Improving monocular plane-based SLAM with inertial measures

Fabien Servant, Pascal Houlier, E. Marchand

Improving monocular plane-based SLAM with inertial measures

Fabien Servant, Pascal Houlier, Eric Marchand

***Abstract*— This article presents a solution to the problem of fusing measurements acquired from a monocular camera with inertial data to achieve simultaneous localization and mapping (SLAM) tasks. This paper describes the models used to correctly integrate inertial and vision data in an EKF-SLAM based application, and ways to perform the fusion on low cost hardware. Both synthetic and real sequences show that our method work and greatly enhance classical SLAM estimation.**

I. INTRODUCTION

Allowing long-term localization without knowledge about the environment, vision-based Simultaneous Localization and Mapping (SLAM) [1], [2], [3] is widely discussed and used in many applications such as robot localization, or odometry. Using only one camera, previous works proposed to track the camera pose using SLAM and got very impressive results [3], [4], [5], [6], [2], [7].

Considering low cost hardware, monocular SLAM had to rely on robust image measurements which may be tracked even when noise or vision artifacts appear. In this paper, our work consider that common urban environment or indoor environment are mainly made with planes [8]. Thus, planes are used instead of the classically used 3D points to describe the environment. Their image movement can be perfectly modeled using an homography. The regions can thus be tracked in real time on cheap hardware using a fast and robust template tracker [9]. Previous work on SLAM has been done considering such planar structures. In [10], the system allows to recognize known planar objects and incorporate their geometry within the environment map. In [11] sparse features are "folded in" in a bottom-up process in order to form higher level structure such as planar surface.

However, considering only information provided by a vision sensor may cause severe problems due to the intrinsic principle of cameras. Images without any interesting or pertinent information are often acquired, fast camera motion may affect the measurements due to motion blur in the image, etc. Moreover, because both maps and camera pose are estimated using 2D measurements, they are only known up to a global scale (which may create estimation problems when re-measuring old map items). Using an auxiliary sensor allows to alleviate those problems and to perform robust long-term estimation in real world scenes. Inertial sensors offer measures in 3D which are completely decorrelated from those given by the camera. While vision is very robust and give a lot of information when its requirements are met, inertial sensors are less accurate but deliver 3D movement (no dimension loss) measurement whenever needed. Previous works were done on using simultaneously ("fusing") inertial and monocular vision [12], [13], [14]. All of this papers suppose that sensors are synchronized using dedicated hardware or timestamps. This is not possible on every hardware configuration (especially cheap one).

In this paper, an EKF based solution to SLAM is considered that allows to fuse information provided by both a monocular camera and an inertial sensor. To cut prices, robots –to be localized– may be built using low cost hardware. A common laptop or mobile device with basic connectors may be enough for the computation needs and associated with integrated or cheap inertial and vision sensors. A cheap USB webcam grabbing images at 15Hz is used. The IMU, also USB connected, is the XSens MTx. It contains 3 accelerometers, 3 gyroscopes and 1 magnetometer. Only raw data from the accelerometers and the gyroscope are used in our application to be as device independent as possible. Far away from high end aeronautic IMU, these little (MEMS) IMU are noisy and they deliver biased measurements. Therefore, inertial measurements can not be used alone to estimate the movement of the system. Integrating linear accelerations and angular velocities for only one minute will give an estimated position hundreds of meters away from the ground truth. Thus, they can only be used as auxiliary measurement devices or for short term replacement of the main device.

Next section described the evolution model considered in the SLAM approach. Section III presents the vision part of the system while section IV describes the fusion process with an IMU. System calibration (section V) and synchronisation (section VI) are then presented. Results finally validate the proposed approach.

II. CAMERA DYNAMICS STATE AND PREDICTION

The goal of this work is to track the pose of the camera. Camera is considered to be the principal sensor. Thus, estimated state of the system must contains the position $\mathbf{t}$ and the orientation of the camera $\mathbf{r}$ (we consider the $\theta\mathbf{u}$ representation where θ is the angle and $\mathbf{u}$ the axes of the rotation). As in every EKF based method, a prediction step is done to profit from knowledge about the dynamics of the system. There is no odometry available so a generic model must be used to do the prediction. Prediction is very useful in SLAM to track features across images (reducing features search space). A constant linear acceleration and constant angular velocity model is chosen. It is a good compromise between complexity and fitting of reality. Constant linear acceleration means, in order to have necessary values to predict the state, that velocity $\mathbf{v}$ and acceleration $\mathbf{a}$ are estimated. Constant angular velocity means this velocity $\boldsymbol{\omega}$ will also be estimated. Thus, the camera state vector is composed of 5 vectors to describe its pose and its dynamic :

$$\mathbf{x_c} = \begin{bmatrix} \mathbf{t} & \mathbf{r} & \mathbf{v} & \boldsymbol{\omega} & \mathbf{a} \end{bmatrix}^\mathsf{T} \quad (1)$$

