18th International Conference on Information Fusion
Washington, DC - July 6-9, 2015

Camera Geolocation From Mountain Images

Yi Chen, Gang Qian, Kiran Gunda, Himaanshu Gupta, and Khurram Shafique
ObjectVideo, Inc., 11600 Sunrise Valley Drive, Reston, VA 20191
{ychen,gqian,kgunda,hgupta,kshafique}@objectvideo.com

***Abstract*—This paper presents a novel approach to camera geolocation from mountain images. Existing mountain view-based geolocation techniques use only mountain skylines and assume that such skylines are available and can be reliably extracted from query images. However, in real-life scenarios the skyline in a query image may be blurred or invisible, due to occlusions, adverse weather conditions and atmospheric effects, and poor image quality. Geolocating mountain view images with poor skylines is a challenge. In addition, when geolocating a query image, existing techniques do not estimate the camera roll angle, assuming that the roll angle is always small and its impact to the geolocation accuracy is negligible. However, in our research we have observed that even a small camera roll angle of only a few degrees can significantly alter the skyline features extracted from the query image and thus cause geolocation failure due to feature mismatching between the query and feature database. It is another challenge to reliably handle query images with non-negligible camera roll angles. In this paper, we propose a novel solution to these challenges by exploiting additional visual features extracted from mountain ridges beyond skylines and performing search-based camera roll angle estimation. Our proposed approach has been extensively tested on real-world challenging query images from five different regions in three continents. The experimental results from our proposed approach is significantly superior to those obtained from state-of-the-art skyline-based image geolocation approaches.**

***Index Terms*—Image geolocation, digital elevation model, terrain matching, bag-of-words**

I. Introduction

Geolocation of an image is a challenging task that has received significant attention in recent years, largely due to the increasing availability of large amount of geo-tagged images on the Internet (such as social media websites Flickr, Panoramio, etc.) as well as the fast growing computational power of modern computers. However, state-of-the-art automated image geolocation techniques have been largely limited to finding matches with geo-tagged ground imagery [10], [22], [23]. Such approaches are only applicable to frequently visited and photographed areas that are well represented in ground-imagery databases. Unfortunately, this is not the case for a large portion of the world. While recently, some methods have been proposed to address this problem by using a combination of sparse ground-level images with overhead views [13] for localization in small geospatial regions, the applicability of such methods over large geospatial regions (of the order of 10,000 km^2) is yet to be established. Furthermore, none of the above-mentioned techniques are applicable in mountainous regions that typically do not have many reference geo-tagged training images.

It is therefore important to use additional information sources, such as 3D terrain models, to estimate the geolocation of the camera for images taken in mountainous regions. Given these data sources, the challenges are then to i) extract relevant features from the reference data sources as well as (often times noisy) query images; ii) index the extracted features from reference data for efficient searching; and iii) robustly match the features from the query to the reference data. For example, it has been demonstrated that the visible skyline from query image is a robust feature [2], [3], [15], [20], [24] that can be indexed and matched with terrain features for image geolocalization in mountainous regions. In particular, a novel efficient algorithm has been proposed in [2], which relies on only the visible skyline extracted from the query image and its similarity with features extracted from digital elevation models (DEM) to accurately estimate camera location in large geospatial regions.

While visible skyline based matching has been shown to perform well in standard research datasets, this approach has its limitations when dealing with imagery from real-world settings. For example, real-world systems often have to deal with challenging cases where the skyline is occluded, blurred (or completely missing) by atmospheric effects (such as heavy fog) or by other physical objects (such as vegetation and buildings). For example, Fig. 1 shows query images in which the visible skyline is considerably different from the synthetic views rendered from DEM using ground truth camera parameters. In these cases, the skyline cannot be reliably extracted from query images even with the help of a user-in-the-loop, thereby affecting the quality of matching. Furthermore, the real-world imagery often has non-negligible camera roll angles that also pose a challenge to existing approaches. Some query images with camera rolls are shown in Fig. 2. Due to absence of any reference features, e.g. horizon line, it is obvious that estimating the roll angles from these images is not a trivial task. The paper addresses these challenges and proposes a solution that is more practical and applicable in real-world settings. The proposed approach does not assume the visibility of skyline in images and tackles the problem of unreliable skylines by supplementing the skyline features with additional terrain features, such as mountain ridge-lines. It also uses camera roll angles as a search parameter and is therefore able to handle images with non-negligible camera roll angles.

