Single and Multi Camera Simultaneous Localization and Mapping Using the Extended Kalman Filter

On the Different Parameterizations for 3D Point Features

Simone Ceriani · Daniele Marzorati · Matteo Matteucci · Domenico G. Sorrenti

Received: 15 February 2012 / Accepted: 14 February 2013

This work has been partially supported by the Italian Ministry of University and Research (MIUR) through the PRIN 2009 grant "ROAMFREE: Robust Odometry Applying Multi-sensor Fusion to Reduce Estimation Errors".

S. Ceriani · M. Matteucci
Politecnico di Milano (DEIB), via Ponzio 34/5, 20133 Milano, Italy

S. Ceriani
e-mail: simone.ceriani@mail.polimi.it

M. Matteucci
e-mail: matteucci@elet.polimi.it

D. Marzorati
InfoSolution S.p.A., via della Burrona 51, 20090 Vimodrone (Mi), Italy
e-mail: d.marzorati@infosolution.it

D. G. Sorrenti (✉)
Università degli Studi di Milano - Bicocca (DISCo),
Building U14, viale Sarca 336, 20126 Milano, Italy
e-mail: domenico.sorrenti@unimib.it

1 Introduction

One of the most studied problems in Robotics deals with the navigation of a mobile robot in an unknown environment. It is nowadays clear that a truly autonomous agent needs the capacity to build and maintain its own representation of the working environment. This problem is called Simultaneous Localization And Mapping (SLAM). In SLAM, the robot uses the information obtained from its sensors to estimate simultaneously the environment map and its position in this map. To solve the SLAM problem became necessary after the observation that structuring the agent workspace failed in most practical situations. In SLAM, both the trajectory of the mobile robot and the position of all scene features, i.e., the map, are estimated online during the exploration. Pure localization can be seen as the case where the map is known, while pure mapping as the case where the observer pose is known, at all time instants.

Consider a mobile robot moving in a scene and activating its sensing apparatus at discrete time instants. Each sensor activation allows to measure some of the scene features,[1] in a reference frame relative to the robot. At each time t, we have the following relevant quantities: the pose of the robot, $\mathbf{\Gamma}_t$, which changes with time; the location $\mathbf{Y} = \{\mathbf{y}_i\}$ of scene features, i.e., the map, which are considered static;[2] the control $\mathbf{u}_t$ applied, at previous time, to move the robot to the current pose; the observations of the scene features $\mathbf{z}_t$, at the current time t. The SLAM problem can be recursively expressed in terms of the joint probability distribution of state $\mathbf{X}_t = [\mathbf{\Gamma}_t, \ \mathbf{Y}]$, as a function of the control inputs and the data collected by the observer sensing apparatus. Such probability $P(\mathbf{\Gamma}_t, \mathbf{Y}|\mathbf{z}_{1:t}, \mathbf{u}_{1:t}, \mathbf{\Gamma}_0)$ can be determined in a recursive form, by assuming the Markov property for the motion, and the conditional independence of same time measurements, given the current robot position.

The frequently used representation of this recursive formulation is in the form of a state-space model, with Gaussian noise. This representation allows the usage of the Kalman filtering, in its *extended* variant (EKF), to solve the SLAM problem. The need for the extended version of the Kalman filter comes from the non-linear nature of the dynamic part of the state transition relationship, with respect to the

[1] Here the term feature is left intentionally general; the case of 3D point features will be treated in details in the rest of the paper.

[2] In the case of a certain amount of features moving in the environment, tracking their movement while performing SLAM is the so called Simultaneous Localization and Mapping with Moving Object Tracking (SLAMMOT) problem [19, 23]; this case is not treated in this paper.

state variables, and from the non linearity in the measurement equation. Other representations use sigma points, in the Unscented Kalman Filter (UKF), or samples from the posterior distribution, in Particle Filters (PF), to perform, respectively, a better Gaussian uncertainty propagation or a non Gaussian representation of uncertainty [37]. Filtering approaches to visual SLAM are questioned by non linear optimization based approaches; see, e.g., [35] for the reasoning, and [18, 36] as examples of the good results that can be obtained with such approaches. In this paper we will focus on approaches based on the EKF, which is a well established technique, and it is known being able to deal with large scale environments, thanks to the adoption of sub-mapping, see e.g., [28].

In order to understand the peculiarities of the SLAM problem, it is important to notice that the uncertainty on the estimates of any two scene features, observed from the same pose, is not uncorrelated as one might expect, because the pose where the observations have been gathered is uncertain, too. After the robot moves to a new pose and observes again the scene, it might collect just a subset of the features previously observed. The estimates of the latter observed scene features, and also of the robot pose, can be updated by integrating the new measures; this in turn updates also the scene features that were seen previously. Moreover, the correlations between the estimates of the features grow with the subsequent observations. After some time, the features and the robot pose will be correlated [13].

The first successes in SLAM were initially obtained by systems exploiting mainly laser scanners [20], but some negative aspects of such devices (e.g., cost, weight, power consumption) pushed research towards less expensive, less power-hungry and less bulky sensors. Nowadays, the main sensors satisfying these criteria (low cost, power, and weight) are cameras. Although appreciable under these points of view, data from cameras suffer for a specific issue, which made visual SLAM (VSLAM) a research topic widely studied in recent years: the inherent impossibility to recover both the range and the bearing of a scene feature, when using a single image. The case of using one single camera is known as the Monocular SLAM problem. The case of using more than one camera, i.e., gathering measures on the same scene feature from more than one point of view, at the same time, it is known as Stereo, or multi-camera, SLAM problem.

A single camera actually allows quite accurate measurement of bearing, but it only allows an uniform uncertainty on the depth of the feature, being its position equally likely along the entire viewing ray. For this reason, in Monocular SLAM, the observer needs to gather data from different points of view, in order to develop a more peaked belief about the feature position. After the seminal work by Davison [11], we assisted to the diffusion of approaches trying to solve Monocular SLAM: subsequent works showed quite impressive results also in fields such as augmented reality [18] or 3D dense reconstruction [25].

The original work by Davison [11] used an EKF in combination with a delayed initialization of 3D features, representing 3D points with their Euclidean coordinates. Delayed initialization implies that a feature is added to the filter only after some parallax has been acquired, i.e., after the camera moved enough, following a parallax-providing trajectory. Unfortunately, delayed initialization makes data association difficult at the beginning, i.e., when the feature has been observed very few times, which is exactly when it is most important to get it correct. To overcome the shortcomings of delayed initialization, Solà [33] proposed an approach based on a

Mixture of Gaussians. However, this solution requires the presence of multiple depth hypotheses in the EKF state, and this affects the performance of the proposal.

Differently from these previous works, Montiel et al. [24] proposed the Unified Inverse Depth (UID) parametrization, the first instance of *inverse* parametrization of features; they demonstrated that it is possible to correctly represent the feature uncertainty using a single Gaussian, by changing the representation of the scene feature. This allowed un-delayed initialization of features, while allowing the usage of the well-known EKF machinery.

Nowadays, to the best of our knowledge, the problem of un-delayed feature initialization in EKF-based Monocular SLAM has four solutions, all related to how 3D points are represented in the filter state: UID [24], Inverse Scaling (IS) [22], Anchored Homogeneous Point (AHP) [31], and Framed Homogeneous Point (FHP) [7]. All these parametrizations allow to represent a depth uncertainty that is skewed and extends up to infinity [15], by using a standard Gaussian distribution.

The VSLAM literature presents different approaches for recursive Bayesian estimation: Extended Kalman Filtering [11, 24, 33], Unscented Kalman Filters [2, 8, 16] and Particle Filters [4, 30]); in this paper we focus on Extended Kalman Filtering. After a detailed presentation of EKF-based VSLAM in Section 2), we review in Section 3 the known parametrizations for EKF-based VSLAM, altogether with the novel Framed Inverse Depth (FID) parametrization. FID is an evolution of the FHP parametrization, whose requirements in terms of space per feature are in between AHP and FHP, while considering the case of shared anchors (see Section 3.6) it could often need less space. For consistency,[3] FID behaves similarly to the other four parametrizations.

The consistency of the parametrizations is verified in Section 4 with an extensive validation on a simulated environment, similar the one proposed by Solà in [31], together with results on real data from the RAWSEEDS datasets [6].

2 EKF SLAM with Cameras

In this section we focus on the use of EKF to solve the VSLAM problem, i.e., to recursively estimate, from visual measurements, the joint distribution of the current camera pose and the 3D points in the environment, as a single multivariate Gaussian distribution. We will briefly explain the basic techniques that allow to treat the EKF-VSLAM problem in real time, taking advantage of the structure of the problem. We will define the state vector, the prediction step equations, and the measurement equations in a general form; further details will be given in the next section.

Let us remark, here, two relevant concepts. Firstly, EKF-VSLAM was demonstrated able to treat the *Monocular SLAM* problem, i.e., to solve the VSLAM problem, using a single camera as source of information; this is of course true only up to a non observable scale factor [24]. This result can be obtained thanks to specific *parametrizations* (see Section 3) for scene point features, which allow to represent in an appropriate way the unknown depth of a point perceived from a single image.

[3]Consistency, a concept that will be reviewed in a subsequent section, represents the effectiveness of the SLAM system to correctly handle the inherent uncertainties; a consistent filter allows the real values of the unknowns to be in a certain range from their estimate.

If we are interested in retrieving also the scale of the observed scene we need to introduce some additional constraint in the process, e.g., a second camera as in the stereo case or some odometric information. Secondly, it was shown in [32] that the use of these parametrizations simplifies the implementation of multi-camera systems (e.g., a stereo or trinocular camera rig); we base on this approach when presenting Stereo SLAM.

2.1 State Vector

In Monocular EKF-SLAM, the state of the filter $\mathbf{X}_t$, at time t, is composed by the actual camera pose $\mathbf{\Gamma}_t$ in a world reference frame,[4] and by the map $\mathbf{Y}$, represented by a set of features $\mathbf{Y} = \{\mathbf{y}_i\}$. Each feature encodes a 3D point of the environment expressed with respect to the world reference frame, according to some representation, e.g., as an Euclidean point or by one of the *parametrizations* that will be presented in Section 3. The camera pose $\mathbf{\Gamma}_t = \left[\mathbf{t}_t^T, \mathbf{q}_t^T\right]^T$ includes a translation term $\mathbf{t}_t = \left[t_{x_t}\ t_{y_t}\ t_{z_t}\right]^T$ and an orientation term $\mathbf{q}_t = \left[q_{w_t}\ q_{x_t}\ q_{y_t}\ q_{z_t}\right]^T$, which is usually coded with quaternions.[5] Other parameters $\mathbf{\Lambda}_t$, describing camera motion (e.g., tangential and rotational speed or accelerations), could be present in the EKF state as well. They could be represented with respect to the camera reference frame or the world reference frame, in accordance with the *motion model* of the system.

The complete state of the filter is thus represented as:

$$\mathbf{X}_t = \left[\ \mathbf{\Gamma}_t^T\ \mathbf{\Lambda}_t^T\ \mathbf{y}_1^T\ \mathbf{y}_2^T\ \cdots\ \mathbf{y}_n^T\ \right]^T, \tag{1}$$

and the EKF-SLAM algorithm follows the steps described in Algorithm 1 to update recursively the state estimate, together with its covariance matrix $\mathbf{\Sigma}_t$. In the following, this algorithm will be explained in details.

2.2 Initialization

Unless we have some prior information, at $t = 0$ we start with an empty map and the camera pose is usually considered the origin of the world ($\mathbf{\Gamma}_0 = [0, 0, 0,\ 1, 0, 0, 0]^T$) with no uncertainty. If we have also a description of the camera motion, we can initialize these parameters to zero ($\mathbf{\Lambda}_0$), but with a non zero initial covariance, so to ease the estimation of the camera motion in the first steps.

$$\mathbf{X}_0 = [\mathbf{\Gamma}_0^T, \mathbf{\Lambda}_0^T]^T, \quad \mathbf{\Sigma}_0 = \begin{bmatrix} 0 & 0 \\ 0 & \mathbf{\Sigma}_{\mathbf{\Lambda}_0} \end{bmatrix}. \tag{2}$$

[4] In this paper we assume a *world-centric* approach to SLAM, where camera and features are represented with respect to a common world reference frame; in [21] the robocentric approach has proven able to reduce linearization errors, thus improving filter consistency at a higher computational cost. A comparison of the two approaches is out of the scope of this paper so we address the interested reader to [5, 21].

[5] Several representations for rotations are possible [14]; in this paper, being this marginal w.r.t. the main issue of feature representation, we use quaternions to allow comparison with visual EKF-SLAM systems publicly available. It has to be noticed that quaternions present several advantages, e.g., they allow linear interpolation of orientations.

