Hybrid Scene Compression for Visual Localization

Federico Camposeco[1] Andrea Cohen[1] Marc Pollefeys[1,2] Torsten Sattler[3]
[1]Department of Computer Science, ETH Zurich [2] Microsoft [3]Chalmers University of Technology

Abstract

Localizing an image w.r.t. a 3D scene model represents a core task for many computer vision applications. An increasing number of real-world applications of visual localization on mobile devices, e.g.*, Augmented Reality or autonomous robots such as drones or self-driving cars, demand localization approaches to minimize storage and bandwidth requirements. Compressing the 3D models used for localization thus becomes a practical necessity. In this work, we introduce a new hybrid compression algorithm that uses a given memory limit in a more effective way. Rather than treating all 3D points equally, it represents a small set of points with full appearance information and an additional, larger set of points with compressed information. This enables our approach to obtain a more complete scene representation without increasing the memory requirements, leading to a superior performance compared to previous compression schemes. As part of our contribution, we show how to handle ambiguous matches arising from point compression during RANSAC. Besides outperforming previous compression techniques in terms of pose accuracy under the same memory constraints, our compression scheme itself is also more efficient. Furthermore, the localization rates and accuracy obtained with our approach are comparable to state-of-the-art feature-based methods, while using a small fraction of the memory.*

1. Introduction

Visual localization constitutes an essential step in 3D computer vision. It plays an important role in large scale Structure-from-Motion (SfM) [1, 15, 36] and SLAM [11]. Visual localization is also a key task for both robotics, *e.g.*, self-driving cars [14], and mobile device applications such as virtual and augmented reality [25].

The classical approach to visual localization requires a sparse 3D scene model, where each point is associated to a 3D position and one or more image descriptors [22, 32]. In order to localize a query image, a set of 2D-3D correspondences can be established by using descriptor matching between features in the image and 3D points in the scene.

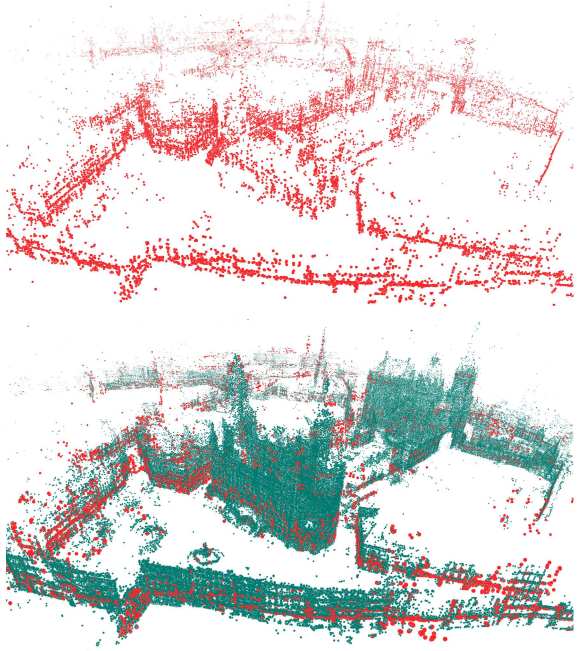
Figure 1. **Comparison of compressed scenes**. (Top) 3D model obtained using a state-of-the-art model compression scheme [9]. (Bottom) Output of our hybrid approach. Full-descriptor points are shown in red and compressed points in green. Both models use the same amount of memory, but ours leads to a better localization performance: For a compression rate of ∼2%, the percentage of localized query images improves from ∼70% to more than 85%.

These matches can then be used for robust pose estimation based on RANSAC and a minimal pose solver [13].

In many applications, coarse localization priors are available, *e.g.*, from GPS or WiFi signals. These priors can be used to coarsely determine the part of the 3D model shown in a given query image within tens to hundreds of meters. Thus, it is often only necessary to match against a part of a larger 3D model. Still, it is important to compress these small- to medium-scale parts, *e.g.*, to be able to transfer them to mobile devices or to be able to handle more requests in a server-side application [28]. In this work, we thus focus on efficiently handling scenes of this size.

Previous works on 3D scene compression mostly fall under two categories: either the models are compressed by selecting a subset of all original 3D points while keeping their full visual information (*i.e.*, full feature descriptors) [9, 12, 22], or all or a subset of the points are kept but their descriptors are compressed [25, 31]. On a different line of work, CNN-based localization approaches such as PoseNet [17–19, 42] or DSAC [4, 5] implicitly represent a scene by the weights stored in a network and can thus also serve as compression methods. However, we show that our approach outperforms CNN-based methods in terms of pose accuracy, memory consumption, or both. Our approach is also more flexible in the presence of scene changes [34] as no expensive retraining of a CNN is required.

In this paper, we present a hybrid 3D scene compression method that selects two subsets of scene points: the first subset consists of very few, carefully selected points with full descriptors, while the second, larger subset consists of points associated to quantized descriptors. The first set of points is used to obtain high-quality correspondences via full descriptor matching that can be used to generate pose hypotheses inside RANSAC. As the number of points in this set is very small due to memory requirements, they might not lead to enough matches for pose verification. Therefore, the second (and larger) set of points for which we only store compressed descriptors is used. While the resulting matches are not unique and are thus not well suited to generate pose hypotheses, they are sufficient for pose verification. Due to compressing the descriptors of the second set, our approach allows us to select significantly more points at the same memory consumption as standard approaches that store full descriptors per 3D point. One central insight of this paper is that selecting more points for pose verification is important for accurate pose estimation. We thus show that storing these two sets of points provides more geometric information for localization while keeping the same memory consumption as other compression methods. An example of our compression output is visualized on the right in Fig. 1.

This work also introduces a RANSAC variation for robust pose estimation that exploits the two types of 3D point subsets, as well as 3D point co-visibility by using a guided sampling strategy (inspired by [21, 32]) that aims to choose only co-visible 3D points to produce minimal subsets. This sampling strategy increases the chances of finding a good minimal sample, which can be critical for accuracy and performance when dealing with lower inlier ratios.

