A Hybrid Structure/Trajectory Constraint for Visual SLAM

Angelique Loesch, Steve Bourgeois, Vincent Gay-Bellile
CEA LIST, Point Courrier 94, Gyf-sur-Yvette, F-91191 France
{angelique.loesch, steve.bourgeois, vincent.gay-bellile}@cea.fr

Michel Dhome
Pascal Institute, Blaise Pascal University, Clermont-Ferrand, FR
michel.dhome@univ-bpclermont.fr

Abstract

This paper presents a hybrid structure/trajectory constraint, that uses output camera poses of a model-based tracker, for object localization with SLAM algorithm. This constraint takes into account the structure information given by a CAD model while relying on the formalism of trajectory constraints. It has the advantages to be compact in memory and to accelerate the SLAM optimization process. The accuracy and robustness of the resulting localization as well as the memory and time gains are evaluated on synthetic and real data. Videos are available as supplementary material.

1. Introduction

Monocular model-based tracking relies on camera pose estimation with respect to an object of interest. This technique exploits 3D features extracted from a model, that are matched with image features in the image of a video stream [6]. However the accuracy and robustness of the camera localization depend on the visibility of the object as well as on the motion of the camera.

To better constrain the localization, recent solutions rely on environment features that are reconstructed online, in addition to the model ones. These approaches combine SLAM and model-based tracking solutions by using structure constraints from the 3D model of the object of interest. This model can indeed be used to constrain the SLAM initialization [1] or their optimization process (Bundle Adjustment or BA) [18, 8, 16, 10]. Adding this constraint in the Bundle Adjustment process of the SLAM results in a drift free localization. However, such approaches can suffer from high memory consumption compared to usual SLAM process due to the storage of additional data. This problem is particularly penalizing for methods using a textureless CAD model as *a priori* model, such as [18, 8].

In this article, a new solution is proposed to constrain a SLAM with a textureless CAD model, the constraint exploiting the output pose of a model-based tracking algorithm. This method provides the same level of accuracy and robustness than [18, 8], and reduces drastically the memory footprint (as illustrated on Figures 1(c) and 1(d)).

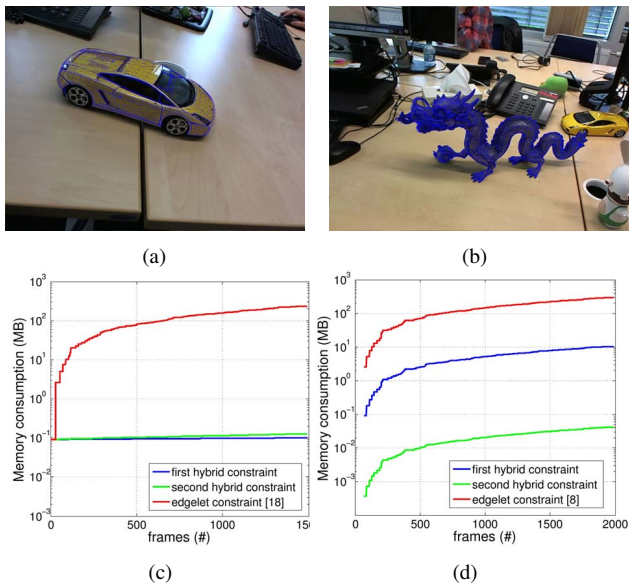


Figure 1: Our solution tracks with accuracy polyhedral (a) or curved (b) objects. The memory footprint associated to the car sequence (c) and the dragon one (d), is drastically decreased with the use of the first and second hybrid constraint (see Section 4) compared to edgelet constraint [18, 8]. The memory consumption is expressed in MB with a logarithmic scale on the *y* axis.

978-1-5090-5407-7/16 $31.00 © 2016 IEEE
DOI 10.1109/3DV.2016.12

2. Related works

Simultaneous Localization And Mapping (SLAM) solutions [4, 11] are iterative processes that reconstruct 3D primitives with respect to 2D observations and estimate the camera pose for each image thanks to 2D/3D correspondences between the previously reconstructed primitives and those extracted from the images. To limit the error accumulation, a possible solution is to use a non-linear optimization process called Bundle Adjustment. It refines camera poses and 3D feature points by minimizing the re-projection errors [20]. Standard SLAM algorithms [4, 11] are often used to estimate the localization of a camera in an unknown environment. However, they are not well adapted for object tracking. Camera poses are indeed expressed in an arbitrary coordinate frame and with an arbitrary scale that is subject to drift over time. Thus, the idea to use constrained Bundle Adjustments (cBA) that minimize simultaneously the multi-view geometry and a constraint term has been developed. Two categories of constraints are proposed in the literature to limit error accumulation by constraining different Degrees of Freedom (DoF) of SLAM algorithms: structure and trajectory constraints.

On the one hand, structure constraints exploit an *a priori* partial knowledge of the scene geometry, usually a 3D model. They can be expressed with a metric error [16] corresponding to the 3D distance between 3D features reconstructed by SLAM and the model of the object. However combining this error with a pixel one in the cBA requires an adaptive weight to deal with heterogeneous measurements. Structure constraints can also be expressed through a re-projection error of 3D features reconstructed by SLAM and projected on the model while some of their DoF are fixed during the optimization [9, 19]. The constraint can furthermore be defined with a re-projection error between 3D features extracted from the model and their associated image observations. These features are considered as an absolute information, and are then fixed in the Bundle Adjustment process. They may come from a 3D point cloud [10] or, when objects of interest are textureless, correspond to 3D oriented points (called edgelets) from edges of a CAD model [18, 8]. Edgelet-constrained approaches are a generic solution, since they can deal with a large set of objects. Their associated cost function measures the distance between reprojected edgelets and the closest image contours along their normals. This cost function is not compact in memory due to contour images that have to be stored. Moreover sets of dynamic edgelets extracted at each key-frame are also saved when occluding contours are used [8]. Therefore, memory consumption grows with the duration and the resolution of the video. Besides, this edgelet constraint is time consuming since the 2D/3D correspondences have to be re-estimated at each cBA on all the optimized key-frames.

