Bi-Objective Bundle Adjustment With Application to Multi-Sensor SLAM

Julien Michot[1] julien.michot.fr@gmail.com
Adrien Bartoli[2] adrien.bartoli@gmail.com
François Gaspard[1] françois.gaspard@cea.fr

[1]CEA, LIST, Vision and Content Engineering Lab, Point Courrier 94, Gif-sur-Yvette, F-91191 France;
[2]Clermont Université Clermont-Ferrand, France

Abstract

We consider multi-sensor data fusion problem in a Simultaneous Localization And Mapping (SLAM) application. More precisely, we tackle the integration of inertial information in Bundle Adjustment. The BA cost function is composed of two weighted objectives, one from each of the sensor, and interest the problem of finding an efficient weight that relies the two sensors errors.

An investigation of three fully automatic methods for determining the weight in an extended Bundle Adjustment is presented. The methods are real-time and can be employed with any generic least squares constraint (on the orientation, the position etc.). We present and compare the three weighting methods with two multi-sensors SLAM, with an odometer or a gyroscope. Results show that the inertial integration with an automatic weighting method decreases the drift on the final localization.Best results are obtained with the L-Curve weighting technique.

1. Introduction

Real-time Simultaneous Localization And Mapping (SLAM) has been an active research field in robotics for decades. It has been introduced more recently in computer vision with the emergence of embedded camera devices. The aim of real-time SLAM is to provide fast and robust localization (up to a scale) of a system in a unknown rigid environment. One of the main benefits of camera sensors is their ability to provide redundant informations on the observed scene and on the system displacement. Recent results show that with only one camera, one can localize a system for multiple applications such as autonomous vehicles, augmented reality mobility, etc. [6, 16, 11].

However, monocular Visual SLAM is not a robust solution to localize a system in all circumstances (poor lighting conditions, blur, etc.). For VSLAM based on incremental Structure-from-Motion (SfM) techniques, there also exist critical motions (for instance, a pure rotation) that are not well managed by SfM algorithms and lead to an undetermined or untrusted localization. Besides, image noise and model approximation are sources of an error accumulation that leads to a growing error on the final localization, known as the drift (on position, orientation and scale).

These drawbacks can be reduced using external information, from heterogeneous sources. Many systems now embed multiple sensors, depending on the applications and system characteristics. For outdoor systems, the Global Positioning System (GPS) is generally employed [1] since it has the great advantage to provide absolute information, with a low frequency. Vehicles can also embed odometers on wheels [3, 5, 1]. For indoor environments, inertial sensors such as gyrometers and accelerometers, usually integrated in a Inertial Measurement Unit (IMU) [10, 19, 9, 17] can be employed. One can also use heterogeneous devices that sense the world, such as laser range-finders and sonars.

These redundant information are gathered to recover the system localization that statistically best explains all measurements. In order to achieve this inverse problem, it is usually expressed in a Bayesian Filtering framework. Measurements and system state are considered as random variables for which we want to estimate their probability density function (*pdf*). When Gaussian *pdf* are assumed, the problem is then reduced to a nonlinear least squares optimization issue. Classical Bayesian Filters are the Extended Kalman Filter (EKF)[6, 19, 15, 4] and the Particle Filter (PF)[14]. More recently, Bundle Adjustment (BA)[20] has demonstrated its efficiency on localizing a camera-based system on different configurations (hand-held, vehicles...). BA is a nonlinear least squares refinement based on a damped Gauss-Newton optimization, such as Levenberg-Marquardt. It is known that BA is slower than EKF but has better results and runs in real-time.

Visual SLAM are usually either based on Kalman filter (and its variants) or Bundle Adjustement. The robotic community has been very productive in Kalman based multi-sensor SLAM. One can for instance see Strelow *et al.* [19] and the state-of the art therein. Kalman-based visual SLAM has the great advantage to handle easily measurement covariances, but they are less accurate than BA-based SLAM for large-scale environments. Therefore, we interest the BA-based SLAM and tackle the problem of inertial data integration in Bundle Adjustment. In a related work, Konolige *et al.* [12] describe a precise and real-time visual odometry system with integration of inertial data. They perform a visual BA-based SLAM based (with a Local Bundle Adjustement) and do visual and IMU fusion with an EKF. Strelow *et al.* [19] a batch (off-line) Accelerometers (gravity) and gyroscopes data are used to filter the system orientation. Cumani *et al.* [5] also propose a monocular SLAM with an odometer. The odometer is used to estimate the initial scale factor of the trajectory and to correct the SLAM scale factor drift. The method correct the scale *a posteriori* and not directly in the Bundle Adjustement like we propose. Strelow *et al.* [19] is the closest work to this paper. They present a batch algorithm based on Levenberg-Marquart and an online IEKF in order to fuse visual and inertial data. Here we propose an online visual and inertial SLAM based on Levenberg-Marquart.

The previous methods rely on the fact that measurement covariances are known *a priori*. But in the case that covariances are not available (no *a priori* on the noise), are fixed (*a priori* given by the manufacturer, or user defined) or are not accurate enough (only upper bound), the state-of-art methods (Local BA or EKF) will not handle that properly. It is also known that sensor covariances may change over time.