In short, this paper presents the following contributions: **1)** We propose a novel hybrid 3D scene compression algorithm that takes into account the visibility of the 3D points, their coverage of the scene and the visual uniqueness of their appearance. Both coverage and uniqueness are enforced using a novel technique that allows for better coverage while offering faster compression times than the state-of-the-art. **2)** We obtain two different types of compressed points: spatially compressed but fully descriptive points that result in good unique matches, and a larger set of points compressed in appearance space that result in multi-matches. This second set of points has a low memory consumption while being very helpful for avoiding a loss in localization performance. **3)** We introduce a novel RANSAC variant that exploits multi-matches for model evaluation during robust pose estimation and uses a co-visibility-based sampling prior. This modified RANSAC improves both the localization rate and the pose accuracy. **4)** We validate our method on six small- to medium-scale datasets, showing that our compression scheme outperforms state-of-the-art compression methods both in terms of number of localized images, as well as pose estimation accuracy. Our approach is competitive with state-of-the-art localization approaches while using significantly less memory.

2. Related Work

The literature on 3D scene compression for visual localization can be divided into three main categories: image retrieval, neural network-based scene representations, and scene compression for feature-based localization methods.

Image retrieval techniques [30, 31, 44] perform compression by representing each image through a set of visual words (or using an even more compact representation such as the VLAD descriptor [2, 3, 41]), meaning that only little data needs to be kept in memory at all times. These methods may provide an approximate pose estimate via the known pose of the retrieved database images. In order to improve retrieval performance, [16] synthesizes a set of minimal views from the 3D scene in order to cover the whole scene and reduce memory requirements. As a next step, the data needed for accurate pose estimation can either be loaded from disk on demand (*e.g.* 3D points and their descriptors) [8, 16], or the local structure can be recomputed on the fly [34]. As such, the retrieval step is typically efficient, but the next stages are either slow due to loading from disk (which can be alleviated by using compressed models) or due to heavy computations (solving a local structure-from-motion problem). Especially the latter methods trade in memory compression rate for run-time and are not fit for embedded applications.

Rather than matching local features, approaches for pose [17–19, 29, 42] or scene coordinate regression [4, 5, 26] use CNNs for localization. They thus store information about the 3D scene implicitly in the network rather than explicitly in a point cloud. Pose regression approaches such as PoseNet and its variants [17–19, 42] claim to offer compact scene representations. However, our experiments show that our approach outperforms this type of methods both in terms of memory consumption and pose accuracy. Scene coordinate regression approaches predict the coordinates of a corresponding 3D point for a given patch [4, 5, 26, 37]. They thus replace the descriptor matching stage of classical approaches through machine learning. While they set the state-of-the-art in pose accuracy in small-scale scenes [5], they require significantly more memory than our approach.

The approach presented in this paper falls under the third category: scene compression for localization methods based on 2D-to-3D descriptor matching. Such methods compress scenes by selecting a subset of points [9, 12, 22, 39] or by compressing the point descriptors [25, 31]. Our method introduces a hybrid approach, showing that we can combine a set of points with full descriptors with a larger set of points with quantized descriptors. [25] presents an approach which combines both point selection with descriptor quantization. They use both compression techniques on top of each other, maintaining only one small set of points with quantized descriptors. This double compression drastically reduces the localization rate. They compensate by camera pose tracking over multiple frames. In contrast, we are interested in compressing the scene with minimal loss in localization performance when using a single query image.

[22] formulates scene compression as a K-covering problem: The newly compressed scene is composed of a subset of points with high visibility, selected such that at least K of them are seen from each of the database images used to construct the scene. The visibility of a 3D point is defined as the number of database images that participated in the reconstruction of this point. A point with high visibility usually has a more accurate position and a higher probability of being observable from a large set of viewpoints [22]. Ensuring that at least K points are seen in each database image ensures that the compressed scene covers the same area as the original scene. Our work modifies the K-covering formulation in order to enforce a more uniform distribution of the selected 3D points when projected into the database images. Instead of trying to cover K points per image, we first divide each image into q uniformly-sized cells and require the compressed scene to cover K/q 3D points per cell. This improves localization performance w.r.t. previous methods for the same compression rates without an increase in compression times.

Cao and Snavely [9] show that, although visibility is important when choosing the best subset of points to represent the 3D scene, the distinctiveness, or visual uniqueness, of the point descriptors should also be taken into account during compression. This ensures that points with an unique descriptor are selected, which improves matching performance, especially for high compression rates. Distinctiveness is introduced to the compression by checking that each new point added to the compressed scene has a minimum descriptor distance to all the points that have already been chosen. This procedure of comparing each new point to all previously selected points is computationally expensive. In this work, we also extend the K-covering algorithm by taking distinctiveness into account. We show that it is sufficient to approximate similarity via quantization rather than explicitly computing descriptor distances. We exploit quantization both during K-cover to choose a proper set of distinctive and descriptive 3D points, as well as for selection and appearance compression of the second, larger subset of 3D points. The use of quantization allows us to decrease compression run-times considerably w.r.t. [9]. As such, our approach is well-suited for scenarios in which the scene model needs to be re-compressed frequently, *e.g.*, in dynamic scenes in which the geometry of the scene changes over time. At the same time, our hybrid sets of points increase localization performance w.r.t. [9].

Similarly to our method, [31] also exploits one-to-many 2D-3D matches or multi-matches via quantization. Yet, they attempt to resolve these ambiguities before pose estimation by ensuring that matches are locally unique. Non-unique matches are simply discarded. In contrast, our work actively uses ambiguous multi-matches during the hypothesis evaluation step of pose estimation. This eliminates the chance of rejecting correct matches before RANSAC. [31] requires a large codebook of 16M words to avoid introducing too many ambiguous matches. Thus, [31] can be used to compress large-scale scenes, but is unsuitable for the smaller scene fragments used in practice [34].

[27] propose a RANSAC variant that handles multi-matches based on matching probabilities (computed from matching scores) and their involvement in (un)successful hypotheses. Their method is not directly applicable in our setting due to our binary similarity measure based on whether descriptors fall into the same visual word. But sampling multi-matches during hypothesis generation in our RANSAC if they were inliers to previously generated poses could potentially allow even higher compression rates.

3. 3D Scene Compression

A 3D scene is composed of 3D points and database images. Each 3D point is associated to a set of SIFT descriptors corresponding to the image features from which the point was triangulated. These descriptors can then be averaged into a single SIFT descriptor that describes the appearance of that point to reduce memory consumption [22].

