Large Scale Visual Geo-Localization of Images in Mountainous Terrain

Georges Baatz, Olivier Saurer, Kevin Köser, and Marc Pollefeys

Department of Computer Science, ETH Zurich, Switzerland
{gbaatz,saurero,kevin.koeser,marc.pollefeys}@inf.ethz.ch

Abstract. Given a picture taken somewhere in the world, automatic geo-localization of that image is a task that would be extremely useful e.g. for historical and forensic sciences, documentation purposes, organization of the world's photo material and also intelligence applications. While tremendous progress has been made over the last years in visual location recognition within a single city, localization in natural environments is much more difficult, since vegetation, illumination, seasonal changes make appearance-only approaches impractical. In this work, we target mountainous terrain and use digital elevation models to extract representations for fast visual database lookup. We propose an automated approach for very large scale visual localization that can efficiently exploit visual information (contours) and geometric constraints (consistent orientation) at the same time. We validate the system on the scale of a whole country (Switzerland, 40 000km^2) using a new dataset of more than 200 landscape query pictures with ground truth.

1 Introduction and Previous Work

In intelligence and forensic scenarios as well as for searching archives and organising photo collections, automatic image-based location recognition is a challenging task that would however be extremely useful when solved. In such applications GPS tags are typically not available in the images requiring a fully image-based approach for geo-localization. During the last years progress has been made in urban scenarios, in particular with stable man-made structures that persist over time. However, for recognizing the camera location in natural environments the problem is substantially more challenging since vegetation changes rapidly during the seasons, and lighting and weather conditions (e.g. snow lines) make the use of appearance-based techniques (e.g. using patch-based local image features [1,2]) very difficult. Additionally, dense street-level data commonly used is limited to cities and major roads, and for the mountains or countryside only aerial footage exists that is much harder to relate with terrestrial imagery.

This paper targets camera geo-localization in natural environments. In particular we focus on recognizing the skyline in a query image, given a digital elevation model (DEM) of a country — or ultimately, the world. In contrast to previous work of matching e.g. a peak in the image to a set of mountains known to be nearby, we aggregate shape information across the whole skyline (not only

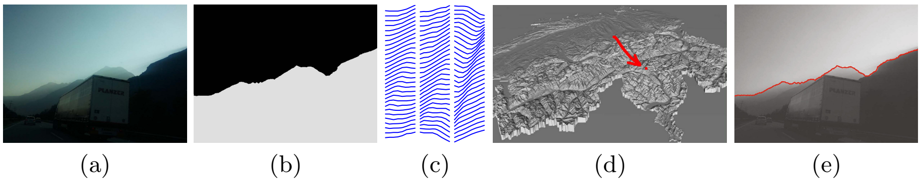


Fig. 1. Different stages in the proposed pipeline: (a) Query image somewhere in Switzerland, (b) sky segmentation, (c) sample set of extracted 10° contourlets, (d) recognized geo-location in digital elevation model, (e) overlaid visible horizon at estimated position

the peaks) and search for a similar configuration of basic shapes in a large scale database that is organized to allow for query images of largely different fields of view. The method is based on sky segmentation (either automatic or easily supported by an operator for challenging pictures such as those with reflection, occlusion or taken from inside a cable car).

Contributions. The main contributions are a novel method for robust contour encoding as well as two different voting schemes to solve the large scale camera pose recognition from contours. The first scheme operates only in descriptor space (it checks where in the model a panoramic skyline is most likely to *contain* the current query picture) while the second one is a combined vote in descriptor and rotation space. We validate the whole approach using a public digital elevation model of Switzerland that covers more than 40 000km^2 and a set of more than 200 images from different sources with ground truth information that is published along with the paper. In particular we show the improvements of all novel contributions compared to a baseline implementation motivated by classical bag-of-words [3] based techniques like [2]. Also, we demonstrate that the horizon is highly informative and can be used effectively for localization.

Previous Work. To the best of our knowledge there has never been a similar attempt at large-scale localization of a photo based on a digital elevation model before. The closest works to ours are smaller scale navigation and localization in robotics [4,5], and building/location recognition in cities [1,6,2] or with respect to community photo collections of popular landmarks [7]. These, however, do not apply to landscape scenes of changing weather, vegetation, snowlines, or changing lighting conditions. The robotics community has considered the problem of robot navigation and robot localization using digital elevation models for quite some time. Talluri et al. [8] reason about intersection of known viewing ray directions (north, east, south, west) with the skyline and relies thus on the availability of 360° panoramic query contours and the knowledge of vehicle orientation (i.e. north direction). Thompson et al. [9] suggest general concepts of how to estimate pose and propose a hypothesize and verify scheme. They also rely on known view orientation and match viewpoint-independent features (peaks, saddle points, ...) from a DEM to features found in the image, this way ignoring much of the skyline information. In [10], computer vision techniques are used to extract mountain peaks which are matched to a database of nearby mountains to support a remote operator in navigation. However, we believe that their approach of considering relative positions of absolute peaks detected in a DEM is too restrictive and would not scale to our orders of magnitude larger problem, in particular with respect to less discriminative locations. [11] proposes to first match three features of a contour to a digital elevation model and estimate an initial pose from that before a non-linear refinement. Also here the first step of finding three good correspondences is again not easily possible in larger databases. [5] assumes panoramic query data with known heading, and computes super-segments on a polygon fit, however descriptiveness/robustness is not evaluated on a bigger database, while [12] introduces a probabilistic formulation for a similar setting. The key point is that going from tens of potential locations to millions of locations requires a conceptually different approach, since exhaustive image comparison or trying all "mountain peaks" simply does not scale to a large-scale geo-localization problem. Similarly, for urban localization, in [13] an upward looking 180° field of view fisheye is used for navigation in urban canyons. The authors render untextured city models near the predicted pose and extract contours for comparison with a query image. Also this is meant as a local method for navigation. Most recently, in [14] the authors optimize the camera orientation given the exact position, i.e. they estimate the viewing direction given a good GPS tag. None of the above systems considered large scale recognition.

