Indoor SLAM for Micro Aerial Vehicles Control using Monocular Camera and Sensor Fusion

Authors and affiliation (omitted for revision)

***Abstract*— This paper represents research in progress in Simultaneous Localization and Mapping (SLAM) for Micro Aerial Vehicles (MAVs) in the context of rescue and/or recognition navigation tasks in indoor environments. In this kind of applications, the MAV must rely on its own onboard sensors to autonomously navigate in unknown, hostile and GPS denied environments, such as ruined or semi-demolished buildings. This article aims to investigate a new SLAM technique that fuses visual information and measurements from the inertial measurement unit (IMU), to robustly obtain the 6DOF pose estimation of a MAV within a local map of the environment. The monocular visual SLAM algorithm along with the IMU calculate the pose estimation through an Extended Kalman Filter (EKF). The system consists of a low-cost commercial drone and a remote control unit to computationally afford the SLAM algorithms using a distributed node system based on ROS (Robot Operating System). Some experimental results show how sensor fusion improves the position estimation and the obtained map under different test conditions.**

Keywords—micro aerial vehicles; indoor navigation; sensor fusion; simultaneous localization and mapping; robot operating system

I. INTRODUCTION

The growing research on MAVs and the consequent improvement of technologies like microcomputers and onboard sensor devices have increased the performance requirements of such kind of systems. Enabled by GPS and MEMS inertial sensors, MAVs that can fly in outdoor environments without human intervention have been developed [1,2,3]. Unfortunately, most indoor environments remain without access to external positioning systems, and autonomous MAVs are very limited in their ability to operate in these areas.

Traditionally, unmanned ground vehicles operating in GPS-denied environments can rely on dead reckoning and onboard environmental sensors for localization and mapping using SLAM techniques. However, attempts to achieve the same results with MAVs have not been as successful due to several reasons: the inaccuracy and high drift of Inertial Navigation Systems (INS) compared to encoder-based dead reckoning, the limited payload for sensing and computation, and the fast and unstable dynamics of air vehicles, are the main challenges which must be tackled.

Especially, pose estimation is essential for many navigation tasks, including localization, mapping and control. The technique used depends mainly on the available on board sensors, which in aerial navigation must be carefully chosen due to payload limitations. Through their low weight and consumption, most commercial MAVs incorporate at least one monocular camera, so VSLAM (Visual SLAM) techniques have been widely used [4, 5]. However, most of these works have been limited to small workspaces that have definite image features and sufficient sunlight. Furthermore, computational time is too high for the fast dynamics of aerial vehicles, making difficult to control them. On the other hand, despite their greater weight and consumption, range sensors such as RGB-D cameras or laser range sensors have also been used on MAVs due to their fast distance detection.

The work presented in this paper is part of the ISLAMAV project (*affiliation*) whose final objective is to fuse several sensors to improve the pose estimation for MAVs in indoor environments. As a strategy of the fusion algorithm, each of the sensors must be able to provide its own pose estimation to endow the system with some redundancy that allows it to work in different environmental conditions. In [6] we presented the whole architecture –which includes laser, vision and inertial sensing-, while in this paper we focus only on monocular camera and IMU fusion.

To face the computational requirements, the system is composed of a flight and a ground unit, so that code can be distributed in different nodes using ROS (Robot Operating System).

The study explained in this paper uses two monocular VSLAM algorithms to calculate the pose estimation (along with the measurements from the IMU) and the map of the environment: LSD-SLAM [7] and ORB-SLAM [8].

One of the main problems of monocular camera VSLAM algorithms is the fact that it cannot calculate the scale of the data of tracking and mapping. It leads to a system that is not working with real-scale data, what could affect the integrity of an aerial robot. To solve this problem, our system uses the data from the IMU to calculate the dynamic scale of the SLAM and return the real-time pose of the MAV without scale ambiguity.

The remaining part of this paper is organized as follows. Section 2 discusses related work. Section 3 describes the overall system. The SLAM approach is explained in section 4. The experimental results are presented in Section 5. Finally, it is followed by the conclusion and future work in Section 6.

II. Related Work

The most challenging part of SLAM for MAVs is to obtain the 6DOF pose of the vehicle without odometry information. To do this, different sensor sources have been suggested, such as laser range sensors [9], monocular cameras [4], stereo cameras [5] or RGB-D sensors [10].

