REGULATORY COMPLIANT UAS NAVIGATION BASED ON A BITMAP GRID LOCAL PATH PLANNING ALGORITHM

A. Pablo De Porcellinis*, B. Jesús Villadangos, C. Daniel Aláez, and D. Manuel Prieto
Public University of Navarre, Pamplona, Spain

ABSTRACT

Infrastructure inspection nowadays is a mandatory task to maintain the life quality of the society. As these tasks have to be performed safely, ensuring that waypoints satisfy the most strict current Unmanned Aircraft Systems (UAS) regulations like not put in risk (overfly) items as persons, animals, etc., this work proposes an algorithm for UAS navigation during the power-line inspection that works autonomously. The proposed algorithm determines all along the flight path the most suitable regulatory compliant waypoints (RCWPs) to satisfy international regulations during the whole flight. This solution only requires knowing the exact position of the UAS during the flight and uses an artificial vision system for item identification in the environment along the route.

1 INTRODUCTION

From the beginning of modern civilizations, the use of different kinds of infrastructures has made easier the life of humans. Therefore, it is possible to establish a relationship between the life quality of the different countries nowadays and the quality of their infrastructures. To maintain the quality of these infrastructures it is mandatory to apply to them a maintenance. In the case of Spain for example, it is required by law to inspect the whole power-line infrastructure every three years.

There are several methods that have been applied to perform these inspections, but in the last decade the use of mobile robots and more specifically UAS, commonly known as drones, have become more common. The world of drones has a strong regulation and depending on the characteristics of the intended operation, a different risk level is assigned [1].

The infrastructures can be located in different scenarios that imply an increase in the level of risk, for example when they are close to urban areas or in contrast in very remote areas where the previous information about the operational scenario conditions is limited.

*Email address(es): deporcellinis.150939@e.unavarra.es

To reduce the level of risk of the operation, it is necessary to equip the aircraft with systems that improve its assurance and integrity levels and allow it to perform such inspections.

The level of risk assigned to the operations is mainly divided into ground and risk levels [2]. There are several items that are present in the areas of real inspections and need to be faced to overcome this risk. Some examples of these real-time problems that can be present in the inspection area are the wrong exact location of the power-line assets, uninvolved persons, moving vehicles, or even groups of grazing animals. The problem with these items is that it is not possible to predict them in advance, as they are real-time problems.

Although these elements are not considered obstacles, since they do not endanger the integrity of the aircraft, they should not be flown over to comply with the regulation. Therefore, it is necessary to come up with some solution to tackle the problem of these unforeseen actors and to allow the dynamic change of aircraft trajectory. More precisely, it is required a UAS navigation system that allows real-time and autonomous planning of the aircraft flight to ensure a safety flight that fulfills the requirements imposed by the regulator for the authorized inspection flights.

1.1 State of Art

In the literature, there is a wide variety of UAS navigation techniques that are artificial vision-based. In [3], they establish a taxonomy of the vision-based UAS navigation systems in three different groups depending on whether they are mapping-based methods for visual localization, object detection and avoidance approaches, or path planning-based approaches.

Regarding the path planning approaches, there are two different techniques used: Global Path Planning and Local Path Planning. The main difference between these two techniques is that Global Path Planning allows planning the path offline but requires the whole previous knowledge of the workspace and it is not possible to perform any change on the studied scenario while Local Path Planning allows to modify the path in a real-time basis, based on the characteristics of the environment [4].

In [5] a global path planning method using an A* algorithm is developed. This algorithm focuses on the mission planning using cost functions for computing the best route. This standard route algorithms are good for generating a min-

imum cost route but are limited as they are based on a predefined cost function. The solution studied in [6] presents a different route planning approach being able to generate mission adaptable routes based on different constraints included in the cost function, such as minimum route length or maximum turning angle. Although this method is categorized as global path planning it is able to modify this constraints along the mission, getting closer to a more adaptive method. However, the mission parameters modification is not based on the visual detection of elements in real time.

Regarding global path methods that are capable of deal with obstacle detection, in [7] they introduce with a stepping neural network approach a way of producing fast and optimal motion plans for static environments. The problem of these global path methods is still that is not possible to modify the path in a real time basis, so it is necessary to completely know where the obstacles are placed prior to the UAS flight.

