2021 International Conference on Unmanned Aircraft Systems (ICUAS)
Athens, Greece. June 15-18, 2021

Aircraft Inspection by Multirotor UAV Using Coverage Path Planning*

Patrick Silberberg and Robert C. Leishman[1]

Abstract— **All military and commercial aircraft must undergo frequent visual inspections in order to identify damage that could pose a danger to safety of flight. Currently, these inspections are primarily conducted by maintenance personnel. Inspectors must scrutinize the aircraft's surface to find and document defects such as dents, hail damage, broken fasteners, etc.; this is a time consuming, tedious, and hazardous process. The goal of this work is to develop a visual inspection system which can be used by an Unmanned Aerial Vehicle (UAV), and to test the feasibility of this system on military aircraft. Using an autonomous system in place of trained personnel will improve the safety and efficiency of the inspection process. Open-source software for coverage path planning (CPP) is modified and used to create a path from which the UAV can view the entire top surface of the aircraft. Simulated and experimental flight testing is conducted to validate the generated paths by collecting imagery, flight data, and coverage estimates. Simulation is also used to predict UAV performance for an inspection of a full-size aircraft. Analysis shows that multirotor UAVs are a viable inspection platform for military aircraft.**

Index Terms— **aircraft inspection, coverage path planning (CPP), multirotor UAV**

I. INTRODUCTION

Advances in Unmanned Aerial Vehicle (UAV) technology have allowed drone utilization to expand from a niche community of enthusiasts to a wide array of commercial applications. One of the fields in which UAVs have the potential for growth is aviation maintenance. Visual inspections of aircraft are required in order to identify defects such as lightning strikes and corrosion. These inspections are typically performed by trained human inspectors using ladders or other ground support equipment to view all portions of the aircraft. It is a lengthy process that must be performed regularly to deem the aircraft safe for flight.

Multirotor UAVs are uniquely suited to inspection tasks due to their small size, maneuverability, and ability to carry a range of sensors. They are currently being used in the inspection of bridges [1] [2], oil fields [3], power transmission lines [4], and commercial aircraft [5] [6]. The Department of Defense (DOD) is interested in using autonomous UAVs to improve the safety and efficiency of the inspection process for military aircraft. Maintenance personnel spend hours each week on visual inspections; the amount of time spent scrutinizing aircraft for surface flaws could be drastically reduced by leveraging the speed and mobility of multirotor vehicles. Additionally, these personnel jeopardize their safety every time they climb on top of an aircraft. The Air Force had over 200 fall mishaps a year at airfields between 2015 and 2019, including 15 fatalities and permanent disabilities due to falls [7]. Using UAVs to inspect the aircraft would remove that fall risk and improve aviation safety.

To conduct an inspection, the UAV must fly a collision free path that allows full visibility of the target. During inspections, the target environment can typically be considered static, thus the path can be generated offline by solving a coverage path planning (CPP) problem. CPP is the process of creating a feasible path that contains a set of points from which the UAV can view the entire target environment [8] [9].

This work describes the development a system that enables a UAV to autonomously inspect the top surface of a military aircraft. The proposed system modifies an open-source CPP algorithm [9] [10] to create the inspection paths. Software-in-the-loop simulation and hardware experimentation were used to demonstrate the feasibility of the generated paths and validate the system on a scaled aircraft model. Analysis of these results proves the viability of using multirotor UAVs to conduct inspection missions for defense aircraft.

II. RELATED WORKS

Coverage path planning research has seen substantial progress in recent years as interest in UAV inspections has grown. The variety of missions and path optimization parameters has driven the development of new algorithms and the integration of different sensors. For example, [5] describes a simple system to keep a UAV at a fixed distance from the target, and [11] outlines the capability of conducting a flow penetrant inspection by attaching an ultraviolet light to a quadcopter.

Many algorithms solve the CPP problem by first solving a viewpoint planning problem (VPP) then solving a traveling salesman problem (TSP). The VPP solution provides the viewpoints that meet the coverage requirement, while the TSP result gives the shortest path that visits all of the viewpoints. In [12], the VPP is solved by using combinatorial optimization to minimize the number of viewpoints. Conversely, [13] does not attempt to minimize viewpoints, but tries to minimize the distance between viewpoints in order to find the shortest inspection path. Instead of constraining the VPP to discrete viewpoints, Jing et al. [14] takes advantage of the sensor's ability to continuously capture images. Jing et al. utilize path primitives with a primitive coverage graph to determine the coverage gained between individual viewpoints. A greedy neighborhood search is then used to minimize the inspection path's distance while meeting the coverage requirement [14].

*This work was supported by the Air Force Research Laboratory.

