Parsimonious real time monocular SLAM

Guillaume Bresson, Thomas Féraud, Romuald Aufrère, Paul Checchin, Roland Chapuis

Parsimonious Real Time Monocular SLAM

Guillaume Bresson*, Thomas Féraud*, Romuald Aufrère*†, Paul Checchin* and Roland Chapuis*
*Institut Pascal - UMR 6602 CNRS – †LIMOS - UMR 6158 CNRS
Clermont Université, Université Blaise Pascal - Aubière, France
firstname.name@univ-bpclermont.fr

***Abstract*—This paper presents a real time monocular EKF SLAM process that uses only Cartesian defined landmarks. This representation is easy to handle, light and consequently fast. However, it is prone to linearization errors which can cause the filter to diverge. Here, we will first clearly identify and explain when those problems take place. Then, a solution, able to reduce or avoid the errors involved by the linearization process, will be proposed. Combined with an EKF, our method uses resources parsimoniously by conserving landmarks for a long period of time without requiring many points to be efficient. Our solution is based on a method to properly compute the projection of a 3D uncertainty into the image frame in order to track landmarks efficiently. The second part of this solution relies on a correction of the Kalman gain that reduces the impact of the update when it is incoherent. This approach was applied to a real data set presenting difficult conditions such as severe distortions, reflections, blur or sunshine to illustrate its robustness.**

I. Introduction

In order to be autonomous, a vehicle must be able to localize itself in an unknown environment. This problem, known as the Simultaneous Localization And Mapping (SLAM) problem, has been extensively studied throughout the last two decades [11]. Though mathematical solutions have been proposed [10], they are usually not sufficient in real world applications. Moreover, they mostly consider range and bearing sensors [2]. Laser Range Finders (LRFs), for instance, are expensive and rarely furnish enough information to efficiently track a feature. These two constraints are essential and must be avoided in order to develop and spread the use of Intelligent Vehicles.

Conversely to LRFs, cameras are cheap and rich in terms of information. However, they add other constraints. First, cameras are bearing-only sensors which means that the depth of a landmark cannot be accurately estimated without having several observations with a sufficient parallax. Then, environmental factors can degrade the image quality: blur while turning, distortions, reflections or sunshine. These effects appear a lot especially in urban environments which are information rich. This is why a system must be tested under difficult conditions in order to prove its capabilities in real life applications.

To do so, it is necessary to take into account the way features are initialized and handled within the state vector. Indeed, with difficult conditions, it is obvious that landmarks will be discarded quickly. With a bearing-only sensor, it is an major issue as several observations of the same landmark are needed to have an accurate estimate of its position. Two main solutions exist in the literature: delayed and undelayed initializations. The first way is straightforward: it only uses well-defined landmarks [1][9]. This means that no wrong information is integrated into the state vector. However, it also means that not all the information available is utilized. During a bend, for example, landmarks will get out of the field-of-view without having been added to the state vector leading to a large uncertainty on the robot pose.

On the contrary, the undelayed approach proposes to integrate landmarks as soon as they are detected and then to refine their pose. It is thus possible to add a 3D point to the state vector right away, by simply creating it, with a big uncertainty, on the line-of-sight of its first 2D observation. Yet, issues will quickly appear during the linearization process which occurs while tracking a landmark or updating its position. This is due to the fact that the fictitious point initialized as the estimate could potentially be far from the real one [3]. Other landmark representations have been imagined to avoid this problem. In [17] and [22], the authors create multiple hypotheses for a single 3D point. While the landmark is re-observed, the wrong hypotheses are removed from the state vector until there is just one left. Nevertheless, this approach can be time consuming depending on how many hypotheses per landmark are considered. Most of the time, the Inverse Depth (ID) parametrization is chosen to characterize landmarks [8][18]. It consists of 6 parameters: 3 for the landmark and 3 others corresponding to the position of the camera during its first observation (anchor). As a consequence, this representation is also computationally costly since the size of the state vector is duplicated. It can be problematic with long distance SLAM which is our case. Usually, landmarks are transformed from the ID to the Cartesian representation after convergence [7]. In this paper, we will focus on how to solve or avoid linearization issues with a simple 3D Cartesian representation in order to have a system as fast (and memory thrifty) as possible.

SLAM is usually accomplished using Extended Kalman Filters (EKFs) [21], particle filters [12][23], bundle adjustment based methods [16] or Unscented Kalman Filters (UKFs) [6]. None of these methods is perfect as they all can be time and memory consuming. Nevertheless, EKFs have the advantage to be light as only several landmarks are needed to have an accurate localization, thus allowing to map large environments. Moreover, they give a direct access to the uncertainty of each point and make the integration of other sensors straightforward. Submapping methods [4][13] can also be utilized in order to limit the load of the filter.