In order to localize a given image w.r.t. the 3D scene, 2D-to-3D matches are first established, which are then used in RANSAC-based pose estimation [13]. As in [32], we employ a vocabulary-based approach to feature matching. Using a pre-built vocabulary-tree, we first assign each 3D average descriptor to its corresponding K-means cell (visual word). At search time, each query image descriptor is assigned to its closest visual word w. We then select every 3D point in the scene that has the same visual word and search for a nearest neighbor in descriptor space among those selected points. As it will be seen next, we will make use of this vocabulary tree again during compression. This will allow us to both compress and query the 3D scene with a single visual vocabulary. This yields faster compression times and better localization performance w.r.t. previous work.

After averaging the SIFT descriptors, most of the memory consumption of the 3D scene is concentrated in the averaged descriptors of the points. Therefore, we will aim to reduce memory consumption by 1) reducing the set of 3D points and 2) compressing the descriptor of a subset of non-previously selected points using a visual word.

3.1. Reducing the Number of 3D Points

Let the initial set of 3D points be $\mathcal{P}$. The goal of compression is to select a subset $\mathcal{P}' \subset \mathcal{P}$ such that $|\mathcal{P}'|$ is minimal under the condition that $\mathcal{P}'$ can be used to localize as many query images as possible. To tackle this problem, we begin with the assumption that the spatial distribution of query images will be close to the distribution of the database. This common assumption [7, 22] is sensible given that local features are not invariant to viewpoint changes.

We aim to select points such that each of the database cameras sees *enough* 3D points, *i.e.*, we want to enforce that each of the cameras in the database are *covered* by at least K points. This problem of choosing the optimal set of 3D points that cover all database cameras is an instance of the Set Cover Problem (or K-cover when at least K points must be seen from each camera) [22], which is NP-hard. Furthermore, by choosing two 3D points whose descriptors are too similar, a query image descriptor could be matched to either of those points, resulting in ambiguities. This problem arises in the presence of repeated structures that produce very similar descriptors that are actually meters apart. Therefore, the *visual uniqueness* of the selected 3D points should also be taken into account.

Our primary concern is to select points which have a complete coverage of the scene, while penalizing points whose descriptors can be confused with other descriptors selected. We can thus, as proposed by [9], cast this problem into an instance of the Weighted K-cover problem [23], where the weight reflects the discriminative power of the descriptor associated with a point. Since the Weighted K-cover problem is NP-hard, we will follow common practice [16, 22, 32] and employ a greedy approach in order to arrive at an approximate solution.

For compression, we make use of the visibility graph [22] defined by each SfM model. The nodes of this bipartite graph correspond to the 3D points and database images in the reconstruction. The graph contains an undirected edge between a point node and an image node if the point was triangulated from the image. Let the binary matrix M represent the visibility graph of the SfM model[1], where $M_{i,j}$ is 1 if the i-th image I_i observes the j-th point p_j. Starting from an empty set of points $\mathcal{P}'$, our objective at each iteration is to find p_j that maximizes the gain

$$\mathcal{G}(p_j, \mathcal{P}') = \boldsymbol{\alpha}(p_j, \mathcal{P}') \cdot \sum_{I_i \in \mathcal{I} \setminus C} M_{i,j} \quad , \tag{1}$$

where C is the set of images that have already been covered by K points. The weights $\boldsymbol{\alpha}(p_j, \mathcal{P}')$ measure how visually distinctive point p_j is w.r.t. the already selected points $\mathcal{P}'$.

In [9], $\boldsymbol{\alpha}$ is computed by comparing the descriptor of each candidate p_j against all of the already selected 3D points. This produces a rather slow procedure, since each descriptor comparison can be costly and an adaptive search structure (such as a KD-Tree) cannot be used as the size of $\mathcal{P}'$ increases with each iteration.

Instead, we exploit the fact that we perform vocabulary-based image localization using a pre-built vocabulary tree, *i.e.*, we assign each 3D point to a visual word before 2D-to-3D matching. Thus, we define the weighting term as

$$\boldsymbol{\alpha}(p_j, \mathcal{P}') = 1 - \frac{|P'_w|}{\beta} \quad . \tag{2}$$

Here, w is the visual word of the descriptor of point p_j, P'_w is the set of selected 3D points with descriptors assigned to w, and β is the maximum number of allowed points per word (set to 10 in our experiments). We thus penalize the inclusion of 3D points whose visual word is too populated with a linear function. The intuition behind this is that the number of points assigned to a visual word gives an idea of how unique points in this visual word are.

In contrast to [9], we opt to enforce that not only each image is covered by K 3D points but that the distribution of those points on the images is as even as possible. We thus subdivide each database image into a grid with q equal-area cells. We then regard the image cells instead of the images themselves as the elements to be covered by the 3D points. Let $c_h \in \mathcal{I}^c$, where $h = 1, \ldots, q \times N$ and N is the total number of database images, be the h-th image cell to be covered. $\mathcal{I}^c$ represents the set of all images cells and C^c the set of cells already covered. The gain to maximize in each iteration thus becomes

$$\mathcal{G}(p_j, \mathcal{P}') = \boldsymbol{\alpha}(p_j, \mathcal{P}') \cdot \sum_{c_h \in \mathcal{I}^c \setminus C^c} M_{i,j} \quad . \tag{3}$$

Although the number of "cameras" to cover increases to $q \times K$, the number of non-zero entries remains the same. We thus found no noticeable increase in compression runtime since we also reduce the number of 3D points that are enforced to be viewed by each image cell to K/q. As we will show in Sec. 5, our approach has a positive impact on the localization rates: the point selection has a more even coverage of the scene and prevents the selection process to be biased towards highly structured or textured parts of the scene, which might not necessarily be visible at query time.

[1] In practice, M is very sparse and typically stored as an adjacency list.

Additionally, a uniform distribution in 2D typically leads to more stable pose estimates compared to finding all matches in a single region of an image [16].

Our approach achieves the same complexity reduction as previous work: for n points and m images, the average number of points per image is n/m. K-cover methods such as [9, 22] reduce this number to $\Theta(K)$. The space reduction for any constant K is thus $\Theta(n/(K \cdot m))$, *i.e.*, linear in the number of 3D points. Eq. 3 replaces m images by $q \cdot m$ cells and K with K/q, resulting in the same space complexity.

