B). This is followed by the definition of the new smoothing algorithm in Subsection C. This new smoothing algorithm is tested on the A* (Subsection D), RRT (Subsection E), RRT without step size constraint (Subsection F) and MRRT (Subsection G). Conclusions are then drawn based on the analysis of the results.

B. Scope of improving the original smoothing algorithm

The primary scope of the smoothing algorithm defined in II, Subsection F is to optimise the path produced by the considered sampling–based algorithms. Literature confirms that paths produced by the standard RRT, RRT without step size constraints and MRRT may not be optimal and require smoothing.[1, 22–24] In fact, the smoothed path length for RRT is between 50% and 54% of the unsmoothed path length for all considered scenarios. On the contrary, paths produced by the A* algorithm are almost optimal with smoothing only producing a marginal reduction in path length (less than 20%[1]).

The smoothing algorithm introduced in II Subsection F and explained in detail in[1] was repeated for 1000 times, since results suggest that beyond this point improvement is negligible (<1%) for all algorithms considered. Although applying the same smoothing algorithm for the same number of times to all path planning algorithms may seem a fair comparison, further analysis shows that A* requires fewer iterates to reach a point where the reduction in path length is negligible when opposed to the other sampling–based algorithms. Therefore the times the smoothing algorithm shall be repeated must depend on the prospective improvement that future iterates will introduce. The path length reduction of the last set of iterates will give an indication of the level of improvement of future smoothing iterates. The path length reduction margin asymptotically reduces as the number of iterates increases and the path length approaches the optimal one. Consequently, the path length reduction of the previous arbitrary set of iterates will yield a path length reduction higher than the next equal future sample of iterates. This line of thought is considered for the improvement in the path smoothing algorithm.

C. The new smoothing algorithm

The smoothing algorithm developed in[1] is amended as explained in Algorithm 1 to incorporate this advancement. $\tau_{unsmoothed} \in \mathbb{R}^M$ represents the path points generated from the A* or RRT in M dimensions. $\tau_{new} \in \mathbb{R}^M$ are the new path points after the last smoothing iterate. $l \in \mathbb{R}$ stores the path length of previous iterates. $excess \in \mathbb{R}$ is an arbitrary value used to define the number of backward iterates needed to retrieve a path for quantification of path length reduction. The $position \in \mathbb{R}$ variable is set to 1 if the smoothing iterates are less than or equal to $excess$ or $(iterate\_count - excess)$ otherwise, where the current iterate number is stored in $iterate\_count \in \mathbb{R}$. l_{excess} denotes the path length at the defined $excess$ position. The smoothing algorithm is limited to a maximum number of iterates ($max\_iterate \in \mathbb{R}$) set to 1000 iterates due to the reason explained previously. x_{p1} and x_{p2} both $\in \mathbb{R}^M$ are two path points in the path being smoothed. x_{pnt1} and x_{pnt2} both $\in \mathbb{R}^M$ are two randomly–selected points, one on the segment connecting x_{p1} to the next path point $x_{p1+1} \in \mathbb{R}^M$ and the other on the segment connecting x_{p2} to the next path point $x_{p2+1} \in \mathbb{R}^M$.

If $excess$ is too low the comparison between the current path length and the path length $excess$ iterates before would lead to values smaller than 1%, even if there is room for further improvement in excess of 1% in the next iterates. On the other hand, if $excess$ is too high more iterates would be computed increasing smoothing time although improvement would be negligible after certain amount of iterates. Based on this consideration, an analysis on the rate of path length reduction with smoothing iterates for all considered path planning algorithms will yield to an $excess$ value. Results shows that generally (some situations for some path planning algorithms take more some take less), it takes approximately 20 iterates for a 1% path reduction to be recorded, if path reduction is possible. Therefore the variable $excess$ in the new smoothing algorithm is set to 20.

