Empirical evaluation of multi UAV coverage path planning for aerial surveying

Jamie Wubben*, João P. Matos-Carvalho†‡, Dário Pedro§, Slavisa Tomic†‡, and Carlos T. Calafate*
*Department of Computer Engineering, Universitat Politècnica de Valéncia, Valencia, Spain
†Center of Technology and Systems, UNINOVA, 2829-516 Caparica, Portugal
‡COPELABS, Universidade Lusófona, Campo Grande 376, 1749-024 Lisbon, Portugal
§BV - Beyond Vision, 2610-161 Lisbon, Portugal
Correspondence: jwubben@disca.upv.es

***Abstract*—Nowadays, Unmanned Aerial Vehicles (UAVs) are being increasingly used for various tasks, including aerial surveys. In these aerial surveys, a drone flies over a region of interest and takes photos. Often, these photos are then used to create one (3D) map, which can be used for later inspection. However, in some cases, this region of interest can be rather large, and performing the survey with a single UAV takes a long time. Besides the inconveniences, this can also worsen the quality of the (3D) map due to changing weather conditions over the day. Consequently, it is advantageous to parallelize this task by deploying multiple UAVs simultaneously. However, when using multiple UAVs, the path-planning stage becomes more complicated. In this work, we provide a straightforward method to create such a multi-UAV survey mission. Although, this multi-UAV coverage path planning is, of course, an essential part of our research, we are mostly focused on the practical implications and outcome. In particular, we want to know how much faster we can perform a coverage mission with multiple UAVs, and if the quality of the survey map can be maintained. For that reason, we performed simulations in various scenarios, but most importantly also performed a real test to confirm the usability of our work. From our experiments, we can conclude that we can reduce the total time to cover an area significantly without lowering the quality of the (3D) map.**

***Index Terms*—Multi-UAV, survey mission, path planning, 3D mapping.**

I. Introduction

The use of Unmanned Aerial Vehicles (UAVs) is becoming more mainstream, and UAVs are now used in numerous applications such as: (i) surveillance and reconnaissance, (ii) agriculture, (iii) disaster management, (iv) delivery services, and (v) aerial photography. In the particular case of aerial photography, UAVs are used quite frequently by offering a more flexible, accessible, and cost-effective platform for capturing aerial imagery compared to other alternatives [1]. For that reason, they are often used in precision data acquisition applications such as: (i) Oil and gas exploration, (ii) Infrastructure inspection, (iii) Environmental monitoring, (iv) Agricultural field mapping, (v) Real estate and property management, and (vi) Emergency response and disaster relief. In all of these applications, UAVs obtain data with the use of cameras (or other sensors). In order to obtain this data, the UAVs have to cover a specific Region Of Interest (ROI) specified by the user. This ROI can be quite a large area (for instance in agriculture), and, in that case, covering the entire area using a single UAV takes a long time.

A possible approach to reduce coverage time is using multiple UAVs simultaneously. Yet, in order to do so, we need a multi-UAV path planning algorithm. This algorithm must divide the ROI into smaller areas, which can then be surveyed by the different UAVs. As explained in section II, different (multi-UAV) coverage path planning (CPP) algorithms already exist. These algorithms provide the path a UAV must travel. Such path is often calculated by optimizing the flight time (e.g. minimal numbers of turns, shortest distance, etc.), while taking into account some restraints (e.g. covering the entire area, avoiding no-fly zones, avoiding obstacles, etc.). Due to the many available works in the area of CPP, in this work we do not focus on improving upon the current CPPs. Instead, we are interested in the implications and outcome quality of a multi-UAV survey mission. For that reason, we use a simple but effective CPP algorithm, and also capture images while performing a real multi-UAV survey mission. Afterward, we use those images to create one single map of all the images captured from multiple UAVs. We then compare the quality of the map generated by one UAV, and the map captured by multiple UAVs. We hypothesize that we can create a map of similar quality in a considerable shorter time.

The rest of this article is structured as follows: In section II we go over relevant related works. Then, in section III, we explain the path planning algorithm that we used, and the algorithm behind creating one map from different images. Afterward, we use these algorithms in various experiments. We explain our experiments and results in section IV. We perform several experiments by using a simulator but, most importantly, also perform our experiment with real UAVs. Finally, in section V, we conclude our work and discuss future work.

Partially supported by the project REMARKABLE, Marie Skłodowska-Curie grant No. 101086387 under the European Union's Horizon Europe Research and Innovation Programme. This work has been partially supported by R&D project PID2021-122580NB-I00, funded by MCIN/AEI/10.13039/501100011033 and "ERDF A way of making Europe". This research was also partially funded by Fundação para a Ciência e a Tecnologia under Projects 2021.04180.CEECIND and CEECINST/00147/2018/CP1498/CT0015.

II. Related works

As the name suggests, coverage path planning algorithms (CPP) are part of the more general path planning algorithms. Due to its importance in many fields such as: robotics, logistics, navigation, etc. path planning algorithms have already been studied for a long time. For instance, the well-known Dijkstra algorithm [2] was already published in 1959, and has shown its use in numerous fields. Around 2010, UAVs brought, quite literary, an extra dimension to this field. Nowadays, a plethora of papers considering UAVs and path planning have been published. Within the field of UAVs, CPP is a very important topic since it is a key element in many applications (e.g. monitoring tasks). In this work, we will give a brief explanation of the different types of CCP algorithms.

