sensors

Article

Performance Analysis of Localization Algorithms for Inspections in 2D and 3D Unstructured Environments Using 3D Laser Sensors and UAVs

Paul Espinosa Peralta [1], Marco Andrés Luna [1], Paloma de la Puente [1], Pascual Campoy [1,*], Hriday Bavle [2], Adrián Carrio [1,3] and Christyan Cruz Ulloa [1]

1 Centro de Automática y Robótica (CAR), Universidad Politécnica de Madrid (CSIC-UPM), 28006 Madrid, Spain; paul.espinosa.peralta@alumnos.upm.es (P.E.P.); marco.lunaa@alumnos.upm.es (M.A.L.); paloma.delapuente@upm.es (P.d.l.P.); adrian.carrio@dronomy.es (A.C.); christyan.cruz.ulloa@alumnos.upm.es (C.C.U.)
2 Automation and Robotics Research Group, Interdisciplinary Centre for Security, Reliability and Trust, University of Luxembourg, 1855 Luxembourg, Luxembourg; hriday.bavle@uni.lu
3 Dronomy, Paseo de la Castellana 40, 28046 Madrid, Spain
* Correspondence: pascual.campoy@upm.es

Abstract: One of the most relevant problems related to Unmanned Aerial Vehicle's (UAV) autonomous navigation for industrial inspection is localization or pose estimation relative to significant elements of the environment. This paper analyzes two different approaches in this regard, focusing on its application to unstructured scenarios where objects of considerable size are present, such as a truck, a wind tower, an airplane, a building, etc. The presented methods require a previously developed Computer-Aided Design (CAD) model of the main object to be inspected. The first approach is based on an occupancy map built from a horizontal projection of this CAD model and the Adaptive Monte Carlo Localization (AMCL) algorithm to reach convergence by considering the likelihood field observation model between the 2D projection of 3D sensor data and the created map. The second approach uses a point cloud prior map of the 3D CAD model and a scan-matching algorithm based on the Iterative Closest Point Algorithm (ICP) and the Unscented Kalman Filter (UKF). The presented approaches have been extensively evaluated using simulated as well as previously recorded real flight data. We focus on aircraft inspection as a test example, but our results and conclusions can be directly extended to other applications. To support this assertion, a truck inspection has been performed. Our tests reflected that creating a 2D or 3D map from a standard CAD model and using a 3D laser scan on the created maps can optimize the processing time, resources and improve robustness. The techniques used to segment unexpected objects in 2D maps improved the performance of AMCL. In addition, we showed that moving around locations with relevant geometry after take-off when running AMCL enabled faster convergence and high accuracy. Hence, it could be used as an initial position estimation method for other localization algorithms. The ICP-NL method works well in environments with elements other than the object to inspect, but it can provide better results if some techniques to segment the new objects are applied. Furthermore, the proposed ICP-NL scan-matching method together with UKF performed faster, in a more robust manner, than NDT. Moreover, it is not affected by flight height. However, ICP-NL error may still be too high for applications requiring increased accuracy.

Keywords: UAV; 3D sensor; particle filter; localization

check for updates

Citation: Espinosa Peralta, P.; Luna, M.A.; de la Puente, P.; Campoy, P.; Bavle, H.; Carrio, A.; Cruz Ulloa, C. Performance Analysis of Localization Algorithms for Inspections in 2D and 3D Unstructured Environments Using 3D Laser Sensors and UAVs. *Sensors* **2022**, *22*, 5122. https://doi.org/10.3390/s22145122

Academic Editor: Sindhuja Sankaran

Received: 29 April 2022
Accepted: 2 July 2022
Published: 7 July 2022

1. Introduction

Nowadays, industrial inspections using UAVs are common. In the last years, they have significantly improved in terms of flexibility, efficiency, less cost, etc., but more advances are necessary. Additionally, some of them are executed in dangerous and difficult environments for humans. Several tasks are executed in buildings [1], industries [2,3], power grids [4–6], agriculture areas [7–10], solar plants [11], and wind turbines [12,13]. Therefore, the development of autonomous robotic systems has become one of the most critical challenges in reducing human intervention in different types of missions.

Localization in these circumstances is often hard: the chosen method or algorithm must solve common problems such as initial positioning and kidnapping. During an autonomous inspection task, the dynamic nature of many unstructured environments is one of the major issues: objects can change their position, and new objects may appear in the scene. Updating the initial map for each new mission usually requires a lot of time and resources. In this context, the estimation of the robot pose considering the uncertainty in the sensors data and the vehicle dynamics within an unstructured environment is widely studied. A preliminary analysis of the best components must be performed to get good results and robust task execution.

This work's main contribution is to adapt and analyze two different localization techniques for inspecting relevant objects in unstructured environments, which can help address most typical problems present in inspection scenarios. The first technique, based on AMCL, focuses on 2D localization, while the second technique, based on ICP and UKF, focuses on 3D localization. For each approach, a map is created in advance by different means. This article also studies different initialization options and convergence parameters. Conclusions from our experiences and results are drawn and highlighted in the article.

This paper is structured as follows: Section 2 collects and summarizes information from similar articles; in Section 3 procedures and algorithms are introduced. Then, the experimental setup, hardware, and results follow in Section 4, while the discussions are in Section 5. To conclude, Section 6 summarizes the main findings.

2. Related Work

Simultaneous Localization and Mapping (SLAM) [14] deals with the challenge of creating a map of an unknown environment and performing localization on that map at the same time. Robust SLAM methods are now available for mapping static, structured environments of limited size. However, performing SLAM in dynamic, large, and unstructured environments is still an open problem [15], so new methods are developed every year in order to gain more efficiency. Avola et al. [16] proposed a SLAM algorithm that provides the traveled route as well as the flight height by using both a calibration step and visual features extracted from the acquired images. The authors of [17], for instance, use two SLAM approaches, based on a 2D laser sensor and a monocular camera, to obtain logarithmic efficiency, compensating for the errors coming from them. Besides high computational cost, another important problem when applying online SLAM during task execution is robustness: it is less robust than localization on a previously validated map. Another approach [18] proposes a stereo SLAM method for UAVs to improve the robustness in rotating and motion scenes using two strategies, based on image quality and motion state.

Focusing on localization on a given map, several sensors or techniques may be combined with increased performance in some applications, but all of them have advantages and disadvantages. One of the most widely used pose estimation algorithms is the Extended Kalman Filter (EKF) [19]. It uses previous state information, control inputs, and sensor measurements. In [20], the authors present a work that uses a Global Position System (GPS) together with an Inertial Measurement Unit (IMU). Others, such as Mourikis et al. [21], combine visual characteristics from an omnidirectional camera and IMU data. The main limitation of this kind of method is the dependence on the initial position.

When the observation and state models are highly nonlinear, the UKF [22] is preferred since this filter calculates the pose using nonlinear models, resulting in a more accurate estimation than the EKF. In this regard, Ref. [23] made a comparison between UKF and EKF applied to low-cost sensors and very noisy measurements. Among the localization techniques that use nonlinear solutions for 3D environments, others are based on scan matching methods such as the Iterative Closest Point algorithm (ICP) and its variations [24–26], and others employ the Normal Transformation Distribution (NDT) [27–29], and some works, such as [30], use manifold Gaussian processes (mGPs) with geodesic kernel functions for mapping surface information to get potential spatial correlations and plan informative paths online in a receding horizon manner.

The Markov method [31] may be used when the initial pose is unknown. It works well with very noisy data, reducing ambiguities. This technique has some variants, including grid localization [32], where the map is divided into a grid, and each cell represents the occupancy probability. Another efficient and simple technique is Monte Carlo localization (MCL) [33]. This technique can solve global localization problems. Its principle is based on a particle filter distributed over the entire map with an estimated probability value. When the algorithm converges, the particle cloud's size decreases. A variant called Adaptive Monte Carlo Localization (AMCL) [34] changes the number of samples over time instead of using a fixed-size particle cloud. This technique consumes less computational resources, is efficient, accurate, and able to solve the kidnapped robot problem. In this context, Xu et al. [35] proposed a hybrid localization method based on WiFi fingerprints and AMCL, Javierre et al. [36] combined a laser sensor and an omnidirectional camera, and [37] used a 2D/3D laser sensor with an ellipsoidal energy model.

For robot-assisted visual aircraft inspection, multiple approaches can be found in the literature, such as mobile platforms [38] or fixed cameras [39]. These methods simplify or avoid the localization problem, but their installation and operation could be more difficult than other solutions. Teleoperated climbing robots are able to perform autonomous inspections [40], but their navigation requires inputs from an operator. Other solutions present autonomous wheeled robots [41] and autonomous UAVs [42,43].

3. Methodology

In this work, we have chosen aircraft inspection as a test example because it takes place in an unstructured environment and aircraft positions may be different in each parking lot. Furthermore, the airplane geometry has several characteristics to be recognized, and the inspector UAV may take off from different positions. For this work, an open 3D CAD airplane model was used. We focus on aircraft inspection, but the selected methods and procedures would be similar in other inspection tasks, and other UAV models would also serve the same purpose. In addition, we have performed an inspection on a truck to test our methodology and compare the data acquired.

3.1. Maps

A map of the environment and an octomap (Figure 1a) [44] were built from the CAD model so as to test the performance of the algorithms. The octomap constitutes a probabilistic representation used to model unknown areas. It is efficient because it uses a tree structure called octree. Each node in an octree represents the space in a cubic volume, defined as a voxel. This volume is recursively subdivided into eight sub-volumes until a given minimum voxel size is reached.

An occupancy grid map [45] was built as a 2D map. It represents the environment as a block of cells, either occupied or unoccupied with some probability. To build this map, the octomap was projected into a horizontal plane (Figure 1b). The algorithm iterates over all octree leaves and marks the 2D map cells accordingly.

Another way to obtain a 3D map comes by applying SLAM. We selected a graph SLAM [46] algorithm, which uses a graph whose nodes correspond to the robot's poses at different time points and whose edges represent constraints between the poses (Figure 1c). Once such a graph is constructed, the map can be computed by finding the nodes' spatial configuration that is most consistent with the measurements modeled by the edges.

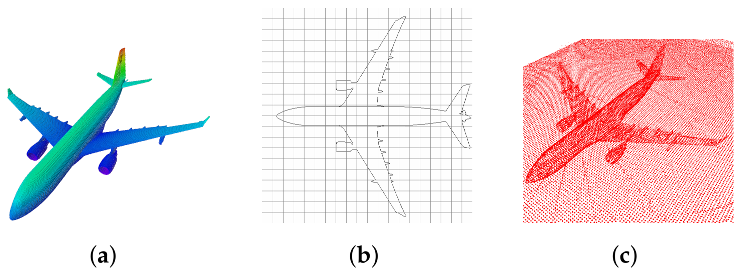


Figure 1. Kind of maps to be used. (**a**) Three-dimensional Occupancy grid Map (Octomap). (**b**) Two-dimensional Occupancy grid Map. (**c**) Graph SLAM map.

3.2. 2D Localization

Localization in 2D is performed by means of the AMCL [47] algorithm. A particle filter integrates a robot motion model, an observation model, and an adaptive algorithm that adds random particles based on its convergence; Figure 2 and Video S1 show an example of the algorithm performance. The following relation can summarize it:

$$X = f(X_{t-1}, U_t, Z_t, m) \tag{1}$$

where:

- X , current set of particles.
- X_{t-1}, the previous set of particles.
- U_t, last motion and odometry measurements.
- Z_t, last laser rangefinder measurements.
- m, max number of particles.

In addition, there are several aspects to consider, such as:

- Motion kinematics: e.g., differential or omnidirectional.
- Uncertainty of the robot odometry, which determines the error in translation α_1, α_2 or rotation $\alpha_3, \alpha_4, \alpha_5$.
- Observation model: e.g., beam model or probability field model.
- Measurements errors: e.g., measurement noise z_{hit}, unexpected objects z_{short}, object detection failures z_{max}, unexplained random noise z_{rand}.
- Number of random particles, defined by the probabilities of long-term α_{slow} and short-term α_{fast} measurements.

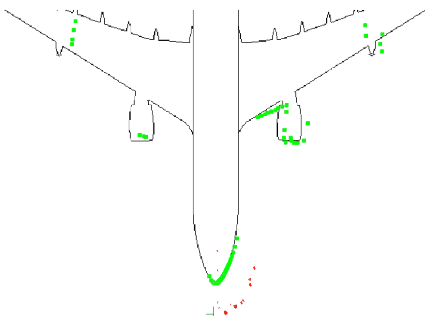


Figure 2. Example of 2D localization using AMCL algorithm (red), 2D laser Scan (green) and an Occupancy grid map (black).