Once the greedy Weighted Set Cover algorithm is completed, we are left with a subset of 3D points $\mathcal{P}' \subset \mathcal{P}$ which should maximize scene coverage and descriptor uniqueness. Instead of discarding the rest of the 3D points [9], we choose to select a second subset $\mathcal{P}'' \subset \mathcal{P}$ of points for which only a compressed descriptor will be kept. This procedure is detailed next.

3.2. Selecting Multi-Matches

As previously mentioned, the most memory-consuming part of a 3D scene are the feature descriptors. As such, we may only select a small number of them to ensure that we substantially reduce the amount of memory required to represent a 3D model. If we only keep the 3D points selected by our Set Covering procedure, we are prone to miss matches due to the imperfect nature of the descriptors (features might not match due to large viewpoint changes) and the matching procedure. Additionally, only a few points might be visible in the query image as the database images are only an approximation to the set of all viewpoints. To mitigate this, we select a second subset of 3D points $\mathcal{P}'' \subset \mathcal{P} \setminus \mathcal{P}'$, where we will keep only the quantized descriptor (word assignment) for each selected 3D point and its 3D location. The purpose of this second subset of *compressed points* is the following: while the points in $\mathcal{P}'$ result in high-quality matches that can be used for generating pose hypotheses during RANSAC, they might not be enough to properly verify these hypotheses. The set $\mathcal{P}''$ is thus used for verification. As these last matches are only used with a given hypothesis, they do not need to be unique as they are disambiguated by the pose. Therefore, they can be stored at little memory consumption.

We note that for each full 3D point we must store the 3D point position, the full mean descriptor, and a list of cameras that see the 3D point (which will be used within RANSAC to discard bad samples, see Sec. 4). For compressed points, we only need to store the 3D position and one visual word index (1 integer). For 128 byte SIFT descriptor and a 32-bit visual word, we measure that on average we can store 26.5 compressed descriptor for each full descriptor. Since the only way of distinguishing compressed points is by their visual word, their selection process will thus focus on word uniqueness. To select the points for compression, we use the following procedure: 1) score each point by its word occupancy (assigning a high score to low occupancy) and 2) select the X points that have the highest score. This procedure selects points without any regard to their camera coverage. However, this criterion has already been prioritized by the points selected in the first step of the compression.

Since we will, in general, have more than one 3D point with the same visual word, we must consider each 2D-to-3D match against compressed points as a *multi-match* (which will be handled differently within RANSAC, as explained in the next section). Nevertheless, the selection procedure for our compressed 3D points already minimizes the number of points in each word. Thus, we can expect to have a low number of multi-matches per query descriptor.

4. Multi-Match RANSAC

As explained in Sec. 3, matching the 2D features of the query image against $\mathcal{P}'$ using a standard ratio test [24] provides a set of one-to-one 2D-3D matches. We will denote this set of matches as $\mathcal{U}$. Matching against $\mathcal{P}''$ results in a set of multi-matches, denoted as $\mathcal{W}$. The total set of matches will be referred to as $\mathcal{M} = \{\mathcal{U} \cup \mathcal{W}\}$. This section explains how $\mathcal{M}$ can be used during the robust RANSAC-based pose estimation stage in order to improve both localization rates and run-time when dealing with highly compressed scenes. Alg. 1 summarizes our approach.

Robust pose estimation is usually achieved by running RANSAC [13] with a minimal geometric pose solver (*e.g.* P3P [20] for the calibrated setting or P4P [6] for the unknown focal length case). A general RANSAC approach randomly selects minimal samples from $\mathcal{M}$, generates pose hypotheses with the minimal solver, and then evaluates these poses by counting the number of inliers according to a given threshold on the reprojection error. However, the lower the inlier ratio, the more samples are required by RANSAC to ensure that the correct pose is found with a certain probability. Indeed, the number of samples required increases exponentially with the outlier ratio. Thus, using the multi-matches $\mathcal{W}$ during the sampling stage potentially introduces a high number of outliers, resulting in a prohibitively large number of RANSAC iterations. Therefore, our RANSAC variant limits the sampling only to the set $\mathcal{U}$ of unique matches, since these matches have a higher probability of being actually good matches. Once a model has been sampled using only unique matches, it needs to be evaluated. Having a larger number of inliers increases the confidence in the quality of a sampled model. Therefore, the whole set of matches $\mathcal{M}$, including the multi-matches $\mathcal{W}$, is used to evaluate each pose.

In order to further reduce the number of RANSAC iterations, previous methods such as [10] use a guided-sampling strategy in order to increase the chances of finding a good sample. Inspired by this strategy, we propose an efficient variation of the method introduced by [21] in the form of a simple co-visibility-based sampling strategy. This addresses the problem that even in $\mathcal{U}$ inlier ratios can be very small in a highly compressed scene and that it is therefore not enough to just limit sampling to this subset. It is desirable that samples should be drawn from sets of matches that correspond to co-visible points, *i.e.*, those that are seen from the same camera or neighboring cameras, as these matches have a higher chance of being geometrically consistent. Similarly to [21], our sampling should thus increase the probability of drawing from such subsets of matches.

More specifically, given $\mathcal{U} = \{s_j \leftrightarrow p_j\}$, where s_j denotes a 2D feature and p_j a 3D point, and n the cardinality of a minimal sample $\mathcal{S} \subset \mathcal{U}$ drawn in RANSAC, we would like all points $p_i \in \mathcal{S}$ to be co-visible. In [21], points are considered to be co-visible if they are all observed together in at least one database image. However, we found this definition of co-visibility to be a rather slow at sampling time due to the need to perform multiple set intersection, requiring about 20ms per RANSAC iteration.

We therefore use a slightly different definition of co-visibility: for a set of points $\mathcal{Q} = \{p_i, i = 1 \ldots n\}$, let $\mathcal{C}(p_i), p_i \in \mathcal{Q}$, be the set of database images in the scene from which point p_i was reconstructed. The set $\mathcal{Q}$ can be represented as a graph $\mathcal{G}_{\mathcal{Q}}$, where each point p_i is a node and an edge is added between a pair of points $p_i, p_j, i \neq j$, if they share at least one image in the scene, *i.e.*, if $\mathcal{C}(p_i) \cap \mathcal{C}(p_j) \neq \emptyset$. We consider the points in $\mathcal{Q}$ as co-visible if each pair of points in $\mathcal{Q}$ are at most 2 edges apart in the graph $\mathcal{G}_{\mathcal{Q}}$.