Note that this vector contains the linear acceleration and angular velocity. These terms are the data measured by

Fabien Servant is now with INRIA Sophia Antipolis, France. Pascal Houlier is with Orange Labs, Cesson Sévigné, Eric marchand is with Université de Rennes 1, IRISA, INRIA Rennes Lagadic team, France. E-mail: eric.marchand@irisa.fr

inertial sensors and the following sections will explains why they are useful in the vision and inertial data fusion process. Prediction model f transforms each subvector of the state independently :

$$f(\mathbf{x_c}, \delta t) = \begin{bmatrix} f_{\mathbf{t}}(\mathbf{x_c}, \delta t) \\ f_{\mathbf{r}}(\mathbf{x_c}, \delta t) \\ f_{\mathbf{v}}(\mathbf{x_c}, \delta t) \\ f_{\boldsymbol{\omega}}(\mathbf{x_c}, \delta t) \\ f_{\mathbf{a}}(\mathbf{x_c}, \delta t) \end{bmatrix} \quad (2)$$

Because linear acceleration and angular velocity are considered constant by the motion model, difference between ground truth and predicted state is considered as noise in both motion parameters. To simplify computation, error is modeled as an additive white noise.

$$\mathbf{a} = \mathbf{a} + \mathbf{n_a} \quad \text{and} \quad \boldsymbol{\omega} = \boldsymbol{\omega} + \mathbf{n_\omega} \quad (3)$$

Noise for both parameters being centered on 0, they are not affected by prediction and simply copied in the new state.

$$f_{\boldsymbol{\omega}}(\mathbf{x_c}, \delta t) = \boldsymbol{\omega} \quad \text{and} \quad f_{\mathbf{a}}(\mathbf{x_c}, \delta t) = \mathbf{a} \quad (4)$$

Predicted velocity is updated by summing old value with acceleration integrated over the prediction time.

$$f_{\mathbf{v}}(\mathbf{x_c}, \delta t) = \mathbf{v} + \mathbf{a}\delta t \quad (5)$$

Rotation is stored as a $\theta\mathbf{u}$ vector. Angular velocity is integrated over prediction time and transformed in a rotation matrix using the Rodrigues formula $R()$ as well as the camera orientation. The product of both rotation is the new predicted rotation which is transformed back to $\theta\mathbf{u}$ representation using Rodrigues inverse formula $\phi()$.

$$^{c2}\mathbf{R}_{c1} = R(\boldsymbol{\omega}\delta t) \quad (6)$$

$$^{c2}\mathbf{R}_{o} = R(\mathbf{r}) \quad (7)$$

$$f_{\mathbf{r}}(\mathbf{x_c}, \delta t) = \phi(^{c2}\mathbf{R}_{c1}^{T}\, ^{c2}\mathbf{R}_{o}) \quad (8)$$

All those equations are classical. Prediction of the new position is more innovative compared to other papers about monocular/inertial fusion. Let us recall the classical cinematic differential equation of a point :

$$\dot{\mathbf{t}}(\delta t) = -\mathbf{v} - \boldsymbol{\omega} \times \mathbf{t} - \mathbf{a}\delta t \quad (9)$$

Of course, $\dot{\mathbf{t}}(\delta t)$, and by integration $\mathbf{t}(\delta t)$, depends on velocity parameters but also on angular velocity. Most papers about fusion for EKF-SLAM consider that position can be predicted simply by summing the integration of velocity and acceleration over time :

$$f_{\mathbf{t}}(\mathbf{x_c}, \delta t) = \mathbf{t} + \mathbf{v}\delta t + \mathbf{a}\frac{\delta t^2}{2} \quad (10)$$

Predicting this way is theoretically correct. However, the leverage produced by the angular velocity is melted with velocity. This means that even when there is only angular velocity, system will consider that acceleration has changed because of angular velocity effects on position change. This synthetic decorrelation of acceleration and angular velocity provokes artifacts and will results in less smooth estimation of the position. This will also create problems in correctly fusing data from the two sensors.