Algorithm 1 EKF-SLAM

{Initialization}
$t \leftarrow 0, \mathbf{X}_t \leftarrow [\boldsymbol{\Gamma}_0^T \quad \boldsymbol{\Lambda}_0^T]^T, \boldsymbol{\Sigma}_t \leftarrow diag\left(\boldsymbol{\Sigma}_{\boldsymbol{\Gamma}_0}, \ \boldsymbol{\Sigma}_{\boldsymbol{\Lambda}_0}\right)$
loop
  {Prediction step}
  $t \leftarrow t + 1$
  $\hat{\mathbf{X}}_t \leftarrow f(\mathbf{X}_{t-1}, \mathbf{u}_t, \boldsymbol{\eta} = 0)$
  $\hat{\boldsymbol{\Sigma}}_t \leftarrow \mathbf{F}_{\mathbf{X}} \boldsymbol{\Sigma}_{t-1} \mathbf{F}_{\mathbf{X}}^T + \mathbf{F}_{\boldsymbol{\eta}} \boldsymbol{\Sigma}_{\eta} \mathbf{F}_{\boldsymbol{\eta}}^T$
  {Measurement step}
  $\mathbf{e} \leftarrow \varnothing, \mathbf{H} \leftarrow \varnothing, \mathbf{R} \leftarrow \varnothing$
  for all $\mathbf{y}_i$ **do**
    if $\exists \mathbf{z}_i$ compatible with $h_i(\hat{\mathbf{X}}_t)$ **then**
      $\mathbf{e} \leftarrow [\mathbf{e}; \ \mathbf{z}_i - h_i(\hat{\mathbf{X}}_t)]$
      $\mathbf{H} \leftarrow [\mathbf{H}; \ \mathbf{H}_{i\mathbf{X}}]$;
      $\mathbf{R} \leftarrow diag(\mathbf{R}, \ \mathbf{H}_{i\upsilon} \boldsymbol{\Sigma}_{\upsilon} \mathbf{H}_{i\upsilon})$
    end if
  end for
  {Update step}
  if $\mathbf{e} \neq \varnothing$ **then**
    $\mathbf{S}_t \leftarrow \mathbf{H} \hat{\boldsymbol{\Sigma}}_t \mathbf{H}^T + \mathbf{R}$
    $\mathbf{K}_t \leftarrow \hat{\boldsymbol{\Sigma}}_t \mathbf{H}^T \mathbf{S}_t^{-1}$
    $\mathbf{X}_t \leftarrow \hat{\mathbf{X}}_t + \mathbf{K}_t \mathbf{e}_t$
    $\boldsymbol{\Sigma}_t \leftarrow \hat{\boldsymbol{\Sigma}}_t - \mathbf{K}_t \mathbf{S}_t \mathbf{K}_t^T$
  else
    $\mathbf{X}_t \leftarrow \hat{\mathbf{X}}_t, \ \boldsymbol{\Sigma}_t \leftarrow \hat{\boldsymbol{\Sigma}}_t$
  end if
  {New feature addition}
  for all *new_measurement* **s do**
    $\mathbf{y}_{\text{new}} \leftarrow g(\mathbf{X}_t, \mathbf{s}, \boldsymbol{\xi})$
    $\mathbf{X}_t \leftarrow [\mathbf{X}_t^T \quad \mathbf{y}_{\text{new}}{}^T]^T$
    $\boldsymbol{\Sigma}_t \leftarrow \begin{bmatrix} I & 0 \\ \mathbf{G}_{\mathbf{X}} & 0 \end{bmatrix} \begin{bmatrix} \boldsymbol{\Sigma}_t & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} I & 0 \\ \mathbf{G}_{\mathbf{X}} & 0 \end{bmatrix}^T + \begin{bmatrix} 0 & 0 \\ 0 & \mathbf{G}_{\xi} \boldsymbol{\Sigma}_{\xi} \mathbf{G}_{\xi}^T \end{bmatrix}^T$
  end for
end loop

2.3 Prediction Step

At each iteration, we perform the prediction step of the Kalman filter; this step aims at the prediction of the next system state, using the prior state value and an optional control input $\mathbf{u}_t$; we assume a non additive Gaussian noise $\eta \sim N(0, \boldsymbol{\Sigma}_{\eta})$ perturbates the motion. To update the state covariance matrix we use the Jacobian matrices[6] $\mathbf{F}_{\mathbf{X}} = \partial f(\mathbf{X}, \mathbf{u}, \boldsymbol{\eta}) / \partial \mathbf{X}$ and $\mathbf{F}_{\eta} = \partial f(\mathbf{X}, \mathbf{u}, \boldsymbol{\eta}) / \partial \boldsymbol{\eta}$ evaluated at $\hat{\mathbf{X}}_t$, $\hat{\mathbf{u}}_t$, and $\boldsymbol{\eta}_t = 0$. Odometric information, if available, is considered as the control input $\mathbf{u}_t$, in

[6]It could be easily verified that, if noise η is additive to input $\mathbf{u}$ then $\frac{\partial f}{\partial \eta} = \frac{\partial f}{\partial \mathbf{u}}$.

the absence of odometry we would have a pure Visual EKF-SLAM, which uses the velocity motion model represented by $\mathbf{\Lambda}_t$:

$$\hat{\mathbf{X}}_t = f(\mathbf{X}_{t-1}, \mathbf{u}_t, \boldsymbol{\eta}_t = 0), \tag{3}$$

$$\hat{\mathbf{\Sigma}}_t = \mathbf{F}_\mathbf{X} \mathbf{\Sigma}_{t-1} \mathbf{F}_\mathbf{X}^T + \mathbf{F}_{\boldsymbol{\eta}} \mathbf{\Sigma}_\eta \mathbf{F}_{\boldsymbol{\eta}}^T. \tag{4}$$

Since features are fixed elements in the scene, the state prediction equation simplifies into:

$$\left[\hat{\mathbf{\Gamma}}_t^T,\ \hat{\mathbf{\Lambda}}_t^T\right]^T = f(\mathbf{\Gamma}_{t-1}, \mathbf{\Lambda}_{t-1}, \mathbf{u}_t, \eta_t = 0)$$

$$\hat{\mathbf{Y}}_t = \mathbf{Y}_{t-1}, \tag{5}$$

and the uncertainty propagation can exploit sparse Jacobians:

$$\mathbf{F}_\mathbf{X} = \begin{bmatrix} \dfrac{\partial f(\cdots)}{\partial(\mathbf{\Gamma}, \mathbf{\Lambda})} & 0 \\ 0 & \mathbf{I} \end{bmatrix} \qquad \mathbf{F}_\eta = \begin{bmatrix} \dfrac{\partial f(\cdots)}{\partial \boldsymbol{\eta}} \\ 0 \end{bmatrix}. \tag{6}$$

A reduction in the computational complexity of the prediction step, from $O(n^{2.807})$ to $O(n)$, is possible when taking into account this special structure; here n is the number of features. This is important when the map, and thus the state vector, becomes populated by a large number of features.

2.4 Feature Addition

In SLAM we aim at the online construction of the environment map, it is therefore necessary to increase the set of state variable each time a new landmark is perceived (see steps 27–31 of Algorithm 1). When a new feature $\mathbf{y}_{\text{new}}$ is created, its initialization is computed, by composing the current pose w.r.t. the world, which is part of the current state, with the 3D position of the new feature, in the camera frame. As for all measurements, uncertainties corrupt the measure, and they are especially intense in VSLAM. Besides the measurement error on the projected 2D point, which is perhaps minor, we have the large uncertainty in the 3D position, since the depth of point can not be perceived by a single camera, that is implied by the usage of a bearing-only sensor. If we call $\xi \sim N(0, \mathbf{\Sigma}_\xi)$ the Gaussian noise acting on the imaged point, and $\mathbf{s}$ the a priori expectation on the depth of a 3D feature, then the initial 3D value of the feature is a function of the current state, $\mathbf{s}$, and the noise. The noise is in general non additive.

$$\mathbf{y}_{\text{new}} = g(\mathbf{X}_t, \mathbf{s}, \xi = 0). \tag{7}$$

The new feature is then added to the filter

$$\mathbf{X}_t = [\mathbf{X}_t^T \quad \mathbf{y}_{\text{new}}{}^T]^T, \tag{8}$$

and its covariance updated by using its Jacobians, evaluated at $\mathbf{X}_t$ and $\xi = 0$

$$\mathbf{G}_\mathbf{X} = \frac{\partial g(\cdots)}{\partial \mathbf{X}}, \qquad \mathbf{G}_\xi = \frac{\partial g(\cdots)}{\partial \xi}, \tag{9}$$

$$\mathbf{\Sigma}_t = \begin{bmatrix} \mathbf{I} & 0 \\ \mathbf{G}_\mathbf{X} & 0 \end{bmatrix} \begin{bmatrix} \mathbf{\Sigma}_t & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} \mathbf{I} & 0 \\ \mathbf{G}_\mathbf{X} & 0 \end{bmatrix}^T + \begin{bmatrix} 0 & 0 \\ 0 & \mathbf{G}_\xi \mathbf{\Sigma}_\xi \mathbf{G}_\xi^T \end{bmatrix}^T. \tag{10}$$

2.5 Measurement

The standard equations of the update step in EKF-SLAM, can be considered consisting of two different stages: the *measurement step* and the *update step*. The measurement step (see steps 9–16 of Algorithm 1) allows, for each feature $\mathbf{y}_i$, to predict its expected measure, according to the measurement model $h_i(\mathbf{X}, \nu)$, being $\nu \sim N(0, \mathbf{\Sigma}_\nu)$ the model of the measurement noise.[7]

The total innovation $\mathbf{e}$ is computed as the difference between the expected measures and the real ones. It is determined by iteratively appending the difference between an expected measure $h_i(\hat{\mathbf{X}}_t)$ and its *individually compatible real measure* $\mathbf{z}_i$, whenever it exists:

$$\mathbf{e} \leftarrow [\mathbf{e};\ \mathbf{z}_i - h_i(\hat{\mathbf{X}}_t)]. \tag{11}$$

The individual compatibility of measurements estimation is usually verified by a Mahalanobis distance test on:

$$d = (\mathbf{z}_i - h_i(\hat{\mathbf{X}}_t))^T \mathbf{S}_i^{-1} (\mathbf{z}_i - h_i(\hat{\mathbf{X}}_t)), \tag{12}$$

with

$$\mathbf{S}_i = \mathbf{H}_{i\mathbf{X}} \hat{\mathbf{\Sigma}}_t \mathbf{H}_{i\mathbf{X}}^T + \mathbf{H}_{i\nu} \mathbf{\Sigma}_\nu \mathbf{H}_{i\nu}^T, \tag{13}$$

where

$$\mathbf{H}_{i\mathbf{X}} = \frac{\partial h_i(\cdots)}{\partial \mathbf{X}}, \qquad \mathbf{H}_{i\nu} = \frac{\partial h_i(\cdots)}{\partial \xi}, \tag{14}$$

are the Jacobian matrices of $h_i(\cdot)$ with respect to the state and noise.

As for the innovation vector, Jacobian matrices are properly stacked to $\mathbf{H}$ and to the block diagonal matrix $\mathbf{R}$:

$$\mathbf{H} = [\mathbf{H};\ \mathbf{H}_{i\mathbf{X}}], \tag{15}$$

$$\mathbf{R} = diag(\mathbf{R},\ \mathbf{H}_{i\nu} \mathbf{\Sigma}_\nu \mathbf{H}_{i\nu}^T). \tag{16}$$

2.6 Update Step

Thanks to the measurement step, a set of observations is created and the innovation is stored in the $\mathbf{e}$ vector, with the matrices $\mathbf{H}$ and $\mathbf{R}$. The Kalman gain is computed as

$$\mathbf{S}_t = \mathbf{H}_{\hat{\mathbf{X}}_t} \hat{\mathbf{\Sigma}}_t \mathbf{H}_{\hat{\mathbf{X}}_t}^T + \mathbf{R}, \tag{17}$$

$$\mathbf{K}_t = \hat{\mathbf{\Sigma}}_t \mathbf{H}_{\hat{\mathbf{X}}_t}^T \mathbf{S}_t^{-1}, \tag{18}$$

and the filter state is updated as

$$\mathbf{X}_t = \hat{\mathbf{X}}_t + \mathbf{K}_t \mathbf{e}_t, \tag{19}$$

$$\mathbf{\Sigma}_t = \hat{\mathbf{\Sigma}}_t - \mathbf{K}_t \mathbf{S}_t \mathbf{K}_t^T. \tag{20}$$

[7] The independency of the measurement process is guaranteed under the hypothesis of independence between the noises ν_i and ν_j, affecting each pair of measurement equations $h_i(\cdot)$ and $h_j(\cdot)$ with $i \neq j$.

Notice that, in case there are no measurements available (i.e., the innovation vector is empty), the update step is skipped, and the predicted state is assumed as the new state.

There might be alternative ways to perform the EKF update. For instance, it is possible to iterate the update step in the following way: (1) sort the $\mathbf{e}$ vector in decreasing order of innovation (evaluated with Eq. 12); (2) integrate one measure at a time, recomputing the Jacobians at each step. This procedure increases the complexity of the update step, but provides a better linearization of Jacobians. Notice that, in practical cases, the straightforward methods above may fail, in the presence of outliers, e.g., wrong data associations. With real data, the update step has to be complemented by a robust data association procedure.

Different techniques have been developed to this extent, in particular one of the simplest and most effective is the 1-Point RANSAC technique presented in [10], which executes two update steps on two disjoint sets of measurements characterized, respectively, by low and high innovation, discarding possible outliers.

2.7 Remarks on Filter Sparseness

In Section 2.3 it was shown that the structure of the motion model equations allows to simplify the computation of the new state and covariance matrix. It has to be noticed that also in the feature addition, in the measurement, and in the update steps, there is space for some simplifications, which are not immediately obvious.

In the feature addition step we are creating a new feature, and this is done through the $g(\cdot)$ function in Eq. 7, and using the $\mathbf{s}$ and the current state; the $\mathbf{s}$ represents the "a priori" for a new feature in the sensor reference frame. The aim of this step is to compute the position of the new feature in world coordinates, i.e., to combine $\mathbf{s}$ and the actual pose $\mathbf{\Gamma}_t$, i.e., the already existing features $\mathbf{y}_i$ are independent on $\mathbf{s}$ and $\mathbf{\Gamma}_t$. This means that:

$$\mathbf{y}_{\text{new}} = g(\mathbf{X}_t, \mathbf{s}, \xi) = g(\mathbf{\Gamma}_t, \mathbf{s}, \xi), \tag{21}$$

and therefore the Jacobian matrix $\mathbf{G}_{\mathbf{X}}$ differs from zero only with respect to Γ:

$$\mathbf{G}_{\mathbf{X}} = \left[\frac{\partial g(\cdots)}{\partial \mathbf{\Gamma}} \; 0 \cdots 0 \right]. \tag{22}$$

As for the prediction step, the complexity of a feature addition is linear in the number of features in the map. Furthermore, each measurement function $h_i(\mathbf{X})$ actually depends only on the sensor pose $\mathbf{\Gamma}$ and on the i-th observed feature $\mathbf{y}_i$; it follows that we have a sparse Jacobian:

$$\mathbf{H}_{i\mathbf{X}} = \frac{\partial h_i(\cdots)}{\partial \mathbf{X}} = \left[\frac{\partial h_i(\cdots)}{\partial \mathbf{\Gamma}} \quad 0 \cdots 0 \quad \frac{\partial h_i(\cdots)}{\partial \mathbf{y}_i} \quad 0 \cdots 0 \right], \tag{23}$$

and the stacked matrix $\mathbf{H}$ is also sparse:

$$\begin{bmatrix} \frac{\partial h_1(\cdots)}{\partial \mathbf{\Gamma}} & \frac{\partial h_1(\cdots)}{\partial \mathbf{y}_1} & 0 & 0 & 0 \\ \frac{\partial h_2(\cdots)}{\partial \mathbf{\Gamma}} & 0 & \frac{\partial h_2(\cdots)}{\partial \mathbf{y}_2} & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ \frac{\partial h_i(\cdots)}{\partial \mathbf{\Gamma}} & 0 & 0 & \frac{\partial h_i(\cdots)}{\partial \mathbf{y}_i} & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ \frac{\partial h_m(\cdots)}{\partial \mathbf{\Gamma}} & 0 & 0 & 0 & \frac{\partial h_m(\cdots)}{\partial \mathbf{y}_m} \end{bmatrix}. \tag{24}$$

Since only few features are observed at each time, the computation of the Kalman gain is speeded up both by the sparsity of the Jacobians and by the reduced dimension of the measurement vector.

