Closing Loops With a Virtual Sensor Based on Monocular SLAM

Rodrigo Munguía and Antoni Grau

***Abstract*—Monocular simultaneous localization and mapping (SLAM) techniques implicitly estimate camera ego-motion while incrementally building a map of the environment. In monocular SLAM, when the number of features in the system state increases, maintaining a real-time operation becomes very difficult. However, it is easy to remove old features from the state to maintain a stable computational cost per frame. If features are removed from the map, then previously mapped areas cannot be recognized to minimize the robot's drift; alternatively, in the context of a real-time virtual sensor that emulates typical sensors as laser for range measurements and encoders for dead reckoning, this limitation should not be a problem. In this paper, a novel framework is proposed to build in real time a consistent map of the environment using the virtual-sensor estimations. At the same time, the proposed approach allows minimizing the drift of the camera-robot position. Experiments with real data are presented to show the performance of this frame of work.**

***Index Terms*—Autonomous vehicles, computer vision, distributed estimation, robot vision systems, virtual sensors (VSs).**

I. Introduction

THE ONLINE robot position estimation from measurements of self-mapped features is a class of problem known, in the robotics community, as the simultaneous localization and mapping (SLAM) problem, which is one of the most common affecting robotics. SLAM consists of increasingly building a consistent map of the environment and, at the same time, localizing the robot's position while exploring its world. SLAM is one of the most active research fields in robotics, with excellent results obtained during the last years, but until recently, it was mainly restricted to the use of laser range-finder sensors and, predominantly, to build 2-D maps (see [1] and [2] for a recent review).

Cameras have become more and more interesting for the robotic research community as sensors, because they yield a lot of information. It is a sensor from which 3-D information can be extracted. Even for indoor robots whose pose can be represented in 2-D, the ability to gather 3-D information on the environment is essential. Cameras are well adapted for embedded systems: they are light, cheap, and power saving. As computational power grows, an inexpensive camera can be used to simultaneously perform range- and appearance-based sensing, by replacing typical sensors such as laser and sonar rings for range measurement and encoders for dead reckoning. Nevertheless, more complex monocular pseudostereo-vision systems have been used as a robot's sensor [3]. A wide variety of algorithms can be obtained from the vision research community to extract high-level primitives from the image and match them with primitives stored in the map, thus allowing reliable data association. This is one the most important problems to solve in SLAM.

In this context, the 6-DOF monocular camera case (monocular SLAM) possibly represents the harder variant of SLAM. Monocular SLAM is closely related to the structure-from-motion (SFM) problem of reconstructing scene geometry. SFM techniques [4], [5] have successfully been used for recovering camera position and scene structure. However, the SFM techniques coming from the vision community research have been formulated as offline algorithms and required batch simultaneous processing of all the images acquired in the sequence. In contrast to SFM approaches that rely on global nonlinear optimization, recursive estimation methods allow an online operation, which is highly desirable for a SLAM system. Some hybrid techniques (SFM–Kalman filtering), as quoted in [6], which is based on stereo vision, have also appeared. Nevertheless, some of the drawbacks (in a purely robotic context) of these methods remain due to the global optimization nature of the SFM methods.

In recent years, monocular SLAM approaches [7]–[9] have shown good results in real-time 6-DOF camera pose and orientation estimation, as well as in building 3-D maps of 50–100 sparse features. Several important improvements to the robustness of this kind of methods have also appeared [10]–[12]. There are also some recent approaches for increasing the amount of features in the map maintaining a real-time operation [13], [14]. Nevertheless, two of the main challenges at this moment are probably the use of these methods for applications that require more features in the map and the closing of loops with a big drift in the estimation.

In an authors' recent work [15], a new kind of feature initialization called delayed inverse-depth feature initialization is proposed to increase the robustness of monocular SLAM methods, when a reference is used to recover the metric scale of the environment. In that sense, a new method to recover the scale is additionally proposed. In monocular SLAM methods such as those referred to in [7] and [8], in which the number of features in the system is increasing, maintaining a real-time

Manuscript received January 30, 2008; revised August 10, 2008. Current version published July 17, 2009. This work was supported in part by Consolider Ingenio 2010 under Project CSD2007-00018, by the Comisión Interministerial de Ciencia y Tecnología under Project DPI2007-61452, and by the European Community Union under Project IST-045062. The Associate Editor coordinating the review process for this paper was Dr. Annamaria Varkonyi-Koczy

The authors are with the Department of Automatic Control, Technical University of Catalonia (UPC), 08028 Barcelona, Spain (e-mail: rodrigo.munguia@upc.edu; antoni.grau@upc.edu).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TIM.2009.2016377

operation becomes very difficult. Therefore, the map size is limited to typically around 100 features, reducing the working area of these methods. These authors' previous work also shows that it is easy to remove old features from the state to maintain a stable computational cost per frame and, therefore, a real-time operation. Moreover, if features are removed from the map, then previously mapped areas cannot be recognized in the future, and loops cannot be detected. Therefore, the implicit drift in the estimations cannot be minimized. Nevertheless, a modified monocular SLAM method to maintain a stable computational operation can be viewed as a complex real-time "virtual sensor" (VS), which provides appearance-based sensing and emulates typical sensors such as laser for range measurements and encoders for dead reckoning (visual odometry).

The aim of this work is to show that a VS based on monocular SLAM can be used as the basis for building a consistent and larger map of the environment, even when using a single camera as the only sensory input.