Algorithm 1: The Amended Smoothing Algorithm

```
1:  Calculate the path length from $\tau_{unsmoothed} \rightarrow l_{excess} \rightarrow l(1)$
2:  Set $l(1 : max\_iterate) = 0$; $position = 1$; $l_{new} = 0$; $iterate\_count = 1$;
3:  **While** $((l(iterate\_count) < l(position)$ - $(0.01 \times l_{excess}))$ **AND** $(iterate\_count < max\_iterate))$ **OR**
    $(iterate\_count <= excess)$
4:    **IF** $(iterate\_count >= excess)$ **THEN** $position = iterate\_count - excess + 1$; $l_{excess} = l(position)$
5:    **ELSE** $position = 1$ **END**
6:    **IF** $(iterate\_count \neq 1)$ calculate the path length from $\tau_{new}$
7:    Randomly select $x_{p1}$ and $x_{p2}$ from $\tau_{new}$
8:    Swap $x_{p1}$ and $x_{p2}$ if $x_{p2}$ is a node further than $x_{p1}$ to the goal node $x_{goal}$
9:    Randomly select $x_{pnt1}$ and $x_{pnt2}$ from the path points
10:   **IF** segment connecting $x_{pnt1}$ and $x_{pnt2}$ is obstacle free **THEN** remove all intermediate nodes
      i.e. from $x_{p1+1}$ to $x_{p2}$ assign new path to $\tau_{new}$ and re-calculate the path length from
      $\tau_{new} \rightarrow l(iterate\_count)$
11: **END**
```

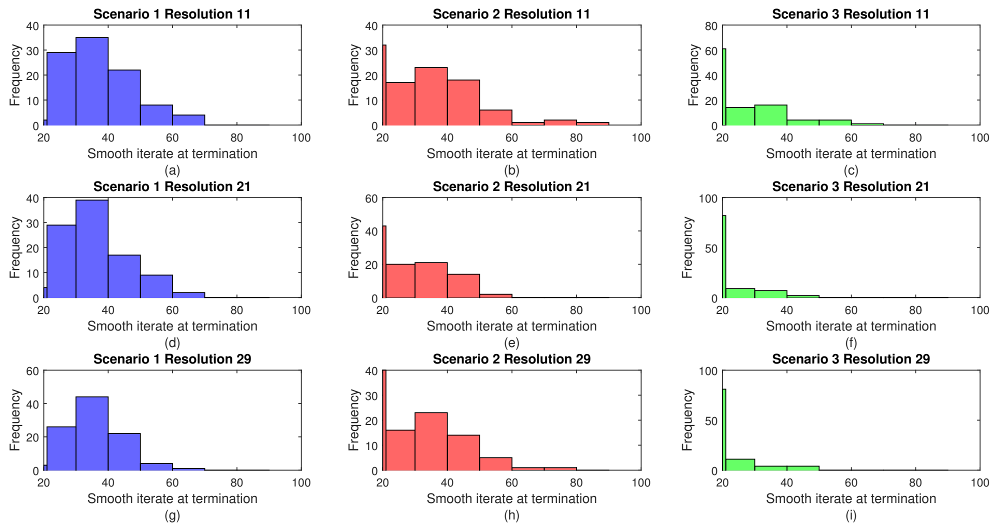

Figure 7. Frequency distribution for the original A* algorithm for Resolutions 11, 21 and 29 for all the 3 considered scenarios for 100 iterates. The x–axis denote the iterate at which the smoothing iterate stopped i.e. $iterate\_count$ **when the smoothed path length of the last iterate was less than 1% shorter than the path length 20 iterates before**

D. A* Algorithm

Figure 7 shows the frequency distribution for the A* algorithm for selected resolutions. Due to space limitations only resolutions 11, 21 and 29 are considered. The iterate at which the smoothing iterate stops is selected for the performance measure.