Using the EKF in an Iterative version, we propose to validate our approach on real experiments with the already mentioned conditions (blur, distortions, reflections and sunshine). A complete SLAM, called Monocular SLAM (MSLAM), has

been developed to reach this goal. Our contributions are:

- Avoiding or reducing linearization issues during the tracking and update steps
- Combining the EKF with our corrections for a parsimonious use of resources (light maps, low processing time)
- Validating our approach with a trajectory presenting difficult conditions using a camera and proprioceptive sensors

The next section will explain how the linearization issues appear when performing a naive monocular SLAM. Then, Section III will expose the proposed corrections to avoid them. Next, Section IV will present the experiments realized to illustrate the efficiency of our approach. Finally, Section V will give some conclusions and perspectives concerning MSLAM.

II. Linearization issues in monocular SLAM

The linearization problems previously mentioned, are bound to the way points are initialized in the 3D world. Shi-Tomasi features [20] (in our case) are extracted from the image to obtain 2D landmarks. However, the depth of these points is unknown. Therefore, a fixed distance, associated to a big uncertainty, must be used to fully initialize a point.

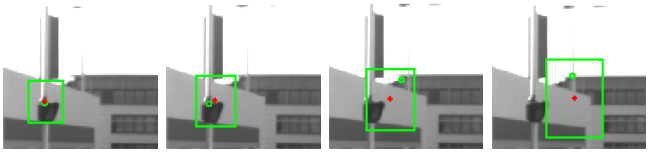
The classical way, using the EKF, would be to have an arbitrary uncertainty in which we are sure the point is. Then, thanks to an update with a null innovation, the EKF would be able to refine the initial uncertainty. However, the covariance would still encompass wrong locations (behind the observer for instance). To avoid these limitations, MSLAM builds the uncertainty depending on the knowledge it has about the point (unknown depth but accurate azimuth and elevation angles). More details can be found about the initialization process in [5].

Once the point is fully initialized, it is tracked in the next images. The different observations made will allow the EKF to refine the position of the landmark (especially its depth) in order to make it converge towards its real location. The tracking process starts by predicting the position of the 3D point in the image by projecting the last estimate available. The covariance associated is then used to obtain a window in the image where the point could potentially be.

Let $\hat{\mathbf{x}}$ be the state vector (including 6D camera pose and 3D landmark position). Let $h(\hat{\mathbf{x}}_{k|k-1})$ be the non-linear function (observation function) predicting the measurement from the predicted state $\hat{\mathbf{x}}_{k|k-1}$ at time k and $\mathbf{H}_k$ be the Jacobian associated to h. This Jacobian can be divided into two parts. The first one, $\mathbf{H}_{c,k}$, concerns the covariance of the mobile sensor position $\mathbf{P}_{c,k|k-1}$. The second part, $\mathbf{H}_{l_w,k}$, deals with the covariance of the landmark l in the world frame $\mathbf{P}_{l_w,k|k-1}$. With these notations, the covariance prediction of the landmark l in the image frame can be computed as follows:

$$\mathbf{P}_{l_i,k|k-1} = \mathbf{H}_{c,k}\mathbf{P}_{c,k|k-1}\mathbf{H}_{c,k}^T + \mathbf{H}_{l_w,k}\mathbf{P}_{l_w,k|k-1}\mathbf{H}_{l_w,k}^T \quad (1)$$

Nevertheless, the resulting 2D covariance can be wrong. Indeed, the linearization around the landmark position, induced by the Jacobian, can be erroneous if this position is far from the real one (meaning it has not converged yet). An illustration of this phenomenon can be found in Fig. 1. This figure shows the tracking of a 2D feature using Eq. (1). An evolution model using odometric information was utilized and no updates were performed.



(a) Frame 1: tracking okay (b) Frame 2: tracking okay (c) Frame 3: tracking failure (d) Frame 7: tracking failure

Fig. 1. Tracking failure due to significant linearization errors. The green rectangle is the bounding box of the covariance projection in the image (area where the point is supposed to be). The red cross is the estimate of the landmark position. The green circle is the observation.

Section III will present the method developed to compute a proper projection of the covariance in the image.