On the other hand, trajectory constraint uses absolute pose information to limit the SLAM drift with an error term measuring the distance between poses. This error is more compact in memory than structure constraints but is not formalized as a re-projection error. The cost function integrating the pose constraint mixes metric and angular errors from position and orientation parameters of the camera poses. Thus adapted weights, usually defined with the covariance matrix of the absolute pose, are added. This allows to take into account the DoF that are poorly constrained.

The absolute poses used by trajectory constraint can be provided by a re-localization algorithm [10], a GPS or other sensors [5, 7, 17]. However these poses correspond to the output of a pre-processing that may be inaccurate. For example, the pose provided by a re-localization algorithm is estimated through a set of 2D/3D correspondences that contains outliers. To better deal with the inaccuracy of the absolute poses, Lhuillier [7] proposes a two-step optimization process. It firstly refines the re-projection error of the reconstructed 3D points without constraints and then optimizes the constraints with a regularization term based on the results of the first optimization. The regularization term guarantees that the SLAM trajectory is not deteriorated by inaccurate poses. More details about its approach are given in Section 3.3.

Trajectory constraints are less expensive in memory than structure ones. However their implementation have to deal with absolute pose inaccuracy and heterogeneous error terms.

In this paper, we propose a new approach to constrain a SLAM with a CAD model. Compared to edgelet-constrained SLAM (edgelet-cSLAM) [18, 8], our solution reduces the memory footprint while ensuring a similar accuracy. Our first contribution consists in defining a SLAM that is constrained to a textureless CAD model through a trajectory constraint (Section 3). A such approach reduces the memory footprint but also deteriorates the accuracy. Therefore, to prevent this loss of accuracy, our second contribution consists in replacing the usual trajectory constraint by a hybrid structure/trajectory constraint that evaluates the deviation to the reference trajectory through a structure error (section 4). Our tracking solution with a low memory footprint is then evaluated on several polyhedral and curved objects, and compared with edgelet-cSLAM [18, 8] on synthetic and real data in Section 5. A conclusion is given in Section 6.

Notations. Matrices are designated by sans-serif capital font such as M and vectors by bold font such as $\mathbf{v}$ or $\mathbf{V}$. The projection matrix P associated to a camera is given by $\mathsf{P} = \mathsf{K}\mathsf{R}^\top(\mathsf{I}_3|-\mathbf{t})$, where K is the matrix of intrinsic parameters and $(\mathsf{R},\mathbf{t})$ the extrinsic ones. $\mathbf{X}$ corresponds to all the optimized parameters in a BA: the extrinsic parameters of the optimized cameras $\{\mathsf{R}_j,\mathbf{t}_j\}_{j=1}^{S}$ and the 3D point posi-

tions $\{\mathbf{Q}_i\}_{i=1}^{N_Q}$. $\bar{\mathbf{X}}$ corresponds to the vector concatenating all the optimized pose parameters $\bar{\mathbf{X}}_j = \{\mathsf{R}_j, \mathbf{t}_j\}_{j=1}^{S}$.

3. Model-based cSLAM framework

Our cSLAM framework is built on key-frame-based SLAM [18], as shown in Figure 2. The camera localization is estimated at each frame thanks to a matching algorithm that establishes 2D/3D correspondences. The mapping process is then performed when a key-frame is detected. Generally it is decomposed into two important steps, the triangulation that extends the 3D map with new 3D features, and the refinement of camera poses and 3D feature positions through a Bundle Adjustment.
In [18], the BA is constrained with edgelets extracted from the CAD model of the object of interest. These edgelets form an accurate structure constraint and are not questioned in the cBA. Our approach is slightly different since the CAD model is used to estimate a pose (named model-based pose as opposite of the SLAM pose) for each new key-frame. The resulting model-based pose is then used as a trajectory constraint in the Bundle Adjustment step. Another difference is that [18] optimize simultaneously the structure constraints provided by the CAD model and the multi-view geometry. However, our trajectory constraint may present inaccuracies. Optimizing simultaneously this last one and the multi-view constraints could result in a deterioration of the multi-view geometry relationships, that may causes localization failures. Our approach consists in two steps. First, these constraints are separately optimized (section 3.1 and 3.2). Then, the results are combined (section 3.3).

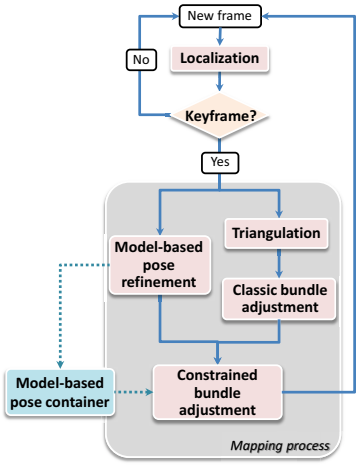


Figure 2: Our cSLAM framework with a four-step mapping process: triangulation, model-based pose refinement, classic and constrained Bundle Adjustment.