This definition allows the efficient sampling ($\sim$1μs) of sets of covisible points: we choose a random match $m_1 = s_{j_1} \leftrightarrow p_{j_1}$ from $\mathcal{U}$ as the first element of $\mathcal{S}$. We sequentially draw the following m_i, $i = 2 \ldots n$, samples and test whether they have an image in common with p_{j_1}. If they do, then the sample m_i is accepted and included in $\mathcal{S}$. Otherwise, m_i is dropped and re-sampled. If no valid m_i can be found after F attempts, we drop the complete sample $\mathcal{S}$. Both for speed and simplicity, we choose to always look for intersections with $\mathcal{C}(p_{j_1})$ instead of checking all of the points that are already part of $\mathcal{S}$. Since the first match is drawn uniformly at random, this choice does not restrict sampling. As a result of our simple sampling strategy, only potentially good minimal subsets are used for pose estimation and evaluation.

5. Experimental Evaluation

In this paper, we focus on a hybrid scene compression that enables usage of a pre-computed 3D sparse scene for visual localization. Thus, especially for computationally constrained platforms, we are interested in: 1. the run-time of the compression procedure, 2. the memory reduction with respect to using an uncompressed scene, 3. the localization rate, and 4. the accuracy of the obtained poses.

Algorithm 1 Modified RANSAC

```
Require: Minimal sample size $n$, minimal solver $PnP$, matches $\mathcal{M} = \{\mathcal{U} \cup \mathcal{W}\}$,
    max. number $F$ of sample trials, max. number $T$ of iterations, inlier threshold $\sigma$
 1: repeat
 2:     $\mathcal{S} = \emptyset, f = 0$
 3:     Randomly sample $m_1 = s_1 \leftrightarrow p_1$ from $\mathcal{U}$
 4:     $\mathcal{S} \leftarrow m_1$
 5:     while $|\mathcal{S}| < n$ and $f < F$ do
 6:         Randomly sample $m_i = s_i \leftrightarrow p_i$ from $\mathcal{U}$
 7:         if $\mathcal{C}(p_i) \cap \mathcal{C}(p_1) \neq \emptyset$ then
 8:             $\mathcal{S} \leftarrow m_i$
 9:         else
10:             $f \leftarrow f + 1$
11:     if $|\mathcal{S}| < n$ then
12:         Jump to next iteration at line 1
13:     Compute pose $\theta = PnP(\mathcal{S})$
14:     for all $m_i \in \mathcal{M}$ do
15:         if $e(m_i; \theta) < \sigma$ then
16:             $Inliers(\theta) \leftarrow Inliers(\theta) + 1$
17:     if $Inliers(\theta) > Inliers(\theta^*)$ then
18:         $\theta^* \leftarrow \theta$
19: until $T$ iterations are reached
20: return best model $\theta^*$
```

Grid	MR	Median time (ms)		% reg.	Median error	
		Query	RANSAC	images	Pos. (m)	Rot (°)
		181	2.4	95.7%	2.09	0.42
●		182	**1.5**	96.0%	2.02	0.37
	●	190	7.7	97.4%	2.09	0.39
●	●	**191**	6.5	**97.6%**	**1.97**	**0.35**

Table 1. Comparison of all the variants of our method for the Dubrovnik dataset compressed to 1.5% of its original size. Where a ● in "Grid" means we use image subdivisions and a ● in MR stands for the usage of our modified RANSAC.

In order to properly validate the proposed method, we evaluate it with 6 different real-world datasets (*cf*. Tab. 2). We chose these datasets since they are representative of small- to medium-scale scenes and the use-cases we aim to tackle with our method. By including datasets that have a wide variation in the number of points and imaging conditions, we show that our methods can work well in different scenarios. For the evaluation, we use SIFT [24] descriptors throughout and use a single 6K-word visual vocabulary, pre-computed using the Dubrovnik dataset [22]. In practice, we did not observe strong evidence that suggested that using a vocabulary trained per dataset yielded consistently better results. This is also shown in [32]. For the pose computation, we use the P4P solver by [6].

This section is organized as follows: first, we do an ablation study that analyzes the impact of the compression method introduced in Sec. 3 and the RANSAC variant of Sec. 4. Next, the best setting of our method is compared against previous works, in terms of compression run-time, localization rates, and localization accuracy.

5.1. Ablation Study

Tab. 1 compares four different settings of our method. We test its performance with and without image subdivisions and with and without our modified RANSAC on the Dubrovnik dataset. When not using our modified RANSAC, we do not take into account $\mathcal{P}''$ and its cor-

Dataset	# DB images	# 3D points	# Query images	Compression Time [s]	
				Ours	KCD [9]
Dubrovnik [22]	6,044	1.88M	800	**27.9**	50.4
Aachen [35]	3,047	1.54M	369	**21.6**	45.5
King's College [19]	1,220	503K	343	**1.7**	2.5
Old Hospital [19]	895	308K	182	**3.6**	12.5
Shop Facade [19]	231	82K	103	**0.43**	1.15
St Mary's Church [19]	1,487	667K	530	**8.74**	26.1

Table 2. Datasets used and compression times (to compress the model to 1.5% of the original points) for our method compared to "KCD" [9] (using their implementation). The run-times for our method are significantly faster than those of [9].

responding multi-matches. Tab. 1 shows an improvement with each of the proposed modifications. In the following, we refer to using both image subdivisions and our modified RANSAC as "Ours".

5.2. Compression Run-time

For each dataset in Tab. 2, we compress the scene to 1.5% of its size (factoring both the number of full-descriptor 3D points and compressed points into this memory budget) and compare the run-times of our method to the state-of-the-art in model compression [9]. We have chosen 1.5% to be the default compression rate since it exhibits a good trade-off between performance and memory size. However, the relative speed-up in compression time of our method vs. [9] does not vary for different compression rates. Tab. 2 shows that we achieve consistently lower compression times than [9]. This is due to the fact that, as explained in Sec. 3, our method only needs to check the occupancy of a visual word to figure out the cost of selecting a 3D point. Our approach is thus more suitable in scenarios where the scene changes over time, *e.g.*, due to seasonal changes [33], and the 3D model used for localization needs to be adapted frequently.

