2016 International Conference on
Unmanned Aircraft Systems (ICUAS)
June 7-10, 2016. Arlington, VA USA



# LIDAR-Inertial Integration for UAV Localization and Mapping in Complex Environments

Roberto Opromolla, Giancarmine Fasano,
Giancarlo Rufino, Michele Grassi
Department of Industrial Engineering
University of Naples Federico II
Naples, Italy
roberto.opromolla@unina.it,
giancarmine.fasano@unina.it,
giancarlo.rufino@unina.it, michele.grassi@unina.it

Al Savvaris
School of Aerospace, Transport and Manufacturing
Cranfield University
Cranfield, United Kingdom
a.savvaris@cranfield.ac.uk

***Abstract*— This paper presents customized techniques for autonomous localization and mapping of micro Unmanned Aerial Vehicles flying in complex environments, e.g. unexplored, full of obstacles, GPS challenging or denied. The proposed algorithms are aimed at 2D environments and are based on the integration of 3D data, i.e. point clouds acquired by means of a laser scanner (LIDAR), and inertial data given by a low cost Inertial Measurement Unit (IMU). Specifically, localization is performed by exploiting a scan matching approach based on a customized version of the Iterative Closest Point algorithm, while mapping is done by extracting robust line features from LIDAR measurements. A peculiarity of the line detection method is the use of the Principal Component Analysis which allows computational time saving with respect to traditional least squares techniques for line fitting. Performance of the proposed approaches is evaluated on real data acquired in indoor environments by means of an experimental setup including an UTM-30LX-EW 2D LIDAR, a Pixhawk IMU, and a Nitrogen board.**

***Keywords*— LIDAR; inertial sensors; localization; mapping; UAV; line detection***

## I. INTRODUCTION

Over the last years, great interest has arisen towards the necessity of developing innovative techniques aimed at improving the level of autonomy of Unmanned Aerial Vehicles (UAV), especially when flying in complex environments. This term generally refers to those mission profiles which can be both indoor and outdoor, which require the UAV to fly into unknown areas, potentially dangerous to the human life and could incorporate static and/or mobile obstacles.

In the case of widely open outdoor scenarios, where it is possible to detect reliable satellite signals, autonomous safe navigation of UAVs can be ensured by integrating the Inertial Measurement Unit (IMU) with the Global Navigation Satellite System (GNSS) received signal, e.g. GPS, utilizing sensor fusion architecture typically indicated as GPS-INS [1, 2]. This solution, usually based on an Extended Kalman Filter (EKF), has been proved reliable by many researchers over the last few decades both in the case of fixed wing and rotary-wing UAVs [3, 4]. On the other hand, there are several scenarios covering both military and civilian applications (e.g. urban surveillance and pipeline monitoring), the GPS signal may be completely absent (i.e. indoor) or unreliable due to multipath, absorption and jamming phenomena (e.g. in urban or natural canyons and under forest foliage). These environments, also known as GPS-denied or GPS-challenging respectively, require the adoption of alternative hardware and algorithmic solutions in order to ensure high-level state estimation and perception capabilities to Micro-UAVs (MAVs). Specifically, this result can be achieved for example by integrating inertial with Electro-Optical sensors or other ranging systems. Nevertheless, it is worth outlining that even though the GPS signal is available, the adoption of these technologies might still be required in order to get a higher level of autonomy according to the ranking provided by Kendoul about Guidance, Navigation and Control capabilities of Rotorcraft Unmanned Aircraft Systems (RUAS) [5].

The sensors potentially suitable for this kind of applications can be active or passive systems. The former category mainly refers to Light Detection and Ranging (LIDAR) instruments, but it also includes ultrasonic rangefinders and RADAR, while the latter one refers to monocular and stereovision cameras operating in the visible band of the electromagnetic spectrum. A hybrid alternative is given by RGB-depth cameras, which are able to simultaneously register passive RGB images as well as depth images of the same scene in an active way, i.e. by sending a structured pattern of infrared (IR) light and analyzing the echo coming from background or measuring time-of-flight [6]. Although passive systems are certainly lighter in terms of weight, less expensive and less power consuming than their active counter-parts, this work focuses on active sensors. Indeed, they are less sensitive to ambient light

978-1-4673-9333-1/16/$31.00 ©2016 IEEE

variations and can work day and night, thus providing an higher degree of autonomy compared to passive cameras. Moreover, they can provide directly 3D information about the observed scene without requiring any computationally expensive image processing approach. Among the previously mentioned active sensors, LIDARs represent a convenient choice since they can provide measurements at farther operating ranges than ultrasonic rangefinders and depth cameras. However, if one consider radar systems, they are certainly able to provide, almost in any weather condition, high resolution and wide-area coverage thus representing an optimal solution for collision detection for larger UAVs. The problem is that they are still too heavy and power consuming, and produce large amount of data, to be installed on board any kind of MAV. However, it is worth mentioning that great amount of research efforts are currently in progress towards the integration of compact radar systems on board MAVs [7].

In this framework, Simultaneous Localization And Mapping (SLAM) is a key enabling function to perform missions in complex environments, both indoor and outdoor. Although the SLAM problem can be considered completely solvable from the theoretical point of view [8], and several improvements were carried out in the last two decades [9], there are still many open issues regarding its real-time implementation which are particularly relevant to flying robots [10]. Hence, this paper presents innovative techniques, for the two main steps of SLAM (localization and mapping), which are based on the integration of laser scan data provided by a two-dimensional (2D) LIDAR, and inertial measurements, i.e. vehicle's acceleration, angular velocity and attitude given by a low cost IMU. At this stage of the research activity the two steps are separated meaning that no feedback on the localization solution is foreseen from mapping information.

