Optimal Reduction of Large Image Databases for Location Recognition

Michal Havlena Wilfried Hartmann Konrad Schindler
Institute of Geodesy and Photogrammetry, ETH Zürich
Wolfgang-Pauli-Strasse 15, 8093 Zürich, Switzerland
{michal.havlena,wilfried.hartmann,schindler}@geod.baug.ethz.ch

Abstract

For some computer vision tasks, such as location recognition on mobile devices or Structure from Motion (SfM) computation from Internet photo collections, one wants to reduce a large set of images to a compact, representative subset, sometimes called "keyframes" or "skeletal set". We examine the problem of selecting a minimum set of such keyframes from the point of view of discrete optimization, as the search for a minimum connected dominating set (CDS) of the graph of pairwise connections between the database images. Even the simple minimum dominating set (DS) problem is known to be NP-hard, and the constraint that the dominating set should be connected makes it even harder. We show how the minimum DS can nevertheless be solved to global optimality efficiently in practice, by formulating it as an integer linear program (ILP). Furthermore, we show how to upgrade the solution to a connected dominating set with a second ILP if necessary, although the complete method is no longer globally optimal. We also compare the proposed method to a previous greedy heuristic. Experiments with several image sets show that the greedy solution already performs remarkably well, and that the optimal solution achieves roughly 5% smaller keyframe sets which perform equally well in location recognition and SfM tasks.

1. Introduction

Location recognition from an image taken by the user is a useful application of computer vision, even more so now that practically all mobile phones and tablets are equipped with cameras. The state-of-the-art methods to build such a service [10, 4] rely on the "big data" paradigm—large precomputed image databases which reside on computer clusters, to which user images must be submitted via Internet or cellular networks to query for the location. With increasing computational power and storage capacity of the mobile devices, it becomes possible to run such an application even locally, without network access.

For the described mobile application of location recognition, the compactness of the pre-computed database is crucial, but the need for a compact, yet meaningful subset of big image data is not restricted to this particular use case. Mobile robotic SLAM [6] and short- or wide-baseline sequential visual odometry methods [22, 29] often rely on a set of keyframes to robustify the computation and/or to overcome the drift of the estimated camera poses over time, by performing loop-closing. Compactness of the keyframe set is important for the efficiency of loop-closing, which must share limited processing power with the actual per-frame SLAM computation. Note that the image selection procedure is even more complicated in the SLAM case because of the on-line nature of the computation.

The contribution of this paper is twofold. First, we explore optimal methods for reducing a large, redundant input image set by selecting the most relevant representatives via solving the same graph problem as [14]. Secondly, we compare the selections made by the proposed near-optimal selection algorithm with the output of the existing suboptimal greedy algorithm used by [14] and show that the proposed method gives more compact image subsets while the performance of the location recognition task remains unchanged. We also show that the difference between the sizes of the subsets obtained using suboptimal and optimal methods is rather small: while the proposed approach is arguably more principled, our results vindicate the widely used greedy heuristic, but have a slight edge for applications with strict storage constraints.

The bulk of visual location recognition techniques relies on image retrieval based on visual words [27]. Further advances using either a more compact data representation [5] or achieving higher efficiency thanks to vocabulary trees [23] allow for location recognition from city-scale databases [26]. Recently, methods have appeared which offer precise positioning of a given query image [15, 25]. These methods provide the camera pose w.r.t. a precomputed 3D point cloud, instead of just an approximate localization. Once the point cloud is geo-referenced, a precise location on the Earth can be obtained [19, 12].

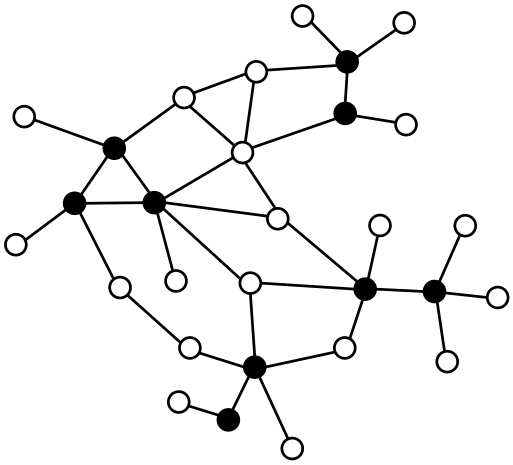

Figure 1. Unconnected dominating set denoted by black vertices.

We use location recognition as a benchmark for reducing big image databases to representative subsets. Our aim here is to decrease the size of the keyframe set by revisiting the problem of selecting relevant representatives from a large image collection. To our best knowledge, no systematic attempts have been made so far to select such image subsets optimally, by formally minimizing the size of the subset that has the required properties. Note however, in other contexts the formulation of the minimum dominating set problem as an integer linear program has been known for several years [16].