5.3. Registration Rates

For the second experiment (see Fig. 2), we want to evaluate the efficiency of our compression and pose estimation methods in terms of image registration power. Following [9, 21, 22, 32], we consider an image as registered if the best pose found by RANSAC has at least 12 inliers.

To make the compression rate comparison with [9] fair, we select a combination of compressed and non-compressed points in order to match the memory used by [9] in each experiment. This is done in the following way: for a given budget of M 3D points selected by [9], we divide the budget into two parts. The first 75% of the budget is utilized by selecting $0.75M$ uncompressed points by performing our weighted K-cover. The second 25% of the budget is then spent on compressed 3D points for multi-matches. The number of compressed points selected is actually larger than $0.25M$, since we can store (on average) 26.5 compressed points for each full point (see Sec. 3). Thus, we select $0.25 \times 26.5M$ compressed points. For example, for

Method	MB Used	#reg. images	Median pos. error
Ours (1.5% 3D pts)	**3.79**	782	1.97m
PoseNet [18]	~50	800	7.9m
DenseVLAD [41]	94.44	800	3.9m
Camera Pose Voting [43]	288.57	798	1.69m
Camera Pose Voting + RANSAC [43]	288.57	794	**0.56m**
City-Scale Localization [40]	288.57	798	**0.56m**
Active Search [32]	953	795	1.4m

Table 3. Accuracy for the Dubrovnik [38] dataset. Our method achieves a comparable accuracy and registration rate at a significantly lower memory consumption.

a compression rate of 2.04% for Aachen, [9] selects 40,377 full 3D points while we select 30,282 full 3D points plus 267,517 compressed points for the same memory budget. We experimented with different rates of compressed vs full 3D points, and found that a 1:3 ratio on average performed best over all datasets, although other splits might work better on individual datasets.

As can be seen from Fig. 2, our method vastly outperforms the state-of-the-art method by [9]. This is due to our two contributions, namely using an additional large set of compressed points and a more even distribution of the selected points in image space, as these are the main differences between our method and [9]. Notice that all compared methods achieve consistently better performances for Dubrovnik than for Aachen. This is due to the different acquisition modes of the two datasets. For Dubrovnik, the database images come from an internet photo-collection (Flickr) and thus tend to cluster around touristic areas. Thus, good camera coverage can be achieved with fewer points. Query images for this dataset were obtained by randomly removing images from a larger 3D reconstruction. They thus follow a similar distribution as the database images and can be localized with relative ease. For Aachen, the database images were taken more regularly to cover the area more completely, presenting less overlap between cameras. Thus, more points might be needed to properly cover all cameras. The query images also were taken separately and do not follow the same distribution as the database images, resulting in a harder localization scenario.

5.4. Localization Accuracy

Finally, we want to focus on the impact our scene compression method has on the accuracy of the resulting camera poses. To this end, we compare to state-of-the-art feature-based methods [32, 40, 43], recent deep-learning-based methods [5, 17–19, 42], and an image retrieval approach using compact image-level descriptors [41]. Results are shown for the Dubrovnik [38] dataset in Tab. 3 and the Cambridge Landmarks datasets [19] in Tab. 4. The methods in [22, 32, 40, 43] are similar to ours in that they also make use of SIFT features to localize a given query image to a sparse 3D scene. For these approaches, we are able to accurately compute the memory consumption for representing

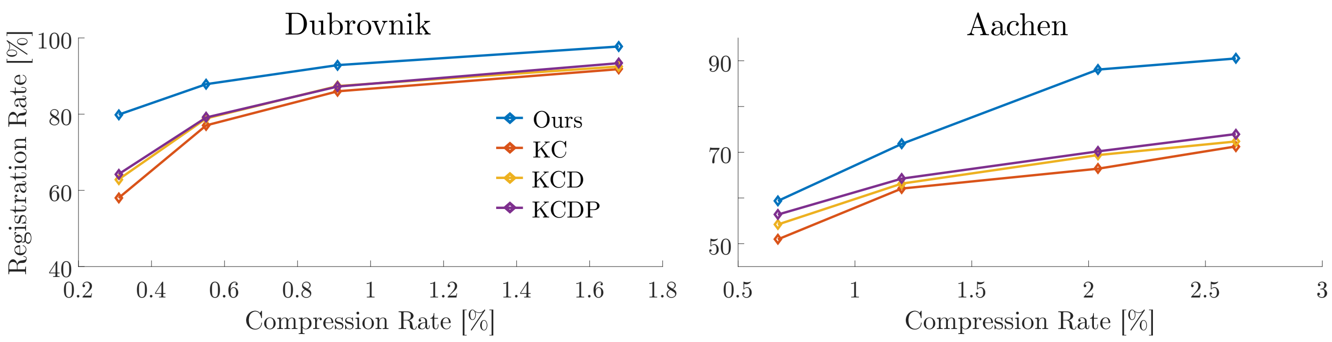

Figure 2. Comparison of the registration rates (percentage of localized query images) for different compression methods at different compression rates. Our method consistently outperforms the current state of the art approaches from [9] for all compression rates.

Method	King's College [19]			Old Hospital [19]			Shop Facade [19]			St Mary's Chruch [19]		
	MB used	# reg. images	Median errors [m,°]	MB used	# reg. images	Median errors [m,°]	MB used	# reg. images	Median errors [m,°]	MB used	# reg. images	Median errors [m,°]
Ours (@ 1.5%)	**1.01**	343	0.81, 0.59	0.62	178	0.75, 1.01	0.16	103	0.19, 0.54	1.34	530	0.50, 0.49
DenseVLAD [41]	10.06	343	2.80, 5.72	13.98	182	4.01, 7.13	3.61	103	1.11, 7.61	23.23	530	2.31, 8.00
PoseNet [19]	50	343	1.92, 5.4	50	182	2.31, 5.38	50	103	1.46, 8.08	50	530	2.65, 8.48
Bayes PoseNet [17]	50	343	1.74, 4.06	50	182	2.57, 5.14	50	103	1.25, 7.54	50	530	2.11, 8.38
LSTM PoseNet [42]	$\sim$50	343	0.99, 3.65	$\sim$50	182	1.51, 4.29	$\sim$50	103	1.18, 7.44	$\sim$50	530	1.52, 6.68
σ^2 PoseNet [18]	$\sim$50	343	0.99, 1.06	$\sim$50	182	2.17, 2.94	$\sim$50	103	1.05, 3.97	$\sim$50	530	1.49, 3.43
geom. PoseNet [18]	$\sim$50	343	0.88, 1.04	$\sim$50	182	3.20, 3.29	$\sim$50	103	0.88, 3.78	$\sim$50	530	1.57, 3.32
DSAC++ [5]	207	-	**0.18, 0.3**	207	-	**0.20, 0.3**	207	-	**0.06, 0.3**	207	-	**0.13, 0.4**
Active Search [32]	275	343	0.57, 0.7	140	180	0.52, 1.12	38.7	103	0.12, 0.41	359	530	0.22, 0.62