The major contribution of this paper is an algorithm that is able to accurately geolocate real-world images with mountain views in large world regions. The proposed algorithm can handle missing, occluded, and unreliable skylines in query

Fig. 1: Examples of query images with occluded skylines. These query images (top row) have visible skylines considerably different from the synthetic views rendered from DEM using ground truth camera parameters (bottom row). Our proposed approach successfully geolocated these query images while approaches using only skylines (e.g., [2]) failed.

(a) roll: $-4.4°$ (b) roll: $-2.3°$ (c) roll: $-2.3°$

Fig. 2: Examples of query images that cannot be geolocated by the baseline approach in [2] due to non-negligible camera rolls, but were successfully located by the proposed approach.

images by employing a coarse-to-fine estimation mechanism. The proposed method extends the bag-of-curves model of [2] to obtain a coarse estimate of the camera position, heading, and roll angle using both skyline and mountain ridge-line features. The coarse estimation process considers geometric consistency of the matching ridge-lines and provides a ranked list of camera parameters (locations, heading, and roll angles) in a quantized space. These quantized parameters are then refined by first identifying stable image and DEM features from multiple terrain views, establishing the correspondences between image and terrain features, and exploiting them to search for the optimal camera position and orientation parameters. The paper demonstrates the effectiveness and accuracy of the proposed approach on real-world images from a variety of large geospatial regions around the world.

II. Related work

Many efficient and effective methods have been proposed in literature to solve the image geolocalization problem. In this section, we limit our discussion to only the approaches that use 3D reference data to geolocate images. These approaches can be characterized based on the type of scenes they handle (urban, mountainous, etc.) and the type of reference data they use (elevation maps, LIDAR, or image-based reconstructions). For example, various approaches have recently been proposed to geolocate images in large urban areas [11], [12], [17], [19] by using direct and indirect matching between local features from the 2D query images and points from the 3D models. These methods typically utilize 3D point cloud models obtained from large unstructured Internet photo collections [1]. All of these approaches require the 3D models that are built from ground images and have features that are compatible with query images. Therefore, such approaches are not applicable in mountainous regions that typically do not have good coverage of available ground imagery.

Baboud et al. [3] proposed an automatic photo-to-terrain alignment algorithm for annotation. This approach matches the image edges to silhouette edges extracted from synthetic panoramic views rendered from DEM. However, it requires prior knowledge of of the viewpoint location as well as the camera field-of-view (FOV).

Baatz et al. [2] proposed a novel grid-based method for large-scale mountain image geolocalization that does not require any prior knowledge of the camera location or camera field-of-view. They used skyline (i.e., the mountain outline) as the only feature and used a bag-of-curves model to represent the skyline features. Their algorithm matches the extracted skyline from image with synthetic skylines rendered from DEM on a uniform geospatial grid. Specifically, the image and synthetic skylines are broken down into small overlapping segments that are then uniformly sampled, normalized, and quantized. A 24-bit contour word is created by concatenating the quantized samples for each segment. The synthetic contour words are organized into an inverted index table, with the 24-bit integer as the key, and location and viewing direction as the content. The image contour words are then matched to the synthetic contour words and the matching scores are used to vote for the corresponding locations and directions. The approach takes geometric consistency into account in the voting process so the output list contains not only the matching location but also the heading direction. Since the method only uses skyline, only geometric consistency in the horizontal direction is considered. The resulting camera estimates fall on a regular grid, the resolution of which depends on the quantization of the geospatial region used for sampling the digital elevation model.

This grid-based approach has also been adapted by [9], [21] where different models were used to represent the visible skyline for image geolocalization in mountainous regions. Tzeng et al. [21] used concavity-based features to represent user-labelled query skylines and synthetic skylines. Whereas Hammoud et al. [9] used the Chamfer distance to compare the similarity between query skylines and synthetic mountain profiles. It is not clear whether these representations provide significant improvement over the bag-of-curves model.

All of these approaches use image skyline as their only feature. As mentioned in the previous section, such approaches become inapplicable if the skyline is not visible or reliable. The proposed approach is most closely related to Baatz et al. [2]. However, as opposed to [2], the proposed approach does not assume visibility of the skyline in image, rather it exploits all features and geometric consistency of all available internal ridge-lines visible in the image to overcome the issues that arise from occlusions and reliability of the feature extraction (Fig. 1). In addition to camera location and heading,

the proposed approach also estimates camera roll angle and therefore can reliably geolocate images with non-negligible camera roll angles (Fig. 2). Finally, instead of re-ranking the coarsely estimated cameras as in [2], the proposed refinement step optimally solves for the camera position and orientation parameters using feature correspondence between image and digital elevation model. Therefore, the resulting estimates are not restricted to quantized parameter space.