Most techniques for reducing an image set rely on the same idea, namely to first compute pairwise distances within the database, and then prune it such that for every image there is a sufficiently near representative in the keyframe set (or equivalently, a representative for each cluster of images). The differences lie mostly in the proxy used to estimate the distance. One possibility is to employ global image descriptors, *e.g*. GIST [24]. Although this approach was successfully used to speed-up Structure from Motion (SfM) computation [17, 8], we believe that methods based on visual words are more suitable for the location recognition task. On the other extreme, one could estimate similarity through the number of keypoint matches or even through the relative pose, however such approaches in most cases defy the purpose, as matching and pose estimation are computationally too expensive.

We use a technique based on visual words, similar to the one of [14]. Note however, our approach is generic and applies to any distance/similarity matrix. We show that thanks to modern integer linear program (ILP) solvers, it is possible to solve this graph problem optimally even for rather large image sets, and that there is a noticeable gap between the sizes of the optimal and greedy solutions. It is worth mentioning that also the widely used large-scale SfM pipeline based on skeletal image sets [28, 1] greedily solves the same graph problem, so our results apply to city-scale SfM, too.

2. Method

Let us assume for the moment that a connected graph is available whose vertices are the input images, and in which an edge exists between two vertices if and only if the two corresponding images share a significant part of the viewfield, *i.e*. are co-located. Then, to make a compact selection covering all views present in the image set one needs to select a minimum subset of vertices, in such a way that *(i)* all the vertices in the graph are either in the subset or connected to it by an edge (meaning that every database image is co-located with at least one keyframe), and *(ii)* the subgraph induced by the selected subset is connected. In graph theory a subset that fulfills *(i)* is known as the *minimum dominating set* (DS) of the graph, see Figure 1, while *(ii)* is called the *minimum connected dominating set* (CDS). Searching a CDS avoids database fragmentation, but makes the problem considerably harder. In some applications, such as finding a skeletal set for SfM, connectedness is mandatory, while in others it might be dropped or only encouraged rather than enforced.

Following [11], the minimum CDS problem is defined as follows. Given a graph $G = (\mathcal{V}, \mathcal{E})$, find a minimum size subset of vertices $\mathcal{S} \subset \mathcal{V}$, such that the subgraph induced by $\mathcal{S}$ is connected and $\mathcal{S}$ forms a dominating set in G. In a graph with a dominating set $\mathcal{D} \subset \mathcal{V}$, each vertex $V \in \mathcal{V}$ is either in the dominating set or adjacent to some vertex in the dominating set, $V \in \mathcal{D} \vee \exists V' \in \mathcal{D} : (V, V') \in \mathcal{E}$. We say that V is covered by V' in the latter case. The problem of finding the minimum CDS is known to be *NP*-hard [9].

2.1. Minimum CDS formulated as ILP

By definition, to find the optimal solution to an *NP*-hard problem one has to evaluate an exponential number of candidates in the worst case. Nevertheless, a large class of *NP*-hard problems can be formulated as integer linear programs (ILPs); and for many instances of such ILPs, modern solvers can find the optimum quickly. The general ILP formulation reads:

$$\begin{aligned} \text{minimize} \quad & \mathbf{c}^T\mathbf{x} \\ \text{subject to} \quad & \mathtt{A}\mathbf{x} \geq \mathbf{b} \\ & l_j \leq x_j \leq u_j,\ x_j \in \mathbb{Z} \end{aligned} \tag{1}$$

The high efficiency of ILP solvers is achieved thanks to the following strategy: *(i)* obtain a lower bound of the objective function in low polynomial time by solving the relaxed LP problem, where the integer constraint is dropped; and *(ii)* use the lower bound in branch-and-bound type methods to quickly rule out large portions of the search space.

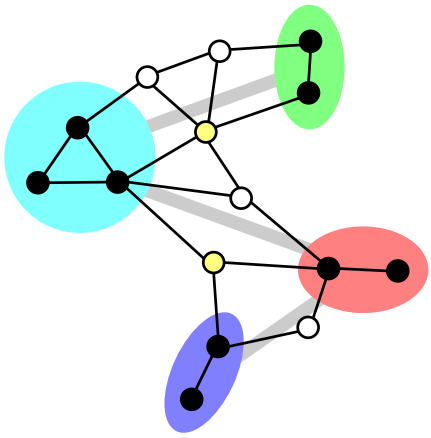

Figure 2. Four components of the subgraph of G induced by $\mathcal{D}$ denoted by color ellipses are connected by adding two yellow vertices into $\mathcal{S}$. These two vertices form the solution of Program 3 for a 3×5 matrix $\mathtt{D}$ (the union of the sets $\mathcal{C}$ and $\mathcal{C}'$ for the three gray edges of the corresponding maximum spanning tree of the constructed weighted graph contains four 1-vertex and one 2-vertex bridges), see text.