With regards to the localization step of the SLAM process, it consists in estimating the vehicle's trajectory, i.e. the time evolution of its attitude and position parameters (pose). One possible solution to this problem is to perform LIDAR odometry through scan matching, which means to recursively compute the pose variation between two successive time instants by comparing the corresponding LIDAR scans. Three different approaches to scan matching exist, i.e. feature-to-feature, point-to-feature and point-to-point algorithms [11]. The attention here is focused on the latter category since it is more robust in cluttered environments where it is difficult to identify peculiar structures in the acquired datasets, and it allows saving on the required computational load for the feature extraction step. Hence, a customized version of the Iterative Closest Point (ICP) algorithm [12] is here proposed to carry out point-to-point scan matching. With respect to previous ICP applications for SLAM, mainly addressed at optimizing the computational time required by the matching step [13], peculiarities of this approach are the introduction of an outlier rejection step to cope with the errors caused by wrong point correspondence determination as well as of a strategy for autonomous failure detection. Other examples of point-to-point scan matching techniques for autonomous localization of MAVs can be found in [10] and [14].

The mapping step of the SLAM process aims at building a map of the observed environment as well as at using the collected information to improve the accuracy of the estimated trajectory. However, mapping can also be used as an independent processing step that complements localization/odometry In addition to the raw data representation [15] which can lead to a huge amount of data to be stored, two approaches are mainly adopted in the literature to carry out this task, namely the occupancy grid [14, 16] and the feature-based ones [17, 18]. In this work, the latter approach is preferred to the occupancy grid method since it leads to compact representations of the environment with a high speed of execution and little memory requirement [18], thus being more compliant with MAVs' limitation both in terms of weight and power consumption. Of course, the use of feature-based mapping involves the associated drawbacks to be considered, e.g. possible loss of information due to sparse representation of the environment, difficulties intrinsic to the feature extraction and the data association processes. An advantage of the proposed approach is the use of the of PCA [19] for line fitting, which ensures a good improvement in terms of computational load when compared to the classical least squares method [18, 20, 21].

Hereinafter, the paper is organized as follows. Section II describes in detail the solutions envisaged for LIDAR/inertial localization and line-based mapping. Section III presents the setup and the results of the experimental tests carried out for performance evaluation. Finally, section IV contains the conclusion.

## II. LIDAR/Inertial Localization And Mapping

Before entering the details of the techniques developed for localization and mapping, some preliminary information regarding the mathematical notation are provided. Specifically, italic type is used for scalar quantities and quaternion, italic type with a single underline is used for vectors, and italic type with double underline is used for matrices. The pose of the MAV is described by a set of 6 parameters representing the position and the attitude of its body reference frame (BRF) with respect to a local inertial reference frame, i.e. the East-North-Up (ENU). Specifically, $\underline{T}$ is the 3D position vector of the MAV with respect to the ENU and expressed in the ENU, while a 321 sequence of Euler angles, i.e. heading ($\gamma$), pitch ($\beta$), and roll ($\alpha$), is used to represent the attitude of the BRF with respect to the ENU. For the sake of clarity, two assumptions are made. Firstly, the MAV's BRF is considered coincident and aligned with the LIDAR reference frame (LRF, having its $x$-axis in the boresight direction, its $z$-axis perpendicular to the scan plane and the $y$-axis oriented to obtain a right-handed reference frame). Secondly, the ENU origin is assumed to be coincident with the initial position of the MAV. However, the proposed approach can be easily extended to a more general case.

### A. Localization by scan matching

Localization is carried out by a hybrid LIDAR/Inertial Odometry algorithm (L/I-O). This means that the flying vehicle pose is tracked by integrating attitude information from an IMU with position information obtained by recursively registering two consecutive scans provided by a 2D LIDAR by means of a scan matching approach. Of course, the initial pose of the MAV must be fully known at the start of the trajectory, in order to be compliant with the concept of odometry [22]. Generally speaking, 2D scan matching is the problem of registering two sets of 2D data (i.e. a reference scan and a current scan) by looking, in the pose search space, for the optimal rotation and translation, i.e. the ones that provide the best alignment by minimizing a purposely defined error function [23]. Typically, the reference scan represents the environment in which the vehicle moves and it can be a pre-built map or a previous scan, while the current scan is the measurement dataset provided by the available range sensor at the time of interest.

The L/I-O algorithm extracts the attitude directly from an IMU at high frequency (about 90 Hz), while the position is estimated at lower frequency (up to 36 Hz) using a LIDAR odometry technique based on a point-to-point scan matching algorithm. A block diagram to describe this system architecture is provided in Fig. 1.

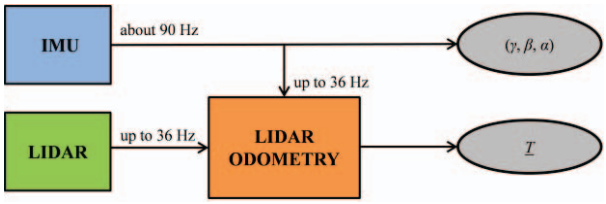


Fig. 1. Block diagram describing the system architecture for localization.

The scan matching algorithm is described in detail as follows. Let $t_k$ and $t_{k+1}$ be two successive time instants at which two LIDAR scans are acquired and let $\underline{\underline{P}}^{LRF}(t_k)$ and $\underline{\underline{P}}^{LRF}(t_{k+1})$ be the corresponding point clouds in LRF. Firstly, a coordinate transformation is applied to convert these point clouds from LRF to the Vehicle Reference Frame (VRF), which is a new frame aligned to ENU with the origin at the current position of the MAV (and so coincident to the LRF origin). This is done as shown in (1)

$$\begin{aligned} \underline{\underline{P}}^{VRF}(t_k) &= \underline{\underline{R}}_{VRFtoLRF}(\gamma_k, \beta_k, \alpha_k)^T \underline{\underline{P}}^{LRF}(t_k) \\ \underline{\underline{P}}^{VRF}(t_{k+1}) &= \underline{\underline{R}}_{VRFtoLRF}(\gamma_{k+1}, \beta_{k+1}, \alpha_{k+1})^T \underline{\underline{P}}^{LRF}(t_{k+1}) \end{aligned} \quad (1)$$

where $\underline{\underline{R}}_{VRFtoLRF}$ is the rotation matrix representing the attitude of LRF with respect to VRF. Secondly, a customized version of the ICP algorithm is applied to find the best estimate of the rotation and translation necessary to align $\underline{\underline{P}}^{VRF}(t_{k+1})$, i.e. the current scan, to $\underline{\underline{P}}^{VRF}(t_k)$, i.e. the reference scan. Specifically, the algorithm provides in output an estimate of the variation of the Euler angles ($\Delta\alpha$, $\Delta\beta$, $\Delta\gamma$) and of the position vector ($\underline{\Delta T}$) between the two VRFs, occurred during the time interval from $t_k$ to $t_{k+1}$.