The purpose of a CPP algorithm is to create a path so that a specific Region Of Interest (ROI) can be covered. Often the idea is to cover the ROI only once, i.e. by applying a single CPP. Nevertheless, works considering repetitive covering also exist [3]. The ROI is always a polygon, and earlier works only consider convex polygons since they are slightly simpler to cover. Nevertheless, nowadays, a concave polygon can also be solved. Furthermore, the polygon might include obstacles, no-fly zones etc. Often, path planning is done before the actual flight. Those CPP algorithms are called offline CPP algorithms, and need to have prior knowledge of the entire area (including obstacles). Obviously, offline CPP algorithms only work in static environments. This is, however, not a problem, since many real-world UAV applications, such as inspection flights, precision agriculture, etc., take place in a (reasonably) static environment. Nevertheless, in more urban environments, this is not true. Therefore, we can also find online CPP algorithms [4] which are based on real-time environment data retrieval. They can adapt to unforeseen situations, and do not need prior knowledge of the ROI. Online CPP algorithms do, however, use more CPU power, and often provide less efficient paths [5]. When the ROI is one simple geometric form (e.g. a rectangle), CPP strategies are rather straightforward, and a simple back and forth motion along the longest side of the polygon can be used [6]. However, when the ROI is more complicated, it is often first decomposed into multiple simple shapes. This is often accomplished by decomposing the ROI into a fine-grid based representation, where the cells are all the same size and shape. This method is called the approximate cellular decomposition, and the idea was pioneered by Elfes [7] and Moravac [8]. It is popular because it makes the CPP easier, especially when considering obstacles and no-fly zones. Nonetheless, it is an approximation because using this decomposition method it is impossible to get an exact representation of the ROI. The size of each cell represents a trade-off between the accuracy and performance of this decomposition. When cells are too small, the CPP algorithms will take longer to calculate, but when cells are too large, accuracy is lost. When considering multi-UAV CPP the essential problem remains the same, but we are faced with new challenges and opportunities. The ROI must be split up in various regions for each UAV (note that one UAV might cover multiple regions). Hence, new challenges such as how to split up the ROI, and how to optimally assign the subregions arise. Recently, different authors have worked on those challenges. W. Hu et al. [9] proposed a distributed online solution that aims to minimize task completion time. Since it is an online CPP, their approach is also able to respond to unknown obstacles. L. Marco Andrés et al. [10] specifically deal with multi-UAV CPP for disjoint regions. Their results evidence that adopting a spiral pattern optimizes the cost of the mission, while at the same time improving the task distribution of the mission planning system.

Although various works about multi-UAV CPP algorithms exists, almost all of them verify their approach exclusively using simulation. Our work differs since we actually use real UAVs in an outdoor environment in order to verify our approach.

III. Methods

To complete a multi-UAV survey mission, and to use the imagery data captured in a useful way, we must develop and implement a few key components. Firstly, we must create a CPP algorithm to transform a user-specified ROI into different mission files. A mission file consists of GPS waypoints that one UAV will follow in an orderly manner. We must then upload these mission files to our UAVs, so that they will be able to fly the mission automatically. During the flight, a camera must capture a sufficient amount of photos (as outlined in III-B). After the flight, those images need to be geotagged (i.e. add the GPS location of where the image is taken to the metadata of the image). Finally, we must stitch all of these images together so that we obtain one map of the surveyed area.

A. Coverage Path planning algorithm

Our CPP algorithm consists of two stages. First, the ROI is divided into n subregions, where n is equal to the number of UAVs that will participate in the survey mission. To balance the workload of the different UAVs, we decided to split the ROI into subregions having a similar size. To do so, we developed the iterative algorithm illustrated in Figure 1.

First, we calculated the entire area A of the ROI using the Shoelace theorem [11] (also known as Gauss's area formula). Then, we create a rectangular bounding box around the ROI. We use the width w of the rectangle to estimate where we must split the ROI. The initial cut line is placed at w/n. Then, we calculate the area SA of the subregion (also with the Shoelace theorem). Of course, our initial guess of where to split the ROI might be off. Hence, we compare if the subregion's area SA is indeed close to $1/n$ of the entire area A. If this is not the case, the cut line is moved. Once $SA = \frac{A}{n}$, we continue with the rest of the ROI in a similar fashion until we have n regions with area $\frac{A}{n}$.

After splitting the entire ROI into smaller subsections, we still have to perform the CPP for each subregion. In order to do so, we use a simple back and forth motion. Although this

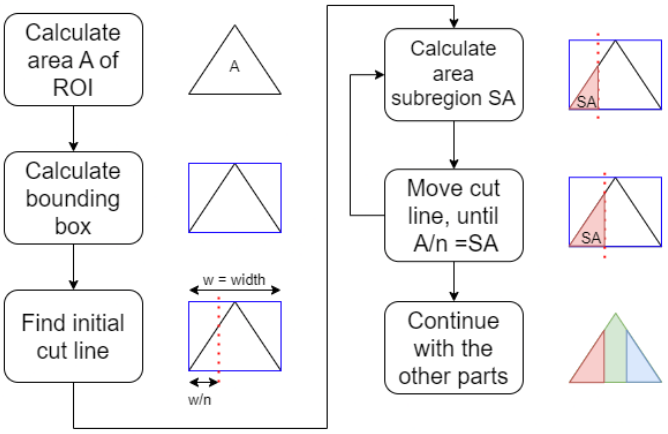


Fig. 1: Splitting the ROI into n parts for each UAV.