Although this can in the worst case still lead to an exponential number of candidates, the actual convergence times are quite short for our ILPs as will be shown in Section 3.4.

We believe that, unfortunately, it is not possible to formulate the search for the minimum CDS as an ILP, because of the comprehensiveness of the connectivity constraint. On the contrary, the ILP formulation of finding a (possibly disconnected) minimum DS is straightforward [16]. We thus resort to the same strategy also used by the greedy formulation (see Section 2.2), namely first find the minimum dominating set $\mathcal{D}$. If the subgraph induced by $\mathcal{D}$ is connected, $\mathcal{D}$ is at the same time the minimum connected dominating set of G. Otherwise, we need to include additional vertices in $\mathcal{S} \supset \mathcal{D}$. While one can also select these "connecting" vertices in a one-shot optimization, the total CDS is no longer guaranteed to be the globally most compact one.

To prevent excessive fragmentation of the found minimum dominating set $\mathcal{D}$, and thereby increase the chance that the subgraph induced by $\mathcal{D}$ is already connected, we require even the selected vertices to be covered, so the minimum possible size of a component is 2 and not 1. The connectivity of $\mathcal{D}$ can be further improved by requiring each vertex to be covered by more than one vertex. We call the number of covering vertices of a keyframe set its *cover number* (CN).[1] This changes the minimum component size to 3 or even more, and often directly yields a minimum CDS, at the cost of obtaining larger keyframe sets, see Section 3.

[1] Note the relation to the k-nearest neighbor strategy: a cover number of Q guarantees that at least Q useful neighbors are available for localization, respectively pose estimation.

To formulate the search for the minimum dominating set as an ILP, we construct an $n \times n$ matrix $\mathtt{A}$ where n is the number of vertices of G and $\mathtt{A}_{ij} = 1 \iff (V_i, V_j) \in \mathcal{E}$ with the remaining elements of $\mathtt{A}$ being zero. By zeroing the diagonal of $\mathtt{A}$, we require the selected vertices to be covered also. Our program then reads:

$$\begin{array}{ll} \text{minimize} & \mathbf{1}^T\mathbf{x} \\ \text{subject to} & \mathtt{A}\mathbf{x} \geq \mathbf{b} \\ & 0 \leq x_j \leq 1,\ x_j \in \mathbb{Z} \end{array} \tag{2}$$

The desired cover number can be set through parameter vector $\mathbf{b}$. The solution of the program is a Boolean vector $\mathbf{x}$ of length n, indicating the membership of vertices in the set $\mathcal{D}$. The minimized objective value is the number of vertices in the DS.

If the subgraph induced by $\mathcal{D}$ consists of multiple, say m (disconnected) components, then for every such component there exists another one to which it can be connected by adding at most two additional vertices, thanks to the properties of the dominating set [11]. We examine all pairs of components to find vertices which would form such 1-vertex or 2-vertex bridges, and then try to find the smallest subset of vertices that need to be added to $\mathcal{D}$ to obtain a connected graph.

First, to simplify the task, the $(m-1)$ pairs of components which are going to be connected are pre-selected. For each pair of components i and j, the set of feasible 1-vertex bridges $\mathcal{C}_{ij}$ and the set of feasible 2-vertex bridges $\mathcal{C}'_{ij}$ are found. Second, a weighted graph is constructed whose vertices are the individual components, and whose edges are the potential connections, weighted according to numbers of feasible bridges: the weight of a given edge is either the size of $\mathcal{C}'_{ij}$ (if $\mathcal{C}_{ij}$ is empty) or the size of $\mathcal{C}_{ij}$ plus the size of the largest $\mathcal{C}'$ in the entire graph otherwise. By selecting the $(m-1)$ pairs forming the maximum spanning tree of this graph, the pairs of components which can be connected using a 1-vertex bridge are preferred. The secondary criteria then is the number of feasible 1-vertex (or 2-vertex) bridges, because this increases the chance that some of these bridges could potentially connect other component pairs, too. Then, a second ILP is constructed:

$$\begin{array}{ll} \text{minimize} & \mathbf{c}^T\mathbf{y} \\ \text{subject to} & \mathtt{D}\mathbf{y} \geq \mathbf{1} \\ & 0 \leq y_j \leq 1,\ y_j \in \mathbb{Z} \end{array} \tag{3}$$

where $\mathtt{D}$ is a $(m-1) \times k$ matrix, k is the size of the union of sets $\mathcal{C}$ and $\mathcal{C}'$ of the pre-selected component pairs, and $\mathbf{c}$ is a vector of length k which is 1 for 1-vertex bridges and 2 for 2-vertex bridges. The minimized objective function then represents the number of added vertices. The entries of $\mathtt{D}$ are $\mathtt{D}_{ij} = 1$ if the i-th component pair would be connected by the j-th bridge, and zero otherwise. The constraints ensure that all $(m-1)$ components pairs will be connected,

