View synthesis for pose computation

Pierre Rolin, Marie-Odile Berger, Frédéric Sur

LORIA UMR 7503
CNRS, Inria, Université de Lorraine

Abstract

Geometrical registration of a query image with respect to a 3D model, or pose estimation, is the cornerstone of many computer vision applications. It is often based on the matching of local photometric descriptors invariant to limited viewpoint changes. However, when the query image has been acquired from a camera position not covered by the model images, pose estimation is often not accurate and sometimes even fails, precisely because of the limited invariance of descriptors. In this paper, we propose to add descriptors to the model, obtained from synthesized views associated with virtual cameras completing the covering of the scene by the real cameras. We propose an efficient strategy to localize the virtual cameras in the scene and generate valuable descriptors from synthetic views. We also discuss a guided sampling strategy for registration in this context. Experiments show that the accuracy of pose estimation is dramatically improved when large viewpoint changes makes the matching of classic descriptors a challenging task.

Keywords: *pose estimation, view synthesis, feature matching, RANSAC*

This is a pre-copyedit version of an article published in Machine Vision and Applications. The final authenticated version is available online at:
`http://dx.doi.org/10.1007/s00138-019-01045-5`

1 Introduction

Camera pose estimation is a recurring problem in computer vision and has applications in various domains such as augmented reality applications [1, 2], vision-based robot positioning [3], or aerial image geo-registration [4]. The initialization of the pose is a critical step in all pose estimation methods and is needed either when starting a system or when pose tracking fails. In some application domains, it is possible to initialize the pose with additional information such as GPS localization. It is, however, often not possible to rely solely on GPS localization, either because the GPS estimate is not accurate enough for the application under study, or because the environment is GPS-denied. In this article we address the problem of camera pose initialization, or kidnaping problem, from a single image and a point cloud model.

In many applications, the pose is estimated relative to a model of the scene reconstructed from images through a structure-from-motion (SfM) algorithm. Local photometric descriptors of the 3D points are extracted from the construction views as, e.g., SIFT features [5]. Afterward, these descriptors are used to match interest points of the query view and 3D points, which makes it possible to solve the perspective-n-point (PnP) problem [6] and estimate the pose. This approach presents a major issue when the construction views do not cover the whole set of potential viewpoints. Indeed, a query view taken from an uncovered viewpoint is likely to give too few reliable point correspondences because of the limited invariance of the photometric descriptors to viewpoint changes [7]. A good example of such a situation is described by the authors of [4] who aim at registering a view from an aerial drone to a model built from ground-level construction views. It is worth noting that some approaches do not rely on feature matching for pose estimation, such as PoseNet [8] and methods derived from it, which use a convolutional neural network to directly compute the pose associated with an input image. To this date, the accuracy of these approaches is significantly lower than the accuracy of feature based methods, as mentioned in [9, 10]. In addition, as a regression, the method does not allow coping with viewpoints which are significantly different from those used during learning, although the authors of a very recent paper [9, 10] propose to augment the training data with synthetic viewpoints and claim attaining a pose accuracy comparable to feature-based methods. In this context, the present paper is focused on geometric feature matching over large viewpoint changes. In line with [9, 10], the proposed contribution is of interest for regression methods, although these latter approaches are not within our scope.

To solve the matching problem, two approaches exist: either defining descriptors invariant to viewpoint changes, or simulating these viewpoint changes and extracting local descriptors from the synthesized views. The authors of [11] compare the robustness of theoretically affine invariant descriptors to various transformations. Concerning robustness to viewpoint changes, Hessian-affine descriptors [12] and MSER [13] perform best, but matching still fails for angles larger than 40°. Another comparison paper [7] concludes that no descriptor performs well under viewpoint changes larger than 30°. Recently, learned descriptors such as LIFT [14] made use of deep learning techniques

to achieve robustness to various conditions changes. While robustness to illumination changes outperforms existing descriptors, robustness to viewpoint changes is still limited and on par with affine invariant descriptors. An alternative approach to enhance robustness to viewpoint changes consists in synthesizing new local patches through simple geometric transformations of a rough scene model. For instance, an adaptation of [15] to SLAM relocalisation is proposed in [16]. Instead of synthesizing with limited-range transformations (for example, camera tilt varies between $-30°$ to $+30°$ in [15]), it is possible to synthesize new views with large baseline changes. The objective of the papers in this research line is to complete a set of real views with synthetic views from uncovered viewpoints. For example, it is proposed in [17] to use view synthesis for data augmentation in image classification tasks. View synthesis is also the ground of ASIFT [18] for image matching and of [19] for pose estimation. Although these approaches outperform invariant descriptors, they are, however, generally dedicated to specific scene types or do not scale up well. The contribution of the present article is precisely to show that view synthesis dramatically improves pose computation and that both the synthesis process and the pose computation step can be achieved in an efficient way. The following section discusses the literature related to view synthesis for pose estimation through feature matching.

1.1 View synthesis for pose estimation

In [17, 20], view synthesis is used to augment data for image classification, in order to go beyond the limited invariance of descriptors. In [17], transformations modeling viewpoint changes are iteratively selected to augment the training data, improving classification accuracy significantly. In [20], it is proposed to generate new views by interpolation. This requires a relatively dense coverage of the scene for the interpolation to produce images consistent with actual observations. The authors of [21] (after [15]) use affine transformations randomly drawn to train a classifier for keypoint recognition. In [22, 23], it is proposed to iteratively select pertinent transformations rather than applying a set of pre-defined or random ones. While such approaches aim at limiting the computational burden, they essentially rely on heuristics to define synthetic viewpoints. Let us mention a mathematical analysis of these heuristic choices available in a recent paper [24].

Since these works deal with image classification or image registration and not with pose estimation, naively estimating pose from such methods gives a limited accuracy. In particular, no geometric model of the considered scene or object of interest is used, and the transformations are thus likely to produce incoherent results. Although it is not crucial for image classification tasks, accurate and efficient pose estimation requires well founded view synthesis, since aberrant synthetic views give spurious matches that have to be discarded by random sampling strategies such as RANSAC, at the price of a potentially high calculation cost.

Concerning pose estimation, several authors use viewpoint simulation strategies with application-dependent assumptions limiting the range of the required transformations, which is another way of alleviating the computational burden. The authors of several papers [25, 26, 27] generate fronto-parallel views of planar structures, which comes down

to choosing a single virtual position in front of the considered scene planes. Robustness to viewpoint changes is improved but still limited in case of slanted views of the plane. In the context of real-time tracking, the authors of [28] make use of a 3D rendered model to match edges. Place recognition and pose estimation in a urban environment are addressed in [29]. The virtual positions lie on a dense grid at street level and a rough 3D planar model of the scene is used. Synthetic views, generated by ray-tracing, are matched to the query view, which gives reliable place recognition. Synthetic views from street level are also used in [30, 31] to improve image registration to urban models. The authors of [4] address the ground-to-aerial registration problem where this simplifying assumption does not hold. Nevertheless, they assume that an estimation of the aerial position is available from GPS tags. This makes it possible to generate a synthetic view from a dense reconstruction of the scene corresponding to the GPS position, which can be accurately registered to the query view. A similar idea is exploited in [3] for vision-based robot localization, where it is assumed that a coarse estimation of the pose is available to drive view synthesis. The same assumption is used in some simultaneous localization and mapping (SLAM) applications to generate synthetic patches, after [32], or in tracking-by-synthesis methods [33, 34]. The authors of [35] propose to refine bundle adjustment by taking into account photometric reprojection error, which amounts to rendering synthetic views based on a mesh model of the scene.

It should be noted that most of these works require either a dense scene model (or a multiplanar textured reconstruction of the scene) [30, 26, 29, 27, 28], or an initial guess for the pose [3, 4, 34, 35]. As a conclusion of this short survey, and to the best of our knowledge, it seems to us that existing view synthesis approaches generally need some prior information depending on the application domain or do not simply scale up to larger scenes.

1.2 Contributions

This paper is an extension of our previous works on pose initialization. In [19], we propose an approach where no initial guess is available and the scene model is an unstructured 3D point cloud. It is assumed, however, that virtual viewpoints are regularly distributed on a sphere centered on the model. This restricts the applicability to relatively small object-centered scenes. In addition, all viewpoints have to be simulated to produce synthetic patches for all 3D points, making the algorithm quite demanding in terms of computation time. In [36], we propose a more efficient synthesis approach that is usable with any scene that contains planar areas. The problem of increased matching time due to the amount of added descriptors has yet to be addressed in this latter work.

The present paper deals with pose estimation from a query view, based on an SfM model of the scene, without any initial pose guess. As mentioned earlier, an application domain is the initialization of a tracking process. Each 3D point of the model is endowed with the collection of the corresponding SIFT descriptors matched in the SfM step. SIFT keypoints from the query view are matched to the model points by nearest-neighbor matching followed by RANSAC [37] and PnP [6]. Our goal is to add SIFT descriptors coming from synthesized patches in order to facilitate keypoint matching when the query

view is not covered by the construction views. The additional SIFT descriptors are extracted around the reprojected scene points in the synthesized patches. The synthesis itself relies on a coarse segmentation of the 3D scene model in planar areas, the virtual cameras being distributed around these areas.

Three contributions are proposed. In Section 2 we discuss the transformation model used for synthesis and we show that view synthesis dramatically improves pose computation. In Section 3 we introduce a method for positioning virtual viewpoints and generating synthetic patches from these viewpoints. The basic assumption of the method is that scenes are piecewise planar, which is sound in man-made environments. This method makes minimal hypothesis on the scene, essentially that there are some planar areas in the scene, and is tractable for large scene models. In Section 4 we propose a random sampling approach for efficiently searching image-model correspondences. Section 5 discusses experimental results.

2 View synthesis for pose estimation: A proof of concept

In this section, we discuss how to produce synthetic views and use them to enrich the SfM model with new descriptors.

The problem can be formulated as follows: Given an SfM model, as described in the introduction, and a viewpoint, defined by a camera with its intrinsic and extrinsic parameters, how to generate a new view of the scene? The model being built from a set of images, the idea is to render new views of some parts of the scene by applying a geometric transformation to an input image. Augmenting the SfM model to a textured mesh is not an option here, since it basically consists in a limited-accuracy point cloud. Assuming that the scene is locally planar, which is verified in a wide variety of environments, producing synthetic views consists in using geometric transformations of existing views of the scene planes. Homographic and affine transformations are considered, corresponding to pinhole and affine camera models, respectively.

We assume for a while that the virtual viewpoints have been chosen beforehand and we focus on the synthetic view generation and model enrichment. The positioning of the virtual viewpoints is thoroughly discussed in Section 3.1.

2.1 Pinhole camera

Considering pinhole camera models, the transformation between the images of a plane from two different viewpoints is given by a homography [38]. Let $P_1 = K_1[R_1|T_1]$ and $P_2 = K_2[R_2|T_2]$ be the two camera projection matrices, P_1 corresponding to a real observation and P_2 to a virtual viewpoint. A plane described by its equation $n^T X + d = 0$ is considered, n being a normal vector. The transformation induced by the plane between the two cameras is the homography H given by the following equation:

$$H = K_2(R - Tn^T/d)K_1^{-1} \tag{1}$$

where $R = R_2R_1^{-1}$ and $T = T_2 - RT_1$.