A distributed scheme is proposed, where a classic SLAM method is plugged into the VS (decoupled from the camera's frame rate) to build and to maintain the global map and final camera pose. In our implementation, both estimation processes, the VS and the classic SLAM [called in this work as global SLAM (GS)], concurrently run in different local network PCs communicated by a Transmission Control Protocol (TCP)/IP.

In a very recent work [16], a submapping technique for building maps over large camera trajectories has been presented. By applying such technique, real-time collected submaps are processed offline for refinements and closing-loop detection using an iterated Kalman filter. This method and the presented one in this paper share the idea of using an extra estimation process for building the final map. In addition, the architecture of the method proposed in this work is different from [16] among other aspects, because it looks for a fully concurrent and continuous online estimation of the global map.

The paper is organized as follows. Section II describes the VS based on monocular SLAM. Section III explains why a typical monocular SLAM method cannot be well suited for solving the whole problem when the trajectory of the camera requires a vast number of natural landmarks in the map or whenever there is a drift in the estimation. Section IV describes the GS. Section V exposes experimental results with real data. Finally, conclusions are presented in Section VI.

II. VS Based on Monocular SLAM

A. Six-DOF Monocular Slam

An unconstrained constant-velocity camera motion prediction model can be defined by the following equation [7], [17]:

$$f_v = \begin{bmatrix} r_{k+1}^{WC} \\ q_{k+1}^{WC} \\ v_{k+1}^{W} \\ \omega_{k+1}^{W} \end{bmatrix} = \begin{bmatrix} r_k^{WC} + \left(v_k^W + V_k^W\right)\Delta t \\ q_k^{WC} \times q\left(\left(\omega_k^W + \Omega^W\right)\Delta t\right) \\ v_k^W + V^W \\ \omega_k^W + \Omega^W \end{bmatrix} \quad (1)$$

with $q((\omega_k^W + \Omega^W)\Delta t)$ being the quaternion defined by the rotation vector $(\omega_k^W + \Omega^W)\Delta t$. At every step, it is assumed that there is an unknown linear and angular velocity with acceleration zero-mean and known-covariance Gaussian processes a^W and α_W, producing an impulse of linear and angular velocity: $V^W = a^W \Delta t$ and $\Omega^W = \alpha^W \Delta t$. The superscripts ^{W} and ^{WC} denote magnitudes expressed in world reference and camera reference, respectively. The camera state $\hat{\mathrm{x}}_v$ is defined by

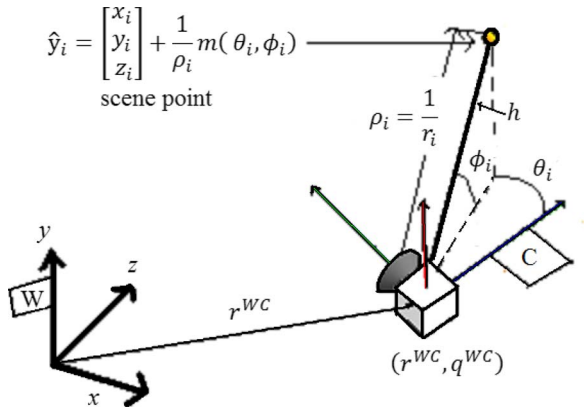

Fig. 1. Six-DOF monocular SLAM camera and features parameterization.

$$\hat{\mathrm{x}}_v = [\, r^{WC} \quad q^{WC} \quad v^W \quad \omega^W \,]^T \quad (2)$$

where $r^{WC} = [x, y, z]$ represents the optical center position of the camera, $q^{WC} = [q_0, q_1, q_2, q_3]$ represents the camera orientation by a quaternion, and $v^W = [v_x, v_y, v_z]$ and $\omega^W = [\omega_x, \omega_y, \omega_z]$ denote linear and angular velocities, respectively. These terms are the same as those used in (1). An extended Kalman filter (EKF) propagates the camera pose and the velocity estimates, as well as the feature estimates. The complete state that includes the features $\hat{\mathrm{y}}$ consists of $\hat{\mathrm{x}} = [\hat{\mathrm{x}}_v^T, \hat{\mathrm{y}}_1^T, \ldots, \hat{\mathrm{y}}_n^T]^T$, where a feature $\hat{\mathrm{y}}_i$ represents a point i in the 3-D scene defined by the following 6-D state vector:

$$\hat{\mathrm{y}}_i = [x_i, y_i, z_i, \theta_i, \varphi_i, \rho_i] \quad (3)$$

which models the 3-D point located at

$$[x_i, y_i, z_i]^T + (1/\rho_i) m(\theta_i, \varphi_i) \quad (4)$$

where x_i, y_i, and z_i are the optical center coordinates of the camera in which the feature was first observed, and θ_i and φ_i represent the azimuth and the elevation (in relation to the world reference W) for the directional unitary vector $m(\theta_i, \varphi_i)$. The point depth r_i along the ray is coded by its inverse $\rho_i = 1/r_i$, as quoted in [8]. Fig. 1 illustrates the camera and feature parameterization.