Looking for local path planning algorithms to find a real time path planning solution, there are several methods that are capable of avoid obstacles and find an optimized path at the same time. As an example of these methods, in [8] they purpose a combination of an Artificial Potential Field (APF) method and A-star fusion algorithm to avoid dynamic obstacles and find a shorter path simultaneously. The APF method use the gravitational field to control the flight direction of the UAV. The problem of this complex proposed algorithm is that is more suitable to fall into a local minima, and the path not being sufficiently smooth.

More recent studies as [9] are capable of produce a three dimensional (3D) vortex fields for path planning of different drones that works simultaneously in a swarm environment. This solution is capable of detecting moving obstacles and the sequential cooperation will allow to reduce the local minimum problem. For the proposed use case in this article, the cooperation is at the same time a problem, as it is not efficient to make a swarm flight to detect obstacles and find a local path. In addition, the detected obstacles are physical elements that if not avoided, will cause an accident of the aircraft.

Reference	Path Planning Type	Obstacles / Items Detection	Main Target	Safety Oriented
[5]	Global	No.	Cost Function. Distance Optimization	No
[6]	Global	No.	Mission adaptable routes (Route Length, Max. turning angle, ...)	No
[7]	Local	Yes. Static obstacles	Optimal motion plans with minimal performance loss	No
[8]	Local	Yes. Moving obstacles	Obstacle Avoidance and route length	No
[9]	Local	Yes. Moving obstacles	Path Oscillations and distance Travelled	No
RCLPP	Local	Yes. Moving Items	Safety Compliance. Regulatory Basis.	Yes

Table 1: Path Planning Algorithms Comparison

In this paper it is proposed a new regulatory compliant local path planning algorithm (RCLPP) which evaluates dynamically the UAS path. The path calculated is composed by Regulatory Compliant Waypoints (RCWPs).Each one ensures items location and local path planning strategies to find a feasible solution from the definition of a security region about those items.

The use of this classical definition of safety region has been previously used for a global path planning algorithms but has not deeply exploited for local path planning methods. For the proposed use case the main objective is to face the problem of not flying through a moving item to be compliant with the regulation limitations, so therefore is not necessary to apply a cost function [10] to decide the more cost effective solution based on desired parameters, as the first viable solution reached by the algorithm is the one that will be selected.

The main difference between the proposed method and the ones presented on the state of the art is that is the only safety oriented solution with a regulatory basis, to comply with the constraints of the authorized inspection flights.

2 REAL TIME UAS NAVIGATION ALGORITHM

This section is divided into two main parts. In the first part, the real-time UAS navigation algorithm is explained, where a previously defined flight plan is complemented with the local path planning solution for those regions where regulatory problems have to be faced.

In the second part, this local path planning solution is explained in detail, highlighting one by one the functions used to find the regulatory compliant path.

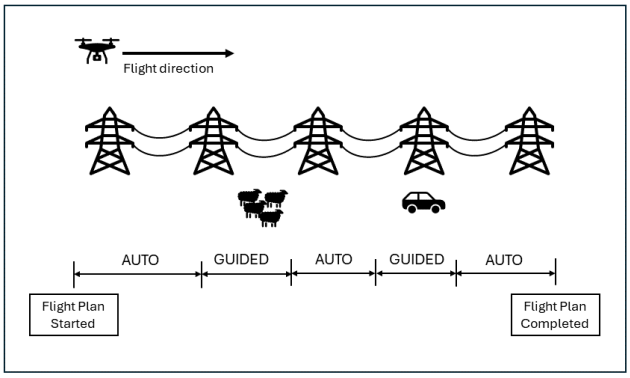

Figure 1: Expample of the proposed solution

In Figure 1, it is represented graphically the operating concept of the proposed solution. The idea is to complete the power-line inspection in an automatic way following a Flight Plan. The special characteristic of this Flight Plan is that is dynamic and if some regulatory challenging items appear in the expected trajectory of the UAS, it is capable to perform a Flight Plan reconfiguration. As can be seen, when the items

are detected, the flight mode has changed between "AUTO" and "GUIDED", to be able to adopt this real-time Flight Plan reconfiguration.