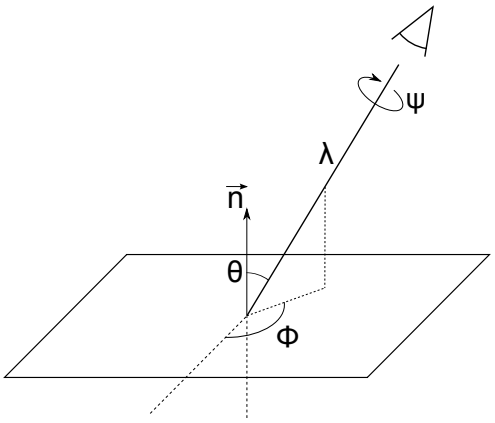

Figure 1: Affine camera parameters $(\lambda, \theta, \phi, \psi)$ relative to a plane described by its normal n. λ is a scale factor, ψ the rotation around the optical axis, θ the latitude and ϕ the longitude.

Matrix R_2 can be written $R_2 = R_z(\kappa)R_y(\phi)R_x(\omega)$ where (X, Y, Z) is an orthonormal basis such that Z is the camera axis and (κ, ϕ, ω) are the associated Euler angles. SIFT descriptors being invariant to any planar similarity, any rotation around the optical axis or focal change of camera 2 produces the same descriptors. Therefore, we only need to set the camera parameters up to a rotation around the optical axis and the focal can be fixed arbitrarily. As pointed out in [18], this argument based on ideal, continuous, images remains valid with sampled images if the Shannon-Nyquist conditions are met.

The normal n to the plane is required to compute the homography H. It is obtained by principal component analysis [39].

2.2 Affine camera

With affine cameras, the transformation between the images of the plane becomes an affine transformation. Considering a plane $n^T X + d = 0$, an affine camera is fully defined by its orientation relative to the plane $(\psi_i, \theta_i, \phi_i)$ and a zoom parameter λ_i, as illustrated in Figure 1. The affine transformation between the fronto-parallel view and the image from this camera can be written as:

$$A_i = \lambda_i \begin{pmatrix} \cos(\psi_i) & -\sin(\psi_i) \\ \sin(\psi_i) & \cos(\psi_i) \end{pmatrix} \begin{pmatrix} t_i & 0 \\ 0 & 1 \end{pmatrix} \begin{pmatrix} \cos(\phi_i) & -\sin(\phi_i) \\ \sin(\phi_i) & \cos(\phi_i) \end{pmatrix} \tag{2}$$

with $t_i = 1/\cos(\theta_i)$ the tilt of the camera [18]. As SIFT descriptors are invariant to similarities, the same descriptors are, theoretically, computed for any value of λ_i and ψ_i. As a consequence, we can arbitrarily set $\lambda = 1$ and $\psi = 0$ for any affine camera we consider. The two remaining parameters, θ and ϕ, describe the view direction of the

camera with respect to the plane. The affine transformation A between the images of the plane viewed from two affine cameras (θ_1, ϕ_1) and (θ_2, ϕ_2) is eventually obtained by composition:

$$A = A_2 A_1^{-1} \tag{3}$$

2.3 Adding new descriptors to the model

Using these transformation models, it is possible to generate synthetic views by transforming a real view of the scene. In this section, synthetic views are patches of size 100×100 pixels, centered on a point of the model. The scene plane inducing the transformation is given by the point position and the normal to the scene surface at this point.

Once a synthetic view is generated, we use it to add new descriptors to the model. We have two possibilities to extract the descriptors: Either we extract them at the exact location where the point of the model project in the image, or we run a standard SIFT keypoint detection on the synthetic view, giving potentially several candidate descriptors. We use the latter approach for two reasons. First, in order to extract descriptors at a specific location we have to define a scale of extraction, and we did not find a suitable heuristic for this purpose. Second, our goal is to produce descriptors that would be obtained from the virtual viewpoint, which is what we obtain when extracting SIFT keypoints from the synthetic view.

Finally, we add a descriptor from the synthetic view to the set of descriptors attached to a 3D point of the model provided that it has been extracted at a distance smaller than a certain threshold to the reprojection of the 3D point. The distance threshold is set to the average reprojection error in the SfM model.

2.4 Affine or homographic transformation?

We can use the above-mentioned method to enrich SfM models with homographic or affine synthesis, and compare how they improve pose computation. The experimental protocol consists in computing 100 poses from each model to compare the repeatability of the pose computation alongside pose accuracy. Poses are computed from the model using approximate nearest neighbor matching of the descriptors [40], followed by RANSAC filtering of the correspondences between the query image and the 3D model, the pose being eventually estimated by direct least square PnP [6].

Figure 2 illustrates the results typically obtained with a planar scene. We do not have a ground truth for the computed pose, so the accuracy is estimated visually by displaying the computed poses in the scene geometry and projecting some of the scene edges according to the computed poses. Without synthesis, the inlier rate in the image-model matching step is 5% and computed poses are completely scattered, and therefore unreliable. When using the model enriched with affine synthesis, the inlier rate is 20% and the computed poses are grouped around the expected pose. In Figure 2 we see that book edges are projected close to their correct position but not properly aligned. This shows that affine synthesis adds here new descriptors that ease pose computation enough

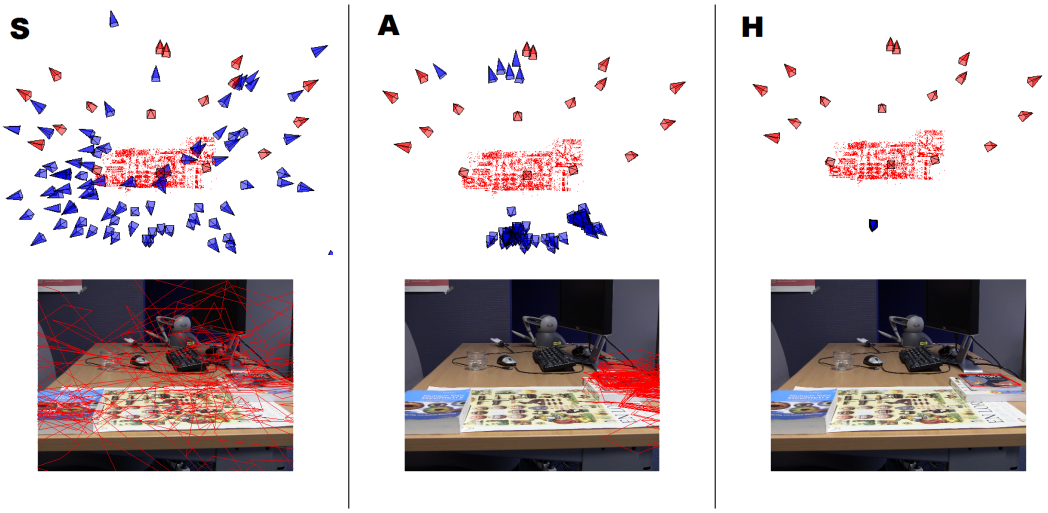

Figure 2: One hundred pose computation results using the SfM model (left), the model enriched with descriptors from affine synthesis (middle) and the model enriched with descriptors from homographic synthesis (right). Top row shows the computed poses (blue) and bottom row shows the reprojection of the edges of the book on the right of the image according to the computed poses. The red viewpoints are the ones used to build the model with SfM.

to get a coarse localization. Finally, with the model enriched by homographic synthesis the inlier rate is 43% and all 100 computed poses are superposed and close to the correct position.

The affine model is generally used when the scene geometry is not available, as in [3]. In our context, the main point in favor of the less complex affine transformations is that they are easier to parameterize, as we do not need to set the camera-scene distance. However, this is the case because we fixed the scale factor $\lambda = 1$ in Equation 2, which relies on a hypothetically perfect invariance of SIFT to scale changes. In practice, it is only valid up to a certain scale change [41]. In addition to this, affine camera model is only a coarse approximation of the epipolar model which is still needed for common focal lengths and scene depths. For these reasons, we use homographic synthesis in the remainder of this article.

3 Geometrically consistent synthesis

Although the method of the previous section proves that view synthesis yields more accurate pose estimation, it does not scale to environments larger than object-centered scenes. In this proof of concept, a patch is indeed synthesized for each model point and each virtual viewpoint. A lot of overlapping patches are computed, which means that some parts are synthesized multiple times. Beside the complexity issue, the po-

Input: A set of images
Output: A point model of the scene, each point being associated with sets of SIFT descriptors, coming either from input images or from synthetic images associated with virtual viewpoints

1. Build an SfM model with, e.g., [52]: 3D points are endowed with SIFT descriptors from the input images
2. Segment the resulting point model into planar patches (Section 3.1.2)
3. Patch synthesis:
   - Sample the virtual viewpoint directions around the planar patches (Section 3.1.1)
   - Generate synthetic views of the planar patches from the real images (Section 3.2.1)
   - Add new descriptors from synthetic views to 3D points by adding the corresponding SIFT descriptors, provided the 3D point is actually visible from the virtual viewpoint (Section 3.2.2)

Figure 3: View synthesis for enriching a point model of the scene.

sitioning of the virtual viewpoint proposed in the previous section is only adapted to small-scale object-centered scenes. We therefore propose to segment the SfM model into independent planar components, virtual viewpoints being positioned with respect to these components and the available real cameras. This permits us synthesizing views of the whole planar components, reducing in turn the algorithmic complexity. For ease of reading, the complete algorithm is summarized in Figure 3.

3.1 Virtual viewpoint positioning

The following sections discuss how to sample virtual viewpoints around a planar patch, and how to segment a point model into a set of planar patches.

3.1.1 View direction sampling

Considering a planar patch, we want any potential view of the patch to be close enough either to one of the simulated viewpoints or to one of the construction views, in order that SIFT features extracted from them can be matched. With affine cameras, the transition tilt, defined in [18], is a good indication of how easy it is to match SIFT features. Although it has been shown in [19] and recalled in Section 2 that homographic synthesis yields better results than affine synthesis, the affine model, as a first order approximation, is sufficient to position virtual viewpoints. If A is the affine map between two images I_1

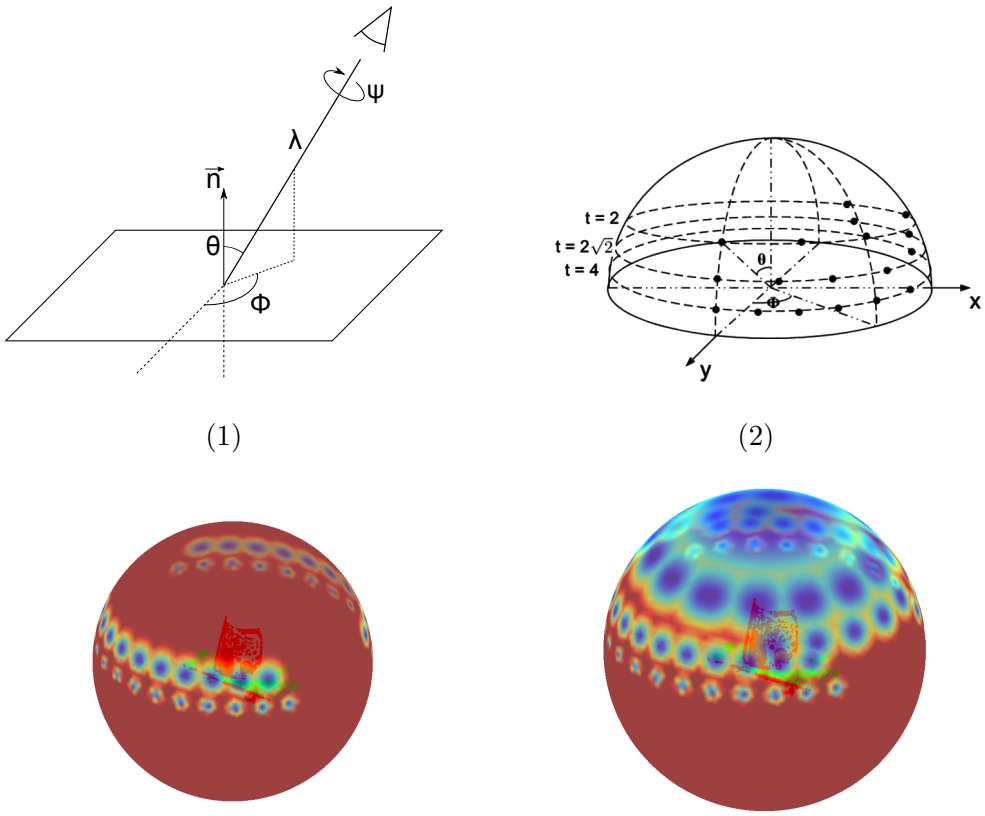

Figure 4: Parameterization of an affine camera pointing to a planar patch (1). Sampling of the virtual viewpoints around a planar patch (from [18]) (2). Map of the transition tilts to the closest viewpoint (blue is 0 and red is greater than the $\sqrt{2}$) for a planar patch of the pot dataset: with respect to the real viewpoints only (3) and with respect to the additional virtual viewpoints (4). The centers of the blue patches correspond to the viewpoint positions. We can see in (4) that most potential viewpoints are within a limited tilt of a real or virtual viewpoint, making it possible to match SIFT features.