Nowadays, there is no general method other than calibration or trial and error to set the parameters of motion model noise, measurement model noise, and particle decay rate. Assigning values without any previous experience is very complicated.

The motion model selected in this work corresponds to Omnidirectional dynamics , typical of UAV's motion (DJI Matrice 100 and hummingbird UAV). The DJI Matrice 100 has very accurate odometry samples, and the error is minimal, so the values related to the noise of the motion model are small. The error values in simulated odometry sensors mounted on the Hummingbird UAV were based on the default values.

The observation model is the Probability field measurement model because of its lower computational cost and its smooth error distributions. It is also known that the Velodyne laser sensor does not present many errors in terms of reflection, maximum distance, and loss of object detection. The main source of error is the measurement noise itself, so z_{hit} should have a large value, z_{rand} a small value, and z_{short}; z_{max} much smaller. The AMCL simulation test parameters are given in Appendix C, and those of AMCL real flights are presented in Appendix D.

AMCL estimates only the position of x and y, and the z-coordinate is always set to zero because the algorithm assumes that all objects are on the horizontal surface. Therefore, flights must be performed at some horizontal and vertical distance from the aircraft to guarantee that the 3D laser detects complete point clouds which present a correct projection. The horizontal distance between the UAV and the aircraft is between 2 and 3 m, depending on the inspection trajectory (Figure 3). Moreover, some flights at various altitudes were performed to find the best altitude.

Several flights were executed. The take-off position and the initial uncertainty were random because we wanted to test how the system would work with arbitrary large errors in the initial pose estimation. Trajectories with the smallest error were selected, and covariance values given by AMCL cov_x, cov_xy, cov_y were recorded, as well as the number of occupied cells observed by the laser. These values give an indication of convergence and error. These results were used in real flights to determine a convergence position in the future; it could be used as an initial position to execute another inspection task.

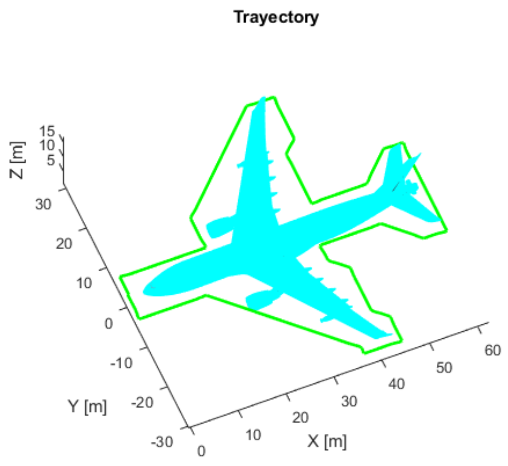


Figure 3. Proposed inspection trajectory (green). Three-dimensional CAD model (cyan).

3.3. *3D Localization*

Three-dimensional localization is performed by adapting the algorithm presented in [29] for NDT and IMU data with UKF. The algorithm used in this work replaces the proposed scan-matching techniques: NDT, ICP, and Generalized ICP by nonlinear ICP [48]. Figure 4 and Video S4 show an example of the algorithm's performance.

The algorithm is based on three stages:

- Subsampling: Reduction of the number of points in the sample using Voxel grid filter and Pass-through Filter techniques [49] to optimize processing time.
- Pose estimation with nonlinear ICP: Based on the ICP technique presented in [50]. It takes as inputs a source and a target point cloud matched under the nearest-neighbor criterion. Singular Value Decomposition (SVD) is applied to obtain an estimate of the transformation matrix that aligns them. This process is repeated until a termination criterion is met, removing outliers and redefining the correspondences.

- Unscented Kalman Filter-based localization: This is an improvement of EKF for application to highly nonlinear systems. This approach uses the unscented transform to take a set of samples called sigma points, which are propagated by nonlinear functions and used to calculate the mean and variance. Unlike EKF, UKF eliminates the need for a Jacobian, facilitating calculations on complex functions.

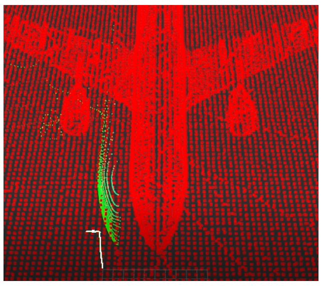


Figure 4. Example of 3D localization using nonlinear ICP algorithm (white), 3D laser Scan (green) and a 3D Occupancy grid map (red).

To obtain the 3D pose, the algorithm takes as inputs: the 3D point cloud map of the environment, the previous state including position and orientation, LIDAR sensor data, and IMU data together with linear and angular velocities. The result is the 3D pose. First, the map and LIDAR sensor data are subsampled. Next, pose estimation is performed based on the IMU linear and angular accelerations within a defined time frame. These data are used in the filter prediction stage; then, the scan matching algorithm performs the pose estimation by aligning the map point cloud and the sensor point cloud. This estimation will be used for the filter correction stage.

The pseudocode for this solution is presented in Algorithm 1 :

Algorithm 1: Localization 3D (X_{t-1}, V_t, Δ_{qt}, $3D\_sensor\_points$, $3D\_map$)

```
1 simple_map = subsampling (3D_map);
2 while True do
3     $simple\_3D\_points = subsampling(3D\_sensor\_points);$
4     $X_t = (p_{t-1} + v_{t-1} * \Delta_t, q_{t-1} * \Delta qt; v_{t-1})$ UKF prediction;
5     $(p_t^{obs}, q_t^{obs}) = ICP\_NL(simple\_map; simple\_3D\_points)$ ;
6     $(p_t, q_t) = (p_t^{obs}, q_t^{obs})$ UKF correction
  Result: $X_t$
```

To test the performance of the ICP-NL and NDT algorithms in simulation, the trajectory of Figure 3 was executed at different heights, and the ATE and convergence time was calculated. In addition, the coincidence between the 3D laser sensor point cloud and the generated 3D map is qualitatively verified in simulation and real data, and the deviations between the estimated 3D position and the real position are also observed.

4. Results

Initially, we specify the software and hardware used, as well as the metric to determine the trajectory estimation errors. Then the 2D localization results in simulation and real flights, and finally, the 3D localization results.

4.1. Software

This work was developed by using the popular Robot Operating System (ROS) [51] and Gazebo [52] simulation tools, which help create simulations of robotic applications in three-dimensional indoor and outdoor environments. The Aerostack framework [53], which allows developers to design and test control architectures for aerial robotic systems, was also extensively used. Rotors [54] provided the simulation of UAVs and localization sensors. Figure 5 shows the complete software system used to perform the localization task in simulations and in real flights.

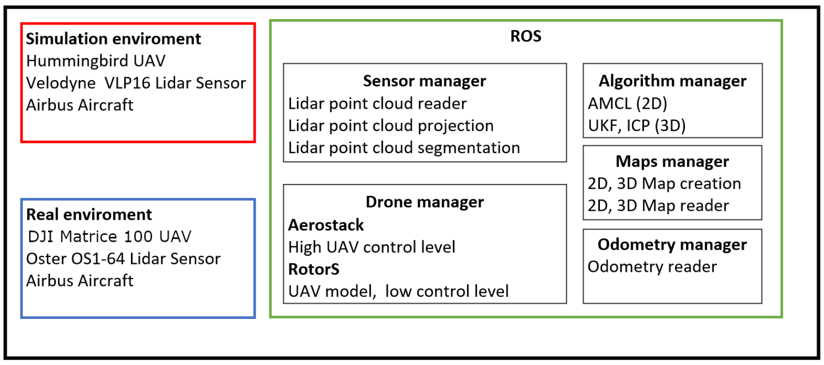


Figure 5. Proposed system architecture for simulations and real flights.

4.2. Hardware

About the actual UAV assembly, the hardware components used in simulation flights are as follows:

- Personal Computer (PC) Intel core i7 2.70 GHz, 4 Gb RAM.
- Hummingbird UAV provided by Rotors (Figure 6a).
- Odometry measurements from a configurable odometry sensor (see Appendix A for configuration details).
- Velodyne VLP16 3D laser sensor [55] provides a point cloud with 300,000 points per second and ±3 cm accuracy, 100 m range, 360° horizontal and 30° vertical field of view (see Appendix B for setting other parameters). Due to computational limitations in the simulation, we worked with 5120 points per second.

In real flights, we used:

- Manifold Intel i7 1.8 Ghz, 2 GB RAM.
- DJI Matrice 100 UAV (Figure 6c).
- Odometry measurements were provided by a sensor fusion algorithm [56] that merges the onboard DJI sensors: Altimeter, Velocity and IMU, plus the 3D Light Detection and Ranging (LIDAR) IMU.
- 3D LIDAR Ouster OS1-64, providing a point cloud with 327,680 points per second and ±1 cm accuracy 100 m range, 0.3 cm resolution, 360° horizontal and 45° vertical field of view.

In both setups, we used the Airbus A330 3D CAD model [57] to build the prior maps. The teleoperation flights and trajectory tracking were controlled by Aerostack.

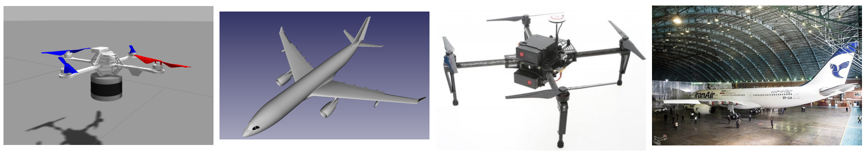

(**a**) (**b**) (**c**) (**d**)

Figure 6. Components used in the simulation (**a**,**b**), and real flights (**c**,**d**). (**a**) Velodyne sensor and Hummingbird UAV. (**b**) Aircraft Airbus A330 model. (**c**) Three-dimensional LIDAR Ouster and DJI Matrice 100 UAV. (**d**) Airbus A330 [58].

4.3. Error Metric

The Absolute Trajectory Error (ATE) is a measure to check that the solution to a benchmark problem is valid. ATE is applicable in algorithms for robot trajectory estimation. It measures the difference between the ground-truth position and the robot's estimated position at the same instant in time. Further, it should be noted that the robot pose's orientation component is not considered [59,60]. In this work, ATE is only computed in simulation environments due to a lack of ground-truth data in real environments.

$$ATE = \sqrt{\frac{1}{n}\sum_{i=1}^{n} T_i^2} \tag{2}$$

where:

- T_i, is the magnitude of the Euclidean distance along the horizontal plane between the estimated and ground-truth poses at frame i.
- n, number of frames.

4.4. 2D Localization

To build the 2D occupancy grid map, a 3D CAD model in Digital Asset Exchange (DAE) format for Gazebo was imported (Figure 7a). An octomap of the environment was created with a 0.1 m voxel resolution (Figure 7b), and then it was projected on a horizontal flat plane to make a 2D occupancy map with a 0.1 m cell resolution (Figure 7c).

Subsequently, this map was modified by an external script to obtain the image's edges. This could improve the algorithm's performance because the black pixels represent an occupied area on the map. When comparing the map with the laser measurements, the algorithm could incorrectly interpret the internal occupied cells and give a wrong estimation of the position. The new 2D map is shown in Figure 7d.

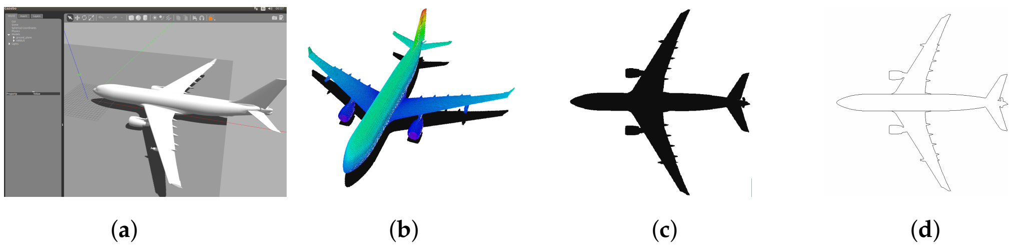
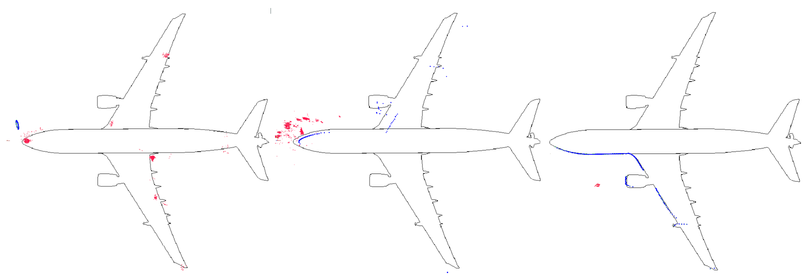


Figure 7. 2D occupancy grip map creation process. (**a**) Airbus A330 Gazebo Model. (**b**) Octomap generated with 0.1 cm voxel resolution. (**c**) Octomap horizontal projection. (**d**) Modified 2D occupancy grip map.