Obviously, linearization errors are also involved during the update step of the EKF. As a reminder, the equations related to the update are as follows:

$$\hat{\mathbf{x}}_{k|k} = \hat{\mathbf{x}}_{k|k-1} + \mathbf{K}_k(\mathbf{z}_k - h(\hat{\mathbf{x}}_{k|k-1})) \quad (2)$$

$$\mathbf{P}_{k|k} = \mathbf{P}_{k|k-1} - \mathbf{K}_k\mathbf{H}_k\mathbf{P}_{k|k-1} \quad (3)$$

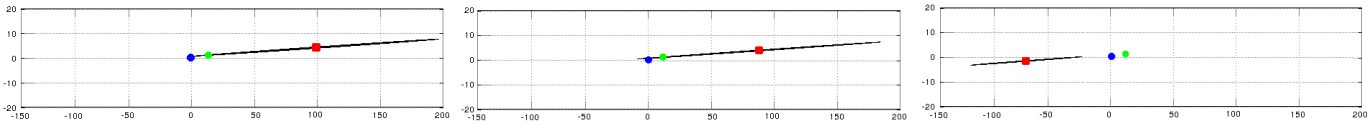
$$\mathbf{K}_k = \mathbf{P}_{k|k-1}\mathbf{H}_k^T(\mathbf{H}_k\mathbf{P}_{k|k-1}\mathbf{H}_k^T + \mathbf{R}_k)^{-1} \quad (4)$$

where $\mathbf{z}_k$ is the observation at time k, $\mathbf{K}_k$ is the Kalman gain and $\mathbf{R}_k$ is the observation noise. The problem here is similar to the previous one. Indeed, the use of the Jacobian in the Kalman gain can lead to incoherent cases where the landmark is updated behind the observer despite the fact that it has just been observed. The farthest the estimate is from the real point, the more wrong updates could happen. Figure 2 shows, with the same data set of Fig. 1, an example where the landmark new estimate is incoherent. The same feature was also used to initialize the 3D point (at 100 meters). When the 2D feature is re-observed, the EKF is able to update the corresponding 3D position. However, the second update is wrong because of severe linearization issues.

Linearization errors concerning the update have also been stated for the ID parametrization [19]. Though a solution is proposed by the authors, it simply consists in detecting and rejecting points updated behind the observer. The next section will present a solution to drastically reduce the impact of the linearization effect on the Kalman gain.

III. Proposed solution

The first part of the solution proposed in this paper concerns the tracking process. In our method, tangent planes to the uncertainty ellipsoid (covariance) of the landmarks are used. MSLAM computes the exact projection of the 3D uncertainty in the image plane. It does so by finding the planes tangent to the ellipsoid that give the maximum size of the corresponding uncertainty in the image. As performing correlation on a rectangular window is faster than on an ellipse, only the four tangent planes giving the rectangular bounding box of the ellipse are actually needed. A 2D example of the solution



(a) Initialization of the point at 100 meters on the line-of-sight

(b) First update: integrity is preserved

(c) Second update: landmark updated behind the observer

Fig. 2. Top view of a point updated behind the observer. The blue circle is the position of the vehicle. The red square is the landmark. The green circle is the real position of the landmark. The black ellipse is the uncertainty associated to the landmark after its initialization.

proposed here can be found in Fig. 3 as an illustration of the equations. Let $\mathbf{P}_{l_c}$ be the covariance matrix of a landmark in the camera frame (Fig. 3(a)):

$$\mathbf{P}_{l_c} = \begin{pmatrix} a & b & c \\ b & d & e \\ c & e & f \end{pmatrix}$$

Consequently:

$$\begin{aligned} \mathbf{P}_{l_c}^{-1} &= \frac{1}{\det \mathbf{P}_{l_c}} \begin{pmatrix} df-e^2 & ce-bf & be-cd \\ ce-bf & af-c^2 & cb-ae \\ be-cd & cb-ae & ad-b^2 \end{pmatrix} \\ &= \begin{pmatrix} A & B & C \\ B & D & E \\ C & E & F \end{pmatrix} \end{aligned}$$

In order to find the tangent planes, points on the surface of the ellipsoid $\mathscr{E}$, generated by $\mathbf{P}_{l_c}$, must be kept. They can be found thanks to the Mahalanobis distance (Fig. 3(b)):

$$\begin{pmatrix} x-x_0 \\ y-y_0 \\ z-z_0 \end{pmatrix}^T \mathbf{P}_{l_c}^{-1} \begin{pmatrix} x-x_0 \\ y-y_0 \\ z-z_0 \end{pmatrix} = 1 \tag{5}$$

where $\begin{pmatrix} x_0 & y_0 & z_0 \end{pmatrix}^T$ is the center of $\mathscr{E}$. The tangent planes to the surface of the ellipsoid are also orthogonal to the normals of those points (Fig. 3(c)). This constraint can be expressed thanks to the gradient:

$$\begin{aligned} \vec{n} &= \nabla \mathscr{E}(x,y,z) = \begin{pmatrix} \frac{\partial \mathscr{E}}{\partial x} & \frac{\partial \mathscr{E}}{\partial y} & \frac{\partial \mathscr{E}}{\partial z} \end{pmatrix}^T \\ &= \begin{pmatrix} 2(x-x_0)A + 2(y-y_0)B + 2(z-z_0)C \\ 2(x-x_0)B + 2(y-y_0)D + 2(z-z_0)E \\ 2(x-x_0)C + 2(y-y_0)E + 2(z-z_0)F \end{pmatrix} \end{aligned} \tag{6}$$

Points part of a plane p, orthogonal to the normal, verify the following relationship (Fig. 3(d)):

$$\begin{pmatrix} x_p \\ y_p \\ z_p \end{pmatrix} \in p \Leftrightarrow \begin{pmatrix} x_p - x \\ y_p - y \\ z_p - z \end{pmatrix}^T \cdot \vec{n} = 0 \tag{7}$$

The planes must obviously pass by the origin (camera). This allows to simplify Eq. (7) (Fig. 3(e)):

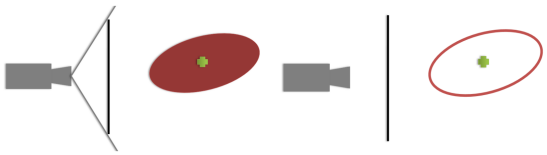
$$\begin{pmatrix} x & y & z \end{pmatrix} \cdot \vec{n} = 0 \tag{8}$$

In the 2D case, the constraints previously defined are sufficient (Fig. 3(f)). However, in 3D, there is still an infinite number of solutions. It is necessary to add another constraint. As we are looking for a rectangular bounding box, it is possible to discard all the points whose tangent planes are not intersecting the image plane vertically or horizontally. The vertical planes contain the $\vec{z}$ axis and the horizontal ones the $\vec{y}$ axis. Thus, for the points belonging to the vertical planes:

$$\vec{z} \cdot \vec{n} = 0 \Leftrightarrow (x-x_0)C + (y-y_0)E + (z-z_0)F = 0 \tag{9}$$

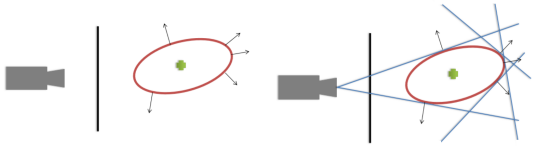
and for those part of the horizontal planes:

$$\vec{y} \cdot \vec{n} = 0 \Leftrightarrow (x-x_0)B + (y-y_0)D + (z-z_0)E = 0 \tag{10}$$



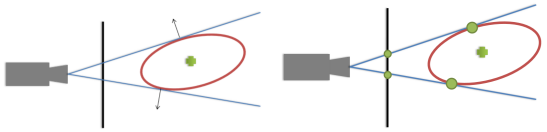
(a) Camera, image, landmark (green) and its uncertainty (red)

(b) Outline of the ellipse



(c) Normals of those points (examples)

(d) Tangent planes associated (in blue)



(e) Tangent planes passing by the origin

(f) Solution in 2D (projection of the points in the image)

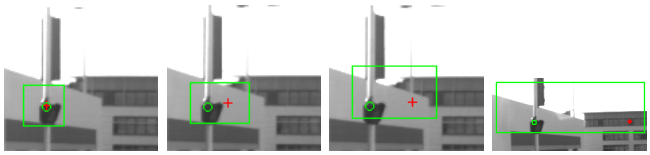
Fig. 3. 2D example of the constraints used to compute a bounding box in the 3D case.

To sum up, these constraints give the following systems to solve (respectively for the vertical and horizontal planes):

$$\begin{cases} \begin{pmatrix} x-x_0 \\ y-y_0 \\ z-z_0 \end{pmatrix}^T \mathbf{P}_{l_c}^{-1} \begin{pmatrix} x-x_0 \\ y-y_0 \\ z-z_0 \end{pmatrix} = 1 \\ \begin{pmatrix} x & y & z \end{pmatrix} \cdot \vec{n} = 0 \\ (x-x_0)C + (y-y_0)E + (z-z_0)F = 0 \end{cases} \tag{11}$$

$$\begin{cases} \begin{pmatrix} x-x_0 \\ y-y_0 \\ z-z_0 \end{pmatrix}^T \mathbf{P}_{l_c}^{-1} \begin{pmatrix} x-x_0 \\ y-y_0 \\ z-z_0 \end{pmatrix} = 1 \\ \begin{pmatrix} x & y & z \end{pmatrix} \cdot \vec{n} = 0 \\ (x-x_0)B + (y-y_0)D + (z-z_0)E = 0 \end{cases} \tag{12}$$