and I_2 of a planar scene (that is, $I_2 = AI_1$), then A has a unique decomposition:

$$A = \lambda\, R(\psi)\, T_t\, R(\phi) \tag{4}$$

$$= \lambda \begin{pmatrix} \cos(\psi) & -\sin(\psi) \\ \sin(\psi) & \cos(\psi) \end{pmatrix} \begin{pmatrix} t & 0 \\ 0 & 1 \end{pmatrix} \begin{pmatrix} \cos(\phi) & -\sin(\phi) \\ \sin(\phi) & \cos(\phi) \end{pmatrix} \tag{5}$$

where $R(\psi)$ and $R(\phi)$ are rotation matrices, and $t \geq 1$ is the transition tilt between the two views. If one of the views is fronto-parallel, the parameters correspond to the notations of Figure 4 (1), with $t = 1/\cos(\theta)$. Parameter t expresses how much the view

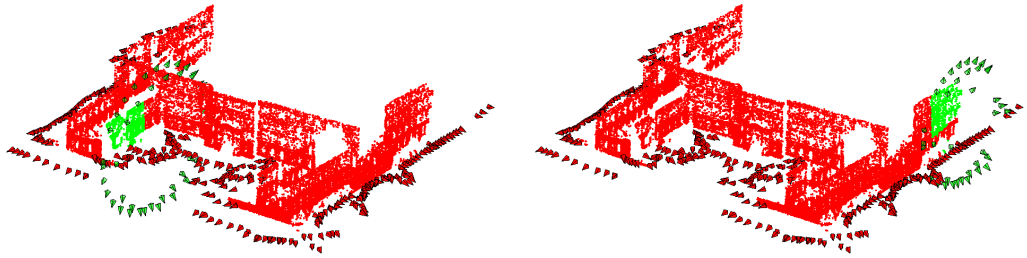

Figure 5: Virtual viewpoint positioning relative to some of the segmented patches (green points). In the first scene, only two rings of virtual viewpoints (in green) are added as the other potential viewpoints would have been close to existing viewpoints (in red).

is flattened out. SIFT features being invariant to similarities, Equation (5) shows that, at fixed t and ϕ, any λ and ψ give the same features. This motivates to position the virtual viewpoints around the planar patch similarly to [18], that is, at (t, ϕ) such that $t = 2^{m/2}$ ($m \in \{1, 2, 3\}$) and $\phi = n\,72°/t$ (n such that ϕ spans $[0, 360°]$. The resulting sampling is shown in Figure 4 (2). It should be noted that only affine cameras are considered in ASIFT [18]. This justifies that ASIFT limits ϕ to $[0, 180°]$ (for symmetry reasons) and does not consider the distance to the scene. Since we consider pinhole cameras, we have to set the distance of the virtual camera to the planar patch. We use the average distance of the real cameras to limit interpolation artifacts during synthesis. This camera positioning strategy is similar to the one used in ASIFT: In both cases, it is based on the observation that tilts, and therefore appearance changes, are larger for view direction with a large angular deviation to the normal to a plane, which calls for a finer sampling of the viewpoints when moving away from the normal.

In order to limit information redundancy, we do not add virtual viewpoints near real viewpoints. Virtual viewpoints are thus added only if the transition tilt to one of the real viewpoints is larger than $\sqrt{2}$. This is illustrated in Figure 4 (3-4).

The next section explains how to segment the scene into planar patches, each one of them being associated with virtual viewpoints through the preceding process, as in Figure 5.

3.1.2 Planar segmentation

It turns out that SfM point clouds are noisy and non-uniformly sampled. We first estimate the normals at each point, via PCA on the neighboring points as in [39]. A simple iterative RANSAC scheme is used: RANSAC (with a fitting criterion based on both the distance between the points and the plane and the consistency of the normals at each point, described in [42]) gives points lying on a plane, which are iteratively removed until 90% of the model points are associated with a plane. Note that the fitting criterion eliminates points around the edges of the scene, the normal of these points being not consistent [43]. This makes the estimation of n, needed for Equation (1), more robust. It should also be noted that this segmentation scheme might fail for large point clouds

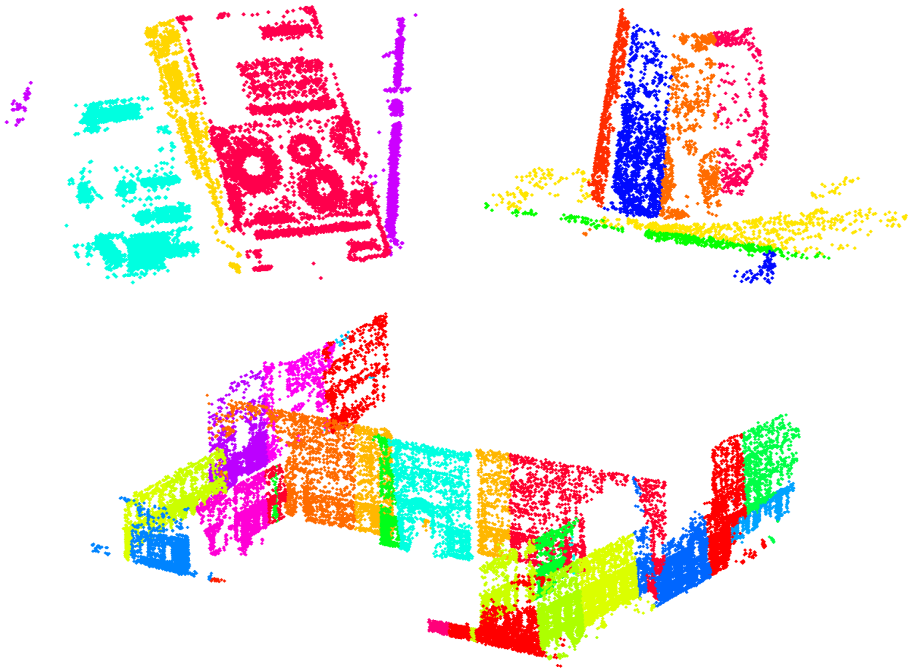

Figure 6: Three segmentation examples on an object-centered scene (top) and a building (bottom). The datasets are the Book, Pot and CAB datasets that can be seen in Figure 10. Each planar patch is in a different color. We can see that the curved surface of the pot is correctly approximated by a set of planar patches.

when each plane only fits a small fraction of the points. This typically occurs for models of the scale of a city. However, the segmentation process is still adapted for models of the scale of a building, which is the case in our application scenario where we expect a rough GPS localization.

Synthesizing the appearance of a patch far away from a virtual camera is likely to suffer from image quantization. This typically happens when synthesizing the appearance of large planes such as building façades. We therefore segment further the planes into smaller sets of points included in square cells oriented along the two principal directions of the plane, and of width equal to the average distance between a point and the cameras that reconstructed it. Note that these cells are not necessarily aligned with the scene edges. Pieces of planes obtained by this segmentation are called *planar patches*.

Illustrative examples of such a planar segmentation are shown in Figure 6. Virtual viewpoints are positioned around the planar patches as in Figure 5.

3.2 Patch synthesis

This section describes the synthesis process. At this step, the scene model is supposed to be segmented into planar patches, each one of them being associated with a set of virtual viewpoints, according to the algorithm of Section 3.1.

3.2.1 Image transformation

For each virtual viewpoint, the aspect of each planar patch is synthesized with the homography given in Equation (1). SIFT descriptors are extracted from the simulated views and associated with the corresponding 3D points. The simulated view is obtained by transformation of a single real view. One could argue that fusing the information from multiple views to generate the new one could be beneficial, using super resolution for example. Unfortunately, the cameras poses associated with the real views are subject to small errors as they are known through the SfM reconstruction. In practice we observe reprojection errors up to 10 pixels. As such, gathering information from multiple views to generate a new one is not practical. This in an intermediate approach between [30] where synthetic views come from full images, and [19] (additional experiments available in [44]) where many small overlapping patches are produced.

The remaining problem is to define from which construction view the synthetic patches should be simulated. As a patch may not have been fully observed in a single construction view, we may have to use several construction views. The views are selected using a greedy approach, by iteratively selecting the construction viewpoints in which the largest number of model points belonging to the considered planar patch are visible. The stopping criterion is that 90% of the patch points are visible from at least one of the viewpoints. A point of the model is considered visible in a construction view if there is a descriptor extracted from this view associated with this point in the SfM model. In all our experiments, we needed at most five views to cover 90% of the points on a planar patch.

Figure 7 shows the set of points in a planar patch and the associated construction view in the same color. SIFT descriptors extracted from simulated views based on the construction view are associated with these 3D points.

This way to perform view synthesis is much more efficient than the method of Section 2 in which all the synthetic patches were synthetized for all the 3D points and all the virtual viewpoints. In this latter case the number of patch synthesis is $nb\_3d\_points \times nb\_virtual\_views$, whereas with the proposed method, the number is less than $nb\_planes \times nb\_cameras\_per\_plane$. In our experiments, the number of planes was less than 20 and the number of cameras used per plane was 32.