The difference between the used flight modes for this solution is that in the "AUTO" mode, the WPs are automatically introduced and followed based on a predefined route while in the "GUIDED" the UAS processes only one WP at the same time, so it is necessary to set the next target point each time the previous one is reached. Therefore, each of them is suitable for specific flight conditions.

```
1   FlightPlan <- read_FlightPlan()
2   flight_start(ARM, AUTO)
3   while (FlightPlan is not completed)
4         RCWPs <- emptyset
5         while (items_detected(onBoard_camera) == false)
6               NextWP <- get_nextWP(currentGPS, FlightPlan)
7               RemainingFlightPlan <- update(FlightPlan)
8               Flight_mode = AUTO
9         while (items_detected(onBoard_camera) == true)
10              ItemsLocation<- calculate_ItemsLocation()
11              RCWPs <- LPP(itemsLocation, currentUAVPos, last(FlightPlan))
12              If RCWPs <> emptyset
13                    NextWP <- second(RCWPs)
14                    Flight_mode = GUIDED
15                    RemainingFlightPlan <- update(FlightPlan)
16              Else
17                    Flight_mode = LAND
```

Figure 2: Real Time UAS Navigation Algorithm

The Flight Plan for the mission was previously designed based on the position of the power-line assets. This Flight Plan is designed using a map-based solution where additional aspects such as the communication network coverage, external services coverage (GNSS services, U-Space, etc), or already obstacle locations play an important role in the definition of the Flight Plan WPs.

The real-time UAS navigation algorithm is described in 2. In the first step of the proposed method, the $read\_FlightPlan()$ function reads the predefined route WPs and transfers it to the command unit of the aircraft, providing it with what is needed to develop the flight in an automatic way.

The next step is to start the flight, providing the command for arming the aircraft and setting the "AUTO" flight mode, with which the UAS is capable of autonomously following the predefined Flight Plan.

While the flight plan is not completed, the proposed solution will continuously check in which of the possible situations the aircraft is flying in. The system responsible for providing the input for the selection of the method is the RGB camera on-boarded in the UAS. This forward-looking optical sensor captures in real time the flight environment information to search for any item that should not be flown over to comply with the regulation.

To localize the items, the methodology for 2D object detection and tracking proposed in [11] is used. This methodology is based on assigning a rectangular Region of Interest (ROI) to detected items by a camera over the course of successive video frames. This detection method combines a complete Robot Operating System (ROS), a fast object detector (You Only Look Once (YOLO) series [12]) and a visual object tracking method.

As the YOLO network need to be trained to detect with accuracy the desired type of elements, in this case the training was performed to recognise groups of people and grazing animals. The reason for selecting those type of elements principally is because they are identified as the most common on-ground regulatory barrier elements [1]. However, as the YOLO has the capacity of being trained, the proposed model allows, if required, to add new elements to be detected in the planning phase, depending on the characteristics of the scenario and on user needs.

While the onboard camera does not detect any potential item, the proposed UAS navigation method will command to the aircraft to continue flying in AUTO mode. This flight mode is based on a global path planning method, in which all the proposed waypoints are fixed prior to the start of the flight. For this flight mode, the system will continuously check in each iteration $get\_nextWP()$ function which is the next pre-planned WP to be reached, and commands automatically to the UAS to reach this position. To search for this WP the function uses as input the GNSS position of the UAS in each captured instant and compares it with the pre-planned flight plan, to select the closest one to this position.

Inside the same status loop, the $update()$ function is used to update the status of the pre-planned Flight Plan. It has two main functionalities:

- Update the Flight Plan to consider in the next iteration only the remaining WP of the pre-planned route that is not left behind yet.
- Set the Flight Plan as "completed" once it is finished, to end the loop and finish the flight.

If a potential item is detected, i.e. $items\_detected()$ function is true, the proposed method will swap to the second status loop described in Figure 2 and will remain inside this loop until the onboard camera does not detect any potential item in the next iteration. Inside this loop, the proposed UAS navigation method will command to the aircraft to fly in "GUIDED" mode.

The first step of this status loop is to transform the item information captured by the camera into a location for the item. To perform this task, the methodology described in [11] is used.