Due to weight limitations (in addition to power consumption), most of the works only use the onboard camera and IMU to apply VSLAM (Visual SLAM) techniques [11,12,13,14,15,16,17]. These systems demonstrate autonomous flight in limited indoor environments using VSLAM techniques that are out-dated, what results in inaccurate estimations and poor control results.

In this work, up-to-date VLSAM algorithms are fused with measurements from the IMU to solve the SLAM problem in complex indoor environments and robustly estimate the 6DOF pose of the MAV, using a distributed system with a flight unit and a ground station. Furthermore, the system is able to calculate the dynamic scale of the measurements, what makes it a scale-aware system. Due to it, the EKF and the control stage work with real scaled data, in contrast to other monocular VSLAM systems.

III. System Overview

We address the problem of autonomous indoor MAV localization as a software challenge, focusing on high-level algorithms integration rather than specific hardware. For this reason, we use a low-cost commercial platform with minor modifications and an open-source development platform (ROS), so that drivers of sensors and some algorithms can be used without development.

A. Hardware Architecture

Our quadrotor MAV, shown in *Fig. 1*, is the Bebop from Parrot [18], a lighter (400 gr) and smaller (33x38x3.6cm) drone than the earlier ARDrone 2.0. This MAV can carry up to 200g of payload for about 5min and is equipped with a frontal "Fisheye" camera. It counts with another vertical camera, which is used for stabilization and horizontal velocity estimation. Besides, it has an ultrasonic altimeter, a 3-axis accelerometer, 2 gyroscopes and a barometer. It incorporates an onboard controller 8 times more powerful than the one from the ARDrone 2.0 (dual-core processor Parrot P7), a quad-core graphic processor, flash memory of 8Gb and a Linux distribution. It is controlled via Wi-Fi (it provides its own net) and a SDK is available for application development.

Although the Bebop comes with some software for basic functionality, it's neither open-source nor easy to modify, and so we treat the drone as a black box, using only the available W-LAN communication channels to access and control it.

Specifically, these are the inputs/outputs we use in our SLAM system:

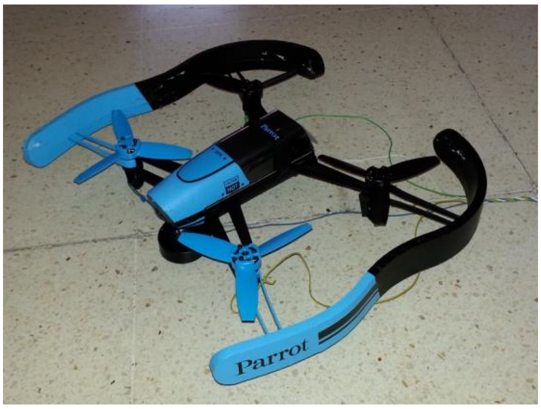

Fig. 1. Bebop Drone from Parrot, the commercial drone used as flying unit in our experiments.

- Video channel, to receive the video stream of the forwards facing camera, with maximal supported resolution of 640x368 and frame rate of 30fps.

- Navigation channel, to read onboard sensor measurements every 5ms. The data used by our system are:

  1. Drone orientation as roll, pitch and yaw angles $(\overline{\Phi}, \overline{\Theta}, \overline{\Psi})$.
  2. Horizontal velocity in drone's coordinate system $(\overline{vdx}, \overline{vdy})$, calculated onboard by an optical-flow based motion estimation algorithm [19].
  3. Drone height $\overline{h}$, obtained from the ultrasound altimeter measurements.

- Command channel, to send the drone control packages, with the desired velocities of x and y axis; vertical speed and yaw rotational velocity:

$$\boldsymbol{u} = \left(\widehat{vx}, \widehat{vy}, \widehat{vz}, \widehat{\Psi}\right) \qquad (1)$$

B. Software Architecture

As it's shown in *Fig. 2*, the onboard controller and processor perform sensor readings and basic control of the MAV. The ground station executes our SLAM system and also the control and planning strategies, the last ones being out of the scope of this paper.

The SLAM system explained in this paper consist of two major components: (a) a monocular VSLAM system that obtains a 6DOF pose estimation (and a 3D map of the environment); (b) an Extended Kalman Filter that fuses the last estimation with the navigation data provided by the onboard sensors of the MAV to obtain a robust 6DOF estimation of the position of the robot in the generated map. Besides, we have implemented a PID controller that allows the MAV to reach goal poses using the estimated position.