3.2.2 Visibility from virtual viewpoints

The preceding procedure simulates the aspect of planar shapes from virtual viewpoints, but it does not take into account potential occlusions from other parts of the scene. This means that it could simulate the appearance of some parts of a patch from a

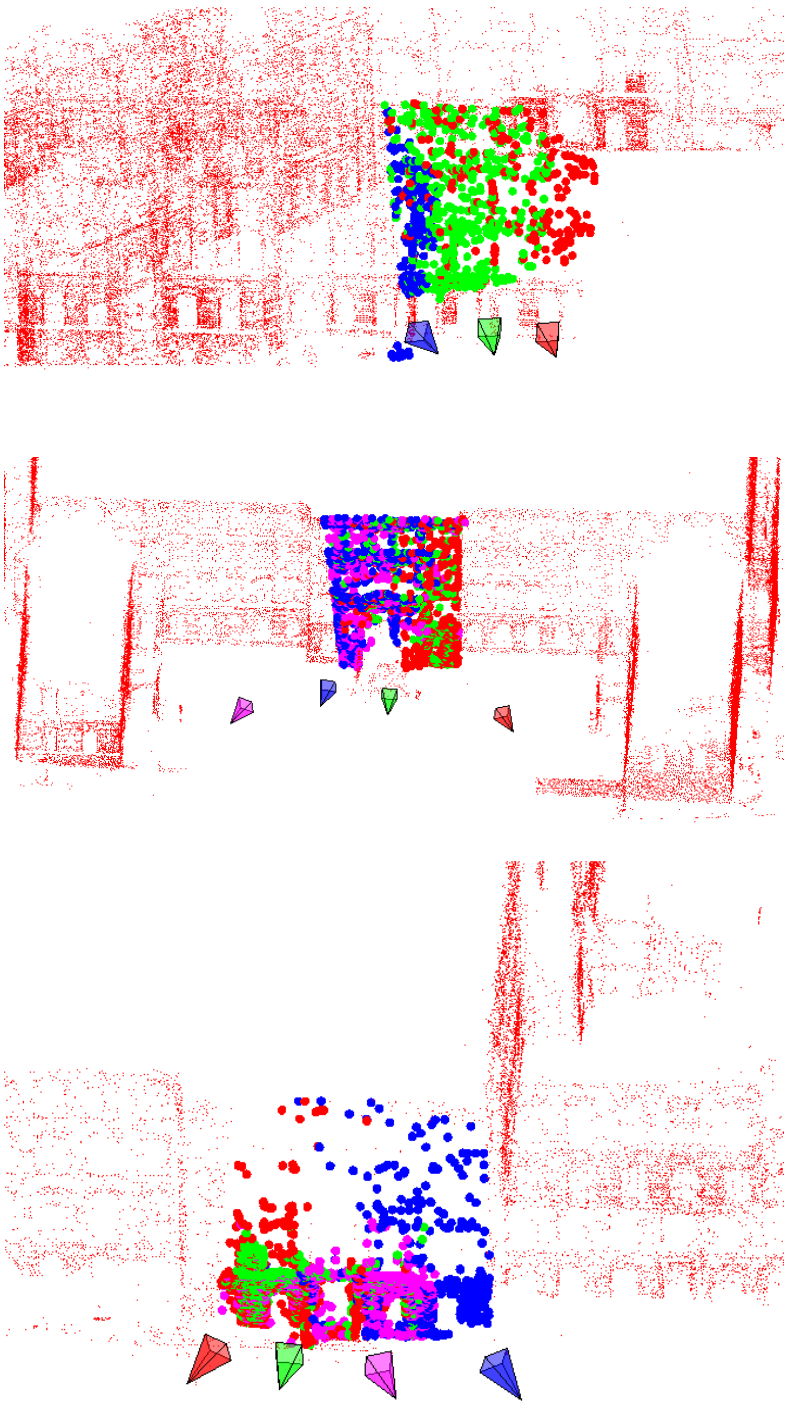

Figure 7: Examples of real viewpoint selection to be used in patch synthesis. The points and their associated construction camera are drawn with the same color.

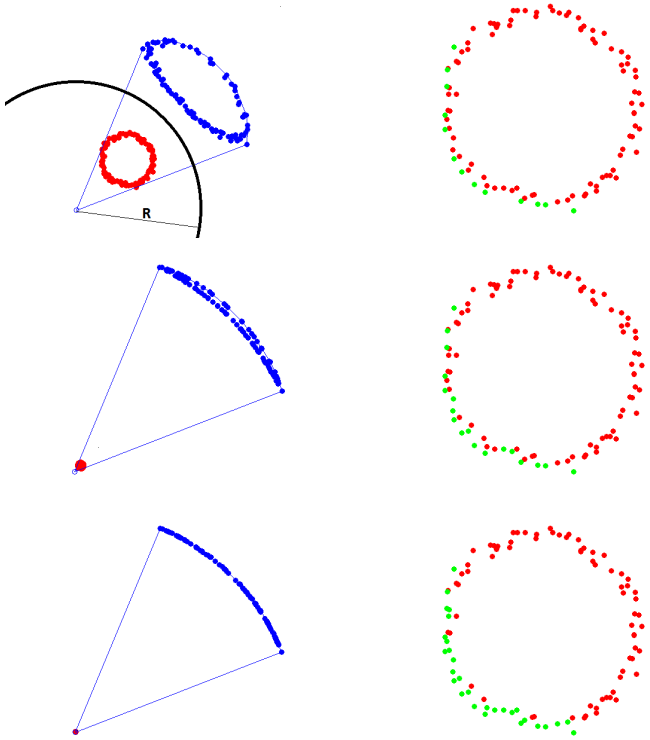

Figure 8: 2D examples of the behavior of the hidden points removal algorithm [45] in the presence of noise. The points (in red) are on a circle, with some added noise. On the left we show the transformed points (in blue) according to different R parameters; on the right, we show which points are marked visible by the algorithm (in green).

position where they are actually not visible. The resulting descriptors would not only unnecessarily increase the complexity of the model, but would also increase the outlier rate in the matching stage.

We therefore impose an additional visibility constraint. The authors of [45] propose an efficient method to directly extract visibility information from the point cloud (no meshing is needed) which performs well on point clouds with a non uniform density. The set of model points visible from a point O is computed as follows. The model points are transformed using the so-called flipping transformation given by:

$$p' = p + 2(R - ||p||)\, p/||p|| \tag{6}$$

where p is a vector from O to a point of the model and R is a smoothing parameter. Let P' be the set of the transformed points. The set of points visible from O belongs to the preimage of the convex hull of $P' \cup O$. It is shown in [45] that the number of visible points increases with R, as shown in Figure 8. This method was originally designed for

point models with noise levels much lower than in an SfM reconstruction, which imposes us to set R carefully.

In order to choose R, we isolate each planar patch and run visibility tests from a fronto-parallel viewpoint for increasing R. Noise-free points lying on a plane should all be visible; this is not the case here. We choose R as the smallest value such that at least 90% of the points in the isolated patch are visible. Whereas R is a critical parameter in the original paper, it is not the case in our application. Indeed, we only want to eliminate points that we are sure they are not visible. To that end, we can over evaluate the value of R, which will result in accepting more points as visible.

It is worth noting that we can tolerate a few mislabelling as pose estimation is performed with RANSAC. The gain from this step is contextual, depending on the presence of occlusions in the scene. It may, however, be significant. For instance, in the pot dataset, the size of the model goes from $160,000$ descriptors to $134,000$ when using this visibility constraint, see Figure 10. All of these discarded descriptors would anyway incorporate spurious data in the model.

4 Efficient guided matching for image-model correspondences

The proposed enrichment process increases the number of descriptors in the model significantly and consequently the time needed by the image-model RANSAC matching step. Indeed, though RANSAC can find the correct solution even in the presence of large outlier rates, the number of samples increases dramatically. There has been a lot of efforts aimed at increasing the efficiency of RANSAC algorithms, see [46] for a review. We propose here a variant of guided-sampling approaches [47] which seeks to modify the sampling process in order to generate preferentially the more plausible correspondences. This method integrates the observation that point correspondences in query image are not distributed on many images of the model but that the correspondences are concentrated on a small set of real or virtual images of the model.

4.1 Hypothesis generation

The first step of pose computation is to generate a set of candidates correspondences between points in the 2D image and points in the 3D model. Note that with the enriched model, each 3D point p is associated with a set D_p of SIFT descriptors generated either from reconstruction views or from our synthetic views. The distance between an image feature f characterized by its descriptor d_f and a model point p is defined as the minimum distance between the image descriptor and any of the model descriptors:

$$dist(f,p) = \min_{d_p \in D_p} (||d_f - d_p||) \tag{7}$$

The set of correspondence hypotheses is the set of all pairs formed by an image feature and its closest point in the model.

As shown in [48], using nearest neighbors is substantially better than bag of words when trying to find correspondences between an image and a model. This is especially true when using our view synthesis approach, because correct descriptors might be isolated in the model. However, the main issue with nearest neighbor search is the complexity. For that reason, we use approximate nearest neighbor algorithm with kd-trees to generate the set of candidate correspondences [49].

4.2 Ordering functions

Depending on the inlier ratio among the image-model correspondences, the matching computation time ranges from one second to one minute in our experiments. To overcome this issue we use a progressive sampling strategy in RANSAC instead of a uniform drawing over the set of candidate 2D/3D correspondences. The critical points of such algorithms are the ordering function, the sampling strategy and the stopping criterion. We describe in Sections 4.2.1 and 4.2.2 two ordering functions that are commonly used to score the quality of the correspondences and then describe our strategy in Section 4.2.3.

4.2.1 Best to second best distance ratio

The score of a match in the nearest neighbor sense between an image feature f and a model point p is given by $\frac{\text{dist}(f,p)}{\text{dist}(f,p')}$ where $dist$ is the distance between the associated descriptors and p' is the second closest point to f in the model, as in Equation 7. Correspondences are sorted along increasing distance ratios. The reason is that reliable correspondences are supposed to be well separated from the second closest point, thus $\text{dist}(f,p) << \text{dist}(f,p')$. This corresponds to the "quality measure" of [47], adapted here to the pose estimation problem.

4.2.2 Inverse best to second best ratio

In [50], the authors argue that the large amount of descriptors in the model makes the best to second best distance ratio unreliable, as $\text{dist}(f,p')$ is more likely to be close to $\text{dist}(f,p)$. They propose to use instead the ratio $\frac{\text{dist}(f,p)}{\text{dist}(f',p)}$ where f' is the second closest feature to p in the image, and to sort the correspondences along increasing values of this ratio.

4.2.3 View count

The previously discussed ordering functions do not fit our problem correctly because they rely solely on photometric information but do not take into account the multi-view context of the matching.

An interesting example of work considering the multi-view context is [51] where an ordering of the samples (i.e. groups of four correspondences) based on co-occurrence is proposed. The score of a group is the number of images in which their respective 3D points appear together. In the context of worldwide localization, co-occurrence over

multiple views is a good yet tractable quality measure. Nevertheless, in the problem of interest, namely strong viewpoint changes between the construction views and the query view, the range of the scene is limited and co-occurrence of correspondences is not informative due to possible point occlusions depending on the viewpoints.

However, taking this idea a little further, we propose an ordering function based on the distribution of correspondences with respect to the views - real or virtual - used to build the model. Given a representative query view, we observe that most of the model descriptors among the correct correspondences are extracted from a relatively small set of views, as shown in Figure 9. The inlier ratio is also significantly higher when we only consider correspondences obtained from these views. This leads us to define the score of a correspondence (f, p) as follows: Let v be the view in which d_p was extracted. The score is the total number of correspondences associated with v in the query image. As a result, all correspondences associated with the same view obtain the same score. Correspondences are then sorted along decreasing view counts. To refine this score, we further sort correspondences with equal view counts according to increasing values of the previously described best to second best distance ratio.

4.3 Random sampling strategy

Let $e_1, e_2, \ldots, e_N$ be the sequence of N candidate correspondences, ordered by one of the function described in the preceding paragraph, and $\mathcal{E}_n = \{e_1, \ldots, e_n\}$ for any $1 \leq n \leq N$ be the nested sets made of the n first correspondences. The progressive sampling strategy consists in drawing sets of minimum four-samples from $\mathcal{E}_4$, then $\mathcal{E}_5$, etc. We use four-samples as estimating a projection matrix requires at least $m = 4$ correspondences. As in PROSAC [47], for a given n, a sample consists in the n-th correspondence e_n and three others randomly chosen in $\mathcal{E}_{n-1}$. This ensures that new samples are actually drawn as n increases. For each n, t_n samples are successively drawn from $\mathcal{E}_n$, t_n being defined as a proportion α of all possible samples among n, that is,

$$t_n = \lceil \alpha \binom{n-1}{3} \rceil \tag{8}$$

where $\lceil x \rceil$ is the smallest integer greater than x. In our experiments, α was fixed to 0.1. In practice, we increment n whenever the current number of iterations t is greater than $t_4 + \cdots + t_n$, the first drawing set being $\mathcal{E}_4$, for which $t_1 = 1$.