Integrating equation (9) over time δt gives a prediction model of the new position which correctly correlate motion parameters :

$$f_{\mathbf{t}}(\mathbf{x_c}, \delta t) = R(-\boldsymbol{\omega}\delta t)\mathbf{t} - S(-\boldsymbol{\omega}\delta t)\mathbf{v}\delta t - T(-\boldsymbol{\omega}dt)\mathbf{a}\frac{\delta t^2}{2} \quad (11)$$

where $S()$ and $T()$ are integrals of $R()$:

$$S(\theta\mathbf{u}) = \int_0^{\delta t} R(\theta\mathbf{u}\delta t)\mathbf{v} \quad (12)$$

$$= \mathbf{I} + \frac{1 - cos(\theta)}{\theta^2}[\mathbf{u}]_\times + \frac{\theta - sin(\theta)}{\theta^3}[\mathbf{u}]_\times^2 \quad (13)$$

$$T(\theta\mathbf{u}) = \int_0^{\delta t} R(\theta\mathbf{u}\delta t)\mathbf{a}\delta t \quad (14)$$

$$= \mathbf{I} + 2\frac{\theta - sin(\theta)}{\theta^3}[\mathbf{u}]_\times + \frac{\theta^2 - 2(1 - cos(\theta))}{\theta^4}[\mathbf{u}]_\times^2 \quad (15)$$

These equations were adapted from [15] which considers the inverse problem (static camera and dynamic target which position is to be predicted). Experiments proved that using these equations, estimation of linear acceleration and angular velocity are very close to ground truth measured by inertial sensors and much smoother.

III. Vision-based information

The proposed approach considers that SLAM map can be represented by associations of cameras and planes tracked over frames in the image.

A. Introducing planar surfaces in SLAM process

A plane is represented by a normal vector $\mathbf{n}$ and a scalar d which codes for the orthogonal distance to the origin so that $\mathbf{nX} + d = 0$ where $\mathbf{X}$ is a point of the plane.

Both $\mathbf{n}$ and d are needed to compute the measurement model. However, we choose to represent the angle defined by $\mathbf{n}$ by its corresponding spherical coordinates θ and ϕ which does not need to be normalized after the update step of the EKF and make its initial uncertainty determination easier. A function $n(\mathbf{s})$ is used to transform a spherical coordinates vector $\mathbf{s}$ in a normal vector : $\mathbf{N} = \begin{bmatrix} \phi & \theta & d \end{bmatrix}$. The planes dimensions are not estimated but used as constants to reduce the image search space when looking for planes in the sequence.

B. Homography

Let us define $\mathbf{X}$ as a point on the plane $\mathbf{N}$ and $w(.)$ a function which normalizes a vector in $\mathbb{R}^3$ by dividing it with its third component. There exists a matrix $^{\mathbf{b}}\mathbf{H}_{\mathbf{a}}$ called an homography which transforms a projected point in frame a (the reference camera frame) into its projected coordinates in frame b. This matrix is defined up to a scale factor.

$$^b\mathbf{X} = w(^b\mathbf{H}_a\, ^a\mathbf{X}) \quad (16)$$

$$^b\mathbf{H}_a = ^b\mathbf{R}_a + \frac{^b\mathbf{t}_a}{^a d}n(^a\mathbf{s})^T \quad (17)$$

In our method, we use the ESM [9] homography tracker which gives the homography (in pixels) $^b\mathbf{G}_a$ for a specified patch between a reference image and the current one. Using the camera internal parameters matrix $\mathbf{K}$, the homography in meters is retrieved with

$$^b\mathbf{H}_a = \mathbf{K}^{-1}\, ^b\mathbf{G}_a\mathbf{K} \quad (18)$$

This method [9] is inspired by differentials trackers [16] and takes advantage of an efficient second order optimization method. This tracking approach has proved to be very efficient in various context such as robotics or augmented reality. The tracker was modified using robust estimation method (M-Estimators) and is able to handle occlusion (up to 60 percent of the patch in some scenes).

C. Adding new features in the EKF

For this paper, a template recognition method (based on [17]) is used to detect patches of interests. Once we estimate that a given image zone represents one of the planes, we can use this information in our SLAM method. A new homography tracker is instantiated with the current image as the reference one and a series of i points which define the reference image zone to be tracked. EKF needs an estimate of the measurement to compare and update its state. Since we already have an estimate of the current camera pose, we need to add both the reference camera pose and the plane associated to the new patches.

The plane is added to the state vector once for all and may be shared by many tracked patches. Of course, information and uncertainty about the plane is not dependent of cameras or anything else. This information is already in world frame coordinates and uncertainty comes from the database only. Hence there is no covariance at initialisation between the new plane and the other parts of the state vector. Let $g(\widehat{\mathbf{x}}_c, \mathbf{l})$ be the function which augments the state vector estimation $\widehat{\mathbf{x}}$ with the vector $\mathbf{l}$.

$$g_{plane}(\widehat{\mathbf{x}}, \mathbf{N}) = \left[\begin{array}{cc} \widehat{\mathbf{x}} & \mathbf{N} \end{array} \right] \tag{19}$$

Contrary to the plane, the information and uncertainty about the reference camera is strictly correlated with the current camera pose. There may be multiple reference cameras for a same plane :

$$g_{refcam}(\widehat{\mathbf{x}}) = \left[\begin{array}{ccc} \widehat{\mathbf{x}} & \mathbf{t} & \theta\mathbf{u} \end{array} \right] \tag{20}$$

The current camera frame will be noted $c2$ while the reference camera frame will be noted $c1$ in further equations. A map element may then be considered as a combination of one plane and one reference camera.