The different locations of the camera, along with the location of the already-mapped features, are used to predict the feature position h_i. The model observation of a point $\hat{\mathrm{y}}_i$ from a camera location defines a ray expressed in the camera frame as

$$\begin{aligned} h^C &= [h_x, h_y, h_z] \\ &= R^{CW}\left([x_i, y_i, z_i]^T + (1/\rho_i) m(\theta_i, \varphi_i) - r^{WC}\right) \end{aligned} \quad (5)$$

where h^C is observed by the camera through its projection in the image. R^{CW} is the transformation matrix from the global reference frame to the camera reference frame. The projection is modeled by using a full-perspective wide-angle camera. The active feature search [18] is constrained to elliptical regions

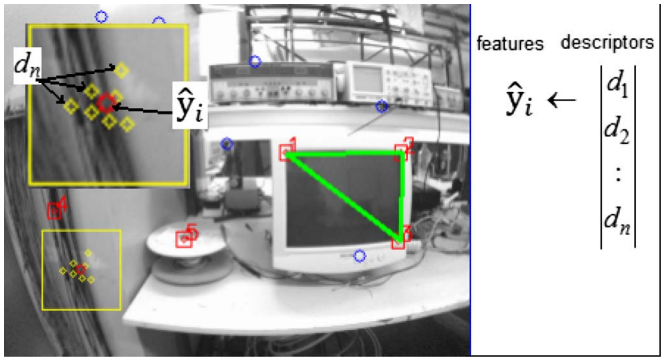

Fig. 2. Saliency operator is used to detect points of interest to initialize new features $\hat{y}_i$. When $\hat{y}_i$ is initialized, descriptors are extracted from a $p \times p$ pixel patch centered in $\hat{y}_i$ and associated to a feature.

around the predicted h_i. The elliptical regions are defined by the innovation covariance matrix

$$S_i = H_i P_{k+1} H_i^T + R \tag{6}$$

where H_i is the Jacobian of the sensor model with respect to the state, P_{k+1} is the prior state covariance, and the measurements z are assumed to be corrupted by zero-mean Gaussian noise with covariance R. The initialization of new features (delayed inverse-depth method) and recovering the metric scale of the world are widely explained in [15].

B. Adapting Monocular SLAM as a VS

To adapt a monocular SLAM method as a VS as the described above, the sensor must be modified to maintain a stable computational cost per frame. When a new feature is initialized (Fig. 2) in the VS, the proposed approach is to apply some of the saliency operators used in most mono-SLAM methods (Shi–Tomasi, Canny, etc.) and to extract n speeded-up-robust-feature (SURF) descriptors d_i from a $p \times p$ patch centered in the point detected by the saliency operator [19]. All extracted descriptors are stored and related to with the $\hat{y}_i$ feature. This way, each feature $\hat{y}_i$ can have a lot of useful information to be matched in the future. SURF descriptors are employed because they not only prove to have performance that is similar to that of the scale-invariant feature transform (SIFT) [20] but also have a lower computational cost [21].

VS and GS are communicated by TCP/IP. In our implementation, the VS is defined as the server that serves requests asynchronously coming from the GS. When a request is received, camera movement and feature information (emulating a real range sensor) are sent. Fig. 3 illustrates the camera movement information o_{vs} sent to the GS defined by

$$o_{vs} = \begin{bmatrix} o_x \\ o_y \\ o_z \\ o_\theta \\ o_\varphi \end{bmatrix} = \begin{bmatrix} \Delta x^C \\ \Delta y^{WC} \\ \Delta z^C \\ \Delta\theta^{WC} \\ \Delta\varphi^{WC} \end{bmatrix} = \begin{bmatrix} \Delta x^W \cos\left(\theta_{k-n}^{WC}\right) - \Delta z^W \sin\left(\theta_{k-n}^{WC}\right) \\ y_k^{WC} - y_{k-n}^{WC} \\ \Delta x^W \sin\left(\theta_{k-n}^{WC}\right) - \Delta z^W \cos\left(\theta_{k-n}^{WC}\right) \\ \theta_k^{WC} - \theta_{k-n}^{WC} \\ \varphi_k^{WC} - \varphi_{k-n}^{WC} \end{bmatrix} \tag{7}$$

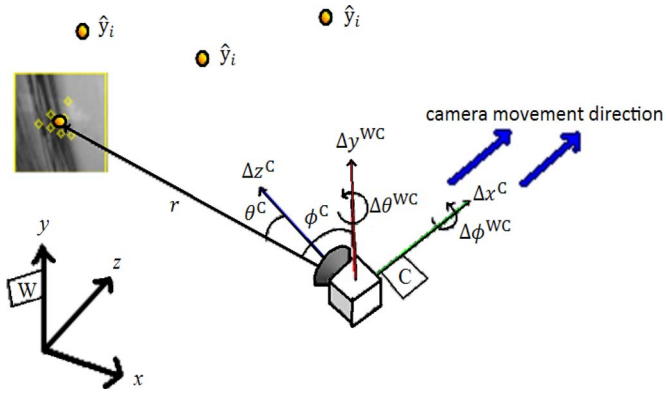

Fig. 3. Parameterization for data sent by the VS.

where $\Delta x^W = x_k^W - x_{k-n}^W$, $\Delta z^W = z_k^W - z_{k-n}^W$, and $(\theta^{WC}, \varphi^{WC})$ denote the camera orientation, and they have been obtained from the quaternion q^{WC}. The k subscript denotes the current step, and the $k-n$ subscript is equal to the last k step at the specific moment in which information was sent to the GS.

Information about each feature $\hat{y}_i = [x\ y\ z\ \theta^{WC}\ \varphi^{WC}\ \rho]$ is only sent if its estimated depth converges. This information is related to a minimum number i of descriptors d_i. The estimated depth r of a feature is considered to be converging if $100(\sigma_\rho)/\rho < l$ (in the experiments, $l = 5$). The feature pose information is sent emulating a 3-D range sensor (Fig. 3)

$$s_i = \begin{bmatrix} r \\ \theta^C \\ \varphi^C \end{bmatrix} = \begin{bmatrix} 1/\rho \\ \text{atan2}(\Delta z^W, \Delta x^W) - \theta_k^{WC} \\ \text{atan2}\left((\Delta x^W)^2 + (\Delta z^W)^2, \Delta y^{WC}\right) - \Delta\varphi_k^{WC} \end{bmatrix} \tag{8}$$

where r is the range and (θ^C, φ^C) defines the direction of r in the camera frame coordinates. The entire packet sent to the GS is made of $[o_{vs}, s_1, s_2, \ldots, s_n]$ and descriptors d_i. `atan2` is a two-argument function that computes the arctangent of y/x given y and x, within a range of $[-\pi, \pi]$.