4.4 Stopping criterion

4.4.1 Online inlier ratio estimation

As the inlier ratio greatly varies from one experiment to the other, an adaptive stopping criterion is needed. The standard stopping criterion for RANSAC is presented in [38]. The inlier ratio is estimated at runtime according to the cardinality of the current best consensus set $I(P)$ associated with the pose P:

$$\tau = \frac{\#I(P)}{N} \tag{9}$$

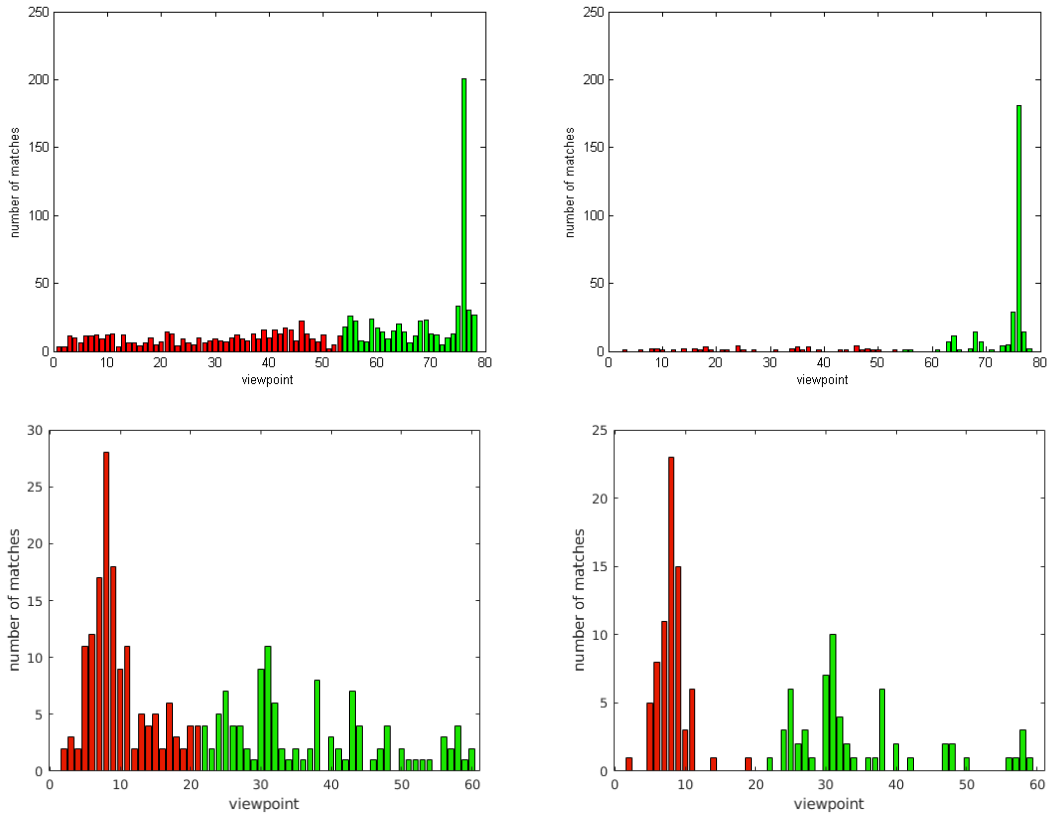

Figure 9: Number of correspondences associated with each viewpoint after the 2D/3D matching step (left) and number of ground truth inliers associated with each viewpoint (right). Contribution from real viewpoints (resp. virtual viewpoints) is in red (resp. in green). The top contributing viewpoints in terms of correspondences are also the ones that produce inliers. Top raw is computed for one image of the Book sequence whereas second raw is computed with one image of the Pot sequence. In the Book sequence, the first image is very far from the construction views with most inliers coming from one synthetic view.

Assuming that the samples are independently drawn and that each correspondence has a probability τ to be correct, it is easy to show that, with probability p^*, a sample made of four correct correspondences is drawn after t^* iterations where

$$t^* = \frac{\log(1-p^*)}{\log(1-\tau^4)} \tag{10}$$

This criterion relies on the estimation of the inlier ratio made on the whole set of correspondences. As the proposed sampling scheme is estimated over $\mathcal{E}_n$, we therefore

estimate the inlier ratio only on the n best correspondences:

$$\tau_n = \frac{\#I_n(P)}{n} \tag{11}$$

where $I_n(P) = I(P) \cap \mathcal{E}_n$ is the consensus set restricted to the n best correspondences in the sense of the chosen ordering function. The number of iterations is calculated as in (10). As the first correspondences have a higher probability of being correct, τ_n is likely to be greater than τ and the required number of iterations to be lower than t^* required by classic RANSAC.

4.4.2 Low values of n

The preceding criterion is, nevertheless, not reliable when n is small (typically $n \leq 10$). In this case, the estimator of the inlier rate (11) has a large variance, which potentially yields unwanted early stops. A minimal number of samples n_0 that have to be drawn to ensure the confidence in the solution must thus be defined. In PROSAC [47], this issue is handled by considering the probability β that *"an incorrect model calculated from a random sample containing an outlier is supported by a correspondence not included in the sample".* This probability is used to prevent stopping after the first few iterations. However, this probability is never known in practice and implementations of the method usually set it to an experimentally validated value.

In our implementation, we define the minimal number of correspondences n_0 as the number of correspondences obtained from the view that produced the more correspondences, i.e., on the set $\mathcal{E}_{n_0}$. This ensures that τ is evaluated on a set of correspondences with a high inlier ratio, according to the observations made in Section 4.2.

Figure 1 summarizes the whole algorithm which is referred to as COSAC in the following.

5 Experimental results

After describing the experimental setup in Section 5.1, we show in Section 5.2 that a model enriched with the proposed method leads to more accurate poses, and even gives accurate poses when pose estimation simply fails without synthesis. Computation times are discussed in Section 5.3. Examples of augmented views are shown in Section 5.4. Finally, the efficiency of the guided matching strategy is demonstrated in Section 5.5.

5.1 Experimental setup

The experimental setup consists in estimating the pose of a query view independent from the construction views, the model being built with VisualSfM [52]. The query view is typically chosen far away from the construction views. Poses are computed from the model using approximate nearest neighbor matching of the descriptors [40], followed by filtering of the query/model correspondences, the pose being eventually estimated by

input : $\mathcal{E}$: set of ordered candidate correspondences
n_0: number of correspondences associated with the predominant view

begin
  $t \leftarrow 0$;
  $n \leftarrow m$;
  $t_{\text{stop}} \leftarrow t_{\max}$;
  while $t < t_{stop}$ **do**
    $t++$;
    if $t > t_4 + ... + t_n$ **then**
      $n++$;
      $\tau_n \leftarrow \dfrac{\#I_{\max(n,n_0)}(\mathcal{P})}{\max(n, n_0)}$;
      $t_{\text{stop}} \leftarrow \dfrac{\log(1-p_0)}{\log(1-\tau_n^m)}$;
    end
    draw sample s in $\mathcal{E}_{n-1}^3 \times \{e_n\}$;
    compute pose $\mathcal{P}$ from s;
    if $I_N(\mathcal{P}) > I_N(\mathcal{P}^*)$ **then**
      $\mathcal{P}^* \leftarrow \mathcal{P}$;
      $\tau_n \leftarrow \dfrac{\#I_{\max(n,n_0)}(\mathcal{P})}{\max(n, n_0)}$;
      $t_{\text{stop}} \leftarrow \dfrac{\log(1-p^*)}{\log(1-\tau_n^m)}$;
    end
  end
  compute final pose $\mathcal{P}_{\text{final}}$ from $I_N(\mathcal{P}^*)$;
end
output: the camera pose $\mathcal{P}_{\text{final}}$

Algorithm 1: The proposed progressive RANSAC algorithm (COSAC). As we can see, the stopping parameter t_{stop} is updated whenever n increases or a larger consensus set is found.

Direct PnP [6]. RANSAC stopping criterion is based on the online estimation of the inlier ratio proposed in [38].

Experiments have been conducted on five datasets which go from a small object to a large building (Figure 10). The size of the scenes is limited to a few objects or buildings, which is a realistic assumption even in city-scale environments if a rough localization is available (through GPS for instance). Poster is a simple planar scene. Pot and book are small object-centered scenes from [53]. Tower is a relatively simple outdoor scene from [54] that essentially consists in two planar facades. CAB is a larger outdoor scene from [55]. This model is significantly larger ($49,000$ points and $325,000$ descriptors, reconstructed from 300 images).

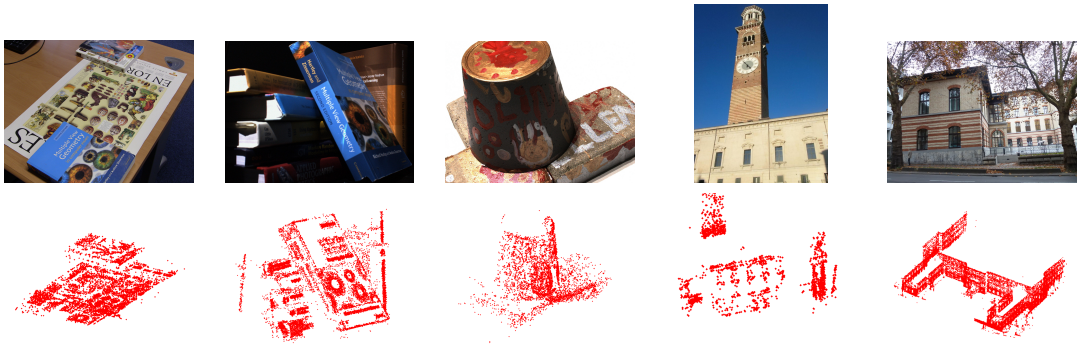

Figure 10: Sample images of five datasets and the reconstructed 3D point clouds. From left to right: poster (17 images), book (53 images), pot (21 images), tower (21 images) and CAB (300 images).

For most cases, we have or we built a SfM reconstruction from a large image sequence but consider for our experiments only a sub-part of the sequence and the associated SfM reconstruction. Test images are views from the large sequence which are far from the sub-sequence. The associated ground truth can thus be computed by registering the SfM reconstruction associated with the whole and to the partial sequence. Registration is based on the set of camera centers which are common to the whole and the partial sequence. First, Procrustes analysis on this set of matched points allows computing scale and displacements between the SFM coordinate frames. All the camera poses can then be expressed into the partial SfM frame (see Appendix A for details). Tests have been also realized in the case of CAB with images taken from Google Street View and, in this case, the recovered position is only visually assessed.

We consider several evaluation criteria in order to assess the accuracy of the pose associated with a query image. The reprojection error and the distance to the camera ground truth are classic criteria. Since augmented reality applications are targeted, we also consider visual means to assess the quality and the stability of an augmented scene.

5.2 Evaluation of pose accuracy

One hundred runs of the pose estimation estimated with the SfM and the enriched models are shown in Figure 11 for the representative pot and tower datasets.