[1] The authors are with the Autonomy and Navigation Technology (ANT) Center, Air Force Institute of Technology, Wright Patterson Air Force Base, OH 45433, USA. patrick.silberberg@usmc.mil, robert.leishman@afit.edu

2021 International Conference on Unmanned Aircraft Systems (ICUAS) | 978-1-6654-1535-4/21/$31.00 ©2021 IEEE | DOI: 10.1109/ICUAS51884.2021.9476718

The CPP algorithm this work is based on is the Adaptive Search Space Coverage Path Planner (ASSCPP) created by Almadhoun et al. [9] [15]. It is an offline, model-based algorithm that focuses on improving the inspection path's accuracy (measured by sensor noise) and coverage. The ASSCPP has three parts: viewpoint generation, coverage path planning, and coverage evaluation. During the viewpoint generation step, the search space surrounding the target is passed through collision, distance, and coverage filters to produce a set of viewpoints that meet the specified requirements. Next, an A* search algorithm is used to identify the viewpoint with the least cost and add it to the path. The coverage generated by the path is evaluated, and viewpoints are added until the coverage goal is met [9] [15]. Modifications this work has made to the ASSCPP are described in the next section.

III. PROPOSED SYSTEM

A. Coverage Path Planning Algorithm

While multiple CPP algorithms were presented in the literature, the ASSCPP by Almadhoun et al. [9] [15] was chosen as the foundation for this work due to its ease of implementation and because it outperformed other open-source planners in terms of path length, accuracy, and number of viewpoints. This subsection describes the modifications this work has made to the ASSCPP. The original open-source algorithm is available at [10], while the updated code presented in this work is available at [16].

The primary contribution made by this work to the algorithm is in the realm of usability. Frequently changed parameters that were previously buried in the code are now consolidated in the launch file for easy modification. Being able to change the target mesh file, UAV starting position, search space details, etc. in a single place without a time-costly recompile facilitates faster, more detailed path computation.

The path planning process was also modified in two ways. First, the code was adjusted to prevent the UAV from returning to a point it had already visited. To avoid repeat points, the A* algorithm used in the path planner was modified. A list of previously visited points was created and constantly updated with the chosen path points. When considering a potential waypoint to add to the path, the new point was checked against the list. If the point being considered had already been visited, the algorithm proceeded to check the potential waypoint with the next lowest cost until a unique location was found. This change led to more intuitive inspection paths.

The second major modification to the ASSCPP was the addition of a parameter that lets the user choose whether or not to allow multiple orientations at the same point. This distinction allows two different types of paths to be created: one with the possibility of stopping and rotating to a new orientation at waypoints and one without. The purpose of adding this capability was to facilitate the two different types of path following options described in Section III-B.3. In-depth descriptions of these changes and documentation on the algorithm's functions can be found in [17], while an open-source implementation of the code used in this work is available at [16].

B. Experimental Setup

1) Test Item Description: The UAV used during real-world testing was a 3DR X8 coaxial octorotor. The X8 had a Prosilica GE 1660C 1.9 megapixel camera with an 8 mm lens mounted on the nose at a fixed -30° angle. The camera was set to capture images at 5 frames per second (fps). A Pixhawk 2 loaded with ArduPilot Copter [18] was used for flight control and navigation. Position and orientation data were provided by a VICON motion capture system; the accuracy of the VICON system allowed the UAV to consistently hit waypoints with a 15 cm radius. Autopilot and sensor data were stored in Lightweight Communication and Marshalling (LCM) logs on the SD card of an Odroid XU4 for postflight analysis.

A $\frac{1}{7}$ scale F-15 model measuring 2.78 m x 1.87 m x 0.47 m was used as the inspection target. The model was mounted on a 1.45 m stand in order to allow the UAV to fly outside of ground effect. A ground control station (GCS) running Mission Planner was used to send commands to the UAV and to monitor telemetry.

2) Simulation Environment: Gazebo [19] was chosen as the simulation environment for this work due to its ability to work with ROS and ArduPilot. The SwiftGust plugin [20] was used integrate ArduPilot's Simulation-in-the-Loop (SITL) software with Gazebo. Simulation was used to ensure the UAV would not collide with the aircraft, check that the UAV accurately hit the generated waypoints, obtain approximate flight times for each path, and estimate whether the paths provided the desired coverage.

A stock world and an Iris quadcopter were modified to meet the testing objectives. The Iris's sensor parameters, such as field of view (FOV), depth of field (DOF), and frame rate, were changed to match the camera used during experimentation. A mesh model of an F-15 was obtained, scaled appropriately, and inserted into the simulation environment in order to recreate the experimental inspections. An F-35 was chosen as the target for the full-scale inspection because it is an airframe that is relevant to multiple branches of the armed forces, and a mesh model was readily available. The simulated models were untextured, as the scope of this work was only concerned with coverage estimates. Future work will seek to incorporate textured mesh models into the simulation in order to be able to identify both coverage area and defects in simulation.