On the earth scale, the authors of [15] source photo collections and aim at learning location probability based on color, texture, and other image-based statistics. Conceptually, this is not meant to find an exact pose based on geometric considerations but rather discriminates landscapes or cities with different (appearance) characteristics on a global scale. The authors of [16] exploit the position of the sun (given the time) for geo-localization. In the same work it is also shown that identifying a large piece of clear sky without haze provides information about the camera pose (although impressive given the data, still more than 100km average localization error is reported). Both approaches are appealing for excluding large parts of the earth from further search but do not aim at exactly localizing the camera within a few hundred meters.

Besides attacking the DEM-based, large scale geo-localization problem we propose new techniques that might also be transferred to bag-of-words approaches based on local image patches (e.g. [3,1,2]). Those approaches typically rely on pure occurrence-based statistics (visual word histogram) to generate a first list of hypotheses and only for the top candidates geometric consistency of matches is checked. Such a strategy fails in cases where pure feature coocurrence is not discriminative but where the relative locations of the features are important. Here, we propose to do a (weak) geometric verification already in the histogram distance phase. Furthermore, we show also a representation that tolerates largely different document sizes (allowing to compare a panorama in the database to an image with an order of magnitude smaller field of view).

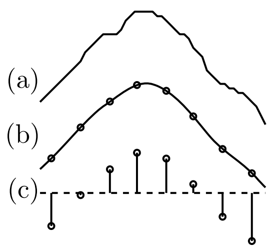
2 Mountain Recognition Approach

The location recognition problem in its general form is six-dimensional since three position and three orientation parameters have to be estimated. We make the assumption that the photo has been taken not too far from the ground and use the fact that people rarely twist the camera relative to the horizon [17] (i.e. small roll). We will propose a method to solve that problem using the outlines of mountains against the sky (a skyline), which we call the **visible horizon**. For the visual database we seek a representation that is robust with respect to tilt of the camera which means that we are effectively left with estimating the 2D position on our height model (latitude and longitude) and the viewing direction of the camera. The visible horizon of the DEM is extracted offline at regular grid positions (360° at each position) and represented by a collection of vector-quantized local contourlets (contour words, similar in spirit to visual words obtained from quantized image patch descriptors [3]). In contrast to visual word based approaches, additionally an individual viewing angle α_d ($\alpha_d \in [0; 2\pi]$) relative to north direction is stored. At query time, a sky segmentation technique is applied that copes with the often present haze. Subsequently the extracted contour is robustly described by a set of local contourlets plus their relative angular distance α_q with respect to the optical axis of the camera. Then, we use an inverted file system for the contour words to find the most promising location and simultaneously vote for the viewing direction, which is an integrated geometric verification already during the bag-of-words search.

2.1 Processing the Query Image

Sky Segmentation. The estimation of the visible horizon can be cast as a foreground-background segmentation problem. Virtually any good segmentation method could be used here and the focus of this paper is on the recognition, not on the segmentation part. Consequently, due to space limitations in the following we can only sketch our implementation roughly:

Since sky and mountains have different local statistics with respect to color, gradient and other cues' distribution, we decided to follow a standard approach based on unary data costs [18,19] for a pixel being assigned sky or ground and smoothness costs along the visible horizon that depend also on the local image structure [20]. As we assume almost no camera roll and since overhanging structures are extremely rare, finding the highest foreground pixel (foreground height) for each image column provides an excellent approximation, at the same time allowing for a dynamic programming solution (as suggested already in e.g. [21]). To obtain the data term for a candidate height in a column we sum all foreground costs below the candidate contour and sky costs above the contour, where we have trained a pixel's color and gradient likelihoods for ground and sky. Furthermore, we dehaze the image according to [22] and exploit the (continuous) relative depth information (obtained from dehazing and normalized to maximum depth 1) to steer the smoothness term similar to [20]. The intuition is that when running along the visible horizon from one candidate height in



(d)