IV. SLAM Approach

In the following subsections, we describe the modules of the SLAM system.

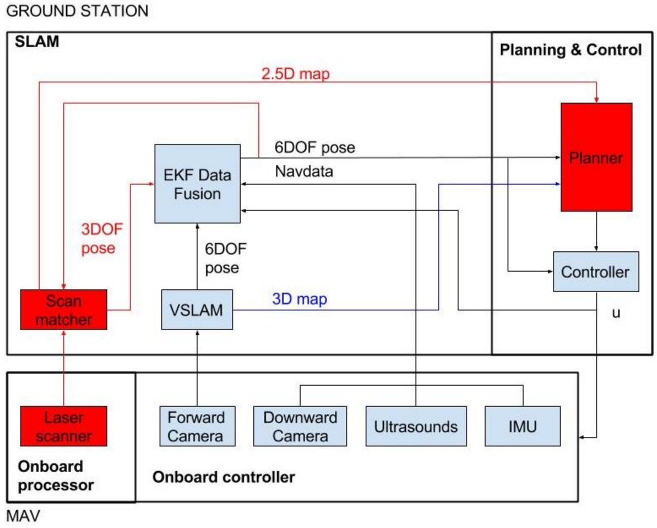

Fig. 2. Software architecture of the ISLAMAV project (red modules are out of the scope of this paper).

A. *Monocular VSLAM*

After a study of the state-of-art monocular VSLAM algorithms, we decided to implement two of these algorithms in our system: LSD-SLAM (Large-Scale Direct Monocular SLAM) and ORB-SLAM (Oriented FAST and Rotated BRIEF SLAM), both available as ROS packages.

LSD-SLAM is a direct (feature-less) monocular SLAM algorithm which, along with highly accurate pose estimation based on direct image alignment, reconstructs the 3D environment in real-time as pose-graph of keyframes with associated semi-dense depth maps. Due to the later implementation of the laser SLAM node and its 2,5D map, we are only using the 6DOF pose estimation of this algorithm as an input to the data fusion filter. We chose to use the laser's map instead the one created by LSD-SLAM because of the better accuracy of the first one and due to the computational requirements needed by the last one.

Fig. 3 shows the 3D map and pose estimation obtained by the LSD-SLAM technique in a room (up); and the 3D map and pose estimation obtained in the same room and across two corridors (down). While results are good in this case, the system needs a high amount of visual characteristics that are not available in dark zones, where it needs to be fused with other sensors. Furthermore, it is very sensitive to pure rotational movement.

On the other hand, ORB-SLAM is a feature-based monocular SLAM. ORB-SLAM estimates the drone's position in an extremely accurate way. It makes it perfect for be implemented over a system based on a MAV due to its fast and unstable dynamics. Furthermore, thanks to a smart development of the algorithm it is able to do a reliable loop closing.