2.8 Perspective Camera Model and Measurements

In VSLAM the measurement model states that when a 3D point is perceived by a camera, only the direction of its viewing ray is measured from its image, and its depth remains unknown, i.e., all the points that lie on the viewing ray are projected to the same pixel of the image (see Fig. 1). This influences both the feature addition and the measurement steps.

Let us consider a 3D point parametrized by its Euclidean coordinates in a reference frame centered in the camera optical center $\mathbf{P}^C = [X, Y, Z]^T$; we can define $\mathbf{d}^C = [X/Z, Y/Z, 1]^T$ as the direction of the viewing ray from $\mathbf{P}^C$ to the camera optical center. We can observe that the first two elements of $\mathbf{d}^C$ are the intersection of the viewing ray with the xy plane of a reference system translated of one unit in the z axis, from the camera optical center, i.e., they are the coordinates of a point on a normalized image plane with focal distance equal to 1. We denote with $\pi_{2D} : \mathbb{R}^3 \rightarrow \mathbb{R}^2$, $\pi_{2D}([X, Y, Z]) = [X/Z, Y/Z]$ the function that performs this normalization.

To model the geometry of the image formation for a real camera, i.e., the process that allows to calculate where on the image a 3D point is perceived, we need two other steps: (1) the application of a function that models distortion due to lens; (2)

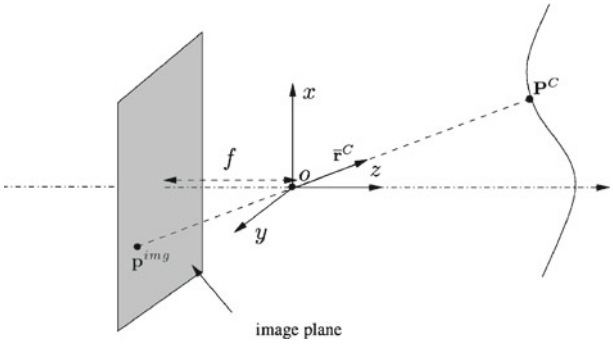


Fig. 1 The figure shows an ideal projection, i.e., a projection through ideal optics

the projection on the image in pixel coordinate. The first operation is expressed by a distortion function $dist(\mathbf{p}_\pi, \mathbf{k}) : \mathbb{R}^2 \to \mathbb{R}^2$ that maps the 2D point $\mathbf{p}_\pi$ on the normalized image plane to a distorted plane using the coefficient $\mathbf{k} = [k_1, k_2]^T$; this is quite a widespread distortion model, although others models could be used for the same purpose.

$$\mathbf{p}_d = dist(\mathbf{p}_\pi, \mathbf{k}) = \mathbf{p}_\pi \left(1 + \mathbf{k}^T \cdot \left[\|\mathbf{p}_\pi\|^2 \ \|\mathbf{p}_\pi\|^4\right]^T\right). \tag{25}$$

The second operation is expressed by the camera projection model $K(\mathbf{p}_d, \mathbf{f}, \mathbf{c}) : \mathbb{R}^2 \to \mathbb{R}^2$ that projects a 2D point from the distorted plane to the image coordinate, given the focal length of the camera $\mathbf{f} = [f_x, f_y]^T$, and the center of the image $\mathbf{c} = [c_x, c_y]^T$:[8]

$$K(\mathbf{p}_d, \mathbf{f}, \mathbf{c}) = diag(\mathbf{f}) \cdot \mathbf{p}_d + \mathbf{c}. \tag{26}$$

A 3D point is therefore projected to a point on the image plane by

$$\begin{bmatrix} u \\ v \end{bmatrix} = \mathbf{p}^{img} = K(dist(\pi_{2D}(\mathbf{P}^C), \mathbf{k}), \mathbf{f}, \mathbf{c}). \tag{27}$$

Conversely, when we have a point on the image plane $\mathbf{p}^{img}$, we can calculate its position on the undistorted normalized plane by inverting the previous formula:

$$\mathbf{p}_\pi = dist^{-1}(K^{-1}(\mathbf{p}^{img}, \mathbf{f}, \mathbf{c}), \mathbf{k}). \tag{28}$$

We can also define the viewing ray as $\mathbf{r}^C = \left[\mathbf{p}_\pi^T \ 1\right]^T$ and the unit vector $\bar{\mathbf{r}}^C = \mathbf{r}^C / \|\mathbf{r}^C\|$.

When a point P is referenced to the world reference frame W ($\mathbf{P}^W$), and we know the camera pose $\mathbf{\Gamma} = \left[\mathbf{t}, \mathbf{q}\right]$ in the same reference frame, we can compute the projection in the image by first referring the point to the camera reference frame C

$$\mathbf{P}^C = \mathbf{R}^T(\mathbf{q}) \left(\mathbf{P}^W - \mathbf{t}^W\right), \tag{29}$$

where $\mathbf{R}(\cdot)$ is the function that computes a 3×3 rotation matrix from the quaternion representation of the orientation of the camera frame. We can then apply the projection Eq. 27.

We can now present a general form of the measurement equations in Visual EKF SLAM: to measure a landmark $\mathbf{y}_i$ in the camera pose $\mathbf{\Gamma}_t$, we apply Eq. 27 to a function $feature^{3D}$ that calculates the position of the feature in the camera reference frame as a 3D point:

$$\begin{bmatrix} u \\ v \end{bmatrix} = K(dist(\pi_{2D}(feature^{3D}(\mathbf{\Gamma}_t, \mathbf{y}_i)), \mathbf{k}), \mathbf{f}, \mathbf{c}) + v. \tag{30}$$

The $feature^{3D}$ function is strictly dependent on the chosen parametrization of features. Details on this issue are given in the next section, where different parametrizations are presented. Notice that we consider the zero mean Gaussian noise

[8] Usually the projection is expressed in homogeneous coordinate as $\left[u \ v \ 1\right]^T = \mathbf{K}\left[\mathbf{p}_d^T \ 1\right]^T$, with $\mathbf{K} = \begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix}$. Here we prefer to express $\mathbf{K}$ as a function, to allow the usage of possibly different camera projection models (e.g., catadioptric cameras).

ν to be additive, in the measurement equation. A common choice is to consider $\Sigma_{\nu} = diag(1, 1)$, i.e., the measurement process is affected by a noise of a 1 pixel standard deviation, and whose components are independent.

Similarly, a general form for the feature addition function could be presented as:

$$\mathbf{y}_{\text{new}} = g(\mathbf{\Gamma}_t, \mathbf{s} = \left[\mathbf{p}^{img^T}, d\right]^T, \xi), \tag{31}$$

i.e., the inputs are represented by the pixel coordinate of the new point and a parameter d that acts as "initial depth" of the point along the viewing ray. We can also consider that, in a general formulation, ξ will be a 3-elements noise: ξ_1, ξ_2 will be additive noise on the pixel coordinate with independent covariance of 1 pixel, to model the error in the detection of the point, while the third element will model the uncertainty on the depth. Details depend on the specific parametrization and are shown in the next section.

So far, we have used the camera reference frame in the notation, but all equations could be easily modified to consider any reference frame rigidly connected to it; this is, for instance, the case of a mobile robot, where we know the camera pose in the robot odometric frame, or the case of a stereo camera, where we know the pose of one camera with respect to the other. Starting from this observation it is rather easy to see that multi-camera EKF-SLAM system represents a natural extension of the Monocular EKF-SLAM.

Let us consider a multi-camera system with J cameras, and denote with $\mathbf{\Gamma}_j$ the fixed rototranslation of the j-th camera with respect to one common reference frame $\mathbf{\Gamma}_t$ that could be either one of the cameras or the odometric center of the robot. The motion model of this reference frame remains the same of Eq. 3 whereas the measurement equations for each camera are slightly different from the Monocular case in Eq. 30; when the feature $\mathbf{y}_i$ is perceived by the j-th camera we apply

$$\begin{bmatrix} u \\ v \end{bmatrix} = K^j(dist^j(\pi_{2D}(feature^{3D}(\mathbf{\Gamma}_t \oplus \mathbf{\Gamma}_j, \mathbf{y}_i)), \mathbf{k}^j), \mathbf{f}^j, \mathbf{c}^j) + \nu^j, \tag{32}$$

being $\mathbf{\Gamma}_t \oplus \mathbf{\Gamma}_j$ the composition of the rototranslation of the odometric reference of the multi camera rig at time t, w.r.t. the world reference frame, with the rototranslation of the j-th camera w.r.t. the odometric center of the rig. In the case of a stereo rig, we could consider the optical center of one of the cameras as the odometric center; this is what was originally proposed by Solà in [32] under the name of BiCamSLAM.

Implementing a multi camera system as the fusion of many single camera systems, which share the same state of the environment, has several advantages. It can exploit the information coming from the observation of features perceived by just one single camera, being triangulation not required; of course, for such cases, the uncertainty will reduce only after some parallax will have been gained, i.e., by means of the observer motion. Moreover, and most importantly, we can leverage on the parametrizations used for Monocular SLAM to reduce the feature uncertainty, at the same time, as if we had moved. We have to point out that triangulation is done implicitly each time a feature is observed by more than one camera. If new, a feature is initialized from its view from the first camera where it appears; if it has been already observed, it is just updated as presented before. The multi-camera advantage comes when the feature is also observed by another camera(s). In such a case, its expected measure is predicted in the other camera(s); if the data association produces a valid result, the feature gets also measured from there, i.e., from the place

where the other camera is, therefore the uncertainty on the depth is strongly reduced. The uncertainty on the depth, modeled by the chosen parametrization, will allow to implicitly treat the epipolar geometry in searching for feature correspondences. In the following section we review the state of the art in feature representation, i.e., the different ways to express the $g(\cdot)$ and $feature^{3D}(\cdot)$ function, in the monocular case, while the extensions to stereo or multi-camera are easily derivable.

3 Point Feature Parametrizations in 3D

The simplest way to represent a 3D point is by means of its Euclidean coordinates: $\mathbf{y}^E = [XYZ]^T$. However, in the context of EKF, Euclidean points can be used directly only in a stereo setting and only with small depth-to-baseline ratio, i.e., only when the uncertainty on their reconstruction is reasonably approximated by a normal distribution. In other words, being impossible to estimate their depth from a single image, they are not suited for Monocular SLAM, unless we accept some delay in their initialization, as proposed in [11], i.e., to observe the same point feature in more than one image, with different camera poses, so that an appropriate parallax is gained and the uncertainty on the feature turns reasonably normal. The problem being that the potentially uniform distribution of probability on the depth is not directly representable as a Gaussian variable. This intrinsic limit of Euclidean points motivated proposals of different parametrizations suited for undelayed initialization [7, 22, 24, 33]; in the following, we review these proposals and present a new one, underlining their computational complexity and non linearity.

3.1 Unified Inverse Depth (UID)

In the Unified Inverse Depth parameterization proposed by Montiel et al. in [24], a 3D scene point $\mathbf{y}^{UID}$ (see Fig. 2) is defined by the vector

$$\mathbf{y}^{UID} = \left[\mathbf{t}^T \ \vartheta \ \varphi \ \varrho\right]^T, \tag{33}$$

which represents a 3D point located at $\mathbf{t} + (1/\varrho) \cdot \mathbf{m}(\vartheta, \varphi)$. Here $\mathbf{t}^T$ is the camera projection center when the 3D point was first observed; ϑ and φ are respectively the

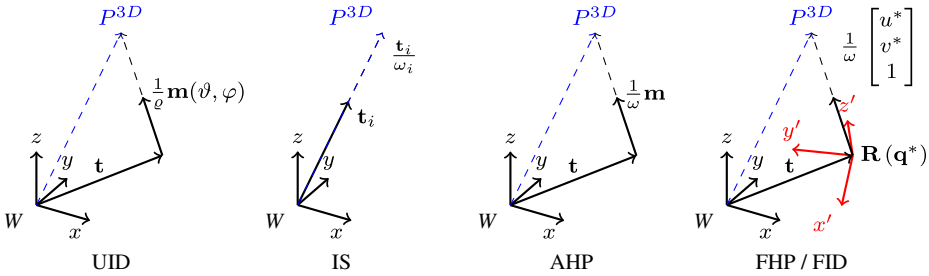


Fig. 2 The geometrical meaning of the different parameterization for a 3D point

azimuth and the elevation (in a world-aligned reference frame) of the line $\mathbf{m}(\vartheta, \varphi) = [\cos(\varphi)\sin(\vartheta), -\sin(\varphi), \cos(\varphi)\cos(\vartheta)]^T$, and $\varrho = 1/d$ is the inverse of the point depth along this line (see [24] for more details). The initialization is performed by:

$$\mathbf{y}_{\text{new}}^{UID} = g^{UID}(\mathbf{\Gamma}_t, \mathbf{s} = \left[\mathbf{p}^{img}, \varrho_0\right]^T, \xi) = \left[\mathbf{t}_t^T,\ \theta(\mathbf{r}^W),\ \phi(\mathbf{r}^W),\ \varrho_0\right]^T, \tag{34}$$

where

$$\mathbf{r}^W = \mathbf{R}(\mathbf{q}_t) \cdot \mathbf{r}^C, \tag{35}$$

is the viewing ray of $\mathbf{p}^{img}$ in a frame that is oriented as the world reference frame, $\mathbf{R}(\mathbf{q})$ converts the quaternion $\mathbf{q}$ (camera orientation) to a 3×3 rotation matrix, and

$$\theta(\mathbf{r}) = \text{atan2}(\mathbf{r}_x, \mathbf{r}_z);\ \phi(\mathbf{r}) = \text{atan2}(-\mathbf{r}_y, \sqrt{\mathbf{r}_x^2 + \mathbf{r}_z^2}). \tag{36}$$

The noise ξ is a 3-elements zero mean Gaussian variable. The first two elements represent additive noise on the $\mathbf{p}^{img}$, while the third element represents the uncertainty on the initial inverse depth value ϱ_0. The covariance matrix is $\mathbf{\Sigma}_\xi = diag(1, 1, \sigma_{\varrho_0})$. The values of ϱ_0 and σ_{ϱ_0} are chosen according to the minimum and maximum depth of the feature. The advantages of inverse depth resides mainly in the ability to include the infinity ($\varrho = 0$) without loosing the ability to code near points. It has to be noticed that $\vartheta(\cdot)$ and $\varphi(\cdot)$ are applied to the viewing ray represented in a world aligned frame, i.e., to the composition of the current orientation of the robot with the direction of the viewing ray in the robot frame.