III. SLAM Paradox

From the estimation point of view, it might seem redundant if we used a SLAM process to build a global map by using the output of a modified monocular SLAM process (VS) as its input. However, a closer view reveals that monocular SLAM is a particular case of the general SLAM problem, and it is constrained to very specific considerations. The objective of this section is to justify the motivations behind our proposed approach, in spite of this apparent paradox.

Monocular SLAM methods typically use patch cross correlation and active feature search to match the features in subsequent frames. Patch cross correlation is simple and fast, and active search is attractive because it only directly focuses on the area of interest, by maximizing computational resources and minimizing the chance of obtaining a mismatch [9], [18]. These techniques are suitable when the incoming video has to be processed online at frame rate. Such techniques can be

applied if a single camera is being used as the only sensorial input for the system.

In [22], where SIFT descriptors are used for the matching process, a high frame rate cannot be addressed due to the computational cost of the descriptors. Then, the robot's odometry is used for reducing the propagation of the uncertainty when the robot moves. If any other sensor—apart from a single camera—is to be used, a high frame rate is recommendable because the baseline between subsequent frames is small. Therefore, the propagation of uncertainty is also small, constraining the search for matching features to reduced regions. Nevertheless, when the robot moves far away from its initialization point and describes a different path (closing-loop problem), patch cross correlation and active search are not suitable.

Consider a typical run of a monocular system, as the one described in Section II, for a sequence of 1750 frames taken following a cycled trajectory (illustrated by the rectangle) in a laboratory environment (Fig. 4, upper). The final map contains about 350 features, and therefore, it has not been built in real time, but it is useful for illustrative purposes. It can be appreciated that there is a huge drift at the end of the estimation. Note that the map and trajectory (except for the side effects due to the second turn) are tolerably consistent. Consequently, it is important to observe that the uncertainty in the camera position is maintained stable over the whole sequence (Fig. 4, lower). Therefore, it does not reproduce the real error propagation. As a result, when the camera returns near to its initial position, the active search technique is neither able to predict the position of the old features nor able to find the closing loops after long trajectories to minimize the drift.

In addition, when new features have to be added to the map and the covariance matrix P is updated [(9), shown at the bottom of the page], a bias is introduced in the system [23] due to the nonlinearity in the measurement process. After several initializations, the bias introduced to the map could be substantial. This bias can also affect the active search.

Furthermore, when the camera moves far away from its initialization point and describes a different path, it is likely that the difference between the camera pose (when the features have first been observed) and the current camera pose is substantial, inducing huge changes in the image feature appearance due to variations in illumination or point of view. If the image feature appearance excessively differs, the patch cross-correlation technique will be unfeasible to address the data association problem. In this context, image feature descriptors [19], [20] are techniques adapted for addressing the data association problem. Nevertheless, descriptors are difficult to directly apply to monocular SLAM methods due to their high computational cost.

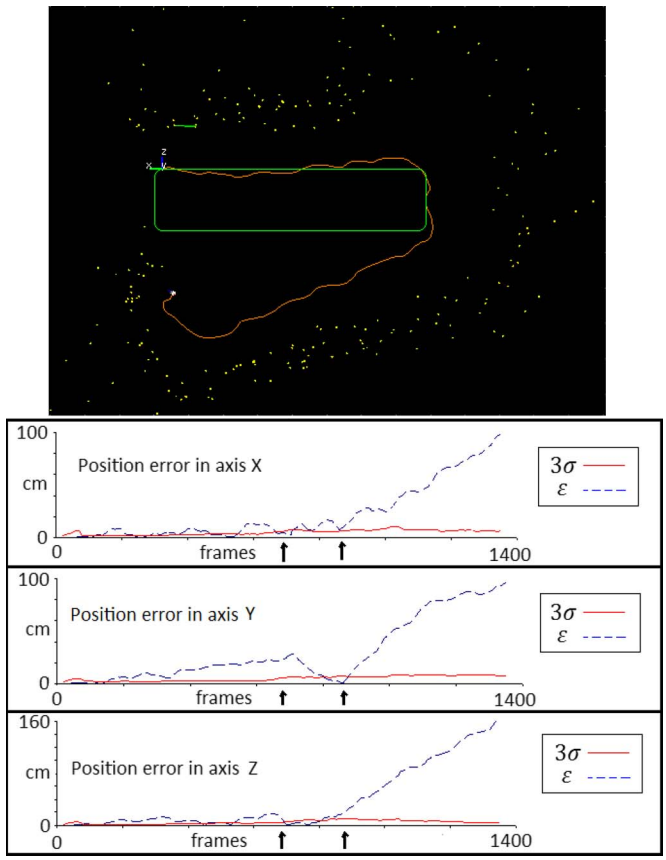

Fig. 4. Drift in monocular SLAM estimations. The upper plot illustrates the real and estimated camera trajectory for a sequence of 1700 frames. Lower plots show the estimation errors of camera position and their corresponding 3 σ variance for 1350 frames. Note that the second turn is the main reason for error propagation.