D. Observation Model

The measurement vector is given by $\mathbf{^{c2}H_{c1}}$. This measurement can be estimated using the observation model :

$$\begin{aligned}
{}^{\mathbf{n}}c1 &= R({}^{c1}\theta\mathbf{u}_o)n(\mathbf{s}) && (21)\\
{}^{d}c1 &= d - ({}^{\mathbf{n}}c1^{Tc1}\mathbf{t}_o) && (22)\\
{}^{c2}\mathbf{R}_{c1} &= R({}^{c2}\theta\mathbf{u}_o)R({}^{c1}\theta\mathbf{u}_o)^T && (23)\\
{}^{c2}\mathbf{t}_{c1} &= -{}^{c2}\mathbf{R}_1{}^{c1}\mathbf{t}_o + {}^{c2}\mathbf{t}_o && (24)\\
h_j(\mathbf{x}_{c1}, \mathbf{x}_{c2}, \mathbf{N}) &= {}^{c2}\mathbf{R}_{c1} + \frac{{}^{c2}\mathbf{t}_{c1}}{{}^{d}c1}{}^{\mathbf{n}}c1^T && (25)
\end{aligned}$$

The Jacobian $\mathbf{J_h}$ of the measurement model $h(.)$ will help transfer innovation of the measurement to both cameras and the plane.

$$\mathbf{J_h} = \left[\begin{array}{cccccc} \frac{\partial \mathbf{h}}{\partial \mathbf{x_{c2}}} & \mathbf{0} & \frac{\partial \mathbf{h}}{\partial \mathbf{x_{c1}}} & \mathbf{0} & \frac{\partial \mathbf{h}}{\partial \mathbf{N}} & \mathbf{0} \end{array} \right] \tag{26}$$

For new patches, the test can be used to check if there is a plane in our database which produce a homography estimation similar to the measurement. If the test is successful, the new patch is associated to the plane and used in our pose estimation process (see section III-C).

E. Measurement noise

The uncertainty of the measurement is a key feature of the EKF. Wrong uncertainty will lead to a bad estimation of the innovation covariance and thus will badly update the whole state. Because each element of the computed homography is tightly related to the others, the homography covariance is not easy to estimate and must be built from a simpler geometric structure.

Let's define $\mathbf{p}$ as a vector containing a list of points uniformly distributed in the reference image tracked patches. Let's define $\mathbf{p}'$ as the points $\mathbf{p}$ transformed through the homography $\mathbf{H}$. The uncertainty (defined by the covariance $\mathbf{\Sigma_{p'}}$) of $\mathbf{p}'$ is only coming from the uncertainty of $\mathbf{H}$ (defined by the covariance $\mathbf{\Sigma_h} \in \mathbb{R}^{8,8}$). Let $\mathbf{J}_h$ be $\frac{\partial \mathbf{w}(\mathbf{Hp})}{\partial \mathbf{H}}$, then

$$\mathbf{\Sigma_{p'}} = \mathbf{J}_h \mathbf{\Sigma_h} \mathbf{J}_h^T \tag{27}$$

If we can have a coarse estimate of $\mathbf{\Sigma_{p'}}$, $\mathbf{\Sigma_h}$ can be estimated using

$$\mathbf{\Sigma_h} = (\mathbf{J}_h^T \mathbf{\Sigma_{p'}}^{-1} \mathbf{J}_h)^+ \tag{28}$$

In our experimentations, we set $\mathbf{\Sigma_{p'}}$ to be a diagonal matrix (no correlation between x and y coordinates) and use the same covariance for each point.

IV. Modeling the inertial sensors

For visual odometry, update is done by comparing the measurements with the predicted ones. Prediction of the measures is done using data contained in the state vector. Both the camera pose parameters and the measured items are contained in this vector. This is the basic of EKF-SLAM. The same principle is used to integrate inertial data with some differences. Because inertial sensors are blind, they only use the camera state vector and there is no problem of association between map and measurements. They are also much more frequent and must be handled quickly. To predict the measurements, a good model of the sensor is necessary. Let us note that both inertial sensors measurement are noisy and that, like the visual measurements, this noise is modeled by a white noise random variable. The covariance matrix of this noise is static and fixed after offline calibration (see Fig. 1). This covariance matrix will be used in the update step of the EKF. In the following equations, this measurement noise is hidden.

