AN INCREMENTAL SLAM ALGORITHM FOR INDOOR AUTONOMOUS NAVIGATION

M. Di Castro, A. Masi, G. Lunghi, R. Losito

CERN, Switzerland

***Abstract* – Robots indoor autonomous navigation is a crucial requirement for many challenging applications like the inspection of unstructured and harsh environments. Simultaneous Localization and Mapping (SLAM) is a key component to navigate autonomously. A novel embedded incremental algorithm that could be used on ground or aerial robots to localize and create 3D maps of an unknown GPS-denied environment, is presented. The algorithm is processing-wise light, is executed on board in real time and is suitable to be integrated in a full framework for autonomous navigation. It has been validated using a minimal set of sensors composed of a laser scanner and an inertial unit. Preliminary test results of the algorithm are presented using the CERN test facilities as case study.**

I. INTRODUCTION

Many robotic applications require the ability to acquire in real-time a 3D model of the environment in order to navigate safely in it and accomplish complex tasks. Simultaneous Localization and Mapping (SLAM) algorithms incrementally build map of the environment and estimate the pose of the robot at the same time [1].

In the absence of global positioning information like in GPS-denied environment, the robot's state (the robot's pose and map) is essential to obtain a safe and reliable autonomous navigation. Obtaining large and accurate maps is still an open problem, especially when performed in real-time.

There are different approaches to create SLAM algorithms, but basically all the algorithms can be divided in three main categories: landmark-based SLAM, grid-based SLAM and pose-graph SLAM.

Landmark-based SLAM identifies landmarks in the environment and, considering the relative position of the landmarks, computes the state of the robot. Landmark-based SLAM requires some previous information about the environment, since the robot has to recognize a landmark and associate it with the previously observed landmarks. This operation is called data association and is highly dependent on the environment [2][3].

Grid-based SLAM uses an occupancy grid map to represent the environment. With this approach, no assumptions about the environment are required [4][5].

Pose-graph based SLAM uses a graph to represent a problem. Every node in the graph corresponds to a pose of the robot during mapping. Every edge corresponds to the spatial constraints between them. [6]

For each type of the described SLAM algorithms, different approaches have been proposed.

One of these specific approaches is the EKF SLAM that uses an extended Kalman filter to estimate the state of the robot. Usually, this class of algorithms are landmark based. Although EKF is optimal for linear Gaussian noise, the SLAM problem is not a linear problem, so some linearization must be performed. If the non-linearities are large, EKF SLAM can diverge.

Particle Filter SLAM solves the Gaussian limitation, using a particle filter to represent the posterior, so no likelihood parameterization is required and any kind of distribution can be represented[7][8][9].

In this paper an incremental approach for a SLAM algorithm based on particle filters is proposed. Because a parametric form for the likelihood distribution of the robot's pose and the map can't be defined, in the proposed method the space around the last odometry measurement is sampled trying to find the most likely pose. Instead of keeping a map for every particle and computing the probability of each particle using the sensor model [10], only one particle is kept and the probability of this particle to give the most correct map at each step is maximized. Using this approach, only one map at each step can be carried on reducing the computational time and allowing the use of more particles around the odometry point measured increasing the map accuracy with respect to standard particle filter approaches. If the time between two steps is short, the inconsistency of the map is kept low.

SLAM algorithm compute data coming from various sensors needed for the autonomous navigation. Different types of sensor have been used for SLAM in the last decade.

3D time of flight sensors are usually heavy, expensive and have high power consumption.

Stereo cameras and monocular cameras are used for visual odometry and landmark extraction. Nevertheless information extraction for building an occupancy grid map is computationally heavy.

2D laser range finders are the most common choice and have been intensively used on robots for SLAM so far. Being compact, light and with an update rate in the order

ISBN-14: 978-92-990073-2-7

of tens of Hz, 2D laser range finders (lidar) are a suitable choice to be used as sensor for SLAM.
For the work presented in this paper, a lidar and an inertial measurement unit (IMU) have been used as positioning sensor.
The presented approach uses a 2D laser range finder and an IMU to estimate the 6 DoF state of the robot and a 3D map of the environment. Unless standard particle filter SLAM, the presented approach is more suitable for real-time. Other approaches like Hector SLAM [11] provide a 6 DoF state estimation but a 2D map. The approach proposed in this paper provides a full 3D map, more suitable for autonomous navigation.
The proposed method is suitable both for UAVs and AGVs. The AGVs can provide also wheels odometry information that can replace the UAV's IMU.
The next section defines the SLAM problem and is followed by experimental results. The paper then ends with a brief discussion about the future works and the conclusions.