-0.14	-0.01	0.10	0.17	0.15	0.04	-0.10	-0.21

(e)

001	011	101	111	110	100	001	000

(f) 001011101111110100000100

Fig. 2. Contour word computation: (a) raw contour, (b) smoothed contour with n sampled points, (c) sampled points after normalization, (d) contourlet as numeric vector, (e) each dimension quantized to 3 bits, (f) contour word as 24-bit integer

one column to a candidate height in the next column, there should be local evidence in terms of an orthogonal depth gradient on the connection line. We train weights and foreground/sky likelihoods on a manually labelled offline data set. Further, we add interaction capabilities for a potential user where simple strokes can mark foreground or sky in the query picture forcing these pixels and all below or above to these labels. This provides a simple and effective means to correct very challenging situations (e.g. reflections in picture from inside a cable car, power lines or people corrupting the skyline) or previously unseen colors (e.g. picture at sunset or night).

Contourlet Extraction. In the field of shape recognition, there are many shape description techniques that deal with closed contours, e.g. [23]. However, recognition based on partial contours is still a largely unsolved problem, because it is difficult to find representations invariant to viewpoint. For the sake of robustness to occlusion, to noise and systematic errors (inaccurate focal length estimate or tilt angle), we decided to use local representations of the visible horizon curve (compare also [24] for an overview on shape features).

To describe the contour, we consider overlapping curvelets of width w (imagine a sliding window, see Fig. 1). These curvelets are then sampled at n equally spaced points, yielding each an n-dimensional vector $\tilde{y}_1, \ldots, \tilde{y}_n$ (before sampling, we low-pass filter the horizon to avoid aliasing). The final descriptor is obtained by subtracting the mean and dividing by the feature width (see Fig. 2(a)–(d)):

$$y_i = \frac{\tilde{y}_i - \bar{y}}{w} \text{ for } i = 1, \ldots, n \qquad \text{where} \qquad \bar{y} = \frac{1}{n}\sum_{j=1}^{n} \tilde{y}_j \tag{1}$$

Mean subtraction makes the descriptor invariant w.r.t. vertical image location (and therefore robust against camera tilt). Scaling ensures that the y_i's have roughly the same magnitude, independently of the feature width w.

The next step is vector quantisation (Fig. 2(e)–(f)). Since our features are very low-dimensional compared to traditional patch-based feature descriptors like SIFT [25], we choose not to use a vocabulary tree. Instead, we directly quantize each dimension of the descriptor *separately*, which is both faster and much more memory-efficient compared to a vocabulary tree and the best bin is guaranteed to be found. Each y_i falls into one bin and the n associated bin numbers are concatenated into a single integer (contour word). For each descriptor, we also make note of the viewing direction α_q (relative to the camera's optical axis) where the feature was observed, which can be calculated using the camera's intrinsics. We have verified that an approximate focal length estimate is enough. In case the focal length is completely unknown, it is possible to sample several tentative values (see evaluation in Section 3).

2.2 Visual Database Creation

The digital elevation model we use for validation is available from the Swiss Federal Office of Topography, and similar data sets exist also for the US and other countries. There is one sample point per 2 square meters and the height quality varies from 0.5m (flat regions) to 3m-8m (above 2000m elevation) average error[1]. This data is converted to a triangulated surface model with level-of-detail support in a scene graph representation[2]. At each position on a regular grid on the surface (every 0.001° in N-S direction and 0.0015° in E-W direction, i.e. 111m and 115m respectively) and from 1.80m above the surface[3], we render a cube-map of the textureless DEM (face resolution 1024×1024) and extract the visible horizon by checking for the rendered sky color. Overall, we generate 3.5 million cubemaps. Similar to the query image, we extract contourlets, but this time with *absolute* viewing direction. We organize the contourlets in an index to allow for fast retrieval. In image search, inverted files have been used very successfully for this task [3]. We extend this idea by also taking into account the viewing direction, so that we can perform rough geometric verification on-the-fly. For each word we maintain a list that stores for every occurrence the panorama ID and the azimuth α_d of the contourlet.

2.3 Recognition and Verification

Baseline. The baseline for comparison is an approach borrowed from patch based systems (e.g. [26,1,2]) based on the (potentially weighted) L1-norm between normalized visual word frequency vectors:

$$D^E(\tilde{\mathbf{q}},\tilde{\mathbf{d}}) = \|\tilde{\mathbf{q}}-\tilde{\mathbf{d}}\|_1 = \sum_i |\tilde{q}_i - \tilde{d}_i| \quad \text{or} \quad D^{E_w}(\tilde{\mathbf{q}},\tilde{\mathbf{d}}) = \sum_i w_i |\tilde{q}_i - \tilde{d}_i| \qquad (2)$$

$$\text{with} \quad \tilde{\mathbf{q}} = \frac{\mathbf{q}}{\|\mathbf{q}\|_1} \quad \text{and} \quad \tilde{\mathbf{d}} = \frac{\mathbf{d}}{\|\mathbf{d}\|_1} \qquad (3)$$

1 `http://www.swisstopo.admin.ch/internet/swisstopo/en/home`
2 `http://openscenegraph.org`
3 Synthetic experiments verified that taking the photo from ten or fifty meters above the ground does not degrade recognition besides very special cases like standing very close to a small wall.