3) Flight Paths and Path Following Methods: Paths at four distances away from the F-15 were created to gather diverse imagery sets and test UAV performance. The paths were 3 m, 2 m, 1 m, and 0.4 m away from the model, with the distance representing the closest a waypoint in the path could be to the model. These paths were flown in both simulation and during the experimental flight testing. The 3 m outer limit was dictated by the size of the testing facility, but it is also a reasonable maximum distance in terms of image quality from which to inspect a full-scale aircraft. The 0.4

m path was chosen as the inner limit to test the precision of the UAV and show that it is capable of maneuvering in close proximity to an aircraft. Additionally, at 0.4 m the portion of the F-15 model in the sensor's FOV will be representative of what the sensor would see on a full-scale aircraft from 3 m away. The 2 m and 1 m paths were chosen to allow for fine tuning of any control issues prior to the 0.4 m flight. These paths will also provide flight data and imagery that will help determine at which distance the inspection of full-scale aircraft might best be flown.

To fly the paths described above, the UAV used ArduPilot's stock path following algorithms. ArduPilot Copter offers two different path following options [18]. In the first method, hereafter referred to as "continuous", waypoints are considered complete when the virtual target the UAV is following hits the waypoint. Depending on the velocity and control tuning of the UAV, waypoint completion can occur well before the UAV reaches the actual waypoint. The second option, hereafter called "noncontinuous", requires a delay at the waypoints, and a waypoint is considered complete once the UAV crosses the waypoint radius. However, this option allows the user to specify radii as small as 5 cm, compared to a minimum of 1 m for the continuous method. Thus the noncontinuous method results in slower, more accurate flights while the continuous method provides faster but less precise paths.

4) System Relationships: Figure 1 shows a simplified diagram of how the ASSCPP interacts with the other systems involved in flight testing. The user specifies the input parameters, and the ASSCPP generates the path prior to the inspection. Next, the waypoints are loaded into the ground control station (GCS); Mission Planner was used as the GCS for this work. The GCS can then communicate with the autopilot to control the UAV and record data during either simulated or real-world flights.

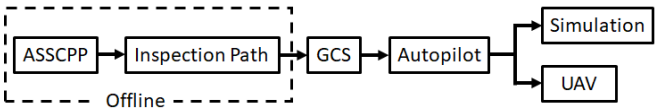

Fig. 1: A flow diagram of the major path generation and flight execution systems.

C. Test Objectives

The main objective of this work was to determine the feasibility of using multirotor UAVs to conduct visual inspections of military aircraft. To be a viable inspection platform, the UAV must be able to accurately follow the generated inspection path, capture images of sufficient quality for defect identification, and collect imagery that fully covers the desired area.

Both the continuous and noncontinuous path following methods were tested. The 0.4 m and 1 m paths were flown with only the noncontinuous option, as a greater degree of accuracy was required at these distances. However, the 2 m and 3 m paths were flown using both the continuous and noncontinuous methods in order to compare flight performance as well as image quality. To test whether or not the two methods were sufficiently accurate, the UAV's flight path was compared to the inspection path. Location errors were calculated by determining the minimum distance between the UAV's actual path and each generated waypoint.

To determine if the image quality was suitable for defect identification, the collected imagery was manually assessed by the ground station operator (GSO). Similarly, the coverage area generated by the inspection was manually estimated by the GSO. To separate usable and non-usable images, open-source code [21] was used to detect blurry images. The program used the OpenCV [22] library and applied the variance of the Laplacian method to determine image blurriness. A threshold of 100 was used for the images taken from 0.4 m and 1 m away from the F-15 and a threshold of 120 for the images taken from 2 m and 3 m. The threshold was determined by the GSO after visually assessing the image quality.

Finally, to provide a realistic expectation for full-scale aircraft inspections, the simulation environment had to be shown to produce a good approximation of real-world performance. To test this, the tuned autopilot parameters from the experimental test flights were placed back into simulation. The flight paths and imagery captured from the same paths were compared to determine if the simulation environment gave realistic results.

IV. RESULTS AND DISCUSSION

A total of 45 real-world inspection flights were performed on the F-15 model. During these flights, the UAV's speed, acceleration, and control gains were adjusted in order to fine tune the UAV's performance. Shown in the subsections below are flight data and imagery representative of the performance that a working prototype would provide.

A. Experimental Results

1) Path Following: One of the research objectives was to determine whether ArduPilot's stock path following algorithms were able to accurately follow the CPP generated inspection paths. Both the continuous and noncontinuous methods were evaluated. Examples of the resulting paths using each method are shown in Figures 2 and 3. The UAV's trajectory for the 3 m path while using the noncontinuous method is shown in Figure 2, while Figure 3 depicts the UAV's path using the continuous method.