II. THE PROPOSAL

The SLAM problem can be defined as the maximization of the following likelihood:

$$p(x_t, m_t | \, x_{t-1}, m_{t-1}, z_t, u_t) \quad (1)$$

where $x_t$is the 6-DOF pose of the robot at time t, m_t is the occupancy grid map at time t, x_{t-1} is the previous pose of the robot, m_{t-1} the occupancy grid map built so far, z_t is the latest lidar measurement and u_t is the latest IMU measurement.

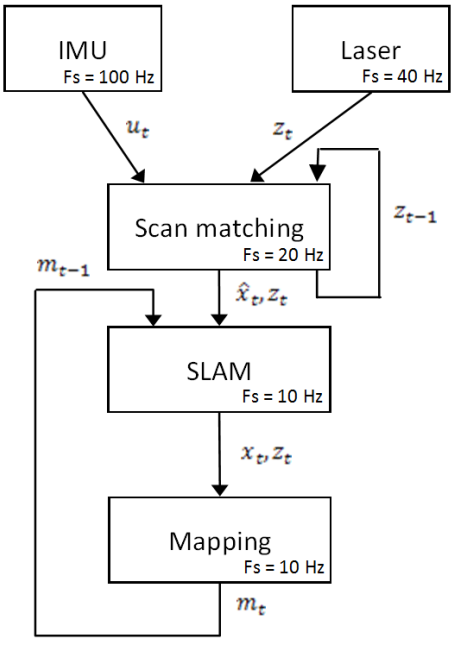


Fig 1. The software architecture

A. The development platform

The development platform is based on two sensors: an IMU and a 2D laser range finder. A full 6-DOF SLAM is implemented, making the approach completely robotic platform independent. For the experimental validation, the focus has been put on the autonomous navigation for Unmanned Aerial Vehicles (UAVs), since they have the tightest constraints on autonomy, payload and processing power.

B. The software architecture

The software architecture described in Fig.1 is highly modular. The scan matching modules takes data from the IMU and laser sensors and computes the odometry. The SLAM module represents the innovative contribution to the state of the art, characterized by the alternative approach to the standard particle filter SLAM. In this step an odometry error correction is performed, using the map built so far. The last step updates the map.

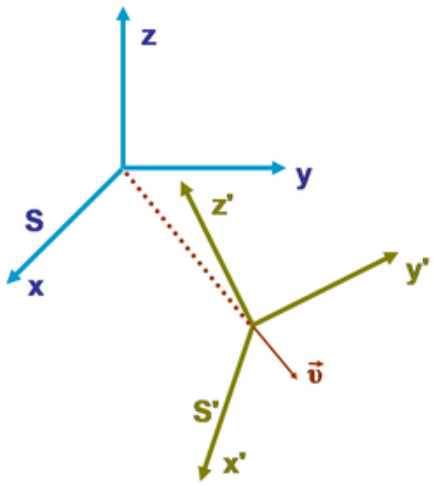


Fig 2. Global reference frame S and robot's reference frame S'

i. The IMU Module

$$u_t = [\Delta x, \Delta y, \Delta z, \propto, \beta, \Delta\gamma]^{\mathrm{T}}$$

$\Delta x, \Delta y, \Delta z$ represent the relative distances from step t-1 and t on robot's reference system S' (Fig 2.). These values are calculated with a double integration using data from IMU's accelerometers. The drift that is present due to the double integration can be neglected, since this operation is only done at initialization for the scan matching algorithm and it is not taken into account during the navigation.
$\propto, \beta$ are respectively roll and pitch angles of the robot. They are provided in the global reference

frame, considering the relative rotation from the gravitational vector.

Δγ is the variation on the yaw angle, provided from IMU's gyroscopes.