This methodology first identifies the position of the bounding box relative to the center of the camera. Once it

[1] The grazing animals training was used also by UPNADrone team to fulfil the requirements for the IMAV 2024 outdoor competition

is identified, it is possible to obtain the distance between the item and the camera using the pitch angle θ of the camera, the vertical angle between the rays that project to the camera focal center and to the middle bottom of the boundary box and the height of the aircraft h. After this distance computation, a transformation between the camera reference frame and the North-East-Down (NED) coordinate system [13] is used (in the camera reference frame the X axis is pointing in the forward direction of the camera plane and the Y axis is pointing right in the camera plane). This transformation is performed to have the position of the items in the common reference frame used for navigation.

Once the position of the items is known, the status loop will arrive at the $LPP()$ function, where the Local Path Planning Algorithm described in the following section is used to fulfill the objective of finding a new route based on RCWP.

Once the status loop exits the $LPP()$ function there are two possibilities:

- The function gives as output a RCWPs empty set, which means that is not possible to find a regulatory compliant path, so the $Flight\_mode$ changes to "LAND" to perform a safety maneuver.

- The function gives as output a RCWPs matrix with the new regulatory compliant track to follow. As the $Flight\_mode$ is "GUIDED", the UAS will take the first element of the set as the next location to be reached.

2.1 *Local Path Planning (LPP) Algorithm for Regulatory Compliant (RC) Waypoints Selection*

In this subsection, the Local Path Planning Algorithm (LPP) is explained in detail, to understand how the regulatory compliant path is computed based on the detection of real-time identification of non-compliant items. This algorithm is described in Figure 3.

```
LPP(itemsLocation, currentUAVPos, last(FlightPlan))
    Grid(1:N, 1:N) = 1
    Prohibited, start, end <-insert_items(itemsLocation, Grid, currentUAVPos,...
    ...last(FlightPlan))
    Updated_Grid <- set_RC_safetyareas(Prohibited, Grid, RC_safety_factor)
    Final_Grid <- safety_areas_environment(Updated_Grid. RC_safety_factor)
    InitialPath <- get_initial_path(start, end, Final_Grid)
    FinalPath <- get_optimized_path(InitialWPs)
    RCWPs <- points_to_RCWP(FinalPath)
```

Figure 3: LPP Algorithm for RCWPs selection

This algorithm works with a grid representation with dimensions NxN. Each of the grid points represents a combination of latitude and longitude GNSS coordinates.

The N parameter can be selected depending on different variables of the problem, such as how reactive is the UAS being used or the separation between the GNSS coordinates translated into the grid, as N is directly related to the time to reach the next contiguous point.

The algorithm converts the grid representation into a bitmap, depending if the coordinate can be reached by the UAS or not. Being the grid a matrix $G\{NxN\}$ and $x, y \in N$ the possible coordinates inside the matrix, G(x,y) = 1 is a reachable position, and G(x,y) = 0 is blocked by an item position. Each time that the $LPP()$ function runs, it is assigned to each grid point an initial value of 1.

Figures 4, 5, 6, 7 and 8 will represent different stages of the solution in the form of a bitmap grid. These figures are used simply as a graphical sample since the bitmap grid is not a metric setting.

In the first stage of the computation with the function $insert\_items()$, the algorithm reads the coordinates of the items seen by the camera and relates them to their relative position in the grid. In each of the grid points where an item has been placed, the algorithm automatically changes its value to 0, meaning that at this instant, this position is blocked and can not be selected as a reachable point by the aircraft. This function additionally reads the current UAS position in that instant and the next WP from the Flight Plan and assigns them to the variables $Startpoint$ and $Endpoint$ respectively.

As it has been mentioned before, the main objective of this proposed solution apart from completing the power-line inspection correctly is to comply with the regulation, guaranteeing the safety of the flight and the surroundings. This is exactly what is done inside the function $set\_RC\_safety\_areas()$, applying a safety margin around the position of the detected items. For the results presented in this article, a safety margin of $R = 1$ is applied to make the solution simpler. This safety margin is translated into the bitmap grid, assigning to the neighbor cross points from the item position (i.e., grid points located at North, South, East, and West positions from the reference point) a new value of 0 for blocking it.