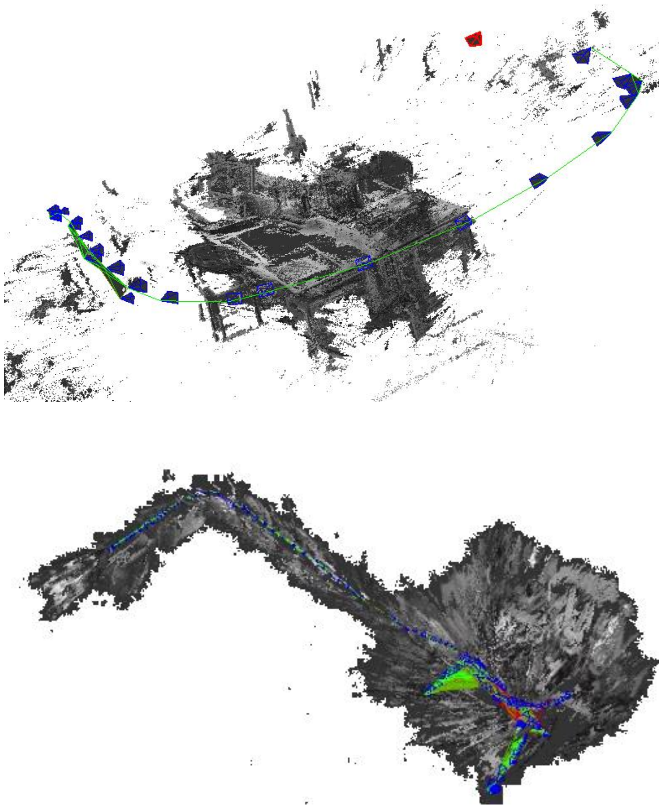

Fig. 3. Results of LSD-SLAM. The first picture represents the translation of MAV's camera around a room. The second one represents the results of the translation around the same room and along two corridors. The green line indicates de track where the camera went over. The blue marks are the camera's poses where the VSLAM algorithm captured a kayframe. The red marks correspond with the actual pose of the camera. The grey-scale shapes are the 3D map of the environment made by LSD-SLAM.

Fig. 4 shows the pose estimation obtained with ORB-SLAM in the same environment of *Fig. 3*. It can be deduced that data from other sensors is needed to correctly estimate the position of the MAV. While the tracking is correct in the room and along the corridor, it fails calculating the rotation angle after turning the corner. Furthermore, the changing scale makes to get a wrong estimation of distances (the length of the corridor after the corner is shortened).

Fig. 5 shows the results obtained when the algorithm estimates de position of the camera around a square of 35m approximately. The loop closure algorithm allows the VSLAM technique to accurately track the real time pose of the camera.

As said before, one of the main problems when working with monocular VSLAM is scale ambiguity. As we need to work with a scale-aware system, we developed a method to calculate the scale based on onboard sensing. Due to it our system works with real-scale magnitudes. To solve this problem, the system uses the altitude measurements from the altimeter and VSLAM for calculating the scale as follows:

$$scale = \frac{h_{IMU}}{h_{VSLAM}} \quad (2)$$

$$x_{REAL-SCALE} = x_{VSLAM} \cdot scale \quad (3)$$

$$y_{REAL-SCALE} = y_{VSLAM} \cdot scale \quad (4)$$

$$z_{REAL-SCALE} = z_{VSLAM} \cdot scale \quad (5)$$

The scale is calculated at each iteration of the node to avoid problems due to dynamic changes.

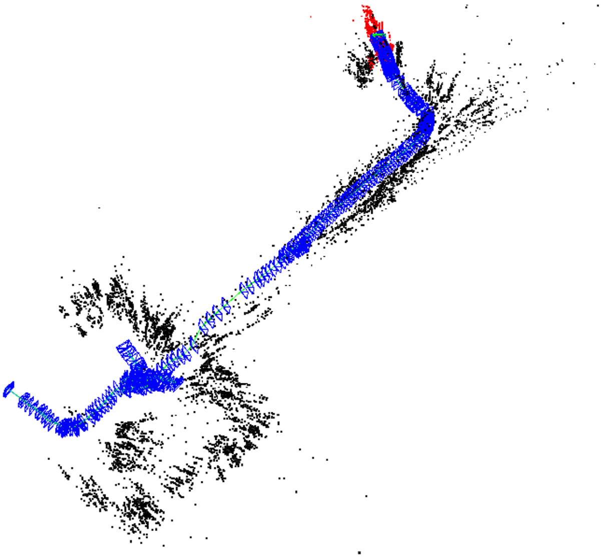

Fig. 4. Results of ORB-SLAM estimating the pose of the camera in the same environment of *Fig. 3*.

B. Data Fusion with EKF

In order to fuse all available data, we employ an Extended Kalman Filter (EKF). This EKF is also used to compensate for the different time delays in the system, as detailed described in [17], arising from wireless LAN communication and computationally complex visual tracking.

The EKF uses the following state vector:

$$\boldsymbol{\chi}_t := \left(x_t, y_t, z_t, vx_t, vy_t, vz_t, \Phi_t, \Theta_t, \Psi_t, \dot{\Psi}_t\right)^T \in \Re^{10} \quad (6)$$

where $(x_t, y_t. z_t)$ is the position of the MAV in meters (m); (vx_t, vy_t, vz_t) the velocity in meters/second (m/s); Φ_t, Θ_t, Ψ_t the roll, pitch and yaw angles in radians (rad); and $(\dot{\Psi}_t)$ the yaw-rotational speed in radians/second (rad/s). All of them are evaluated in world coordinates. In the following, we define the prediction and observation models.