This ICP algorithm is characterized by a sequence of steps, i.e. initialization, matching, outliers rejection, selection and minimization of an error metric function, which are iteratively repeated until a convergence criterion is met. Each time a new scan is available, the ICP algorithm is initialized by setting to zero all the previously defined parameters, i.e. $\Delta\alpha$, $\Delta\beta$, $\Delta\gamma$ and $\underline{\Delta T}$. With regards to the matching step, which mainly determines the algorithm's computational load and the accuracy level, the classical Nearest Neighbor (NN) approach (i.e. each current scan point is associated to the closest one in the reference scan according to the Euclidean metric) is adopted. In addition, the reference scan is pre-processed to build a K-D tree [24], in order to accelerate the NN search. The outliers rejection step is introduced to compensate for wrong point associations that may arise if there is a poor overlap between the scenes observed in two LIDAR acquisitions (i.e. a large number of measurements in the actual scan do not have real correspondences in the reference scan), since they can lead to significant error in the ICP pose solution. Hence, if $\underline{d}$ is the set of distances between correspondent points, the reference/actual scan matches characterized by a relative distance which is outside the interval defined by the mean and mode values of $\underline{d}$ are considered as outliers. As regards the error metric function ($f$), it is selected as the mean squared distance of corresponding points between the two scans, see (2), and it is minimized thanks to a closed form solution based on quaternion representation [25].

$$f(q) = \frac{1}{N(t_{k+1})} \sum_{i=1}^{N(t_{k+1})} \left| \underline{P}_i^{VRF}(t_k) - \underline{\underline{R}}_{(VRF(t_k)toVRF(t_{k+1}))}(q)^T \left( \underline{P}_i^{VRF}(t_{k+1}) + \underline{\Delta T} \right) \right|^2 \quad (2)$$

In (2), $\underline{P}_i^{VRF}(t_{k+1})$ and $\underline{P}_i^{VRF}(t_k)$ are respectively the i$^{th}$ point of the actual scan and the corresponding one in the reference scan, $N(t_{k+1})$ is the number of points in the current scan, $\underline{\underline{R}}_{(VRF(tk)toVRF(tk+1))}$ and $q$ are respectively the rotation matrix representing the attitude variation of the VRF between the time instants of the two scans and the corresponding quaternion. Once $q$ is estimated, the corresponding values of $\Delta\alpha$, $\Delta\beta$, and $\Delta\gamma$ can be extracted, and, finally, $\underline{\Delta T}$ is computed as the difference between the centroids of the two scans, as given by (3).

$$\underline{\Delta T} = \underline{\underline{R}}_{(VRF(t_k)toVRF(t_{k+1}))}(q)\, Mean\{\underline{\underline{P}}^{VRF}(t_k)\} - Mean\{\underline{\underline{P}}^{VRF}(t_{k+1})\} \quad (3)$$

At this point, the value of $f$ can be updated and the procedure is repeated until the variation between two successive iterations goes below a threshold (e.g. $10^{-6}$ m$^2$).

It is now fundamental to point out that, since by definition the two VRFs have the same orientation and a different position (unless the MAV is not moving), the estimated values of $\Delta\alpha$, $\Delta\beta$, and $\Delta\gamma$ are not used. On the other hand, the output of (3) can be used to update the MAV's position vector using (4).

$$\underline{T}_{t_{k+1}} = \underline{T}_{t_k} + \underline{\Delta T} \quad (4)$$

The logic behind the proposed L/I-O algorithm is summarized by the flow diagram shown in Fig. 2.

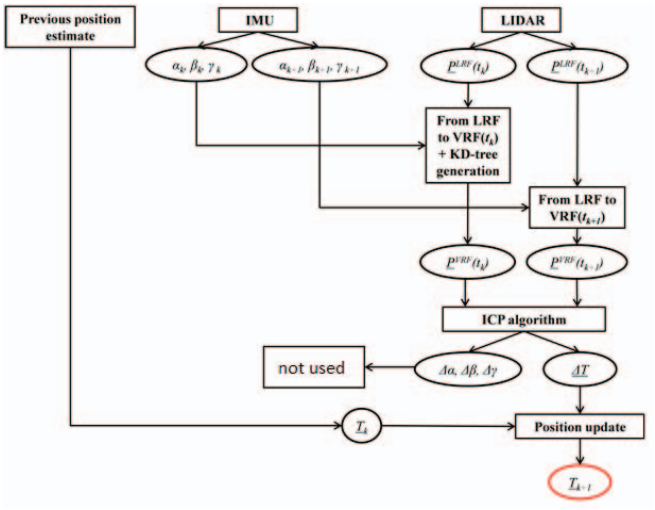


Fig. 2. Flow diagram describing the proposed L/I-O algorithm for localization. The red circle contains the final output.

An additional peculiar feature of the presented ICP algorithm is its strategy for autonomous failure detection. Specifically, since the value at convergence of the ICP cost function ($f_{CONV}$) is a direct measure of the algorithm's accuracy level [26], it is possible to determine whether the algorithm has failed or not by comparing it to a given threshold ($f_{LIM}$). In case $f_{CONV}$ is larger than $f_{LIM}$ the MAV's position is updated by integrating the inertial acceleration provided by the IMU, according to (5)

$$\underline{T}_{t_{k+1}} = \underline{T}_{t_k} + \underline{v}_{t_k-1}(t_{k+1} - t_k) + \frac{1}{2}\underline{a}_{t_k}(t_{k+1} - t_k)^2 \quad (5)$$

where $\underline{v}$ and $\underline{a}$ represent velocity and acceleration of the MAV, respectively. These latter quantities are converted in ENU thanks to the attitude estimates.