Where q_i and d_i is the number of times visual word i appears in the query or database image respectively, and $\tilde{q}_i$, $\tilde{d}_i$ are their normalized counterparts. w_i is the weight of visual word i (e.g. as obtained by the term frequency - inverse document frequency (tf-idf) scheme). This gives an ideal score of 0 when both images contain the same visual words at the same proportions, which means that the L1-norm favors images that are *equal* to the query.

[26] suggested transforming the weighted L1-norm like this

$$D^{E_w}(\tilde{\mathbf{q}}, \tilde{\mathbf{d}}) = \sum_i w_i \tilde{q}_i + \sum_i w_i \tilde{d}_i - 2 \sum_{i \in Q} w_i \min(\tilde{q}_i, \tilde{d}_i) \qquad (4)$$

in order to enable an efficient method for evaluating it by iterating only over the visual words present in the query image and updating only the scores of database images containing the given visual word.

"Contains"-Semantics. In our setting, we are comparing 10°–70° views to 360° panoramas, which means that we are facing a 5×–36× difference of magnitude. Therefore, it seems ill-advised to implement an "equals"-semantics, but rather one should use a "contains"-semantics. We modify the weighted L1-norm as follows:

$$D^C(\mathbf{q}, \mathbf{d}) = \sum_i w_i \max(q_i - d_i, 0) \qquad (5)$$

The difference is that we are using the raw contour word frequencies, q_i and d_i without scaling and we replace the absolute value $|\cdot|$ by $\max(\cdot, 0)$. Therefore, one only penalizes contour words that occur in the query image, but not in the database image (or more often in the query image than in the database image). An ideal score of 0 is obtained by a database image that contains every contour word at least as often as the query image, plus any number of other contour words. If the proposed metric is transformed as follows, it can be evaluated just as efficiently as the baseline:

$$D^C(\mathbf{q}, \mathbf{d}) = \sum_{i \in Q} w_i q_i - \sum_{i \in Q} w_i \min(q_i, d_i) \qquad (6)$$

This subtle change makes a huge difference, see Fig. 5(a) and Table 1: (B) versus (C). Note that this might also be applicable to other cases where a "contains"-semantics is desirable.

Location and Direction. We further refine retrieval by taking geometric information into account already during the voting stage. Earlier bag-of-words approaches accumulate evidence purely based on the frequency of visual words. Voting usually returns a short-list of the top n candidates, which are reranked using geometric verification (typically using the number of geometric inliers). For performance reasons, n has to be chosen relatively small (e.g. $n = 50$). If the correct answer already fails to be in this short-list, then no amount of reordering

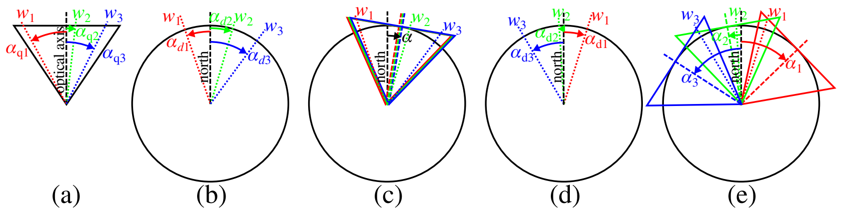


Fig. 3. Voting for a direction is illustrated using a simple example: We have a query image (a) with contour words w_i and associated angles α_{qi}. We consider a panorama (b) with contour words in the same *relative* positions α_{di} as the query image. Since the contour words appear in the same order, they all vote for the same viewing direction α (c). In contrast, we consider a second panorama (d) with contour words in a different order. Even though the contour words occur in close proximity they each vote for a different direction α_i, so that none of the directions gets a high score (e).

can bring it back. Instead, we check for geometric consistency already at the voting stage, so that fewer good candidates get lost prematurely. Not only does this increase the quality of the short-list, it also provides an estimated viewing direction, which can be used as an initial guess for the full geometric verification. Since this enables a significant speedup, we can afford to use a longer short-list, which further reduces the risk of missing the correct answer.

If the same contour word appears in the database image at angle α_d (relative to north) and in the query image at angle α_q (relative to the camera's optical axis), the camera's azimuth can be calculated as $\alpha = \alpha_d - \alpha_q$. Weighted votes are accumulated using soft binning and the most promising viewing direction(s) are passed on to full geometric verification. This way, panoramas containing the contour words in the right order get many votes for a single direction, ensuring a high score. For panoramas containing only the right mix of contour words, but in random order, the votes are divided among many different directions, so that none of them gets a good score (see Fig. 3). Note that this is different from merely dividing the panoramas into smaller sections and voting for these sections: Our approach effectively requires that the order of contour words in the panorama matches the order in the query image. As an additional benefit, we do not need to build the inverted file for any specific field-of-view of the query image.