Recent work [18] proposes a method that fuse multi-sensor image data. They estimate the measurement noise for each group of sensor measurements using the Variance Component Estimation, and then perform BA to recover 3D object coordinates. They did not mention whether the method is real-time or not. Lavely *et al.* [13] proposes a multi-sensor fusion algorithm using BA for image registration and target recognition. It considers priors on measurement noise and estimates the system state covariance in a (joint) inverse problem with multiple camera sensors. In this paper we do not consider measurement noise to be known *a priori*. Estimating the noise variance ratio between the sensors is one of our goals.

We present a new generic approach to perform multi-sensor fusion within a BA based framework, that we call Bi-Objective BA, and show its application in SLAM. The problem is presented as a bi-objective least squares problem with a compound objective function, composed of two weighted error terms (from each of the two sensors). Since we consider no priors on measurement noises (at least for one sensor), it remains the problem of estimating the weight (or the ratio) between each objective. We investigate three distinct methods based on either trade-off criteria or machine learning to dynamically select an efficient weight, in real time. The methods can be used with any least squares constraints and not only in a multi-sensor fusion problem (for instance camera smoothing, etc).

Paper organization. Section 2 introduces SLAM and sensor fusion problems. We describe our new approach, a bi-objective BA based SLAM in section 3 and the Weighting Selection methods in section 4. Finally, the last section reports experimental results on two different multi-sensor fusion: with an odometer and a gyroscope.

Notation. Scalars are in italics (*e.g.* x), vectors in bold (*e.g.* $\mathbf{p}$), and matrices in sans-serif and calligraphic fonts (*e.g.* Σ, $\mathcal{P}$). $\mathcal{I}$ is the identity matrix. Homogeneous vector are written with a tilde (*e.g.* $\mathbf{x} = \Psi(\tilde{\mathbf{x}}) = \frac{1}{\tilde{x}_3}\begin{bmatrix}\tilde{x}_1 \\ \tilde{x}_2\end{bmatrix}$). We define the vector that stacks two vectors $\mathbf{x}$ and $\mathbf{y}$, by $[\mathbf{xy}] = [\mathbf{x}^\top \mathbf{y}^\top]^\top$.

2. Background

2.1. Simultaneous Localization And Mapping

The Simultaneous Localization and Mapping (SLAM) problem aims at recovering simultaneously the system location (pose) and the scene structure. The state vector $\mathbf{x} = [\mathbf{ps}]$ is thus composed of two elements: the system dynamics $\mathbf{p}$ and the sparse and static environment map $\mathbf{s}$.

The measurements delivered by sensors are affected by noise with (usually) unknown characteristics. It leads to corrupted system state estimates. To overcome this difficulty, we have to increase the number and the variety of sensors and properly merge all the measurements.

2.1.1 Inverse Problem

The sensor fusion problem can be formulated as an inverse problem, discretized in the time domain. The observation process performed by a sensor k is defined by:

$$\mathbf{z}_t^k = H_t^k(\mathbf{x}_t) + \mathbf{n}_t^k, \qquad (1)$$

where H_t^k is a projection function performed by sensor k at discrete time t. It relates a real object or information $\mathbf{x}_t$ (here, system location and environment map) to its observation vector $\mathbf{z}_t^k$. H_t^k includes the coordinate frame change from the world coordinate frame in which we express our system location and map, into the local sensor reference frame in which measurements are expressed. This can be estimated by a calibration procedure.

The error vector $\mathbf{n}_t^k$ is usually assumed to be zero mean normally distributed random variables and models the inaccuracy introduced during the measurement phases.

The aim of inverse problem is to recover the unknown real vector $\mathbf{x}_t$ from multiple observations.

For a dynamic system, we model its movement by

$$\mathbf{x}_t = F_t(\mathbf{x}_{t-1}) + \mathbf{w}_t. \tag{2}$$

F_t relates the previous system state to the new one and $\mathbf{w}_t$ is the error vector of the modeling process. Its entries consist of Gaussians with zero mean. Static systems are written without the t subscript , *e.g.* $\mathbf{x}$.

2.1.2 Recursive Bayesian Estimation

Statistic filtering consist of estimating the state of a dynamic system that best validates the uncertain measures and *a priori*, knowing that measurements can be noisy. We consider that the system state (like for all sensor measurements) is a random variables vector for which we want to find its pdf.

A recursive Bayesian estimator aims at finding the pdf, minimizing the posterior expected value of a cost function. The most common cost function is the Mean Squares Error (MSE), for which we want to minimize the square error between an estimator and its observations.

For a general sensor k, we define the MSE as

$$\varepsilon_k^2(\mathbf{x}_t, \mathbf{z}_t^k) = \frac{1}{n_t^k} \boldsymbol{\Delta}_t^{k\top} \left(\Sigma_t^k\right)^{-1} \boldsymbol{\Delta}_t^k \tag{3}$$

with $\boldsymbol{\Delta}_t^k$ is the innovation or measurement residual vector and n_t^k its length, such as

$$\boldsymbol{\Delta}_t^k = \mathbf{z}_t^k - H_t^k(\mathbf{x}_t). \tag{4}$$

When all measurements are considered as independent, with covariance $\Sigma_t^k = \sigma_t^k \mathcal{I}$, Equation 3 becomes

$$\varepsilon_k^2(\mathbf{x}_t, \mathbf{z}_t^k) = \frac{1}{n_t^k} \frac{1}{(\sigma_t^k)^2} \|\mathbf{z}_t^k - H_t^k(\mathbf{x}_t)\|^2. \tag{5}$$

Sensor fusion problem aims at minimizing all sensor MSE. An example of recursive Bayesian estimator is Bundle Adjustment.