It is a clear tradeoff between the need of a high-frame-rate operation and the data association techniques required for addressing the whole problem.

IV. GS: Minimizing Drift

In the past, a single estimation process (EKF, particle filter, etc.) has been tried for solving the whole problem of single-camera SLAM. However, it seems that trying to solve the whole problem at once is a very hard task. As a possible alternative, a distributed approach is proposed in which the whole task is divided into two concurrent estimation processes, each one designed to handle in a natural way the previously mentioned tradeoff.

In the context of the GS process, the VS described in Section II is considered as a black box that provides

$$P = \begin{bmatrix} P_{xx} & P_{xy_1} & P_{xy_2} \\ P_{y_1x} & P_{y_1y_1} & P_{y_1y_2} \\ P_{y_2x} & P_{y_2y_1} & P_{y_2y_2} \end{bmatrix} \quad P_{\text{new}} = \begin{bmatrix} P_{xx} & P_{xy_1} & P_{xy_2} & P_{xx}\frac{\partial y_i^T}{\partial x_v} \\ P_{y_1x} & P_{y_1y_1} & P_{y_1y_2} & P_{y_1x}\frac{\partial y_i^T}{\partial x_v} \\ P_{y_2x} & P_{y_2y_2} & P_{y_2y_2} & P_{y_2x}\frac{\partial y_i^T}{\partial x_v} \\ \frac{\partial y_i}{\partial x_v}P_{xx} & \frac{\partial y_i}{\partial x_v}P_{xy_1} & \frac{\partial y_i}{\partial x_v}P_{xy_2} & \frac{\partial y_i}{\partial x_v}P_{xx}\frac{\partial y_i^T}{\partial x_v} + \frac{\partial y_i}{\partial h_i}R\frac{\partial y_i^T}{\partial h_i} \end{bmatrix} \tag{9}$$

appearance-based sensing in the form of feature descriptors. Furthermore, it emulates typical sensors such as laser for range measurements and encoders for dead reckoning.

A very simple scheme of SLAM was used for implementing the GS, but other methods such as [24] can be used.

For the interprocess communication, the GS is defined as a client connected to the VS. In the GS, an EKF is also used to propagate the camera pose and the map. When a Kalman step is completed, a request is sent to the VS to obtain the latest odometry and range information $[o_{vs}, s_1, s_2, \ldots, s_n]$, as well as the descriptors.

As far the GS is concerned, the camera-robot state is defined by

$$\hat{\mathrm{x}}_{\mathrm{v}} = [\, x \quad y \quad z \quad \theta \quad \varphi \,]^T \tag{10}$$

denoting the pose and orientation in the world coordinate frame. The prediction model $f_v(\hat{\mathrm{x}}_{v(k)}, u_{(k)})$ is

$$f_v = \begin{bmatrix} x_{(k+1)} \\ y_{(k+1)} \\ z_{(k+1)} \\ \theta_{(k+1)} \\ \varphi_{(k+1)} \end{bmatrix} = \begin{bmatrix} x_{(k)} + u_x \cos\left(\theta_{v(k)}\right) - u_z \sin\left(\theta_{v(k)}\right) \\ y_{(k)} + u_y \\ z_{(k)} + u_x \sin\left(\theta_{v(k)}\right) - u_z \cos\left(\theta_{v(k)}\right) \\ \theta_{(k)} + u_\theta \\ \varphi_{(k)} + u_\phi \end{bmatrix} \tag{11}$$

where $u_{(k)}$ is the control input

$$u_{(k)} = u_{n(k)} + v_{(k)} \tag{12}$$

and $u_{n(k)} = [\, u_x \quad u_y \quad u_z \quad u_\theta \quad u_\varphi \,]^T$ is taken from the data o_{vs} (7) sent by the VS, then $u_x = o_x$, $u_y = o_y$, $u_z = o_z$, $u_\theta = o_\theta$, and $u_\varphi = o_\varphi$, with $v_{(k)}$ being the noise added to the control input $u_{(k)}$ for modeling the uncertainty propagation of the VS odometry measurements. Then, v_k is defined as Gaussian noise with zero mean and a known diagonal covariance matrix U

$$U = diag\left(\sigma_x^2, \sigma_y^2, \sigma_z^2, \sigma_\theta^2, \sigma_\varphi^2\right). \tag{13}$$

In the experiments, variances are empirically tuned. Furthermore, in Fig. 4, it can be appreciated that the translational error is lower than the rotational error (a common behavior in visual SLAM). If it is assumed that the camera axis x is generally aligned with the translational camera movement (Fig. 3) (a common assumption in monocular SLAM using a wide-lens camera), then it can be assumed that $\sigma_x < \sigma_z$.

The complete state that includes the features $\hat{\mathrm{y}}$ is built as $\hat{\mathrm{x}} = [\hat{\mathrm{x}}_v^T, \hat{\mathrm{y}}_1^T, \ldots, \hat{\mathrm{y}}_n^T]^T$, where a feature $\hat{\mathrm{y}}_i$ represents a 3-D scene point i defined by $\hat{\mathrm{y}}_i = [x_i, y_i, z_i]$, denoting a Euclidean position in the world reference.