To assess whether the UAV adequately hit the waypoints while following the inspection path, the UAV's location errors for each path were calculated. Table I shows the minimum, maximum, and average distances from the UAV's trajectory to the generated waypoints. Taking into account the number of waypoints in each path, the UAV had an average error of 5.7 cm over all four paths using the noncontinuous path following method. For the two flights that utilized the continuous method, the average error was 21.3 cm. The results show that the noncontinuous method does allow accurate path following. However, the same cannot be said

TABLE I: Experimental Flight Path Data

Noncontinuous Flights							
Path	**Path Length**	**Dist Flown**	**Path Flt Time**	**Total Flt Time**	**Min Error**	**Mean Error**	**Max Error**
0.4 m	13.48 m	15.75 m	1 min 21 sec	2 min 5 sec	1.6 cm	5.1 cm	14.7 cm
1 m	15.83 m	16.87 m	1 min 44 sec	2 min 22 sec	1.0 cm	4.7 cm	12.4 cm
2 m	26.38 m	30.34 m	2 min 11 sec	2 min 39 sec	2.2 cm	8.0 cm	13.4 cm
3 m	25.03 m	27.35 m	2 min 10 sec	2 min 48 sec	0.8 cm	4.9 cm	9.3 cm

Continuous Flights							
Path	**Path Length**	**Dist Flown**	**Path Flt Time**	**Total Flt Time**	**Min Error**	**Mean Error**	**Max Error**
2 m	26.38 m	26.55 m	1 min 14 sec	1 min 51 sec	4.5 cm	17.5 cm	36.7 cm
3 m	25.03 m	23.99 m	59 sec	1 min 37 sec	5.7 cm	24.7 cm	58.5 cm

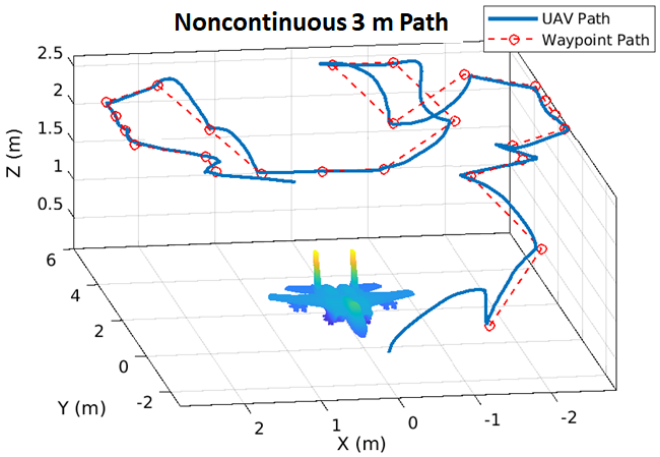

Fig. 2: The 3 m noncontinuous flight path (blue) compared to the actual waypoint path (red). The UAV flew within 10 cm of all the waypoints and had an average error of 4.9 cm.

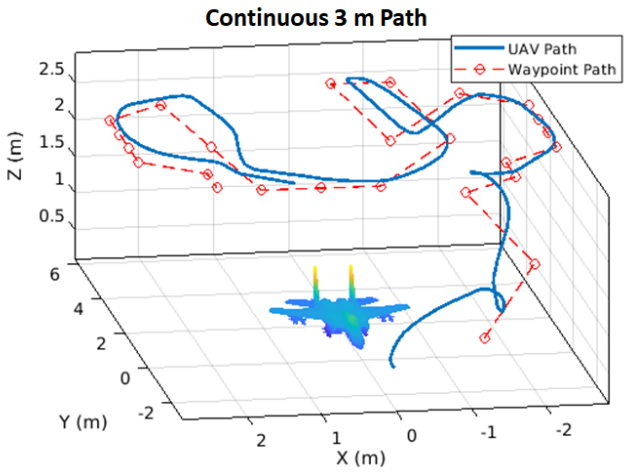

Fig. 3: The red circles are the CPP generated waypoints for the 3 m path, and the blue line depicts the UAV's trajectory while following the reference path using the continuous method. On average, the UAV flew within 24.7 cm of the waypoints during the continuous flight, which is 5 times the average error of the 3 m noncontinuous flight.

about the continuous method. Although the UAV roughly followed the inspection path's shape, the continuous method resulted in errors of up to 58.5 cm.

The noncontinuous method was more accurate, but because it required a delay at each waypoint, it was also slower. As seen in Table I, the continuous method resulted in path flight times that were 43-55% faster than its noncontinuous counterpart. The UAV was able to fly faster during the continuous flights as well: 0.35 m/s on average versus an average speed of 0.2 m/s during the noncontinuous flights.

2) Coverage and Imagery: The desired coverage area for the experimental inspection paths was the top surface of the F-15. Both the continuous and noncontinuous flights were found to meet that coverage goal. However, it is likely that the continuous flight only provided full coverage due to the scale of the F-15 model and the 2-3 m distance between the UAV and the model. Due to the large errors associated with the continuous flights, complete coverage for a full-scale aircraft using the continuous method cannot be guaranteed.