Geometric Verification. We verify the top 1000 candidates with a geometric check. Verification consists in calculating an optimal alignment of the two visible horizons using iterative closest points (ICP). While we consider in the voting stage only one angle (azimuth), ICP determines a full 3D rotation. First, we sample all possible values for azimuth and keep the two other angles at zero. The most promising one is used as initialization for ICP. In the variants that already vote for a direction, we try only a few values around the highest ranked ones. The average alignment error is used as a score for re-ranking the candidates.

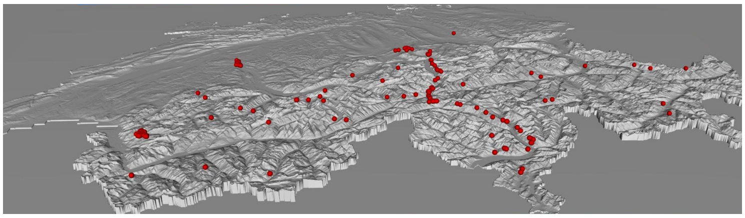


Fig. 4. Oblique view onto Switzerland model (40 000km^2). Spheres indicate the 213 query images' ground truth coordinates (size reflects 1km tolerance radius). Source of DEM data: Bundesamt für Landestopografie swisstopo (Art. 30 GeoIV): 5704 000 000

3 Evaluation

Query Set. In order to evaluate the approaches we assembled a set of photos from different sources such as internet photo collections and on site. For all of the photos, we verified the GPS tag or location estimate by comparing the skyline to the surface model. For a large part the information is consistent, only for a small fraction the position did not match the digital elevation model view. This can be explained by a bad cell phone GPS tag, because of bad/no GPS reception at the time the photo was taken. For these cases, we use dense geometric verification (on each 111×115m grid position up to a 10km radius around the tagged position) to generate hypotheses for the correct GPS tag. We verify this by visual inspection and remove the image in case of no agreement. The whole set of query images is available at the project website[4] and the distribution of pictures on the map is displayed in Fig. 4. For all of the query images FoV information is available (e.g. from EXIF tag). However, we have verified experimentally that also in case of fully unknown focal length the system can be applied by sampling over this parameter (see Fig. 8 for an example).

Parameter Selection. The features need to be clearly smaller than the images' field-of-view, but wide enough to capture the geometry rather than just discretization noise. We consider descriptors of width $w = 10°$ and $w = 2.5°$. The number of sample points n should not be so small that it is uninformative (e.g. $n = 3$ would only distinguish concave/convex), but not much bigger than that otherwise it risks being overly specific, so we choose $n = 8$. The curve is smoothed by a Gaussian with $\sigma = \frac{w}{2n}$, i.e. half the distance between consecutive sample points. Descriptors are extracted every σ degrees.

Each dimension of the descriptor is quantized into k bins of width 0.375, the first and last bin extending to infinity. We chose k as a power of 2 that results in roughly 1 million contour words, i.e. $k = 8$. This maps each y_i to 3 bits, producing contour words that are 24 bit integers. Out of the 2^{24} potential contour words, only 300k–500k (depending on w) remain after discarding words that occur too often (more than a million) or not at all.

[4] `http://cvg.ethz.ch/research/mountain-localization`

Table 1. Overview of tested recognition pipelines

	Voting scheme	Descriptor width	Dir. bin size	Geo. ver.	Top 1 correct
(A)	random	N/A	N/A	no	0.008%
(B)	"equals"	$10°$	N/A	no	8%
(C)	"contains"	$10°$	N/A	no	30%
(D)	loc.&dir.	$10°$	$2°$	no	45%
(E)	loc.&dir.	$10°$	$3°$	no	44%
(F)	loc.&dir.	$10°$	$5°$	no	46%
(G)	loc.&dir.	$10°$	$10°$	no	42%
(H)	loc.&dir.	$10°$	$20°$	no	38%
(I)	loc.&dir.	$2.5°$	$3°$	no	30%
(J)	loc.&dir.	$10°$&$2.5°$	$3°$	no	62%
(K)	loc.&dir.	$10°$&$2.5°$	$3°$	yes	88%

Recognition Performance. All of the tested recognition pipelines (see Table 1) return a ranked list of candidates. We evaluate them as follows: For every $n = 1, \ldots, 100$, we count the fraction of query images that have at least one correct answer among the top n candidates. We consider an answer correct if it is within 1km of the ground truth position (see Fig. 4).