4.4.1. Simulation Tests

Regarding localization performance, the AMCL parameters were calibrated in the simulation to verify the result's quality. When the UAV takes off, the particle distribution is very sparse, as shown in Figure 8a. As the UAV advances, the particle distribution gets more compact, as shown in Figure 8b. When the UAV keeps flying longer, the algorithm converges, and the 2D laser scan matches the map, as shown in Figure 8c. For a complete visualization, please refer to Video S1.

To find the best flight altitude in combination with the inspection trajectory (Figure 3) and 3D laser scan configuration (range [0.1, 100] meters; horizontal and vertical field of view respectively, [−360, 360] degrees, [−15, 15] degrees) 20 flights were executed in simulation at each altitude. Table 1 shows the result, where 5 m corresponds to the altitude with the lowest ATE. Therefore, 5 m is the altitude executed in all simulated flights to evaluate 2D performance.



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

Figure 8. AMCL convergence process. Two-dimensional laser scan (blue), AMCL particle cloud (red). (**a**) UAV takes off, particles are distributed all over the map. (**b**) UAV moves forward, particle scattering begins to decrease The 2D laser scan does not match the map. (**c**) AMCL converges, particle scattering is small, and the laser data matches the map.

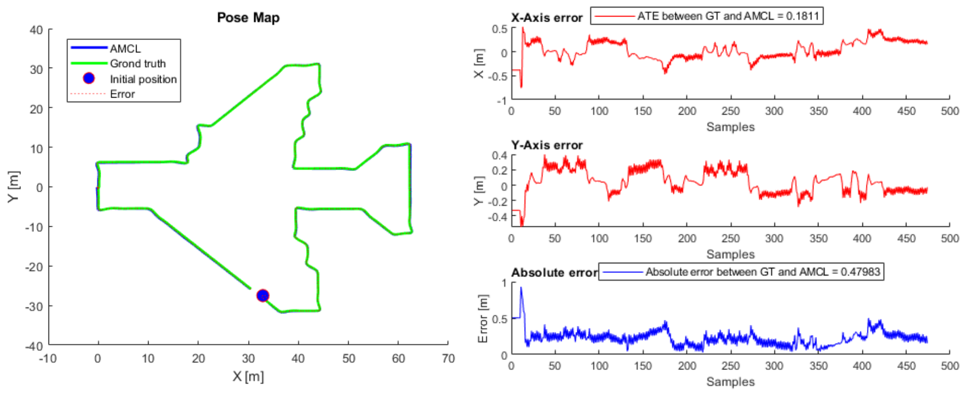
Table 1. Calculation of ATE at different altitudes for 20 simulated flights.

Altitude (m)	**3**	**4**	**5**	**6**	**7**	**8**
Mean ATE (m)	0.3460	0.3287	0.3127	0.3376	0.4077	2.1437

To evaluate the performance, 81 different flights were tested. The proposed initial error was three meters maximum for each axis reference (x,y). Each flight starts at a random initial position taken from the inspection trajectory (Figure 3) and presents an initial error not always smaller than three meters, so in some cases, the algorithm may not converge. Table 2 presents the average, minimum, and maximum ATE errors. Figure 9 presents the trajectory with the lowest error, and Figure 10 the trajectory with the highest error; in this case, the algorithm does not converge because the initial covariance is smaller than the error in the initial position.

Table 2. Calculation of minimum, maximum, and average ATE for 81 simulated flights.

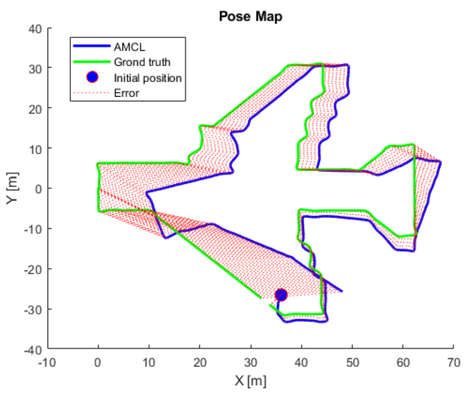
Tests	**ATE Min (m)**	**Ate Max (m)**	**Mean ATE (m)**
81	0.18	5.63	0.34



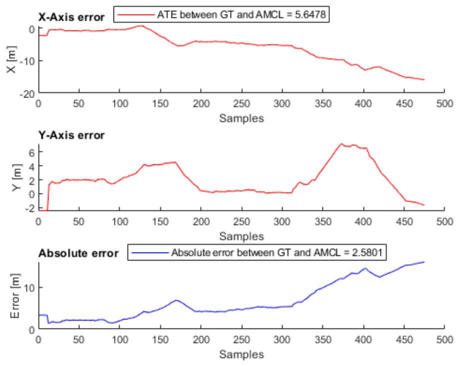
(**a**) (**b**)

Figure 9. (**a**) AMCL simulation test with the lowest ATE. AMCL (blue), ground-truth (green), take-off position (blue circle), error (red). (**b**) AMCL errors.

Table 3 shows the initial position and the ground truth (GT) position, with the maximum and minimum ATE. When the difference between the GT and AMCL pose is large, the algorithm starts from erroneous estimates. If the initial covariance is large enough regarding the error in the initial position, the algorithm should converge.



(**a**)



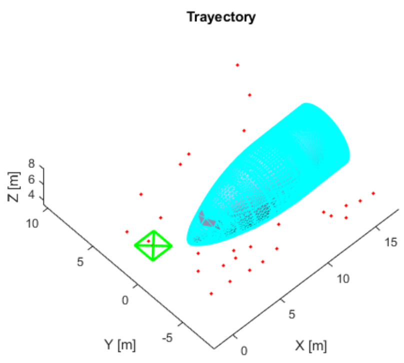
(**b**)

Figure 10. (**a**) AMCL simulation test with the highest ATE. AMCL (blue), ground-truth (green), take-off position (blue circle), error (red). (**b**) AMCL errors.

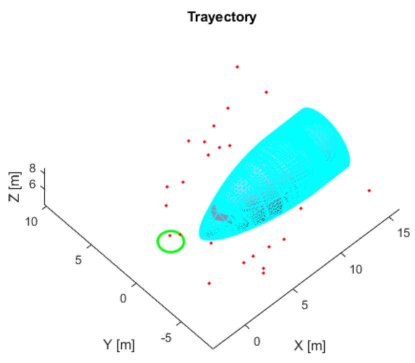
Table 3. Initial positions for lowest and highest ATE trajectories.

ATE	GT Initial Pos (x,y) (m)	AMCL Initial Pos (x,y) (m)	Initial Covariance (xx,yy) (m)	Initial Error (x,y) (m)
Min ATE 0.18	32.53, −27.83	32.91, −27.5	0.5, 0.5	−0.38, −0.33
Max ATE 5.63	33.72, −29	36.01, −26.56	0.5, 0.5	−2.29, −2.44

The execution of a short flight before starting the inspection can help improve localization in these kindd of tasks, providing the resulting pose as the new initial position. It reduces the uncertainty in the initial position and allows for correct position estimation and a safe flight. Two types of trajectories have been defined in this work, a circular trajectory (Figure 11b) and a squared trajectory (Figure 11a). A total of 91 flights were performed with each trajectory and different initial positions (red point in Figure 11) along the inspection trajectory (Figure 3). The analysis of the results indicates that the squared trajectory worked best, with a mean ATE of 2.429 (m).



(**a**)



(**b**)

Figure 11. Proposed initial flight (green), Aircraft (cyan), take-off point (red). (**a**) Squared path. (**b**) Circular path.

To further analyze the results, the particle covariance matrix in the x and y directions was obtained. We also logged the number of matches between occupied cells and 2D laser scan points for those tests with an initial error lower than 0.3 m. Table 4 shows these values.

Table 4. Relationship between the ATE, covariance in x and y directions, and matches of occupied grids cells and 2D laser scan points.

Error between AMCL and Ground Truth (m)	Mean Covariance(x,y)	Mean Occupancy Grip Map	Mean 2D Laser Scan	Mean Relation Matches/Laser-Scan
<=0.3	0.0557	41,820	245,917.5	0.17005

We realized other simulations to compare the results. We created a 2D occupancy grid map from a 3D CAD truck model using the same procedure explained in Section 4.4. Figure 12 shows the different steps of the map-building process.

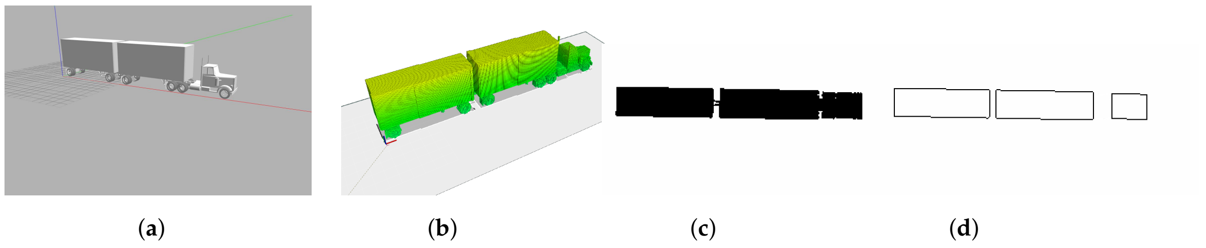


Figure 12. Two-dimensional occupancy grip map creation process. (**a**) Truck Gazebo Model. (**b**) Octomap generated with 0.1 cm voxel resolution. (**c**) Octomap horizontal projection. (**d**) Modified 2D occupancy grip map.

Thirty-six different flights were performed using the same aircraft simulation parameters, each flight starting at a different point around the trajectory proposed in Figure 13. Figure 14 shows the flight with the lowest error, and the first row in Table 5 shows the initial parameters. In this case, there is a considerable AMCL pose estimation error at the beginning, but when the UAV continues flying, the algorithm converges until reaching some complicated truck place. Video S7 shows the full flight. Figure 15 shows the flight with the highest ATE, and the second row of Table 5 shows the initial parameters. In this flight, the initial error is the lowest. Even though AMCL does not converge along some parts of the trajectory, most of the time, the algorithm estimates the correct pose.

The tests performed allowed data indicating when the algorithm converges to be obtained. We take all the points of the flights with an error of less than 30 cm and calculate the mean of the particle covariance matrix in x and y, the number of matches between occupied cells, and the 2D laser scan point. Data can be seen in Table 6.

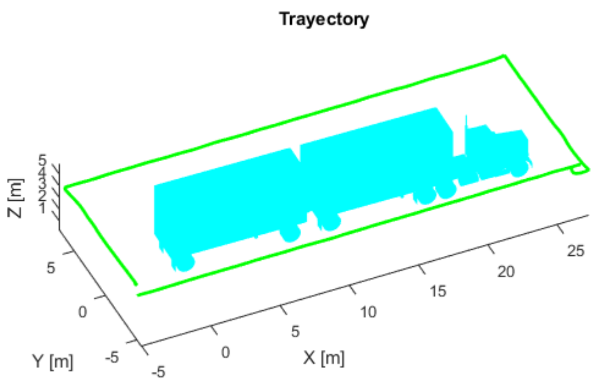
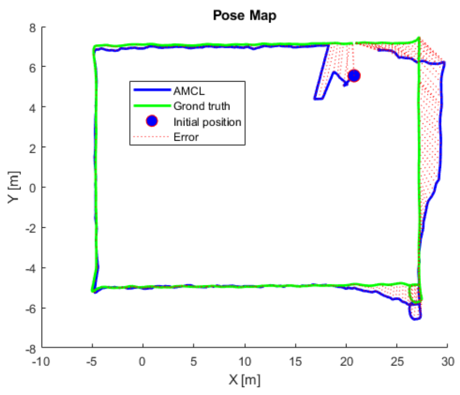
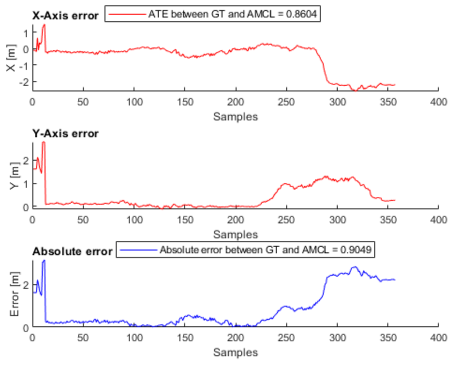


Figure 13. Proposed inspection trajectory (green). Truck 3D CAD model (cyan).