Table 4. Comparison with several state-of-the-art methods on the Cambridge Landmarks datasets [19].

the scene, but ignore potential overhead due to search structures. For the deep learning methods, an approximate size of the network is provided.

It should be noted that the deep learning methods do not intrinsically provide a way to judge if a particular queried image was successfully localized (whereas in geometric methods one can rely on, *e.g*., inlier statistics). Thus, registration rates cannot be fairly compared to our approach. The registration rates for DSAC++ are not reported in [5].

As can be seen in Tab. 3 and Tab. 4, our method clearly presents the best trade-off between memory consumption and performance. Compared to the retrieval baseline [41] and the pose regression techniques [17–19, 42], our approach achieves both a lower memory consumption and higher pose accuracy. Compared to Active Search [32], our approach provides a comparable pose accuracy while achieving a reduction in memory consumption by more than order of magnitude. DSAC++ [5] estimates poses that are significantly more accurate than those computed by our approach. However, DSAC++ is not able to provide a compact representation for small scenes. Note that the high pose accuracy obtained by [43] and [40] on the Dubrovnik dataset comes at a high run-time costs of more than 2 seconds per image on average.

Overall, our results show that our method represent the best of both worlds: it has a very small memory footprint (up to orders of magnitude lower than in the non-compressed setting) while achieving a performance close to the state-of-the-art feature-based methods for efficient localization.

6. Conclusion

In this paper, we have proposed a hybrid 3D scene compression scheme and a RANSAC variant that jointly manage to efficiently and accurately localize a given query image, all while using a small fraction ($\sim 1.5\%$) of the original memory requirements. Differently to previously proposed methods, we compress the 3D scene by producing two disjoint sets of 3D points: 1) a set of points with their full visual descriptors that can be used to produce one-to-one matches given a query descriptor and 2) a second set of points for which we only store a compressed descriptor. This second set of points is used to produce one-to-many matches, *i.e*., multi-matches. The hybrid output of our compression scheme is carefully selected to ensure a good scene coverage and visual uniqueness of the selected 3D points. To properly handle the special two-fold output of our compression, we design a RANSAC variant to handle multi-matches such that they are only used during the model verification step, thus increasing number of inliers without negatively affecting the number of iterations. In addition, we propose a highly efficient co-visibility-based strategy to guide sampling within RANSAC.

We have validated our approach using several real-world datasets. Our results show that our method achieves localization rates and a pose accuracy comparable to state-of-the-art feature-based and CNN-based methods at a significantly lower memory footprint.

Acknowledgements. We thank Google's Visual Positioning System for their support.

References

[1] Sameer Agarwal, Noah Snavely, Ian Simon, Steven M. Seitz, and Richard Szeliski. Building Rome in a Day. In *ICCV*, 2009. 1

[2] Relja Arandjelović, Petr Gronat, Akihiko Torii, Tomas Pajdla, and Josef Sivic. NetVLAD: CNN architecture for weakly supervised place recognition. In *CVPR*, 2016. 2

[3] Relja Arandjelovic and Andrew Zisserman. All About VLAD. In *CVPR*, 2013. 2

[4] Eric Brachmann, Alexander Krull, Sebastian Nowozin, Jamie Shotton, Frank Michel, Stefan Gumhold, and Carsten Rother. DSAC - Differentiable RANSAC for Camera Localization. In *CVPR*, 2017. 2

[5] Eric Brachmann and Carsten Rother. Learning Less is More-6D Camera Localization via 3D Surface Regression. In *CVPR*, 2018. 2, 3, 7, 8

[6] Martin Bujnak, Zuzana Kukelova, and Tomás Pajdla. A general solution to the P4P problem for camera with unknown focal length. In *CVPR*, 2008. 5, 6

[7] Federico Camposeco, Torsten Sattler, Andrea Cohen, Andreas Geiger, and Marc Pollefeys. Toroidal Constraints for Two-Point Localization Under High Outlier Ratios. In *CVPR*, 2017. 4

[8] Song Cao and Noah Snavely. Graph-Based Discriminative Learning for Location Recognition. In *CVPR*, 2013. 2

[9] Song Cao and Noah Snavely. Minimal scene descriptions from structure from motion models. In *CVPR*, 2014. 1, 2, 3, 4, 5, 7, 8

[10] Ondrej Chum and Jiri Matas. Matching with prosac-progressive sample consensus. In *CVPR*, 2005. 5

[11] Andrew J. Davison, Ian D. Reid, Nicholas D. Molton, and Olivier Stasse. Monoslam: Real-time single camera slam. *PAMI*, 29:2007, 2007. 1

[12] Marcin Dymczyk, Simon Lynen, Titus Cieslewski, Michael Bosse, Roland Siegwart, and Paul Furgale. The gist of maps - summarizing experience for lifelong localization. In *ICRA*, 2015. 2, 3

[13] Martin A. Fischler and Robert C. Bolles. Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography. *ACM*, 24(6):381–395, June 1981. 1, 3, 5

[14] Christian Häne, Lionel Heng, Gim Hee Lee, Friedrich Fraundorfer, Paul Furgale, Torsten Sattler, and Marc Pollefeys. 3D Visual Perception for Self-Driving Cars using a Multi-Camera System: Calibration, Mapping, Localization, and Obstacle Detection. *IMAVIS*, 68:14 – 27, 2017. 1

[15] Jared Heinly, Johannes L Schönberger, Enrique Dunn, and Jan-Michael Frahm. Reconstructing the world* in six days *(as captured by the yahoo 100 million image dataset). In *CVPR*, 2015. 1