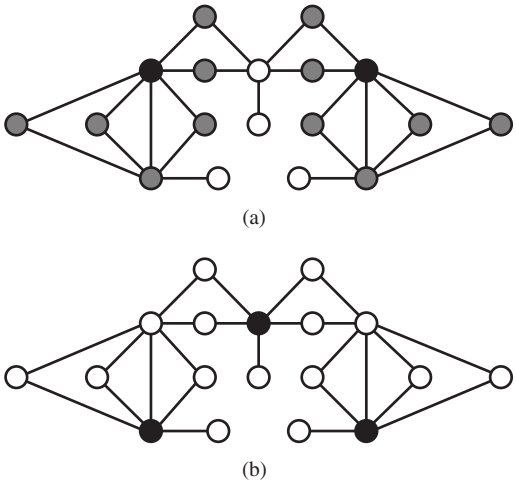

Figure 3. (a) Illustration of the computation of Algorithm 1 (Step II.). Two black vertices have been added to $\mathcal{D}$ so far, the gray vertices are already covered. Note that three more vertices need to be added to construct the dominating set whereas the optimal solution (b) consist of three vertices in total.

resulting in a subgraph of G with a single component, *i.e*. a CDS, see Figure 2.

2.2. Greedy Algorithm

A (suboptimal) greedy solution to the minimum CDS problem is given in [11], see Algorithm 1. Being greedy, the algorithm inherently has polynomial time complexity. Moreover, its approximation ratio is in the worst case $\ln \triangle + 3$, with $\triangle$ the maximum vertex degree in the graph.

The greedy algorithm selects one vertex at a time, picking the one that covers the largest number c_V of uncovered vertices, see Figure 3(a). Note that this strategy is suboptimal, Figure 3(b). If the desired cover number is greater than one, the algorithm can be extended by introducing fractional coverage. Each not fully covered vertex contributes as many times to the value of c_V as it needs to be further covered. Thereby, the selection prefers vertices whose neighbors received less coverage from the previously selected vertices. In our experience this works better than counting all not yet fully covered vertices equally, as proposed originally.

Once the greedy selection has found a DS, it is connected to a CDS with another greedy search procedure. For all component pairs, one checks whether a connection by a 1-vertex bridge or 2-vertex bridge is possible. Then one iteratively selects the bridge which connects the highest number of components. For a graph consisting of m components, at most $(m-1)$ iterations are needed to obtain the final $\mathcal{D}$. This can be rather slow for highly fragmented graphs because graph components need to be updated after each iteration.

Algorithm 1 Approximate min CDS computation [11]

Input	$G = (\mathcal{V}, \mathcal{E})$	Unweighted undirected graph.
Output	$\mathcal{S}$	Vertices of the min CDS of G.

I. Label all vertices $V \in \mathcal{V}$ white.

II. Set $\mathcal{D} := \{\}$ and repeat until no white vertices are left:
  1: For all black vertices $V \in \mathcal{V}$ set $c_V := 0$.
  2: For all gray and white vertices V set $c_V :=$ number of white neighbors of V.
  3: Set $V^* := \arg\max_V c_V$.
  4: Label V^* black and add it into $\mathcal{D}$.
  5: Label all neighbors of V^* gray.

III. Set $\mathcal{S} := \mathcal{D}$ and connect components of the subgraph induced by $\mathcal{D}$, by adding at most 2 vertices per component into $\mathcal{S}$.

IV. Return $\mathcal{S}$.

2.3. Constructing the Graph

Until now, we have assumed that the graph G representing the co-located images is available. In fact, constructing such a graph properly, *i.e*. by matching all image pairs and verifying pairwise geometries, is rather costly. We use a standard image indexing technique instead to obtain a proxy more efficiently [27]. Images are resampled to 3Mpix resolution, and a visual vocabulary of 200,000 visual words trained on images of city landmarks is used to quantize SURF [2] features detected in the database images. The entries of the similarity matrix are dot products of the images' *tf-idf* vectors, and the edges of graph G can be obtained by thresholding the similarity matrix.

Special care must be taken to ensure there is a solution: if the graph G is not connected, no CDS exists. Moreover, already the DS is infeasible if G has small isolated components whose vertex count is lower than the desired cover number. To bypass this issue, we pre-process the graph and recursively remove uncoverable vertices until the problem is solvable. Note, this step can already significantly decimate the database size if a high similarity threshold is used.

3. Experiments

Three landmark image sets, each consisting of roughly 4,000 images, were downloaded from Flickr [32] as results for queries "di Trevi", "duomo milano", and "old town square prague". The variability of the image sets increases as DITREVI represents a single wall, DUOMO represents a single building, while OLDTOWN represents a larger area with multiple buildings. Only images which were available in at least 0.75Mpix resolution were used.