To define a measurement equation we have only to specify the $feature^{3D}(\cdot)$ function of Eq. 30, i.e., we have to express the 3D point in the current robot frame; after some manipulation the $feature^{3D}(\cdot)$ turns to:

$$feature_{UID}^{3D}(\mathbf{\Gamma}_t, \mathbf{y}_t) = \varrho\,(\mathbf{t} - \mathbf{t}_t) + \mathbf{m}(\vartheta, \varphi). \tag{37}$$

where $\mathbf{t}$ is the part of the feature state that represents the camera position when the point was observed the first time.

While an Euclidean point requires three parameters, the UID representation requires the storage of six parameters for each map feature as state vector in the filter. It is possible to convert UID points into Euclidean 3D points once their estimation is sufficiently accurate, as suggested in [9] to reduce the computation time, and is relevant as EKF complexity is $O(n^2)$, with n being the size of the state variable. This holds for the other parametrizations too.

3.2 Inverse Scaling (IS)

In [22] the Inverse Scaling parametrization, a.k.a. Homogeneous Point (HP) in [31], was presented; the IS name came to the authors because the point is obtained by scaling the triangle with vertex in the projection centre, the focal length and the line from the image centre to the image of the point. It is also the homogeneous representation of a 3D point (see Fig. 2)

$$\mathbf{y}^{IS} = \left[\mathbf{t}^T\ \omega\right]^T. \tag{38}$$

The transformation of a point from IS to Euclidean coordinates is simply $\mathbf{t}/\omega$, while the initialization of a feature could be performed in two ways:

$$\mathbf{y}_{\text{new}}^{IS_N} = g^{IS_N}(\mathbf{\Gamma}_t, \mathbf{s} = \left[\mathbf{p}^{img}, \omega_0\right]^T, \xi) = \left[\left(\bar{\mathbf{r}}^W + \omega_0 \mathbf{t}_t\right)^T \omega_0\right]^T, \tag{39}$$

$$\mathbf{y}_{\text{new}}^{IS_S} = g^{IS_S}(\mathbf{\Gamma}_t, \mathbf{s} = \left[\mathbf{p}^{img}, \omega_0\right]^T, \xi) = \left[\left(\mathbf{r}^W + \omega_0 \|\mathbf{r}^C\| \mathbf{t}_t\right)^T \omega_0 \|\mathbf{r}^C\|\right]^T, \tag{40}$$

where $\bar{\mathbf{r}}^W = \mathbf{R}(\mathbf{q}_t) \cdot \bar{\mathbf{r}}^C$ is the unit vector of the viewing ray rotated in the world reference frame. Notice that $\|\mathbf{r}^W\| = \|\mathbf{r}^C\|$ because of the module preserving property of the rotation operation. The difference between the two initialization relies on the use of the viewing ray $\mathbf{r}^W$ or the unitary viewing ray $\bar{\mathbf{r}}^W$. Both initializations grant the initial distribution of the depth uncertainty to be comparable to the UID initialization with $\omega_0 = \varrho_0$ and the points lie on a sphere at $1/\omega_0$ from the optical center.

The $feature^{3D}$ part of the measurement equation for IS turns into the following:

$$feature_{IS}^{3D}(\mathbf{\Gamma}_t, \mathbf{y}_t) = \mathbf{R}(\mathbf{q}_t)^T (\mathbf{t} - \omega \mathbf{t}_t). \tag{41}$$

IS requires only four parameters to represent a 3D point, thus reducing the time complexity of the EKF. This speed up, however, comes at the cost of consistency as demonstrated in [7, 31].

3.3 Anchored Homogeneous Point (AHP)

The Anchored Homogeneous Point (AHP) parametrization was proposed in [31], and combines aspects of the UID and the IS [22] parametrizations, to overcome the consistency issues of IS. In AHP, see Fig. 2, a point is represented by a seven elements vector; AHP takes from UID the idea of representing the Euclidean position of the optical center at the initialization time, and from IS the use of homogeneous coordinates to represent a 3D point, i.e., a four elements vector encoding the viewing ray direction and the inverse distance. An AHP point is hence coded as:

$$\mathbf{y}^{AHP} = \left[\mathbf{t}^T \ \mathbf{m}^T \ \omega\right]^T, \tag{42}$$

where $\mathbf{t}$ is the position of the camera, and $[\mathbf{m} \quad \omega]^T$ is the 3D point in homogeneous coordinates, where $\mathbf{m}$ is a vector applied in the camera pose pointing towards the feature, and ω is the inverse of the scale of $\mathbf{m}$. The transformation into a 3D point is described by the sum of two vectors $\mathbf{t} + \mathbf{m}/\omega$.

AHP initialization could be performed in two ways:

$$\mathbf{y}_{\text{new}}^{AHP_N} = g^{AHP_N}(\mathbf{\Gamma}_t, \mathbf{s} = \left[\mathbf{p}^{img}, \omega_0\right]^T, \xi) = \left[\mathbf{t}_t^T \ \bar{\mathbf{r}}^{W^T} \ \omega_0\right]^T, \tag{43}$$

$$\mathbf{y}_{\text{new}}^{AHP_S} = g^{AHP_S}(\mathbf{\Gamma}_t, \mathbf{s} = \left[\mathbf{p}^{img}, \omega_0\right]^T, \xi) = \left[\mathbf{t}_t^T \ \mathbf{r}^{W^T} \ \omega_0 \cdot \|\mathbf{r}^C\|\right]^T. \tag{44}$$

With AHP_N we mean the use of a unit module viewing ray $\bar{\mathbf{r}}^{W^T}$ for initialization; with AHP_S the initial inverse scaling value is scaled by the norm of the viewing ray. As for IS parametrizations, also in this case both initializations grant the initial distribution of the depth uncertainty to be comparable to the UID initialization with $\omega_0 = \varrho_0$ and the points lie on a sphere at $1/\omega_0$ from the optical center.

The $feature_{AHP}^{3D}(\cdot)$ function in measurement Eq. 30 becomes:

$$feature_{AHP}^{3D}(\mathbf{\Gamma}_t, \mathbf{y}_t) = \mathbf{R}(\mathbf{q}_t)^T (\omega \cdot (\mathbf{t} - \mathbf{t}_t) + \mathbf{m}). \tag{45}$$

3.4 Framed Homogeneous Point (FHP)

In [7] the FHP parametrization has been presented; FHP extends the anchor point used in AHP and UID to the complete frame, i.e., the position and orientation of the camera when the feature was initialized (see Fig. 2). Since the camera frame is completely specified, we can represent the viewing ray $\mathbf{r}^C$ using only its first two elements, as the third is 1 in the camera frame; the last parameter is again the inverse distance along the viewing ray. The parametrization results in a ten elements vector:

$$\mathbf{y}^{FHP} = \left[\mathbf{t}^T \ \mathbf{q}^T \ \mathbf{p}_\pi^T \ \omega\right]^T, \tag{46}$$

where $\mathbf{t}$ and $\mathbf{q}$ are the camera position and orientation when the feature was initialized, $\mathbf{p}_\pi$ the first two elements of a viewing ray in camera coordinates, i.e., the coordinate on the normalized undistorted image plane, while ω is the inverse scaling factor. The transformation into a 3D point is:

$$\mathbf{t} + \frac{1}{\omega}\,\mathbf{R}(\mathbf{q}^*) \cdot \begin{bmatrix} \mathbf{p}_\pi \\ 1 \end{bmatrix}, \tag{47}$$

being $\mathbf{q}^* = \mathbf{q}/\|\mathbf{q}\|$ the normalized quaternion. This normalization is needed because the EKF update does not guarantee quaternions to remain normalized. The initialization is:

$$\begin{aligned} \mathbf{y}_{\text{new}}^{FHP} &= g^{FHP}(\mathbf{\Gamma}_t, \mathbf{s} = \left[\mathbf{p}^{img}, \omega_0\right]^T, \xi) \\ &= \left[\mathbf{t}_t^T, \mathbf{q}_t^T, dist^{-1}(K^{-1}(\mathbf{p}^{img})), \omega_0 \cdot \|\mathbf{r}^C\|\right]^T, \end{aligned} \tag{48}$$

where the parameters, but $\mathbf{p}^{img}$, in $dist(\cdot)$ and $K(\cdot)$ are skipped in order to compact the notation. Notice that, as in Eq. 44, we scale the initial inverse depth value to guarantee that the initial distribution of the depth uncertainty is comparable to the UID initialization. This was not done in [7], so the FHP results presented here are not directly comparable to the ones presented in the original publication.

From the analysis of the Eq. 48 we observe that, differently from UID and AHP, FHP sports a more linear initialization: the elements that are already in the state (the camera pose $\mathbf{t}_t$ and $\mathbf{q}_t$) are simply copied to the new element, while the pixel coordinates of the new feature and its initial depth are put into the respective elements without any nonlinear operation involving state variables. This implies that the viewing ray and the initial depth are initially uncorrelated with the rest of the filter state, and no information on the camera pose when the feature was initially perceived is lost due to linearization. The $feature^{3D}(\cdot)$ function in measurement Eq. 30 becomes:

$$feature_{FHP}^{3D}(\mathbf{\Gamma}_t, \mathbf{y}_t) = \mathbf{R}(\mathbf{q}_t)^T \left(\omega\,(\mathbf{t} - \mathbf{t}_t) + \mathbf{R}(\mathbf{q}_i^*) \begin{bmatrix} \mathbf{p}_\pi \\ 1 \end{bmatrix}\right). \tag{49}$$

3.5 Framed Inverse Depth (FID)

Analyzing the FHP parametrization, we can observe that the element $\mathbf{p}_\pi$ represent, in the filter state, the initial pixel coordinates on the normalized image plane. Under the hypothesis of such point being accurately measured, we can assume the initial point on the normalized plane not to change too much during the updates, and approximate it with its initial value. Such hypothesis implies a properly calibrated

camera, a relatively high image resolution, and a low noise intensity, and it will never be perfectly valid. On the other hand, this implies that we can reduce the FHP parametrization to the following eight elements vector, giving a significant saving in space, and making FID a significant trade-off of accuracy and time-complexity, as for EKF savings in space means savings in time-complexity.

$$\mathbf{y}^{FID} = \left[\mathbf{t}^T \; \mathbf{q}^T \; \omega\right]^T, \tag{50}$$

In FID the image point $\mathbf{p}_{img}$, which allows to recompute $\mathbf{p}_\pi$, is stored outside the filter state. Initialization is done in a very similar way to FHP, sharing the benefits that come from the linearity of simply putting new values into the new variables without combining them. The FID initialization becomes thus

$$\mathbf{y}^{FID}_{\text{new}} = g^{FID}(\mathbf{\Gamma}_t, \mathbf{s} = [\omega_0], \xi) = \left[\mathbf{t}_t^T, \mathbf{q}_t^T, \omega_0 \cdot \|\mathbf{r}^C\|\right]^T. \tag{51}$$

Notice that, in this case, the input vector $\mathbf{s}$ contains only one element, the usual inverse depth initial value. Similarly, the ξ noise is unidimensional and represents only the initial variance of the inverse depth σ^2_ω.

Since FID parametrization does not re-estimate the initial point on the normalized image plane, we could model this source of uncertainty by an additional noise in the measurements equation. Thus, the $feature^{3D}(\cdot)$ function in Eq. 30 could be:

$$feature^{3D}_{FID}(\mathbf{\Gamma}_t, \mathbf{y}_t) = \mathbf{R}(\mathbf{q}_t)^T \left(\omega\,(\mathbf{t} - \mathbf{t}_t) + \mathbf{R}(\mathbf{q}^*)\begin{bmatrix}\mathbf{p}^*_\pi \\ 1\end{bmatrix}\right), \tag{52}$$

where

$$\mathbf{p}^*_\pi = dist^{-1}\left(K^{-1}(\mathbf{p}_{img} + \nu_{3,4})\right), \tag{53}$$

is the viewing ray, which takes into account the additional zero mean noise $\nu_{3,4}$ on the initial point (i.e., in this case the standard additive noise ν is enlarged with two elements that take in account some noise on the measurement at feature creation).

To be precise, the difference between the true and the initially estimated $\mathbf{p}_{img}$ is a *bias* and not a zero mean Gaussian noise. This $\nu_{3,4}$ noise performs a sort of covariance inflation, to reduce the small bias implied by the FID trade-off, which might be preventing the SLAM algorithm to perform proper data association and measurements. Since this might be controversial, the experimental evaluation will be conducted with and without such noise, resulting in two different parametrizations (FID_0, i.e., no extra noise, and FID_1, with extra noise).

As for feature initialization, the variance of the $\nu_{3,4}$ Gaussian variable is considered as a 2×2 identity matrix, in order to represent the 1px uncertainty (i.e. the complete covariance matrix of ν is a 4×4 identity matrix). This parametrization appears very similar to the one presented in [26], which differs in the representation of the feature orientation part.

3.6 Remarks on 3D Point Parametrizations

The advantages of an anchored parametrization are mainly due to feature initialization. When a new feature is initialized, uncertainty in camera position becomes correlated to the anchor point, while the uncertainties in feature measurement and representation are taken into account in the viewing ray description. Consequently, when an update is performed, these correlations, together with a more detailed

measurement equation, allow to distribute the errors in a more effective way. This implies that the more details are dropped in the parametrization, the worst are the results in tracking the feature position.

This is the main reason that explains why inverse scaling parametrization performs worse than all others parametrization, as shown in [31, 34]. In UID and AHP parametrizations, the anchor point codes the position of the camera when the feature is initialized, while the viewing ray carries mixed information about the attitude of the camera, when the feature was initialized, and the viewing ray in the camera frame. FHP and FID parametrizations have the advantage of keeping these sources of information separated.