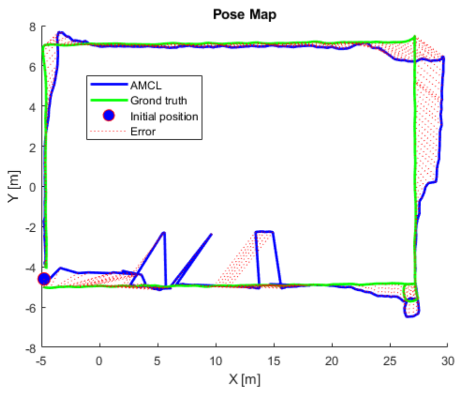


(**a**)

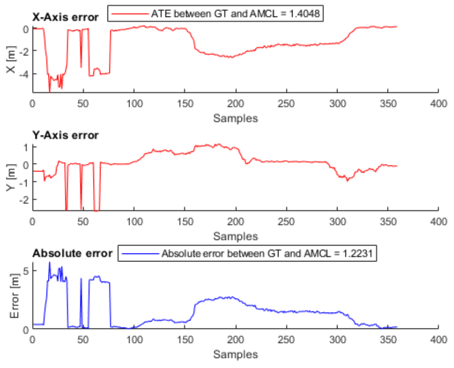


(**b**)

Figure 14. (**a**) AMCL simulation test with the lowest ATE. AMCL (blue), ground truth (green), take-off position (blue circle), error (red). (**b**) AMCL errors.



(**a**)



(**b**)

Figure 15. (**a**) AMCL simulation test with the highest ATE. AMCL (blue), ground truth (green), take-off position (blue circle), error (red). (**b**) AMCL errors.

Table 5. Initial positions for lowest and highest ATE trajectories in truck inspections.

ATE	GT Initial Pos (x,y) (m)	AMCL Initial Pos (x,y) (m)	Initial Covariance (xx,yy) (m)	Initial Error (x,y) (m)
Min ATE 0.86	20.66, 7.16	20.77 , 5.56	0.5, 0.5	−0.11, −1.6
Max ATE 1.4	−4.82, −5	−4.77, −4.6	0.5, 0.5	−0.05, −0.4

Table 6. Relationship between the ATE, covariance in x and y, and matches of occupied grid cells and 2D laser scan points in the truck inspection.

Error between AMCL and Ground Truth (m)	Mean Covariance (x,y)	Mean Occupancy Grip Map	Mean 2D Laser Scan	Mean Relation Matches/Laser-Scan
<=0.3	0.0115	18,037	76,847	0.2347

4.4.2. Real Tests

To collect data in a real environment, the UAV pilot performed several flights where the object to inspect had relevant characteristics; in each flight, we recorded information from the UAV such as altitude, velocity, IMU, and the point cloud provided by a 3D laser in rosbag files. Then we played rosbag files with the same timestamp in all topics to get the saved information.

Three real, short flights were chosen before starting the inspection to check the algorithm's convergence. The proposed trajectories are shown in Figure 16. In real tests, stairs are present in 3D laser scan data Figure 17a. Therefore, stairs were added to the initial 2D occupancy grid map to better represent the environment in Figure 17b.

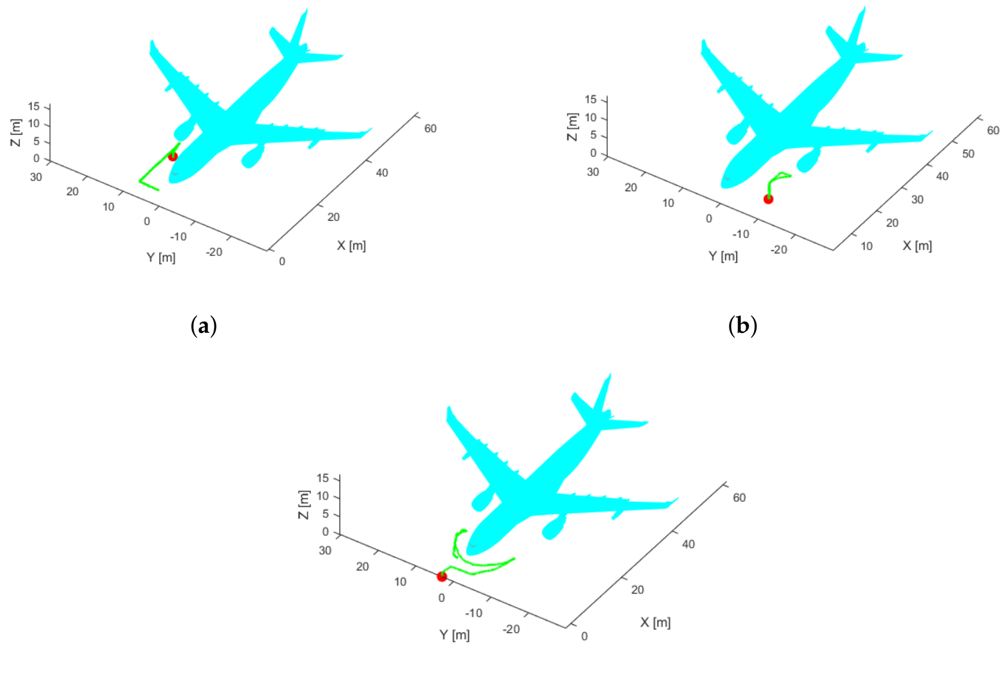


Figure 16. Proposed trajectories for real flights. Take-off and landing point (red circle). (**a**) Path 1. (**b**) Path 2. (**c**) Path 3.

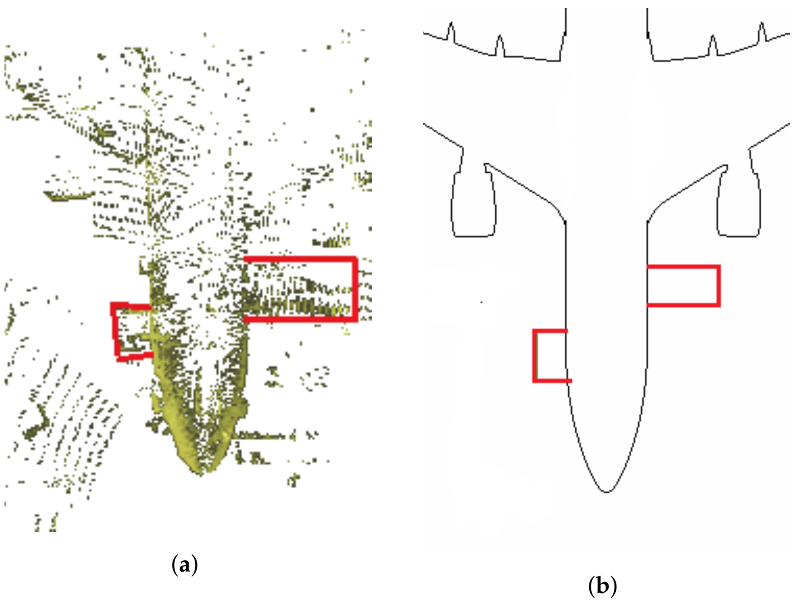
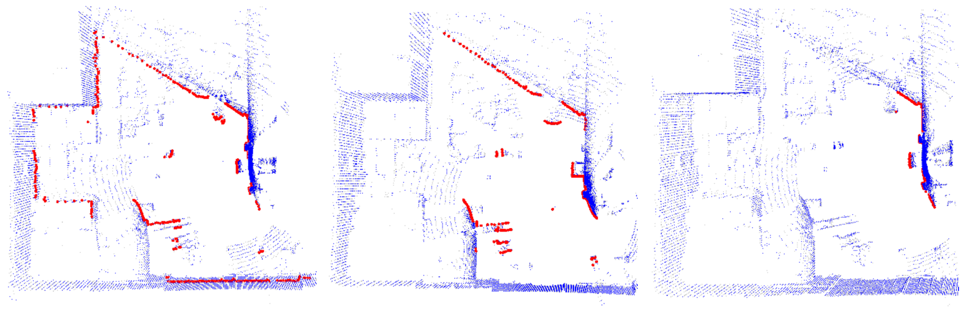


Figure 17. Map update, stairs (red). (**a**) 3D Laser scan real data. (**b**) New occupancy grip map.

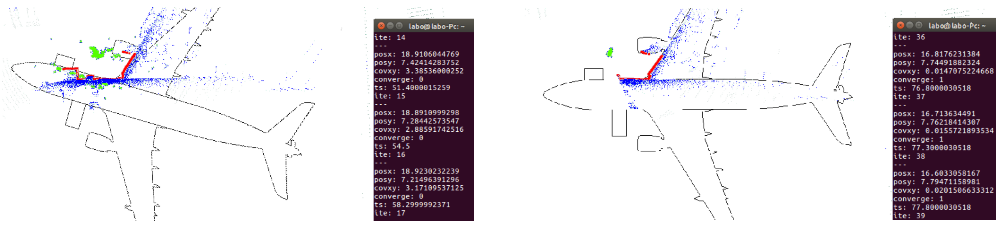
In the real flights, walls were not considered, and this had a negative effect on the algorithm performance. To solve this problem, the first option was the method proposed in [61], based on plane segmentation by means of RANSAC [62] and the implementation available in the Point Cloud Library (PCL) [63]. Figure 18a shows the 3D Point Cloud and the original conversion to a 2D laser scan; Figure 18b shows the wall segmentation and the projection to a 2D laser scan. The second option is to limit the 2D laser scan data in range and angle of view, as shown in Figure 18c.



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

Figure 18. Wall removal. Three-dimensional LIDAR point Cloud (blue), 2D laser scan (red). (**a**) Original 2D laser scan data and 3D LIDAR point cloud. (**b**) Wall segmentation by RANSAC PCL. (**c**) Wall segmentation by limiting 2D laser scan data. Range [0.1, 15] meters, angular field of view [−135, 135] degrees.

The first proposed trajectory was tested with a laser range limitation of 15 m and an angular field of view restricted to [−135, 135] degrees. Figure 19 shows that the particles are very spread at the beginning of the trajectory. When the UAV moves, the scattering decreases, the laser matches the map, and the covariance in x and y decreases below the desired values. In this case, 37 iterations were required to converge. Figure 20 shows the results using wall segmentation for RANSAC PCL, which required 250 iterations to converge, and Video S2 shows the complete flight.

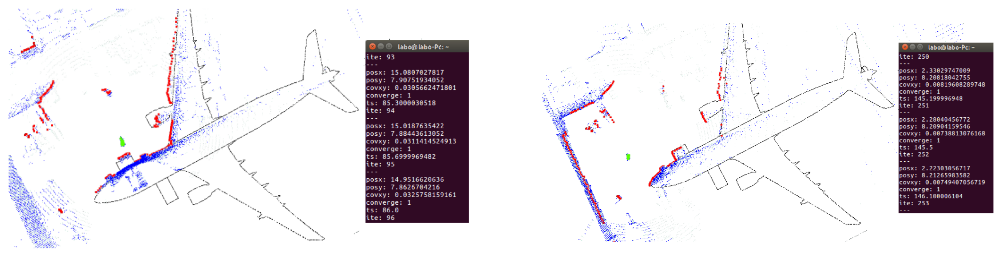


```
ite: 14
---
posx: 18.9106044769
posy: 7.42414283752
covxy: 3.38536000252
converge: 0
ts: 51.4000015259
ite: 15
---
posx: 18.8910999298
posy: 7.28442573547
covxy: 2.88591742516
converge: 0
ts: 54.5
ite: 16
---
posx: 18.9230232239
posy: 7.21496391296
covxy: 3.17109537125
converge: 0
ts: 58.2999992371
ite: 17
```

```
ite: 36
---
posx: 16.8176231384
posy: 7.74491882324
covxy: 0.0147075224668
converge: 1
ts: 76.8000030518
ite: 37
---
posx: 16.713634491
posy: 7.76218414307
covxy: 0.0155721893534
converge: 1
ts: 77.3000030518
ite: 38
---
posx: 16.6033058167
posy: 7.79471158981
covxy: 0.0201506633312
converge: 1
ts: 77.8000030518
ite: 39
```

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

Figure 19. Convergence tests on Path 1 (Figure 16a) real flight with limited laser ranges. Three-dimensional LIDAR point cloud (blue), 2D laser scans (red), AMCL particles (green). (**a**) UAV position before algorithm convergence. (**b**) UAV position when the algorithm converged at 37 iterations.