A. Gyroscope model

Gyroscope directly gives its own angular velocity with some bias. This bias $\mathbf{b}_{gyr}$ (up to 10^-2 rad/s) is not constant and as such must be added to the state vector :

$$\mathbf{h}_{gyr} = {}^{i}\boldsymbol{\omega} + \mathbf{b}_{gyr} \tag{29}$$

Inertial sensors and camera being rigidly mounted, they have the same angular velocity. However, they are not given in the same reference frame. The gyroscope measurement model is given by the following equation :

$$\mathbf{h}_{gyr} = {}^{i}\mathbf{R}_c{}^{c}\boldsymbol{\omega} + \mathbf{b}_{gyr} \tag{30}$$

where ${}^{i}\mathbf{R}_c$ is the rotation between the camera frame $\mathcal{F}_c$ and the IMU frame $\mathcal{F}_i$.

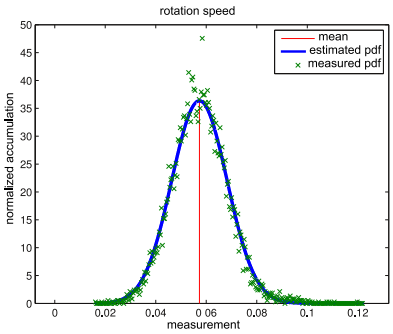


Fig. 1. Estimation of a probability density function of the gyroscope measures when the sensor is not moving

B. Accelerometer model

Accelerometer model is more complex because many parameters change the measured acceleration. First of all, there is a bias $\mathbf{b}_{acc}$ like in the gyroscope measurements which must also be added to the state vector :

$$\mathbf{h}_{acc} = {}^{i}\mathbf{a} + \mathbf{b}_{acc} \quad (31)$$

Moreover, accelerometers measure their own gravity in the earth frame. As every body on earth, the IMU is accelerated by the gravity vector $\mathbf{g}$ (which may be considered constant in the reference frame). The measurement model is modified to handle this :

$$\mathbf{h}_{acc} = {}^{i}\mathbf{a} - {}^{i}\mathbf{R}_{o}\mathbf{g} + \mathbf{b}_{acc} \quad (32)$$

The centripetal acceleration, which is one of the side effects of the angular velocity on acceleration, must be handled. The prediction model (11) allows to correctly predict this acceleration as each term is correctly estimated :

$$\mathbf{h}_{acc} = {}^{i}\mathbf{a} + [{}^{i}\boldsymbol{\omega}]_{\times}{}^{i}\mathbf{v} - {}^{i}\mathbf{R}_{o}\mathbf{g} + \mathbf{b}_{acc} \quad (33)$$

The other side effect is the Coriolis force but is neglected due to its low value. Because the measurement prediction must be computed given state vector parameters, they must be converted from $\mathcal{F}_c$ to $\mathcal{F}_i$:

$$\mathbf{h}_{acc} = {}^{i}\mathbf{R}_{c}({}^{c}\mathbf{a} + [{}^{c}\boldsymbol{\omega}]_{\times}{}^{c}\mathbf{v} - {}^{c}\mathbf{R}_{o}\mathbf{g}) + \mathbf{b}_{acc} \quad (34)$$

Translation ${}^{i}\mathbf{t}_{c}$ between the camera and the IMU also have effects on the measured acceleration : their position relative to the instantaneous rotation axis is not the same and as such, sensors are not equally affected by angular velocity. However, in our system, both sensors are very close to each other (${}^{i}\mathbf{t}_{c}$ is small) so this effect is neglected.

C. Biases

To estimate biases, camera state vector is updated with the two components :

$$\mathbf{x_c} = \begin{bmatrix} \mathbf{t} & \mathbf{r} & \mathbf{v} & \boldsymbol{\omega} & \mathbf{a} & \mathbf{b}_{gyr} & \mathbf{b}_{acc} \end{bmatrix}^{\mathsf{T}} \quad (35)$$

Their means across experiments were null so they can be initialized as such. Biases change over time is considered constant so prediction model does not modify their values. However, the prediction model covariance matrix is modified such that variance is added for the bias terms at each prediction step to handle changes.

V. IMU-Camera calibration

${}^{i}\mathbf{R}_{c}$ is part of the inertial sensors prediction model. Camera and inertial unit are rigidly attached to each other meaning that their relative rotation can be calibrated offline. Because this rotation is used to estimate the measured gravity, high precision for the calibration is mandatory. The very high norm of the gravity cause important biases in the accelerometers measurement estimation if the calibration is not accurate. A chessboard pattern is placed in the environment and filmed with the camera n times at different poses. The upper left pattern corner is considered as the reference frame $\mathcal{F}_o$. Linear acceleration ${}^{i(j)}\mathbf{a}$ is measured simultaneously. The system being static during capture, acceleration only contains the gravity. From the chessboard pattern, the camera rotation ${}^{c(j)}\mathbf{R}_{o}$ is computed at each step j.