The previous reasoning is better explained in the following comparison. Let us consider the initialization of a landmark in the different parametrizations (Eqs. 34, 39, 43, 44 and 48). UID, AHP, and IS parametrizations are initialized by a non linear function (i.e., the rotation of the viewing ray in the world reference frame implies a linearization), while FHP and FID parametrizations have a linear initialization. This implies that when a new feature is initialized, no information on uncertainty is dropped due to linearization in the first three parametrizations. Moreover, we believe that the continuous estimation of the viewing ray in world reference frame (in AHP and UID) or initial image point (in FHP) could have negative effects on the consistency of the filter, due to the fact that errors in the pose of the camera when the feature was initialized could be erroneously compensated by changes in the viewing ray. In FID this cannot happen since the initial image point is not re-estimated not being in the filter.

When several features are initialized using the same frame they all share the same anchor point (i.e., UID and AHP) or frame (i.e., FHP and FID). This means that several rows of the covariance matrix are equal making it rank deficient. Besides being an issue from the computational point of view, this also introduces singularities in the covariance matrix, which makes it not invertible. In our experimental activity this sharing is required by the use of Conditional Independent Submapping [27, 28]). To solve this issue, when adding more features from the same position they should share the common anchor, point or frame, depending on the parametrization; this is similar to what is done in [17], for UID, and in [26] whose proposed solution is very similar to FID. Sharing the anchor part of the feature gives great advantages in terms of the dimension of the state vector, and, consequently, on the computational complexity, especially for FID parametrization. Let us consider to initialize n features in the same frame; we can notice that, for $n > 2$, FID representation is more compact than UID, since $7 + n < 3 + 3n$. Obviously, this is not valid for the FHP parametrization that has the same dimension of UID for the non shared part and a bigger shared part due to the usage of an anchor frame instead of an anchor vector. On the other hand, FHP state space requirement is smaller than AHP, for $n > 4$, since $7 + 3n < 3 + 4n$.

As a final remark, we notice how, by storing the full pose of the camera when the feature was perceived the first time, the FID and FHP parametrizations implement some sort of key-framed representation: every time a feature gets updated by the Kalman filter, past camera poses get refined as well. This seems somehow violating the EKF assumption of the current pose being a summary of the whole camera trajectory (i.e., past poses are marginalized out) in favour of an optimization-based approach. What is happening here is that, by using a framed parametrization, the EKF-based Monocular SLAM implements a recursive filtering/smoothing of selected past camera poses (i.e., key-frames) and 3D features.

Because of the use of framed parametrizations, Monocular EKF-SLAM becomes somehow similar to non-EKF based methods such as FrameSLAM [1] or Graph-SLAM [37]. Indeed, all of them marginalize out the features into a graph of poses, which is then optimized by some conventional optimization technique. By using the FID or FHP parametrizations, this marginalization is not complete, since we keep the inverse depth of the feature in the filter state, and it is implicitly obtained by the EKF machinery.

4 Experimental Results

In this section, we present some numerical results showing the behaviour of the previously presented parametrizations for Visual EKF-SLAM. Our experiments have been conducted on both simulated and real data using MoonSLAM, a C++ framework for EKF-SLAM. The simulated data have been obtained with the setup proposed in [31], while real data are taken from the datasets of the RAWSEEDS[9] project [6]. Notice that, in our experiments on real data, we run a SLAM system in world coordinates, and not a Visual Odometry system (fixed-size map) robocentric, i.e., in the observer reference frame, as done in [31].

4.1 Evaluation on Simulated Data

In the experiments on simulated data, we are interested in evaluating the average Normalized Estimation Error Squared (NEES) [3] during the entire robot trajectory, as in [31]. To evaluate the average NEES, we performed a Montecarlo simulation of $N = 25$ trials with a nominal robot motion $\mathbf{\Gamma}^*_{0:T}$. For each trial, MoonSLAM estimates the trajectory $\mathbf{\Gamma}^n_{0:T}$ by logging at each time t the robot pose $\mathbf{\Gamma}^n_t$ with its associated covariance $\mathbf{\Sigma}^n_t$. The average NEES value for each time t is defined by:

$$\bar{\epsilon}_t = \frac{1}{N} \sum_{n=1}^{N=25} (\mathbf{\Gamma}^n_t - \mathbf{\Gamma}^*_t)^T (\mathbf{\Sigma}^n_t)^{-1} (\mathbf{\Gamma}^n_t - \mathbf{\Gamma}^*_t). \tag{54}$$

Trials differ from each other because of additional Gaussian noise added both to the prediction step, performed using nominal motion, and the measurement step, performed measuring the projection of landmarks in the image plane. Consistency is evaluated by checking the 95 % confidence interval of $\bar{\epsilon}_t$, which is distributed as a $\chi^2_{25\times 6}$ random variable.[10] The filter is considered too optimistic (i.e., errors in camera pose are underestimated) if the NEES value is above the upper bound of the confidence interval (H), while it is conservative when it is below the lower bound (L) of the confidence interval.

The implemented algorithm for the experiments performs an active-search-based SLAM [12] both in the monocular and stereo case. It initializes ten landmarks in the first frame, and then, for each frame, only a subset of the most innovative

[9] http://www.rawseeds.org

[10] In the computation of NEES, quaternions are converted in Euler angles and uncertainty is propagated through this transformation to compute $\mathbf{\Sigma}_{\mu}$ in Eq. 54. This is done to avoid the inversion of a rank deficient covariance matrix.

measurements (10 for the monocular case and 15 for the stereo case) is used in the iterative update step; the update follows the alternative procedure sketched in Section 2.6. At most one new landmark per step is added (taking them from the left camera when stereo is used), if less then 36 landmarks are predicted to be visible in the current frame. Inconsistent (i.e., landmarks with negative inverse depth) and unstable landmarks (i.e., landmarks not matched for at least 50 % of the times they have been projected on the image plane) are removed from the map. The cameras are pointing in the direction of motion and their position with respect to the robot reference frame is known. Image resolution is 640×480 pixels, artificial distortion with coefficients $[0.1, 0.1]^T$ is applied and the field of view is 90°; in the stereo case cameras have a distance of 20 cm on the horizontal axis.

4.2 Monocular Simulated Experiments

We performed ten experiments differing in camera motion, environment setup, initial depth of the feature in the initialization step, and noise acting on the motion. Parameters for all these experiments are summarized in the first part of Table 1 and the most significant are reported also in Fig. 4. Table 2 reports the percentage of time in which the filter operates in the consistent range and in the optimistic range (the remaining percentage represents the conservative behaviour); Table 3 reports the value of $\frac{\int \bar{\epsilon}_t - H}{T_{\epsilon_t > H}}$, where $T_{\epsilon_t > H}$ is the number of frames in which the filter operates in optimistic range. This value represents the average inconsistency of the filter; the higher the value the more the average NEES is far from consistency.

All proposed parametrizations, together with the different initialization schemes, have been evaluated (e.g. AHP_S and AHP_N). In particular, for the FID parametrization we consider two cases: in one case the standard deviation of the additional

Table 1 Experiments setup for the Monocular and Stereo simulated environment

	#	Motion						Noises		Initialization	
		ΔX [m]	ΔY [m]	ΔZ [m]	$\Delta\Theta$ [°]	$\Delta\Phi$ [°]	$\Delta\Psi$ [°]	ΔX [mm]	$\Delta\Psi$ [°]	ρ [m^{-1}]	σ [m^{-1}]
Monocular	1	0.08	0	0	0	0	0.9	2.5	0.025	1	1
	2									0.1	0.5
	3							1.25	0.0125	1	1
	4									0.1	0.5
	5	0.04	0	0	0	0	0.45	2.5	0.025	1	1
	6									0.1	0.5
	7							5	0.05	1	1
	8									0.1	0.5
	9	0.08	0.02	−0.02	0.2	−0.45	0.9	1.25	0.0125	1	1
	10									0.1	0.5
Stereo	1	0.08	0	0	0	0	0.9	2.5	0.025	0.5	1
	2							5.0	0.05		
	3	0.16	0	0	0	0	1.8	2.5	0.025	0.5	1
	4									0.01	0.5
	5							5.0	0.05	0.5	1
	6									0.01	0.5

Experiments differ in motion, noise added on motion, and feature inverse depth/scale initialization

Table 2 Monocular simulated experimental average NEES results

#		IS_N	IS_S	UID	AHP_N	AHP_S	FHP	FID_0	FID_1
1	Cons %	1	0	12	15	**17**	**17**	9	86
	Opt %	99	100	88	84	**83**	**83**	91	10
2	Cons %	0	0	88	89	88	86	**93**	15
	Opt %	100	100	12	11	12	14	**0**	0
3	Cons %	2	2	24	25	36	40	**66**	76
	Opt %	98	98	76	75	64	60	**34**	0
4	Cons %	3	3	98	**99**	**99**	**99**	91	63
	Opt %	97	97	2	**1**	**1**	**1**	7	0
5	Cons %	2	2	42	**46**	**46**	42	14	46
	Opt %	98	98	58	**54**	**54**	57	86	0
6	Cons %	0	0	**82**	80	78	78	36	25
	Opt %	100	100	**17**	19	21	21	64	0
7	Cons %	0	0	4	**5**	**5**	**5**	4	38
	Opt %	100	100	96	**95**	**95**	**95**	96	61
8	Cons %	0	0	3	3	3	3	**6**	84
	Opt %	100	100	97	97	97	97	**94**	16
9	Cons %	0	0	**3**	**3**	**3**	**3**	2	49
	Opt %	100	100	**97**	**97**	**97**	**97**	98	51
10	Cons %	2	1	90	96	94	97	**98**	19
	Opt %	98	99	8	**2**	6	**2**	**2**	0

The percentage of time in which each parametrization operates in consistent (Cons) and optimistic (Opt) range is reported. Best results (i.e., maximum consistency and minimum optimistic percentage) are reported in *bold*. FID_1 parametrization, i.e., with non-zero covariance on $v_{3,4}$, is reported in the last column and it is not compared with others

noise $v_{3,4}$ equals to a 2×2 identity matrix (as hypothesized in Section 3.5), in the second case this noise is set to zero. The need for this double setup comes from the fact that NEES evaluation takes advantage from the inflated covariance used in the first case, and therefore, to obtain a fair comparison between the parametrizations,

Table 3 Monocular simulated experimental results

#	IS_N	IS_S	UID	AHP_N	AHP_S	FHP	FID_0	FID_1
1	550.0	595.2	4.4	4.3	4.2	**4.1**	5.5	0.2
2	90.9	89.2	0.4	0.4	0.3	0.3	**0.1**	–
3	222.6	238.7	3.5	3.6	2.5	2.4	**0.5**	–
4	53.1	50.1	0.2	0.2	**0.1**	**0.1**	0.3	–
5	301.4	280.8	2.1	1.3	**1.2**	1.4	2.9	–
6	84.7	86.4	**0.3**	**0.3**	0.4	0.4	0.9	–
7	1455.9	1417.4	15.5	15.9	15.1	14.8	**14.1**	1.3
8	400.1	402.5	**9.2**	10.2	9.9	9.8	11.8	0.5
9	470.6	504.4	11.5	11.6	13.0	13.3	**4.6**	0.9
10	43.1	38.2	0.2	0.2	0.3	0.2	**0.1**	–

We report here $\frac{\int \bar{\epsilon}_t - H}{T_{\epsilon_t > H}}$, i.e., the average value of inconsistency for time step when the filter operate in optimistic range. Lower (i.e., better) values are in *bold*. FID_1 parametrization, i.e, with non-zero covariance on $v_{3,4}$, is reported in the last column and it is not compared with others. The "–" symbol means that the average values on the inconsistency is not calculable because the parametrization never operates in optimistic range, i.e., $T_{\epsilon_t > H} = 0$

we have to neglect it. These two cases are referred in Tables 2 and 3 respectively as FID_1 and FID_0. The latter will be compared, in the following, with the other parametrizations, while the results of the former will not be considered (although they are reported in the tables). In all the experiments IS shows poor performance in NEES, as already shown in [7, 31, 34], i.e., it almost never operates in conservative range so its results will not be discussed in the following. We recall that the results for FHP here presented differ from the ones presented in [7] because of a different initialization schema, as already discussed in Section 3.4.

Two different environments are provided: in the first setup, used for experiments 1 to 8, 72 points are distributed in a cloister-like environment, as in the setup originally introduced in [31]; the cloister is a square 12×12 m, and the points are in two different planes, respectively at the height $z = 0$ and $z = 1$ m (a screenshot of the MoonSLAM framework at work in the simulated cloister is shown in Fig. 3). In the second setup, used for experiment 9 and 10, 216 points are distributed in a similar environment, still covering a 12×12 m area, but in 6 different planes equally spaced at heights from $z = -5$ to $z = 5$, and the motion is in 6DoF. The feature depth is initialized at two different values. In the first, we use 1 m^{-1} for the mean of the inverse distance (or scale) and 1 m^{-1} for its standard deviation. In the second case 0.01 m^{-1} is used for the mean (i.e., feature very far from the observer) and 0.5 m^{-1} for its standard deviation.