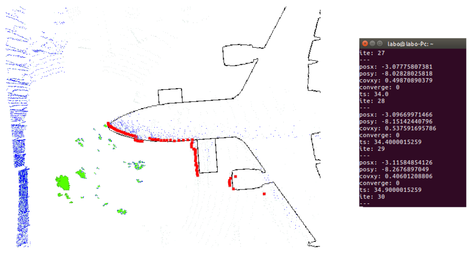
```
ite: 93
---
posx: 15.0807027817
posy: 7.90751934052
covxy: 0.0305662471801
converge: 1
ts: 85.3000030518
ite: 94
---
posx: 15.0187635422
posy: 7.88443613052
covxy: 0.0311414524913
converge: 1
ts: 85.6999969482
ite: 95
---
posx: 14.9516620636
posy: 7.8626704216
covxy: 0.0325758159161
converge: 1
ts: 86.0
ite: 96
```

```
ite: 250
---
posx: 2.33029747009
posy: 8.20818042755
covxy: 0.00819608289748
converge: 1
ts: 145.199996948
ite: 251
---
posx: 2.28040456772
posy: 8.20904159546
covxy: 0.00738813076168
converge: 1
ts: 145.5
ite: 252
---
posx: 2.22303056717
posy: 8.21265983582
covxy: 0.00749407056719
converge: 1
ts: 146.100006104
ite: 253
---
```

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

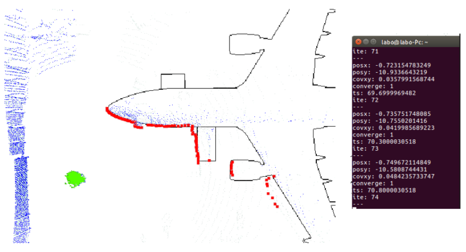
Figure 20. Convergence tests on Path 1 (Figure 16a) real flight with RANSAC PCL segmentation. Three-dimensional LIDAR point cloud (blue), 2D laser scans (red), AMCL particles (green). (**a**) UAV position before algorithm convergence. (**b**) UAV position when the algorithm converges after 250 iterations.

Wall segmentation with RANSAC PCL tests has been performed following Path 2 (Figure 16b) and Path 3 (Figure 16c) because the trajectory is close to the nose of the aircraft, and the laser does not capture enough information of the whole environment. Figure 21 and Video S3 show the Path 2 result, which converged in 71 iterations. As for Path 3, the results are shown in Figure 22. In this case, the algorithm was not able to converge, so the trajectory is not good enough to achieve an acceptable initial position.



```
labo@labo-Pc: ~
ite: 27
---
posx: -3.07775807381
posy: -8.02828025818
covxy: 0.49870890379
converge: 0
ts: 34.0
ite: 28
---
posx: -3.09669971466
posy: -8.15142440796
covxy: 0.537591695786
converge: 0
ts: 34.4000015259
ite: 29
---
posx: -3.11584854126
posy: -8.2676897049
covxy: 0.40601208806
converge: 0
ts: 34.9000015259
ite: 30
---
```

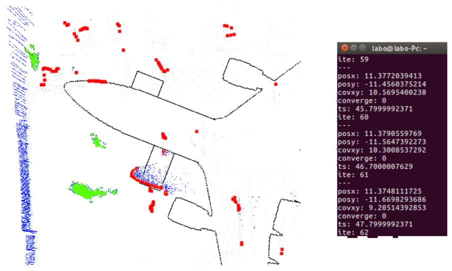
(**a**)



```
labo@labo-Pc: ~
ite: 71
---
posx: -0.723154783249
posy: -10.9336643219
covxy: 0.0357991568744
converge: 1
ts: 69.6999969482
ite: 72
---
posx: -0.735751748085
posy: -10.7550201416
covxy: 0.0419985689223
converge: 1
ts: 70.3000030518
ite: 73
---
posx: -0.749672114849
posy: -10.5808744431
covxy: 0.0484235733747
converge: 1
ts: 70.8000030518
ite: 74
---
```

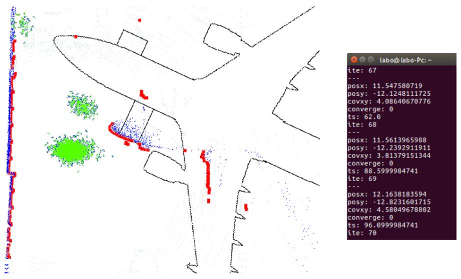
(**b**)

Figure 21. Convergence tests on Path 2 (Figure 16b) real flight with RANSAC PCL segmentation. Three-dimensional LIDAR point cloud (blue), 2D laser scans (red), AMCL particles (green). (**a**) UAV position before algorithm convergence. (**b**) UAV position when the algorithm converges after 71 iterations.



```
labo@labo-Pc: ~
ite: 59
---
posx: 11.3772039413
posy: -11.4560375214
covxy: 10.5695400238
converge: 0
ts: 45.7999992371
ite: 60
---
posx: 11.3790559769
posy: -11.5647392273
covxy: 10.3008537292
converge: 0
ts: 46.7000007629
ite: 61
---
posx: 11.3748111725
posy: -11.6698293686
covxy: 9.20514392853
converge: 0
ts: 47.7999992371
ite: 62
```

(**a**)



```
labo@labo-Pc: ~
ite: 67
---
posx: 11.547580719
posy: -12.1248111725
covxy: 4.08640670776
converge: 0
ts: 62.0
ite: 68
---
posx: 11.5613965988
posy: -12.2392911911
covxy: 3.81379151344
converge: 0
ts: 88.5999984741
ite: 69
---
posx: 12.1638183594
posy: -12.8231601715
covxy: 4.58049678802
converge: 0
ts: 96.0999984741
ite: 70
```

(**b**)

Figure 22. Convergence tests on Path 3 (Figure 16c) real flight with RANSAC PCL segmentation. Three-dimensional LIDAR point cloud (blue), 2D laser scans (red), AMCL particles (green). (**a**) UAV position before algorithm convergence. (**b**) The algorithm does not converge during the flight.

4.5. 3D Localization

4.5.1. Simulation Tests

The 3D map was built based on the graph SLAM algorithm presented in [27,64]. This algorithm uses odometry obtained from NDT-scan matching and loop closure (detection of an area that has already been mapped) with corrections based on GPS data. In this work, this method was simplified by replacing the odometry based on scan-matching with the robot's real position obtained from the simulation environment, obtaining higher accuracy in the map. Figure 23 below shows the 3D map obtained.

The NDT and ICP-NL algorithms have been compared to observe their performance and differences. For this purpose, the ATE has been calculated for the flights performed. The results are presented in Table 7. Figure 24 shows the main results, three stages of operation: starting position, take-off, and navigation, were performed. The trajectories painted in white represent the algorithm estimation, and the trajectories painted in green represent the real movement.

The estimated initial position for the two cases differs by 5 m in x and y from the real one. When using NDT, the algorithm tries to estimate a trajectory, but it fails to converge to the actual pose (Figure 24b). In ICP-NL, the algorithm converges during take-off because the sensor observes more points, and during navigation, the error remains low (Figure 24a). For a complete visualization, please refer to Video S4.

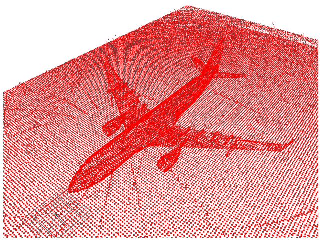


Figure 23. Aircraft 3D map generated by graph SLAM.

Table 7. Errors obtained in NDT and ICP-NL simulation tests.

Algorithm	**ATE**
ICP-NL	0.68
NDT	289.93

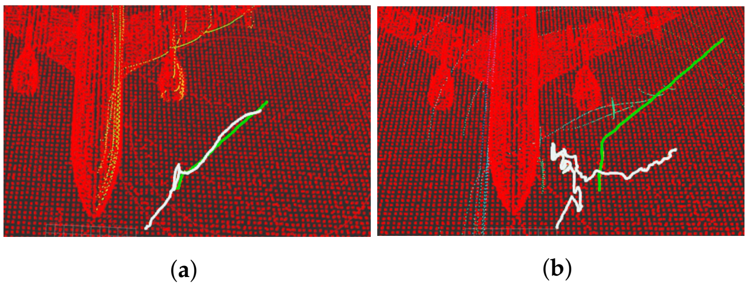


Figure 24. Position estimate by the algorithms Non-linear ICP (**a**) and NDT (**b**). Three-dimensional graph SLAM map (red), position estimate by the algorithm (white), ground truth (green).

Additionally, several flights were performed by modifying the UAV flying height between 3.5 and 8.5 m in order to study the influence on the results. We observed that ICP-NL maintains its performance regardless of the altitude. On the other hand, NDT works well in the last test altitudes because the higher the altitude, the better the aircraft is observed, and hence the scan-matching algorithm performs better. Considering the initial error established, the established convergence time is the time it takes for each method to minimize the error in Euclidean distance to less than 1m. It was observed that the proposed ICP-NL method was much faster at converging. The height errors and convergence times are presented in Table 8.

Table 8. Aircraft simulation position estimation errors for ICP-NL and NDT algorithms for different altitudes.

Algorithm	**Height (m)**	**ATE (m)**	**Convergence Time (s)**
ICP-NL	3.5	0.7265	2.3
	4.5	0.7077	2.8
	5.5	0.7442	4
	6.5	0.6857	3.1
	7.5	0.6334	1.6
	8.5	0.552	4.1
NDT	3.5	149.6588	181.4
	4.5	511.6744	57.7
	5.5	363.2043	103.3
	6.5	133.5604	50.9
	7.5	0.6597	7
	8.5	1.6485	21.5

The simulated flight with the lowest ATE and the shortest convergence time, corresponding to a height of 7.5 m, was taken as a reference. Figure 25a shows the trajectories generated by the ICP-NL algorithm and the ground truth. The two trajectories are practically overlapped along the whole path, and the ATE and ABS errors in Figure 25b are within the margins allowed in the application. This method provides very good results in the simulation. Figure 26 shows the flight with the highest error.

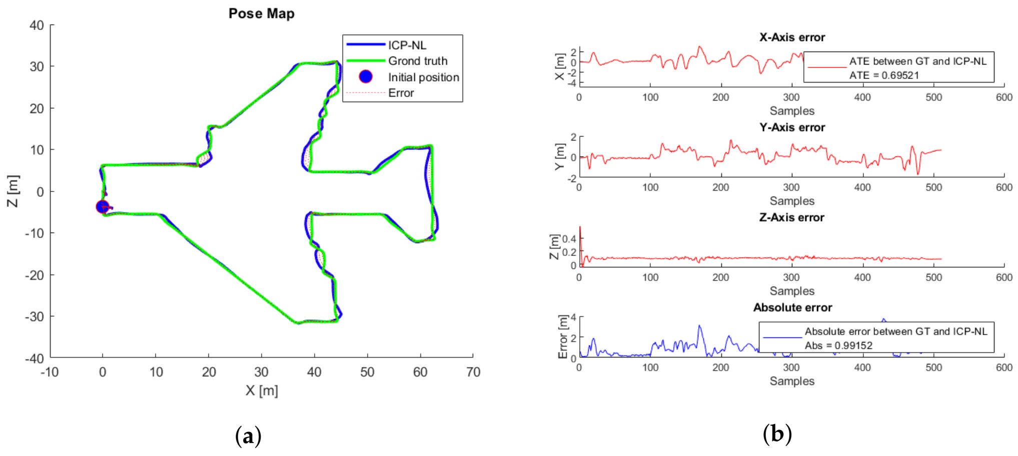


Figure 25. ICP-NL simulation test with the lowest ATE. (**a**) Trajectories. ICP-N (blue), ground truth (green), take-off position (blue circle), error (red). (**b**) ICP-N errors.