Figure 4. Sample images from the three datasets downloaded from Flickr [32]: (a) DITREVI, (b) DUOMO, (c) OLDTOWN.

To evaluate location recognition using the selected subsets of images, 300 images from each image set were randomly set aside as query images and the rest of the images form the datasets of sizes 3,110, 3,104, and 3,255 images respectively, see Figure 4.

Two application scenarios were explored with the held-out query images. First, location recognition based on image retrieval and subsequent geometric verification, using the selected keyframe sets. And second, pose estimation of the query images w.r.t. 3D models reconstructed from the keyframe sets.

3.1. CDS Computation

Connected dominating sets were computed for the three datasets independently, using both the ILP solver and the greedy algorithm. The corresponding graphs G were obtained using the method described in Section 2.3 for three different thresholds of the image similarity, namely 0.075, 0.1, and 0.15. Cover numbers (parameter vector $\mathbf{b}$ in ILP) of 1, 2, and 3 were tested. The sizes of the returned minimum DS and CDS of G are displayed in Figure 5. It can be seen that more images are required for the most complex scene OLDTOWN than for the other two datasets.

As both methods first search for the dominating set, it is also interesting to see how many vertices had to be added to connect the DS into a CDS, "Con" in Figure 5. We observe that, as expected, the fragmentation of the dominating set

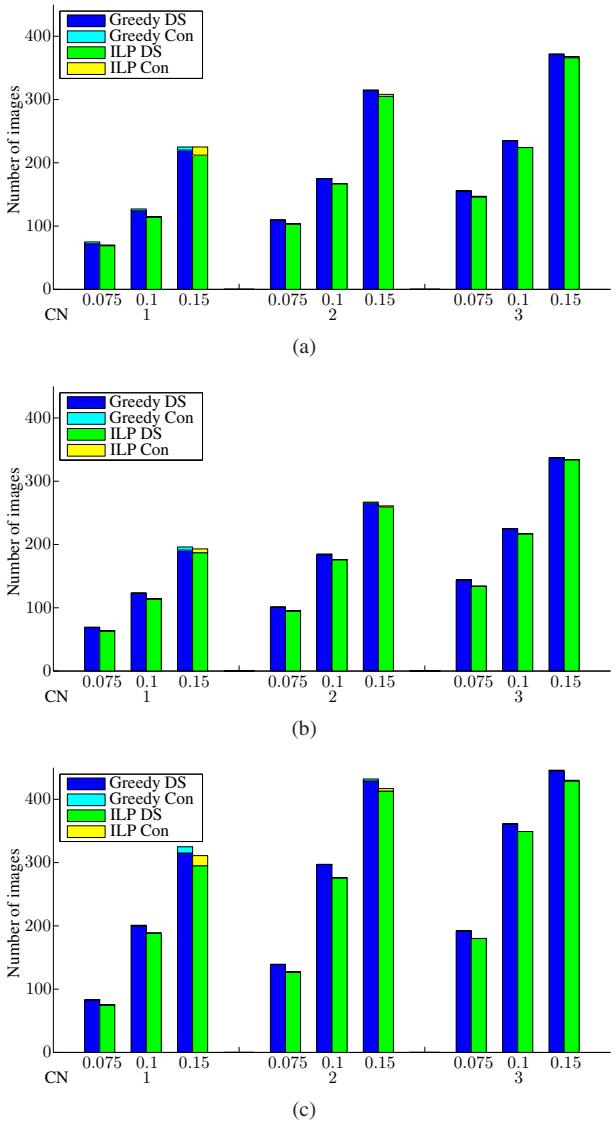

Figure 5. Sizes of the connected dominating sets obtained by the greedy algorithm and by solving the ILP, with different thresholds for image similarity and different cover numbers. Datasets (a) DITREVI, (b) DUOMO, and (c) OLDTOWN.

is smaller when the similarity threshold is lower (and hence the graph is denser), and also when a higher cover number is chosen. In some cases the optimal DS was already connected, in which case it is also the globally optimal CDS. In the remaining cases, and always for the greedy solutions, no optimality guarantee can be given. We also observe that the greedy DS often needs fewer additional vertices to turn it into a CDS; the DS returned by ILP are $\approx 5\%$ smaller than those of the greedy method. That difference slightly decreases for the respective CDS solutions. Apparently the redundancy in the greedy solution means that it is closer to a CDS, and fewer vertices have to be added.