From Figure 7 it can be deduced that the simpler the obstacle scenario the more smoothing iterates are required until further smoothing yielded less than 1% reduction. In fact for scenario 1 the smoothing algorithm stops at the minimum iterate condition i.e. 20 only in less than 5 situations over a 100 runs for all resolutions considered. Oppositely, for more complex scenarios this value increased to over 50. Similarly the mean distribution shifts to larger smoothing iterates at termination. The reason for this is that, for simple scenarios, the majority of the environmental space is obstacle free. Oppositely, in obstacle dense areas the possible path options are highly restricted. Since the smoothing algorithm can use any two points on the generated path irrespective of residing on grid points or not, there is more room for improvement in the simple cases. Oppositely, in complex situations it becomes very difficult to choose two points from two different segments and reduce the overall path length by interconnection.

As illustrated by the results of Figure 7, it is difficult in 20 iterates to improve the path by at least 1% as the options are very restrictive. Therefore, this implies that the level of smoothing in complex scenarios can be reduced only to a smaller percentage when compared to simpler paths for the considered stopping condition. But results of Table 2 show that in 1000 iterates the path length reduction was 13.7% for Scenario 3 as opposed to an 11.9% for Scenario 1. This confirms that although for complex paths it is more difficult to find smoothing combinations, for 1000 smoothing iterates a better path length reduction can be achieved.

From the results of Figure 7 it can be concluded that the resolution has negligible effect on the smoothing performance. This is in line with the theoretical aspect of the smoothing algorithms that neglects the grid point limitation and considers all points not on an obstacle as usable path points. Although the smoothing is a prerogative of the unsmoothed path generated by the A* algorithm, the path difference is made negligible in a few iterates. Furthermore, in less than 90 iterates the smoothing algorithm was able to improve the path up to a point where the reduction over the last 20 iterates was less than 1%. This confirms the optimality of the A* algorithm.

But a thorough analysis of the results show that, although in the last 20 iterates less than 1% reduction is recorded, the new algorithm is not able to reach the same level of smoothing as the initial algorithm, especially for complex scenarios as illustrated in Table 3. For scenarios 1 to 3 a percentage reduction in path length of 9.9%, 4.8% and 0.6% for the new smoothing algorithm, as opposed to 11.9%, 15.7% and 15.7% for the original smoothing algorithm for scenarios 1 to 3, respectively. These results show that for simple scenarios 76% path length reduction was achieved for the new smoothing algorithm with respect to the original smoothing algorithm in only 6% of the time. Oppositely, only 4% path length reduction was achieved in 3% of the time for the new smoothing algorithm with respect to the original smoothing algorithm in scenario 3. From this analysis it can be concluded that for simple scenarios the new smoothing algorithm is very effective but for more complex situations more buffer (more than 20) shall be considered as it is very difficult to find combination of points within the path segments that can eliminate intermediate segments for an A* generated path that is already almost optimal.

Table 3. Mean values for all resolutions for the A* algorithm with the original and new smoothing algorithm

Mean values for A*	Scenario 1	Scenario 2	Scenario 3
Unsmoothed path length	1.2586	1.9864	2.8154
Smoothed path length for original smoothing algorithm	1.1092	1.6749	2.3730
Smoothed path length new smoothing algorithm	1.1340	1.8919	2.7984
Smoothing time for original smoothing algorithm (ms)	27.036	32.397	35.975
Smoothing time for new smoothing algorithm (ms)	1.671	1.481	1.212

E. Rapidly–Exploring Random Tree (RRT)

The same smoothing algorithm with the same predefined stopping conditions is utilised for the RRT algorithm with the results illustrated in Figure 8. The first evident result that can be deduced from Figure 8 is that the new smoothing algorithm takes longer than the A* algorithm for all scenarios and step sizes/resolutions considered. This confirms theory that claims that the A* algorithm is more optimal than the RRT algorithm.