Then, we update the rotation matrix of the robot using the robot's attitude obtained from IMU.

ii. *The Laser Module*

The laser module periodically scans the environment providing z_t in (1).

$$z_t = [r_i, \theta_i] \quad i = 0, \dots, n$$

r_i is the i-th distance obtained from the range finder, and θ_i the angle orientation of the range.

In order to get the altitude of the robot, two approaches have been tested. The first one consists of using an ultrasound sensor looking downward to get the height of the robot. The ultrasound sensor has a cone shaped field of view, so it can sense the distance of the nearest obstacle under the robotic platform.

The second approach consists of using two mirrors reflecting respectively, upwards and downwards, small fractions of the field of view of the laser scanner.

iii. *The scan matching module*

The Scan Matching Module finds the rigid-body transformation (translation + rotation) that better aligns two consecutive scans of the laser ranger finder and the IMU (sensor fusion [12]).

The Scan Matching is based on the Canonical Scan Matcher (CSM) algorithm [13] that is a very fast variation of the Iterative Closest Point (ICP) algorithm that uses a point-to-line metric optimized for range-finder scan matching [14]. It is robust enough to be used in industrial prototypes of autonomous mobile robotics.

Given z_t, z_{t-1} and u_t, the scan matching module returns

$$\underline{\hat{x}}_t = [\hat{x}, \hat{y}, \hat{z}, \alpha, \beta, \hat{\gamma}]^{\mathrm{T}}$$

that is an estimation of the pose of the robot in the global reference frame S (Fig 2.).

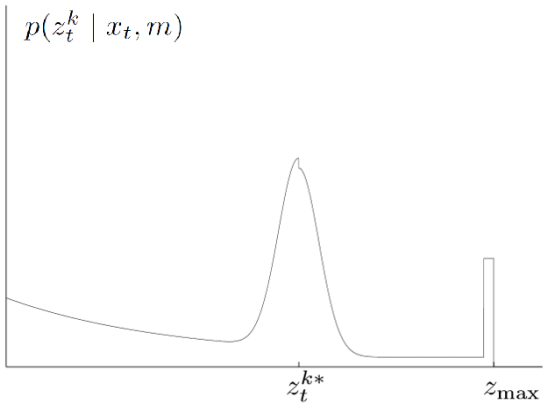


Fig 3. Lidar sensor model

u_t is a rough estimation of the movement and it can be provided by an IMU if the algorithm runs on a UAV or, it can be provided by the wheels odometry estimation, if the algorithm runs on a MAV.

The output of this module, $\underline{\hat{x}}_t$, can be considered as an accurate robot's odometry.

iv. *The SLAM module*

The SLAM module maximizes the probability given in the equation 1 by analysing the input processed in the previous modules.

The main idea follows the particle filter SLAM implementation using a Montecarlo localization based approach [15]. Initially, as illustrated in Fig 4., given the estimation of the pose of the robot $\hat{x}_t$ and the previous calculated map m_{t-1}, the proposed algorithm performs the ray casting [16] extracting the distance z_t^{k*} from the first obstacle on the map m_{t-1} at position $\hat{x}_t$. Using the specific laser sensor model (Fig. 3), the algorithm weights the measured distance, given by the laser, returning the probability to have