In Fig. 5(a), we compare different voting schemes: (B) voting for location only, using the traditional approach with normalized visual word vectors and L1-norm ("equals"-semantics); (C) voting for location only, with our proposed metric ("contains"-semantics); (E) voting for location and direction simultaneously (i.e. taking order into account). All variants use $10°$ descriptors. For comparison, we also show (A) the probability of hitting a correct panorama by random guessing

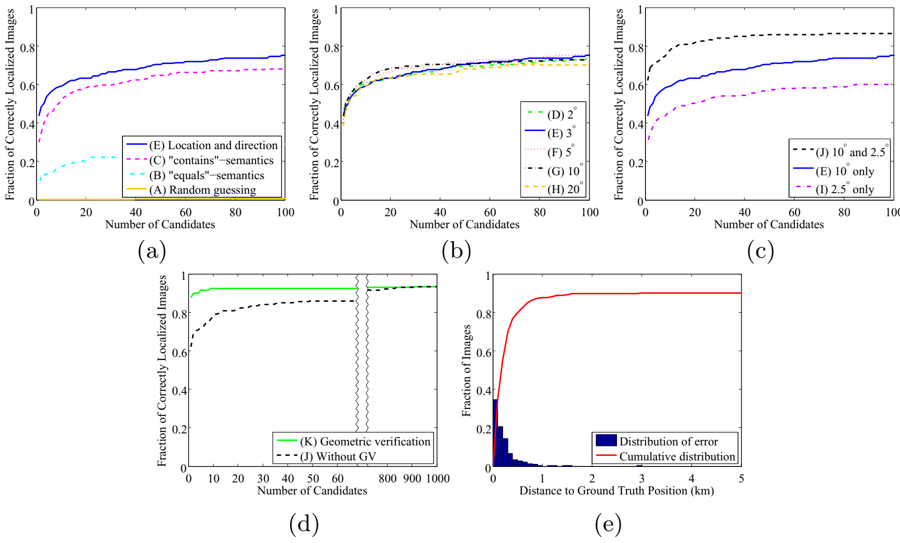


Fig. 5. Retrieval performance for different: (a) voting schemes, (b) bin sizes in direction voting, (c) descriptor sizes. (d) Retrieval performance before and after geometric verification. (e) Fraction of queries having (at most) a given distance to the ground truth position. Not shown: 21 images (9.9%) with an error between 7 and 217km.

(the probability of a correct guess is extremely small, which shows that the tolerance of 1km is not overly generous). Our proposed "contains"-semantics alone already outperforms the baseline ("equals"-semantics) by far, but voting for a direction is even better!

In Fig. 5(b), we analyze how different bin sizes for direction voting affects results. (D)–(H) correspond to bin sizes of $2^\circ, 3^\circ, 5^\circ, 10^\circ, 20^\circ$ respectively. While there are small differences, none of the settings outperforms an other consistently: Our method is quite insensitive over a large range of this parameter.

In Fig. 5(c), we study the impact of different descriptor sizes: (E) only 10° descriptors; (I) only 2.5° descriptors; (J) both 10° and 2.5° descriptors combined. All variants vote for location and direction simultaneously. While 10° descriptors outperforms 2.5° descriptors, the combination of both is better than either descriptor size alone. This demonstrates that different scales capture different information, which complement each other.

In Fig. 5(d), we show the effect of geometric verification by aligning the full countours using ICP: (J) 10° and 2.5° descriptors voting for location and direction, without verification; (K) same as (J) but with geometric verification. We see that ICP based reranking is quite effective at moving the best candidate(s) to the beginning of the short list: With probability 88%, the top-ranked candidate is within a radius of 1km from the ground truth position. See Fig. 5(e) for other radii. In computer assisted search scenarios, an operator would choose an image from a small list which would further increase the percentage of correctly recovered pictures. Besides that, from geometric verification we not only obtain an estimate for the viewing direction but the full camera orientation which can be used for augmented reality. Fig. 6&7 show successful and unsuccessful examples.

We also ran the experiment after artificially adding tilt to the queries and determined that recognition performance is largely unaffected up to 20° tilt.

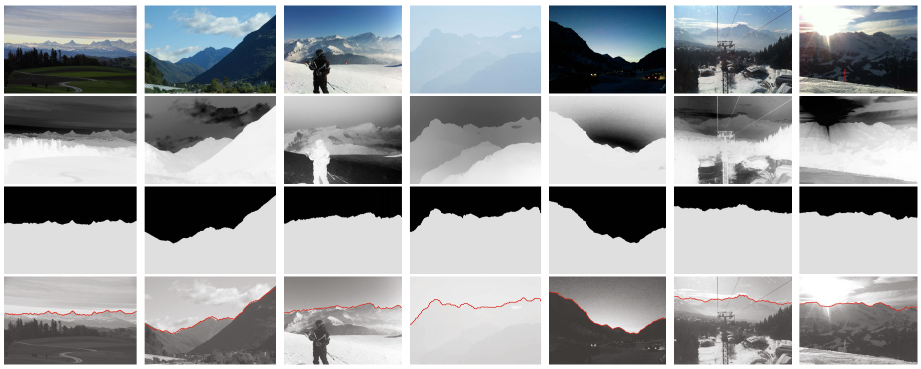


Fig. 6. Sample Results: Input Image (top row), depth image from dehazing (second row), segmentation (third row) and augmentation of retrieved contour from database onto bright version of original image (bottom row). The images in the third and the two right-most columns have been segmented with the help of user interaction.