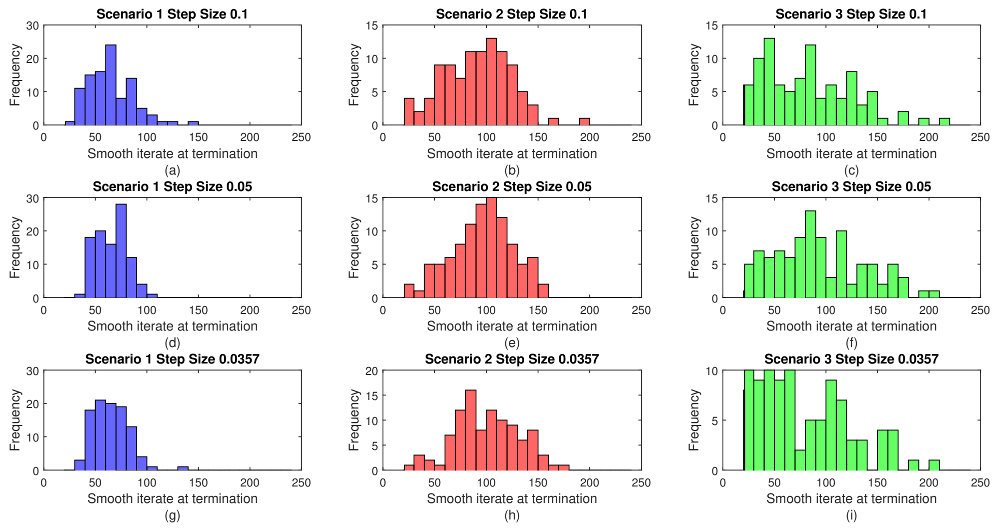

Figure 8. Frequency distributions for the RRT algorithm for step sizes of 0.1, 0.05 and 0.0357 for all the 3 considered scenarios for 100 iterates. The x–axis denote the iterate at which the smoothing iterate stopped i.e. *iterate_count* **when the smoothed path length of the last iterate was less than 1% shorter than the path length 20 iterates before**

It is important to remark that the step size in the RRT algorithm is defined as the distance between the current planner position to the next point in the direction of the nearest randomly generated seed. The resolutions and step sizes are arbitrary selected to offer a fair comparison between two algorithm from two different categories of path planners. A case in point is a resolution of 11 which in a 1x1x1 environmental space would result in a grid point–to–point difference of 0.1 just as the step size for RRT.

Oppositely to the A* algorithm, the more complex the scenario the more dispersed the frequency distribution for the same considered step size. The main difference is mostly evident in the frequency distributions of Scenarios 2 and 3. In RRT, the generated path is not optimal and the degree of improvement is larger than 40%. Therefore, although the environmental space is restrictive the number of combinations for path smoothing remain possible.

Furthermore, the step size has minimal effect on the smoothing iterates required for the stopping condition. As remarked for A* the smoothing algorithm neglects any constraints on distances between points when interconnecting and obstacle planes remain the only limitation.

Table 4 shows the mean path length and smoothing time for the RRT algorithm with the original and new smoothing algorithms. The percentage reduction in path length for the new smoothing algorithm is 44.1%, 40.3%, 27.1% as opposed to 46.4%, 49.5%, 48.4% for the original smoothing algorithm for scenarios 1 to 3, respectively. This show that for Scenarios 1 and 2 the difference in path length reduction is less than 10% while the path smoothing time has reduced by 19 and 13 times respectively. On the other hand for scenario 3, the path length reduction is 21.3% in 15 times less time. Results confirm the validity of the new smoothing algorithm for the RRT algorithm especially in terms of computational demand with minimal effect on path length reduction for simple scenarios. In other words even for RRT the original smoothing algorithm wasted time in trying to smoothen a path which was already almost optimal. For more complex scenarios, just as for the A*, the 20 iterate buffer could be extended further due to the difficulty in finding solutions that can smoothen the path. Such an increase in buffer will increase computational time so this buffer must be further tuned based on the obstacle density.

The mean values of Table 8 also confirm that the A* outperformed the RRT algorithm in terms of

Table 4. Mean values for all resolutions for the RRT algorithm with the original and new smoothing algorithm

Mean values for RRT	Scenario 1	Scenario 2	Scenario 3
Unsmoothed path length	2.1300	3.7432	5.4485
Smoothed path length for original smoothing algorithm	1.1413	1.8888	2.8111
Smoothed path length new smoothing algorithm	1.1902	2.2356	3.9699
Smoothing time for original smoothing algorithm (ms)	46.486	48.808	52.961
Smoothing time for new smoothing algorithm (ms)	2.438	3.692	3.399

smoothed path length and smoothing time since the former was able to generate more optimal paths. The new smoothing times presented in Tables 7 and 8 is only a small fraction (less than 5%, reducing as the resolution or step sizes decreases or increases respectively) of the path generation time making the new smoothing algorithm beneficial for online applications where the cost–gain ratio is low.