3.1. Optimal solution for model-based constraint

For each new key-frame, we determine the optimal pose with respect to the CAD model of the object of interest. This pose corresponds to the one that best aligns the 3D features of the model and their observations in the image. In our context, 2D observations are contour points and 3D features are edgelets. These edgelets can be generated offline from sharp edges if the object is polyhedral [2, 21], or online if the object include curved surfaces [13, 14, 22].
Since the observations that correspond to edgelets are initially unknown, the camera pose and these observations are defined as the parameters that minimize the model-based error given by:

$$H(\bar{\mathbf{X}}_j, \bar{\mathbf{m}}_j) = \sum_{i=1}^{N_M} (\mathbf{n}_{i,j}.(\mathbf{m}_{i,j} - \mathsf{P}_j\mathbf{M}_i))^2, \quad (1)$$

It measures the orthogonal distance between the projection of edgelets $\mathbf{M}_i$, and their corresponding edge $\mathbf{m}_{i,j}$ concatenated in the vector $\bar{\mathbf{m}}_j$, for the j^{th} key-frame. $\mathbf{n}_{i,j}$ is the normal to the projection of the edgelet direction. We define the vector representation of the model-based error $H(\bar{\mathbf{X}}_j, \bar{\mathbf{m}}_j)$ as follow:

$$H(\bar{\mathbf{X}}_j, \bar{\mathbf{m}}_j) = h^\top(\bar{\mathbf{X}}_j, \bar{\mathbf{m}}_j) h(\bar{\mathbf{X}}_j, \bar{\mathbf{m}}_j), \quad (2)$$

with $h^\top(\bar{\mathbf{X}}_j, \bar{\mathbf{m}}_j) = [(\mathbf{n}_{0,j}.(\mathbf{m}_{0,j} - \mathsf{P}_j\mathbf{M}_0))...(\mathbf{n}_{N,j}.(\mathbf{m}_{N,j} - \mathsf{P}_j\mathbf{M}_N))]$.
This minimization problem is usually solved by alternating the estimation of the observations $\bar{\mathbf{m}}_j$ and the estimation of the pose parameters $\bar{\mathbf{X}}_j$. The $\bar{\mathbf{m}}_j$ are estimated by projecting the edgelets with the current pose parameters and matching them to the nearest contour with a similar orientation. The pose parameters are then refined by minimizing $H(\bar{\mathbf{X}}_j, \bar{\mathbf{m}}_j)$ with the Levenverg-Marquardt (LM) algorithm.
To ensure a good convergence, such algorithm requires an initial estimation of $\bar{\mathbf{X}}_j$ close to the solution and a configuration of edgelets/observations that constrains the 6 DoF of the camera. In our context, the first assumption is verified since the optimization process uses the SLAM pose as initial guess. On the contrary, the second assumption cannot be ensured. It is then preferable to keep the poorly constrained DoF to their initial value during the optimization process. This can be achieved by using truncated Levenverg-Marquardt [3]. This approach relies on a different approximation of the Hessian matrix's cost function than the usual LM. While the Hessian matrix of $H(\bar{\mathbf{X}}_j, \bar{\mathbf{m}}_j)$ is approximated with $\mathsf{W}_j \approx \mathsf{J}_j{}^\top \mathsf{J}_j$ in classic LM [15], it is approximated by $\mathsf{W}_j = TSVD(\mathsf{J}_j{}^\top \mathsf{J}_j)$ in a truncated LM, where $TSVD(\mathsf{A})$ represents the truncated SVD of the matrix A.
The resulting orientation and position parameters $\bar{\mathbf{X}}_j^c = \{\mathsf{R}_j^c, \mathbf{t}_j^c\}$ of the optimal model-based pose are

stored to be used as a constraint for this key-frame over the rest of the tracking. The associated projection matrix is denoted P_j^c. The vector $\bar{\mathbf{X}}^c$ represents the set of model-based pose parameters $\bar{\mathbf{X}}_j^c = \{\mathsf{R}_j^c, \mathbf{t}_j^c\}$ for all the key-frames optimized in the cBA. The 2D contour points associated to edgelets on key-frame j with the optimal pose $(\mathsf{R}_j^c|\mathbf{t}_j^c)$, are concatenated in the vector $\bar{\mathbf{m}}_j^c$. Finally, we define $\mathsf{W}_j^c = \mathsf{J}_j^{c\top}\mathsf{J}_j^c$ as the approximation of $H''(\bar{\mathbf{X}}_j^c, \bar{\mathbf{m}}_j)$.

3.2. Optimal solution for multi-view constraint

As for the model-based constraint, we determine the optimal solution of the multi-view relationships for all the optimized key-frames. These key-frames are selected via a covisibility graph [12]. To obtain this optimal solution, a classic BA without any constraint is thus performed.
The cost function $G(\mathbf{X})$ to minimize is defined as follow:

$$G(\mathbf{X}) = \sum_{i=1}^{N_Q} \sum_{j \in L_i} d^2(\mathbf{q}_{i,j}, \mathsf{P}_j \mathbf{Q}_i) \tag{3}$$

The re-projection error d is the euclidean distance between $\mathbf{q}_{i,j}$ the 2D observation of the i^{th} 3D point $\mathbf{Q}_i$ in the j^{th} key-frame and the projection of this 3D point. L_i is the set of the key-frame indexes observing $\mathbf{Q}_i$.

3.3. Constrained Bundle Adjustment

The optimal solution for multi-view constraint and the one for the trajectory constraint are combined in cBA. One possible fusion strategy is to use a cost function that is a weighted sum between the two constraint terms presented above. However, such approach would not be robust to the presence of erroneous model-based poses, that may occur when the object is small in the image or occluded.
Since multi-view constraint is more robust but less accurate than the model-based constraint, we prefer a fusion strategy, which ensures that the degradation of the multiview constraint remains small. To achieve this task, Lhuillier constrained Bundle Adjustment framework [7] is chosen. Thus the cost function of our cBA combining multi-view and hybrid constraints is given by:

$$F(\mathbf{X}) = \frac{\omega}{e_t - G(\mathbf{X})} + E(\bar{\mathbf{X}}), \tag{4}$$

where $E(\bar{\mathbf{X}})$ is the hybrid constraint described in Section 4, and e_t is a threshold which is slightly greater than the squared re-projection error obtained after minimizing Equation 3. The fraction $\frac{\omega}{e_t - G(\mathbf{X})}$ with $\omega > 0$ corresponds to a regularization term that prevents the degradation of the multi-view relationships estimated after the classic BA. The cBA is initialized from the optimal solutions of the multi-view criteria and moves progressively the key-frame poses towards the model-based ones, while preventing over-fitting on inaccurate poses.