2.2. Bundle Adjustment

Bundle Adjustment (BA) is an optimization technique based on non-linear least squares[20]. The objective function E to minimize is generally an image-based MSE: the reprojection error $E(\mathbf{x}_t, \mathbf{z}_t^c) = \varepsilon_c^2(\mathbf{x}_t, \mathbf{z}_t^c)$, which is the sum of the squared distances between 2D observations $\mathbf{z}_t^c$ and reprojections (see section 2.2.1).

Bundle Adjustment can be employed for different purposes, depending on the input variables we want to optimize. Usual BA are: pose refinement $\mathbf{p}_t$, scene structure-only $\mathbf{s}_t$, both scene and motion adjustment $[\mathbf{p}_t \mathbf{s}_t]$, or even for self-calibration where we refine some of the intrinsic camera parameters (focal, principal point, ...). For instance, in order to refine both scene structure and camera motion, we seek

$$\mathbf{x}_t^* = \underset{\mathbf{x}_t}{argmin}\, E(\mathbf{x}_t, \mathbf{z}_t^c). \tag{6}$$

This is a non-linear least squares problem (because of the Ψ function, Equation 7), which can be solved using a damped Gauss-Newton technique, such as Levenberg-Marquardt[20].

2.2.1 Camera Projections

The camera sensor c performs a projection of a subset of the scene $\mathbf{s}$ (in the field of view). With the pinhole camera model, the projection of a 3D feature j (located at $\mathbf{s}_t^j$) on image I_t is defined by

$$\mathbf{y}_t^j \sim H_t^c([\mathbf{p}_t \mathbf{s}_t^j]) = \Psi\left(\mathcal{P}_t \tilde{\mathbf{s}}_t^j\right) \tag{7}$$

where $\mathcal{P}_t$ is the projection matrix. This matrix can be decomposed in some metric coordinate frame as $\mathcal{P}_t = \mathcal{K}_t(\mathcal{R}_t|\mathbf{t}_t)$, where $\tilde{\mathbf{s}}_t^j$ is the homogeneous 3D observed feature and $(\mathcal{R}_t, \mathbf{t}_t)$ represents the orientation and position of the camera in a world coordinate frame. Intrinsic matrices $\mathcal{K}_i$ are considered equal, constant and known. For conciseness, we omit other distortions (radial etc.) in Equation (7).

For monocular BA-based SLAM, the system pose is only represented by its orientation (normalized quaternion $\mathbf{q}_t = R(\mathcal{R}_t)$) and position $\mathbf{t}_t$: $\mathbf{p}_t = [\mathbf{q}_t \mathbf{t}_t]$.

Since all 2D measurements of an image I_t come from the same frame time t, we can make a global camera measurement vector by stacking all 2D features in one vector ($\mathbf{z}_t^c = [\forall \mathbf{y}_t^j]$) and do the same for the reprojections $H_t^c(\mathbf{x}_t)$.

Bundle Adjustment aims at refining the camera pose and scene structure, minimizing Equation 3, with $k = c$.

3. Bi-Objective BA-based SLAM

Since BA is known to be more precise than Kalman filtering, our idea is use BA as a multi-sensor fusion technique. Problems arise when no *a priori* on the measurement noise are introduced since without covariance weighting, each of the sensor MSE does not share the same unit neither the same significance. We here propose a way to estimated this weight.

To be able to do fusion with BA, we write the cost function as a multi-objective least squares, within a compound cost function. The second sensor is used to make a constraint on the system displacement in a incremental SLAM of [16]. Thus, the cost function is composed of the classical reprojection errors (from camera sensor) and the sensor constraint. The weight of the constraint is automatically estimated using one of the three proposed methods. Instead of classical fusion algorithms, our technique (including the weight estimation) is periodically achieved and synchronized with the camera keyframe framerate.

3.1. System Overview

We resume our multi-sensor SLAM technique within the following steps, where **(*)** means a new step or a major change from the original Visual SLAM of [16].

Localization. (Resection)

1. Perform 2D/3D tracking of known features (points) at image I_t
2. Estimate a robust camera pose $\mathbf{p}_t^c$ from the tracking results (3-points algorithm+RANSAC,...).
3. **(*)** Predict the system pose $\hat{\mathbf{p}}_t^k$ from the second sensor measurements and compute the associated constraint ε_k (described in Section 3.3)
4. **(*)** Compute the constraint weight λ_k using one of the three selection methods (Section 4)
5. **(*)** Refine the current pose $\mathbf{p}_t$ with the sensor predicted pose constraint and selected weight, using the vision-based pose estimate $\mathbf{p}_t^c$ as initial solution (Section 3.4)

Map building. (Intersection & Refinement)

1. Perform a triangulation on all new features to recover their 3D position and add them into the current scene estimate $\mathbf{s}_t$.
2. **(*)** Perform a Constrained Local Bundle Adjustment with all weighted constraint previously computed (LBA) (Section 3.5)

3.2. Aggregate Objective Function

The multi-sensor fusion problem can be viewed as a multi-objective optimization: we want to find a solution that answer best to each of the objective requirements. The solutions space for which every objectives are optimal is not a unique point but is a hyper-surface called the *Pareto Frontier*. An easy way to be in the Pareto frontier is to write the problem as a weighted compound function with all the objectives, called the *Aggregate Objective Function*.