F. RRT without step size constraint

Moreover, the smoothing algorithm is applied on the RRT without step size constraints. In the RRT without step size constraint algorithm, the current planner position can be interconnected directly to the nearest random node if no collision results. This creates zig–zagging paths as showed and explained in.[1] This does not imply that the path length is longer as path construction depends upon randomly placed nodes and the RRT produces also non-optimal paths. Therefore the RRT without step size constraints algorithm as the original RRT can generate paths which have margin of improvement through the smoothing algorithm. Smoothed path length and smoothing time results are in line with the results of the standard RRT. In this regard, the more complex the path the more iterates are required since the stopping condition is achieved.

G. Multiple Randomly–Exploring Random Trees (MRRT)

The new smoothing algorithm is further applied to the MRRT algorithm. The frequency distribution results are illustrated in Figure 9. Similar to the other RRT–based algorithm and opposite to A* the more complex the path the more smoothing iterates are required for the stopping condition to be achieved. The non-optimal path generated by the MRRT algorithm as other RRT–based algorithms is the reason for this behaviour.

In MRRT the planner can interconnect trees without any step size constraint and therefore the step size constraint is not considered. The number of seeds per axis parameter is considered instead. As opposed to the standard RRT in which the number of seeds in all axis is 1 i.e. the starting node, in MRRT since there are multiple trees the designer can place tree seeds (tree starting nodes) anywhere in space. The position of these seeds could result on obstacles, which in such case, the tree cannot propagate. Therefore, the accurate positioning of these trees will effect the path length and generation time. In our tests, the seeds were placed evenly in the environmental space.

From Figure 9, the effect of the number of seeds per axis on the stopping iterate is negligible just as for the step size constraint in the RRT algorithm since the smoothing algorithm neglects any node positional constraints.

Table 5 shows the mean path length and smoothing time for the MRRT algorithm with the original and new smoothing algorithms. The percentage reduction in path length for the new smoothing algorithm is 54.7%, 35.1%, 19.3% as opposed to 57.9%, 52.5%, 51.1% for the original smoothing algorithm for scenarios 1 to 3, respectively. The path length achieved by the new smoothing algorithm is in line with that of the original smoothing algorithm only for Scenario 1 with the difference increasing to double as the complexity increases. This is in line with the results achieved for RRT and therefore the same conclusions can be drawn with respect to the A* algorithm. As the complexity increases it becomes more difficult to find options that could reduce the path length and therefore the 20 iterate predefined value shall be increased to further improve smoothing. Such increase in minimum iterates prior stopping will then increase the path smoothing time.

The smoothing time has decreased by the new smoothing algorithm by 22, 17 and 24 times of the original smoothing algorithm highlighting the validity of this new smoothing algorithm especially for low obstacle