Image Retrieval - correctly localized images (of 300)						
DITREVI	Greedy			ILP		
Thr / CN	1	2	3	1	2	3
0.075	140	151	160	137	144	156
0.100	153	165	160	156	156	163
0.150	171	169	169	169	174	172
DUOMO	Greedy			ILP		
Thr / CN	1	2	3	1	2	3
0.075	35	62	57	25	59	69
0.100	68	81	83	67	78	79
0.150	75	86	91	78	87	90
OLDTOWN	Greedy			ILP		
Thr / CN	1	2	3	1	2	3
0.075	82	102	113	81	94	107
0.100	119	111	123	112	121	123
0.150	121	123	131	124	124	135

Table 1. Evaluation of location recognition using image retrieval with geometric verification, with different thresholds for image similarity and different cover numbers.

Pose Estimation - correctly localized images (of 300)						
DITREVI	Greedy			ILP		
Thr / CN	1	2	3	1	2	3
0.075	188	206	199	176	210	218
0.100	213	226	230	221	224	230
0.150	222	229	230	235	230	234
DUOMO	Greedy			ILP		
Thr / CN	1	2	3	1	2	3
0.075	66	73	80	62	63	78
0.100	74	90	84	75	86	82
0.150	85	97	93	94	94	92
OLDTOWN	Greedy			ILP		
Thr / CN	1	2	3	1	2	3
0.075	65	84	85	63	64	76
0.100	85	122	151	85	n/a	148
0.150	134	154	152	133	153	134

Table 2. Evaluation of location recognition using pose estimation w.r.t. the 3D point cloud of the largest partial model, with different thresholds for image similarity and different cover numbers.

3.2. Image Retrieval

Following the same procedure as for the dataset images, we extract SURF [2] features from the query images (resampled to 3Mpix resolution), quantize them with a pretrained visual vocabulary, and represent images by *tf-idf* vectors [27]. Our database is formed by the union of the minimum CDS of all three image collections (DITREVI, DUOMO, and OLDTOWN). Similarity between each query image and all database images is computed as the scalar product of their *tf-idf* scores. The five database images having the highest score are passed to geometric verification with a homography [13]. The original SURF descriptors are matched with approximate NN-search [30] and the homography is then computed with RANSAC [7], with an error threshold of 4 pixels. Among those candidates for which the fit has at least 100 inliers and an inlier ratio better than 10%, the one with the highest inlier ratio is selected. The localization is considered correct if the selected candidate is from the same collection (*i.e*. the same landmark).

Results summarized in Table 1 show that, over all cover numbers and image similarity thresholds, the numbers of correctly localized images are roughly the same for the keyframe sets obtained with the greedy algorithm and those from the ILP solution, although the latter are more compact. The numbers of incorrectly localized images were around 8 for DITREVI and around 11 for DUOMO and OLDTOWN.

Regarding storage, the footprint of the database is $\approx$ 70kB per image for the zipped *tf-idf* vectors, plus $\approx$ 500kB per image to store the image itself (for efficient on-the-fly extraction of SURF features). The size of the visual vocabulary, independent of the dataset size, is 108MB.

3.3. Pose Estimation

VisualSFM [31] was used to create 3D models of the selected keyframe sets. The sizes of the largest reconstructed partial models are small for low similarity thresholds, because with such strict thresholds the image collections disintegrate into many small, unconnected fragments. We note however that with one exception (OLDTOWN, image similarity 0.1, cover number 2) all 3D models are visually correct, albeit incomplete—they capture only the most prominent part of the scene. Dataset DUOMO seems to be the most challenging one because several separate clusters of images exist, *e.g*. the frontal view, the building interior, and the view from the roof. Consequently, only the largest cluster—the frontal view—is reconstructed. The reconstruction of OLDTOWN becomes quite complete when a sufficiently high threshold for image similarity is used.

We used VisualSFM "resume" feature to test whether it is possible to connect a given query image to the constructed landmark models. The numbers of correctly localized images summarized in Table 2 show that the results are again roughly the same for the keyframe sets obtained using the greedy algorithm and those from the ILP solution. In these experiments, not a single image was wrongly localized (*i.e*. connected to the wrong landmark model). One can also see that for the easy DITREVI dataset, the results of localization using pose estimation are better than those obtained from localization using image retrieval. On the other hand, localization using image retrieval works better than via pose estimation for the most complex dataset OLDTOWN, because of the difficulties of building 3D models of complicated scenes from reduced, overly sparse image sets.

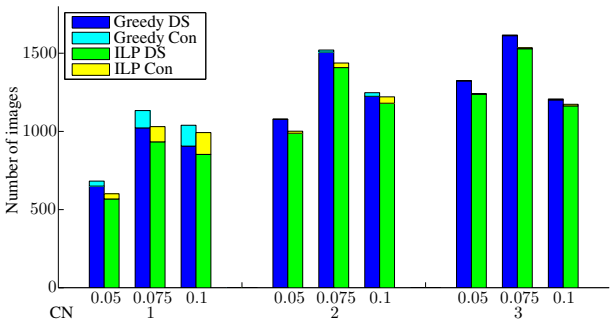

Figure 6. Sizes of the connected dominating sets obtained by the greedy algorithm and by solving the ILP, with different thresholds for image similarity and different cover numbers for dataset DUBROVNIK.