Once the items have been located and placed into the grid, it is important to analyze them together, to see if they are located on nearby grid points. If they are sufficiently close, the safety areas will block the path. Therefore, it is necessary to make these identified points not reachable. For the example representation it is considered again a safety margin of R = 1. Considering G(x,y) as the item position, if there is an item placed at G(x,y-2), it is necessary to assign to G(x-1,y-1) and G(x-1,y-2) a value of 0, to block this points to find a feasible path without crossing safety margins. The same logic applies to an item that is placed at G(x, y+2). This process is done inside the $safety\_areas\_environment()$ function.

After all the items and their surroundings are correctly identified in the bitmap grid, the algorithm looks for a feasible item avoidance path using the function $get\_initial\_path()$. In Figure 4 is represented this initial regulatory compliant path in a bitmap grid format. The variables $Start$ and End are used as the initial and final points of the computed initial

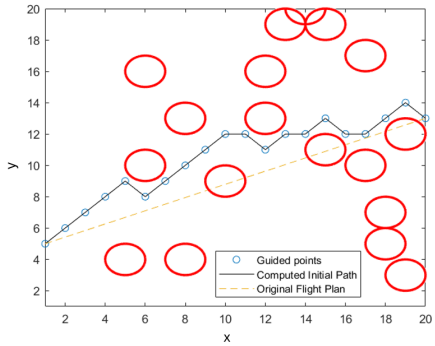

Figure 4: Initial Regulatory Compliant Path

path. There are three cases for the path computation:

1. No blocked space by the detected items and their safety margins (i.e. there are no 0s in the neighbor grid points): the algorithm will try to reach by each movement the y coordinate of the end reference point. In Figure 4 for example, it advances from $G(3,7)$ to $G(4,8)$, as the y coordinate of the final point is 13. If the y coordinate of the path point is equal to the y coordinate of the end reference point, the path will advance straight to the endpoint. In the reference example in Figure 4 this case is not represented, as there are always obstacles in the surroundings when the path reaches the y coordinate of the final point.
2. Partially Blocked space by the detected items and their safety margins (i.e. there are 0s in the neighbor grid points): the algorithm will select the closest free space in the next column of the bitmap grid. In Figure 4 for example, it advances from $G(5,9)$ to $G(6,8)$ to avoid the item placed in $G(6,10)$ and its safety margin.
3. Completely Blocked space by the detected items and their safety margins: the algorithm will finish the function automatically, returning $InitialPath$ as an empty-set.

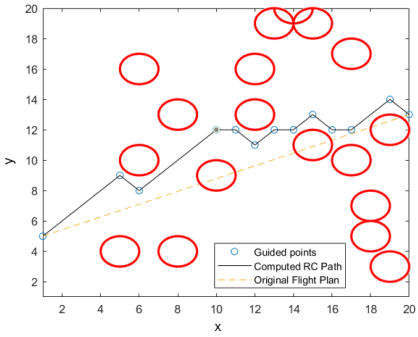

Figure 5: Final Regulatory Compliant path

Once the Initial Path computation is finished the algorithm uses the function $get\_optimized\_path()$ to refine the proposed path and eliminate all the nonrelevant points to achieve the Regulatory Compliant path. This process of path point number reduction is done to assure and facilitate the proper functioning of GUIDED MODE. This final Regulatory Compliant path is represented in Figure 5, again in a bitmap grid format.

As a final step, the $points\_to\_RCWP()$ function transforms the final optimized path grid points into GNSS coordinates, to be able to enter them as feasible Regulatory Compliance Waypoints (RCWPs) for the aircraft GUIDED MODE.

It is considered that the algorithm is capable of reacting to the changes in the environment at a prudent distance as in the represented example. In case the detected item performs a position change when the UAS is sufficiently close to it, there are some emergency procedures defined by the UAS operator to solve the situation safely and securely. These procedures are required by regulation to develop the intended inspection operations.

3 RESULTS

In this section, the results of the experiments performed in Matlab using the proposed method are represented. Firstly, the path planning reconfiguration is tested, to see if it is capable to react to the movement of an item.