$n(n-1)$ couples of different poses ($k = [1, n], l = [1, n]$) are built. The rotations ${}^{i(k)}\mathbf{R}_{i(l)}$ are computed (from their two respective gravity vectorsas well as the rotations matrices ${}^{c(k)}\mathbf{R}_{c(l)}$. Because ${}^{i}\mathbf{R}_{c}$ is constant, the following equation holds :

$${}^{i(k)}\mathbf{R}_{i(l)}{}^{i}\mathbf{R}_{c} = {}^{i}\mathbf{R}_{c}{}^{c(k)}\mathbf{R}_{c(l)} \quad (36)$$

All ${}^{i(k)}\mathbf{R}_{i(l)}$ are stacked in a matrix $\mathbf{A}$ and ${}^{c(k)}\mathbf{R}_{c(l)}$ in $\mathbf{B}$:

$$\mathbf{A}{}^{i}\mathbf{R}_{c} = {}^{i}\mathbf{R}_{c}\mathbf{B} \quad (37)$$

${}^{i}\mathbf{R}_{c}$ can thus be found using the well known Tsai calibration method. Because of noises in measurements and because ${}^{i(k)}\mathbf{R}_{i(l)}$ may be singular or poorly informative, n needs to be large (50 in our experiments) for good results.

VI. Sensors synchronization

Both sensors provides information which is time dependent. To fuse data from different sensors, synchronization of data is mandatory. Otherwise, measurements can be seen as the observation of different systems. Controlling the acquisition time of measurements is a major requirement. In high-end systems, dedicated hardware is used to synchronize measurements by triggering sensors when needed. This is not available on integrated and USB connected cameras. Although absolute measurement time is not needed, real delay between measurements is needed to compute prediction time and to sort measurements.

A. Constraints on fusion

Even at the hardware level, measurement frequency is not constant. Specified frequency of sensors is only correct for the mean of a subset of measurements (e.g. mean of the frequency of 10 consecutive measurements will be equal to the announced frequency). This means that it is impossible to predict offline the delay between measurements even for only one sensor. Moreover, time between measurement and its availability on the computer (mostly transfer times) can not be known but may be considered constant at least for some time. This is of course an approximation of reality but the variation proved to be small enough (in our experiments) to be ignored.

B. Computing delays

Let t_{ine} and t_{cam} be the real measurement time of the sensors (respectively inertial and camera). Let t'_{ine} and t'_{cam} be the computer availability time of the measurements :

$$t_{ine} = t'_{ine} + d_{ine} \quad (38)$$
$$t_{cam} = t'_{cam} + d_{cam} \quad (39)$$

Data transfer times for cameras is more important than inertial measurements. Thus, the difference d of availability delay for data measured simultaneously (at time t) on both sensors is :

$$\begin{aligned} d &= d_{cam} - d_{ine} = (t - t'_{cam}) - (t - t'_{ine}) \\ &= t'_{ine} - t'_{cam} \end{aligned} \quad (40)$$

Then t'_{cam} is considered as t_{cam} (remember we are only interested in time difference) and thus

$$\begin{aligned} t_{ine} &= t'_{ine} + d_{cam} - d = t'_{ine} + t_{cam} - t'_{cam} - d \\ t_{ine} &= t'_{ine} - d \end{aligned} \quad (41)$$

Of course, d must be known prior to integrate measurements in the filter.

One thread is used to acquire at high frequency values from the inertial sensors and place them in a queue buffer. Another thread acquires camera measurements. Difference between current camera frame measurement time and buffered inertial sensors measurements tell us which inertial measurements were measured between the last and current image. Those inertial measurements are then fused using correct order and prediction time. There is no need for intermediate pose between camera frames so inertial data can be fused just before camera measurements.