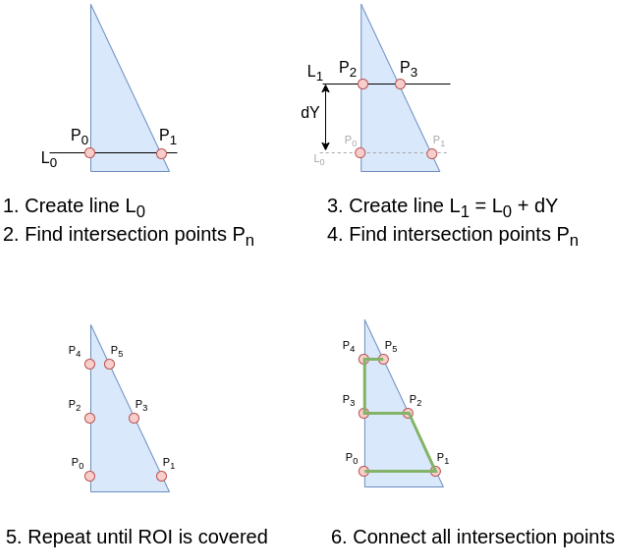
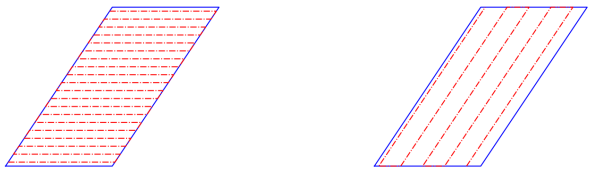


Fig. 2: CPP algorithm: back and forth.



(a) Sweep angle = 0° (b) Sweep angle = 75°

Fig. 3: Influence of the sweep angle on the number of turns.

is a simple procedure, considering just some basic geometry, we have noticed that not many authors include the full details on how to do so. Hence, in this work, we explain this method in more detail. As illustrated in Figure 2, the back and forth method starts by creating a line L_0 and finding the intersection points with the ROI. We can express the ROI as a polygon P defined by its 2D vertices $V = \{V_0, ..., V_n\}$. Each edge E_i is represented as the connection between consecutive vertices: $E_i = \{V_i, V_{i+1}\}$. For each E_i, we need to calculate if there are intersection points P_i, within that line segment. Note that, in the case of a convex polygon, there are at most two intersection points. Then, line L_1 is created by adding the offset dY. This offset corresponds to the distances between the rows. This distance is often determined by the application's requirements. For instance, when taking photos, the camera's field of view and the altitude of the UAV will indicate the distance between the rows. Once all the intersection points are found, we must sort the intersection points of each row based on their x-value. When we alternate between an ascending and descending sort, we obtain a back and forth pattern.

Now that we know the basics of the back and forth CPP, we can improve upon it. As one might imagine, the position of the starting line L_0 can have a significant impact on the end results. In particular, the angle of this line, known as the sweep angle, impacts the number of turns the UAV has to make during the survey. We illustrate this in Figure 3. Since, the UAV has to slow down at each turn, it is in our best interest to minimize the number of turns. In order to do so, we must ensure that L_0 is parallel to the longest edge of the polygon. We can calculate this optimal sweep angle with Algorithm 1 according to [12], [13].

B. Image capturing and geotagging

While the UAVs are flying over the ROI, they will take photos on all predefined positions. Afterward, these photos are stitched together to create one map of the entire ROI. It goes without saying that the specific camera used to take those photos will influence the end result greatly. Hence, we now explain (i) which camera we used, (ii) how we controlled this camera during the flight, and (iii) when we decided to take a photo, and (iv) how we added the location data to the photo for later reference.

In our experiments, we used the SIYI zt30 camera [14]. This lightweight (854 g), high resolution (4K) zoom (focal length 4.8–149 mm) camera, costs around €5000, and consumes on average 9 W. An essential aspect of this camera is its compatibility with the MAVlink protocol [15]; notice that this lightweight message protocol is used to communicate with many flight controllers.

Using the MAVlink protocol, there are two ways to control the camera. We can use the Camera Protocol, which "*is used to configure camera payloads and request their status. It supports photo capture, and video capture and streaming. It also includes messages to query and configure the onboard camera storage.*" [16]. The other way is to include "DO commands" in the mission plan. We chose for this option, since we already have to create a mission plan (in order to cover the entire ROI). To accomplish this, we simply add a waypoint in the mission plan, and on that waypoint we instruct the camera to take a photo. It is of course vital that we take the photo at the right location. These locations on where to