In Experiments 1 and 2 the camera moves for 800 steps on a plane with increments of 8 cm on the X axis and a rotation of 0.9° around the Z axis per step, performing two loops in the cloister. Experiment 1 feature initialization is 1 m, while experiment 2 feature initialization is 100m. The noise added to the motion is the same for both experiments: standard deviation of 2.5 mm on the three axis for translation and of 0.025° around the three axis for rotation. Experiment 1 shows that all the considered parametrizations perform poorly, since they work in consistent range for a very small amount of time. Here AHP_S and FHP obtain the best result (i.e., the maximum

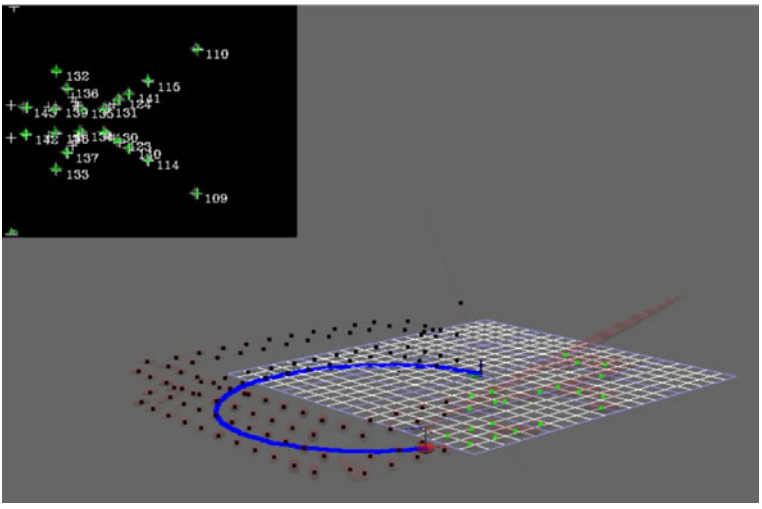
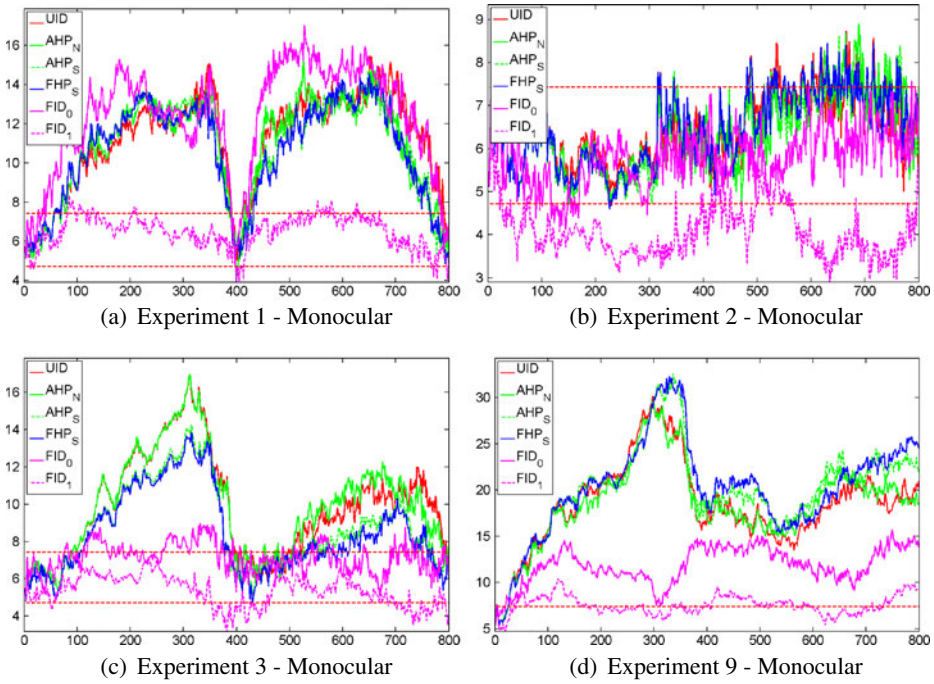


Fig. 3 A screenshot of MoonSLAM framework working in the simulated cloister

consistency and the minimum optimistic rate). The mean value of inconsistency confirms this result. Table 3 confirms that, apart from IS, all the parametrizations perform substantially equal in this case. In Experiment 2, FID_0 is the most consistent (it is never optimistic) and the mean of inconsistency is the lowest, although others parametrizations performs quite well, too. The results of Experiments 1 and 2 are shown in Fig. 4a and b. In Experiments 3 (Fig. 4c) and 4 we run the two previous experiments halving all noises. In Experiment 3 FID_0 parametrization performs definitely better than others: it operates in a consistent way for the 66 % of time while others stop at 40 % for FHP and less for the others. The mean value of inconsistency in FID_0 for Experiment 3 is about 5 time smaller than for the others. In Experiment 4 all the parametrization perform consistently, but in this case FID_0 performs a little bit worse than the others.

Experiments 5 and 6 replicate Experiments 1 and 2, but slowing down the camera speed; in this case, a complete loop takes 800 frames. In both experiments, all parametrizations, except FID_0, perform consistently for about, respectively, 45 % and 80 % of time; FID_0 performs worse with respect to other parametrizations in these cases, but the average value of its inconsistency is comparable. In Experiments 7 and 8 the noise standard deviation is doubled; in this cases no parametrization is able to maintain the consistency for a significant amount of time.

Experiments 9 and 10 use the second environment; the camera translates by 8 cm on the X axis, 2 cm on the Y axis, and 2 cm on the Z axis at each step while rotating by 0.2°, −0.45°, 0.9° respectively around X, Y, Z. The additive noise on the motion



(a) Experiment 1 - Monocular (b) Experiment 2 - Monocular

(c) Experiment 3 - Monocular (d) Experiment 9 - Monocular

Fig. 4 Average NEES values in four selected experiments for the Monocular setup (simulation steps in the abscissa). *Dashed red lines* are the limits of the 95 % confidence interval

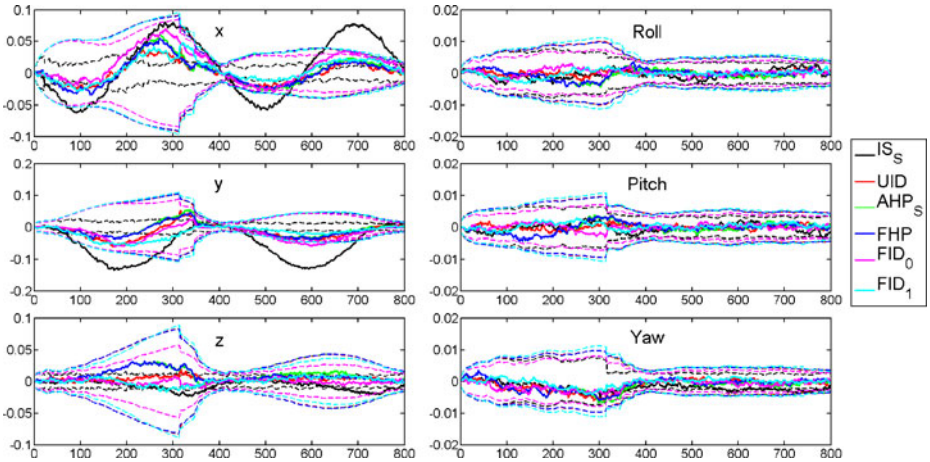


Fig. 5 Estimation errors (*continuous lines*) and estimated standard deviation (*dashed lines*) of a single run of Experiment 1. FID_1 is in *cyan*, while others parametrizations use the colors used in the previous figures

is the same of Experiments 3 and 4, while the initial depth value reflect the scheme used for the other experiments. In Experiment 9 (see Fig. 4d) no parametrization performs consistently, but the average inconsistency is about three time less for FID_0. In Experiment 10 all parametrizations are consistent for the most of time with very low values of average inconsistency.

We perform here also a different comparison between parametrizations; we take a single run of the first experiment and we analyze the estimation error for each coordinate and angle around axis with respect to the ground truth considering the $\pm 3\sigma$ confidence interval of the filter standard deviation. The aim of this comparison is to evaluate the impact of the additional noise $\nu_{3,4}$ in FID_1 parametrization. Results are shown in Fig. 5. As expected, IS underestimates the covariances and makes large errors. FHP, AHP and UID show a very similar behaviour while FID_0 tends to underestimate covariances with respect to them. FID_1 shows very similar covariances to others parametrizations (except IS), confirming that the choice of 1px as standard deviation of additional noise $\nu_{1,2}$ is appropriate and it does not favour FID_1 too much.

4.3 Stereo Simulated Experiments

We performed six experiments in Stereo SLAM differing in camera motion, initial depth of the feature for the initialization step and noise in motion. The parameters of these experiments are summarized in the second part of Table 1, the results of the consistency analysis are reported in Tables 4 and 5, and the most significant experiments are reported also in Fig. 6. Alike in the Monocular case, FID with covariance inflation in measurement, i.e., FID_1, is not compared and we focus on FID_0; IS shows poor performances also in the stereo case.

All the experiments are conducted in the first environment described for the Monocular case (i.e., simple cloister and planar motion); feature depth is initialized at two different values: for the first we use 0.5 m^{-1} for the mean of the inverse distance

Table 4 Stereo simulated experimental average NEES results

#		IS_N	IS_S	UID	AHP_N	AHP_S	FHP	FID_0	FID_1
1	Cons %	2	2	74	76	75	75	**83**	30
	Opt %	98	98	20	**17**	22	21	**17**	0
2	Cons %	0	0	84	84	86	87	**98**	14
	Opt %	100	100	15	14	9	9	**1**	0
3	Cons %	0	0	10	11	10	11	**50**	76
	Opt %	100	100	90	89	90	89	**50**	5
4	Cons %	0	0	28	**37**	13	12	8	62
	Opt %	100	100	72	**63**	87	88	92	30
5	Cons %	0	0	8	6	8	12	**57**	72
	Opt %	100	100	92	94	92	88	**43**	8
6	Cons %	0	0	0	**6**	2	2	0	47
	Opt %	100	100	100	**94**	98	98	100	52

The percentage of time in which each parametrization operates in consistent (Cons) and optimistic (Opt) range is reported. Best results (i.e., maximum consistency and minimum optimistic percentage) are reported in *bold*. FID_1 parametrization, i.e., with non-zero covariance on $v_{3,4}$, is reported in the last column and it is not compared with others

(or scale) and 1 m^{-1} for its standard deviation, in the second case 0.01 m^{-1} for the mean and 0.5 m^{-1} for its standard deviation (i.e, very far from the observer).

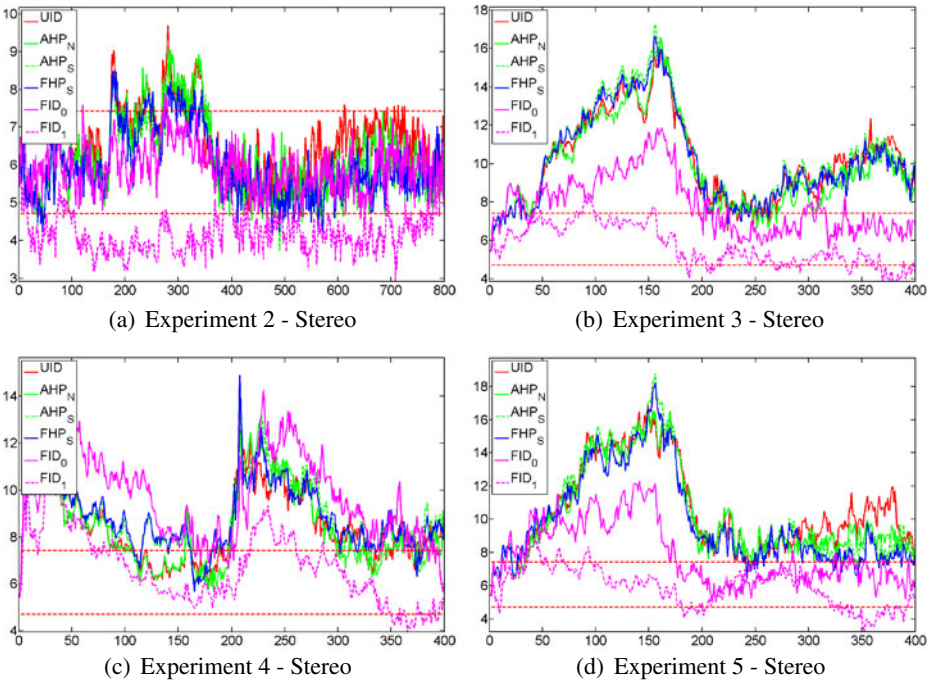
In Experiments 1 and 2, the camera moves for 800 steps on a plane with increments of 8 cm on the X axis and a rotation of 0.9° around the Z axis per step, performing two loops in the cloister. Both experiments initialize feature depth at 2 m with initial uncertainty on the inverse depth of 1 m^{-1}, whereas the standard deviation of the noise added to the motion is 2.5 mm on the three axis for translation and 0.025° around the three axis for rotation in the first experiment and it is doubled in the second. Results of the first experiment show that all parametrizations operate for about 75 % time in consistent range, for FID_0 this is larger than 80 %. Likewise in the second experiment, FID_0 is consistent for the 98 % of the execution, whereas other parametrizations do not exceed the 87 %, although in Fig. 6a we can notice that other parametrizations are quite close to the consistency limit.

In Experiments 3 and 4 we double the robot motion speed (increments are of 16 cm and 1.8°) performing 400 steps and covering two loops of the cloister. The

Table 5 Stereo simulated experimental results

#	IS_N	IS_S	UID	AHP_N	AHP_S	FHP	FID_0	FID_1
1	114.2	106.8	0.9	0.7	0.9	0.9	**0.5**	–
2	134.4	124.8	0.6	0.7	0.4	0.4	**0.2**	–
3	324.1	341.7	3.1	2.8	3.3	3.1	**1.6**	0.4
4	243.2	270.1	**1.7**	1.9	1.8	**1.7**	2.6	1.2
5	360.6	379.1	3.4	3.0	3.1	2.9	**2.3**	0.6
6	465.8	463.7	4.6	3.7	**3.6**	3.7	5.2	1.9

We report here $\frac{\int \bar{\epsilon}_t - H}{T_{\epsilon_t > H}}$, i.e., the average value of inconsistency for time step when the filter operate in optimistic range. Lower (i.e., better) values are in *bold*. FID_1 parametrization, i.e, with non-zero covariance on $v_{3,4}$, is reported in the last column and it is not compared with others. The "–" symbol means that the average values on the inconsistency is not calculable because the parametrization never operates in optimistic range, i.e., $T_{\epsilon_t > H} = 0$



(a) Experiment 2 - Stereo (b) Experiment 3 - Stereo

(c) Experiment 4 - Stereo (d) Experiment 5 - Stereo

Fig. 6 Average NEES values in four selected experiments for the stereo setup (simulation steps in the abscissa). *Dashed red lines* are the limits of the 95 % confidence interval

noise is set as in Experiment 1 to 2.5 mm and 0.025°. In Experiment 3 features are initialized at 2 meter with initial uncertainty on the inverse depth of 1 m^{-1}. Only FID_0 perform consistently for 50 % of the time, this is due to the fact that it remains consistent after the loop closure at frame 200, as showed in Fig. 6b, while other parametrizations stop at about 10 %. In Experiment 4 the initial depth was changed to $100m$ with uncertainty of $0.5m^{-1}$ and in this case the most consistent is the AHP parametrization with the initialization performed through normalization of viewing ray (AHP_N, refer to Eq. 43), although the performance is just the 37 % of time in the consistent range. From the analysis of the average value of inconsistency, and from Fig. 6c, we can notice in this experiment that other parametrizations are quite close to AHP_N performance.