The imagery collected during the experimental inspections was sufficient for defect identification. Large defects, such as a misaligned panel, could be seen on the F-15 from 2-3 m. Figure 4 shows an image taken from 3 m; a gap can be seen on the fuselage's center panel between the 5 and 0. At 1 m and closer, the image resolution allowed identification of defects around the size of a screw head. For example, individual rivets can be seen on the vertical fin in Figure 5. The blur detection program was applied to the images, and the results are shown in Table II. When the F-15 occupied a majority of the camera's FOV, during the 0.4 m and 1 m paths, 85-90% of the images were usable.

TABLE II: F-15 Experimental Imagery

Path	**Total # Images**	**# Blurry**	**% Blurry**
0.4 m Noncontinuous	358	50	14%
1 m Noncontinuous	404	40	10%
2 m Noncontinuous	448	120	27%
2 m Continuous	313	134	43%
3 m Noncontinuous	574	316	55%
3 m Continuous	144	62	43%

B. Simulation Results

While simulations were conducted prior to experimental flight testing to ensure feasibility of the paths, those results are not presented here. The autopilot parameters determined during experimental flight testing were applied

Fig. 4: Imagery from 3 m away. At this distance, the gap caused by misaligned center panel can be seen and the letters on the vertical fin can be read.

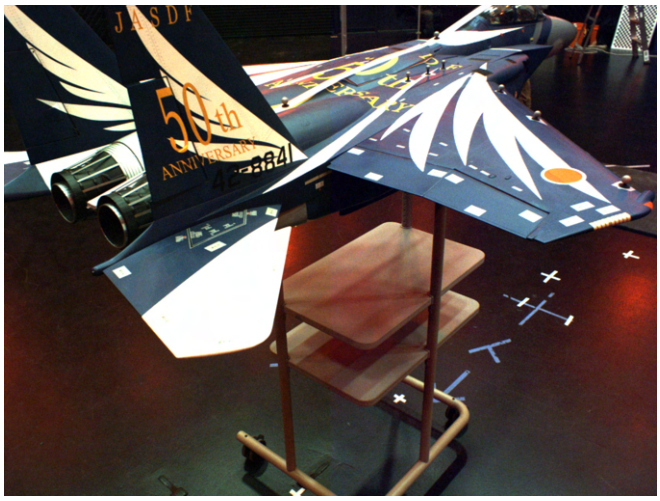

Fig. 5: Imagery from 1 m away. Rivets on the vertical fins and the leading edge of the wings can be identified from 1 m.

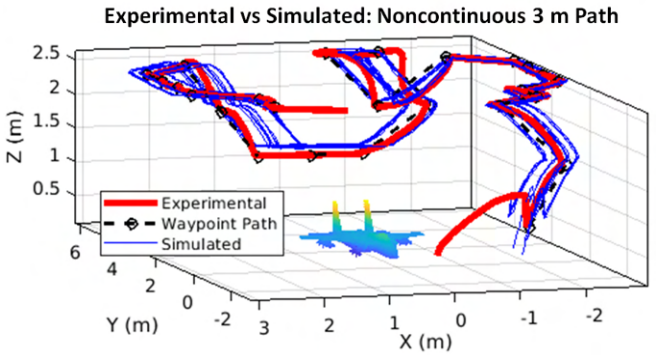

Fig. 6: The reference (black), simulated (blue), and experimental (red) 3 m noncontinuous F-15 model inspection paths. The simulated inspection was repeated 15 times, and the average of these paths closely matched the reference path.

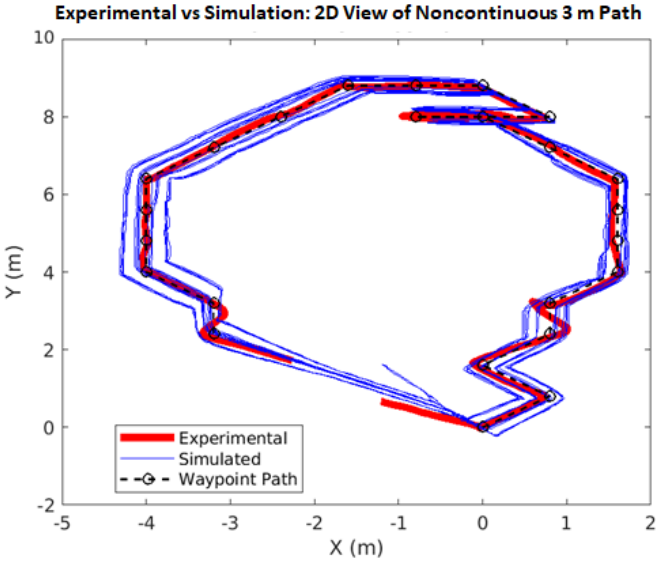

Fig. 7: A top down view of the paths shown in Figure 6. The experimental path (red) remains in the bounds formed by the simulated paths (blue) in all but a few places.

to the simulated UAV, and the simulations were run again. The results from these updated simulations are shown below. By using the experimental flight parameters in simulation, a comparison between experimental and simulated results can be made for the F-15 flight paths, and more realistic results can be provided for the F-35 inspection.