Regarding storage, the most salient SIFT features extracted by VisualSFM occupy $\approx$ 1MB per image. The size of the reconstructed 3D model is dependent on the scene type and the spatial distribution of cameras; for our datasets it was approximately 30kB per reconstructed image.

3.4. DUBROVNIK Dataset

Finally, the performance and scalability of the proposed selection method was tested using the publicly available DUBROVNIK dataset [18] which comprises 6,044 database and 800 query images, with known ground truth poses w.r.t. the 3D point cloud constructed from the database images. As only SIFT [20] descriptors are provided and not the original images, we trained a 200,000 word vocabulary from the database image descriptors. The sizes of the obtained minimum DS and CDS are shown in Figure 6.

Notice that *(i)* the graph is similar to the one of the OLDTOWN dataset but the overall portion of selected images is nearly twice as large due to the even more complex layout of DUBROVNIK; and *(ii)* lower thresholds for image similarity were used because for higher thresholds the removal of uncoverable vertices during pre-processing already eliminated the majority of the images.

Run times are reported for the minimum DS search of our multi-threaded MATLAB implementation of the greedy algorithm and MOSEK [21] ILP solver. Times were measured on a single Intel(R) Core(TM) i7-3930K workstation with Linux 64bit operating system, see Table 3.

We point out that the run times of the greedy algorithm and the ILP solver are similar but there is a much higher variance in the run times of the ILP solver. Also, in general the greedy algorithm gets slower with higher cover number, whereas the time needed to solve the ILP stays the same or even decreases.[2]

[2]This need not be the case for other problems: when applying the proposed method to very sparse graphs we observed much longer run times of the ILP solver.

Greedy			
Thr / CN	1	2	3
0.050	7.43 s	11.71 s	13.98 s
0.075	10.07 s	13.10 s	11.87 s
0.100	4.79 s	5.21 s	3.84 s
ILP			
Thr / CN	1	2	3
0.050	48.24 s	4.20 s	8.63 s
0.075	5.96 s	15.30 s	0.83 s
0.100	0.26 s	0.21 s	0.25 s

Table 3. Run times (in seconds) of the search for the minimum DS of DUBROVNIK; multi-threaded MATLAB implementations of the greedy algorithm and MOSEK [21] ILP solver.

Pose Estimation - registered images (of 800)						
DUBROVNIK	Greedy			ILP		
Thr / CN	1	2	3	1	2	3
0.050	679	724	729	669	705	731
0.075	719	737	734	717	736	732
0.100	644	687	680	650	658	677

Table 4. Evaluation of location recognition using pose estimation w.r.t. the 3D point cloud of the respective submodel, with different thresholds for image similarity and different cover numbers.

For each of the obtained subsets of images, a submodel was created by removing from the 3D model the unselected cameras, as well as 3D points which were left with less than two projections. Then, average SIFT descriptors of the 3D points in the submodel were computed using the remaining projections, and tentative 3D-to-2D matches between the submodel and each of the 800 query images were created with approximate NN-search. The poses of all query images with more than 20 tentative matches were estimated using P4Pfk (P3P with unknown focal length, [3]) in a RANSAC loop, with reprojection error threshold of 2 pixels and the maximum number of iterations limited to 200.

We use the same methodology as [18] and report the number of query images which attained at least twelve inliers supporting the recovered pose, see Table 4. The numbers of registered images are again roughly the same for the keyframe sets obtained using the greedy algorithm and those from the ILP solution, and also similar to the results presented in [18]. Notice that larger subsets do not always yield better registration performance.

Regarding localization accuracy, the median camera position error is around $0.9\,\mathrm{m}$ with quartiles $0.25\,\mathrm{m}$ and $4.5\,\mathrm{m}$, independent of both the threshold for image similarity and the cover number; an exception being the smallest subset (image similarity 0.05, cover number 1) where localization is less accurate. Similar to [18], there are around 100 cameras which have position errors larger than $20\,\mathrm{m}$.

4. Conclusions

We have shown that the search for the minimum dominating set of a graph—the problem which needs to be solved when selecting a compact, yet meaningful subset of a large image collection—can be formulated as an integer linear program, and solved efficiently by current ILP solvers. When comparing the sizes of keyframe sets obtained by the proposed method with those obtained with the state-of-the-art greedy algorithm, we found a gap of about 5%. This result empirically vindicates the greedy method, which performs surprisingly close to optimal, and much better than what could be expected from the theoretical guarantees. Apparently the structure of visual databases is favorable for greedy keyframe selection. Still, the optimal solution delivers consistently smaller keyframe sets, which could be significant for application scenarios with strict memory limitations. As an application example, we have shown that there is no perceptible difference between the greedy and optimal keyframe sets in location recognition problems, both for location recognition based on image retrieval with geometric verification, and for pose estimation w.r.t. a pre-computed 3D point cloud.