Experiments 5 and 6 replicate Experiments 3 and 4, but doubling the noise on the motion (5 mm and 0.05°). In Experiment 5 (Fig. 6d) FID_0 performs definitely better than others: all parametrizations lose consistency for a certain amount of time, but FID_0 is able to recover consistency after a complete loop of the cloister and it remains consistent hereafter. Experiment 6 is the hardest for all the parametrizations: they all lose consistency early and they are not able to recover from that.

4.4 Evaluation on Real Data

We tested the proposed parametrizations also in a real setup, using the MoonSLAM framework. As for the simulated experiments, we use the odometry of the robot as

input in the motion model of the system. We have tested both the monocular and the stereo setup using almost the entire indoor dataset *Bicocca_2009-02-27a* from the RAWSEEDS project [6], covering a path of about one kilometer. The monocular camera has a resolution of 320×240 pixels with a focal length of about 200 pixels and it captures images at 30 Hz. We used about 57000 frames (32 min). Each camera of the stereo dataset has a resolution of 640×480 pixels, with a focal length of about 660 px and a frame rate of 15 fps, the baseline of the rig is about 18 cm; we consider the same path as for the monocular case, which totals about 28500 frames. Figure 7 shows a screenshot during the execution of a monocular run.

The algorithms perform active-search-based SLAM as for the simulated experiments. For the monocular case we initialize at most 16 landmarks in the first frame and search for new landmarks when less than 50 landmarks are predicted in the current image, with a limit of two landmarks added at once. For the stereo case we initialize at most 25 landmarks in the first frame and search for new landmarks when less than 25 landmarks are predicted in the current image, without limits of addition. The difference in the initialization policy are due to an empirical tuning of the system. In the stereo case, landmarks are always initialized in the left camera reference frame, while measurements are obviously performed on both cameras.

Each landmark is described by a patch of 11×11 pixel area around the salient points, which are detected by FAST [29], whereas matching is done by normalized cross-correlation. Initial depth is set to 10 m ($0.1\ \mathrm{m}^{-1}$) with uncertainty of $0.5\ \mathrm{m}^{-1}$ both for monocular and stereo case. The update step is performed using the 1-Point RANSAC technique [10], which was briefly introduced in Section 2.6.

We also apply the Conditional Independent Submapping [27, 28] technique, that allows to treat large problems through subdivision in sub maps without information

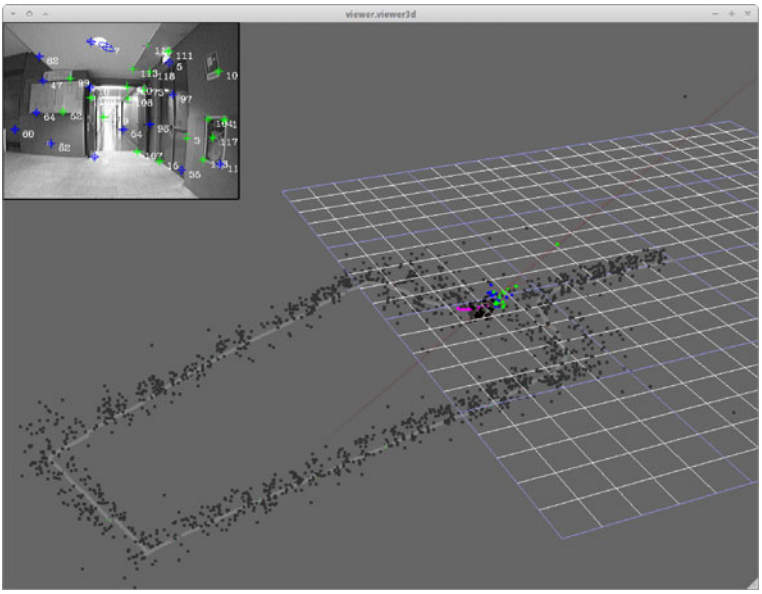


Fig. 7 Screenshot of MoonSLAM framework performing monocular SLAM on a real dataset

loss and keeping the computational complexity bounded by a constant, i.e., O(1) EKF-Slam. The use of this technique implies the reformulation of the parametrizations in terms of *shared features* (discussed in Section 3.6) to avoid singularities in the *back propagation* step. We choose a maximum map size of 70 features and, when this limit is reached, a new map is started.

We have to underline that the presented implementation does not perform the so called *loop detection* and *loop closure* procedures, which are a very important part of SLAM systems. While it has to be noticed that loop closure could reset the errors cumulated along the path before the loop closure point, our choice was not to include this functionality so to increase the effect of the parametrization, continuing the cumulation of all errors beyond the path point where a loop could have been closed.

Figures 8 and 9 summarizes the results respectively of the Monocular and Stereo SLAM for the parametrizations UID, AHP, FHP, and FID_0 while IS is not treated in these runs, due to its poor performance. The *extended ground truth*, i.e., a reference on the real trajectory available for the RAWSEEDS datasets, is plotted to allow comparison together with the robot odometry. Figures 8a and 9a show the top view *smoothed trajectory*, i.e., reconstructed by the sequence of anchor vectors or frames stored in the shared part of the features.[11] Figures 8b and 9b show the top view of the map points (with cross) on top of the *smoothed trajectory*.

From these experiments we see how all parametrizations show quite good performance in the reconstruction of both the trajectory and the corresponding map. As expected the stereo system is able to reconstruct the map with a higher accuracy. Since we used the odometry in the motion model for the monocular case we do not observe any significant *scale drift* phenomenon as observed in [10].

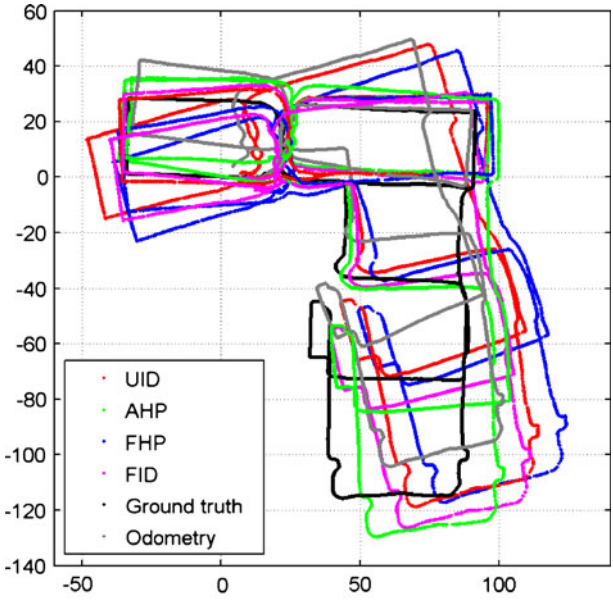
It is difficult from these results to state which parametrization is the best, if any; when using real data, there are several sources of errors, which are not properly modelled in the state of the art approaches, so it might be difficult, to discriminate whether the performance gain (or loss) to be fully due to the parametrization or other sources of error. Different setups (e.g., changes in standard deviation of the noise considered in the model, different depth initialization, etc.) could also affect the results. On the other hand, a perhaps un-expected qualitative conclusion that can be safely drawn, is that the FID_0 parametrization shows a good performance also in a real setup, although it can not re-estimate the direction of the initial viewing ray.

4.5 Time Performances Evaluation

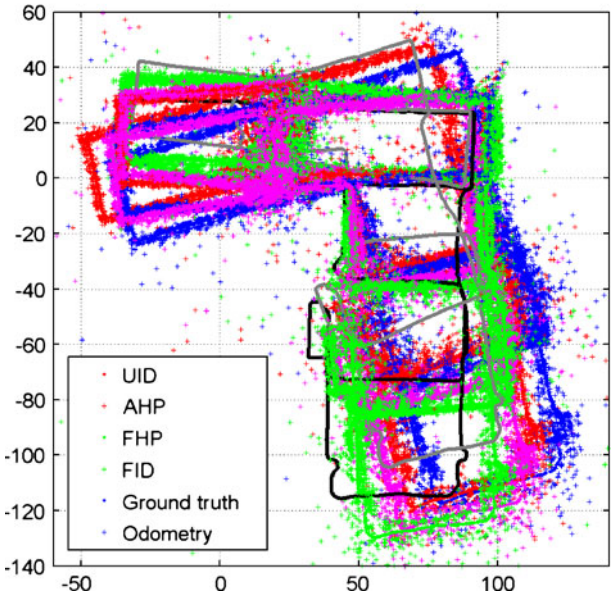
The performance, in terms of execution time, of the EKF SLAM have been evaluated in both simulated and real experiments in order to validate the real-time figures. The experiments were conducted on a laptop equipped with an Intel Core2 Duo T9300 CPU running at 2.50 GHz per core with 3.4 GB of DDR2 RAM.

Figure 10 shows the running frequency in a simulated experiment, in particular Fig. 10b shows the frequency for each parametrization at each iteration and Fig. 10a

[11] This trajectory is very similar to the one computed on-line but takes advantages of the back-propagation step of the Conditional Indipendent Submapping [27, 28].



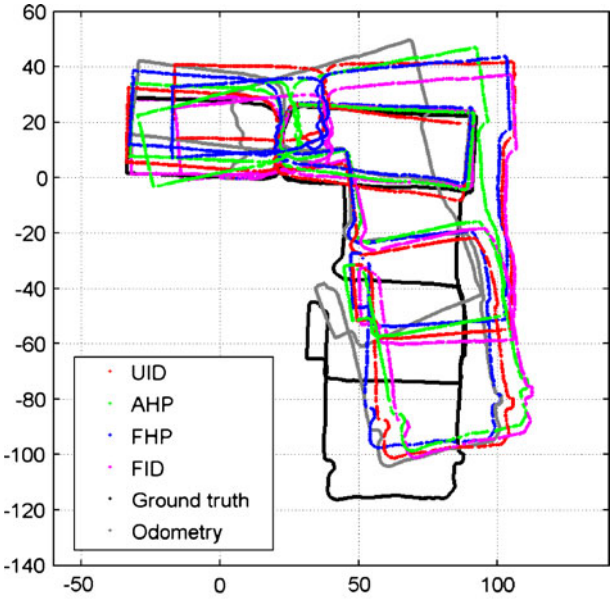
(a) Smoothed trajectory, top view



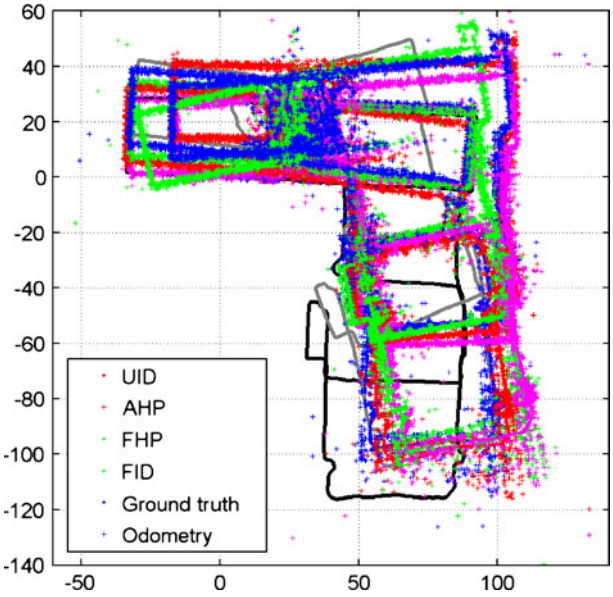
(b) Reconstructed map and smoothed trajectory, top view

Fig. 8 Robot trajectory and map estimates on the real data in the Monocular setup

shows the number of landmarks in the filter at each iteration. We have to point out that in this experiment we configured the landmark addition policy so as to allow the addition of more than one landmark for frame, resulting in an advantage for parameterizations that have a small non common part, e.g. FID shares the



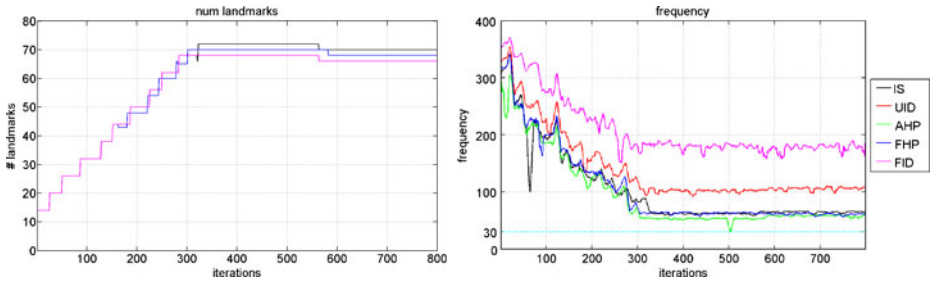
(a) Smoothed trajectory, top view



(b) Reconstructed map and smoothed trajectory, top view

Fig. 9 Robot trajectory and map estimates on the real data in the Stereo setup

common anchor frame of seven elements and it adds only one element for each new landmark initialized at the same frame. At iteration 400 the first turn of the cloister is completed; this implies that all the landmarks are already in the filter and



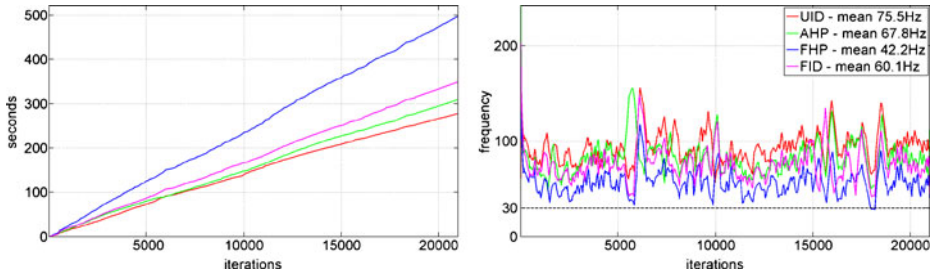
(a) Number of landmarks in the filter (b) Operative frequency

Fig. 10 Performance of EKF SLAM in simulated experiments. In **a** the number of landmarks in the filter at each iterations; in **b** the operative frequency at each iteration; a *cyan dashed line* at 30 Hz is provided as reference real-time performances

the computation time becomes constant hereafter. Assuming a nominal frequency of 30 Hz for the camera, results show that all the parameterizations operate at real-time rate. In particular FID is the fastest and it reaches the 150 Hz, UID reaches the operative frequency of about 100 Hz, while FHP, IS and AHP perform a little worse, reaching about 60 Hz. We should notice that in simulated experiments the update step is performed with an iterative procedure (following the procedure sketched in Section 2.6). This negatively affects performance, since Jacobians are recomputed several times per iteration. Moreover, the iteration time is the cumulation of all the operations that are involved in the simulation, including accessory operations such as the simulation of the motion and the projection of the map points.