Algorithm 1 Optimal line sweep angle

distance(e, v): Euclidean distance between edge e and vertex v
for edges in the polygon **do**
    max-dist-edge = 0
    for vertex in the polygon **do**
        if distance(edge,vertex) $>$ max-dist-edge **then** max-dist-edge = distance(edge, vertex) opposed-vertex = vertex
        end if
    end for
    if max-dist-edge $<$ optimal-dist **or** is first edge **then** optimal-dist = max-dist-edge line-sweep = direction FROM edge TO opposed-vertex
    end if
end for

take the photos depend on: (i) the camera specifications, (ii) the altitude of the UAV, and (iii) the overlap we want between the adjacent photos. In order to create a detailed map, the desired overlap between the photos is between 75% and 90%. Furthermore, depending on the application, a ground sample distance (GSD) needs to be selected (i.e. the distance between two adjacent pixels in one photo). The GSD is a key limiting factor in the accuracy of the map, and, although it highly depends on the applications, in general a GSD of 1 cm/pixel is recommended [17]. Given this GSD, and the specifications of the camera, the flying altitude can be calculated using the following formula:

$$Altitude = \frac{GSD \times I \times F}{S} \tag{1}$$

$$Where:$$

$$GSD = ground\ sample\ distance\ [m/pixel]$$
$$I = Image\ width\ [pixel]$$
$$F = focal\ length\ of\ camera\ [m]$$
$$S = Sensor\ width\ [m]$$

We can now calculate the true size of each photo by multiplying GSD and I; this way, given a certain percentage overlap between two adjacent photos, we can easily calculate where to take each photo. Finally, when we take the photo, we must include the location of the UAV into the metadata of the photo. The location of the UAV is estimated by the flight controller (using an extended kalman filter) based on (i) GPS data, and (ii) other sensors such as barometers, accelerometers, etc. Depending on the quality of the sensor used, this estimated location has a certain accuracy. In particular, the GPS signal can be quite inaccurate (e.g. 5 meters of error). In order to improve the GPS signal, and thus obtain a higher overall accuracy, more advanced setups use an Real-time kinematic (RTK) correction signal [18].

C. Map generation algorithm

In order to be able to process the images from the drone's camera and validate them, it is necessary to carry out georeferenced mapping. In this section, we will explain the system that we used. Our system is able to effectively map the drone's images and ensure precise georeferencing. Afterward, we will use this system in order to know if the quality of the map coming from one drone is the same as when it is made by multiple drones, using the algorithm proposed in subsection III-A.

In order to build a georeferenced map, we need to perform six main steps, as illustrated in Figure 4 [19], [20]:

1) **Structure from Motion (SfM)**: is a method used to create three-dimensional models from two-dimensional images. It involves extracting features from the images and matching them to create a sparse reconstruction, which is then used to increase the resolution of the model. SfM has been instrumental in improving the resolution of the produced models by a significant amount, and it is a key component in the process of creating digital models based on aerial images. The output of this step are undistorted images and the reconstruction.
2) Using the undistorted images and the reconstruction, a denser point cloud can be computed in this second step, known as **Multi View Stereopsis (MVS)**. It involves analyzing the images to determine the depth and spatial relationships of the objects in the scene. Various methods such as Volumetric Graph-Cuts, Patch-Based Stereo, and Graph Cuts on the Dual of an Adaptive Tetrahedral Mesh [21]–[23] have been used to achieve accurate and dense multi-view stereopsis.
3) **Mesh reconstruction stage** involves creating 2.5D and 3D mesh models, from the dense point cloud generated in the MVS stage. This mesh model represents the surface geometry of the objects and terrain captured in the aerial images. The process of mesh reconstruction is crucial for creating accurate and detailed 3D models from the aerial imagery data.
4) **Texture Reconstruction** is the fourth step, that takes the 2.5D and 3D meshes created in the previous phase, along with the images created in phase one. It involves applying texture to the mesh model using the image set. The texture reconstruction will output a colored texture 2.5D and 3D mesh model.
5) **Georeferencing** is a crucial step in the workflow, as it involves attaching the geolocation of the resulting model into the real-world reference. This process utilizes the extracted coordinates from the images, and the relative position of the images aligned with GNSS information.
6) The last step in creating a georeferenced map is the **orthophoto generation phase**. This phase receives the 3D model from the Texture Reconstruction phase and returns a 2D model (without the Z axis), so that it can be viewed from a top-down view perspective. The orthomap is then converted into a GeoTIFF file to