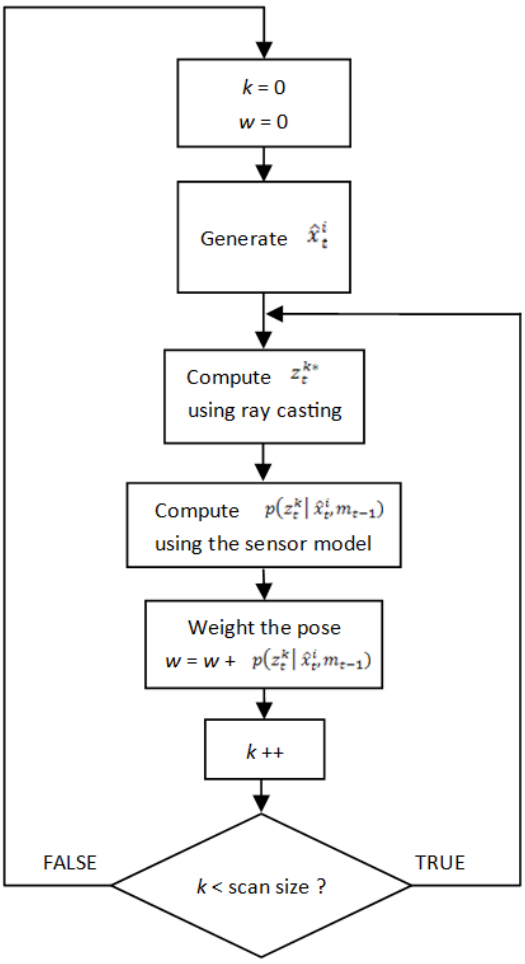


Fig 4. SLAM module flow chart

the measured distance given the position $\hat{x}_t$ and the map m_{t-1} ($p(z_t|\hat{x}_t, m_{t-1})$).

Following, the algorithm repeats the previous weights operation on several $\hat{x}_t$ generated according to the motion model of the robot returning a set of weighted poses.

Unlike particle filter SLAM, instead of keeping every particle with its own weight and its own map for all the execution time, in the new proposed method, only the position that has the highest weight is kept. This operation is performed for all the ranges at different angles of the 2D laser scanner and could be used also on 1D or 3D laser scanners.

Standard Particle Filter SLAM algorithms, running on embedded platforms, use around 10 particles during their execution having 10 consistent maps but only using 10 samples of the robot motion model at each step. The new proposed approach, running on embedded platform, uses up to 100 samples at each step. In this way, the space around $\underline{\hat{x}_t}$ is sampled with more points providing much more accurate maps.

Moreover, Standard Particle Filter SLAM algorithms need to bring always the same number of particles during the whole algorithm process, while the new proposed approach is flexible and, not being linked to previously maps, it can use different particles number according to the map error estimation and needs.

The algorithm is fast enough allowing two consecutive calculated points to be very close reducing the map errors that could come from the deletion of all previous calculated maps. We can consider that at every step, the algorithm does an error correction of the estimated pose $\hat{x}_t$.

Furthermore, referring always to a previous calculated map, the proposed algorithm keeps a global consistency, assuming that the map gathers together all the previous measurements.

v. The Mapping module

The mapping module uses OctoMap [17], an efficient probabilistic 3D mapping framework based on Octrees. The OctoMap library implements a 3D occupancy grid mapping approach, providing data structures and mapping algorithms in C++ particularly suited for robotics. The map implementation is designed to meet the following requirements: fully 3d model, updatability, flexibility and compactness. It runs on a different thread, allowing the localization algorithm not to stop when data are added to the map.

III. EXPERIMENTAL RESULTS

The validation platform is based on an IMU Xsens MTi-200 that provides the robot's accelerations and angular velocities, an Hokuyo UTM-30 LX laser scanner that provides long range (30 meters) high frequency scans (about 40 Hz sample frequency) with a wide field of view (270 degrees).

All the code runs on an Advantech MIO-2661 embedded board, with an Intel Atom N2600 (Dual Core, 1.6 GHz with Intel Hyper-Threading), 4 GB RAM DDR3 and a 150 GB SSD mSATA as storage.

Lubuntu 12.04 with the low latency kernel is used as the operating system. The algorithm is implemented as a multi thread system, with timing in order to simulate a soft real-time behaviour.