The observation model $h(x_k, w_k)$ predicts a measurement $z_k = [r \quad \theta \quad \varphi]^T$ of a 3-D point $\hat{\mathrm{y}}_i$ as follows:

$$h^C = \begin{bmatrix} h_x \\ h_y \\ h_z \end{bmatrix} = \begin{bmatrix} \sqrt{(x_i - x_v)^2 + (y_i - y_v)^2 + (z_i - z_v)^2} \\ \mathrm{atan2}(z_i - z_v, x_i - x_v) - \theta_v \\ \mathrm{atan2}\left(\sqrt{(x_i - x_v)^2 + (z_i - z_v)^2}, y_i - y_v\right) - \varphi_v \end{bmatrix} \tag{14}$$

Fig. 5. Real video, recorded in a laboratory environment, has been used in experiments. An inexpensive webcam is the only sensor system.

where w_k models uncertainty for the VS feature measurements, assuming Gaussian noise with zero mean and covariance matrix R.

If a feature measurement $z_i = [r \quad \theta \quad \varphi]^T$ incoming from the VS [represented by s_i in (8)] is not matched against a previously mapped feature, then it is initialized as a new feature $\hat{\mathrm{y}}_i$ in the map

$$\hat{\mathrm{y}}_{\mathrm{new}} = \begin{bmatrix} x_i \\ y_i \\ z_i \end{bmatrix} = \begin{bmatrix} x_v + r\sin(\varphi + \varphi_v)\cos(\theta + \theta_v) \\ y_v + r\cos(\varphi + \varphi_v) \\ z_v + r\sin(\varphi + \varphi_v)\sin(\theta + \theta_v) \end{bmatrix}. \tag{15}$$

When a feature is initialized its related d_n descriptors are stored in a database and labeled with its corresponding $\hat{\mathrm{y}}_i$ feature. To minimize the probabilities of mismatch, a matching of an incoming feature with an already-mapped one is only considered under the following circumstance: at least two different descriptors d_n from the measured feature must match with two different descriptors related with a single feature in the map. A fast approximate k-nearest neighbor technique is used for matching descriptors. A positive matching is considered if the Euclidean distance to the second d_{k2} nearest neighbor is shorter than 0.7 d_{k1} to the nearest neighbor [19].

V. Experimental Results

VS and GS are implemented in C++. Communications between these two items are implemented over TCP/IP, and they can run in the same computer or in different ones. In our experiments, two laptops connected in a local network are used to test the system. Network delays between the virtual and global sensors are very small and, therefore, insignificant; for this reason, they have been obviated. Nevertheless, if the method is extended to challenging networking cases (e.g., long-distance communications), more attention have to be put to the protocol implementation.

First, a video containing about 1750 frames has been recorded, walking over a predefined cycled trajectory inside a laboratory environment. A wide-lens camera has been used as the sensor. This video is used as the input of the VS. The system runs at video frame rate (real time), (Fig. 5). In this paper, the term "real time" means that the time elapsed between acquired