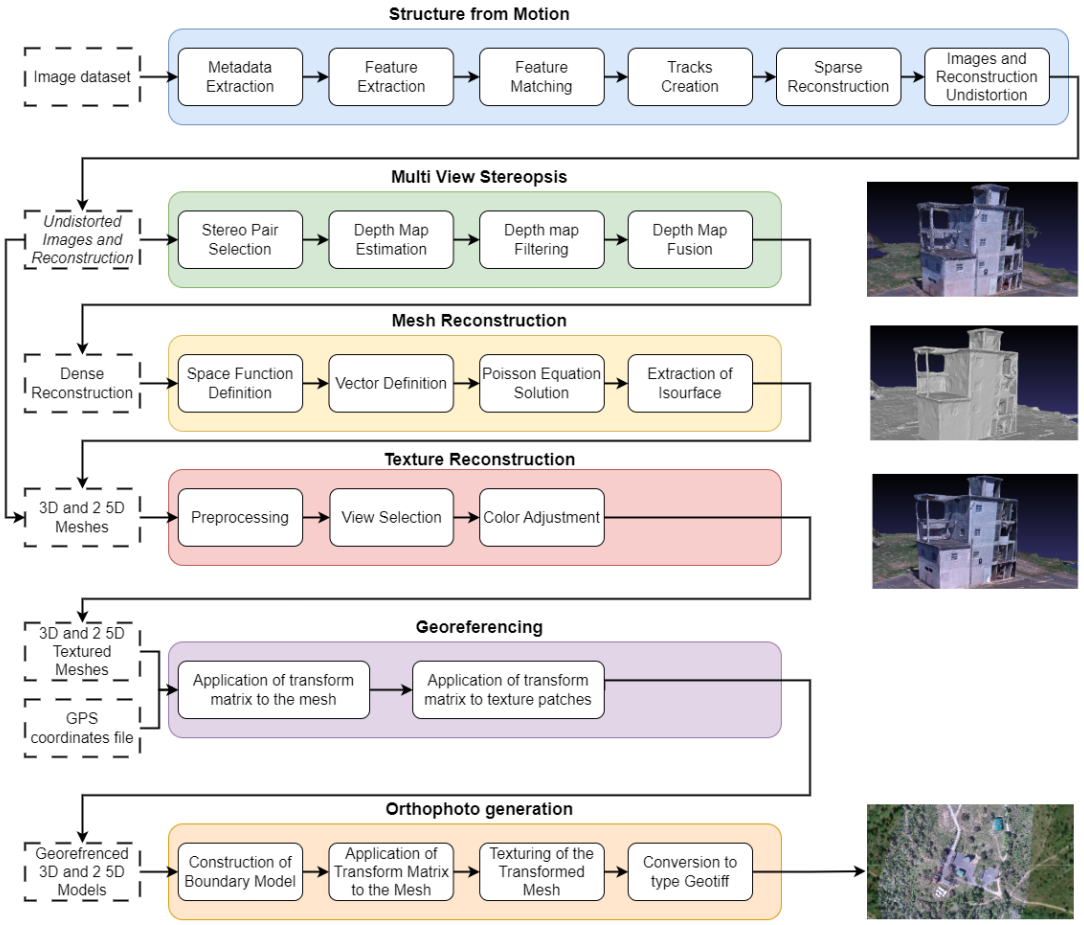


Fig. 4: Step-by-step representation of the map generation algorithm.

store georeferencing information. This is done through a rasterization process using the GDAL library, preserving the georeferenced coordinates of the model.

IV. Experiments and Results

In order to validate our approach, we use two methods: simulation, and real deployment. Simulations allow for (i) testing in a safe environment, and (ii) for scaling. However, simulation can only mimic reality to a certain degree, and it is vital that we deploy our algorithms on real UAVs. For our experiments, we used a proprietary software called BeXStream®, developed by company Beyond Vision [24]. This software allows us to: (i) monitor UAVs, (ii) control real UAVs, (iii) simulate UAVs, (iv) plan missions, (v) process maps, (vi) analyze log files from flights, etc. Pictures of the platform are shown in Figure 6. We extended the mission planning part with the CPP algorithm explained in subsection III-A. We then used the platform to automatically create various mission files of a specific ROI. Afterward, we send these mission files to either simulated or real UAVs.

A. Simulation

Before testing our experiments with real UAVs, we performed various simulations using the BeXStream® platform. We performed these experiments for two reasons. First and foremost, because it is a safe environment where we can perform many tests quickly, and without being preoccupied about real world distortions. Furthermore, it allows us to test with more UAVs, and for larger areas than we would be (currently and legally) able to. We tested our approach in four different ROI: (i) a square of 1 Ha (i.e. 10 000 m^2), (ii) a square of 10 Ha (i.e. 100 000 m^2), (iii) a triangle of 1 Ha, and (iv) a triangle of 10 Ha. We covered each ROI three times with: 1 UAV, 2 UAVs, and 5 UAVs. All UAVs were flying at an altitude of 30 meters, which, due to the specifications of our camera, separated the rows by 9.24 meters. The idea behind this simulation is twofold. First, we want to provide an estimation on how long it takes a number of UAVs to cover a field of a certain area. Second, we want to investigate the impact on irregular ROIs (e.g. triangles). It is worth pointing out that we developed this approach for smart agriculture purposes (hence the units in Ha, and the chosen size of the ROI). Nevertheless, our algorithm can be used for any type of scenario.

We present our results in Table I and Table II. From our experiments, we can conclude three things. First, increasing the size of the ROI also increases the time it takes to cover

the entire area (as expected). However, this is not a 1-by-1 increase. As we can see, in all cases when we increased the size of the area by 10, the time increase was smaller (around $\times 5$). Secondly, the overall time to complete the mission is only slightly dependant on the geometry of the ROI. As we can see, the flight times for the triangle ROI are slightly longer than for the square ROI. When the ROI becomes more irregular, the UAV will have to make more turns. For each turn, the UAV has to slow down. Hence, the total mission time will increase. However, since we use algorithm 1, we can minimize this effect. Finally, we can also observe that increasing the number of UAVs participating in the mission decreases the time to completion significantly. However, like our first observation, doubling the number of UAVs does not cut the mission's time in half. Notice that the factor by which we can reduce the mission's time depends on two elements. First, it depends on the geometry of the ROI. As we can see, the reduction in time is greater in the case of a square ROI. This occurs because, in a square ROI, it is easier to split the ROI into various sections for each UAV. Secondly, we can also observe that the time reduction depends highly on the size of the ROI. The larger the ROI, the more we can benefit from using multiple UAVs at the same time.