We start formulating the global MSE for a two sensors fusion, when all uncertainties are considered Gaussian and variables independent:

$$E(\mathbf{x}_t, \mathbf{Z}_t) = \frac{1}{(\sigma_t^c)^2 n_t^c}\|\mathbf{z}_t^c - H_t^c(\mathbf{x}_t)\|^2 + \frac{1}{(\sigma_t^k)^2 n_t^k}\|\mathbf{z}_t^k - H_t^k(\mathbf{x}_t)\|^2, \tag{8}$$

which can be rewritten as

$$E(\mathbf{x}_t, \mathbf{Z}_t) = \frac{1}{n_t^c}\|\mathbf{z}_t^c - H_t^c(\mathbf{x}_t)\|^2 + \lambda_t^2 \frac{1}{n_t^k}\|\mathbf{z}_t^k - H_t^k(\mathbf{x}_t)\|^2. \tag{9}$$

with $\lambda_t = \frac{\sigma_t^c}{\sigma_t^k}$.

When at least one of the variances σ_t^k is unknown or untrusted, arise the problem of finding a good weight λ_t. We propose different solutions to solve this weighting selection problem in section 4.

3.2.1 Solving the Compound BA

After estimating the constraint weight λ_t, we simply perform the classical Levenberg-Marquardt algorithm to solve the appropriate problem:

$$\min_{\mathbf{x}_t} E(\mathbf{x}_t, \mathbf{Z}_t) = \frac{1}{2}\mathbf{\Delta}_t^\top \, \Sigma_t^{-1} \, \mathbf{\Delta}_t \tag{10}$$

where $\mathbf{\Delta} = [\frac{\mathbf{\Delta}_t^c}{n_t^c} \frac{\mathbf{\Delta}_t^k}{n_t^k}]$ and $\Sigma_t = \mathcal{I}$.

Obviously, if a variance/covariance matrix for one of the sensor is known, we advice to replace the identity matrix by the variance/covariance matrix.

3.3. Sensor Prediction and Constraint

The second sensor constraint is formulated as a least squares cost. Since the sensors measurements are not synchronized, we need to predict the system pose (or a part of it) at the camera frame time t from the second sensor measurements $\mathbf{z}_t^k = \mathbf{z}_{t' \in]t-1,t]}^k$. Then we build the constraint from this prediction.

3.3.1 Pose Prediction from Second Sensor

We assume we have a prediction model (Equation 2) of the system dynamic, for instance, a random walk with constant translation and rotational speeds. Since our BA does not optimize velocities, these quantities have to be either back-estimated, or came directly from the second sensor. We also suppose to have the inverse observation model H_t^{-1} which provide an estimate of the system state $\mathbf{x}_{t'}^k$ at time $t' \leq t$, corresponding to the last measurement $\mathbf{z}_{t'}^k$ of sensor k. Thus, with the system prediction model, at time t we have the following predicted pose:

$$\hat{\mathbf{p}}_t^k = F_t(H_{t'}^{-1}(\mathbf{z}_{t'}^k - \mathbf{n}_{t'}^k)) + \mathbf{w}_t. \tag{11}$$

The model estimates a prediction $\hat{\mathbf{p}}_t^k$ of the system state at the time of keyframe t, from the second sensor measurements $\mathbf{z}_t^k$ and the previous system state $\mathbf{x}_{t-1}$. Here the error vectors $\mathbf{n}_t^k$ and $\mathbf{w}_t$ are supposed to be null. This prediction is then used to make a constraint on the system displacement.

3.3.2 Sensor Constraint

The constraint type is based on the nature of the sensor measurements. As an example, we propose some constraints for sensors that sense the system dynamic.

Position: $\varepsilon_k^t(\mathbf{x}_t, \mathbf{z}_t^k) = \frac{1}{\sqrt{3}}\|\hat{\mathbf{t}}_t^k - \mathbf{t}_t\|$.

Orientation: $\varepsilon_k^R(\mathbf{x}_t, \mathbf{z}_t^k) = \frac{1}{\sqrt{9}}\|\hat{\mathcal{R}}_t^k \mathcal{R}_t^\top - \mathcal{I}_3\|$.

Scale (relative translation norm): $\varepsilon_k^s(\mathbf{x}_t, \mathbf{z}_t^k) = \left|\|\hat{\mathbf{t}}_t^k - \mathbf{t}_{t-1}\| - \|\mathbf{t}_t - \mathbf{t}_{t-1}\|\right|$.

Each constraint will have a different impact on the system localization and will tend to correct the corresponding drift. Our method is generic, so the user defined constraint can be any least squares error term that constrains the optimized variables in BA.

3.4. Bi-Objective Pose Refinement

In the tracking process of an incremental SfM, when a new keyframe is detected a refinement is performed over the camera location.