Resolving these systems is trivial and gives a couple of 3D points on the ellipsoid matching the vertical constraint and another couple for the horizontal one. By projecting them into the image, we obtain the bounding box limits. The advantage of this method is that it avoids the linearization approximations that can cause the system to search for a point at the wrong place. Here, the bounding box is exactly fitting the projection of the ellipsoid into the image. Thus, it is possible to track a landmark much more longer than before. Figure 4 shows, with the same set of images used in Fig. 1, how, when linearization issues are avoided, our algorithm is able to track a landmark successfully (still with no updates).



(a) Frame 1: tracking okay (b) Frame 3: tracking okay (c) Frame 7: tracking okay (d) After 18 frames

Fig. 4. Tracking with the method proposed in this paper. The green rectangle is the predicted window where the landmark is supposed to be. The red cross is the estimate of the landmark position. The green circle is the observation (patch matched with NCC).

As stated before, the update step can also raise some linearization issues. In order to reduce their impact, a corrective of the Kalman gain was introduced in [14] and [15]. The key element of this method is to know, whether or not, the update is coherent. With the kind of initialization MSLAM is using, the best information it has about a landmark is its observation. This can be explained by the fact that the error made on an observation is very small compared to the one of the 3D estimate. This means that, once updated, the projection of a landmark is compulsorily between the observation and the projection of the prediction used to update it. Therefore, the corrective factor will only be applied when the landmark is out of these bounds. When it is so, the idea is to have the following relationship verified:

$$\mathbf{z}_k = h(\hat{\mathbf{x}}_{k|k}) \tag{13}$$

Let $\mathbf{\Omega}_k$ be the Kalman gain once the corrective factor r has been applied to make Eq. (13) true. It is thus defined as follows:

$$\mathbf{\Omega}_k = r \cdot \mathbf{K}_k \tag{14}$$

In a monocular SLAM, with a landmark $\mathbf{x}_l = \begin{pmatrix} x_l & y_l & z_l \end{pmatrix}^T$ in the world frame, the observation function can be expressed as:

$$\begin{cases} u_{est} = \dfrac{\mathbf{F}_1 \mathbf{R}_{cw}^T (\mathbf{x}_l - \mathbf{t}_{cw})}{\mathbf{F}_3 \mathbf{R}_{cw}^T (\mathbf{x}_l - \mathbf{t}_{cw})} \\ v_{est} = \dfrac{\mathbf{F}_2 \mathbf{R}_{cw}^T (\mathbf{x}_l - \mathbf{t}_{cw})}{\mathbf{F}_3 \mathbf{R}_{cw}^T (\mathbf{x}_l - \mathbf{t}_{cw})} \end{cases} \tag{15}$$

where u_{est} and v_{est} are the estimated position of the landmark projected into the image, $\mathbf{F}_i$ is the i^{th} line of the intrinsic parameters matrix, $\mathbf{R}_{cw}$ is the rotation matrix passing points from the camera frame to the world frame and $\mathbf{t}_{cw}$ is the translation associated to the rotation. For an observation $\mathbf{z}_k = \begin{pmatrix} u_{obs} & v_{obs} \end{pmatrix}^T$ and by taking into account (13) and (14), we can write:

$$\begin{cases} u_{obs} = \dfrac{\mathbf{F}_1 \mathbf{R}_{cw}^T (\mathbf{x}_l + \mathbf{\Omega\Delta} - \mathbf{t}_{cw})}{\mathbf{F}_3 \mathbf{R}_{cw}^T (\mathbf{x}_l + \mathbf{\Omega\Delta} - \mathbf{t}_{cw})} \\ v_{obs} = \dfrac{\mathbf{F}_2 \mathbf{R}_{cw}^T (\mathbf{x}_l + \mathbf{\Omega\Delta} - \mathbf{t}_{cw})}{\mathbf{F}_3 \mathbf{R}_{cw}^T (\mathbf{x}_l + \mathbf{\Omega\Delta} - \mathbf{t}_{cw})} \end{cases} \tag{16}$$

with the innovation $\mathbf{\Delta} = \mathbf{z}_k - h(\hat{\mathbf{x}}_{k|k-1})$ and $\mathbf{\Omega}$ the corrected Kalman gain associated to $\mathbf{x}_l$. This gives two potential corrections for the Kalman gain:

$$\begin{cases} r_u = \dfrac{(u_{obs} - u_{est}) \mathbf{F}_3 \mathbf{R}_{cw}^T (\mathbf{x}_l - \mathbf{t}_{cw})}{(\mathbf{F}_1 - u_{obs} \mathbf{F}_3) \mathbf{R}_{cw}^T \mathbf{K\Delta}} \\ r_v = \dfrac{(v_{obs} - v_{est}) \mathbf{F}_3 \mathbf{R}_{cw}^T (\mathbf{x}_l - \mathbf{t}_{cw})}{(\mathbf{F}_2 - v_{obs} \mathbf{F}_3) \mathbf{R}_{cw}^T \mathbf{K\Delta}} \end{cases} \tag{17}$$

A scalar superior to 1 would mean that the projected update is between the defined bounds. In that case MSLAM does not apply the corrective factor, otherwise it would lead to an overconfident estimation. For the same reason, the algorithm will keep the lowest value between r_u and r_v leading to $r = min(r_u, r_v, 1)$. Thanks to this corrective factor, linearization errors are greatly reduced avoiding thus a landmark to be updated behind the observer (cf. Fig. 2). In a few iterations, the landmark will now converge towards its real value.

IV. Experiments

The validation of our algorithm with real data is an extremely important step. Nowadays, it is essential to be able to easily deploy an algorithm in various conditions. Moreover, some constraints (real time execution for instance) can be imposed by the nature of the applications aimed. In order to be as close as possible to a real scenario, the experiments were accomplished in a realistic urban platform called PAVIN. It recreates a human environment with paved roads, crosswalks, roundabouts, building façades... allowing to test MSLAM in real conditions. An electrical vehicle (VIPALAB), equipped with a single camera and proprioceptive sensors providing the odometry and the steering angle, was used. A Real Time Kinematic GPS (RTK GPS) was also provided but only served as a ground truth for comparison purposes. The camera was located inside the passenger cell, on top of it, and running at 7.5 frames per second. A trajectory of 170 meters was performed with a velocity going from null to almost 2 meters per second (the major part of the trajectory is approximately at 1 meter per second). Data association was accomplished using Normalized Cross Correlation (NCC) between a patch of 15x15 pixels, extracted around the first observation of the landmark (Shi-Tomasi features), and the computed bounding box.

The different conditions, mentioned in the introduction, are exposed in Fig. 5 which also provides an overview of the environment used for the experiments. At different moments

of the trajectory, reflections will appear. On the left-bottom part of each image composing the trajectory, an artifact, due to a reflection on the front window, is visible. This means that landmarks initialized there must not be kept otherwise they would make the SLAM diverge. This is why the tracking process has to be good in order not to match them while the vehicle is moving forward. Distortions, though quite severe here, are not taken into account. MSLAM has to cope with it by being sure that it can track a landmark in any way. Blur while turning is a very common phenomenon that requires once again to be able to accurately predict where the landmark is going to be located in order to potentially match it. Finally, sunshine, as it is depicted in Fig. 5, is difficult to handle in the tracking process because it modifies the landmarks aspect preventing the algorithm to match them properly. All these conditions require the algorithm to be as robust as possible. Moreover, it must be able to initialize new landmarks very quickly to avoid losing the localization.

The trajectory is composed of various bends, including a roundabout, and straight lines. This allows to confirm that MSLAM is able to track successfully landmarks during straight lines while still being able to initialize and take advantage of new points during bends. A naive implementation of the monocular SLAM has also been used for comparison purposes. It does not apply the corrections presented in this paper. This means that tracking windows can be incoherent and that the Kalman gain is not corrected when an update is wrong. Figure 6 shows the results for both algorithms with a maximum of 10 landmarks per image (new landmarks are initialized only when less than 5 are tracked successfully).

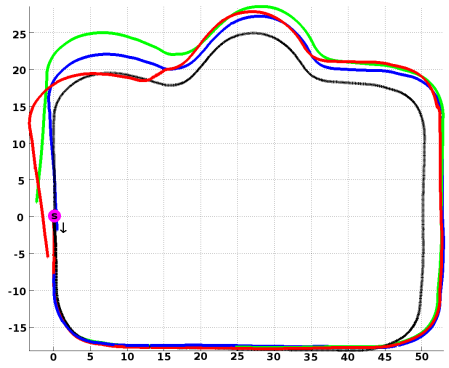


Fig. 6. Localization of the vehicle. **Black:** ground truth (RTK GPS). **Green:** prediction only (based on proprioceptive sensors). **Red:** naive EKF SLAM. **Blue:** MSLAM.