TABLE I: Flight time for a square ROI.

# UAVs		1	2	5
1 Ha	Time [min]	08:03	06:20	05:19
	Decrease [%]		21.33	16.05
10 Ha	Time [min]	40:05	27:13	18:51
	Decrease [%]		32.10	30.74

TABLE II: Flight time for a triangle ROI.

# UAVs		1	2	5
1 Ha	Time [min]	08:04	06:33	05:28
	Decrease [%]		18.80	16.54
10 Ha	Time [min]	40:05	29:04	20:41
	Decrease [%]		27.48	28.84

B. Real flight experiments

After verifying that our algorithm is working correctly in simulation, we proceeded to validate our approach using real UAVs. For our real experiment, we used two hexacopters from Beyond Vision®. In Figure 5, we show the two UAVs used, one of which is flying. Due to different restrictions associated to legislation, space, safety, battery consumption, etc., we performed a test covering only a small area (around 8000 m^2) close to Lisbon, Portugal. The main idea of this experiment was to verify whether the mapping results remain consistent even when the photos are taken from multiple UAVs. Hence, we attached cameras to the UAVs, and took geotagged photos (as explained in subsection III-B) while covering the field. Afterward, we created an orthomap, as explained in subsection III-C.

Fig. 5: Two Beyond Vision® UAVs, used to perform the real experiment [25].

Regarding the mapping results, it is necessary to evaluate them not only in visual terms, but also in statistically, so as to achieve an objective validation. With this in mind, a visual assessment of the models was carried out in the first place. The analysis of the models generated from the implemented workflow yielded satisfactory results with no visible artifacts, as shown in Figures 6 a) and b).

In addition, due to geographic coordinates being stored in the image's metadata, the position of the model in the Earth's surface could be evaluated using mapping platforms like Google maps, or interactive maps such as Leaflet. This evaluation is depicted in Figure 7.

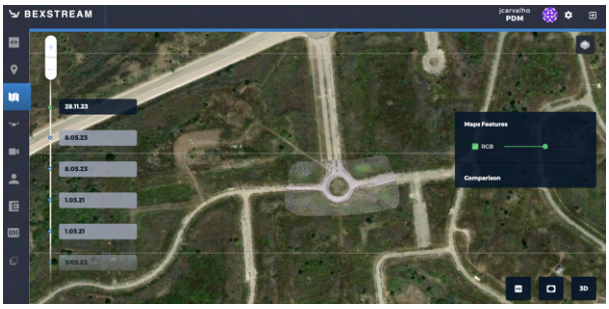
With regard to statistical measures between the two maps created by multiple drones and just one, it is important to analyse the texture map information [28], [29] to compare both maps. Three static textures were extracted using a Gray-Level Co-Occurrence Matrix (GLCM). This algorithm extracts statistical texture features that represent textures based on the relation and the distribution between pixels of a given map.

In this paper, we choose a distance, d, between the pixels equals to 50, the angle $\theta = 0$ and a symmetric matrix to build our GLCM matrix. Then we computed three texture features to statistically compare the values between the two georeferenced RGB maps; this was achieved using the following formulas:

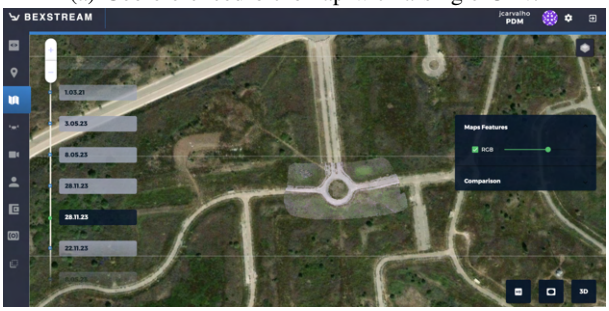
(a) Orthomap with a single UAV.

(b) Orthomap with two UAV.

Fig. 6: Georeferenced RGB Orthomaps.



(a) Georeferenced orthomap with a single UAV.



(b) Georeferenced orthomap with two UAVs.

Fig. 7: Georeferenced models overlapped on Leaflet interactive map from on a platform developed by [26], [27].

$$Contrast = \sum_{i=0}^{N-1}\sum_{j=0}^{N-1} |i-j|^2 \cdot p(i,j) \tag{2}$$

$$Correlation = \sum_{i=0}^{N-1}\sum_{j=0}^{N-1} \frac{(i-\mu_x)(j-\mu_y)\cdot p(i,j)}{\sigma_x \cdot \sigma_y} \tag{3}$$

$$Energy = \sum_{i=0}^{N-1}\sum_{j=0}^{N-1} p(i-j)^2 \tag{4}$$

where $p(i,j) = (i,j)^{th}$ entry in a GLCM normalized matrix, N is the GLCM dimension (in this case equals to 256) and:

$$\mu_x = \sum_{i=0}^{N-1}\sum_{j=0}^{N-1} i \cdot p(i,j) \tag{5}$$

$$\mu_y = \sum_{j=0}^{N-1}\sum_{i=0}^{N-1} j \cdot p(i,j) \tag{6}$$

$$\sigma_x^2 = \sum_{i=0}^{N-1}(\sum_{j=0}^{N-1} p(i,j) - \mu_x)^2 \tag{7}$$

$$\sigma_y^2 = \sum_{j=0}^{N-1}(\sum_{i=0}^{N-1} p(i,j) - \mu_y)^2 \tag{8}$$

The results are shown in Table III; we can see that the values of the three features for the two generated maps remain very close, which makes it possible to conclude that the maps are statistically identical, as intended.