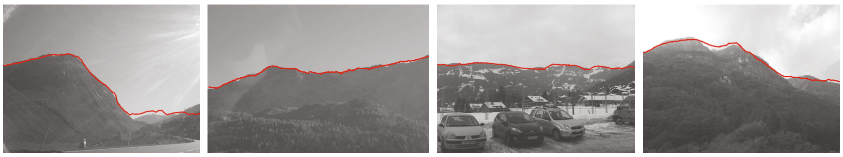


Fig. 7. Some incorrectly localized images. This usually happens to images with a relatively smooth horizon and only few distinctive features. The pipeline finds a contour that fits somewhat well, even if the location is completely off.

Field-of-View. Fig. 8 illustrates the effect of inaccurate or unknown field-of-view (FoV). For one query image, we run the localization pipeline (K) assuming that the FoV is 11° and record the results. Then we run it again assuming that the FoV is 12° etc, up to 70°. Fig. 8 shows how the alignment error and estimated position depend on the assumed FoV.

In principle, it is possible to compensate a wrong FoV by moving forward or backward. This holds only approximately if the scene is not perfectly planar. In addition, the effect has hard limits because moving too far will cause objects to move in or out of view, changing the visible horizon. Between these limits, changing the FoV causes both the alignment error and the position to change smoothly. Outside of this stable range, the error is higher, fluctuates more and the position jumps around wildly.

This has two consequences: First, if the FoV obtained from the image's meta-data is inaccurate it is usually not a disaster, the retrieved position will simply be slightly inaccurate as well, but not completely wrong. Second, if the FoV is completely unknown, one can get a rough estimate by choosing the minimum error and/or looking for a range where the retrieved position is most stable.

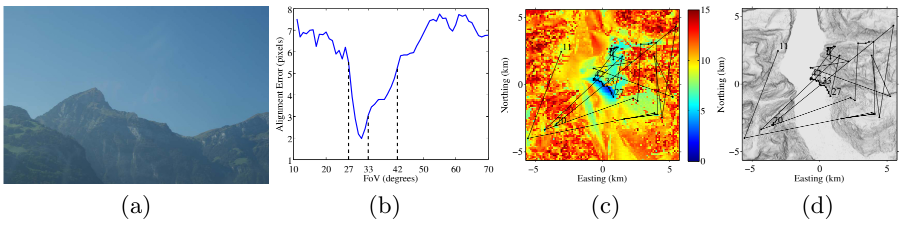


Fig. 8. (a) Query image. (b) Alignment error of the best position for a given FoV. Dashed lines indicate the limits of the stable region and the FoV from the image's EXIF tag. (c) Alignment error of the best FoV for a given position. For an animated version, see `http://cvg.ethz.ch/research/mountain-localization`. (d) Shaded terrain model. The overlaid curve in (c) and (d) starts from the best location assuming 11° FoV and continues to the best location assuming 12°, 13°, etc. Numbers next to the markers indicate corresponding FoV.

Runtime. We implemented the algorithm partly in C/C++ and partly in Matlab. On the evaluation dataset 51% of the images were segmented completely automatically, 42% required small user interaction, such as correcting for occlusions and 7% needed more intensive manual labeling i.e. correcting for small snow fields and reflections etc. After segmentation, one image takes 10 seconds to find the camera's position and rotation in an area of 40 000km^2 (Exhaustively computing an optimal alignment between the query image and each of the 3.5M panoramas would take on the order of several days). For comparison, the authors of [14] use a GPU implementation and report 2 minutes for determining rotation alone (assuming that the camera position is already given).

4 Conclusion and Future Work

We have presented a system for large scale location recognition based on digital elevation models. This is very valuable for geo-localization of pictures when no GPS information is available (for virtually all video or DSLR cameras, archive pictures, in intelligence and military scenarios). We extract the sky and represent the visible horizon by a set of contour words, where each contour word is represented together with its offset angle from the optical axis. This way, we can do a bag-of-words like approach with integrated geometric verification, i.e. we are looking for the panorama (portion) that has a similar frequency of contour words with a consistent direction. We show that our representation is very discriminative and the full system allows for excellent recognition rates close to 90% on our challenging dataset including different seasons, landscapes and altitudes. We believe that this is a step towards the ultimate goal of being able to geo-localize images taken anywhere on the planet, but for this also other additional cues of natural environments have to be combined with the given approach. This will be the subject of future research.

Acknowledgements. This work has been supported through SNF grant 127224 by the Swiss National Science Foundation. We would also like to thank Simon Wenner for his help with rendering the DEMs.