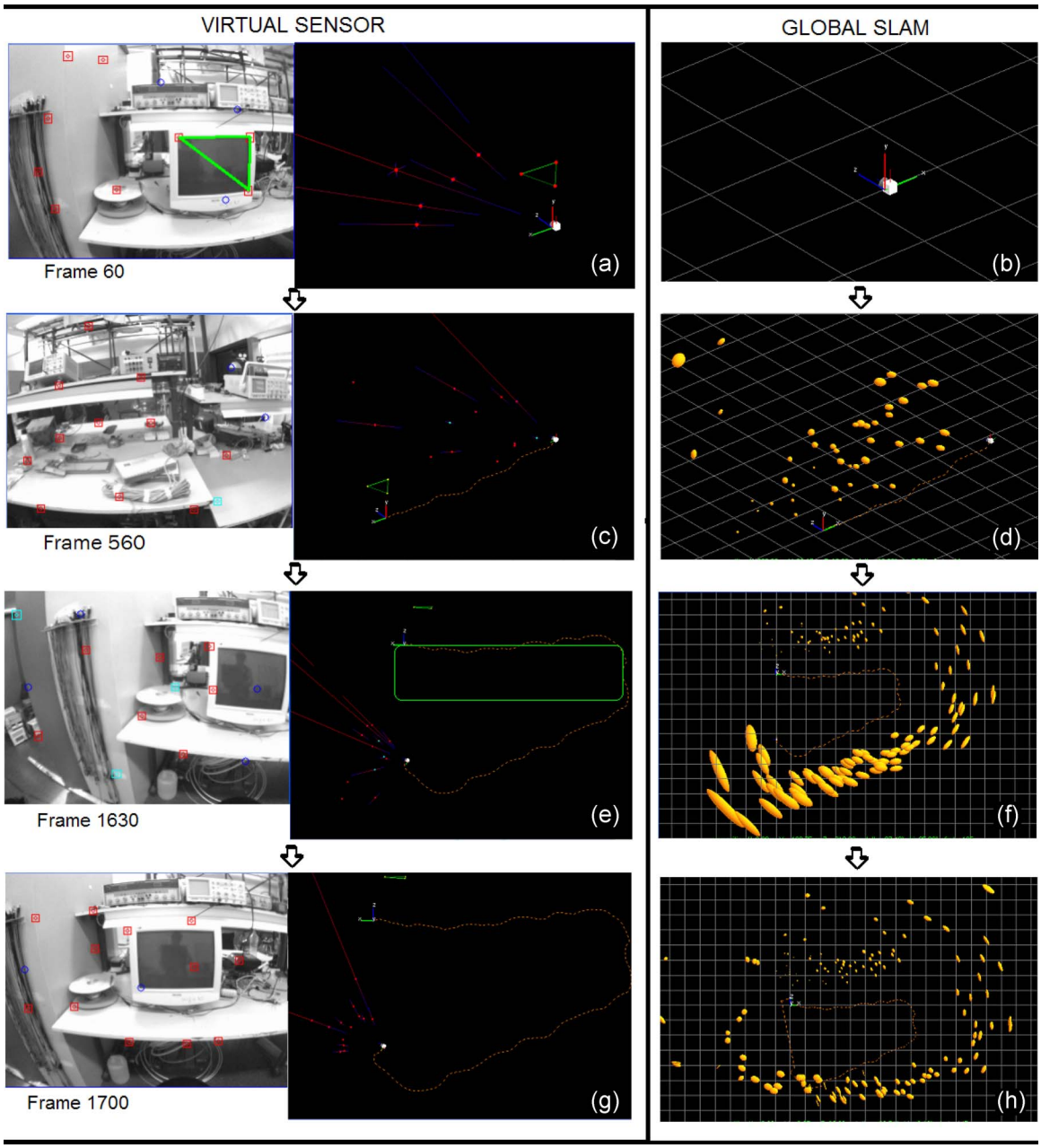

Fig. 6. Progression for the camera pose (frames 60, 560, 1630, and 1700) and map estimates. (a), (c), (e), and (g) VS. (b), (d), (f), and (h) GS. A video of 1700 frames has been captured following a predefined closed trajectory. Note that the VS map contains a stable number of features along the trajectory. At frame 1630, the drift in the trajectory and the uncertainty propagation (represented by the ellipses) in the GS features can be appreciated (xz *view*). At frame 1700, the GS has successfully detected some matches, the map has correctly been built, and the camera pose has been set right. Furthermore, note how the features uncertainties have been minimized.

camera frames (in the VS) is large enough to process a step of the algorithm in the VS and in the GS.

Fig. 6 shows the results for the distributed monocular SLAM system. Fig. 6(a), (c), (e), and (g) (left side) illustrates the output of the VS, and Fig. 6(b), (d), (f), and (h) (right side) shows the output of the GS, corresponding to frames 60, 560, 1630, and 1700. At frame 60, some features have been initialized in the VS [Fig. 6(a)]; the GS has not yet received information about these features [Fig. 6(d)] since their depth has not yet converged in the VS. Furthermore, note that the screen of a PC monitor has been used as the only reference to recover the metric world. The grid of the GS plots measures 0.5 m $\times$ 0.5 m. At frame 560, several features have been added to the GS map [Fig. 6(d)]. Note that the VS has dropped several features from the map to maintain a stable frequency operation [Fig. 6(c)]. At frame 1630 [Fig. 6(e) and (f)], there is a huge drift among the trajectory and the map. In Fig. 6(e), note the difference between the estimated trajectory and the real one (illustrated by the rectangle). In this case, it can be appreciated that turns are the main cause of the error propagation (note the turn at the second corner). Fig. 6(f) shows the uncertainty propagation for the measured features in the GS.

At frame 1700, the camera has returned near to its initial position, and the GS has successfully detected matches, and it nicely closes the loop. Fig. 6(h) shows the xz view of the final map and camera position estimates. It can be observed how the map has correctly been built, how the feature uncertainty has been minimized, and how the drift in the camera position has been fixed. At the end, about 360 features have been initialized in the VS (maintaining in the state an average of 20) on one hand. On the other hand, 147 features have been initialized in the GS. These latter features contain a lot of useful information for data association (descriptors).

VI. Conclusion and Future Work

A new distributed framework for addressing the general problem of 3-D visual SLAM using a single free moving camera has been presented in this paper. The key idea is to divide the whole task into two concurrent estimation processes.

One attribute of this approach is that it decouples the frequency operation of the input (camera) and the output (final camera position and map estimation). If only a single camera is to be used as sensory input, a high-frame-rate operation is desirable. However, when the map grows while maintaining the camera frequency operations, it becomes very difficult to keep the rate of processing. In the distributed scheme, the operation frequency of the GS does not depend on the operation frequency of the VS. Therefore, if the GS map grows, it could be updated at slower rates in relation to the camera frame rate. Monocular SLAM could profit from the use of different data association techniques such as active search and image feature descriptors. However, there is a tradeoff between the need for a high-frame-rate operation and the data association techniques required for addressing the whole problem. The proposed scheme naturally handles these data association techniques. The VS benefits from the use of the active search for matching features frame to frame at high operation frequency. The process of matching features using image descriptors (useful for detecting loops after long camera trajectories or when there is a huge drift in the estimation) is not necessarily implemented with an active image search approach, because descriptors representing the current detected features can be matched with a database of previously stored feature descriptors. This kind of matching approach is coherent with a general SLAM scenario (such as the GS), where observations come from an abstract sensor and the influx of observations does not depend on the estimation machinery.

In the experiments presented, real image sequences have been used. The C++ system implementation runs in real time. VS and GS reside in different local network PCs communicated by TCP/IP. The experimental results are positive. In the example presented in this paper, the map and camera trajectory estimated by the VS shows a huge drift, but it has successfully been minimized by the GS. The modularity of the method allows other techniques for implementing VS and GS. The presented framework is also suitable for cooperative mapping contexts using more than one VS and a single GS connected by a network.

Acknowledgment

The authors would like to thank the reviewers for their valuable suggestions. This research was conducted in the Automatic Control Department, Universitat Politecnica de Catalunya.

References

[1] H. Durrant-Whyte and T. Bailey, "Simultaneous localization and mapping: Part I," *IEEE Robot. Autom. Mag.*, vol. 13, no. 2, pp. 99–110, Jun. 2006.

[2] T. Bailey and H. Durrant-Whyte, "Simultaneous localization and mapping (SLAM): Part II," *IEEE Robot. Autom. Mag.*, vol. 13, no. 3, pp. 108–117, Sep. 2006.