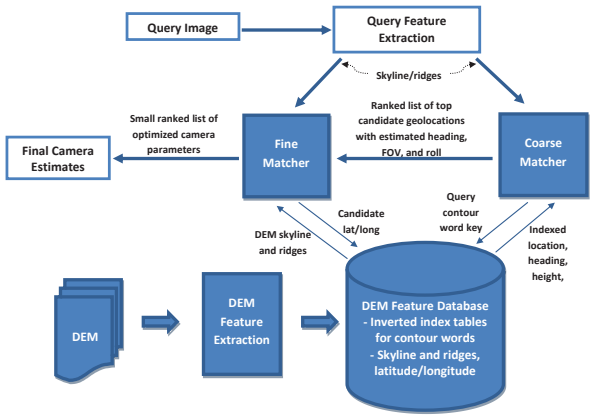


Fig. 3: Overview of the proposed approach

III. Proposed Approach

An overview of the proposed approach is shown in Fig. 3. Given a geospatial region of interest (ROI), an associated visual feature database is built by extracting and indexing skyline and ridge features from synthetic panoramic depth maps rendered using the corresponding DEM data. Given a query image from this ROI (e.g., Fig. 4(a)), the skylines and ridges are first extracted from the query image (Fig. 4(b)) and then fed to an coarse estimation module to obtain a ranked list of camera geolocation candidates in the quantized space (Figs. 4(c) and (d)). The results from coarse estimation are further refined to obtain a short list of final camera geolocations (Fig. 4(e)), which are used to render synthetic views of the query image (e.g., Fig. 4(f)) for visual inspection.

A. Building multi-ridge visual feature database

To build the multi-ridge visual feature database, we first render panoramic depth maps on a uniformly sampled grid within the ROI. In our experiments, the grid sample distance is 250m per sample. At each position on the grid, a panoramic depth map is generated by concatenating non-overlapping views rendered from the DEM data at this position, at an altitude of 1.6 meters relative to ground. The altitude parameter is chosen based on the assumption that the query image is taken by a person standing on ground.

Skylines and non-skyline mountain ridges are extracted from the synthetic depth maps based on depth discontinuity. Let $d_{r,c}$ be the depth value at rth row and cth column of the depth map. In each column in the depth map, the skyline pixel is taken as the topmost pixel with finite distance to the

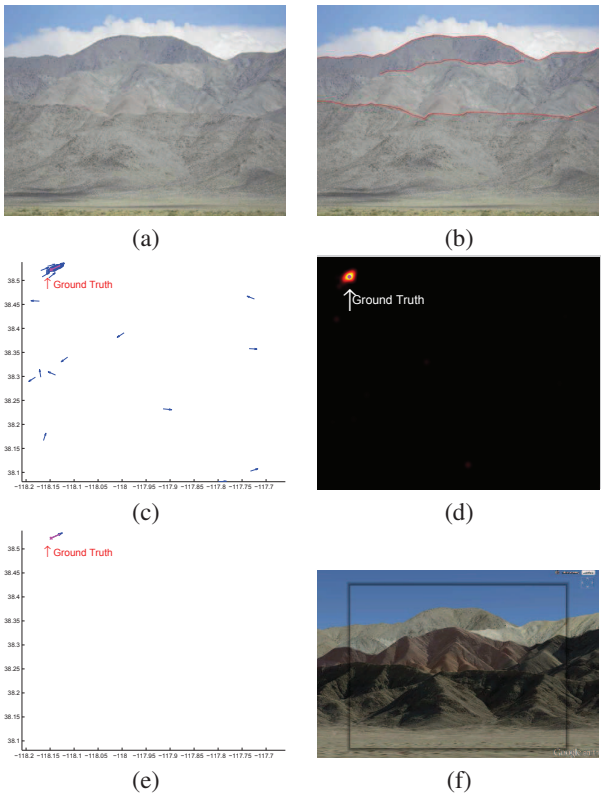


Fig. 4: An example of the system input and output: (a) query image, (b) extracted skyline and ridges, (c) coarse estimates of the camera locations and heading (blue arrows) and the ground truth location and headings (magenta arrow), (d) probability heat map generated from the coarse estimates, (e) final output: a small list of refined camera estimates (there is only one in the displayed region), and (f) a synthetic view rendered using the refined camera parameters. In (c)-(e), only a quarter of the geospatial region is shown for illustrative purposes.