### B. Mapping: line detection (PCA vs. LS fitting method)

The feature-based mapping technique presented in this paper exploits lines identified in the observed environment by means of a line detection algorithm which processes the 3D data provided by the 2D laser scanner. The algorithm's operation can be divided into three steps: clusterization, line identification, and line storage.

The first step is the clusterization process and is characterized by two hierarchical levels. Firstly, the scan is subdivided into separate clusters by looking for those locations, indicated as break-points, at which the inter-point distance (i.e. the distance between consecutive points in the scan) is larger than a specific threshold (*DTh*). This part of the algorithm is similar to the clusterization process proposed in [27], where the radial distance between consecutive points is compared to a threshold. However, the proposed approach foresees a second level of clusterization, meaning that additional break-points are identified within each cluster, by exploiting the polar structure of the point cloud acquired by the 2D laser scanner. Basically, both the $x$ and $y$ coordinates of the measured points can be considered function of the scan angle $\theta$. Hence, the sub-cluster search aims at looking for those locations $i$ at which either (6) or (7) is satisfied. These two conditions, where $N$ is the number of points in the cluster, compare the local value of the $x$ and $y$ derivatives with respect to $\theta$, to the sum of their mean and standard deviation computed over the cluster.

$$\left\{\frac{dx(\theta_i)}{d\theta} > \left[\frac{1}{N}\sum_{i=1}^{N}\frac{dx(\theta_i)}{d\theta} + \frac{1}{N-1}\sum_{i=1}^{N}\left(\frac{dx(\theta_i)}{d\theta} - \frac{1}{N}\sum_{i=1}^{N}\frac{dx(\theta_i)}{d\theta}\right)^2\right]\right\} \quad (6)$$

$$\left\{\frac{dy(\theta_i)}{d\theta} > \left[\frac{1}{N}\sum_{i=1}^{N}\frac{dy(\theta_i)}{d\theta} + \frac{1}{N-1}\sum_{i=1}^{N}\left(\frac{dy(\theta_i)}{d\theta} - \frac{1}{N}\sum_{i=1}^{N}\frac{dy(\theta_i)}{d\theta}\right)^2\right]\right\} \quad (7)$$

After clusterization, the identification step can start. Unlike the techniques listed in [27], this approach relies on the PCA, which allows finding the principal directions of a multidimensional dataset by analyzing the eigenvectors of its covariance matrix [19]. In this case, the PCA is applied by assigning to each cluster the ratio between the maximum and minimum eigenvalues ($r$) associated with its covariance matrix. Hence, if $r$ is larger than a fixed threshold (*ETh*), a cluster is considered as line feature and its direction is given by the eigenvector corresponding to the maximum eigenvalue (whose components are $\lambda_x$, $\lambda_y$ and $\lambda_z$). Specifically, this line passes through the centroid of the cluster and it is oriented as the eigenvector corresponding to the maximum eigenvalue of the covariance matrix. Finally, the line detection algorithm foresees a storage step aimed at assigning to each detected line a list of 8 parameters to be stored in memory. Firstly, $\alpha_L$ is the angle associated to the line direction, according to (8).

$$\alpha_L = \tan^{-1}\left(\frac{\lambda_y}{\lambda_x}\right) \quad (8)$$

Secondly, $x_C$ and $y_C$ are the coordinates of the centroid of the cluster, while $x_{E1}$ and $y_{E1}$, as well as $x_{E2}$ and $y_{E2}$, are the coordinates of the two ends of the line segments. They are found by projecting the first and the last elements of the cluster on the edge direction according to (9),

$$\begin{bmatrix} x_{E1} \\ y_{E1} \end{bmatrix} = \begin{bmatrix} \lambda_x \\ \lambda_y \end{bmatrix}\rho_{E1} + \begin{bmatrix} x_C \\ y_C \end{bmatrix}$$
$$\begin{bmatrix} x_{E2} \\ y_{E2} \end{bmatrix} = \begin{bmatrix} \lambda_x \\ \lambda_y \end{bmatrix}\rho_{E2} + \begin{bmatrix} x_C \\ y_C \end{bmatrix} \quad (9)$$

where $\rho_{E1}$ and $\rho_{E2}$, i.e. the distances of the two ends from the line segment centroid, can be computed using (10).

$$\rho_{E1} = \sqrt{(x_{E1} - x_C)^2 + (y_{E1} - y_C)^2}\left(\frac{\begin{bmatrix} x_{E1} \\ y_{E1} \end{bmatrix} - \begin{bmatrix} x_C \\ y_C \end{bmatrix}}{\sqrt{(x_{E1} - x_C)^2 + (y_{E1} - y_C)^2}} \cdot \begin{bmatrix} \lambda_x \\ \lambda_y \end{bmatrix}\right)$$
$$\rho_{E2} = \sqrt{(x_{E2} - x_C)^2 + (y_{E2} - y_C)^2}\left(\frac{\begin{bmatrix} x_{E2} \\ y_{E2} \end{bmatrix} - \begin{bmatrix} x_C \\ y_C \end{bmatrix}}{\sqrt{(x_{E2} - x_C)^2 + (y_{E2} - y_C)^2}} \cdot \begin{bmatrix} \lambda_x \\ \lambda_y \end{bmatrix}\right) \quad (10)$$

Finally, $d_L$ is the distance of the line segment from the origin of the reference frame in which it is represented, and it can be computed using (11).

$$d_L = \frac{-y_C + \tan(\alpha_L)x_C}{\sqrt{1+\tan^2(\alpha_L)}} \tag{11}$$

Since the sub-cluster search could generate break-points within a real edge (due to the LIDAR measurement noise), an intermediate merging step is implemented. It allows merging two consecutive clusters if the corresponding lines satisfy the condition defined by (12), where $\alpha_{CC}$ is the orientation of the direction of the segment which links the two centroids and $\alpha_T$ is a very small angular threshold (e.g. 0.05°).

$$\{(\alpha_{L1}-\alpha_{L2})<\alpha_T\}\cap\{[((\alpha_{L1}-\alpha_{CC})<\alpha_T)\cap((\alpha_{L2}-\alpha_{CC})<\alpha_T)]\cup[((\alpha_{L1}-(180-\alpha_{CC}))<\alpha_T)\cap((\alpha_{L2}-(180-\alpha_{CC}))<\alpha_T)]\} \tag{12}$$

This means that the line detection algorithm is applied to the aggregate cluster.