4. Hybrid pose/structure constraint

In the previous Section, a SLAM framework integrating the outputs of model-based tracker as a pose constraint in a cBA has been proposed. However, as noted by [10], minimizing a structure error (similar to [18] with edgelets) is more relevant than minimizing a distance between SLAM and model-based poses. As detailed in Section 2, structure constraint has the advantages to be expressed through a pixel error and takes into account badly constrained DoF of the camera poses. Nevertheless it generates a costly data storage such as contour images for 2D/3D matching steps.
To combine the benefits of both trajectory and structure constraints, different pose distances are introduced in this section. They measure the deviation of pose parameters from $\bar{\mathbf{X}}^c$ by their impact on the structure constraint. After introducing the ideal distance in section 4.1, two approximations of this distance are introduced to reduce the memory footprint and the processing cost.

4.1. Main idea

We propose to use a difference of structure errors as a pose distance. More precisely, it consists in measuring the difference between the structure error (Equation 1) corresponding to a key-frame pose $(\mathsf{R}_j|\mathbf{t}_j)$ and the one corresponding to the model-based pose with the parameters $\bar{\mathbf{X}}_j^c$ associated to the same j^{th} key-frame. The corresponding distance is then defined by the following equation:

$$E(\bar{\mathbf{X}}) = \sum_{j=1}^{S} H(\bar{\mathbf{X}}_j, \bar{\mathbf{m}}_j^c) - \sum_{j=1}^{S} H(\bar{\mathbf{X}}_j^c, \bar{\mathbf{m}}_j^c), \tag{5}$$

Notice that to ensure the structure error to be minimal at the pose $(\mathsf{R}_j^c|\mathbf{t}_j^c)$, $E(\bar{\mathbf{X}})$ is evaluated by using the 2D contour points $\bar{\mathbf{m}}_j^c$ associated to edgelets for the key-frame j. We define the vector representation of our hybrid error $E(\bar{\mathbf{X}})$ as follow:

$$E(\bar{\mathbf{X}}) = H(\bar{\mathbf{X}}, \bar{\mathbf{m}}^c) - H(\bar{\mathbf{X}}^c, \bar{\mathbf{m}}^c), \tag{6}$$

with $\bar{\mathbf{m}}^c$ the vector concatenating the 2D contour points associated with the parameters $\bar{\mathbf{X}}^c$ for all the edgelets and for all the optimized key-frames. An illustration of this hybrid error for a given key-frame and a given edgelet is represented in the Figure 3. With this definition we ensure that $E(\bar{\mathbf{X}})$ is minimal when the structure constraint $H(\bar{\mathbf{X}}, \bar{\mathbf{m}}^c)$ is minimal (*i.e.* at the pose parameters $\bar{\mathbf{X}}^c$). Besides, the pose defined by the parameters $\bar{\mathbf{X}}^c$, the 2D contour points $\bar{\mathbf{m}}^c$, and consequently the structure error associated to the parameters $\bar{\mathbf{X}}^c$ are constant. Since the 2D contour points $\bar{\mathbf{m}}^c$ are no longer re-estimated, it is not required to store the contour images. Furthermore, contrary to [18], the edgelet/contour matching step is not necessary anymore, which allows to reduce computation time. Only the vector of the 2D contour

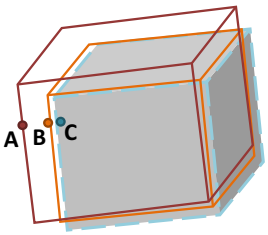


Figure 3: Point A (respectively B) is an edgelet extracted from a cube model and projected according to a SLAM key-frame pose in red (respectively according to the model-based pose in orange). Point C is the 2D contour point associated to the edgelet B. The error $E(\bar{\mathbf{X}})$ of Equation 6 can be interpreted as the difference $d_{ortho}(A,C) - d_{ortho}(B,C)$, with d_{ortho} the orthogonal distance.

points $\bar{\mathbf{m}}^c$ and, in case of dynamic edgelets, the edgelet constellation are memorized for each key-frame. Compared to [18], it implies a large reduction of memory footprint and computation cost. However, this memory footprint can remain important if the number of edgelets and key-frames is high. In the following Section, we introduce two approximations of $E(\bar{\mathbf{X}})$ that reduce the memory footprint.

4.2. First approximation

This first approximation relies on the hypothesis that the model-based tracker has converged to a local minimum. Thus it is reasonable to consider that the distance between the projection of the edgelets with respect to the parameters $\bar{\mathbf{X}}^c$ and their corresponding 2D contour points is negligible. In Figure 3, this assumption corresponds to approximate $d_{ortho}(A,C) - d_{ortho}(B,C)$ by $d_{ortho}(A,B)$.
Under such hypothesis, the 2D contour points $\bar{\mathbf{m}}^c_j$ of Equation 5 are replaced by the projection of their corresponding 3D edgelets with respect to the parameters $\bar{\mathbf{X}}^c_j$. The resulting approximation of $E(\bar{\mathbf{X}})$ is given by:

$$E(\bar{\mathbf{X}}) = \sum_{j=1}^{S} H(\bar{\mathbf{X}}_j, \bar{\mathbf{M}}^c_j), \tag{7}$$

where $\bar{\mathbf{M}}^c_j = \{\mathsf{P}^c_j \mathbf{M}_i\}_{i=1}^{N_M}$ is the vector concatenating edgelets $\mathbf{M}_i$ projected according to P^c_j.
With this approach, it is not necessary to keep in memory the 2D associated contours, since only the model-based pose parameters are exploited. In the case of polyhedral object, the memory footprint of the resulting cBA is very small, since only a set of edgelets shared by all the key-frames has to be stored. However, the memory consumption is higher with dynamic edgelets extracted from occluding contours, since they depend on the point of view. They are generated online and stored for each key-frame. That is why a second approximation of Equation 6 is proposed to reduce memory consumption.