[3] T. Pachidis and J. N. Lygouras, "Pseudostereo-vision system: A monocular stereo-vision system as a sensor for real-time robot applications," *IEEE Trans. Instrum. Meas.*, vol. 56, no. 6, pp. 2547–2560, Dec. 2007.

[4] H. Jin, P. Favaro, and S. Soatto, "A semi-direct approach to structure from motion," *Vis. Comput.*, vol. 19, no. 6, pp. 377–394, Oct. 2003.

[5] A. W. Fitzgibbon and A. Zisserman, "Automatic camera recovery for closed or open image sequences," in *Proc. ECCV*, 1998, p. 311.

[6] Y. K. Yu, K. H. Wong, S. H. Or, and M. Y. Chang, "Robust 3-D motion tracking from stereo images: A model-less method," *IEEE Trans. Instrum. Meas.*, vol. 57, no. 3, pp. 622–630, Mar. 2008.

[7] A. Davison, "Real-time simultaneous localization and mapping with a single camera," in *Proc. ICCV*, 2003, p. 1403.

[8] J. M. M. Montiel, J. Civera, and A. Davison, "Unified inverse depth parametrization for monocular SLAM," in *Proc. Robot.: Sci. Syst. Conf.*, 2006. [Online]. Available: http://www.roboticsproceedings.org/

[9] A. Davison, I. D. Reid, N. D. Molton, and O. Stasse, "MonoSLAM: Real-time single camera SLAM," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 29, no. 6, pp. 1052–1067, Jun. 2007.

[10] B. Williams, G. Klein, and I. Reid, "Real-time SLAM relocalisation," in *Proc. ICCV*, 2007, pp. 1–8.

[11] D. Chekhlov, M. Pupilli, W. Mayol-Cuevas, and A. Calway, "Real-time and robust monocular SLAM using predictive multi-resolution descriptors," in *Proc. 2nd Int. Symp. Vis. Comput.*, Nov. 2006, pp. 276–285.

[12] P. Smith, I. Reid, and A. Davison, "Real-time monocular SLAM with straight lines," in *Proc. BMVC*, 2006, pp. 17–26.

[13] E. Eade and T. Drummond, "Scalable monocular SLAM," in *Proc. IEEE Conf. CVPR*, 2006, pp. 469–476.

[14] J. Civera, A. Davison, and J. M. M. Montiel, "Inverse depth to depth conversion for monocular SLAM," in *Proc. IEEE ICRA*, 2007, pp. 2778–2783.

[15] R. Munguia and A. Grau, "Monocular SLAM for visual odometry," in *Proc. IEEE Int. Symp. WISP*, pp. 1–6.

[16] L. Clemente, A. Davison, I. Reid, J. Neira, and J. Tardos, "Mapping large loops with a single hand-held camera," in *Proc. Robot.: Sci. Syst. Conf.*, 2007. [Online]. Available: http://www.roboticsproceedings.org/

[17] A. Chiuso, P. Favaro, H. Jin, and S. Soatto, "MFm: 3-D motion from 2-D motion causally integrated over time," in *Proc. ECCV*, 2000, pp. 735–750.

[18] A. Davison and D. Murray, "Mobile robot localisation using active vision," in *Proc. ECCV*, 1998, pp. 809–825.

[19] H. Bay, T. Tuytelaars, and L. V. Gool, "SURF: Speeded up robust features," in *Proc. ECCV*, 2006, pp. 404–417.

[20] D. Lowe, "Distinctive image features from scale-invariant keypoints," *Int. J. Comput. Vis.*, vol. 60, no. 2, pp. 91–110, Nov. 2004.

[21] C. Valgren and A. Lilienthal, "SIFT, SURF and seasons: Long-term outdoor localization using local features," in *Proc. ECMR*, 2007. [Online]. Available: http://ecmr07.informatik.uni-freiburg.de/accepted_p.html

[22] S. Se, D. G. Lowe, and J. Little, "Mobile robot localization and mapping with uncertainty using scale-invariant visual landmarks," *Int. J. Robot. Res.*, vol. 21, no. 8, pp. 735–758, 2002.

[23] A. Davison and N. Kita, "Sequential localization and map-building for real-time computer vision and robotics," *Robot. Auton. Syst.*, vol. 36, no. 4, pp. 171–183, Sep. 2001.

[24] M. Montemerlo and S. Thrun, "Simultaneous localization and mapping with unknown data association using FastSLAM," in *Proc. IEEE ICRA*, 2003, pp. 1985–1991.

Rodrigo Munguía received the B.Eng. degree in engineering from Guadalajara University, Guadalajara Mexico, and the M.S. degree in computer science in 2006 from the Technical University of Catalonia (UPC), Barcelona, Spain, where he is currently working toward the Ph.D. degree with the Department of Automatic Control.

His research interests include computer vision and robotics, mostly focused in areas related to the simultaneous localization and mapping problem.

Antoni Grau received the M.S. and Ph.D. degrees in computer science from the Technical University of Catalonia (UPC), Barcelona, Spain, in 1990 and 1997, respectively.

He is currently a Professor with the Department of Automatic Control, UPC, giving lectures on computer vision, microprocessors, and peripheral devices. He is the Director of the "Control Engineering and Industrial Automation" postgraduate course. He is a coauthor of three books on programmable logic controllers and industrial communications. He is the author of more than 100 published papers. His research areas are computer vision, pattern recognition, robotics, factory automation, and education on sustainable development.

Dr. Grau is member of the International Association for Pattern Recognition and the International Federation of Automatic Control. He has chaired several international conferences.