TABLE III: Statistical comparison between the two generated georeferenced RGB maps.

	Contrast	**Energy**	**Correlation**
1 UAV	1396.1	9.1844e-5	0.5946
2 UAVs	1343.9	9.1299e-5	0.6137
|Difference|	52.2	545e-9	0.0191

V. Conclusion and Future Work

Lately, the interest in adopting UAVs for commercial applications is growing rapidly. This trend is especially evident in the field of aerial surveying, where more and more companies are using UAVs to conduct an aerial survey due to their low cost and flexibility. However, most commercially available UAVs can only fly for a short time (between 15 and 35 minutes). This complicates operations when performance large aerial surveys. Hence, in this work, we proposed an approach for multi-UAV aerial surveying. In order to do so, we started by creating a multi-UAV coverage path planning algorithm. This algorithm divides the entire region of interest (ROI) into smaller parts (one for each UAV). Then for each UAV, a back-and-forth method is used to cover each region of interest. While the UAVs are covering the ROI, they take photos. Afterward, these photos are stitched together into one orthomap.

We first validated our approach using simulation. We performed various experiments in order to gain further insight into the time required to cover a ROI. We can conclude that, when using multiple UAVs, we can reduce the time to

cover the entire ROI significantly. However, there is no 1-by-1 relationship, i.e. when using two UAVs to cover a ROI, the time is not split in half. Furthermore, we performed an experiment with real UAVs. In this experiment, we compared the two ortomaps created by multiple drones, and when having just one. Based on both visual and statistical inspection, we can conclude that the there is no difference in the quality of the map. Hence, when using our approach, one can use multiple UAVs to create an orthomap of the ROI without sacrificing quality, as originally intended.

Building on the foundations set in this paper regarding multi-UAV path planning and surveying, there are several groundbreaking directions that can be explored to extend the capabilities and applications of UAV swarms. Here are some innovative ideas to extend the concept in future work:

1) Adaptive Real-time Path Planning: Improve multi-UAV systems with real-time path planning, that adapts to dynamic changes like weather and obstacles, using advanced sensing and AI.
2) Autonomous Swarm Collaboration: Enhance UAV swarm autonomy for complex missions through task allocation, formation flying, and collaborative sensing, without human intervention.
3) Swarm Intelligence for Environmental Monitoring: Utilize swarm intelligence for efficient environmental monitoring, learning and adjusting paths dynamically, useful for tracking climate change effects.

REFERENCES

[1] M. Pérez, F. Agüera, and F. Carvajal, "Low cost surveying using an unmanned aerial vehicle," *The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, vol. XL-1/W2, pp. 311–315, 2013. [Online]. Available: https://isprs-archives.copernicus.org/articles/XL-1-W2/311/2013/

[2] E. W. Dijkstra, "A note on two problems in connexion with graphs," *Numerische mathematik*, vol. 1, no. 1, pp. 269–271, 1959.

[3] P. Fazli, A. Davoodi, and A. K. Mackworth, "Multi-robot repeated area coverage," *Autonomous robots*, vol. 34, pp. 251–276, 2013.

[4] L. Paull, C. Thibault, A. Nagaty, M. Seto, and H. Li, "Sensor-driven area coverage for an autonomous fixed-wing unmanned aerial vehicle," *IEEE Transactions on Cybernetics*, vol. 44, no. 9, pp. 1605–1618, 2014.

[5] J. Valente, J. Del Cerro, A. Barrientos, and D. Sanz, "Aerial coverage optimization in precision agriculture management: A musical harmony inspired approach," *Computers and Electronics in Agriculture*, vol. 99, pp. 153–159, 2013. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0168169913002251

[6] J. Muñoz, B. López, F. Quevedo, C. A. Monje, S. Garrido, and L. E. Moreno, "Multi uav coverage path planning in urban environments," *Sensors*, vol. 21, no. 21, 2021. [Online]. Available: https://www.mdpi.com/1424-8220/21/21/7365

[7] A. Elfes, "Sonar-based real-world mapping and navigation," *IEEE Journal on Robotics and Automation*, vol. 3, no. 3, pp. 249–265, 1987.

[8] H. Moravec and A. Elfes, "High resolution maps from wide angle sonar," in *Proceedings. 1985 IEEE International Conference on Robotics and Automation*, vol. 2, 1985, pp. 116–121.

[9] W. Hu, Y. Yu, S. Liu, C. She, L. Guo, B. Vucetic, and Y. Li, "Multi-uav coverage path planning: A distributed online cooperation method," *IEEE Transactions on Vehicular Technology*, vol. 72, no. 9, pp. 11 727–11 740, 2023.

[10] M. A. Luna, M. S. A. Isaac, M. Fernandez-Cortizas, C. Santos, A. R. Ragab, M. Molina, and P. Campoy, "Spiral coverage path planning for multi-uav photovoltaic panel inspection applications," in *2023 International Conference on Unmanned Aircraft Systems (ICUAS)*, 2023, pp. 679–686.