4.3. Second approximation

This second approximation relies on the hypothesis that the poses of the SLAM key-frames are located in the neighborhood of their corresponding model-based poses. This approximation is valid in practice since the model-based poses are obtained by refining the SLAM poses with Equation 1. Under this hypothesis, Equation 6 can be approximated with a second order Taylor expansion around the model-based poses $\bar{\mathbf{X}}^c$:

$$E(\bar{\mathbf{X}}) \approx E(\bar{\mathbf{X}}^c) + E'(\bar{\mathbf{X}}^c)\delta + \frac{1}{2}\delta^\top E''(\bar{\mathbf{X}}^c)\delta, \tag{8}$$

where $\delta = \bar{\mathbf{X}} - \bar{\mathbf{X}}^c$ is the difference between pose parameters (positions and orientations) provided by the SLAM and model-based tracking algorithms. $\bar{\mathbf{X}}^c$ is obtained after the model-based refinement and corresponds to the minima of the model-based cost functions (Equation 1) for all the optimized key-frames. Thus $H'(\bar{\mathbf{X}}^c, \bar{\mathbf{m}}^c)$ (Equation 2) is null. Consequently the first derivative $E'(\bar{\mathbf{X}}^c) = H'(\bar{\mathbf{X}}^c, \bar{\mathbf{m}}^c) = 0$. Besides the second derivative $E''(\bar{\mathbf{X}}^c) = H''(\bar{\mathbf{X}}^c, \bar{\mathbf{m}}^c)$ can be approximated by the $S \times S$ diagonal matrix W^c with S the number of optimized key-frames. The diagonal term of this matrix is $\mathsf{W}^c_j = \mathsf{J}^{c\top}_j \mathsf{J}^c_j$, $j \in [0..S]$, which corresponds to the approximation of $H''(\bar{\mathbf{X}}^c_j, \bar{\mathbf{m}}_j)$ introduced in Section 3.1. This matrix is estimated during the model-based refinement step for the key-frame j (see Section 3.1). W^c_j is determined only once per key-frame and stored in addition to the optimal pose parameters $\bar{\mathbf{X}}^c_j$. The hybrid error defined in Equation 8 becomes:

$$E(\bar{\mathbf{X}}) = E(\bar{\mathbf{X}}^c) + \frac{1}{2}\delta^\top \mathsf{W}^c \delta \tag{9}$$

Since $E(\bar{\mathbf{X}}^c) = H(\bar{\mathbf{X}}^c, \bar{\mathbf{m}}^c) - H(\bar{\mathbf{X}}^c, \bar{\mathbf{m}}^c) = 0$ according to Equation 6, the hybrid error definition becomes:

$$E(\bar{\mathbf{X}}) = \frac{1}{2}\delta^\top \mathsf{W}^c \delta \tag{10}$$

This definition of our hybrid constraint presents several advantages. W^c converts the difference between the pose parameters to a structure error in pixels and makes this error homogeneous with the multi-view term of the cBA (Equation 4). It also allows not to store contour images for each key-frame contrary to Equation 1. Moreover, neither the edgelets, nor their associated observations intervene, allowing not to store any of them. Only the model-based pose $\{\mathsf{R}^c_j, \mathbf{t}^c_j\}$ and the 6×6 matrix W^c_j are exploited in the optimization process for each key-frame.
This hybrid constraint results in a Bundle Adjustment with a memory footprint invariant to the video resolution and the model complexity.

5. Experimental results

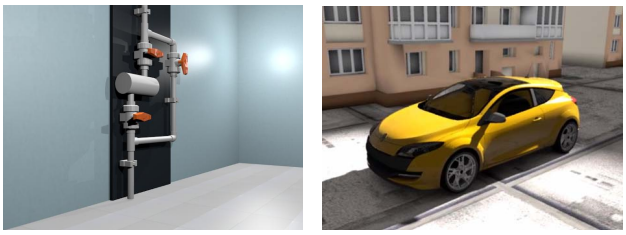
In this section, our solutions are compared with edgelet-cSLAM in term of accuracy, computation time and memory consumption.

5.1. Accuracy evaluation

The localization accuracy of our solutions exploiting the two hybrid constraints is compared with edgelet-cSLAM on curved and polyhedral objects [8, 18] on synthetic and real videos. It thus assess the genericity of our method.

Synthetic data. Several synthetic sequences with a resolution of 640×480 have been generated. A first sequence presents a curved object from the chemical industry called bypass, and mainly composed by pipes. Its CAD model used during the tracking has 152K faces. The object environment is composed of four walls with a grey texture and a paving ground, as illustrated on Figure 4(a). During the sequence, the bypass is not always entirely in the camera field of view. A second sequence has a sedan car as object of interest. The CAD model has 50K faces. The sedan car environment is composed of city buildings. The initialization of the SLAM on these synthetic sequences is performed thanks to the ground truth first pose.
Quantitative results are obtained by measuring the difference between the ground truth and the position and orientation of camera poses estimated by the cSLAM. The position error is expressed as a percentage of the camera/object distance, whereas the orientation error is expressed as an angle in radians. A mean pixel error is also computed by measuring the distance between the edgelet projections with the ground truth and the SLAM poses.
Figures 5(c), 5(d) and 5(e) present the localization error in position and orientation, and the pixel error on the bypass sequence. Figures 5(f), 5(g) and 5(h) present the same errors on the sedan car sequence. We present, as a reference, the errors obtained with the visual SLAM of [11] without constraint. The camera localization is less accurate with visual SLAM than with the edgelet-cSLAM or our proposed solutions. The edgelet constraint and the second hybrid constraint provide similar accuracies, contrary to the first hybrid constraint that presents some lacks of accuracy. On the sedan sequence, the median pixel error is 1.38 for the first hybrid constraint, 1.12 for the second one and 1.63 for the edgelet constraint of [18]. On the bypass sequence, the median pixel error is 1.39 for the first hybrid constraint, 0.99 for the second one and 0.91 for the edgelet constraint of [8]. Only our solution based on the first approximation appears a little bit less accurate than the two others.