References

1. Schindler, G., Brown, M., Szeliski, R.: City-scale location recognition. In: Computer Vision and Pattern Recognition, CVPR 2007, pp. 1–7 (2007)
2. Chen, D., Baatz, G., Köser, K., Tsai, S., Vedantham, R., Pylvanainen, T., Roimela, K., Chen, X., Bach, J., Pollefeys, M., Girod, B., Grzeszczuk, R.: City-scale landmark identification on mobile devices. In: CVPR (2011)
3. Sivic, J., Zisserman, A.: Video Google: A text retrieval approach to object matching in videos. In: Proc. ICCV 2003, vol. 2, pp. 1470–1477 (2003)
4. Woo, J., Son, K., Li, T., Kim, G.S., Kweon, I.S.: Vision-based uav navigation in mountain area. In: MVA, pp. 236–239 (2007)

5. Stein, F., Medioni, G.: Map-based localization using the panoramic horizon. IEEE Transactions on Robotics and Automation 11, 892–896 (1995)
6. Baatz, G., Köser, K., Chen, D., Grzeszczuk, R., Pollefeys, M.: Leveraging 3d city models for rotation invariant place-of-interest recognition. IJCV, Special Issue on Mobile Vision 96 (2012)
7. Li, Y., Snavely, N., Huttenlocher, D.P.: Location Recognition Using Prioritized Feature Matching. In: Daniilidis, K., Maragos, P., Paragios, N. (eds.) ECCV 2010, Part II. LNCS, vol. 6312, pp. 791–804. Springer, Heidelberg (2010)
8. Talluri, R., Aggarwal, J.: Position estimation for an autonomous mobile robot in an outdoor environment. Transact. Robotics and Automation 8, 573–584 (1992)
9. Thompson, W.B., Henderson, T.C., Colvin, T.L., Dick, L.B., Valiquette, C.M.: Vision-based localization. In: Image Understanding Workshop, pp. 491–498 (1993)
10. Cozman, F., Krotkov, E.: Position estimation from outdoor visual landmarks for teleoperation of lunar rovers. In: WACV 1996, pp. 156–161 (1996)
11. Naval, P.C., Mukunoki, M., Minoh, M., Ikeda, K.: Estimating camera position and orientation from geographical map and mountain image. In: 38th Pattern Sensing Group Research Meeting, Soc. of Instrument and Control Engineers, pp. 9–16 (1997)
12. Cozman, F.: Decision Making Based on Convex Sets of Probability Distributions: Quasi-Bayesian Networks and Outdoor Visual Position Estimation. PhD thesis, Robotics Institute, Carnegie Mellon University, Pittsburgh, PA (1997)
13. Ramalingam, S., Bouaziz, S., Sturm, P., Brand, M.: Skyline2gps: Localization in urban canyons using omni-skylines. In: IROS 2010, pp. 3816–3823 (2010)
14. Baboud, L., Cadík, M., Eisemann, E., Seidel, H.P.: Automatic photo-to-terrain alignment for the annotation of mountain pictures. In: CVPR, pp. 41–48 (2011)
15. Hays, J., Efros, A.A.: im2gps: estimating geographic information from a single image. In: Proceedings of CVPR 2008 (2008)
16. Lalonde, J.F., Narasimhan, S.G., Efros, A.A.: What do the sun and the sky tell us about the camera? International Journal on Computer Vision 88, 24–51 (2010)
17. Brown, M., Lowe, D.G.: Automatic panoramic image stitching using invariant features. Int. J. Comput. Vision 74, 59–73 (2007)
18. Luo, J., Etz, S.: A physics-motivated approach to detecting sky in photographs. In: Proc. ICPR., vol. 1, pp. 155–158 (2002)
19. Martin, D.R., Fowlkes, C.C., Malik, J.: Learning to detect natural image boundaries using local brightness, color, and texture cues. PAMI 2004 26, 530–549 (2004)
20. Blake, A., Rother, C., Brown, M., Perez, P., Torr, P.: Interactive Image Segmentation Using an Adaptive GMMRF Model. In: Pajdla, T., Matas, J(G.) (eds.) ECCV 2004. LNCS, vol. 3021, pp. 428–441. Springer, Heidelberg (2004)
21. Lie, W.N., Lin, T.C.I., Lin, T.C., Hung, K.S.: A robust dynamic programming algorithm to extract skyline in images for navigation. Pattern Recognition Letters 26, 221–230 (2005)
22. He, K., Sun, J., Tang, X.: Single image haze removal using dark channel prior. In: Computer Vision and Pattern Recognition, pp. 1956–1963 (2009)
23. Manay, S., Cremers, D., Hong, B.W., Yezzi, A., Soatto, S.: Integral invariants for shape matching. Pattern Analysis and Machine Intelligence (2006)
24. Yang, M., Kpalma, K., Ronsin, J.: A Survey of Shape Feature Extraction Techniques. In: Yin, P.Y. (ed.) Pattern Recognition, pp. 43–90. IN-TECH (2008)
25. Lowe, D.G.: Distinctive Image Features from Scale-Invariant Keypoints. International Journal of Computer Vision 60, 91–110 (2004)
26. Nistér, D., Stewénius, H.: Scalable recognition with a vocabulary tree. In: CVPR (2), pp. 2161–2168. IEEE Computer Society (2006)