We can see that the localization obtained by MSLAM is better than the prediction made by the proprioceptive sensors and than the naive SLAM. The angles, during the bends, are properly corrected. However, the drift induced by the odometers is not corrected. Indeed, the scale factor of the camera is constrained only by the odometric information. This means that a better estimation of the translation cannot be achieved without performing a loop closure. The naive implementation slightly diverges because of linearization issues. However, it is still quite good because the system is well-constrained and a failure in the tracking or during the update is simply not taken into account (the related 3D point is deleted) whereas MSLAM corrects it. In order to better notice the advantage of the method presented in this paper, the number of landmarks which have been initialized and have finally converged is a good indicator. Indeed, with no corrections, landmarks convergence is more difficult to achieve and consequently, more landmarks are going to be discarded during the process (due to linearization issues). The results for both implementations are visible in Table I.

	Naive SLAM	MSLAM
Number of landmarks initialized	472	408
Number of landmarks conserved	267	354

TABLE I
NUMBER OF LANDMARKS.

With MSLAM, less landmarks are initialized and more have converged compared to the naive implementation. These results were expected and clearly show that linearization issues are avoided or reduced. MSLAM took 15 ms in the mean for each image (the proposed corrections took less than 2 μs in the mean for each landmark). However, it depends if new points must be initialized. In that case, Shi-Tomasi detections must be made on the whole image increasing the time required to approximately 55 ms. In the experiments presented here, the application was running in real time and no image or odometric information was missed.

The best way to illustrate the robustness of MSLAM is to work with a slower camera or a faster vehicle. In our case, we used the experiments already exposed here and simply processed one image out of two, simulating a lower frequency of the sensor (or a higher speed of the vehicle). The results of the localization are presented in Fig. 7.

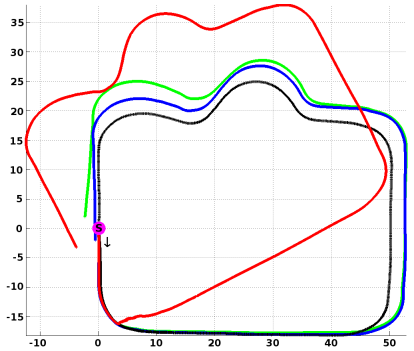


Fig. 7. Localization of the vehicle with less images. **Black:** ground truth (RTK GPS). **Green:** prediction only (based on proprioceptive sensors). **Red:** naive EKF SLAM at 3.75 FPS. **Blue:** MSLAM at 3.75 FPS.

It can be noticed that the results are similar to those of Fig. 6. MSLAM is still able to track and update landmarks properly whereas the naive implementation fails during the first bend. With less images, the main problem is the data association algorithm. Indeed, between successive frames the images can change very quickly (due to illuminations or simply to the movement of the vehicle). This causes the

(a) Reflections (b) Distortions (c) Blur (d) Sunshine

Fig. 5. Some examples of the trajectory made on PAVIN. The different conditions that our algorithm has been exposed to are depicted here.

matching to frequently fail, forcing the initialization of new points. A better data association could avoid this problem.

V. Conclusion

A monocular EKF SLAM avoiding or reducing linearization issues while using only Cartesian defined landmarks has been presented. Combined with an EKF, our approach makes a parsimonious use of the available resources by needing only a few landmarks, and therefore a low computational time, while still achieving good localization results. A direct consequence is the possibility to use MSLAM in long trajectories since the maps built are light. This approach was tested with difficult outdoor conditions such as severe distortions, reflections, blur or sunshine.

During the prediction of the landmarks covariances in the image plane, linearization errors are avoided by computing a bounding box that perfectly fits the 2D uncertainty. Concerning the update state, a corrective factor is applied to the Kalman gain to reduce its impact when the updated position of the landmark is incoherent (behind the observer for instance).

The evaluation of the SLAM process over 170 meters, with different conditions, showed its efficiency. Indeed, it performed better than a naive implementation while requiring less landmarks. This approach is real time and can be easily applied in various outdoor environments.

A more robust data association will be investigated to avoid the loss of features when landmarks aspect has changed too much. Another perspective is to test the robustness of our system under other constraints such as a greater velocity. A loop closure method would also be an important asset in the future.

References

[1] T. Bailey. Constrained Initialisation for Bearing-Only SLAM. In *IEEE International Conference on Robotics and Automation*, volume 2, pages 1966–1971, 2003.