We thus want to recover the current system pose $\mathbf{p}_t$ that best explain the sensors measurements $\mathbf{Z}_t = \left[\mathbf{z}_t^c \mathbf{z}_t^k\right]$. We build the aggregate cost function with the standard visual errors term ε_c^2, plus the selected constraint ε_k^2 based on the predicted pose given by sensor k. The second term is weighted with the previously selected λ_t (see section 4). Hence, the global MSE is

$$E_p(\mathbf{x}_t, \mathbf{Z}_t) = \varepsilon_c^2(\mathbf{x}_t, \mathbf{z}_t^c) + \lambda_t^2 \varepsilon_k^2(\mathbf{x}_t, \mathbf{z}_t^k) \tag{12}$$

The current pose in then optimized using the Levenberg-Marquardt algorithm with the compound cost function $E_p(\mathbf{x}_t, \mathbf{Z}_t)$, to solve:

$$\mathbf{p}_t^* = \underset{\mathbf{p}_t}{argmin}\, E_p(\mathbf{x}_t, \mathbf{Z}_t) \tag{13}$$

3.5. Constrained Local Bundle Adjustment

Periodically, a refinement is operated on both scene structure and system location to reduce the cumulated errors of the incremental SLAM. Good results can be obtained with a local optimization. As in [16], we choose to optimize the three last camera poses (keyframes) and reproject the scene observed in the last $N = 9$ cameras poses.

We need to constrain each of the optimized cameras. A simple way to do is to inject the previous constraints in the LBA cost function. We define $\mathbf{X}_t = [\mathbf{p}_{t-9} \dots \mathbf{p}_t \mathbf{s}_t]$, a vector comprising fixed poses $(\mathbf{p}_{t-9}, \dots, \mathbf{p}_{t-3})$, free camera poses $(\mathbf{p}_{t-2}, \mathbf{p}_{t-1}, \mathbf{p}_t)$ and the current scene estimate $(\mathbf{s}_t)$.

The Local Bundle Adjustment cost function with constraints becomes:

$$E_{BA}(\mathbf{X}_t, \mathbf{Z}_t) = \sum_{t'=t-9}^{t-3} \varepsilon_c^2([\mathbf{p}_{t'} \mathbf{s}_t], \mathbf{z}_{t'}^c) + \sum_{t'=t-2}^{t} E_p([\mathbf{p}_{t'} \mathbf{s}_t], \mathbf{Z}_t) \tag{14}$$

The first part of Equation 14 is the camera-fixed reprojection errors. There is no need to constrain fixed cameras here, so we take the original reprojection errors. The second part is composed of the cameras-constrained error functions for the three most recent camera poses, from Equation 12.

We then optimize the system location and scene structure using the Levenberg-Marquardt algorithm, and solve

$$[\mathbf{p}_{t-2} \mathbf{p}_{t-1} \mathbf{p}_t \mathbf{s}_t]^* = \underset{[\mathbf{p}_{t-2} \mathbf{p}_{t-1} \mathbf{p}_t \mathbf{s}_t]}{argmin} \quad E_{BA}(\mathbf{X}_t, \mathbf{Z}_t). \tag{15}$$

It is important to note that for relative constraints (between two camera poses), the pose predicted by the second sensor has to be updated at each iteration of the LM, with Equations 11.

4. Weight Selection

Here we investigate different solutions to solve the problem of determinating the weighting parameter λ of Equation 9. We propose three methods that inspect the behaviour of each of the two error parts. Our propositions are based on either trade-off criteria or a learning method.

4.1. Trade-off Criteria

L-Curve. The L-Curve is a criterion originally employed in regularization of ill-posed problems, known as Tikhonov regularization (see [8]). The original aim was to find automatically a good regularisation parameter. The method has also been used in multi-objective Least Squares optimization. In this case, we want to find a good compromise between multiple objectives, separated with one weighting parameter λ as in Equation 9.

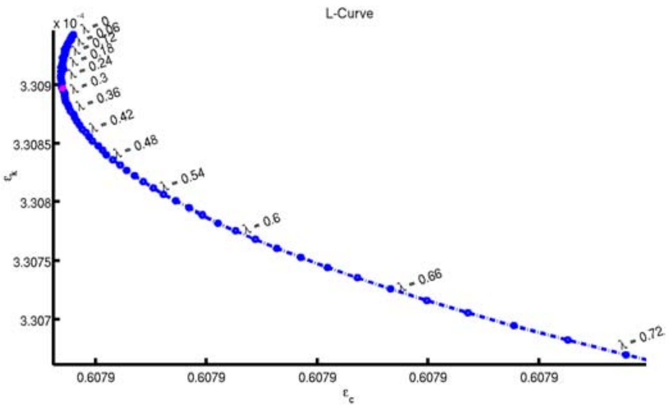

Figure 1. L-Curve criterion

The L-Curve criterion represents the trade-off between the two cost functions, in a *log-log* plot ($\hat{\rho} = \log \varepsilon_c(\mathbf{x}_t, \mathbf{z}_t^c), \hat{\nu} = \log \varepsilon_k(\mathbf{x}_t, \mathbf{z}_t^k)$) as shown in Figure 1. The trade-off usually has an L-shape, where a 'corner' separates the flat and vertical parts of the curve, where the solution of the multiple optimization is dominated by the first or the second error. The curvature $\kappa(\lambda)$ of the L-Curve can be used to find the corner

$$\kappa(\lambda) = 2 \frac{\hat{\rho}' \hat{\nu}'' - \hat{\rho}'' \hat{\nu}'}{(\hat{\rho}'^2 + \hat{\nu}'^2)^{\frac{3}{2}}} \tag{16}$$

where $'$ and $''$ denotes the first and second derivatives with respect to λ. The *good* weighting parameter is located in the corner, where the curvature is at maximum, hence

$$\lambda_{LCurve}^* = \underset{\lambda \in [0,1[}{argmax}\, \kappa(\lambda) \tag{17}$$