The PCA method, adopted to perform line fitting, is compared to the classical Least Squares (LS) approach, by means of numerical simulations. Firstly, a set of $n$ points, randomly distributed along a direction identified by a fixed angular coefficient ($m_l$), and whose coordinates ($x^i$ and $y^i$) are given by (13), is defined in the 2D space.

$$\begin{aligned} x^i &= x_S i \\ y^i &= m_l x^i + \nu_\sigma{}^i \end{aligned}, i = 1\ldots n \tag{13}$$

In (13), $x_S$ is the fixed step between points along the x-axis, and $\nu_\sigma{}^i$ is the i$^{th}$ extraction from a normal distribution with zero mean and standard deviation equal to $\sigma$. Secondly, both the PCA and LS methods are applied obtaining the corresponding lines, each one identified by an angular coefficient ($m_{PCA}$ and $m_{LS}$), and a constant term ($n_{PCA}$ and $n_{LS}$). Finally, the line fitting accuracy is evaluated as the mean squared distance of the assigned points from the estimated line ($Err_{PCA}$ and $Err_{LS}$), according to (14).

$$\begin{aligned} Err_{PCA} &= \frac{1}{n}\sum_{i=1}^{n}(y^i - m_{PCA}x^i - n_{PCA})^2 \\ Err_{LS} &= \frac{1}{n}\sum_{i=1}^{n}(y^i - m_{LS}x^i - n_{LS})^2 \end{aligned} \tag{14}$$

Results are averaged on 10000 Monte-Carlo simulations and are shown in Table I, in terms of line fitting accuracy and computational load. With regards to the simulation inputs, $m_l$ is set to 5, $x_S$ is set to 4 cm, and $\nu_\sigma$ is equal to 5 cm.

TABLE I. COMPARISON BETWEEN PCA AND LS FOR LINE FITTING

| *N* | PCA time saving (%) | $Err_{PCA}$ - rms ($m^2$) | $Err_{LS}$ - rms ($m^2$) | $Err_{PCA}$ - $Err_{LS}$ (%) |
|---|---|---|---|---|
| 25 | 83 | $2.4\cdot10^{-3}$ | $2.4\cdot10^{-3}$ | $1.2\cdot10^{-1}$ |
| 50 | 83 | $2.4\cdot10^{-3}$ | $2.4\cdot10^{-3}$ | $2.9\cdot10^{-2}$ |
| 100 | 83 | $2.5\cdot10^{-3}$ | $2.5\cdot10^{-3}$ | $7.1\cdot10^{-3}$ |
| 250 | 83 | $2.5\cdot10^{-3}$ | $2.5\cdot10^{-3}$ | $1.1\cdot10^{-3}$ |
| 500 | 82 | $2.5\cdot10^{-3}$ | $2.5\cdot10^{-3}$ | $3\cdot10^{-4}$ |
| 1000 | 80 | $2.5\cdot10^{-3}$ | $2.5\cdot10^{-3}$ | $1\cdot10^{-4}$ |
| 10000 | 57 | $2.5\cdot10^{-3}$ | $2.5\cdot10^{-3}$ | $<10^{-4}$ |

This analysis proves that the PCA-based line fitting is almost twice faster than the LS approach while being able to provide the same level of accuracy.

### C. Feature-based map generation

The feature-based mapping algorithm is exploited to obtain a synthetic representation of the environment in which the MAV is moving, thus limiting the amount of data storage compared to occupancy grid methods. To this aim, the first step is to apply the updated position solution to translate the measured point cloud from VRF to ENU. Then, the line detection algorithm, described previously in sub-section II-B of this paper, is applied to find robust features which become candidates to be added to the updating map. Specifically, each candidate has to be compared to every line in the map in terms of two parameters, i.e. $\alpha_L$ and $d_L$. If a correspondence is not found, the candidate line becomes a new element of the map. On the other hand, if the candidate represents a visualization of the same feature of the environment from a different position, the two lines must be merged. This is done by projecting the two ends of the candidate line on the direction of the corresponding one in the map, and by comparing these projections to the pre-existing ends. The updated ends are the ones that determine the maximum length of the updated line in the map.

## III. EXPERIMENTAL TESTS

In order to assess the performance of the algorithms developed for localization and mapping, an experimental setup and a test area are prepared for data recording and testing. The setup (see Fig. 3-a) is composed of the following items: one LIDAR - UTM-30LX-EW, produced by Hokuyo whose specifications can be found in [28]; one autopilot - Pixhawk, produced by 3drobotics; one embedded board - Nitrogen6X, produced by Boundary-Devices; one battery; two voltage regulators, LM2596 Adjustable DC/DC Power Converter. The latter components are needed since the battery has to power both the LIDAR (at 12 V) and the Nitrogen board (5 V). The test area (see Fig. 3-b) is a 2D maze in which the experimental setup is carried by hand.

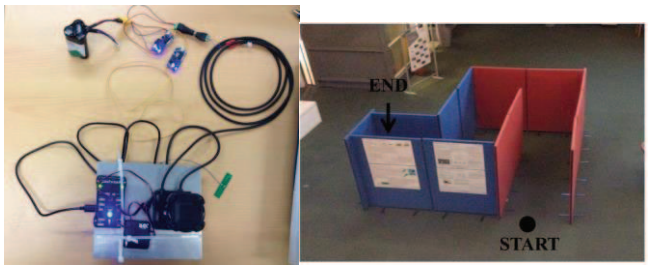


Fig. 3. Experimental setup on the left (a). Test area on the right, with indication of the start and ending points of the travelled path (b).