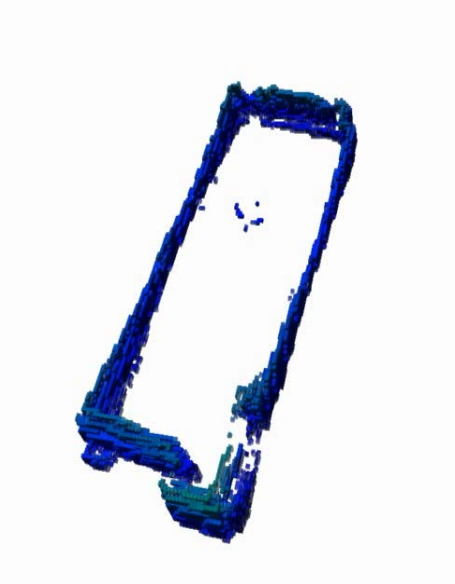


Fig 5. Five cm resolution OctoMap

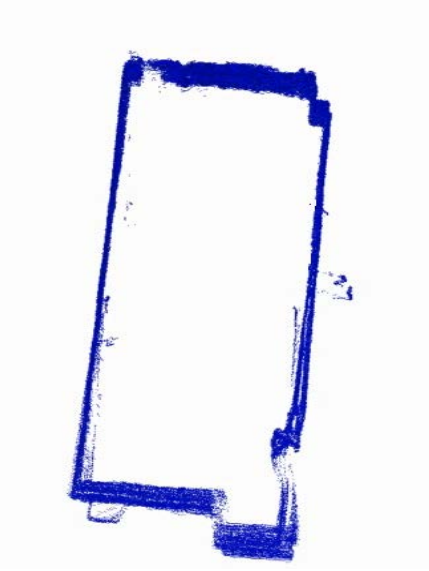
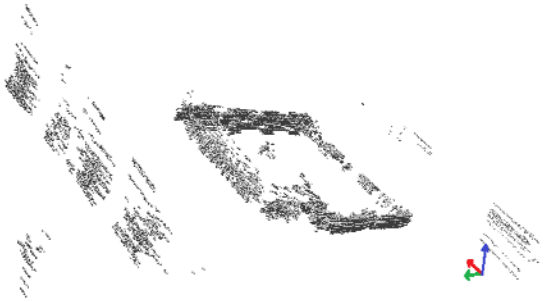
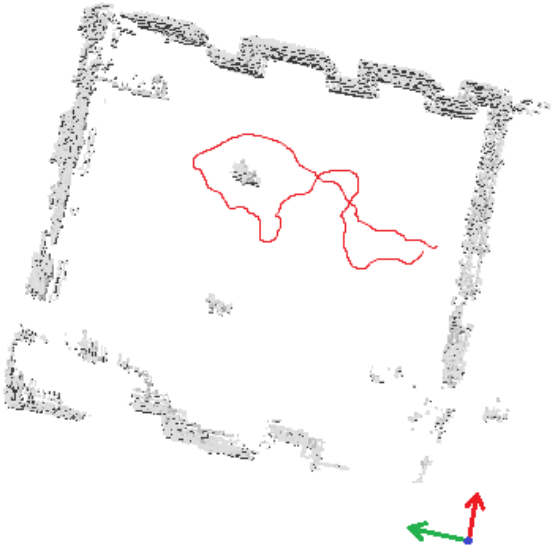


Fig 6 .One cm resolution OctoMap

(a)



(b)



(c)

Fig 7. a: picture of the environment. b: perspective view of the 3D map of the environment where the near buildings due to the open windows are also visible. c: 3D map of the environment from above with the path of the robot in marked red.

The sensors and control platform has been deployed and validated both on a terrestrial and aerial robot. Moreover, on the terrestrial platform, the proposed method has been coupled with a semi-autonomous navigation system including obstacle avoidance.

Fig. 5 shows a 5 cm resolution 3D map of a room while Fig 6. shows a 1 cm resolution 3D map of the same room.

In Fig. 7, a 5 cm 3D map is shown including also the path of the robot and a real picture of the room mapped. Closer objects and also objects present outside the room are mapped through the windows. The proposed algorithm is robust enough to map enviroment where objects with low reflectivity are present, like plastic windows, or where there is the presence of external strong light sources.
The true pathway is not shown because no absolute tracking system has been used.

IV. FUTURE WORK

Solutions to fuse the laser scanner and the IMU with another type of sensor such as a monocular camera or stereo camera are being tested and validated. Monocular and stereo cameras help loop closure operations [18], can be used as visual odometry [19] and also allow building offline 3D visual maps of the environment using some structure from motion algorithms. The use of a loop closure module inside the proposed work framework is foreseen as well as a map batch optimization.

V. CONCLUSIONS

An incremental slam algorithm based on Montecarlo localization for indoor autonomous navigation has been developed and validated.