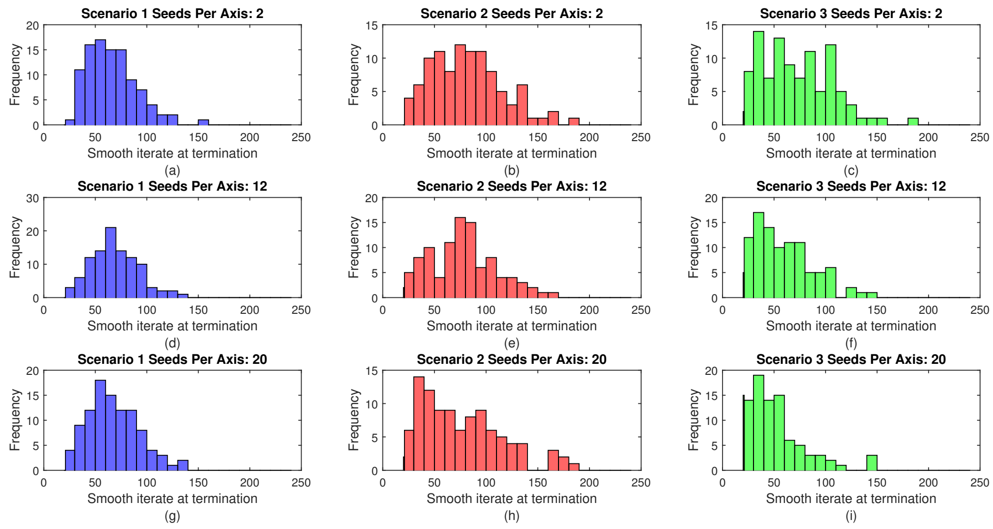

Figure 9. Frequency distributions for the MRRT algorithm for number of seeds per axis of 2, 12 and 20 for all the 3 considered scenarios for 100 iterates. The x–axis denote the iterate at which the smoothing iterate stopped i.e. *iterate_count* when the smoothed path length of the last iterate was less than 1% shorter than the path length 20 iterates before

density scenarios. By increasing the minimum iterate prior stopping the benefits of the path reduction time will be attenuated. Therefore, this empirical parameter shall be chosen attentively based on the obstacle density.

Table 5. Mean values for all resolutions for the MRRT algorithm with the original and new smoothing algorithm

Mean values for MRRT	Scenario 1	Scenario 2	Scenario 3
Unsmoothed path length	2.7164	4.0351	5.8663
Smoothed path length for original smoothing algorithm	1.1428	1.9182	2.8661
Smoothed path length new smoothing algorithm	1.2306	2.6205	4.7343
Smoothing time for original smoothing algorithm (ms)	60.425	61.470	63.577
Smoothing time for new smoothing algorithm (ms)	2.730	3.610	2.695

H. Conclusion

The new smoothing algorithm presented an improvement (by multiple times) in terms of computational time over the original smoothing algorithm. As predicted from theory, this improvement reduced the path length reduction. But the reduction was less than 15% for all A* situations and less than 10% for Scenario 1 of all RRT–based algorithms. This highlights the validity of the new smoothing algorithm although an attentive increase in the minimum stopping iterate condition can improve the smoothing performance of the new smoothing algorithm in complex paths for RRT–based algorithms. This parameter was kept constant to provide a fair comparison between all path planning algorithms. It can be concluded that a path length reduction (20% to 50%) could be achieved at a fraction of less than 1% of the path generation time. This gain is important especially if this path planning and smoothing algorithms (as originally intended) were to be applied to online UAV 3D applications where computational power and autonomy are the bottleneck.

V. Conclusion

This paper presented advancements in the A* and smoothing algorithms in view of 3D UAV path planning. Analysis of our previous work[1] shows that the A* algorithm exhibited a ripple for different resolutions for the same scenario. This ripple is reduced by 46% to 48% in terms of standard deviation for all conditions considered, without increase in path length and computational time, through the developed A* ripple reduction algorithm, that randomly shifts the environment by a distance varying between 0 and half the distance between grid points. From an application's point of view, this advancement creates a clear design rule that the path length decreases asymptotically and generation time increases asymptotically with resolution, eliminating the chance of, for example, increasing the path length by increasing the resolution as before.[1] The improved smoothing algorithm is developed after it was noted, especially for A*, that computational time is wasted in improving a path which is already almost optimal. Therefore, instead of smoothing the path for a predetermined number of times, the smoothing algorithm measures the rate of improvement over the last predetermined number of iterates and stops if the improvement is low. Results confirm an improvement of more than 10 multiples less in computational time for all considered algorithms with less than 10% reduction for A* and between 25% and 45% in path length for RRT and its variants. The path length reduction reduces as the complexity of the scenario increases. This improvement makes the smoothing algorithm a candidate for online 3D path smoothing applications as the computational time is less than 1% of the path generation time of all algorithm considered.