The Nitrogen board (not in sight in Fig. 3-a, as it is located below the metallic plate on which the LIDAR and the Pixhawk are mounted) is used to register data from both the Pixhawk (using USB connection) and the LIDAR (using Ethernet connection) by exploiting the corresponding nodes of the Robot Operating System (ROS) [29], i.e. the *mavros* and the *urg_node,* respectively. In this way the IMU data from the Pixhawk and the range data from the LIDAR can be simultaneously recorded, together with their timestamps, within the same bag-file. This makes it possible to run the proposed algorithms offline in MATLAB environment by directly reading from the generated bag-files. Since IMU data are collected at higher update rate (about 90 Hz) than the LIDAR data (about 35 Hz), the attitude parameters corresponding to the LIDAR timestamps are obtained through linear interpolation. In order to analyze the algorithms' accuracy level in terms of trajectory estimation, a ground truth is required. To this aim, once the indoor scenario is selected, a reference trajectory to be followed is defined, whose length (approximately 9 m) is computed by taking measurements from a single point Laser Range Finder (BOSCH DLR130 Distance Measurer).

Several runs of the L/I-O algorithm are realized to evaluate the effect on performance of its tuning parameters, which are defined hereinafter. The Range Limit (RL) is the value of distance over which LIDAR measurements are disregarded from the acquired scan. The Angular Resolution (AR) is the angle between two consecutive LIDAR measurements which are not deleted from the acquired scan (minimum value for AR is 0.25°). The Odometry rate (OR) is the frequency at which the L/I-O algorithm is applied (maximum value for OR is 36 Hz which is the LIDAR measurement rate). On the other hand, other parameters of the algorithm are kept constant and are listed hereinafter. The ICP maximum number of iteration is set to 30. The minimum value for the time derivative of the ICP cost function at convergence is set to $10^{-6}$ $m^2$. The value of $f_{LIM}$ is set to 0.5 $m^2$. Firstly, the effect of RL is evaluated by considering different values (from 60 m to 3 m) while keeping the AR (0.25°) and the OR (6 Hz) fixed. Results in Table II show that the RL should always be set below 30 m. This is not highlighted by the error on the estimated length of the overall trajectory ($L_{EST}$), which is almost the same for any value of RL, but by looking at the sum of $f_{CONV}$ during the test ($f_{SUM}$) and at its mean ($f_{MEAN}$). Indeed, these parameters represent an index of how well two consecutive scans are aligned by the algorithm, meaning that the lower their value is, the larger the accuracy of localization becomes. When RL is 60 m, $f_{SUM}$ and $f_{MEAN}$ reach the values of 8.8 $m^2$ and 0.149 $m^2$, respectively, which are one order of magnitude larger than for the other runs. This degradation in performance is explained by the fact that LIDAR measurements longer than 30 m are not reliable and can cause wrong point-to-point matching by the ICP thus compromising its operation. Indeed, when the laser intensity reflected back at the detector is below an internal threshold of the sensor (this happens, for instance, when the laser shot passes through a window), a value of range around 60 m is the output. Below 30 m, a reduction of RL gives advantages in terms of computational load (less number of points to be matched by the ICP routine) and localization accuracy. However, if RL is too low (3 m), it may produce an increase in the error in $L_{EST}$, which shows that it is convenient to neglect part of the scan to reduce the computing time, provided that it does not cause excessive loss of information. It is finally worth mentioning that the attained very low values of $f_{SUM}$ and $f_{MEAN}$ do not disprove the previous statements since they are caused by the significant reduction in the amount of points analyzed in the scan.

TABLE II. L/I-O ALGORITHM PERFORMANCE ANALYSIS. EFFECT OF RL

| ***RL* (m)** | **ICP failure *(%)*** | **$f_{SUM}$ ($m^2$)** | **$f_{MEAN}$ ($m^2$)** | **Mean comp. time (s)** | **$L_{EST}$ (m)** | **Error on $L_{EST}$ (%)** |
|---|---|---|---|---|---|---|
| 60 | 10 | 8.799 | 0.149 | 0.134 | 8.16 | **9** |
| 30 | 7 | 1.202 | 0.020 | 0.136 | 8.20 | **9** |
| 15 | 5 | 0.973 | 0.017 | 0.149 | 8.30 | **8** |
| 11 | 5 | 0.984 | 0.017 | 0.130 | 8.44 | **6** |
| 7 | 2 | 0.282 | 0.005 | 0.160 | 8.50 | **6** |
| 3 | 0 | 0.171 | 0.003 | 0.159 | 8.02 | **11** |

Secondly, the effect of AR is analyzed considering three values (0.25°, 0.5° e 1°) while keeping fixed the RL (11 m) and the OR (6 Hz). Results in Table III show that it is convenient to change AR from 0.25° to 0.5° since it causes a faster ICP solution (about 30 %) while ensuring the same accuracy level. It is also possible to state that a further reduction of the resolution (AR set to 1°) is not advisable, not only because it generates a performance worsening but also because it compromises the applicability of the line extraction algorithm for mapping (the analyzed scans become too sparse to obtain robust line features).

TABLE III. L/I-O ALGORITHM PERFORMANCE ANALYSIS. EFFECT OF AR

| ***AR* (°)** | **ICP failure *(%)*** | **$f_{SUM}$ ($m^2$)** | **$f_{MEAN}$ ($m^2$)** | **Mean comp. time (s)** | **$L_{EST}$ (m)** | **Error on $L_{EST}$ (%)** |
|---|---|---|---|---|---|---|
| 0.25 | 5 | 0.984 | 0.017 | 0.130 | 8.44 | **6** |
| 0.5 | 3 | 0.947 | 0.016 | 0.094 | 8.48 | **6** |
| 1 | 7 | 2.603 | 0.044 | 0.063 | 8.40 | **7** |

Thirdly, the effect of OR is analyzed considering four values (36 Hz, 18 Hz, 6 Hz, 3 Hz) while keeping fixed the RL (11 m) and the AR (0.5°). Results in Table IV show that low values of OR (3 Hz and 6 Hz) provide better performance than by applying the localization algorithm at larger frequencies (18 Hz and 36 Hz). This is because the lower rate of execution reduces the propagation of the error, which is bonded to the concept of odometry. Since pose variation is computed by comparing two successive datasets without considering the history of the trajectory, there is no way to correct any mistake committed during the application of the algorithm.