Based on a minimal set of sensor, being light, portable and fully reconfigurable, the new proposed algorithm is suitable to be deployed on mobile robots such UAVs and ground vehicles with limited payload and low processing power.

VI. REFERENCES

[1] Bailey, T. and Durrant-Whyte, H., “Simultaneous localisation and mapping (SLAM): Part II state of the art," Robotics & Automation Magazine, Sep 2006. 20, 81

[2] G. Dissanayake, H. Durrant-Whyte, and T. Bailey. “A computationally efficient solution to the simultaneous localisation and map building (SLAM) problem”. In Proc. of the IEEE Int. Conf. on Robotics & Automation (ICRA), pages 1009–1014, San Francisco, CA, USA, 2000.

[3] K.S. Chong and L. Kleeman, “Feature-based mapping in real, large scale environments using an ultrasonic array,” Int. J. Robot. Res. , vol. 18, no. 1,

pp. 3–19, 1999

[4] G. Grisetti, C. Stachniss, and W. Burgard, GMapping Algorithm (Online). http://www.openslam.org.

[5] Simone Z and Ann E. Nicholson, “Square Root Unscented Particle Filtering for Grid Mapping”, 2009

[6] Grisetti, G., Kummerle, R., Stachniss, C., & Burgard, W. (2010). A tutorial on graph-based SLAM. Intelligent Transportation Systems Magazine, IEEE, 2(4), 31-43.

[7] A. Doucet, J.F.G. de Freitas, K. Murphy, and S. Russel. “Rao-Black-wellized particle filtering for dynamic bayesian networks”. In Proc. Of the Conf. on Uncertainty in Artificial Intelligence (UAI), pages 176–183, Stanford, CA, USA, 2000.

[8] G. Grisetti, C. Stachniss, and W. Burgard, “Improving gridbased SLAM with Rao-Blackwellized particle filters by adaptive proposals and selective resampling,” in Proc. IEEE Int. Conf. Robotics Automation, 2005, pp. 667–672.

[9] M. Montemerlo, S. Thrun, D. Koller, and B. Wegbreit. “FastSLAM: A factored solution to simultaneous localization and mapping”. In Proc. of the National Conference on Artificial Intelligence (AAAI), pages 593–598, Edmonton, Canada, 2002

[10] Thrun, Sebastian. "Probabilistic robotics." Communications of the ACM 45.3 (2002): 124-129.

[11] S. Kohlbrecher, J. Meyer, O. von Stryk, and U. Klingauf, “A Flexible and Scalable SLAM System with Full 3D Motion Estimation,” in Proceedings of IEEE International Symposium on Safety, Security and Rescue Robotics, 2011.

[12] Brooks, Richard R., and Sundararaja S. Iyengar. Multi-sensor fusion: fundamentals and applications with software. Prentice-Hall, Inc., 1998.

[13] Andrea Censi, “An ICP variant using a point-to-line metric”, Proceedings of the IEEE International Conference on Robotics and Automation (ICRA) , Pasadena, USA, May, 2008

[14] Gutmann, J-S., and Christian Schlegel. "Amos: Comparison of scan matching approaches for self-localization in indoor environments." Advanced Mobile Robot, 1996., Proceedings of the First Euromicro Workshop on. IEEE, 1996.

[15] Thrun, Sebastian. "Probabilistic robotics." Communications of the ACM 45.3 (2002): 200-212.

[16] Armin Hornung and Kai M. Wurm and Maren Bennewitz and Cyrill Stachniss and Wolfram Burgard, “OctoMap: An Efficient Probabilistic 3D Mapping Framework Based,” Autonomous Robots, 2013.

[17] Thrun, Sebastian. "Learning occupancy grid maps with forward sensor models." Autonomous robots 15.2 (2003): 111-127.

[18] M. Cummins and P. Newman, “Probabilistic appearance based navigation and loop closing,” in Proc. IEEE Int. Conf. Robotics Automation, Rome, Italy, Apr. 2007, pp. 2042–2048

[19] Scaramuzza, Davide, and Friedrich Fraundorfer. "Visual odometry [tutorial]." Robotics & Automation Magazine, IEEE 18.4 (2011): 80-92.