The ultimate aim of this paper and the previous,[1] is to review, develop and assess promising path planning algorithms for UAV 3D path planning. Through these papers this goal is achieved as an extensive review of the current state-of-the-art was presented, followed by the development of the A* and RRT algorithms as the state-of-the-art representatives of the graph–based and sampling–based methods. The development and testing in different complexity 3D obstacle environment helps in assessing their validity for the application in question. Advancements and thorough analysis fine tuned the developed algorithms to be used for 3D UAV path planning.

These 3D path planning platforms have the potential to be developed and tested for dynamic obstacle scenarios. Further future development shall include the incorporation of uncertainty and constraints in sensory and actuator systems within the developed path planning algorithms.

References

[1]Zammit, C. and van Kampen, E. J., "Comparison between A* and RRT Algorithms for UAV Path Planning", *AIAA Guidance, Navigation and Control Conference*, AIAA SciTech Forum, Kissimmee, FL, 8–12 Jan., 2018.

[2]Sujit, P. B., and Ghose, D., "Search by UAVs with Flight Time Constraints using Game Theoretical Models," *AIAA Guidance, Navigation and Control Conference*, San Francisco, CA, 15–19 Aug. 2005, pp. 1–11.

[3]Tisdale, J., Kim, Zuwhan Kim Zuwhan and Hedrick, J., "Autonomous UAV path planning and estimation," *IEEE Robotics & Automation Magazine*, Vol. 16, No. 2, 2009, pp. 35–42.

[4]Chakrabarty, A. and Langelaan, J. W., "Energy maps for long–range path planning for small-and micro–uavs", *AIAA Guidance, Navigation and Control Conference*, Chicago, IL, 10–13 Aug., 2009, pp. 1–13.

[5]Gros, M., Schöttl, A., and Fichter, W., "Spline and OBB–based Path–Planning for Small UAVs with the Finite Receding–Horizon Incremental–Sampling Tree Algorithm", *AIAA Guidance, Navigation and Control Conference*, Boston, MA, 19–22 Aug., 2013, pp. 1–17.

[6]Amin, J. N., Boskovic, J. D. and Mehra, R. K, "A Fast and Efficient Approach to Path Planning for Unmanned Vehicles", *AIAA Guidance, Navigation and Control Conference*, Keystone, CO, 21–24 Aug., 2006, pp. 1–9.

[7]Park, S., Choi, H.–L., Roy, N., and How, J., "Learning Covariance Dynamics for Path Planning of UAV Sensors in a Large–Scale Dynamic Environment", *AIAA Guidance, Navigation and Control Conference*, Chicago, IL, 10–13 Aug., 2009, pp. 1–18.

[8]Crispin, C. and Sobester, A., "An Intelligent , Heuristic Path Planner for Multiple Agent Unmanned Air Systems", *AIAA Information Technology at Aerospace*, Kissemmee, FL, 5–9 Jan., 2015, pp. 1–13.

[9]Boskovic, J. D., Knoebel, N., Moshtagh, N. and Larson, G.L., "Collaborative Mission Planning & Autonomous Control Technology (CoMPACT) System Employing Swarms of UAVs", *AIAA Guidance, Navigation and Control Conference*, Chicago, IL, 10–13 Aug., 2009, pp. 1–24.

[10]Ryan, A. and Hedrick, J. K., "A mode–switching path planner for UAV–assisted search and rescue", *Proceedings of the 44th IEEE Conference on Decision and Control*, Seville, Spain, 12–15 Aug. 2005, pp. 1471–1476.

[11]Kothari, M., Postlethwaite, I., "A Probabilistically Robust Path Planning Algorithm for UAVs Using Rapidly-Exploring Random Trees", *Journal Intelligent Robotic Systems*, Vol. 71, pp. 231–253, 2013.

[12]LaValle, S., Sharma, R., "A framework for motion planning in stochastic environments: application and computational issues", *IEEE International Conference on Robotics and Automation*, Nagoya, Japan, 21–27 May 1995, Vol. 3, pp. 3063–3068, 1995.