1) *Prediction Model*

The prediction model is based on the full motion model of the quadcopter's flight dynamics and reaction to control commands derived in [17]. A new calibration of the model parameters has been done for the Bebop Drone.

The model establishes that the horizontal acceleration of the MAV is proportional to the horizontal force acting upon the quadcopter, that is, the accelerating force minus the drag force. The drag is proportional to the horizontal velocity of the quadcopter, while the accelerating force is proportional to a projection of the z-axis of the drone onto the horizontal plane, which leads to:

$$\dot{vx}_t = K_1\left(K_2(cos\Phi\; sin\Theta \cos\Psi + \sin\Phi \sin\Psi)\right) \quad (7)$$

$$\dot{vy}_t = K_1\left(K_2(cos\Phi\; sin\Theta \sin\Psi - \sin\Phi \cos\Psi)\right) \quad (8)$$

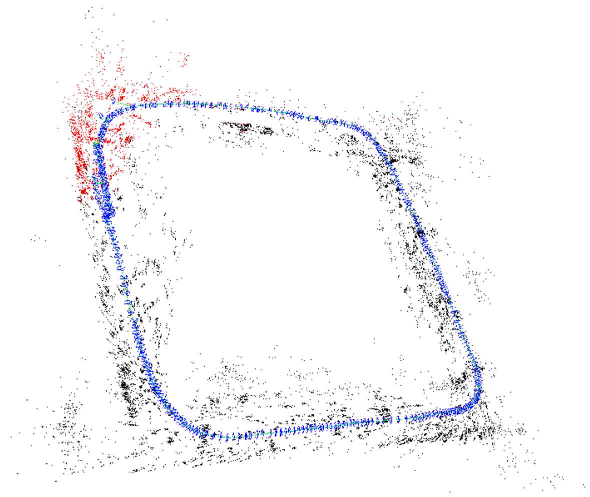

Fig. 5. Results of ORB-SLAM estimating the pose of the camera around a square.

Furthermore, the influence of the sent control command $\mathbf{u} = \left(\widehat{vx}, \widehat{vy}, \widehat{vz}, \widehat{\dot{\Psi}}\right)$ is described by the following linear model:

$$\dot{\Phi}_t = -K_3(K_4\widehat{vy}_t + \Phi_t) \quad (9)$$

$$\dot{\Theta}_t = K_3(K_4\widehat{vx}_t - \Theta_t) \quad (10)$$

$$\dot{vz}_t = K_7(K_8\widehat{vz}_t - \mathrm{vz}_t) \quad (11)$$

$$\ddot{\Psi}_t = K_5\left(K_6\widehat{\dot{\Psi}}_t - \dot{\Psi}_t\right) \quad (12)$$

We estimated the proportional coefficients K₁ to K₈ from data collected in a series of test flights. From equations (7) to (12) we obtain the overall state transition function:

$$\begin{pmatrix} x \\ y \\ z \\ vx \\ vy \\ vz \\ \Phi \\ \Theta \\ \Psi \\ \dot{\Psi} \end{pmatrix}_{t+1} \leftarrow \begin{pmatrix} x \\ y \\ z \\ vx \\ vy \\ vz \\ \Phi \\ \Theta \\ \Psi \\ \dot{\Psi} \end{pmatrix}_{t} + \Delta_t \begin{pmatrix} vx_t \\ vy_t \\ vz_t \\ K_1(K_2(cos\Phi\; sin\Theta \cos\Psi + \sin\Phi \sin\Psi) - vx_t) \\ K_1(K_2(cos\Phi\; sin\Theta \sin\Psi - \sin\Phi \cos\Psi) - vy_t) \\ K_7(K_8\widehat{vz}_t - vz_t) \\ -K_3(K_4\widehat{vy}_t + \Phi_t) \\ K_3(K_4\widehat{vx}_t - \Theta_t) \\ \dot{\Psi}_t \\ K_5\left(K_6\widehat{\dot{\Psi}}_t - \dot{\Psi}_t\right) \end{pmatrix} \quad (13)$$