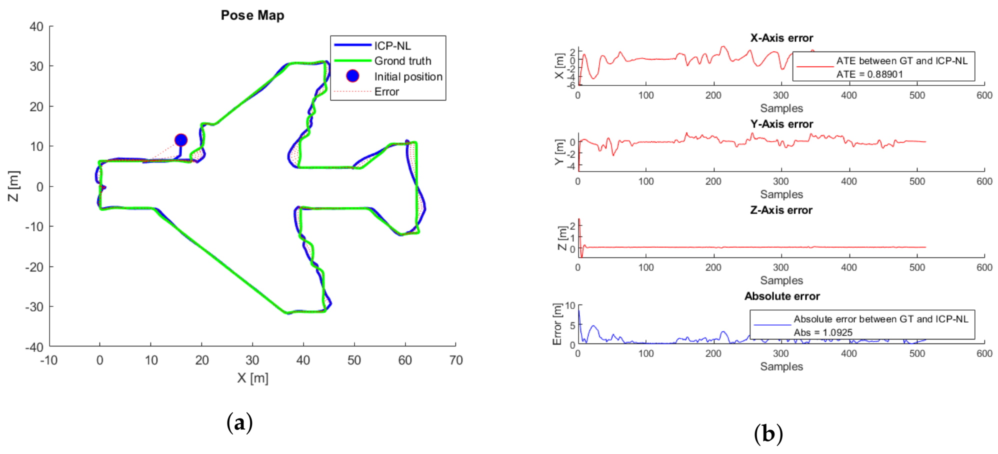


Figure 26. ICP-NL simulation test with the highest ATE. (**a**) Trajectories. ICP-NL (blue), ground truth (green), take-off position (blue circle), error (red). (**b**) ICP-N errors.

Like the 2D case, we evaluate the performance of the ICP-NL using the 3D CAD truck model. The result of the 3D map using the graph SLAM approach is shown in Figure 27, and Video S8 shows how a 3D map is built when the UAV is moving.

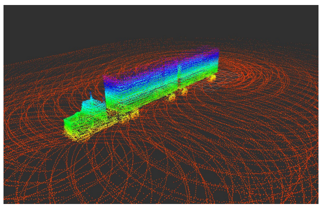


Figure 27. Truck 3D map generated by graph SLAM.

Based on previous results, only the ICP-NL approach was evaluated. Table 9 presents the results at different heights.

Table 9. Truck simulation position estimation errors for ICP-NL algorithms for different altitudes.

Algorithm	Height (m)	ATE (m)	Convergence Time (s)
ICP-NL	2.5	0.3801	2.3
	3.5	0.5552	2.8
	4.5	0.5442	4
	5.5	0.6857	3.1

Different flights were performed using the same aircraft 3D simulation parameters; each flight started at a different initial position with an initial error of less than one meter. Figures 28 and 29 show the flight with the lowest ATE, and Video S9 shows the pose estimation by ICP-NL during the flight. Further, Figure 30 shows the flight with the highest ATE.

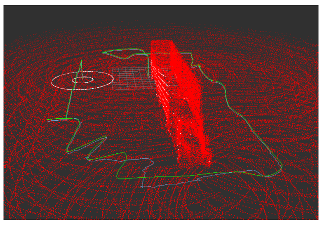
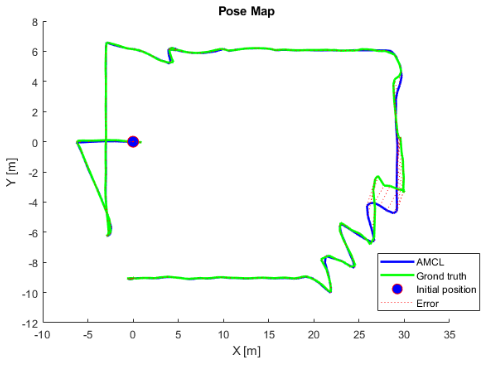
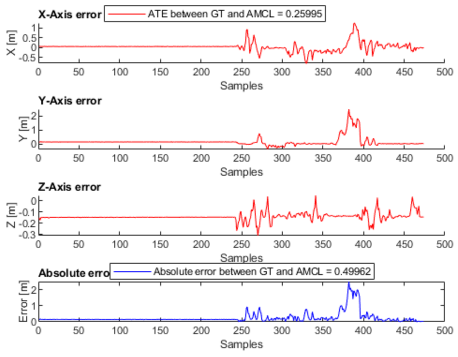


Figure 28. Pose estimation by ICP-NL. Three-dimensional map (red), ground truth path (green), ICP-NL path (blue).

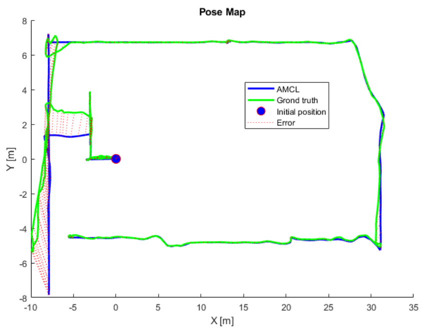


(**a**)

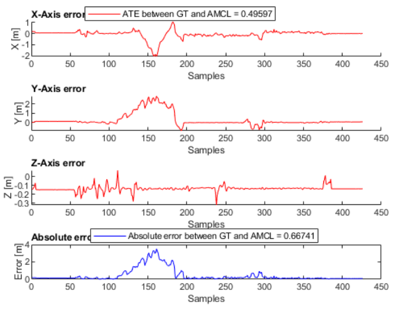


(**b**)

Figure 29. ICP-NL simulation test with the lowest ATE. (**a**) Trajectories. ICP-N (blue), ground truth (green), take-off position (blue circle), error (red). (**b**) ICP-N errors.



(**a**)



(**b**)

Figure 30. ICP-NL simulation test with the highest ATE. (**a**) Trajectories. ICP-NL (blue), ground truth (green), take-off position (blue circle), error (red). (**b**) ICP-N errors.

4.5.2. Real Tests

To check the method's effectiveness, tests were performed with real data as well, using the same files (ROS rosbags) as in the AMCL algorithm tests. Figure 31a and Video S5 correspond to the trajectory of flight Figures 16a and 31b and Video S6 correspond to flight Figure 16b. In Figure 31, the odometry trajectories have been represented in white. The algorithm's correct operation can be checked visually if the data captured by the sensor overlaps with the reference map's data and if the trajectory generated by the odometry is close to the trajectory provided by ICP-NL. However, in this case, the trajectory estimated by ICP-NL has many deviations, and the 3D map data does not match the 3D laser data. The main cause of these errors when evaluating the real environment methods is assumed to be the noise generated by the walls and other unexpected objects adjacent to the aircraft.

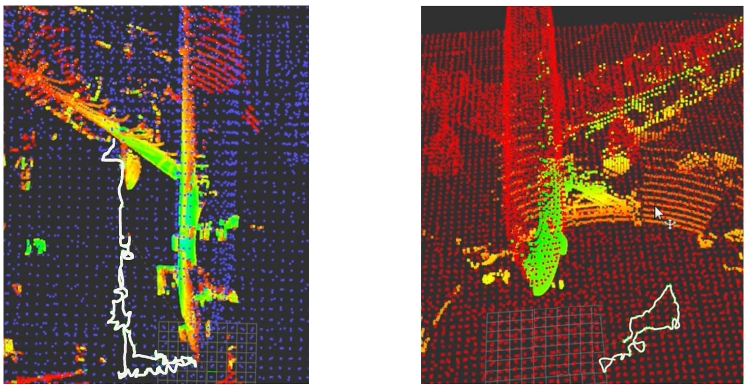

(**a**) (**b**)

Figure 31. Performance results of the 3D localization algorithm with real data. Position estimate by the algorithm (white), 3D point cloud (green-yellow). (**a**) Path 1 test (Figure 16a). 3D graph SLAM map (blue). (**b**) Path 2 test (Figure 16b). 3D graph SLAM map (red).

5. Discussions

5.1. 2D Localization

Using a static 2D map built from a standard CAD model has significant advantages over developing SLAM algorithms to map the environment, as it can save time and resources and provide enhanced robustness. Furthermore, using a 3D laser sensor for 2D environments is very practical, can be cost-effective, and can produce really nice results, as shown in this work.

Unexpected objects not present on the map can be segmented in a simple manner by modifying the laser data range. These parameters must be optimized according to the application and environment to obtain the desired results. Object segmentation can also be performed by applying other typical point cloud processing algorithms, if necessary.

If the map contains objects with particular geometries or noticeable corners, the detection of relevant features is evident. As more data are acquired, the filter will converge easily. In this work, the wings' intersection with the aircraft's body and the corners of the truck provides a unique feature. This area can be used to detect the relevant features. It provides unique features and, therefore, the area around can be used as a take-off point for the algorithm to converge faster.

When using AMCL, a considerable degree of uncertainty in the initial position is admissible if safe navigation is guaranteed, as the algorithm can estimate the location accurately after some time. For a flight to be robust and accurate, AMCL can be used in two stages. The first one aims to determine its initial position with minimal error by performing a specific trajectory and waiting for the algorithm to converge. The second stage starts the desired flight considering the initial estimated position. As expected, the lower error in AMCL initialization, the better the results in estimating the whole trajectory. Further, AMCL can be used to determine a reliable initial position with little error for other algorithms in which this information is critical, such as the EKF.

The results obtained from the truck inspection show that the covariance XY, laser data, and occupied cells are lower than the values from the aircraft inspection. Because the 2D map created and the truck is smaller, the data obtained by each component has less information.

5.2. 3D Localization

Regarding mapping, the graph SLAM changes applied for this work were adequate. The obtained map had little noise, which helped rule out sources of error in the localization algorithms execution. Additionally, the point cloud map could be sub-sampled to optimize processing time. In the simulations, we observed some false positives in pose estimation. Therefore, it is recommended to build the 3D map without using the floor in the world.

Despite the noise generated by elements other than the aircraft, we showed that a map generated in simulation could be used for localization in a real environment with nice results. However, it would be an exciting contribution to test the algorithms in an open field or far from noisy elements to corroborate their better operation.

Overall, the proposed ICP-NL scan-matching method and UKF performed better than NDT in the performed tests. Its behavior was robust, and its results were not affected by parameters such as flight height. Moreover, its convergence time was relatively short despite estimation errors at the initial position. However, the error may still be too high for applications requiring increased accuracy.

NDT achieves high accuracy in the estimation of the x and y coordinates in some of the flights. In the case of NDT, it works very well as long as the initial uncertainty does not grow too much. Otherwise, it has significant costs to recover.

According to the performed tests, nonlinear ICP supports initial positions with a more extensive range of uncertainty than similar algorithms such as NDT, ICP, and Generalized ICP.

6. Conclusions

6.1. 2D Localization

Performing 2D localization on a pre-built map using a CAD model and a 3D laser sensor is very practical. It saves resources and time, and the results are good.

Unexpected objects inside the environment may be deleted in a simple and fast manner by imposing 2D laser range limitations. However, the results can be improved by using PCL segmentation methods.

The initial position used in AMCL could have a considerable degree of uncertainty. Places with relevant features on the map should be used for the initial position to reduce convergence time and improve the quality of localization.

The position estimated by AMCL can be used as an initial position for other algorithms.

6.2. 3D Localization

Applying sub-sampling methods to the point cloud can reduce the time to build a map. To obtain higher accuracy in the 3D map, the robot's real position may be fed to the algorithm proposed by [27,64].

In our tests, the best results were obtained by the ICP-NL scan-matching method together with UKF. It is a robust algorithm; flight height does not affect its performance, it works well with an extensive range of uncertainty, and its convergence time is relatively short. NDT had a good performance in pose estimation in x and y with a long-range of uncertainty. However, in some cases, it presented failures, and the robot got lost.