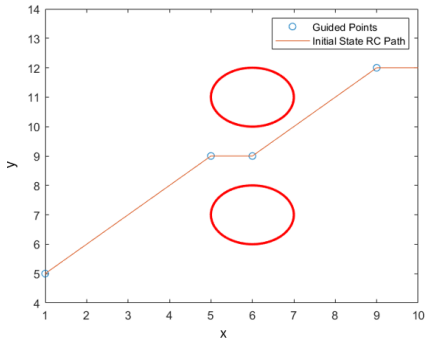

Figure 6: Moving Item Initial State Representation

Figure 6, 7, 8 represents in a bitmap grid format the different stages of a moving item detection and path reconfiguration.

In Figure 6 it is represented the initial situation after the items detection. The UAS is positioned at $G(1,5)$ and the items are initially detected at $G(6,7)$ and $G(6,11)$. As it is appreciable, there is an initial path planning for the guided mode based on this situation. Once the UAS advances one of the items changes its position. This movement is recognized by the camera and translated to the path-planning algorithm.

As it can be seen in Figure 7, the UAS has moved to position $G(1,7)$ (this position corresponds to first frame position $G(3,7)$ in Figure 6, as the UAS has advanced in the "path direction" following the proposed new route). The items are now positioned at $G(3,9)$ and $G(3,11)$, blocking the previous regulatory compliant path.

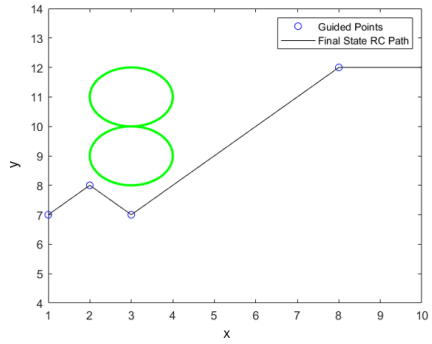

Figure 7: Moving Item Final State Representation

In Figure 8 is represented a combination of both instants, to validate that the algorithm is capable of dynamically changing at every instant, so it proposes new following guided mode points to avoid the items that change its position. As can be seen, the dashed black circles represent the movement followed by the item in previous states, while the green circles represent the safety margin of the items at the final state position. Similarly, the blue dashed line represents the initial state regulatory compliant path, while the red line represents the final state regulatory compliant path.

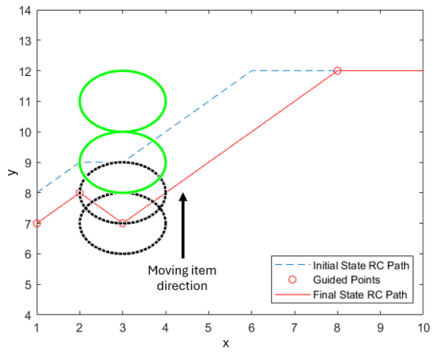

Figure 8: Path reconfiguration after an item movement

Secondly, the Local Path Planning Algorithm for Regulatory Compliant WP Selection was tested for different number of detected items. In total, 25 tests were performed for each of the values.

The Table 2 summarizes the results obtained. The parameter N is set as $N = 20$ for the experiments performed. As it is appreciable, if the number of detected items is smaller than the value of the characteristic dimension of the grid (N), no emergency landing is reported, as it is not probable that the path gets blocked. This is due to the fact that besides having few elements, they are placed randomly, so the probability of being placed surrounding the UAS position at any instant is almost null.

However, when the number of items is increased up to be equal to N, it is appreciable that even the low probability, an emergency landing due to a blocked path was reported. In addition, the number of necessary RCWPs to get the regulatory compliant path increased.

Number of detected items	Number of Guided Points	Emergency Landings
5	5	0
10	7	0
20	9-10	1
30	11-12	3

Table 2: LPP algorithm computation results

Finally, it is observed that when the number of detected items is greater than the characteristic grid dimension, the probability of an emergency landing due to a not feasible path increases as the representative space is saturated. Additionally, it is necessary that the UAS make movements in more than half of the given positions in the bitmap grid.

4 PUBLISHING POLICY

The IMAV proceedings will only be distributed in digital form. The submitting author is responsible for obtaining agreement of all his / her coauthors and any consent required from sponsors before submitting a paper. The authors are obliged to cite relevant prior work.