Real data. The accuracy and robustness of our method is also qualitatively evaluated on real sequences (see Figure 1(b) 1(a) and 6) recorded with a HD resolution (1280×1024). The two objects used are a metal statue of dragon with a 3D model reconstructed by photogrammetry and a sport car, whose the CAD model is available. The SLAM initialization on these real videos is performed manually and followed by a contour refinement process.
Since results obtained with [18, 8] are visually similar to those obtained with the proposed solution whatever the hybrid constraint, only the results with the second hybrid constraint on the sport car and dragon sequences are presented. Our solution is robust to occlusions, thanks to the proposed cSLAM framework, which guarantees the multi-view relationships to be well estimated. Videos are available for supplementary evaluations of the accuracy and robustness.



(a) (b)

Figure 4: Images from the bypass (a) and sedan car (b) synthetic sequences.

5.2. Computation time comparison

In this section, the computational cost of the different solutions are compared. Since these solutions differ only by their Bundle Adjustment, this experiment consists in comparing the processing time required by the mapping process described in Section 3. The mapping process is achieved once per new key-frame, thus we compare the median processing time of these executions. Also, since the computation time depends on the number of optimized key-frames, the different solutions are compared for a set of 3, 10 and 30 optimized key-frames.
This experiment is achieved on the two real sequences introduced in Section 5.1. The sport car illustrates the performances for a polyhedral object, while the dragon sequence illustrates the performances for a curved object. For the car (respectively dragon) sequence, the performances of our two solutions are compared to the performances of [18] (respectively [8]).
In our experiments, we use a laptop with an Intel(R) Core(TM) i7-4800MQ CPU @ 2.70GHz processor and a NVIDIA GeForce GT 730M graphics hardware. The computation time of the mapping process described in Section 3 is presented in Table 1. The optimal solutions for the model-based constraint and the multi-view one are running in parallel, to optimize the computation time. Moreover, even if an edgelet-cSLAM has not the model-based pose refinement step in its framework (contrary to our structure/trajectory-cSLAM), the 2D/3D association useful

	Polyhedral object (sport car)		
	E. constraint	First H.constraint	Second H. constraint
3 KF	47.9	27.9	**23.6**
10 KF	130.5	77.9	**76.2**
30 KF	216.6	203.9	**188.7**

(a)

	Curved object (dragon statue)		
	E. constraint	First H.constraint	Second H. constraint
3 KF	132.4	88.2	**84.4**
10 KF	268.4	170.6	**170.1**
30 KF	390.7	330.0	**284.8**

(b)

Table 1: Computation time in milliseconds of the mapping process for the edgelet (E.), the first and second hybrid (H.) constraints on the sport car and dragon sequences. It is given for 3,10 and 30 optimized key-frames (KF).

for the cBA process slows down its execution. Computation time is globally more important with the tracking of curved object. It is due to the edgelet generation process, which has to be performed online at the last key-frame since the occluding contours depend on the camera point of view. On the contrary, the edgelet extraction is performed offline only once for polyhedral objects. Table 1 shows that the exploitation of any hybrid constraint allows to reduce computation time. Compared to the use of edgelet constraint, the computation time decreases of around 44% for the sport car sequence and around 35% on the dragon one even if a model-based pose refinement is performed in addition to the cBA with the use of hybrid constraint (with 3 and 10 key-frames in the BA). The computation time decrease of around 10% and 21% with the use of 30 key-frames.

5.3. Memory consumption comparison

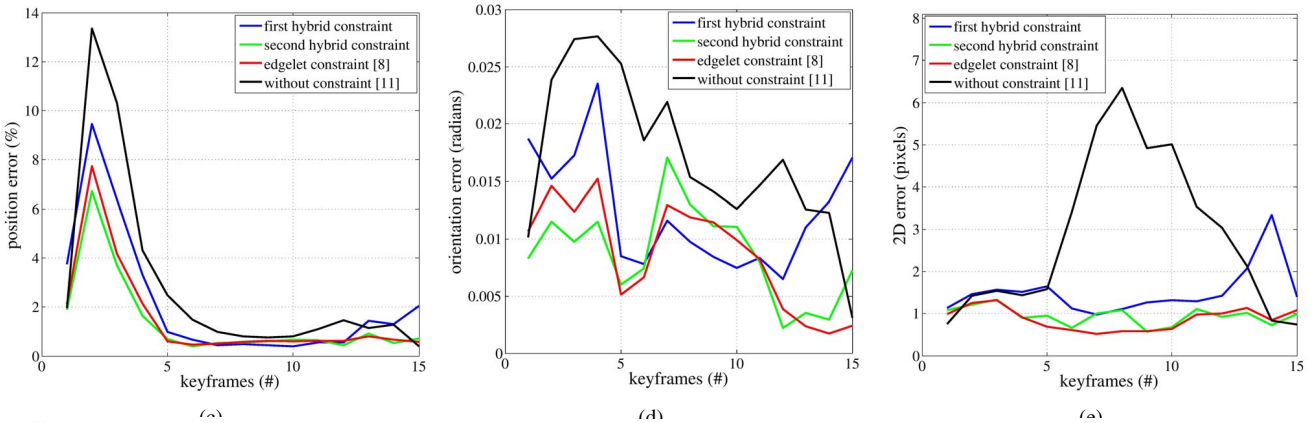
Optimizing a SLAM reconstruction with a BA constrained to a CAD model implies the storage of additional data for each key-frame. In this section, the memory footprint of these data is evaluated, depending on the use of the model-based constraint and the edgelet generation process. These evaluations are conducted on the two real sequences introduced in Section 5.1. The edgelet extraction is performed offline for the car sequence and online for the dragon statue. Our approaches are compared to [18] for the sport car and to [8] for the dragon. The results are summarized in Figures 1(c) and 1(d).