The method has some limitations. It sometimes appears (but rare in our case) multiple corners or 'smooth corner', where all solutions on the pareto frontier are quite equivalent. Moreover, second derivatives can be subject to instabilities.

L-Tangent Norm. *Brunet et al.* propose in [2] a different and empirical criterion based on the tangent information of the L-Curve. Instead of finding the corner of the L-Curve, the LTN propose to find the best weight for which a small variant of its value has the lowest impact in the trade-off between each of the two errors. The LTN optimal weight solution is

$$\lambda^*_{LTN} = \underset{\lambda \in [0,1[}{argmin}\, L(\lambda) \tag{18}$$

where

$$L(\lambda) = \|(\bar{\rho_\lambda}', \bar{\nu_\lambda}')\|_2^2 \tag{19}$$

and $\bar{\rho_\lambda}$ and $\bar{\nu_\lambda}$ are normalized errors.

This variant is cheaper to compute than the classical L-Curve corner and more stable (first derivatives only).

Computing L-Curve based criteria. These two criteria assume that we are able to compute multiple (n_λ) couples $(\varepsilon_c, \varepsilon_k)$ for differents values of λ. This means that for each value of λ we need to do a few iterations of a Gauss-Newton based algorithm on $E_p(\mathbf{x}_t, \mathbf{Z}_t)$. Since we target real-time data fusion, we nedd to choose a small value of n_λ, say $n_\lambda < 100$.

4.2. Learning Method

Cross Validation. The Cross Validation is a popular machine learning tool based on statistical considerations. An adequate model (here the weight) should predict well missing data values. More precisely, if we remove a few measurements from the data set and do the optimization (training phase), then the corresponding solution should predict well the missing observations (test phase).

It exists differents variants of Cross Validation, mainly depending on the training and test sets managment. In our case, the limited number of camera observations ($\sim$ 40 inliers among 200-300 points) enforce us to use a recycling variant, the *Leave-one-out* method. In this case, we build the test set using only one observation and recycle it in the next learning phases.

The $CV_{loo}(\lambda)$ score reflects the prediction error for a weight λ. It represents the mean difference between the real observation $\mathbf{y}_t^j$ of a point j and its predicted value $H_t^c([\mathbf{p}_t^{[j]}\mathbf{s}_t^j])$, where the pose $\mathbf{p}_t^{[j]}$ is learned without the point j.

$$CV_{loo}(\lambda) = \frac{1}{n}\sum_{j=1}^{n} H_t^c([\mathbf{p}_t^{[j]}\mathbf{s}_t^j]) - \mathbf{y}_t^j \tag{20}$$

with $\mathbf{p}_t^{[j]}$ is the solution of $\underset{\mathbf{p}_t}{argmin}\, E_p(\mathbf{x}_t, \mathbf{Z}_t^{[j]})$, from Equation 13.

The aim of Cross Validation is then to maximize the prediction quality of the weighting parameter, and so minimize its prediction errors:

$$\lambda^*_{CV} = \underset{\lambda}{argmin}\, CV_{loo}(\lambda) \tag{21}$$

In a related work [7], an approximation of the Cross Validation is used to smooth the camera trajectory in a monocular SLAM. Here we propose to use this criterion to find an efficient weight in a multi-sensor fusion context, and propose a way to constrain the Local Bundle Adjustment.

5. Experimental Results

We have validated our propositions with two real datasets integrating two distinct types of sensors, odometers and gyroscope, to highlight the genericity of the method.

5.1. Vision with Odometer

The first experiment is a system composed of a camera and an odometer embedded in a vehicle. The vehicle sequence cover roughly 400m in a real traffic conditions ($\sim$2500 images). The odometer sensor, positioned on one wheel, delivers the instant translational speed (m/s).

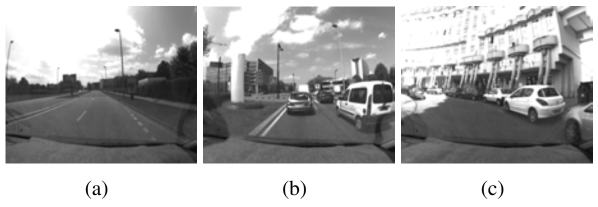

Figure 2. Images from the vehicle sequence.

The fusion was performed over reprojection errors (camera) and the scale factor sensed by the odometer (ε_k^s). Table 5.1 presents a comparison between the different methods for computing the weight. We present results where the choice of the keyframes was fixed or free (KF. Fixed). Each localization is compared to the ground truth, obtained with a high precision Inertial Navigation System (INS+GPS RTK). The mean distance of the localized cameras with the ground truth (Mean 3D RMS) is presented to validate our techniques.