Supplementary Materials: AMCL test in aircraft inspection, Video S1: https://www.youtube.com/watch?v=Uda-_YUio_8. AMCL test in real flight, Video S2: https://www.youtube.com/watch?v=FJlMPQJkjQ4. AMCL test in real flight 2, Video S3: https://youtu.be/tWqs_0hfP-s. ICP-N and NDT test in aircraft inspection, Video S4: https://www.youtube.com/watch?v=sm99JGC_RRo. ICP-NL pose estimation in real flight, Video S5: https://youtu.be/nIe6f0lWN5Y. ICP-NL pose estimation in real fligth 2, Video S6: https://youtu.be/ppRl1bkPFKs. AMCL test in truck inspection, Video S7: https://youtu.be/kYRqHP-5KGM. Graph SLAM map creation, Video S8: https://youtu.be/tT7vpYPzhsk. ICP-N test in truck inspection, Video S9: https://youtu.be/kumKJLJX75M.

Author Contributions: Conceptualization, P.d.l.P., H.B., P.E.P., A.C. and P.C.; methodology, P.d.l.P., P.E.P., A.C. and P.C.; software, H.B., P.E.P., M.A.L. and A.C.; validation, H.B., P.E.P., M.A.L., A.C. and P.C.; formal analysis, P.E.P., M.A.L., A.C., C.C.U. and P.C.; investigation, P.E. and P.C.; resources, P.E.P., M.A.L. and P.C.; data curation, P.E.P. and M.A.L.; writing—original draft preparation, P.E.P., M.A.L. and C.C.U.; writing—review and editing, P.E.P., M.A.L., P.d.l.P. and C.C.U.; visualization, P.E.P., M.A.L., A.C., C.C.U., P.d.l.P. and P.C.; supervision, H.B., A.C., P.d.l.P. and P.C.; project administration, H.B., P.d.l.P. and P.C.; funding acquisition, P.d.l.P. and P.C. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Not applicable.

Acknowledgments: We would thank the Computer Vision and Aerial Robotics Group, Centre for Automation and Robotics, and Universidad Politécnica de Madrid for the provision of their laboratories and equipment for our research. We would also like to thank the government of Ecuador for covering the study grant for the principal author.

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

Abbreviations

The following abbreviations are used in this manuscript:

UAV	Unmanned Aerial Vehicle
CAD	Computer-Aided Design
3D	Three Dimensions
2D	TWO Dimensions
UKF	Unscented Kalman Filter
EKF	Extended Kalman Filter
IMU	Inertial Measurement Units
GPS	Global Positioning System
ICP	Iterative Closest Point Algorithm
ICP-NL	Noun lineal Iterative Closest Point Process
NDT	Normal Transformation Distribution
SLAM	Simultaneous Localization and Mapping)
MCL	Monte Carlo Localization
AMCL	Adaptive Monte Carlo Localization
ROS	Robot Operative System
DAE	Digital Asset Exchange
URDF	Unified Robot Description Format
ATE	Absolute Trajectory Error
MSE	Mean Square Error
PCL	Point Cloud Laser Library
GT	Ground Truth
PC	Personal Computer
SVD	Singular Value Decomposition
LIDAR	Light Detection and Ranging

Appendix A. Hummingbird Simulated Odometry Sensor

```
file: /rotors_simulator/rotors_description/urdf/mav_generic_odometry_sensor.gazebo
mass_odometry_sensor = 0.00001
measurement_divisor = 1
measurement_delay = 0
unknown_delay = 0.0
noise_normal_position = 0 0 0
noise_normal_quaternion = 0 0 0
noise_normal_linear_velocity = 0 0 0
noise_normal_angular_velocity = 0 0 0
noise_normal_position = 0.01 0.01 0.01
noise_normal_quaternion = 0.017 0.017 0.017
noise_uniform_linear_velocity = 0 0 0
noise_uniform_angular_velocity = 0 0 0
enable_odometry_map=false
inertia ixx = 0.00001 ixy = 0.0 ixz = 0.0 iyy = 0.00001 iyz = 0.0 izz = 0.00001 [
    kg m^2]
origin xyz=0.0 0.0 0.0 rpy=0.0 0.0 0.0
```

Appendix B. Velodyne VLP16 Simulated 3D Laser Sensor

```
update rate in hz = 10
samples = 512
minimum range value in meters = 0.9
maximum range value in meters = 130
noise Gausian in meters = 0.008
minimum horizontal angle in radians = −3.14
maximum horizontal angle in radians = 3.14
```

Appendix C. AMCL Simulations Parameters

```
odom_model_type value=omni−corrected

laser_max_beams value=30
min_particles value=200
max_particles value=3000
kld_err value=0.05
kld_z value=0.99

odom_alpha1 value=0.2
odom_alpha2 value=0.2
odom_alpha3 value=0.8
odom_alpha4 value=0.2
odom_alpha5 value=0.2


laser_likelihood_max_dist value=2
laser_z_hit value=0.5
laser_z_short value=0.05
laser_z_max value=0.05
laser_z_rand value=0.5
laser_sigma_hit value=0.2
laser_lambda_short value=0.1
laser_model_type value=likelihood_field

update_min_d value=0.2
update_min_a value=0.5
resample_interval value=1
transform_tolerance value=1.0
recovery_alpha_slow value=0.001
recovery_alpha_fast value=0.1

initial_cov_xx  value=0.5
initial_cov_yy  value=0.5
initial_cov_aa  value=0.15
```

Appendix D. AMCL Real Fligts Parameters

```
odom_model_type value=omni−corrected

laser_max_beams value=50
min_particles value=200
max_particles value=3000
kld_err value=0.01
kld_z value=0.8

odom_alpha1 value=0.3
odom_alpha2 value=0.3
odom_alpha3 value=0.05
odom_alpha4 value=0.05
odom_alpha5 value=0.3


laser_likelihood_max_dist value=0.5
laser_z_hit value=0.8
laser_z_short value=0.05
laser_z_max value=0.05
laser_z_rand value=0.5
laser_sigma_hit value=0.2
laser_lambda_short value=0.1
laser_model_type value=likelihood_field

update_min_d value=0.1
update_min_a value=0.1
resample_interval value=1
transform_tolerance value=1.0
recovery_alpha_slow value=0.001
recovery_alpha_fast value=0.1

initial_cov_xx   value=25
initial_cov_yy   value=25
initial_cov_aa   value=0.15
```

References

1. Myeong , W.C.; Jung, K.Y.; Jung, S.W.; Jung, Y.; Myung, H. Development of a drone-type wall-sticking and climbing robot. In Proceedings of the 2015 12th International Conference on Ubiquitous Robots and Ambient Intelligence (URAI), Goyangi, Korea, 28–30 October 2015; IEEE: Piscataway, NJ, USA, 2015; pp. 386–389.
2. Nikolic, J.; Burri, M.; Rehder, J.; Leutenegger, S.; Huerzeler, C.; Siegwart, R. A UAV system for inspection of industrial facilities. In Proceedings of the 2013 IEEE Aerospace Conference, Big Sky, MT, USA, 2–9 March 2013; IEEE: Piscataway, NJ, USA, 2013; pp. 1–8.
3. Omari, S.; Gohl, P.; Burri, M.; Achtelik, M.; Siegwart, R. Visual industrial inspection using aerial robots. In Proceedings of the 2014 3rd International Conference on Applied Robotics for the Power Industry, Foz do Iguassu, Brazil, 14–16 October 2014; IEEE: Piscataway, NJ, USA, 2014; pp. 1–5.
4. Menendez, O.A.; Perez, M.; Cheein, F.A.A. Vision based inspection of transmission lines using unmanned aerial vehicles. In Proceedings of the 2016 IEEE International Conference on Multisensor Fusion and Integration for Intelligent Systems, Kongresshaus Baden-Baden, Germany, 19–21 September 2016; IEEE: Piscataway, NJ, USA, 2016; pp. 412–417.
5. Jiang, W.; Wenkai, F.; Qianru, L. An integrated measure and location method based on airborne 2D laser scanning sensor for UAV's power line inspection. In Proceedings of the 2013 Fifth International Conference on Measuring Technology and Mechatronics Automation, Hong Kong, China, 16–17 January 2013; IEEE: Piscataway, NJ, USA, 2013; pp. 213–217.
6. Zhang, J.; Liu, L.; Wang, B.; Chen, X.; Wang, Q.; Zheng, T. High speed automatic power line detection and tracking for a UAV-based inspection. In Proceedings of the 2012 International Conference on Industrial Control and Electronics Engineering, Xi'an, China, 23–25 August 2012; IEEE: Piscataway, NJ, USA, 2012; pp. 266–269.
7. Alsalam, B.H.Y.; Morton, K.; Campbell, D.; Gonzalez, F. Autonomous UAV with vision based on-board decision making for remote sensing and precision agriculture. In Proceedings of the 2017 IEEE Aerospace Conference, Big Sky, MT, USA, 4–11 March 2017; IEEE: Piscataway, NJ, USA, 2017; pp. 1–12.
8. Valente, J.; Kooistra, L.; Mücher, S. Fast Classification of Large Germinated Fields Via High-Resolution UAV Imagery. *IEEE Robot. Autom. Lett.* **2019**, *4*, 3216–3223. [CrossRef]