As explained in Section 4, edgelet-cSLAM requires to store edgelets and contour images for each key-frame optimized in the cBA. Consequently the memory footprint increases every time a new key-frame is created. The impact is even greater as the video resolution (for curved objects) and the number of edgelets are high. For the real sequences, the number of edgelets projected and associated to 2D contour points is set to 2000, and 94 (respectively 113) key-frames are created over the car sequence (respectively dragon sequence). Thus, as shown in Figures 1(c) and 1(d), the memory footprint for edgelet-cSLAM is near 235 MB at the end of the sport car sequence and more than 292 MB at the end of the dragon one. The latter sequence requires more memory since edgelets extracted from occluding contours are stored for each key-frame, which is not required with polyhedral objects. The memory footprint drastically decreases with the use of a hybrid constraint, since contour images are not required anymore in the cBA. Pose parameters of the model-based tracker outputs are henceforth stored instead. In addition to the pose parameters storage, our first definition of the hybrid constraint requires to save sets of edgelets. Since they are extracted only once with polyhedral object, the memory consumption is limited to 0.1 MB at the end of the car sequence. However, generating and accumulating sets of edgelets at each key-frame for curved objects, increase the memory footprint to 10 MB at the end of the dragon sequence. Finally the second hybrid constraint allows a memory footprint close to 0.12 MB (respectively 0.04 MB) at the end of the sport car sequence (respectively dragon sequence), only pose parameters and matrices W^c of the model-based tracker outputs (see Section 4.3) being stored. We can notice that with polyhedral objects, the memory consumption is higher for the second hybrid constraint (0.12 MB) than for the first one (0.10 MB). Indeed, in addition to the pose parameters stored for both hybrid constraints, for the first constraint solution, a unique set of edgelets is stored for the whole sequence, whereas for the second hybrid constraint solution, matrices W^c_j are stored at each key-frame creation. This second hybrid constraint allows our memory consumption to be invariant to the number of edgelets and their generation, invariant to the resolution of the tracking video. Thus, if polyhedral (respectively curved) objects are tracked, the first (respectively second) hybrid constraint is more suitable.

6. Conclusion

In this article, we introduce a solution to reduce the memory footprint of SLAM constrained to a CAD model while achieving similar accuracy and computational cost. Our solution relies on a real-time visual SLAM algorithm that integrates a hybrid structure/trajectory constraint in the BA process. Two versions of this hybrid constraint have been presented, the first one being more suitable with polyhedral object, and the second with curved objects. They both exploit the output of a model-based tracking algorithm. Our tracking solution has been tested on polyhedral and curved objects to attest its genericity. Experimental results demonstrate that our solution is as robust and accurate as edgelet-cSLAM [18, 8] while reducing drastically the memory consumption and the computation time.

Acknowledgments This work was partly funded by the french research program FUI through the project *NASIMA*.



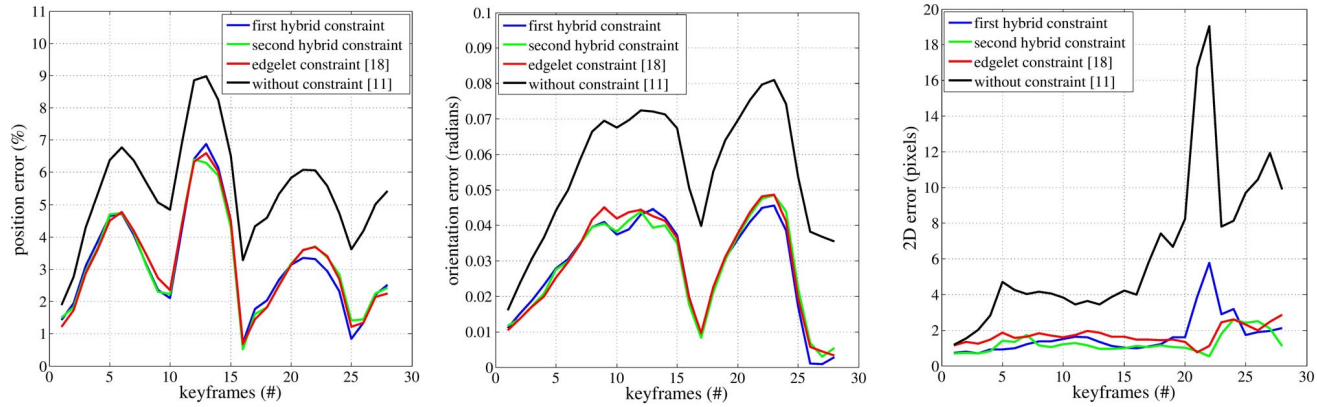


Figure 5: Accuracy comparison between SLAM without constraint [11] (black), edgelet-cSLAM [18, 8] (red) and our cSLAM with the first hybrid constraint (blue) and the second one (green) on the bypass (top) and sedan car (bottom) sequences.