[11] "Shoelace theorem," accessed the 12th january 2024. [Online]. Available: https://artofproblemsolving.com/wiki/index.php/Shoelace_Theorem

[12] M. Torres, D. A. Pelta, J. L. Verdegay, and J. C. Torres, "Coverage path planning with unmanned aerial vehicles for 3d terrain reconstruction," *Expert Systems with Applications*, vol. 55, pp. 441–451, 2016. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0957417416300306

[13] W. Huang, "Optimal line-sweep-based decompositions for coverage algorithms," in *Proceedings 2001 ICRA. IEEE International Conference on Robotics and Automation (Cat. No.01CH37164)*, vol. 1, 2001, pp. 27–32 vol.1.

[14] SIYI, "SIYI ZT30 camera," https://en.siyi.biz/products/siyi-zt30, 2023, accessed: 01/02/2024.

[15] L. Meier, "MAVLink Developer Guide." [Online]. Available: https://mavlink.io/en/

[16] ——, "Camera Protocol." [Online]. Available: https://mavlink.io/en/services/camera.html

[17] Wingtra, "How ground sample distance (GSD) relates to accuracy and drone ROI." [Online]. Available: https://wingtra.com/how-ground-sample-distance-gsd-relates-to-accuracy-and-drone-roi/

[18] A. Cina, P. Dabove, A. M. Manzino, and M. Piras, "Network real time kinematic (nrtk) positioning – description, architectures and performances," in *Satellite Positioning*, S. Jin, Ed. Rijeka: IntechOpen, 2015, ch. 2. [Online]. Available: https://doi.org/10.5772/59083

[19] A. Vong, J. P. Matos-Carvalho, P. Toffanin, D. Pedro, F. Azevedo, F. Moutinho, N. C. Garcia, and A. Mora, "How to build a 2d and 3d aerial multispectral map?—all steps deeply explained," *Remote Sensing*, vol. 13, no. 16, 2021. [Online]. Available: https://www.mdpi.com/2072-4292/13/16/3227

[20] A. Vong, J. P. Matos-Carvalho, D. Pedro, S. Tomic, M. Beko, F. Azevedo, S. D. Correia, and A. Mora, "Open-source mapping method applied to thermal imagery," in *Intelligent Computing*, K. Arai, Ed. Cham: Springer International Publishing, 2022, pp. 43–57.

[21] G. Vogiatzis, C. Hernandez Esteban, P. H. Torr, and R. Cipolla, "Multiview stereo via volumetric graph-cuts and occlusion robust photo-consistency," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 29, no. 12, pp. 2241–2246, 2007.

[22] S. Shen, "Accurate multiple view 3d reconstruction using patch-based stereo for large-scale scenes," *IEEE Transactions on Image Processing*, vol. 22, no. 5, pp. 1901–1914, 2013.

[23] S. N. Sinha, M. Pollefeys, and P. Mordohai, "Multi-view stereo via graph cuts on the dual of an adaptive tetrahedral mesh," in *2007 11th IEEE International Conference on Computer Vision*. Los Alamitos, CA, USA: IEEE Computer Society, oct 2007, pp. 1–8. [Online]. Available: https://doi.ieeecomputersociety.org/10.1109/ICCV.2007.4408997

[24] B. Vision, "Beyond-vision," https://beyond-vision.pt/, 2023, accessed: 30/01/2024.

[25] D. Pedro, P. Lousã, Á. Ramos, J. P. Matos-Carvalho, F. Azevedo, and L. Campos, "Heifu - hexa exterior intelligent flying unit," in *Computer Safety, Reliability, and Security. SAFECOMP 2021 Workshops*, I. Habli, M. Sujan, S. Gerasimou, E. Schoitsch, and F. Bitsch, Eds. Cham: Springer International Publishing, 2021, pp. 89–104.

[26] M. Pino, J. P. Matos-Carvalho, D. Pedro, L. M. Campos, and J. Costa Seco, "Uav cloud platform for precision farming," in *2020 12th International Symposium on Communication Systems, Networks and Digital Signal Processing (CSNDSP)*, 2020, pp. 1–6.

[27] D. Pedro, J. P. Matos-Carvalho, F. Azevedo, R. Sacoto-Martins, L. Bernardo, L. Campos, J. M. Fonseca, and A. Mora, "Ffau—framework for fully autonomous uavs," *Remote Sensing*, vol. 12, no. 21, 2020. [Online]. Available: https://www.mdpi.com/2072-4292/12/21/3533

[28] J. P. Matos-Carvalho, F. Moutinho, A. B. Salvado, T. Carrasqueira, R. Campos-Rebelo, D. Pedro, L. M. Campos, J. M. Fonseca, and A. Mora, "Static and dynamic algorithms for terrain classification in uav aerial imagery," *Remote Sensing*, vol. 11, no. 21, 2019. [Online]. Available: https://www.mdpi.com/2072-4292/11/21/2501

[29] J. P. Matos-Carvalho, J. M. Fonseca, and A. Mora, "Uav downwash dynamic texture features for terrain classification on autonomous navigation," in *2018 Federated Conference on Computer Science and Information Systems (FedCSIS)*, 2018, pp. 1079–1083.