For each query view (column b) the pose computed for each of the 100 trials is displayed using the SFM model (column c) and the enriched model (column d). The improvement brought by the enriched model both on the stability and on the accuracy of the estimated poses can be visually assessed in this figure. The average reprojection error is presented in Table 1. For all the sequences, the reprojection error and the associated standard deviation are considerably reduced. In some cases as poster and tower, the pose estimation computed with the SfM model is very unstable and cannot be used for any localization or AR task, whereas much more accurate estimation is obtained with the enriched model. Finally, there are more inliers obtained with the enriched model than

with the SfM model. Figures 12 and 15 show that they are spread more evenly in the query image, thus leading to a much better accuracy of the computed pose.

In the case of the pot sequence, we have studied the accuracy of the pose with respect to the ground truth. As explained in Section 5.1, the ground truth is obtained by registering the full SfM reconstruction onto the partial SfM used in our experiment. One hundred trials for pose computation both for the SfM based and the enriched method are considered. We then compute for each method the distance from the ground truth to the estimated poses and compute the mean and the standard deviation. For the SfM based method, the mean distance to the ground truth is 1.69 and the standard deviation is 1.26 whereas for the enriched model, the mean distance to the ground truth is 0.277 and the standard deviation is 0.053. To take into account the scale of the scene, we normalize the error with the mean camera-scene distance, which is 8.67. This leads to a relative error on the pose with respect to the mean camera-to-object distance of 19.05% for the SfM based model and of 3.04% for the enriched model. This result is representative of our experiments and proves the improvements in terms of stability and accuracy of our method.

5.3 Computation time and model efficiency

To show the efficiency of our model, we compare in Table 2 the time needed by the offline construction of the model and the time needed by the online pose estimation for both the SfM and the enriched model. The times are obtained on an Intel i7 quad core 2.7 GHz with 16 Gb memory. In all these experiments, the computing time for adding descriptors through patch synthesis was smaller than the reconstruction time, in spite that our implementation of patch synthesis is a MATLAB code, while VisualSfM is a compiled software. Although the construction step is done offline, it is important to note that the additional time needed for computing the enriched model is smaller or of the same order than the SfM reconstruction.

While the enriched models are significantly larger than the initial ones (for instance, it grows from $32,000$ descriptors to $134,000$ descriptors in pot), the pose estimation is not proportionally longer. Table 2 shows that in our experiments the computation times for matching, computed using an approximate nearest neighbor search, are only a few seconds longer when using an enriched model, with a MATLAB implementation. This is primarily because the SIFT descriptors coming from synthesized patches increase the inlier ratio and consequently decrease the number of RANSAC iterations. In all our experiments, the tentative correspondences obtained using the enriched model have always a higher inlier ratio than the ones from the initial model: The inlier ratio increases by 7% for CAB and by 37% for poster. As a result, the time needed by the matching+pose step is most of the time significantly smaller for the enriched model.

5.4 AR applications

When Augmented Reality applications are targeted, another way commonly used to assess the accuracy of the pose is to define objects linked to the 3D model and to check

	SfM model	enriched model
poster	1175±1.72	1.21±0.97
book	3.47±2.31	2.32±1.26
pot	19.79±26.70	4.39±4.50
tower	32.92±50.27	6.72±3.27
CAB	26.94±17.23	15.53±13.72

Table 1: Average pixel reprojection error of 3D scene edges in the query view, plus/minus the standard deviation. The large errors for poster and tower correspond to situation where the RANSAC / PnP step did not actually converge to a reasonable pose, as shown in Figure 11.

	poster	book	pot	tower	CAB
SfM time	6min	11min	15min	5min	18min
synthesis time	3min	6min	10min	4min	9min
	SfM model				
# descriptors	47,643	225,207	32,568	7,774	324,360
matching time	2.53s	3.15s	2.65s	1.48s	7.55s
pose time	35.2	15.64s	10.17s	35.16s	8.29s
	enriched model				
# descriptors	664,848	887,216	134,484	85,949	1,523,298
matching time	5.51s	4.60s	4.38s	3.72s	13.76s
pose time	0.06s	0.80s	0.44s	0.38s	1.30s

Table 2: Computation times for synthesis and for the different steps of pose estimation. Matching times are slightly higher when using an enriched model but pose estimation is significantly faster because of the higher inlier ratio in the tentative correspondences.

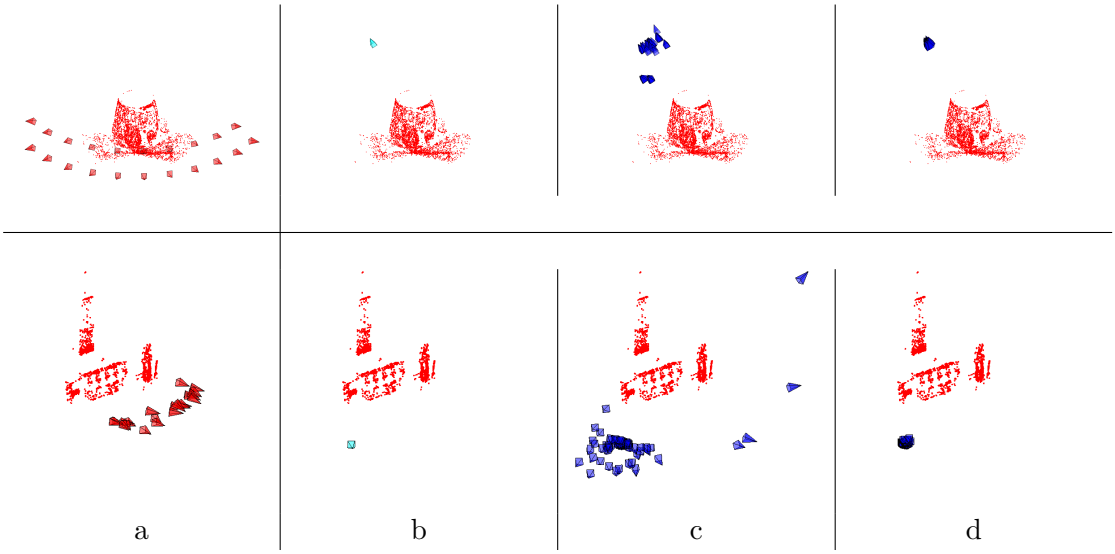

Figure 11: Pose computations on the pot and tower datasets. From left to right: the construction viewpoints, the query viewpoint, 100 tentative pose estimations without synthetic patches, 100 tentative pose estimations with synthetic patches.

that the projected objects really appear as fixed to the scene in the augmented image. In a first step, we consider a set of 3D prominent edges of the scene and check whether their reprojection is visually correctly aligned with the image (Figure 13). These edges were obtained by manually extracting them (column a) in the construction views and reconstructing them using multi-view stereovision. The segments computed with 100 trials using the SFM model are shown in column (b) and with the enriched model in column (c). As can be seen, the reprojected edges are almost superposed with patch synthesis for the three sequences. On the contrary, and depending on the viewpoint changes between the query and the construction views, the reprojection can be inaccurate (poster sequence) or completely wrong (poster and tower sequences) for pose computed with the SfM model.

Views augmented with a 3D chair are provided for the pot sequence in Figure 14. The augmented views are computed when the ground truth pose (left), the pose computed from the SfM (middle) and the pose computed from the enriched model (right) are used. As can be seen, the image computed with the enriched model is very close to the one computed with the ground truth, whereas the one computed with the SfM model is significantly different and appears to be mispositioned on the ground.

Lastly, in the case of the CAB dataset, we consider in Figure 15 a query view taken from Google Street View, for which the acquisition conditions (camera, weather, viewpoint) are significantly different from the ones of the construction views. In this experiment, the intrinsic camera parameters of the query view are estimated from two orthogonal vanishing points using the method described in [56]. Difficulties are of several kinds: Besides the fact that the query viewpoint is far from the initial positions of

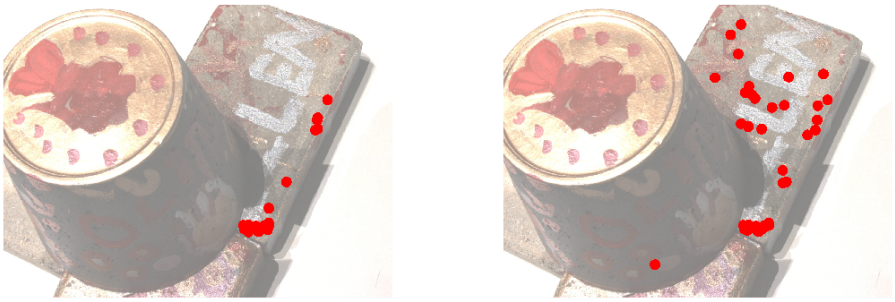

Figure 12: Pot sequence: inlier points when pose is computed from the SFM model (left) and with the enriched model (right).

a b c

Figure 13: Poster, book, pot, and tower datasets. From left to right: query camera and hand-picked scene edges; reprojection of these edges with 100 pose estimations without synthetic patches; reprojection of these edges with 100 pose estimations with synthetic patches.

Figure 14: Pot augmented with a chair (upper right corner of the images) using the ground truth pose, the pose computed from the SfM model and the pose computed from the enriched model (from left to right).

the camera, there are several occluding trees which result in dissimilarities between the model and the query view. In this example the inlier ratio grows from 20% with the SfM model to 27% with the enriched model. Even in this difficult case, the accuracy of the pose is improved using the enriched model.

5.5 Time improvement with the guided matching strategy

The goal of this section is to quantify time improvement with the proposed COSAC guided matching strategy. We first compare the three ordering functions considered in Section 4. Given the n first correspondences ranked by these functions, we plot for two image sequences book and pot (see Figure 16) the ground truth inlier ratio for any n. This inlier ratio can be obtained because the ground truth poses for all images in these sequences is known and we thus can decide whether or not a correspondence is an inlier. As such, these graphs cannot be obtained when running the algorithm.

Figure 16 shows that the view count ordering is always better than the two others. Indeed, its inlier ratio curve in red is almost always above the green and blue curves, especially within the first n_0 correspondences. Consequently, the number of iterations needed by the progressive sampling is significantly smaller.

It should be noted that in experiments, the stopping criterion with the view count ordering is often satisfied before the number of correspondences in the drawing set $\mathcal{E}_n$ reaches n_0. This means that the correspondences in which samples are drawn are all associated with the first, prominent, view. In all of the experiments we did, the maximum number of views used to draw the random samples (before satisfying the stopping criterion) was three, as shown in the last column of Table 3. This shows that preferential sampling among the most productive view is a sound approach.

Our algorithm for pose computation based on view count ordering, COSAC, is compared to the standard RANSAC approach and to an adaptation of PROSAC [47] to pose computation. PROSAC uses two parameters T_N and β. Parameter T_N defines after how many samples the behaviour of PROSAC and RANSAC becomes identical. Parameter β is the probability, that an incorrect model calculated from a random sample containing an outlier is supported by a correspondence not included in the sample. We fixed these parameters to $T_N = 200,000$ and $\beta = 0.01$, as suggested in [47].