2) *Inertial Navigation Observation Model*

This model relates the onboard measurements obtained through the navigation channel of the quadcopter described in section III.A (that we called "navdata" in figure 2) and the state vector. The quadcopter measures its horizontal speed $(\overline{vdx}, \overline{vdy})$ in its local coordinate system, which we transform into the world frame (vx, vy). The roll and pitch angles measured by the accelerometer are direct observations of the corresponding state variables. On the other hand, we differentiate the height measurement and the yaw measurement as observations of the respective velocities. The resulting measurement vector z_{NAVDATA} and observation function $h_{NAVDATA}(\boldsymbol{\chi}_t)$ are:

$$z_{NAVDATA} := \left(\overline{vdx}, \overline{vdy}, \bar{h}_t, \bar{\Phi}, \bar{\Theta}, \bar{\Psi}_t\right) \tag{14}$$

$$h_{NAVDATA}(\chi_t) := \begin{pmatrix} vx_t cos\Psi_t + vy_t sin\Psi_t \\ -vx_t sin\Psi_t + vy_t cos\Psi_t \\ z_t \\ \Phi_t \\ \Theta_t \\ \Psi_t \end{pmatrix} \tag{15}$$

3) *VSLAM Obvservation Model*

When the VSLAM algorithm successfully tracks a video frame, its 6DOF pose estimation is transformed from the coordinate system of the front camera to the coordinate system of the quadcopter, leading to a direct observation of the quadcopter's pose given by:

$$z_{VSLAM,t} := f\left(E_{DC} E_{C,t}\right) \quad \in \Re^6 \tag{16}$$

$$h_{VSLAM}(\chi_t) := (x_t, y_t, z_t, \Phi_t, \Theta_t, \Psi_t)^T \quad \in \Re^6 \tag{17}$$

where $E_{C,t} \in SE(3)$ is the estimated camera pose, $E_{DC} \in SE(3)$ the constant transformation from the camera to the quadcopter coordinate system and $f : SE(3) \to \Re^6$ the transformation from an element of SE(3) to the roll-pitch-yaw representation $(x, y, z, \Phi, \Theta, \Psi)$.

C. *PID Controller*

A PID controller was developed in order to control the movements of the MAV based on the estimated position. A reference $(\hat{x}, \hat{y}, \hat{z}, \hat{\Psi})$ is needed as the desired position of the drone in relation with the surroundings. The EKF will bring the estimation of the pose, as shown in Fig. 6. The difference between the reference and the estimated pose is the error that will be minimized by the PID controller, by sending to the MAV an appropriate control command **u**=$\left(\widehat{vx}, \widehat{vy}, \widehat{vz}, \hat{\Psi}\right)$, that is calculated in the following way:

$$\widehat{vx} = cos\Psi[Kp(\hat{x} - x) + Kd \cdot \dot{x}] + sin\Psi[Kp(\hat{y} - y) + Kd \cdot \dot{y}] \tag{18}$$

$$\widehat{vy} = -sin\Psi[Kp(\hat{x} - x) + Kd \cdot \dot{x}] + cos\Psi[Kp(\hat{y} - y) + Kd \cdot \dot{y}] \tag{19}$$

$$\widehat{vz} = Kp \cdot (\hat{z} - z) + Kd \cdot \dot{z} + Ki \cdot \int(\hat{z} - z) \tag{20}$$

$$\hat{\Psi} = Kp(\hat{\Psi} - \Psi) \tag{21}$$

This controller can be used to move the drone from one point to another one. It allows the algorithm drive the MAV along a series of points in the map so it follows a specific track.

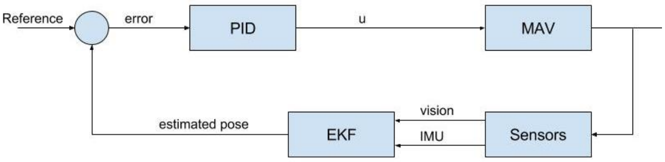

Fig. 6. PID Controller Blocks Diagram

V. RESULTS

For the purpose of testing our system with a reliable ground truth, we used a horizontal motion detector camera, which was installed in the ceiling of the test environment. It allows us to measure the XY movements of the drone using an external sensor. It is not possible to sense the altitude with this method, so we trust in the altimeter integrated in the MAV as the ground truth. This procedure allows us to contrast the position estimated by our algorithm with the true position detected by the external camera.