The simulations are not an exact replica of the experimental flights, however. The UAV used in simulation was a Iris quadcopter, which has different dynamics than the X8 octocopter. Matching the autopilot parameters may approximate similar flight conditions, but they will not be exact. Additionally, the simulation used GPS localization, which was not as accurate as the experimental VICON localization. In experimental flights, the waypoint radius was set to 15 cm. In simulation, the UAV was not able to attain that degree of accuracy. In order to allow the Iris to consistently hit the waypoints without hunting for them, the simulation waypoint radius had to be increased to 35 cm.

1) F-15 Simulations: To determine if the experimental results were accurately represented by simulation, the 3 m F-15 inspection path was simulated 15 times using the noncontinuous path following technique. The results of the simulations are plotted against the reference path and experimental path in Figure 6. Due to the larger waypoint radii and stochastics inherent in the simulation environment, the 15 simulation paths fell into four bands around the reference path. The bands deviated from the reference path primarily in the X-Y plane, which can be seen in Figure 7. The UAV's altitude, however, was consistent throughout the simulations. Despite the variation of the simulations, the average of the 15 simulated paths closely matched the experimental and reference paths; the averaged simulated path had a mean location error of 9 cm.

The individual simulated paths differed from the experimental path in a few key ways. On average, the simulated distances flown were 2.4 m (9%) shorter, 24 seconds (18%) faster, and the individual paths had a 3.7 times larger location error (18.1 cm vs 4.9 cm) than the experimental path. These discrepancies are primarily due to the larger waypoint radii, which allowed the simulated UAV to fly a faster and less precise path. The major difference between the experimental and simulated paths, excluding the effects caused by waypoint radius size, was turning ability. During large turns in the experimental flight, the UAV tended to overshoot the waypoint a small distance before curving towards the next waypoint. The simulated UAV was able to turn sharper, resulting in a straighter path between waypoints.

Taken as a whole, however, the simulations provide realistic bounds for the real-world flight. Although the real-world UAV had a more rounded path with more overshoots, the experimental path departs the bands formed by the simulations only four times. Thus, future researchers will be able to use simulation to provide realistic expectations for their experimental paths, thereby reducing the amount of physical testing required.

A comparison between the simulated and real-world results was also made for the imagery captured during the inspections. Figure 8 shows two images taken from roughly the same point on the 1 m noncontinuous flight during a simulated and experimental inspection. Because the simulation environment allows the user to change the sensor's parameters, the simulated images provide a good approximation of the portion of the aircraft that a real-world camera would capture. Thus, future developers can use the simulated sensor to validate an inspection path's coverage, which also decreases the amount of real-world testing needed.

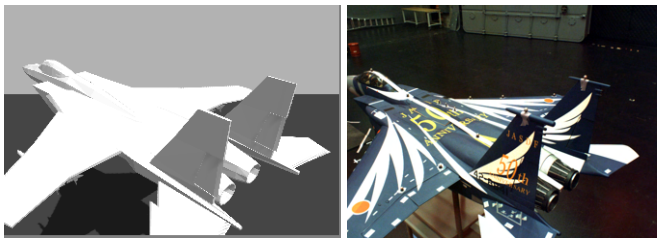

Fig. 8: Imagery from simulated and experimental flights taken from similar locations along the 1 m noncontinuous F-15 inspection path. The simulated imagery provides a realistic expectation of what portion of the aircraft will be in the sensor's field of view.

2) F-35 Simulations: The performance of a UAV conducting an inspection of a full-scale F-35 can be estimated now that the simulation is known to provide reasonable approximations of real-world results. An inspection path 3 m away from the F-35 was created, and then simulated 15 times using both the noncontinuous and continuous methods. Figure 9 shows the CPP generated path as well as the noncontinuous simulation results. As with the F-15 simulations, there were variations in the paths, but the simulations closely matched