TABLE IV. L/I-O ALGORITHM PERFORMANCE ANALYSIS. EFFECT OF OR

| ***OR* (Hz)** | **ICP failure *(%)*** | **$f_{SUM}$ ($m^2$)** | **$f_{MEAN}$ ($m^2$)** | **Mean comp. time (s)** | **$L_{EST}$ (m)** | **Error on $L_{EST}$ (%)** |
|---|---|---|---|---|---|---|
| 36 | 1 | 2.751 | 0.008 | 0.059 | 7.21 | **20** |
| 18 | 1 | 1.682 | 0.010 | 0.077 | 7.93 | **12** |
| 6 | 3 | 0.947 | 0.016 | 0.094 | 8.48 | **6** |
| 3 | 13 | 1.357 | 0.047 | 0.118 | 8.49 | **6** |

This same principle can be used to understand why the accuracy of the algorithm proposed for localization is also affected by the velocity at which the experimental system is moved along the same path. Table V contains the results obtained for the L/I-O algorithm with the same tuning parameters (RL set to 11 m, AR set to 0.5° and OR set to 6 Hz) applied to two different datasets, respectively, which were recorded by moving along the assigned trajectory within the test area but with different velocities.

TABLE V. L/I-O ALGORITHM PERFORMANCE ANALYSIS. EFFECT OF MOTION VELOCITY

| **Time length (s)** | **LIDAR / IMU measurements** | $f_{SUM}$ **(m²)** | $f_{MEAN}$ **(m²)** | **Mean comp. time (s)** | $L_{EST}$ **(m)** | **Error on $L_{EST}$ (%)** |
|---|---|---|---|---|---|---|
| 26.36 | 961 / 2441 | 1.180 | 0.007 | 0.013 | 7.79 | **13** |
| 9.72 | 355 / 903 | 0.947 | 0.016 | 0.018 | 8.48 | **6** |

It is clear that, moving faster within the test area limits the error propagation of the odometry approach.

An example of application of the proposed localization and mapping algorithms is shown in Fig. 4. It is worth outlining that mapping can be carried out at different rate with respect to localization, e.g. in this case the mapping rate is 3 Hz while the localization one is 6 Hz.

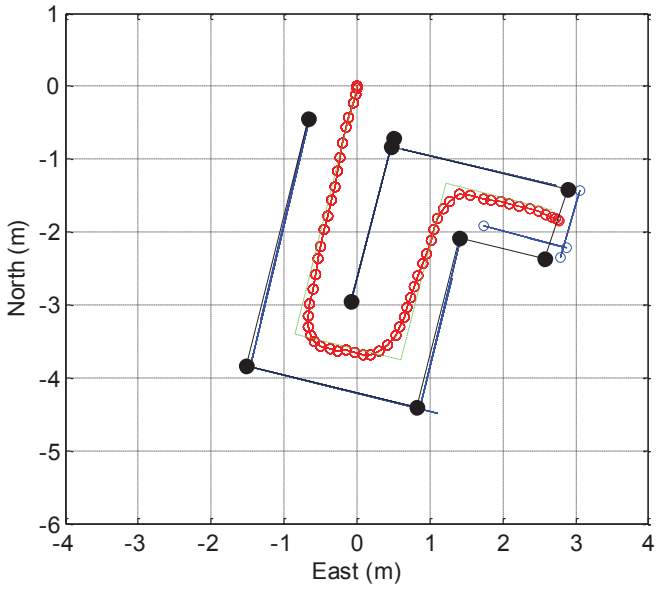


Fig. 4. Example of application of localization and mapping algorithm to the analyzed test case: reference trajectory (dashed-dot green); estimated trajectory red; vertices of the real map (black dots);real map (black lines);estimated map (blue lines).

First of all, it is important to outline that the reference trajectory, depicted in green in Fig. 4, is an approximation of the real travelled trajectory and it is used to determine the error in $L_{EST}$. From the figure, it is possible to state that the line-based mapping technique is able to provide a sparse but accurate representation of the travelled environment. Specifically, all the edges of the real map are accurately extracted in terms of length, location and inclination in the ENU. However, an exception is given by the two lines identified by blue circles at their ends, since they have the same inclination as the corresponding real edges but are displaced of some centimeters from the real lines. This can be justified as a consequence of the error propagated in the estimated trajectory along the travelled path. Future work will be aimed at solving this issue by improving the accuracy of the proposed localization and mapping algorithms. This can be done by exploiting the information given by the map itself in real-time to correct the solution given by the odometry algorithm.

## IV. CONCLUSION

This paper addressed the problem of localization and mapping for micro Unmanned Aerial Vehicles flying in two-dimensional environments (e.g. office-like). Solutions to these problems were obtained by aiding inertial sensors with active EO systems, which globally ensure higher degree of autonomy when compared to passive sensors. Specifically, the presented approaches were based on the integration of inertial data provided by an Inertial Measurement Unit and three-dimensional data acquired by means of a two-dimensional laser scanner (LIDAR). The proposed localization technique, namely LIDAR/Inertial-Odometry algorithm, was used to propagate the vehicle's attitude by relying only on inertial data, while the position was recursively updated by exploiting a customized scan matching approach, based on the iterative Closest Point Algorithm, which integrated inertial and LIDAR data available at different update rates. An additional innovative aspect was represented by the criterion introduced for autonomous failure detection of the scan matching approach. With regards to the mapping issue, an accurate but sparse representation of the observed environment was obtained by introducing an innovative method to quickly and robustly extract lines from LIDAR scans based on the Principal Component Analysis. Indeed, it showed capability to perform line fitting about 75% faster than classical least squares approaches while ensuring the same accuracy level. An experimental setup (including a LIDAR, an Inertial Measurement Unit, and a processing unit) and a test area (two-dimensional maze) were conceived and prepared to assess the performance of the proposed approaches, by means of off-line processing of the acquired data. The effect on performance of localization algorithm parameter tuning was analyzed, and it allowed finding the most convenient parameter settings. Results show the effectiveness of the proposed approach for localization and mapping, with observed errors of order of centimeters. . Performance can be improved by feeding back the mapping information to the localization algorithm, since this can reduce the error in trajectory estimation. This will be the objective of further investigations.