[2] T. Bailey and H. Durrant-Whyte. Simultaneous Localization and Mapping (SLAM): Part II. *IEEE Robotics and Automation Magazine*, 13(3):108–117, 2006.

[3] K. E. Bekris, M. Glick, and L. E. Kavraki. Evaluation of Algorithms for Bearing-Only SLAM. In *IEEE International Conference on Robotics and Automation*, pages 1937–1943, 2006.

[4] M. Bosse, P. Newman, J. Leonard, M. Soika, W. Feiten, and S. Teller. An Atlas Framework for Scalable Mapping. In *IEEE International Conference on Robotics and Automation*, volume 2, pages 1899–1906, 2003.

[5] G. Bresson, T. Féraud, R. Aufrère, P. Checchin, and R. Chapuis. A New Strategy for Feature Initialization in Visual SLAM. In *IEEE/RSJ International Conference on Intelligent Robots and Systems Workshop on Perception and Navigation for Autonomous Vehicles in Human Environment*, pages 115–120, 2011.

[6] D. Checklov, M. Pupilli, W. Mayol-Cuevas, and A. Calway. Real-Time and Robust Monocular SLAM using Predictive Multi-Resolution Descriptors. *Advances in Visual Computing*, 4292:276–285, 2006.

[7] J. Civera, A. J. Davison, and J. M. M. Montiel. Inverse Depth to Depth Conversion for Monocular SLAM. In *IEEE International Conference on Robotics and Automation*, pages 2778–2783, 2007.

[8] J. Civera, A. J. Davison, and J. M. M. Montiel. Inverse Depth Parametrization for Monocular SLAM. *IEEE Transactions on Robotics*, 24(5):932–945, 2008.

[9] A. J. Davison. Real-Time Simultaneous Localisation and Mapping with a Single Camera. In *IEEE International Conference on Computer Vision*, pages 1403–1410, 2003.

[10] M. W. M. G. Dissanayake, P. Newman, H. F. Durrant-Whyte, S. Clark, and M. Csorba. A Solution to the Simultaneous Localization and Map Building (SLAM) Problem. *IEEE Transactions on Robotics and Automation*, 17(3):229–241, 2001.

[11] H. Durrant-Whyte and T. Bailey. Simultaneous Localization and Mapping: Part I. *IEEE Robotics and Automation Magazine*, 13(2):99–110, 2006.

[12] E. Eade and T. Drummond. Scalable Monocular SLAM. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, volume 1, pages 469–476, 2006.

[13] C. Estrada, J. Neira, and J. D. Tardós. Hierarchical SLAM: real-time accurate mapping of large environments. *IEEE Transactions on Robotics*, 21(4):588–596, 2005.

[14] T. Féraud, R. Chapuis, R. Aufrère, and P. Checchin. Improving Results of Rational Non-Linear Observation Functions Using a Kalman Filter Correction. In *International Conference on Information Fusion*, 2011.

[15] T. Féraud, R. Chapuis, R. Aufrère, and P. Checchin. Kalman Filter Correction with Rational Non-Linear Functions: Application to Visual-SLAM. In *European Conference on Mobile Robots*, pages 232–238, 2011.

[16] G. Klein and D. Murray. Parallel Tracking and Mapping for Small AR Workspaces. In *6th IEEE and ACM International Symposium on Mixed and Augmented Reality*, pages 1–10, 2007.

[17] N. M. Kwok and G. Dissanayake. An Efficient Multiple Hypothesis Filter for Bearing-Only SLAM. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 736–741, 2004.

[18] J. Montiel, J. Civera, and A. J. Davison. Unified Inverse Depth Parametrization for Monocular SLAM. In *Robotics: Science and Systems*, 2006.

[19] M. P. Parsley and S. J. Julier. Avoiding Negative Depth in Inverse Depth Bearing-Only SLAM. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 2066–2071, 2008.

[20] J. Shi and C. Tomasi. Good Features to Track. In *Computer Vision and Pattern Recognition*, pages 593–600, 1994.

[21] R. Smith, M. Self, and P. Cheeseman. A Stochastic Map for Uncertain Spatial Relationships. In *4th International Symposium on Robotics Research*, pages 467–474, 1988.

[22] J. Solà, A. Monin, M. Devy, and T. Lemaire. Undelayed Initialization in Bearing Only SLAM. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 2499–2504, 2005.

[23] S. Thrun, M. Montemerlo, D. Koller, B. Wegbreit, J. Nieto, and E. Nebot. FastSLAM: An Efficient Solution to the Simultaneous Localization And Mapping Problem with Unknown Data Association. *Journal of Machine Learning Research*, 4(3):380–407, 2004.