9. Tripicchio, P.; Satler, M.; Dabisias, G.; Ruffaldi, E.; Avizzano, C.A. Towards smart farming and sustainable agriculture with drones. In Proceedings of the 2015 International Conference on Intelligent Environments, Prague, Czech Republic, 15–17 July 2015; IEEE: Piscataway, NJ, USA, 2015; pp. 140–143.
10. Cerro, J.d.; Cruz Ulloa, C.; Barrientos, A.; de León Rivas, J. Unmanned Aerial Vehicles in Agriculture: A Survey. *Agronomy* **2021**, *11*, 203. [CrossRef]
11. Xi, Z.; Lou, Z.; Sun, Y.; Li, X.; Yang, Q.; Yan, W. A Vision-Based Inspection Strategy for Large-Scale Photovoltaic Farms Using an Autonomous UAV. In Proceedings of the 2018 17th International Symposium on Distributed Computing and Applications for Business Engineering and Science(DCABES), Wuxi, China, 9–23 October 2018; IEEE: Piscataway, NJ, USA, 2018; pp. 200–203.
12. Stokkeland, M.; Klausen, K.; Johansen, T.A. Autonomous visual navigation of unmanned aerial vehicle for wind turbine inspection. In Proceedings of the 2015 International Conference on Unmanned Aircraft Systems (ICUAS), Denver, CO, USA, 9–12 June 2015; IEEE: Piscataway, NJ, USA, 2015; pp. 998–1007.
13. Schäfer, B.E.; Picchi, D.; Engelhardt, T.; Abel, D. Multicopter unmanned aerial vehicle for automated inspection of wind turbines. In Proceedings of the 2016 24th Mediterranean Conference on Control and Automation (MED), Athens, Greece, 21–24 June 2016; IEEE: Piscataway, NJ, USA, 2016; pp. 244–249.
14. Bailey, T.; Durrant-Whyte, H. Simultaneous localization and mapping (SLAM): Part II. *IEEE Robot. Autom. Mag.* **2006**, *13*, 108–117. [CrossRef]
15. Cadena, C.; Carlone, L.; Carrillo, H.; Latif, Y.; Scaramuzza, D.; Neira, J.; Reid, I.; Leonard, J.J. Past, Present, and Future of Simultaneous Localization and Mapping: Toward the Robust-Perception Age. *IEEE Trans. Robot.* **2016**, *32*, 1309–1332. [CrossRef]
16. Avola, D.; Cinque, L.; Fagioli, A.; Foresti, G.L.; Massaroni, C.; Pannone, D. Feature-based SLAM algorithm for small scale UAV with nadir view. In Proceedings of the International Conference on Image Analysis and Processing, Trento, Italy, 9–13 September, 2019; Springer: Berlin/Heidelberg, Germany, 2019; pp. 457–467.
17. Chan, S.H.; Wu, P.T.; Fu, L.C. Robust 2D indoor localization through laser SLAM and visual SLAM fusion. In Proceedings of the 2018 IEEE International Conference on Systems, Man, and Cybernetics (SMC), Miyazaki, Japan, 7–10 October 2018; IEEE: Piscataway, NJ, USA, 2018; pp. 1263–1268.
18. Luo, Y.; Li, Y.; Li, Z.; Shuang, F. MS-SLAM: Motion State Decision of Keyframes for UAV-Based Vision Localization. *IEEE Access* **2021**, *9*, 67667–67679. [CrossRef]
19. Julier, S.J.; Uhlmann, J.K. New extension of the Kalman filter to nonlinear systems. In Proceedings of the Signal Processing, Sensor Fusion, and Target Recognition VI. International Society for Optics and Photonics, Orlando, FL, USA, 28 July 1997; Volume 3068, pp. 182–193.
20. Tang, H.; Shen, Z. An attitude estimate method for fixed-wing UAV s using MEMS/GPS data fusion. In Proceedings of the 2017 First International Conference on Electronics Instrumentation & Information Systems (EIIS), Harbin, China, 3–5 June 2017; IEEE: Piscataway, NJ, USA, 2017; pp. 1–5.
21. Mourikis, A.I.; Roumeliotis, S.I. A multi-state constraint Kalman filter for vision-aided inertial navigation. In Proceedings of the 2007 IEEE International Conference on Robotics and Automation, Roma, Italy, 10–14 April 2007; IEEE: Piscataway, NJ, USA, 2007; pp. 3565–3572.
22. Wan, E.A.; Van Der Merwe, R. The unscented Kalman filter for nonlinear estimation. In Proceedings of the IEEE 2000 Adaptive Systems for Signal Processing, Communications, and Control Symposium (Cat. No.00EX373), Lake Louise, AB, Canada, 1–4 October 2000; IEEE: Piscataway, NJ, USA, 2000; pp. 153–158.
23. De Marina, H.G.; Pereda, F.J.; Giron-Sierra, J.M.; Espinosa, F. UAV attitude estimation using unscented Kalman filter and TRIAD. *IEEE Trans. Ind. Electron.* **2011**, *59*, 4465–4474. [CrossRef]
24. Burguera, A.; Oliver, G.; Tardos, J.D. Robust scan matching localization using ultrasonic range finders. In Proceedings of the 2005 IEEE/RSJ International Conference on Intelligent Robots and Systems,Alberta, Canada, 2–6 August 2005; IEEE: Piscataway, NJ, USA, 2005; pp. 1367–1372.
25. Fang, H.; Yang, M.; Yang, R. Ground texture matching based global localization for intelligent vehicles in urban environment. In Proceedings of the 2007 IEEE Intelligent Vehicles Symposium, Istanbul, Turkey, 13–15 June 2007; IEEE: Piscataway, NJ, USA, 2007; pp. 105–110.
26. Nazem, F.; Ahmadian, A.; Seraj, N.D.; Giti, M. Two-stage point-based registration method between ultrasound and CT imaging of the liver based on ICP and unscented Kalman filter: A phantom study. *Int. J. Comput. Assist. Radiol. Surg.* **2014**, *9*, 39–48. [CrossRef] [PubMed]
27. Koide, K.; Miura, J.; Menegatti, E. A portable three-dimensional LIDAR-based system for long-term and wide-area people behavior measurement. *Int. J. Adv. Robot. Syst.* **2019**, *16*, 1729881419841532. [CrossRef]
28. Magnusson, M.; Lilienthal, A.; Duckett, T. Scan registration for autonomous mining vehicles using 3D-NDT. *J. Field Robot.* **2007**, *24*, 803–827. [CrossRef]
29. Sakai, T.; Koide, K.; Miura, J.; Oishi, S. Large-scale 3d outdoor mapping and on-line localization using 3d-2d matching. In Proceedings of the 2017 IEEE/SICE International Symposium on System Integration (SII), Taipei, Taiwan, 11–14 December 2017; IEEE: Piscataway, NJ, USA, 2017; pp. 829–834.
30. Zhu, H.; Chung, J.J.; Lawrance, N.R.; Siegwart, R.; Alonso-Mora, J. Online informative path planning for active information gathering of a 3d surface. In Proceedings of the 2021 IEEE International Conference on Robotics and Automation (ICRA), Xi'an, China, 30 May–5 June 2021; IEEE: Piscataway, NJ, USA, 2021; pp. 1488–1494.

31. Fox, D.; Burgard, W.; Thrun, S. Markov localization for mobile robots in dynamic environments. *J. Artif. Intell. Res.* **1999**, *11*, 391–427. [CrossRef]
32. Shoukry, Y.; Abdelfatah, W.F.; Hammad, S.A. Real-time Markov localization for autonomous UGV. In Proceedings of the 2009 4th International Design and Test Workshop (IDT), Riyadh, Saudi Arabia, 15–17 November 2009; IEEE: Piscataway, NJ, USA, 2009; pp. 1–6.
33. Thrun, S.; Fox, D.; Burgard, W.; Dellaert, F. Robust Monte Carlo localization for mobile robots. *Artif. Intell.* **2001**, *128*, 99–141. [CrossRef]
34. Fox, D.; Burgard, W.; Dellaert, F.; Thrun, S. Monte carlo localization: Efficient position estimation for mobile robots. *AAAI/IAAI* **1999**, *1999*, 2.
35. Xu, S.; Chou, W. An improved indoor localization method for mobile robot based on WiFi fingerprint and AMCL. In Proceedings of the 2017 10th International Symposium on Computational Intelligence and Design (ISCID), Hangzhou, China, 9–10 December 2017; IEEE: Piscataway, NJ, USA, 2017; Volume 1, pp. 324–329.
36. Javierre, P.; Alvarado, B.P.; de la Puente, P. Particle filter localization using visual markers based omnidirectional vision and a laser sensor. In Proceedings of the 2019 Third IEEE International Conference on Robotic Computing (IRC),Naples, Italy, 25–27 February 2019; IEEE: Piscataway, NJ, USA, 2019; pp. 246–249.
37. Yilmaz, A.; Temeltas, H. Self-adaptive Monte Carlo method for indoor localization of smart AGVs using LIDAR data. *Robot. Auton. Syst.* **2019**, *122*, 103285. [CrossRef]
38. Siegel, M.; Gunatilake, P.; Podnar, G. Robotic assistants for aircraft inspectors. *IEEE Instrum. Meas. Mag.* **1998**, *1*, 16–30. [CrossRef]
39. Rice, M.; Li, L.; Ying, G.; Wan, M.; Lim, E.T.; Feng, G.; Ng, J.; Teoh Jin-Li, M.; Babu, V.S. Automating the visual inspection of aircraft. In Proceedings of the Singapore Aerospace Technology and Engineering Conference (SATEC), Suntec, Singapur, 7 February 2018.
40. Ramalingam, B.; Manuel, V.H.; Elara, M.R.; Vengadesh, A.; Lakshmanan, A.K.; Ilyas, M.; James, T.J.Y. Visual inspection of the aircraft surface using a teleoperated reconfigurable climbing robot and enhanced deep learning technique. *Int. J. Aerosp. Eng.* **2019**, *2019*, 5137139. [CrossRef]
41. Donadio, F.; Frejaville, J.; Larnier, S.; Vetault, S. Human-robot collaboration to perform aircraft inspection in working environment. In Proceedings of the 5th International Conference on Machine Control and Guidance (MCG),Vichy, France, 5–6 October 2016.
42. Blokhinov, Y.B.; Gorbachev, V.; Nikitin, A.; Skryabin, S. Technology for the Visual Inspection of Aircraft Surfaces Using Programmable Unmanned Aerial Vehicles. *J. Comput. Syst. Sci. Int.* **2019**, *58*, 960–968. [CrossRef]
43. Miranda, J.; Larnier, S.; Herbulot, A.; Devy, M. UAV-based inspection of airplane exterior screws with computer vision. In Proceedings of the 14h International Joint Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications, Prague, Czech Republic, 25–27 February 2019.
44. Hornung, A.; Wurm, K.M.; Bennewitz, M.; Stachniss, C.; Burgard, W. OctoMap: An Efficient Probabilistic 3D Mapping Framework Based on Octrees. *Auton. Robot.* **2013**, *34*, 189–206. [CrossRef]
45. Jian, X. Occupancy Grids Maps for Localization and Mapping. In *Motion Planning*; Intech: London, UK, 2008; pp. 381–406.
46. Thrun, S.; Montemerlo, M. The graph SLAM algorithm with applications to large-scale mapping of urban structures. *Int. J. Robot. Res.* **2006**, *25*, 403–429. [CrossRef]
47. Fox, D.; Burgard, W.; Thrun, S. Grid and Monte Carlo Localization. In *Probalistic Robotics*; Early Draft; 2000; pp. 187–219. Available online: https://docs.ufpr.br/~danielsantos/ProbabilisticRobotics.pdf (accessed on 11 April 2022).
48. Fantoni, S.; Castellani, U.; Fusiello, A. Accurate and automatic alignment of range surfaces. In Proceedings of the 2012 Second International Conference on 3D Imaging, Modeling, Processing, Visualization Transmission, Zurich, Switzerland, 13–15 October 2012; IEEE: Piscataway, NJ, USA, 2012; pp. 73–80.
49. Rusu, R.B.; Cousins, S. 3d is here: Point cloud library (pcl). In Proceedings of the 2011 IEEE International Conference on Robotics and Automation, Shanghai, China, 9–13 May 2011; IEEE: Piscataway, NJ, USA, 2011; pp. 1–4.
50. Besl, P.J.; McKay, N.D. Method for registration of 3-D shapes. In Proceedings of the Sensor Fusion IV: Control Paradigms and Data Structures, Boston, MA, United States, 30 April 1992; International Society for Optics and Photonics: Bellingham, WA, USA, 1992; Volume 1611, pp. 586–606.
51. Ros. Available online: https://www.ros.org/ (accessed on 11 April 2022).
52. Gazebo Robot Simulation. Available online: http://gazebosim.org/ (accessed on 11 April 2022).
53. Aerostack. Available online: https://github.com/cvar-upm/aerostack/wiki (accessed on 11 April 2022).
54. Furrer, F.; Burri, M.; Achtelik, M.; Siegwart, R. *Robot Operating System (ROS): The Complete Reference (Volume 1)*; Chapter RotorS—A Modular Gazebo MAV Simulator Framework; Springer: Cham, Switzerland, 2016; pp. 595–625. [CrossRef]
55. Velodyne Simulator. Available online: https://bitbucket.org/DataspeedInc/velodyne_simulator/src/master/ (accessed on 11 April 2022).
56. Moore, T.; Stouch, D. A Generalized Extended Kalman Filter Implementation for the Robot Operating System. In Proceedings of the 13th International Conference on Intelligent Autonomous Systems (IAS-13), Padova, Italy, 15–18 July 2014; Springer: Berlin/Heidelberg, Germany, 2014.
57. Airbus Cad Model. Available online: https://www.thingiverse.com/thing:847897 (accessed on 11 April 2022).
58. Aircraft Airbus. Available online: https://commons.wikimedia.org/wiki/File:Arrival_of_Iran_Air_Airbus_A330-200_%28EP-IJA%29_at_Mehrabad_International_Airport_%2812%29.jpg (accessed on 11 April 2022).

59. Sturm, J.; Engelhard, N.; Endres, F.; Burgard, W.; Cremers, D. A benchmark for the evaluation of RGB-D SLAM systems. In Proceedings of the 2012 IEEE/RSJ International Conference on Intelligent Robots and Systems, Vilamoura-Algarve, Portugal, 7–12 October 2012; IEEE: Piscataway, NJ, USA, 2012; pp. 573–580.
60. Absolute Trajectory Error. Available online: http://www.rawseeds.org/rs/methods/view/9l (accessed on 11 April 2022).
61. Anagnostopoulos, I.; Pătrăucean, V.; Brilakis, I.; Vela, P. Detection of walls, floors, and ceilings in point cloud data. In Proceedings of the Construction Research Congress 2016, San Juan, Puerto Rico, 31 May–2 June 2016; pp. 2302–2311.
62. Chum, O.; Matas, J.; Kittler, J. Locally optimized RANSAC. In Proceedings of the Joint Pattern Recognition Symposium, Magdeburg, Germany, 10–12 September 2003; Springer: Berlin/Heidelberg, Germany, 2003; pp. 236–243.
63. Point Cloud Library. Available online: https://pcl.readthedocs.io/projects/tutorials/en/latest/planar_segmentation.html#planar-segmentation (accessed on 11 April 2022).
64. Montemerlo, M.; Thrun, S.; Koller, D.; Wegbreit, B. FastSLAM: A factored solution to the simultaneous localization and mapping problem. *AAAI/IAAI* **2002**, 593598. Available online: https://www.aaai.org/Papers/AAAI/2002/AAAI02-089.pdf (accessed on 11 April 2022).