[16] Arnold Irschara, Christopher Zach, Jan-Michael Frahm, and Horst Bischof. From structure-from-motion point clouds to fast location recognition. In *CVPR*, 2009. 2, 4, 5

[17] Alex Kendall and Roberto Cipolla. Modelling Uncertainty in Deep Learning for Camera Relocalization. In *ICRA*, 2016. 2, 7, 8

[18] Alex Kendall and Roberto Cipolla. Geometric loss functions for camera pose regression with deep learning. In *CVPR*, 2017. 2, 7, 8

[19] Alex Kendall, Matthew Grimes, and Roberto Cipolla. Posenet: A Convolutional Network for Real-time 6-DoF Camera Relocalization. In *ICCV*, 2015. 2, 7, 8

[20] Laurent Kneip, Davide Scaramuzza, and Roland Siegwart. A novel parametrization of the perspective-three-point problem for a direct computation of absolute camera position and orientation. In *CVPR*, 2011. 5

[21] Yunpeng Li, Noah Snavely, Daniel Huttenlocher, and Pascal Fua. Worldwide pose estimation using 3D point clouds. In *ECCV*, 2012. 2, 6, 7

[22] Yunpeng Li, Noah Snavely, and Daniel P Huttenlocher. Location recognition using prioritized feature matching. In *ECCV*, 2010. 1, 2, 3, 4, 5, 6, 7

[23] Ching Lih Lim, Alistair Moffat, and Anthony Wirth. Lazy and Eager Approaches for the Set Cover Problem. In *Australasian Computer Science Conference*, 2014. 4

[24] David G Lowe. Distinctive image features from scale-invariant keypoints. *IJCV*, 60(2):91–110, 2004. 5, 6

[25] Simon Lynen, Torsten Sattler, Michael Bosse, Joel Hesch, Marc Pollefeys, and Roland Siegwart. Get out of my lab: Large-scale, real-time visual-inertial localization. In *RSS*, 2015. 1, 2, 3

[26] Daniela Massiceti, Alexander Krull, Eric Brachmann, Carsten Rother, and Philip H.S. Torr. Random Forests versus Neural Networks - What's Best for Camera Relocalization? In *ICRA*, 2017. 2

[27] Paul McIlroy, Edward Rosten, Simon Taylor, and Tom Drummond. Deterministic Sample Consensus with Multiple Match Hypotheses. In *BMVC*, 2010. 3

[28] Sven Middelberg, Torsten Sattler, Ole Untzelmann, and Leif Kobbelt. Scalable 6-DOF Localization on Mobile Devices. In *ECCV*, 2014. 1

[29] Tayyab Naseer and Wolfram Burgard. Deep regression for monocular camera-based 6-DoF global localization in outdoor environments. In *IROS*, 2017. 2

[30] James Philbin, Ondrej Chum, Michael Isard, Josef Sivic, and Andrew Zisserman. Object Retrieval with Large Vocabularies and Fast Spatial Matching. In *CVPR*. IEEE, 2007. 2

[31] Torsten Sattler, Michal Havlena, Filip Radenovic, Konrad Schindler, and Marc Pollefeys. Hyperpoints and fine vocabularies for large-scale location recognition. In *ICCV*, 2015. 2, 3

[32] Torsten Sattler, Bastian Leibe, and Leif Kobbelt. Efficient & Effective Prioritized Matching for Large-Scale Image-Based Localization. *PAMI*, 39(9):1744–1756, 2017. 1, 2, 3, 4, 6, 7, 8

[33] Torsten Sattler, Will Maddern, Carl Toft, Akihiko Torii, Lars Hammarstrand, Erik Stenborg, Daniel Safari, Masatoshi Okutomi, Marc Pollefeys, Josef Sivic, Fredrik Kahl, and Tomas Pajdla. Benchmarking 6DOF Outdoor Visual Localization in Changing Conditions. In *CVPR*, 2018. 7

[34] Torsten Sattler, Akihiko Torii, Josef Sivic, Marc Pollefeys, Hajime Taira, Masatoshi Okutomi, and Tomas Pajdla. Are Large-Scale 3D Models Really Necessary for Accurate Visual Localization? In *CVPR*, 2017. 2, 3

[35] Torsten Sattler, Tobias Weyand, Bastian Leibe, and Leif Kobbelt. Image Retrieval for Image-Based Localization Revisited. In *BMVC*, 2012. 7

[36] Johannes Lutz Schönberger and Jan-Michael Frahm. Structure-from-motion revisited. In *CVPR*, 2016. 1

[37] Jamie Shotton, Ben Glocker, Christopher Zach, Shahram Izadi, Antonio Criminisi, and Andrew Fitzgibbon. Scene Coordinate Regression Forests for Camera Relocalization in RGB-D Images. In *CVPR*, 2013. 2

[38] Noah Snavely, Steven M. Seitz, and Richard Szeliski. Modeling the world from internet photo collections. *IJCV*, 80(2):189–210, 2008. 7

[39] Hyun Soo Park, Yu Wang, Eriko Nurvitadhi, James C Hoe, Yaser Sheikh, and Mei Chen. 3d Point Cloud Reduction Using Mixed-Integer Quadratic Programming. In *CVPR Workshops*, 2013. 3

[40] Linus Svärm, Olof Enqvist, Fredrik Kahl, and Magnus Oskarsson. City-Scale Localization for Cameras with Known Vertical Direction. *PAMI*, 39(7):1455–1461, 2017. 7, 8

[41] Akihiko Torii, Relja Arandjelović, Josef Sivic, Masatoshi Okutomi, and Tomas Pajdla. 24/7 Place Recognition by View Synthesis. In *CVPR*, 2015. 2, 7, 8

[42] Florian Walch, Caner Hazirbas, Laura Leal-Taixe, Torsten Sattler, Sebastian Hilsenbeck, and Daniel Cremers. Image-based localization using lstms for structured feature correlation. In *ICCV*, 2017. 2, 7, 8

[43] Bernhard Zeisl, Torsten Sattler, and Marc Pollefeys. Camera pose voting for large-scale image-based localization. In *ICCV*, 2015. 7, 8

[44] Wei Zhang and Jana Kosecka. Image based localization in urban environments. In *3DPVT*, 2006. 2