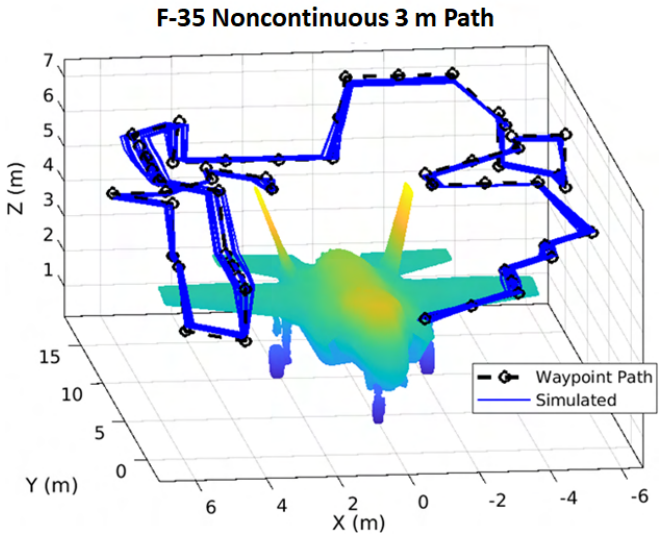

Fig. 9: The black line depicts the F-35 inspection path generated by the CPP algorithm. The inspection was simulated 15 times using the noncontinuous path following method, and the results are shown in blue. A real-world UAV could complete this inspection path with a total flight time of around 7 minutes.

the reference path when averaged.

As expected, the noncontinuous flights followed the path more accurately, with almost half the average error for the individual paths as the continuous flights (18.7 cm vs 34.5 cm). However, the continuous flights took half as long to navigate the waypoints (2 min 23 sec vs 4 min 59 sec). From these results, the total flight time for a real-world inspection of an F-35 can be estimated. The noncontinuous method would result in a total inspection time of around 7 minutes, and the total flight time using the continuous method would be around 4 minutes. Battery analysis of the X8's experimental flights showed it had a maximum flight time of 5 minutes 55 seconds. Thus, an inspection of a full-scale F-35 is feasible for the X8 using the continuous method, but the required endurance for the noncontinuous method is beyond the X8's ability. However, the longer endurance is completely realistic for other multirotors. For example, a Tarot T960 hexacopter with T-Motor MN5208 340 kV motors, 18 inch propellers, and a 6S 10,000 mAh battery could fly for 10 minutes. The hexacopter could also produce 17 lbs of thrust; more than enough to carry the updated sensors and computing power a fully functional prototype would require.

The imagery that would be collected during a real-world F-35 inspection can also be approximated by applying the results from the experimental F-15 imagery analysis. Assuming a 15% blur rate, at 5 fps the UAV would capture around 1,300 usable images during a noncontinuous inspection or 650 during a continuous flight. Simulation did show that in this case the continuous flight provided full coverage of the F-35's top surface. For future prototype design purposes,

however, it would be best to conservatively plan on the higher data requirement of the noncontinuous paths.

V. CONCLUSION

The goal of this work was to test the feasibility of a system that allows a multirotor UAV to inspect a military aircraft. To accomplish this objective, an open-source CPP algorithm was modified and used to generate viewpoints for inspection paths. An updated version of the open-source algorithm is available at [16]. The paths were tested in a Gazebo simulation environment using a simple quadcopter with an ArduPilot plugin. Once satisfactory performance in simulation was observed, real-world flight tests were conducted. The UAV employed in this work was an X8 octocopter equipped with a 1.9 megapixel camera. The UAV utilized a VICON motion capture system for localization, and the inspection target during experimental flight testing was a $\frac{1}{7}$ scale F-15 model. Paths at distances of 0.4 m, 1 m, 2 m, and 3 m away from the model were flown in order to test navigation precision and acquire diverse sets of imagery. The simulations were repeated with the tuned autopilot parameters in order to compare sensor and flight path data from the simulated and real-world flights. Finally, the simulated inspection of a full-scale F-35 was performed in order to predict real-world performance.

The proposed system was found to be suitable for the visual inspection of military aircraft. The combination of VICON localization and the ArduPilot path following algorithms allowed the UAV to fly accurately in close proximity to the aircraft. On average, the UAV flew within 5.7 cm of the desired waypoints using the noncontinuous path following technique. The imagery collected during the experimental flights met the goal of providing coverage for the entire top surface of the aircraft. Screws and small defects could be identified by the sensor at a distance of 1 m from the aircraft. Additionally, the simulation environment was shown to provide realistic expectations for the path flown and imagery taken during real world inspections. Analysis of the F-35 simulations show that a multirotor UAV could complete an inspection of the top surface of the aircraft in 4 minutes using the continuous method, or 7 minutes with the noncontinuous method.

Several options for future research exist that would improve the proposed system. The CPP algorithm could be improved by optimizing the path by solving a Traveling Salesman Problem after the waypoints are chosen. Different localization methods, such as real-time kinematic (RTK) positioning or vision based navigation, could be implemented to facilitate more flexibility in inspection locations. The image quality could by improved by placing the sensor on a gimbal, which would allow the sensor to match the angle of the surface it is capturing. Finally, a textured mesh model could be implemented into the simulation environment. The higher fidelity model would facilitate the training of a neural network to perform imagery analysis such as defect identification and coverage estimation through 3D reconstruction.

REFERENCES

[1] Najib Metni and Tarek Hamel. A UAV for bridge inspection: Visual servoing control law with orientation limits. *Automation in Construction*, 2007.