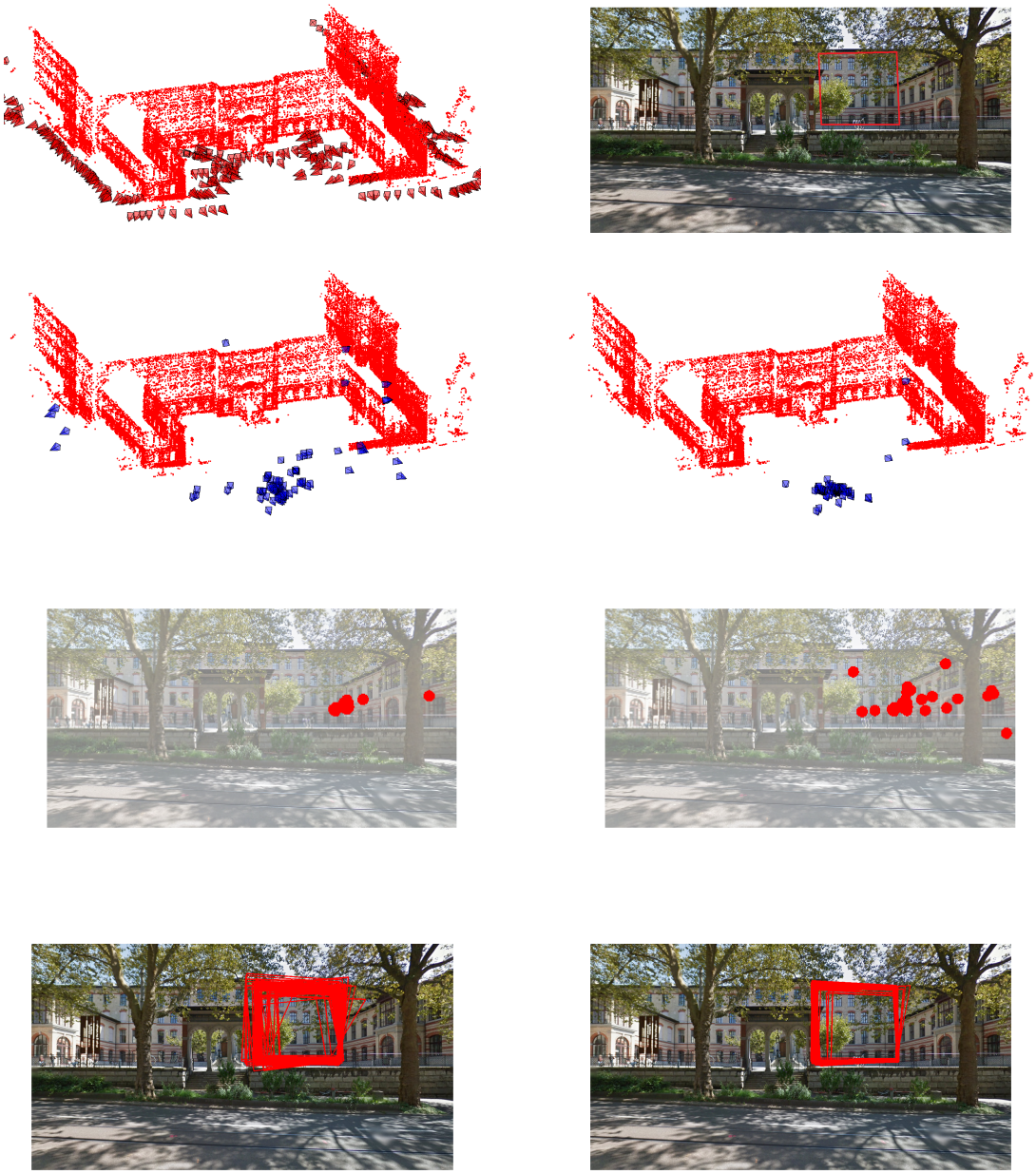

Figure 15: CAB sequence: pose computation with a view taken from Google Street View. First row: the initial SfM model with the associated cameras, and the query view. Second row: 100 trials of pose computation. Third row: the set of inliers. Fourth row: reprojection of a part of the facade (marked in red in the query view) with the computed poses. In second, third, and fourth rows, left and right columns provide results obtained with the SfM model and the enriched model, respectively.

iterations	RANSAC	PROSAC	COSAC	number of views used
book	234±36	413±15	6±0	1
pot	145±30	71±1	69±1	2
CAB	6867±581	625±193	103±0	3

Table 3: Mean and standard deviation of the number of iterations required with the view count ordering function.

Tables 3 and 4 show the required numbers of iterations and the computation times, respectively. Both tables are consistent but are not exactly proportional because of implementation details. Note that all the results have been obtained with a MATLAB implementation so the actual computation times could be improved. However, we can still evaluate performance gains by comparing the numbers of iterations.

As we can see, COSAC converges up to 100 times faster than standard RANSAC. It also converges consistently faster than PROSAC, with an acceleration factor up to 50. The standard deviation of the number of iterations is also significantly lower when using PROSAC or the proposed COSAC.

As in Section 5.2, we compute the average and the standard deviation of the distance of the computed pose to the ground truth on the pot sequence. Table 5 gives the average and the standard deviation of the distance between the computed camera center and the ground truth center. These errors are computed both for the COSAC and the classic RANSAC method over 21 views different from the views used in SfM. The corresponding cameras are shown in green in Figure 17, where the number attached to each camera is the row number in Table 5. The SfM cameras are in red. We give the results for the first 21 camera positions, as camera positions 22 to 27 are very close to the SfM camera positions and give results similar to view 21. The last row of Table 5 gives the median of the error and of the standard deviation over the 21 images. As can be seen the errors are very close with the two methods and the standard deviation is usually smaller with the COSAC method. A standard deviation equal to 0 means that all runs give the same final inlier set from which the pose is calculated. These results prove that, in addition to being computationally efficient, COSAC does not introduce bias in the pose computation. As a sanity check, we also provide the result for classic RANSAC on the raw SfM model, that is, without additional information provided by view synthesis. In this example, camera positions 1 to 3, with large baselines with respect to real views, gives much larger pose errors without view synthesis. When the camera position gets closer to the real views (say, at position no.≥ 12), there is almost no difference between pose estimated with or without view synthesis. Of course, it is at the cost of a much longer computation time, as illustrated by Tables 3 and 4, since much more iterations are needed to find a consistent final inlier set.

time	RANSAC	PROSAC	COSAC
book	1.37	1.62	0.02
pot	15.67	1.94	0.04
CAB	25.48	2.4	0.38

Table 4: Time needed for computing the pose in seconds (averaged over 100 trials).

View no.	COSAC err	COSAC std	inlier number	RANSAC err	RANSAC std	inlier number	SfM err	SfM std	inlier number
1	0.2349	0.0235	48	0.2706	0.0502	47	1.8955	2.0431	34
2	0.4725	0.1949	55	0.2249	0.0553	59	0.5309	0.4518	45
3	0.1658	0.0000	58	0.1502	0.0240	58	0.2704	0.3720	47
4	0.9520	2.8114	35	0.0922	0.0262	38	0.1776	0.2005	42
5	8.2488	5.3150	15	3.2751	4.1145	19	5.7962	5.6045	18
6	3.9636	4.9515	22	0.0695	0.0099	38	0.7857	2.7422	41
7	0.6499	0.5035	38	0.0754	0.0129	67	0.0836	0.0433	68
8	0.1112	0.0089	97	0.1188	0.0361	96	0.1216	0.0348	102
9	0.5567	0.4312	116	0.1382	0.0147	136	0.1547	0.0322	140
10	0.2701	0.1057	122	0.1417	0.0144	128	0.1587	0.0460	108
11	0.2380	0.0263	101	0.2036	0.0353	100	0.2115	0.0361	86
12	0.1624	0.0123	208	0.1528	0.0048	208	0.1667	0.0320	195
13	0.1840	0.0355	290	0.1689	0.0324	290	0.1690	0.0438	261
14	0.1492	0.0034	344	0.1481	0.0068	345	0.1495	0.0231	344
15	0.1092	0.0014	348	0.1200	0.0118	348	0.1408	0.0182	357
16	0.1175	0.0000	291	0.1349	0.0255	291	0.1359	0.0181	292
17	0.0982	0.0037	204	0.1119	0.0295	204	0.0925	0.0146	208
18	0.1294	0.0000	195	0.0770	0.0249	195	0.1019	0.0291	215
19	0.0608	0.0171	122	0.0595	0.0317	122	0.0693	0.0337	137
20	0.0582	0.0035	139	0.0444	0.0175	139	0.0839	0.0456	147
21	0.0471	0.0017	314	0.0552	0.0165	314	0.0687	0.0220	319
med.	0.1658	0.0171		0.1349	0.0249		0.1547	0.0361	

Table 5: Average and standard deviation of the distance between the computed camera center and the ground and number of inliers for the COSAC method (columns 2 to 4) and the classic RANSAC method (columns 5 to 7). The same figures are given for the classic RANSAC method on the raw output of the SfM model without view synthesis (columns 8 to 10). The last row gives median values.

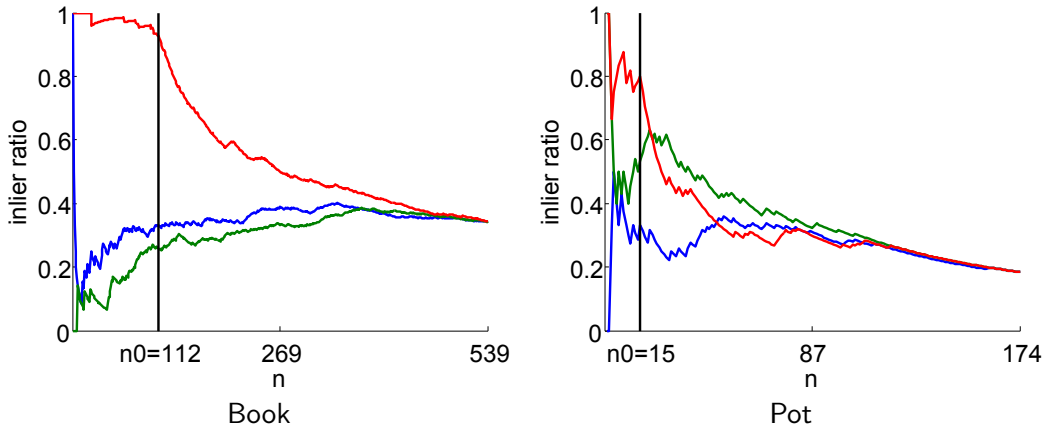

Figure 16: Comparison of the three ordering functions using the ground-truth inlier ratio among the n first correspondences for the book (Left) and the pot (right) sequence. The ordering functions are: best to second best ratio (blue), inverse best to second best ratio (green) and view count (red). The figure shows the graph of the inlier ratio for the n first correspondences against n for the book and pot experiments. The number n_0 of correspondences produced by the top contributing view is marked by the vertical line.

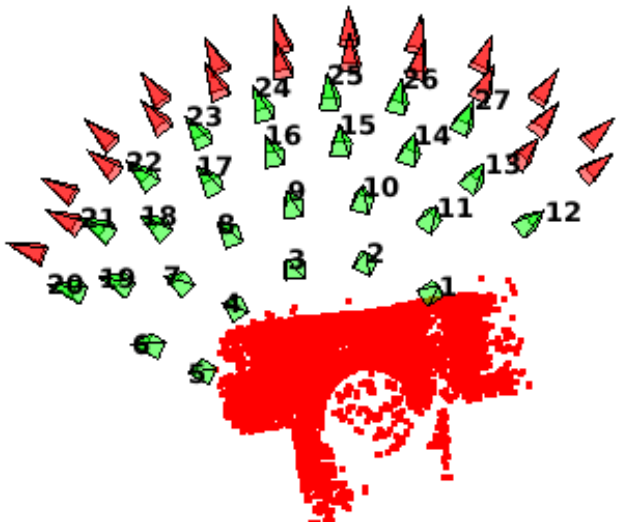

Figure 17: Test with the Pot sequence: cameras in red are used to build the SfM model, whereas cameras in green are used as tests for pose computation.