A top view of the final localization is shown in Figure 5.1. The Ground truth (violet) was obtained with a Trajectometer (IMU+GPS RTK). The yellow result is the localization with the monocular only SLAM, one can observe a drift on the scale factor appearing around the middle of the sequence. The red result is the trajectory constrained with the odometer scale prediction. We see that the drift decreases as expected .

Selection Method	KF. Fixed	Mean 3D RMS (m)
Without fusion[16]	no	3.6704
L-Curve	no	**1.7884**
LTN	no	3.2723
Cross Validation	no	2.3963
L-Curve	yes	**1.792**
LTN	yes	2.9505
Cross Validation	yes	2.6207

Table 1. Statistics for the vehicle sequence (400m) with a scale factor constrained BA.

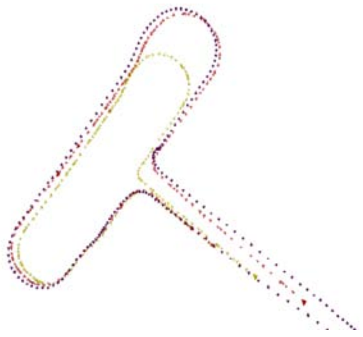

Figure 3. Top view of the localization of the 400m trajectory: (violet: ground truth, red: SLAM Trajectory with odometer and L-Curve, yellow: Visual only SLAM)

Figure 4, shows the 3D RMS of the position with the ground truth (after a global best fit), for different weight selection methods (vehicle sequence with L-Curve, LTN, CV) and without data fusion (red). The L-Curve method seems again to bring the best improvement.

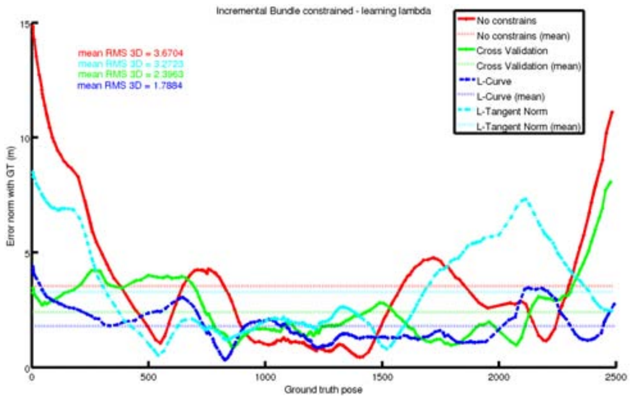

Figure 4. RMS 3D errors (ground truth) for the vehicle sequence, with different weight selection methods (blue: L-Curve, cyan: LTN, green: CV), and the monocular SLAM (red)

5.2. Vision with Gyroscope

We also test our BA based SLAM with a gyroscope integrated in a Inertial Measurement Unit (IMU). The hand-held system was carried by a pedestrian and the sequence is composed of 1400 images (640x480 px, figure 5) shot in an indoor environment with an 8m smooth trajectory[1]. Ground truth was acquired with a high precision laser tracker system.

(a) (b) (c)

Figure 5. Images from the indoor sequence.

We constrain the visual SLAM with the orientation given by the gyroscope (ε_k^R) and compare our localization results with a ground truth obtained with the laser tracker. We compare our methods with the kalman based SLAM of [12].

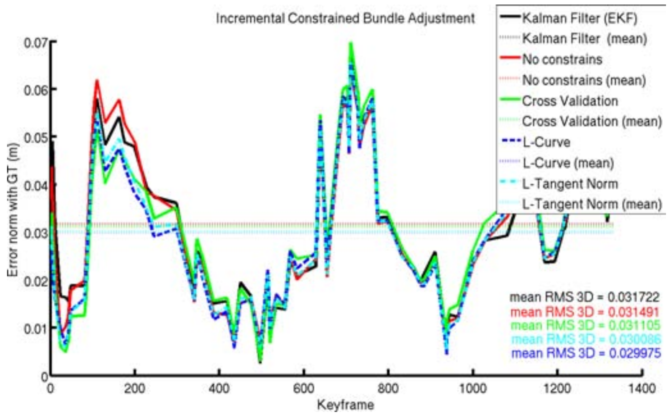

Figure 6. RMS 3D errors (ground truth) for the indoor sequence, with different weight selection methods (red: no, blue: L-Curve, cyan: LTN, green: CV, black : EKF)

Figure 6. In this sequence, the visual only SLAM [16] (red) yet reach an interesting accuracy on the localization (but up to a scale) thanks to the very good lightnings conditions and scene structure. After a global best fit, the means RMS 3D with the ground truth is 3.15cm. With an orientation constraint, we are able to increase slightly the localization accuracy (to 2.96cm with the L-Curve technique: gain of 6%). This improvement can be explained by the local error correction at the beginning of the sequence (red peacks). We also see that the weight selection methods are quite equivalent. We also see that EKF SLAM[12] is less accurate than other methods, even the visual only SLAM. We suggest that this result is mainly due to the approximative covariances estimated from the Local Bundle Adjustement.

[1]The dataset come from the GYROVIZ project.