[13]LaValle S. M. "Probabilistic roadmaps for path planning in high–dimensional configuration spaces", *IEEE Transactions on Robotics and Automation*, Vol. 12, No. 14, pp. 566–580, 1996.

[14]Gonzàlez, D., P*è*rez, J., Milanès, V., and Nashashibi, F., "A Review of Motion Planning Techniques for Automated Vehicles", *IEEE Transactions on Intelligent Transportation Systems*, Vol. 17, No. 4, pp. 1135–1145, 2016.

[15]Ghandi, S. and Masehian, E., "Review and taxonomies of assembly and disassembly path planning problems and approaches", *CAD Computer Aided Design*, Vol. 67–68, No. October, pp. 58–86, 2015.

[16]Short, A., Pan, Z., Larkin, Z. and van Duin, S. "Recent Progress on Sampling Based Dynamic Motion Planning Algorithms", *IEEE International Conference on Advanced Intelligent Mechatronics (AIM)*, Alberta, Canada, Jul. 2016, pp. 1305–1311.

[17]Sathyaraj, M. B., Jain, L. C., Finn, A. and Drake, S., "A Multiple UAVs path planning algorithms: a comparative study", *Fuzzy Optimization and Decision Making*, Vol. 7, No. 3, 2008, pp. 257–267.

[18]Hart, P. E., Nilsson, N. J. and Raphael, B., "A Formal Basis for the Heuristic Determination of Minimum Cost Paths", *IEEE Transactions on Systems Science and Cybernetics*, Vol. 4, No. 3, pp. 100–107, 1968.

[19]Tseng, F. H., Liang, T. T. and Lee, C. H. Chou, L. D. and Chao, H., "A Star Search Algorithm for Civil UAV Path Planning with 3G Communication", *10th International Conference on Intelligent Information Hiding and Multimedia Signal Processing (IIH–MSP)*, Kitakyushu, Japan, 27–29 Aug. 2014, pp. 942–945.

[20]Lavalle S. M. and Kuffner J. J., "Randomized kinodynamic planning", *International Journal of Robotics Research*, Vol. 20, No. 3, pp. 378–400, 2001.

[21]LaValle S. M. "Probabilistic roadmaps for path planning in high–dimensional configuration spaces", *IEEE Transactions on Robotics and Automation*, Vol. 12, No. 14, pp. 566–580, 1996.

[22]LaValle, S. M. and Kuffner, J. J. "Randomized kinodynamic planning", *Proceedings of the IEEE International Conference on Robotics and Automation*, Detroit, MI, 10–15 May 1999, pp. 473–479.

[23]Devaurs, D., Siméon, T. and Cortés, J. "Optimal Path Planning in Complex Cost Spaces With Sampling–Based Algorithms", *IEEE Transactions on Automation Science and Engineering*, Institute of Electrical and Electronics Engineers, 2015.

[24]Geraerts, R. and Overmars, M. "Creating high–quality paths for motion planning", *International Journal of Robotics Research*, Vol. 26, No. 8, pp. 845–863, 2007.

[25]Tsardoulias, E. G., Iliakopoulou, A., Kargakos, A. and Petrou, L. "A Review of Global Path Planning Methods for Occupancy Grid Maps Regardless of Obstacle Density", *Journal of Intelligent and Robotic Systems*, pp. 1–30, 2016.

[26]Clifton, M., Paul, G., Kwok, N., Liu, D. and Wang, D. "Evaluating Performance of Multiple RRTs", *IEEE conference on Mechatronic and Embedded Systems and Application*, Bejing, China, 12–15 Oct. 2009, pp. 564–569.

[27]Paul, G. "Multiple Rapidly–exploring Random Tree (RRT)", *MATHWORKS*, [Online Database], https://www.mathworks.com/matlabcentral/fileexchange/21443-multiple-rapidly-exploring-random-tree--rrt-?requestedDomain=www.mathworks.com, [retreived 30 October, 2016].