## ACKNOWLEDGMENT

This research was carried out in the frame of Program STAR - Linea 2 -, financially supported by UniNA and Compagnia di San Paolo.

## REFERENCES


[1] R. L. Greenspan, "GPS and inertial integration," Global Positioning System: Theory and applications. 2, pp. 187-220, 1996.

[2] J. Farrell, "Aided navigation: GPS with high rate sensors," McGraw-Hill, Inc., 2008.

[3] K. Jong-Hyuk, S. Sukkarieh, and S. Wishart, "Real-time navigation, guidance, and control of a UAV using low-cost sensors," Field and Service Robotics, Springer Berlin Heidelberg, 2006.

[4] J. Wendel, O. Meister, C. Schlaile, and G. F. Trommer, "An integrated GPS/MEMS-IMU navigation system for an autonomous helicopter," Aerospace Science and Technology, Vol. 10 (6), pp. 527-533, 2006.

[5] F. Kendoul, "Survey of advances in guidance, navigation, and control of unmanned rotorcraft systems," Journal of Field Robotics, Vol. 29 (2), pp. 315-378, 2012.

[6] K. Khoshelham, and S. O. Elberink, "Accuracy and resolution of kinect depth data for indoor mapping applications," Sensors, Vol. 12 (2), pp. 1437-1454, 2012

[7] A. F. Scannapieco, A. Renga, and A. Moccia, "Preliminary Study of a Millimeter Wave FMCW InSAR for UAS Indoor Navigation", Sensors, Vol. 15 (2), pp. 2309-2335, 2015.

[8] H. Durrant-Whyte, and T. Bailey, "Simultaneous localization and mapping: part I," IEEE Robotics & Automation Magazine, Vol. 13 (2), pp. 99-110, 2006.

[9] T. Bailey, and H. Durrant-Whyte, "Simultaneous localization and mapping (SLAM): Part II," IEEE Robotics & Automation Magazine, Vol. 13 (3), pp. 108-117, 2006.

[10] A. Bachrach, R. He, and N. Roy, "Autonomous flight in unknown indoor environments," International Journal of Micro Air Vehicles, Vol. 1 (4), pp. 217-228, 2009.

[11] Z. Lu, Z. Hu, and K. Uchimura, "SLAM estimation in dynamic outdoor environments: A review," Intelligent Robotics and Applications, Springer Berlin Heidelberg, pp. 255-267,2009.

[12] P. J. Besl, and N. D. McKay, "Method for registration of 3-D shapes," Proc. SPIE Sens. Fusion IV: Control Paradig. Data Struct. 1611, pp. 586–606, 1992.

[13] S. Winkvist, "Low computational SLAM for an autonomous indoor aerial inspection vehicle," Doctoral dissertation, University of Warwick, 2013.

[14] D. Droeschel, J. Stuckler, and S. Behnke, "Local multi-resolution representation for 6D motion estimation and mapping with a continuously rotating 3D laser scanner," IEEE International Conference on Robotics and Automation (ICRA), pp. 5221-5226, 2014

[15] J. Zhang, and S. Singh, "LOAM: Lidar odometry and mapping in real-time," Robotics: Science and Systems Conference (RSS), 2014.

[16] S. Kohlbrecher, O. Von Stryk, J. Meyer, and U. Klingauf, "A flexible and scalable slam system with full 3d motion estimation," IEEE International Symposium on Safety, Security, and Rescue Robotics (SSRR), pp. 155-160, 2011.

[17] L. Zhang, and B. K. Ghosh, "Line segment based map building and localization using 2D laser rangefinder," Proceedings of the IEEE International Conference on Robotics and Automation ICRA'00, Vol. 3, pp. 2538-2543, 2000.

[18] P. de la Puente, D. Rodriguez-Losada, A. Valero, and F. Matia, "3D feature based mapping towards mobile robots' enhanced performance in rescue missions," Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pp. 1138-1143. 2009.

[19] S. Wold, K. Esbensen, and P. Geladi, "Principal component analysis," Chemometrics and intelligent laboratory systems, Vol. 2 (1), pp. 37-52, 1987.

[20] K. O. Arras, and R. Siegwart, "Feature Extraction and Scene Interpretation for Map-Based Navigation and Map Building," Proceedings of the Symposium on Intelligent Systems and Advanced Manufacturing, 1997.

[21] C. Berger, "Toward rich geometric map for SLAM: Online Detection of Planes in 2D LIDAR," Journal of Automation Mobile Robotics and Intelligent Systems, 2007.

[22] D. Scaramuzza, and F. Fraundorfer, "Visual odometry [tutorial] ," IEEE Robotics & Automation Magazine, Vol. 18 (4), pp. 80-92, 2011.

[23] A. Censi, L. Iocchi, and G. Grisetti, "Scan matching in the Hough domain," Proceedings of the 2005 IEEE International Conference on Robotics and Automation (ICRA'05), pp. 2739-2744, 2005.

[24] J. K. Friedman, J. Bentely, and R. A. Finke, "An Algorithm for Finding Best Matches in Logarithmic Expected Time", ACM Transactions on Mathematical Software, Vol. 3 (3), pp. 209-226, 1977.

[25] B. K. Horn, "Closed-form solution of absolute orientation using unit quaternions," Journal of the Optical Society of America, Vol. 4 (4), pp. 629-642, 1987.

[26] R. Opromolla, G. Fasano, G. Rufino, and M. Grassi, "Uncooperative pose estimation with a LIDAR-based system," Acta Astronautica, vol. 110, pp.287-297, May-June 2015.

[27] V. Nguyen, A. Martinelli, N. Tomatis, and R. Siegwart, "A comparison of line extraction algorithms using 2D laser rangefinder for indoor mobile robotics," IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2005), pp. 1929-1934, 2005.

[28] https://www.hokuyo-aut.jp/02sensor/07scanner/download/products/utm-30lx-ew/, accessed on 07/01/2016.

[29] M. Quigley, K. Conley, B. P. Gerkey, J. Faust, T. Foote, J. Leibs, R. Wheeler, and A. Y. Ng, "ROS: an open-source Robot Operating System," ICRA Workshop on Open Source Software, 2009.