ORB-SLAM was used during the tests which results are represented in *Fig. 7, Fig. 8 and Fig. 9*. We used this VSLAM algorithm instead of LSD-SLAM because we didn't need the 3D map that LSD-SLAM could bring us –so the computational requirements were avoided–. Furthermore, we realized that ORB-SLAM represents a more robust VSLAM technique facing pure rotational movement and fast translations.

As said before, the PID controller allows the MAV to execute a path through a series of points. As a test, we made the drone to fly trying to recreate a square of 1mx1m –which is plotted as a green square in *Fig 7, Fig. 8, Fig. 9.*

As a first test, we run the algorithm with each of the stages of the EKF separately –this is, only with prediction stage, only with IMU correction stage and only with VSLAM correction stage–, shown in *Fig. 7*. As we are not able to test the vertical precision of the algorithm –were the IMU performance stands out– the better tracking of prediction and VLSAM correction stages are obvious. To represent the bad performance of the algorithm when it's using only the IMU measurements, *Fig. 7* plots the results of the method with all its drift. Below, *Fig. 8* includes the same graph but zoomed in order to make easy to see the differences between implementations.

The performance's improvement of the system with the addition of the stages summarized on *B* is evaluated on the *Fig 9*. As shown, the system is most accurate with prediction and both IMU and VSLAM correction stages. That precision is the cause of this project and why we are making the fusion of VSLAM and IMU measurements –and as explained in *VI*, laser as a future new stage–.

VI. CONCLUSIONS AND FUTURE WORK

This paper shows work in progress and initial results of an indoor SLAM system for MAVs that fuses measurements from a monocular camera and onboard sensors to obtain a better estimation of the 6DOF pose of the MAV and a map (3D if LSD-SLAM is being used) of the local environment.

This work provides a scale aware tracking and mapping system, which will be incorporated to the whole architecture of the ISLAMAV project [6]. This will conclude in a system that could calculate in real time the position of the drone without drift and a 2.5D template or map of the environment. This will be extremely useful to estimate the real position of the MAV. Furthermore, this system will be more robust facing problems as lighting changes.

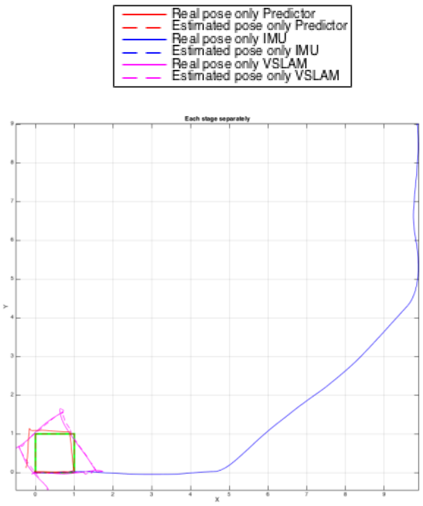

Fig. 7. Zoom out of different stages implemented separately.

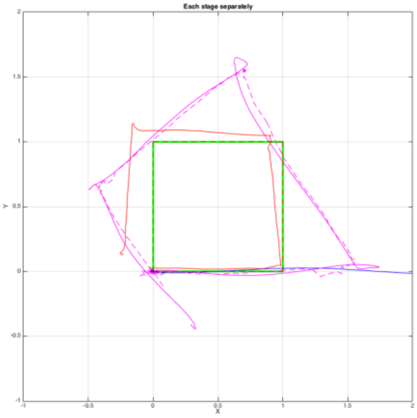

Fig. 8. Zoom in of different stages implemented separately.

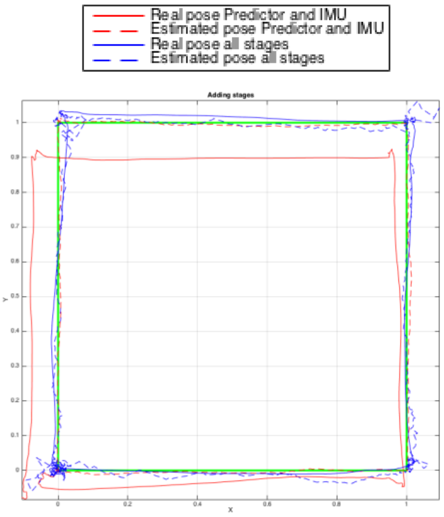

Fig. 9. Perform of added stages.