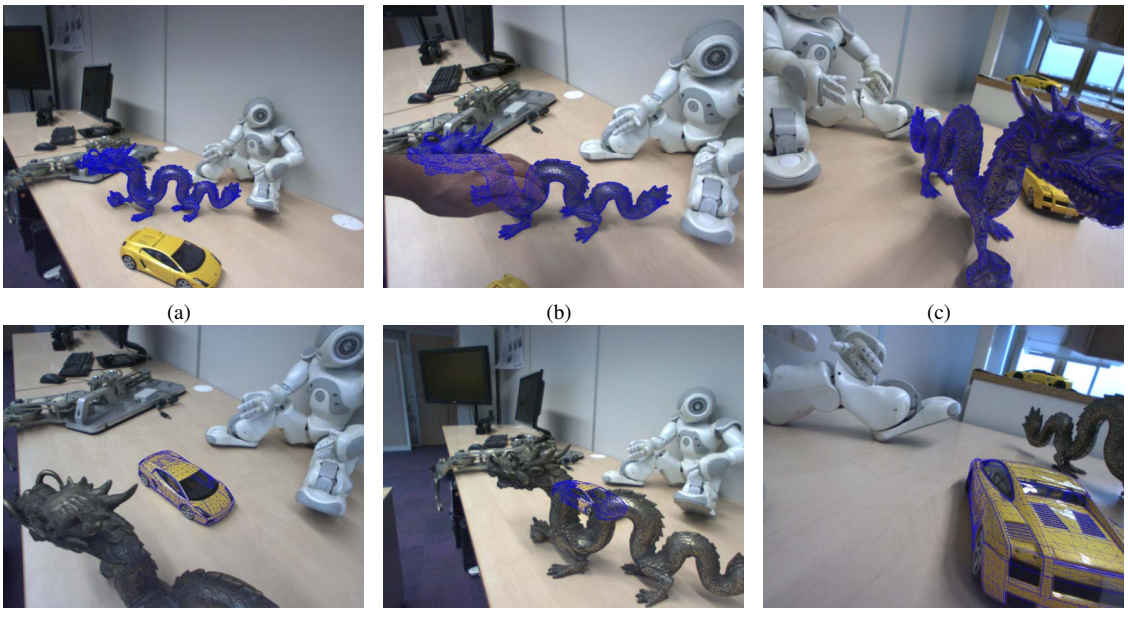


Figure 6: Accurate localization results on the dragon (top) and the sport car (bottom) with our cSLAM with the second hybrid constraint. The accuracy can be appreciated by the projection of the 3D blue models on the object of interest in the image.

References

[1] G. Bleser, H. Wuest, and D. Strieker. Online camera pose estimation in partially known and dynamic scenes. In *ISMAR*, 2006. 1

[2] T. Drummond and R. Cipolla. Real-time visual tracking of complex structures. *PAMI*, 24(7):932–946, 2002. 3

[3] S. Finsterle and M. Kowalsky. A truncated levenberg marquardt algorithm for the calibration of highly parameterized nonlinear models. *Computers & geosciences*, 37(6):731–738, 2011. 3

[4] G. Klein and D. Murray. Parallel tracking and mapping for small AR workspaces. In *ISMAR*, 2007. 2

[5] D. Larnaout, S. Bourgeois, V. Gay-Bellile, and M. Dhome. Towards bundle adjustment with gis constraints for online geo-localization of a vehicle in urban center. In *3D Imaging, Modeling, Processing, Visualization and Transmission*, 2012. 2

[6] V. Lepetit and P. Fua. Monocular model-based 3d tracking of rigid objects: A survey. *Foundations and Trends in Computer Graphics and Vision*, 1(1):1–89, 2006. 1

[7] M. Lhuillier. Incremental fusion of structure-from-motion and gps using constrained bundle adjustments. *Pattern Analysis and Machine Intelligence*, 34(12):2489–2495, 2012. 2, 4

[8] A. Loesch, S. Bourgeois, V. Gay-Bellile, and M. Dhome. Generic edgelet-based tracking of 3d objects in real-time. In *IEEE Conference on Intelligent Robots and Systems*, 2015. 1, 2, 6, 7, 8

[9] P. Lothe, S. Bourgeois, F. Dekeyser, E. Royer, and M. Dhome. Towards geographical referencing of monocular slam reconstruction using 3d city models: Application to real-time accurate vision-based localization. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2009. 2

[10] S. Middelberg, T. Sattler, O. Untzelmann, and L. Kobbelt. Scalable 6-dof localization on mobile devices. In *European Conference on Computer Vision*, 2014. 1, 2, 4

[11] E. Mouragnon, M. Lhuillier, M. Dhome, F. Dekeyser, and P. Sayd. Real time localization and 3d reconstruction. In *CVPR*, 2006. 2, 6, 8

[12] R. Mur-Artal, J. M. M. Montiel, and J. D. Tards. Orb-slam: a versatile and accurate monocular slam system. *Transactions on Robotics*, 31(5):1147–1163, 2015. 4

[13] M. Nienhaus and J. Doellner. Edge-enhancement - An algorithm for real-time non-photorealistic rendering. *Journal of WSCG*, 11(2), 2003. 3

[14] A. Petit, E. Marchand, and K. Kanani. Combining complementary edge, point and color cues in model-based tracking for highly dynamic scenes. In *ICRA*, 2014. 3

[15] W. H. Press. Numerical recipes 3rd edition: The art of scientific computing. pages 282–283. Cambridge university press, 2007. 3

[16] D. Ramadasan, M. Chevaldonn, and T. Chateau. Dcslam: A dynamically constrained real-time slam. In *International Conference on Image Processing*, 2015. 1, 2

[17] D. Ramadasan, M. Chevaldonn, and T. Chateau. Mcslam : a multiple constrained slam. In *Proceedings of the British Machine Vision Conference*, 2015. 2

[18] M. Tamaazousti, V. Gay-Bellile, S. N. Collette, S. Bourgeois, and M. Dhome. Real-time accurate localization in a partially known environment: Application to augmented reality on textureless 3d objects. In *ISMAR Workshop*, 2011. 1, 2, 3, 4, 5, 6, 7, 8

[19] M. Tamaazousti, V. Gay-Bellile, S. Naudet-Collette, S. Bourgeois, and M. Dhome. Nonlinear refinement of structure from motion reconstruction by taking advantage of a partial knowledge of the environment. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2011. 2

[20] B. Triggs, P. McLauchlan, R. Hartley, and A. Fitzgibbon. Bundle adjustment - a modern synthesis. In *International Conference on Computer Vision*, 2000. 2

[21] L. Vacchetti, V. Lepetit, and P. Fua. Stable real-time 3d tracking using online and offline information. *PAMI*, 26(10):1385–1391, 2004. 3

[22] H. Wuest, F. Wientapper, and D. Stricker. Adaptable model-based tracking using analysis-by-synthesis techniques. In *CAIP*, 2007. 3