Figures 11 and 12 show the performance of all parametrizations in the initial 21000 frames of monocular and in the initial 11000 frames of stereo real experiments.[12] We have to point out that in this case the update step is conducted with the standard procedure sketched in Section 2.6, thus performances are not directly comparable to the simulated ones. Moreover, the use of the Conditional Independent Submapping framework of [27, 28] allows a complexity of O(1) for each submap. This could be observed in Figs. 11a and 12a, where the cumulated execution time is roughly linear in the number of iterations. From Fig. 11b we notice that all parametrizations, with a maximum number of 70 landmarks per map, allow the EKF based SLAM algorithm, to perform at a higher frequency than the camera frame rate in the monocular case (30 fps). UID results to be the fastest parametrization (75 Hz), followed by AHP (68 Hz), FID (60 Hz) and FHP (42 Hz). The stereo experiment (Fig. 12a and b) shows again a real-time performance considering we had a 15 fps camera frame rate; the performance of UID, AHP and FID, in this case, is very similar: all of them run at around 28 Hz. Notice that the slowdown, with respect to the monocular case, is due to the use of a stereo system, that implies to double the patch matching operations, and to perform an update step with a higher dimension of the measurement vector.

[12]The Rawseeds datasets are provided as a sequence of images (one image per file). This greatly slows down performance due to the high number of OS requests to open and close files. We choose to evaluate time performance on uncompressed videos generated only on the initial part of the dataset.

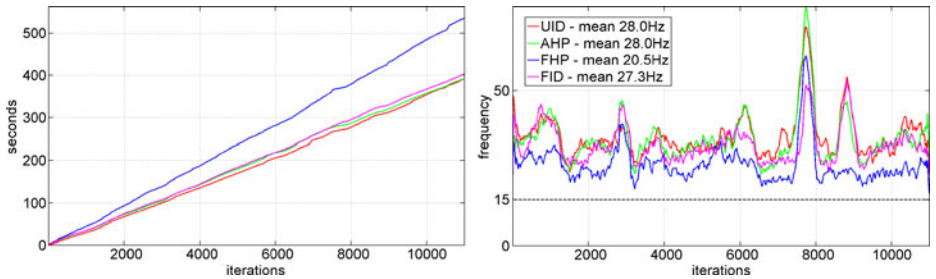


(a) Cumulated time (b) Operative frequency

Fig. 11 Performance of EKF SLAM in the Monocular real experiments

The similarity in performance for UID, AHP and FID could be due to the different policies in feature addition used in this stereo setup (see Section 4.4), which does not put an upper bound to the number of features added in each frame. This allows to exploit the benefits of the shared parametrizations introduced in Section 3.6.

As observed previously in Section 3.6, FID has a lower impact on the state vector size with respect to other parametrizations when more than two features are added at the same time. We then tried to run the monocular real experiment with a different policy for feature addition: add landmarks only when at least three are available, but with a maximum addition of four landmarks per frame. We noticed that the accuracy in the reconstruction of the trajectory and the map has not been affected by this choice, while all (shared) parametrizations took great advantage in time performance by this new policy. UID reaches the 101 Hz frequency, followed by FID (98 Hz), AHP (75 Hz) and FHP (70 Hz). As expected FID takes the greater advantage from this new feature initialization policy with respect to the results of Fig. 11a, whereas AHP performance grows only slightly due to the higher dimension of the non shared part. We expected FID to perform better than UID, but this did not happen; we assume this is mainly due to the presence of unstable landmarks which, when deleted from the filter, reduce the advantages of FID. A more detailed analysis of the execution time of each operation should be done to prove this hypothesis.



(a) Cumulated time (b) Operative frequency

Fig. 12 Performances of EKF SLAM in the Stereo real experiments

5 Conclusions

In this paper we introduced the problem of SLAM and then its variant known as Visual SLAM, which is the case of basing on input from cameras. The approach presented is based on Extended Kalman Filtering, whose usage for Visual SLAM is also reviewed in the paper. We then presented the state of the art in the field of parametrization of 3D point features, which is quite an important issue in order to tackle the Visual SLAM problem with EKF. We also introduced a new parametrization, Framed Inverse Depth (FID), which allows a similar filter consistency (in terms of average NEES) with respect to state-of-the-art parametrizations, at a frequently lower computational cost. We presented an extensive experimental activity, both in a simulated settings, which allows to compute the performance with respect to the Ground Truth, and on real data. The experimental activity has been performed both for the case of monocular and stereo SLAM, with odometry.

The good consistency with lower computational complexity is just one of the interesting features of the FID parametrization. By storing the full pose (a complete anchor frame) of the camera when the feature was perceived the first time, the FID parametrization, as well as the FHP parameterization, implements a sort of keyframed representation: every time a feature gets updated by the Kalman filter, past camera poses get refined as well. This seems somehow violating the EKF assumption of the current pose being a summary of the whole camera trajectory (i.e., past poses being marginalized out). Thanks to the presence of fully framed selected past poses, the EKF-based SLAM implements a recursive filtering/smoothing of the past camera poses (i.e., keyframes) and 3D features. In a sense parametrizations based on a complete anchor, stay in between pure filtering approaches and optimization-based ones. A possible application of these considerations is the integration of batch optimization method (e.g., Bundle Adjustment) within the EKF machinery and the Conditional Independent Submapping framework, in order to provide a system that could deal with Large Scale Visual SLAM in real time with an accuracy comparable to off-line structure from motion methods.

References

1. Agrawal, M., Konolige, K.: Frameslam: From bundle adjustment to real-time visual mapping. IEEE Trans. Robot. **24**(5), 1066–1077 (2008)
2. Andrade-Cetto, J., Vidal-Calleja, T., Sanfeliu, A.: Unscented transformation of vehicle states in slam. In: Robotics and Automation, 2005. ICRA 2005. In: IEEE International Conference on Proceedings of the 2005, pp. 323–328 (2005). doi:10.1109/ROBOT.2005.1570139
3. Bailey, T., Nieto, J., Guivant, J., Stevens, M., Nebot, E.: Consistency of the ekf-slam algorithm. In: Proc. of IEEE Intern. Conf. on Intelligent Robots and Systems, pp. 3562–3568 (2006)
4. Barfoot, T.: Online visual motion estimation using fastslam with sift features. In: 2005 IEEE/RSJ International Conference on Intelligent Robots and Systems, 2005. (IROS 2005), pp. 579–585 (2005). doi:10.1109/IROS.2005.1545444
5. Castellanos, J.A., Martinez-Cantin, R., Tardos, J.D., Neira, J.: Robocentric map joining: improving the consistency of EKF-SLAM. Robot. Auton. Syst. **55**(1), 21–29 (2007)
6. Ceriani, S., Fontana, G., Giusti, A., Marzorati, D., Matteucci, M., Migliore, D., Rizzi, D., Sorrenti, D.G., Taddei, P.: Rawseeds ground truth collection systems for indoor self-localization and mapping. Auton. Robots **27**(4), 353–371 (2009)
7. Ceriani, S., Marzorati, D., Matteucci, M., Migliore, D., Sorrenti, D.G.: On feature parameterization for ekf-based monocular slam. In: proceedings of 18th World Congress of the International Federation of Automatic Control (IFAC), pp. 6829–6834 (2011)

8. Chekhlov, D., Pupilli, M., Mayol-Cuevas, W., Calway, A.: Real-time and robust monocular slam using predictive multi-resolution descriptors. In: 2nd International Symposium on Visual Computing. URL http://www.cs.bris.ac.uk/Publications/Papers/2000571.pdf (2006)
9. Civera, J., Davison, A.J., Montiel, J.M.M.: Inverse depth to depth conversion for monocular slam. In: Proc. of IEEE Intern. Conf. on Robotics and Automation, pp. 2778–2783 (2007)
10. Civera, J., Grasa, O., Davison., A., Montiel, J.M.M.: 1-point RANSAC for Extended Kalman filtering: Application to real-time structure from motion and visual odometry. JFR **27**, 609–631 (2010)
11. Davison, A.: Real-time simultaneous localisation and mapping with a single camera. In: Proc. of IEEE Intern. Conf. on Computer Vision (2003)
12. Davison, A.J., Reid, I.D., Molton, N., Stasse, O.: MonoSLAM: real-time single camera SLAM. IEEE Trans. Pattern Anal. Mach. Intell. **29**(6), 1052–1067 (2007)
13. Durrant-Whyte, H., Bailey, T.: Simultaneous localization and mapping: part i. IEEE Robot. Autom. Mag. **13**(2), 99–110 (2006). doi:10.1109/MRA.2006.1638022
14. Funda, J., Paul, R.: A comparison of transforms and quaternions in robotics. In: Proc. of the 1988 IEEE Intern. Conf. on Robotics and Automation, vol. 2, pp. 886–891 (1988)
15. Hartley, R.I., Zisserman, A.: Multiple View Geometry in Computer Vision, 2nd edn. Cambridge University Press (2004)
16. Holmes, S., Klein, G., Murray, D.: An $o(n^2)$ square root unscented kalman filter for visual simultaneous localization and mapping. IEEE Trans. Pattern Anal. Mach. Intell. **31**(7), 1251–1263 (2009). doi:10.1109/TPAMI.2008.189
17. Imre, E., Berger, M., Noury, N.: Improved inverse-depth parameterization for monocular simultaneous localization and mapping. In: Proc. of IEEE Intern. Conf. on Robotics and Automation, pp. 381–386 (2009)
18. Klein, G., Murray, D.: Parallel tracking and mapping on a camera phone. In: Proc. Eigth IEEE and ACM International Symposium on Mixed and Augmented Reality (ISMAR'09). Orlando (2009)
19. Lin, K.H., Wang, C.C.: Stereo-based simultaneous localization, mapping and moving object tracking. In: IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). Taipei, Taiwan (2010)
20. Lu, F., Milios, E.: Globally consistent range scan alignment for environment mapping. Auton. Robots **4**, 333–349 (1997)
21. Martinez-Cantin, R., Castellanos, J.: Bounding uncertainty in EKF-SLAM: The robocentric local approach. In: Proc. of IEEE Intern. Conf. on Robotics and Automation (2006)
22. Marzorati, D., Matteucci, M., Migliore, D., Sorrenti, D.G.: On the use of inverse scaling in monocular slam. In: Proc. of IEEE Intern. Conf. on Robotics and Automation, pp. 2030–2036 (2009)
23. Migliore, D., Rigamonti, R., Marzorati, D., Matteucci, M., Sorrenti, D.G.: Use a single camera for simultaneous localization and mapping with mobile object tracking in dynamic environments. In: Proceedings of International Workshop on Safe Navigation in Open and Dynamic Environments Application to Autonomous Vehicles (2009)
24. Montiel, J., Civera, J., Davison, A.J.: Unified inverse depth parametrization for monocular slam. In: Proc. of Robotics: Science and Systems (2006)
25. Newcombe, R., Davison, A.: Live dense reconstruction with a single moving camera. In: 2010 IEEE Conf. on Computer Vision and Pattern Recognition (CVPR), pp. 1498–1505 (2010)
26. Pietzsch, T.: Efficient feature parameterisation for visual slam using inverse depth bundles. In: Proc. of BMVC Conf. (2008)
27. Piniés, P., Paz, L.M., Gálvez-López, D., Tardós, J.D.: CI-Graph simultaneous localization and mapping for three-dimensional reconstruction of large and complex environments using a multicamera system. JFR **27**(5), 561–586 (2010)
28. Pinies, P., Tardos, J.: Large-scale slam building conditionally independent local maps: application to monocular vision. IEEE Trans. Robot. **24**(5), 1094–1106 (2008). doi:10.1109/TRO.2008.2004636
29. Rosten, E., Drummond, T.: Machine learning for high-speed corner detection. In: European Conference on Computer Vision, vol. 1, pp. 430–443 (2006). doi:10.1007/11744023_34
30. Sim, R., Elinas, P., Little, J.: A study of the rao-blackwellised particle filter for efficient and accurate vision-based slam. Int. J. Comput. Vis. **74**, 303–318 (2007). doi:10.1007/s11263-006-0021-0
31. Solà, J.: Consistency of the monocular EKF-SLAM algorithm for three different landmark parametrizations. In: 2010 IEEE Intern. Conf. on Robotics and Automation (ICRA), pp. 3513–3518. IEEE (2010)

32. Solà, J., Monin, A., Devy, M.: Bicamslam: two times mono is more than stereo. In: 2007 IEEE Intern. Conf. on Robotics and Automation (ICRA), pp. 4795–4800 (2007)
33. Solà, J., Monin, A., Devy, M., Lemaire, T.: Undelayed initialization in bearing only slam. In: Proc. of Intern. Conf. on Intelligent Robots and Systems, pp. 2499–2504 (2005)
34. Solà, J., Vidal-Calleja, T., Civera, J., Montiel, J.M.M.: Impact of landmark parametrization on monocular EKF-SLAM with points and lines. Int. J. Comput. Vis. Available online at Springer's: http://www.springerlink.com/content/5u5176nj521kl3h0/ (2011)
35. Strasdat, H., Montiel, J., Davison, A.: Real-time monocular slam: why filter? In: Proceedings of IEEE International Conference on and Automation (ICRA), pp. 2657–2664 (2010)
36. Strasdat, H., Montiel, J.M.M., Davison, A.: Scale drift-aware large scale monocular slam. In: Proceedings of Robotics: Science and Systems. Zaragoza, Spain (2010)
37. Thrun, S., Montemerlo, M.: The GraphSLAM algorithm with applications to large-scale mapping of urban structures. IJRR **25**(5/6), 403–430 (2005)