C. Prior estimation of difference

Because it is not possible to get precisely two simultaneously measured data from both sensors, d can not be computed directly. It must be estimated from previously acquired data. Regularly, when camera measurements are good enough, inertial measurements fusion is deactivated. State estimation is then only camera based and used to synchronize sensors. This state contains the angular velocity vector $\boldsymbol{\omega}$. Because camera and inertial sensors are rigidly attached, $||\boldsymbol{\omega}||$ is the same for both. Fig. 2 plots $||\boldsymbol{\omega}||$ for both sensors. Up to a bias in time (in fact d), both curves are similar. The difference is mainly due to high frequency noise. So by minimizing the function

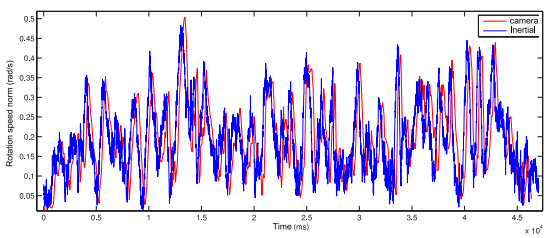


Fig. 2. Comparison of $||\boldsymbol{\omega}||$ for both sensors on a selected measurement subset

$$\underset{d}{\operatorname{argmin}} \sum_{t} (||^{c}\boldsymbol{\omega}(t)|| - ||^{i}\boldsymbol{\omega}(t+d)||)^2 \quad (42)$$

using d as the variable, the delay can be estimated and used for some time to fuse data.

VII. Results

A. Fast motion

To make monocular SLAM efficient, mapped environment features must be observed as many times and as long as possible. By measuring features from multiple views and simultaneously with other features, the map items correlation will be enforced and the estimation improved. Thus, the goal is to avoid loss of image features. To track across images a feature, two major methods are used. Thanks to the prediction model and the estimated state, SLAM offers the possibility to compute a confidence region in the image where the feature should be. The tracker himself, through its minimization process, is also used to estimate the feature parameters. When doing a fast and sudden move, prediction model may be broken and may provide a wrong confidence region. Even if the tracker minimization is efficient, the distance between the estimated state and the ground truth state may be too important for the tracker to converge onto the global minima. Both methods being useless in this case, the feature is lost. By using the inertial sensors, which is working at a higher rate than camera, the predicted state is improved as the movement is directly measured (see Fig. 3).

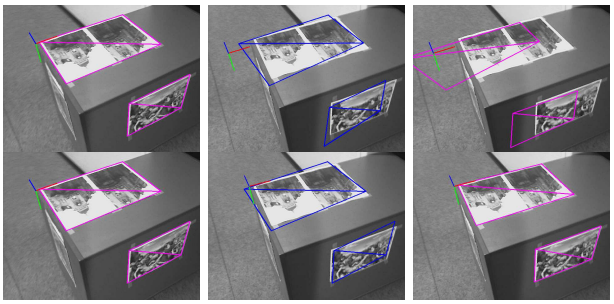


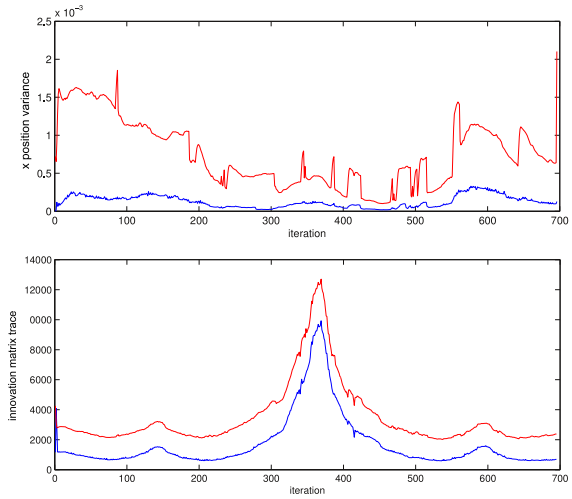
Fig. 3. Fast and sudden move example. Left column shows image acquired before the movement. Middle column shows image after move and the predicted feature state. The right column shows the tracking results. In the top row, inertial sensors are not used. Pink shapes are tracked regions, blue shapes are predicted ones.

In Fig.4, due to blur in the image visual measurement are not available for a few seconds. Without inertial sensors, the predicted state is too far away from ground truth for the tracker to converge when the blur stops.

B. Uncertainty and scale

For both prediction and update steps, EKF linearized the models. The covariance is updated using the models' Jacobians. The more important the variances are, the more important the linearization effects will be. Fig.5a shows how the inertial sensors reduce the uncertainty of the estimated state (improving the quality of estimation). State variance being reduced, the predicted measurement variance is also less important. Fig.5b shows the difference of variance with and without inertial sensors. Reduced predicted measurement variance means reduced covariance region allowing a better tracking.

Fig. 4. Occlusion simulation example. First and last image show images before and after occlusion. This occlusion lasts more than 3 seconds. Thanks to the inertial sensors, intermediate images shows that the pose is efficiently estimated.



a

b

Fig. 5. Comparison of (a) position variance and (b) measurement prediction variance with (in blue) and without (in red) inertial sensors.