6 Conclusion

In this paper, we have discussed a view synthesis approach to pose computation. The scene is segmented in several planar patches, around which virtual cameras are positioned. Synthesized views of these patches allow adding new descriptors to the scene model. These new descriptors permit the matching step to go beyond the limited robustness of standard descriptors as SIFT to viewpoint changes, giving an improved pose accuracy. A new guided matching strategy, adapted to this context, makes registration fast.

A Registering cameras from two SfM reconstructions

We here have two SFM reconstructions. One is built from the full image sequence and the second is built from a subsequence. Let $\mathcal{R}_1$ and $\mathcal{R}_2$ be the world coordinate frames attached to these SfM reconstructions.

Given a camera matrix $P_1 = [R_1 T_1]$ expressed in $\mathcal{R}_1$, our goal is to compute its projection matrix $P_2 = [R_2 T_2]$ expressed in $\mathcal{R}_2$.

The rigid + scale transformation (s, R, T) between $\mathcal{R}_1$ and $\mathcal{R}_2$ can be easily recovered using Procrustes analysis from the set of corresponding camera centers. Since we consider the same camera, the viewing coordinates are the same in the two coordinates frames. As the link between the viewing and the world coordinate in homogeneous coordinates is given by $X^1_{vc} = R^1 X^1_{wc} + T^1$, we can deduce:

$$X^2_{vc} = X^1_{vc} = R^1(sRX^2_{wc} + T) + T^1 = s(R^1 R X^2_{wc} + 1/s\,(R^1 T + T^1))$$

Therefore, the expression of the camera matrix P_1 in $\mathcal{R}_2$ is

$$[R^1 R, 1/s\,(R^1 T + T^1)].$$

References

[1] M. Billinghurst, A. Clark, and G. Lee. A survey of augmented reality. *Foundations and Trends on Human-Computer Interaction*, 8(2-3):73–272, 2015.

[2] E. Marchand, H. Uchiyama, and F. Spindler. Pose estimation for augmented reality: a hands-on survey. *IEEE Transactions on Visualization and Computer Graphics*, 22(12):2633–2651, 2016.

[3] B. Charmette, E. Royer, and F. Chausse. Vision-based robot localization based on the efficient matching of planar features. *Machine Vision and Applications*, 27(4):415–436, 2016.

[4] Q. Shan, C. Wu, B. Curless, Y. Furukawa, C. Hernandez, and S. M. Seitz. Accurate geo-registration by ground-to-aerial image matching. In *Proc. International Conference on 3D Vision (3DV)*, 2014.

[5] D. Lowe. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, 60(2):91–110, 2004.

[6] J.A. Hesch and S.I. Roumeliotis. A direct least-squares (DLS) method for PnP. In *Proc. International Conference on Computer Vision*, 2011.

[7] P. Moreels and P. Perona. Evaluation of features detectors and descriptors based on 3D objects. *International Journal of Computer Vision*, 73(3):263–284, 2007.

[8] A. Kendall, M. Grimes, and R. Cipolla. Posenet: A convolutional network for real-time 6-dof camera relocalization. In *The IEEE International Conference on Computer Vision (ICCV)*, 2015.

[9] P. Purkait, C. Zhao, and C. Zach. SPP-Net: Deep absolute pose regression with synthetic views. 2018. Preprint arXiv:1712.03452.

[10] P. Purkait, C. Zhao, and C. Zach. Synthetic view generation for absolute pose regression and image synthesis. In *Proc. British Machine Vision Conference (BMVC)*, 2018.

[11] K. Mikolajczyk, T. Tuytelaars, C. Schmid, A. Zisserman, J. Matas, F. Schaffalitzky, T. Kadir, and L. Van Gool. A comparison of affine region detectors. *International journal of computer vision*, 65(1-2):43–72, 2005.

[12] K. Mikolajczyk and C. Schmid. An affine invariant interest point detector. In *Proc. European Conference on Computer Vision (ECCV)*, 2002.

[13] J. Matas, O. Chum, M. Urban, and T. Pajdla. Robust wide-baseline stereo from maximally stable extremal regions. *Image and vision computing*, 22(10):761–767, 2004.

[14] K. Yi, E. Trulls, V. Lepetit, and P. Fua. LIFT: Learned invariant feature transform. In *Proc. European Conference on Computer Vision (ECCV)*, 2016.

[15] V. Lepetit and P. Fua. Keypoint recognition using randomized trees. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 28(9):1465–1479, 2006.

[16] B. Williams, G. Klein, and I. Reid. Real-time SLAM relocalisation. In *Proc. International Conference on Computer Vision (ICCV)*, 2007.

[17] M. Paulin, J. Revaud, Z. Harchaoui, F. Perronnin, and C. Schmid. Transformation pursuit for image classification. In *Proc. Conference on Computer Vision and Pattern Recognition (CVPR)*, 2014.

[18] J.-M. Morel and G. Yu. ASIFT: A new framework for fully affine invariant image comparison. *SIAM Journal on Imaging Sciences*, 2(2):438–469, 2009.

[19] P. Rolin, M.-O. Berger, and F. Sur. Viewpoint simulation for camera pose estimation from an unstructured scene model. In *Proc. International Conference on Robotics and Automation (ICRA)*, 2015.

[20] S. Savarese and L. Fei-Fei. View synthesis for recognizing unseen poses of object classes. In *Proc. European Conference on Computer Vision (ECCV)*, 2008.

[21] M. Ozuysal, M. Calonder, V. Lepetit, and P. Fua. Fast keypoint recognition using random ferns. *IEEE Transaction on Pattern Analysis and Machine Intelligence*, 32(3):448–461, 2010.

[22] D. Mishkin, J. Matas, and M. Perdoch. MODS: Fast and robust method for two-view matching. *Computer Vision and Image Understanding*, 141:81 – 93, 2015.

[23] D. Mishkin, J. Matas, M. Perdoch, and K. Lenc. WXBS: Wide baseline stereo generalizations. In *Proc. British Machine Vision Conference (BMVC)*, 2015.

[24] M. Rodriguez, J. Delon, and J.-M. Morel. Covering the space of tilts: Application to affine invariant image comparison. *SIAM Journal on Imaging Sciences*, 11(2):1230–1267, 2018.

[25] K. Köser and R. Koch. Perspectively invariant normal features. In *Proc. International Conference on Computer Vision (ICCV)*, 2007.

[26] M. Kushnir and I. Shimshoni. Epipolar geometry estimation for urban scenes with repetitive structures. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 36(12):2381–2395, 2014.

[27] C. Wu, B. Clipp, X. Li, J.-M. Frahm, and M. Pollefeys. 3D model matching with viewpoint-invariant patches (VIP). In *Proc. Conference on Computer Vision and Pattern Recognition (CVPR)*, 2008.

[28] A. Petit, E. Marchand, and K. Kanani. Tracking complex targets for space rendezvous and debris removal applications. In *Proc. International Conference on Intelligent Robots and Systems (IROS)*, 2012.

[29] A. Torii, R. Arandjelović, J. Sivic, M. Okutomi, and T. Pajdla. 24/7 place recognition by view synthesis. In *Proc. Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015.

[30] A. Irschara, C. Zach, J.-M. Frahm, and H. Bischof. From structure-from-motion point clouds to fast location recognition. In *Proc. Conference on Computer Vision and Pattern Recognition (CVPR)*, 2009.

[31] A. Wendel, A. Irschara, and H. Bischof. Natural landmark-based monocular localization for MAVs. In *Proc. International Conference on Robotics and Automation (ICRA)*, 2011.

[32] N. Molton, A. J Davison, and I. Reid. Locally planar patch features for real-time structure from motion. In *Proc. British Machine Vision Conference (BMVC)*, 2004.

[33] Gerhard Reitmayr and Tom W. Drummond. Going out: Robust tracking for outdoor augmented reality. In *Proc. International Symposium on Mixed and Augmented Reality (ISMAR)*, 2006.

[34] G. Simon. Tracking-by-synthesis using point features and pyramidal blurring. In *Proc. International Symposium on Mixed and Augmented Reality (ISMAR)*, 2011.

[35] A. Delaunoy and M. Pollefeys. Photometric bundle adjustment for dense multi-view 3D modeling. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2014.

[36] P. Rolin, M.-O. Berger, and F. Sur. Enhancing pose estimation through efficient patch synthesis. In *Proc. British Machine Vision Conference (BMVC)*, 2016.

[37] M. Fischler and R. Bolles. Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Communications of the ACM*, 24(6):381–395, 1981.

[38] R. I. Hartley and A. Zisserman. *Multiple View Geometry in Computer Vision*. Cambridge University Press, second edition, 2004.

[39] H. Hoppe, T. DeRose, T. Duchamp, J.McDonald, and W. Stuetzle. Surface reconstruction from unorganized points. In *Proc. SIGGRAPH*, 1992.

[40] M. Muja and D. G. Lowe. Scalable nearest neighbor algorithms for high dimensional data. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 36(11):2227–2240, 2014.

[41] J.-M. Morel and G. Yu. Is SIFT scale invariant? *AIMS Inverse Problems and Imaging*, 5(1):115–136, 2011.

[42] R.B. Rusu and S. Cousins. 3D is here: Point Cloud Library (PCL). In *Proc. International Conference on Robotics and Automation (ICRA)*, 2011.

[43] A. Boulch and R. Marlet. Fast and robust normal estimation for point clouds with sharp features. *Computer Graphics Forum*, 31(5):1765–1774, 2012.

[44] P. Rolin, M.-O. Berger, and F. Sur. Simulation de point de vue pour la mise en correspondance et la localisation. *Traitement du Signal*, 32(2-3):169–194, 2015.

[45] S. Katz, A. Tal, and R. Basri. Direct visibility of point sets. *ACM Transactions on Graphics*, 26(3):24, 2007.

[46] R. Raguram, J.-M. Frahm, and M. Pollefeys. A comparative analysis of RANSAC techniques leading to adaptive real-time random sample consensus. In *Proc. European Conference on Computer Vision (ECCV)*, 2008.

[47] O. Chum and J. Matas. Matching with PROSAC - progressive sample consensus. In *Proc. Conference on Computer Vision and Pattern Recognition (CVPR)*, 2005.

[48] O. Boiman, E. Shechtman, and M. Irani. In defense of nearest-neighbor based image classification. In *Proc. Conference on Computer Vision and Pattern Recognition (CVPR)*, 2008.

[49] D.M. Mount and S. Arya. ANN: A library for approximate nearest neighbor searching, 2010. https://www.cs.umd.edu/ mount/ANN/.

[50] Y. Li, N. Snavely, and D. Huttenlocher. Location recognition using prioritized feature matching. In *Proc. European Conference on Computer Vision (ECCV)*, 2010.

[51] Y. Li, S. Noah, D. Huttenlocher, and P. Fua. Worldwide pose estimation using 3D point clouds. In *Proc. European Conference on Computer Vision (ECCV)*, 2012.

[52] C. Wu. VisualSFM: A visual structure from motion system, 2011. http://homes.cs.washington.edu/∼ccwu/vsfm/.

[53] H. Aanæs, A.L. Dahl, and K.S. Pedersen. Interesting interest points. *International Journal of Computer Vision*, 97(1):18–35, 2012.

[54] http://www.diegm.uniud.it/fusiello/demo/samantha/.

[55] https://cvg.ethz.ch/research/symmetries-in-sfm/.

[56] G. Simon, A. Fond, and M.-O. Berger. A simple and effective method to detect orthogonal vanishing points in uncalibrated images of man-made environments. In *Proc. Eurographics*, 2016.