[2] P. J. Sanchez-Cuevas, G. Heredia, and A. Ollero. Multirotor UAS for bridge inspection by contact using the ceiling effect. *2017 International Conference on Unmanned Aircraft Systems, ICUAS 2017*, pages 767–774, 2017.

[3] Hongjun Wang and Rong Ye. Three-dimensional Local Path Planning of Robot Based on AR-ANT Algorithm and B-spline Curve. *Proceedings of 2019 IEEE International Conference on Mechatronics and Automation, ICMA 2019*, (1):615–620, 2019.

[4] Zifa Liu, Xinyue Wang, and Yunyang Liu. Application of Unmanned Aerial Vehicle Hangar in Transmission Tower Inspection Considering the Risk Probabilities of Steel Towers. *IEEE Access*, 7:159048–159057, 2019.

[5] Umberto Papa and Salvatore Ponte. Preliminary design of an unmanned aircraft system for aircraft general visual inspection. *Electronics (Switzerland)*, 7(12), 2018.

[6] Matthieu Claybrough. System and method for automatically inspecting surfaces. *Patent US 10377485B2*, 2016.

[7] Air Force Safety Center. Fall Prevention Focus. https://www.safety.af.mil/Divisions/Occupational-Safety-Division/Fall-Prevention-Focus/. Accessed: 11.24.2020.

[8] Tauã Cabreira, Lisane Brisolara, and Paulo R. Ferreira Jr. *Survey on Coverage Path Planning with Unmanned Aerial Vehicles*, volume 3. 2019.

[9] Randa Almadhoun, Tarek Taha, Lakmal Seneviratne Jorge Dias, and Yahya Zweiri. Coverage Path Planning for Complex Structures Inspection Using Unmanned Aerial Vehicle (UAV). In *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*. Springer International Publishing, 2019.

[10] Randa Almadhoun. Adaptive Search Space Coverage Path Planner (ASSCPP). https://github.com/kucars/asscpp. Accessed: 10.26.2020.

[11] Konstantinos Malandrakis, Al Savvaris, Jose Angel Gonzalez Domingo, Nick Avdelidis, Panagiotis Tsilivis, Florence Plumacker, Luca Zanotti Fragonara, and Antonios Tsourdos. Inspection of aircraft wing panels using unmanned aerial vehicles. *5th IEEE International Workshop on Metrology for AeroSpace, MetroAeroSpace 2018 - Proceedings*, pages 56–61, 2018.

[12] Wei Jing, Joseph Polden, Wei Lin, and Kenji Shimada. Sampling-based view planning for 3D visual coverage task with unmanned aerial vehicle. *IEEE International Conference on Intelligent Robots and Systems*, 2016-Novem:1808–1815, 2016.

[13] Andreas Bircher, Kostas Alexis, Michael Burri, Philipp Oettershagen, Sammy Omari, Thomas Mantel, and Roland Siegwart. Structural inspection path planning via iterative viewpoint resampling with application to aerial robotics. *Proceedings - IEEE International Conference on Robotics and Automation*, 2015-June(June):6423–6430, 2015.

[14] Wei Jing, Di Deng, Zhe Xiao, Yong Liu, and Kenji Shimada. Coverage Path Planning using Path Primitive Sampling and Primitive Coverage Graph for Visual Inspection. *IEEE International Conference on Intelligent Robots and Systems*, pages 1472–1479, 2019.

[15] Randa Almadhoun, Tarek Taha, Dongming Gan, Jorge Dias, Yahya Zweiri, and Lakmal Seneviratne. Coverage Path Planning with Adaptive Viewpoint Sampling to Construct 3D Models of Complex Structures for the Purpose of Inspection. *IEEE International Conference on Intelligent Robots and Systems*, pages 7047–7054, 2018.

[16] Patrick Silberberg. Open-source implementation of the proposed CPP. https://github.com/psilberberg/asscpp. Accessed: 3.1.2021.

[17] Patrick Silberberg. *Aircraft Inspection by Multirotor UAV Using Coverage Path Planning*. Master's thesis, Air Force Institute of Technology, 2021.

[18] ArduPilot. Open Source UAV Autopilot. https://ardupilot.org/. Accessed: 11.24.2020.

[19] Gazebo Simulation Environment. hthttp://gazebosim.org/. Accessed: 12.23.2020.

[20] SwiftGust. ArduPilot Gazebo Plugin and Models. https://github.com/SwiftGust/ardupilot$_gazebo$. $Accessed$: 12.23.2020.

[21] Adrian Rosebrock. Blur Detection with OpenCV. https://www.pyimagesearch.com/2015/09/07/blur-detection-with-opencv/. Accessed: 1.11.2021.

[22] OpenCV. https://opencv.org/. Accessed: 1.25.2021.