Because camera pose and map are simultaneously estimated using 2D projected measurements, both are estimated up to a scale. This may cause problems when merging two different sub-maps with different scales. Inertial sensors, by measuring 3D movements, remove this unknown (see Fig.6).

C. A complete example

Fig.7 shows a real scene using our SLAM method with the described hardware. After the computation of the camera pose, a poster is artificially inserted in the scene to augment it. Even if only few planes are tracked simultaneously, map is correctly built thanks to the inertial sensors. Without this sensor, scale factor is unknown and the computation fails. Indeed, one plane appears just after that the last one disappears. Therefore, using camera only SLAM, there is no correlation between the previous and the current maps. When the camera loops back to the old items, there is a conflict in the scale and old items states are badly predicted. This problem is easily solved when both camera and IMU are considered.

VIII. Conclusion

This article describes our solution to the problem of fusing inertial and monocular measurements in EKF-SLAM using

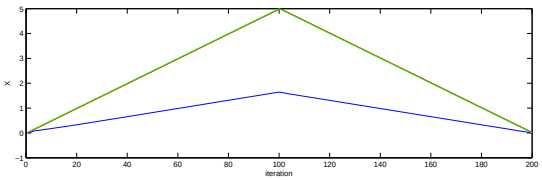


Fig. 6. Unknown scale factor example. Green curve is ground truth position. Blue curve is vision only estimated position. Red curve is the two sensors estimated position.

Fig. 7. An example of our method using both sensors

cheap devices. Our application of pose tracking using plane based SLAM was greatly enhanced by this fusion. From pure numerical problems such as linearization to plane parameters estimation, every part of the algorithm is made more robust.

References

[1] S. Se, D. Lowe, J. Little, "Vision-based mobile robot localization and mapping using scale-invariant features," in *IEEE ICRA'01*, Seoul, Korea, May 2001, pp. 2051–2058.

[2] T. Lemaire, C. Verger, I.-K. Jung, S. Lacroix, "Vision-based slam : Stereo and monocular approaches," *IJCV*, vol. 74, no. 3, Sep. 2007.

[3] A. Davison, D. Murray, "Simultaneous localization and map-building using active vision," *IEEE PAMI*, 24(7) :865–880, Jul. 2002.

[4] J. Montiel, J. Civera, A. Davison, "Unified inverse depth parametrization for monocular slam," in *RSS'07*, Philadelphia, August 2006.

[5] E. Eade, T. Drummond, "Scalable monocular slam," in *IEEE CVPR'2006*, Jun. 2006, pp. 469–476.

[6] A. Davison, I. Reid, N. Molton, O. Stasse, "Monoslam : Real-time single camera slam," *IEEE PAMI*, 29(6) :1052–1067, 2007.

[7] G. Silveira, E. Malis, P. Rives, "An efficient direct approach to visual SLAM," *IEEE Trans. on Robotics*, 24(5) :969–979, Oct. 2008.

[8] F. Servant, E. Marchand, P. Houlier, I. Marchal, "Visual planes-based simultaneous localization and model refinement for augmented reality," in *IAPR ICPR'08*, Tampa, Florida, dec 2008.

[9] S. Benhimane, E. Malis, "Homography-based 2d visual tracking and servoing," *Int. Journal of Robotics Research*, 26(7) :661–676, Jul. 2007

[10] R. Castle et al., "Towards simultaneous recognition, localization and mapping for hand-held and wearable cameras," in *ICRA'2007*, Roma, april 2007.

[11] A. Gee et al., "Discovering higher level structure in visual slam," *IEEE T. on Robotics*, 24(5) :980 –990, Oct. 2008.

[12] P. Gemeiner, P. Einramhof, M. Vincze, "Simultaneous motion and structure estimation by fusion of inertial and vision data," *Int. J. Rob. Res.*, vol. 26, no. 6, pp. 591–605, 2007.

[13] P. Pinies, T. Lupton, S. Sukkarieh, J. Tardos, "Inertial aiding of inverse depth slam using a monocular camera," in *ICRA'07*, Roma, Apr. 2007.

[14] M. Baldwin, Trumpf, "A nonlinear observer for 6 dof pose estimation from inertial and bearing measurements," *IEEE ICRA*, 2009.

[15] Z. Zhang, O. Faugeras, *Three D-Dynamic Scene Analysis : A Stereo Based Approach*. Secaucus, NJ, USA : Springer-Verlag, 1992.

[16] G. Hager, P. Belhumeur, "Efficient region tracking with parametric models of geometry and illumination," *IEEE PAMI*, 20(10) :1025–1039, Oct. 1998.

[17] D. Lowe, "Distinctive image features from scale-invariant keypoints," *Int. Journal of Computer Vision*, vol. 60, no. 2, pp. 91–110, 2004.