Acknowledgements

The authors would like to thank Vojtěch Franc for discussions about the ILP formulation of the CDS problem, and all Flickr users whose images were used in the project.

References

[1] S. Agarwal, N. Snavely, I. Simon, S. Seitz, and R. Szeliski. Building Rome in a day. In *ICCV*, 2009.

[2] H. Bay, A. Ess, T. Tuytelaars, and L. Van Gool. Speeded-up robust features (SURF). *CVIU*, 110(3):346–359, 2008.

[3] M. Bujnak, Z. Kukelova, and T. Pajdla. New efficient solution to the absolute pose problem for camera with unknown focal length and radial distortion. In *ACCV*, 2011.

[4] D. Chen et al. City-scale landmark identification on mobile devices. In *CVPR*, 2011.

[5] O. Chum, J. Philbin, and A. Zisserman. Near duplicate image detection: min-Hash and tf-idf weighting. In *BMVC*, 2008.

[6] A. Davison, I. Reid, N. Molton, and O. Stasse. MonoSLAM: Real-time single camera SLAM. *PAMI*, 29(6):1052–1067, 2007.

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

[8] J.-M. Frahm et al. Building Rome on a cloudless day. In *ECCV*, 2010.

[9] M. Garey and D. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. W.H. Freeman, 1979.

[10] Google. Google Goggles: Use pictures to search the web – `http://www.google.com/mobile/goggles`, 2011.

[11] S. Guha and S. Khuller. Approximation algorithms for connected dominating sets. *Algorithmica*, 20(4):374–387, 1998.

[12] R. Hammoud et al. Overhead-based image and video geolocalization framework. In *Perception Beyond the Visible Spectrum (PBVS)*, 2013.

[13] R. Hartley and A. Zisserman. *Multiple View Geometry in Computer Vision*. Cambridge University Press, second edition, 2003.

[14] M. Havlena, A. Torii, and T. Pajdla. Efficient structure from motion by graph optimization. In *ECCV*, 2010.

[15] A. Irschara, C. Zach, J.-M. Frahm, and H. Bischof. From structure-from-motion point clouds to fast location recognition. In *CVPR*, 2009.

[16] F. Kuhn and R. Wattenhofer. Constant-time distributed dominating set approximation. *Distributed Computing*, 17(4):303–310, 2005.

[17] X. Li, C. Wu, C. Zach, S. Lazebnik, and J.-M. Frahm. Modeling and recognition of landmark image collections using iconic scene graphs. In *ECCV*, 2008.

[18] Y. Li, N. Snavely, and D. Huttenlocher. Location recognition using prioritized feature matching. In *ECCV*, 2010.

[19] Y. Li, N. Snavely, D. Huttenlocher, and P. Fua. Worldwide pose estimation using 3D point clouds. In *ECCV*, 2012.

[20] D. Lowe. Distinctive image features from scale-invariant keypoints. *IJCV*, 60(2):91–110, 2004.

[21] Mosek ApS. MOSEK high performance optimization software – `http://mosek.com`, 2012.

[22] D. Nistér, O. Naroditsky, and J. Bergen. Visual odometry. In *CVPR*, 2004.

[23] D. Nistér and H. Stewénius. Scalable recognition with a vocabulary tree. In *CVPR*, 2006.

[24] A. Oliva and A. Torralba. Modeling the shape of the scene: A holistic representation of the spatial envelope. *IJCV*, 42(3):145–175, 2001.

[25] T. Sattler, B. Leibe, and L. Kobbelt. Fast image-based localization using direct 2D-to-3D matching. In *ICCV*, 2011.

[26] G. Schindler, M. Brown, and R. Szeliski. City-scale location recognition. In *CVPR*, 2007.

[27] J. Šivic and A. Zisserman. Video Google: Efficient visual search of videos. In *Toward Category-Level Object Recognition (CLOR)*, 2006.

[28] N. Snavely, S. Seitz, and R. Szeliski. Skeletal graphs for efficient structure from motion. In *CVPR*, 2008.

[29] A. Torii, M. Havlena, and T. Pajdla. Omnidirectional image stabilization for visual object recognition. *IJCV*, 91(2):157–174, 2011.

[30] A. Vedaldi and B. Fulkerson. VLFeat: An open and portable library of computer vision algorithms – `http://www.vlfeat.org`, 2008.

[31] C. Wu. VisualSFM: A visual structure from motion system – `http://ccwu.me/vsfm`, 2013.

[32] Yahoo! Flickr: Online photo management and photo sharing application – `http://www.flickr.com`, 2005.