5 CONCLUSION

In order to solve the problem of infrastructure inspection the use of mobile robots has been used. There are several methods to automatically define the path that has to be followed by the robots, but none of them is capable of dealing with the regulatory barriers caused by the unexpected moving items. We propose a brand new real-time UAS navigation algorithm that is safety-oriented and combines local and global path planning to comply with the constraints imposed in real-time by the unexpected incoming moving items along the previously planned path. We perform a description of the proposed solution, based on a combination of two different flight modes and a regulatory compliant local path planning algorithm, capable of assigning a safety region around the items and modifying the previous planned path based on the unforeseen items. We perform simulations to evaluate the performance of the proposed method and these simulations reflect that is a feasible solution if the presence of items is reasonable to not saturate the sampling space. The next step is to apply it to a real scenario and improve the algorithm to include cost functions that select the best path based on desired parameters, such as less energy consumption, the shortest path, ... In this case, the solution presented has an on-ground focus so the camera objective is front-down oriented to detect the objects on the ground, but future experiments can include a up-front oriented camera to detect aerial items (always with a YOLO training with new item before) and add new use cases to the solution.

REFERENCES

[1] EASA. Easy access rules for unmanned aircraft systems. https://www.easa.europa.eu/en/downloads/110913/en, 2024.

[2] JARUS. Jarus guidelines on specific operations risk assessment (sora). http://jarus-rpas.org/sites/jarus-rpas.org/files/jar_doc_06_jarus_sora_v2.0.pdf, 2019.

[3] Muhammad Yeasir Arafat, Muhammad Morshed Alam, and Sangman Moh. Vision-based navigation techniques for unmanned aerial vehicles: Review and challenges. *Drones*, 7(2), 2023.

[4] Pablo Marin-Plaza, Ahmed Hussein, David Martin, and Arturo de la Escalera. Global and local path planning study in a ros-based research platform for autonomous vehicles. *Journal of Advanced Transportation*, 2018:1–10, 2018.

[5] D.M. Rouse. Route planning using pattern classification and search techniques. In *Proceedings of the IEEE National Aerospace and Electronics Conference*, pages 2015–2020 vol.4, 1989.

[6] Robert J Szczerba, Peggy Galkowski, Ira S Glicktein, and Noah Ternullo. Robust algorithm for real-time route planning. *IEEE Transactions on aerospace and electronic systems*, 36(3):869–878, 2000.

[7] Mayur J Bency, Ahmed H Qureshi, and Michael C Yip. Neural path planning: Fixed time, near-optimal path generation via oracle imitation. In *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 3965–3972. IEEE, 2019.

[8] Chunyu Ju, Qinghua Luo, and Xiaozhen Yan. Path planning using artificial potential field method and a-star fusion algorithm. In *2020 global reliability and prognostics and health management (PHM-Shanghai)*, pages 1–7. IEEE, 2020.

[9] Rafael Monteiro Jorge Alves Souza, Gabriela Vieira Lima, Aniel Silva Morais, Luís Cláudio Oliveira-Lopes, Daniel Costa Ramos, and Fernando Lessa Tofoli. Modified artificial potential field for the path planning of aircraft swarms in three-dimensional environments. *Sensors*, 22(4):1558, 2022.

[10] Andreas Thoma, Karolin Thomessen, Alessandro Gardi, Alex Fisher, and Carsten Braun. Prioritising paths: An improved cost function for local path planning for uav in medical applications. *The Aeronautical Journal*, 127(1318):2125–2142, 2023.

[11] Daniel Aláez, Vasileios Mygdalis, Jesús Villadangos, and Ioannis Pitas. Real-time object geopositioning from monocular target detection/tracking for aerial cinematography. In *2023 IEEE 25th International Workshop on Multimedia Signal Processing (MMSP)*, pages 1–6. IEEE, 2023.

[12] Rachit Mayur Shah, B Sainath, and Akshansh Gupta. Comparative performance study of cnn-based algorithms and yolo. In *2022 IEEE International Conference on Electronics, Computing and Communication Technologies (CONECCT)*, pages 1–6. IEEE, 2022.

[13] Irene Santos and Anibal Ollero. Deteccion de obstaculos lidar mediante la tecnica de background. Master's thesis, Universidad de Sevilla, 2013.