6. Conclusion

This paper introduces a new technique for multi-sensor fusion based on an extended bundle adjustment. The method relies on previous work on data fusion with a weighting term, adapted by a learning method applied to bundle adjustment. This technique can also be applied for common experimental cases where one can not relies on covariances given by the sensors to perform classical data fusion *e.g.* GPS and Kalman filtering for instance.

Experimental results are presented in two different kind of applications. First, the classical bundle adjustment is considered for vehicle localization. Naturally, odometers presents on the car are considered for the fusion and the results show a significant improvement on the classical BA. Then, in order to highlight the genericity of the method, we have validated our method with a hand-held camera including an IMU. Even with a low cost MEMS IMU and an orientation only constraint, the fusion between the bundle adjustment and the orientations given by the IMU brings a slight improvement of the results.

These two experiments also show that the L-Curve based weighting method is more efficient than the other techniques and seems to be well adapted in this context.

Future work will address a generalization of this fusion problem for integrating multiple sensors.

References

[1] M. Agrawal and K. Konolige. Real-time localization in outdoor environments using stereo vision and inexpensive GPS. In *ICPR*, August 2006. 1

[2] F. Brunet, A. Bartoli, R. Malgouyres, and N. Navab. L-Tangent Norm: A Low Computational Cost Criterion for Choosing Regularization Weights and its Use for Range Surface Reconstruction. 6

[3] F. Chenavier and J. L. Crowley. Position estimation for a mobile robot using vision and odometry. In *IEEE International Conference on Robotics and Automation*, 1992. 1

[4] J. Civera, O. G. Grasa, A. J. Davison, and J. M. M. Montiel. 1-Point RANSAC for EKF-based Structure from Motion. In *IROS*, 2009. 1

[5] A. Cumani, S. Denasi, A. Guiducci, and G. Quaglia. Integrating monocular vision and odometry for SLAM. pages 625–30, 2003. 1, 2

[6] A. J. Davison. Real-time simultaneous localisation and mapping with a single camera. In *ICCV '03: Proceedings of the Ninth IEEE International Conference on Computer Vision*, page 1403, Washington, DC, USA, 2003. IEEE Computer Society. 1

[7] M. Farenzena, A. Bartoli, and Y. Mezouar. Efficient camera smoothing in sequential structure-from-motion using approximate cross-validation. In *ECCV (3)*, pages 196–209, 2008. 6

[8] P. C. Hansen. The L-curve and its use in the numerical treatment of inverse problems. In *Computational Inverse Problems in Electrocardiology*, number 5 in Advances in Computational Bioengineering, pages 119–142. WIT Press, Southampton, 2001. 5

[9] J. Hol, T. Schon, F. Gustafsson, and P. Slycke. Sensor fusion for augmented reality. In *Information Fusion, 2006 9th International Conference on*, pages 1–6, July 2006. 1

[10] A. Huster. *Relative position sensing by fusing monocular vision and inertial rate sensors*. PhD thesis, Stanford, CA, USA, 2003. Adviser-Rock, Stephen M. 1

[11] G. Klein and D. Murray. Parallel tracking and mapping for small AR workspaces. In *Proc. Sixth IEEE and ACM International Symposium on Mixed and Augmented Reality (ISMAR'07)*, Nara, Japan, November 2007. 1

[12] K. Konolige, M. Agrawal, and J. Solà. Large scale visual odometry for rough terrain. In *In Proc. International Symposium on Robotics Research*, 2007. 2, 7

[13] E. M. Lavely, M. Barmin, V. Kaufman, and E. Blasch. Model-based, multi-sensor fusion and bundle adjustment for image registration and target recognition. In *Society of Photo-Optical Instrumentation Engineers (SPIE) Conference Series*, 2006. 2

[14] M. Montemerlo, S. Thrun, D. Koller, and B. Wegbreit. Fastslam 2.0: An improved particle filtering algorithm for simultaneous localization and mapping that provably converges. In *In Proc. of the Int. Conf. on Artificial Intelligence (IJCAI*, pages 1151–1156, 2003. 1

[15] J. Montiel, J. Civera, and A. Davison. Unified inverse depth parametrization for monocular slam. In *Proceedings of Robotics: Science and Systems*, Philadelphia, USA, August 2006. 1

[16] E. Mouragnon, M. Lhuiller, M. Dhome, F. Dekeyser, and P. Sayd. Real time localization and 3d reconstruction. In *CVPR*, June 2006. 1, 3, 4, 5, 7

[17] T. Oskiper, Z. Zhu, S. Samarasekera, and R. Kumar. Visual odometry system using multiple stereo cameras and inertial measurement unit. *Computer Vision and Pattern Recognition, IEEE Computer Society Conference on*, 2007. 1

[18] D. Schneider and H.-G. Maas. Integrated bundle adjustment with variance component estimation – fusion of terrestrial laser scanner data, panoramic and central perspective image data. In *ISPRS Workshop on Laser Scanning*, 2007. 2

[19] D. W. Strelow and S. Singh. Motion estimation from image and inertial measurements. *The International Journal of Robotique Research*, 2004. 1, 2

[20] B. Triggs, P. F. McLauchlan, R. I. Hartley, and A. W. Fitzgibbon. Bundle adjustment - a modern synthesis. In *ICCV '99: Proceedings of the International Workshop on Vision Algorithms*, pages 298–372, London, UK, 2000. 1, 3