VII. Acknowledgment

Omitted for revision

References

[1] Mellinger, D. Kumar, V.: Minimum snap trajectory generation and control for quadrotors. In: Proc.IEEE Intelligent Conference on Robotics and Automation (ICRA). pp. 2520-2525. (2011)

[2] Lindsey, Q., Mellinger, D. Kumar, V.: Construction of cubic structures with quadrotor teams. In: Proceedings on Robotics: Science and Systems (RSS). (2011)

[3] Kushleyev, A., Mellinger, D., Kumar, V: Towards a swarm of agile micro quadrotors. In: Proceedings of Robotics: Science and Systems (RRS). (2012)

[4] Achtelik, M., Lynen, S., Weiss, S., Kneip, L., Chli, M.: Visual-inertial SLAM for a small helicopter in large outdoor environments. In: Proceedings IEEE Conference on Intelligent Robots and Systems (IROS).pp. 2651-2652 (2012).

[5] Fraundorfer, F., Heng, L., Honegger, D., Lee, L., Meier, L., Tanskanen, P.,Pollefeys, M.: Vision-based autonomous mapping and exploration using a quadrotor MAV. In: Proceedings IEEE Inter. Conference on Intelligent Robots and Systems (IROS). (2012)

[6] López, E., Barea, R., Gómez, A., Saltos, A., Bergasa, L., Molinos, E., Nemra, A.: Indoor SLAM for micro aerial vehicles using visual and laser sensor fusion. In Robot 2015: Second Iberian Robotics Conference: Advances in Robotics, Volumen 1 (2015).

[7] Engel, J., Schöps, T., Cremers, D.: LSD-SLAM: Large-scale direct monocular SLAM. In Computer Vision (ECCV 2014), pp.834-849. (2014)

[8] Mur-Artal, R., Montiel, J., Tardos, J.: ORB-SLAM: A Versatile and Accurate Monocular SLAM System. In IEEE Transactions on Robotics, vol. 31, no. 5, pp. 1147-1163 (2015).

[9] Grzonka, S., Grisetti, G., Burgard, W.: Towards a navigation system for autonomous indoor flying. In: Proceedings IEEE Intelligent Conference on Robotics and Automation (ICRA). (2009)

[10] Bylow, E., Sturm, J., Kerl, C., Kahl, F., Cremers, D.: Real-time camera tracking and 3D reconstruction using signed distance functions. In: Proceedings of Robotics: Science and Systems (RSS). (2013)

[11] Tournier, G.P., Valenti, M., How, J.P., Feron, E.: Estimation and control of a quadrotor vehicle using monocular vision and moiré patterns. In: Proceedings of AIAA GNC, Keystone, Colorado. (2006)

[12] Johnson, N.G.: Vision-assisted control of a hovering air vehicle in an indoor setting. Master's thesis, BYU. (2008)

[13] Kemp, C.: Visual Control of a Miniature Quad-Rotor Helicopter. PhD Thesis, Churchill College, University of Cambridge. (2006)

[14] Ahrens, S., Levine, D., Andrews, G., How, J.P.: Vision-based guidance and control of a hovering vehicle in unknown, gps-denied environments. In: IEEE International Conference on Robotics and Automation (ICRA), pp. 2643-2648. (2009)I.S. Jacobs and C.P. Bean, "Fine particles, thin films and exchange anisotropy," in Magnetism, vol. III, G.T. Rado and H. Suhl, Eds. New York: Academic, 1963, pp. 271-350.

[15] Teulière, C., Marchand, E., Eck, L.: 3-D Model-Based tracking for UAV Indoor Localization. In: IEEE Transactions on cybernetics, vol. 45, nº5. (2015)

[16] Brockers, R., Humenberger, M., Weiss, S., Matthies, L.: Towards autonomous navigation of miniature UAV. In: 2014 IEEE Conference on Computer Vision and Pattern Recognition Workshops. (2014)

[17] Engel, J., Sturm, J., Cremers, D.: Scale-Aware Navigation of a Low-Cost Quadrocopter with a Monocular Camera. In Robotics and Autonomous Systems (RAS), volume 62, 2014 (2014).

[18] http://www.parrot.com/products/bebop-drone/

[19] Bristeau, P-J., Callou, F., Vissiere, D., Peit, N.: The navigation and control technology inside the AR.Drone micro UAV. 18th IFAC World Congress, pp. 1477-1484. (2011)