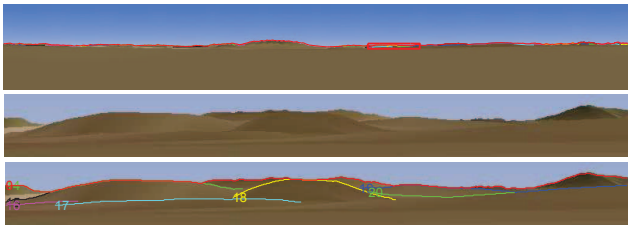


Fig. 5: An example of the synthetic panoramic image (top row), extracted skyline (in red) and multiple ridges (in colors other than red). The middle row shows details within the red block in the panoramic image. The bottom row shows the extracted ridges and their layer indices.

camera. Furthermore, a pixel at location (r,c) in the depth map is detected as a mountain ridge point if a) $d_{r,c} > D_{thld}$, b) the point is on a mountain slope with slope angle $> \theta_{thld}$, and c) the neighborhood depth ratio $\frac{d_{r-1,c}}{d_{r,c}} > r_{thld}$. The first two conditions locate pixels in the mountain regions in the depth map and the last one implies a depth discontinuity.

The ridge pixels are then connected by a tracing algorithm, discarding isolated points and short ridges. In our experiments, $D_{thld} = 500$m, $\theta_{thld} = 10^\circ$, and $r_{thld} = 1.4$.

The skyline is assigned a layer index of 0, and internal ridges are labelled with layer indices starting from 1 in a raster scan fashion from left to right and from top to bottom. An example of the extracted skyline and ridges overlaid on the synthetic panoramic image is shown in Fig. 5. The skyline and ridge pixels are then rectified to the normalized image plane to reduce the distortion around image boundaries. The latitude and longitude of each pixel on the skyline and ridges are calculated from the depth map and camera parameters. The skyline and ridge pixel coordinates and world coordinates are saved as binary files in the database indexed by the grid position. These features are used in the refinement process. The skyline and each of the ridges are handled individually to extract the contour words in the same way as in [2]. We use three different feature windows of width 2.5, 5, and 10 degrees. Fig. 6 illustrates the process of contour word generation. Fig. 6(a) shows the raw skyline (layer index 0) in a feature window of width $w = 2.5$ degrees. The raw segment is rectified and uniformly sampled (Fig. 6(b)). The mean of these samples are removed, and the samples are normalized by w (Fig. 6(c)). The normalized sampled are quantized into 8 disjoint bins (Fig. 6(d)). Finally, the 24-bit integer (contour word) representing this segment is calculated. In this example, the integer is calculated as $(2,3,2,2,4,5,5,5) = (010011010010100101101101) = 5056877$. An inverted index table is created using the 24-bit contour word and feature window width w as the key. The content of the table is the grid position, viewing direction, ridge layer index, and mean height of the segment.

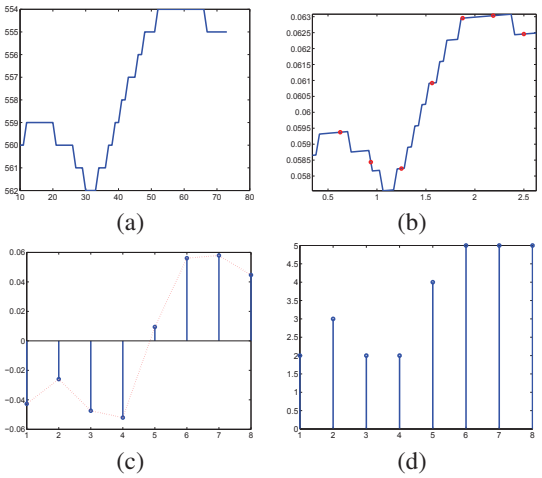


Fig. 6: Contour word extraction from a 2.5°-wide skyline segment : (a) raw skyline segment, (b) uniformly sample the rectified segment, (c) remove mean and normalize the samples, and (d) quantize the samples into 8 bins.

B. Coarse estimation from multiple ridges

The coarse estimation step takes the extracted skyline and ridges directly from the query image as input, and outputs a ranked list of N (e.g., $N = 1000$) camera estimates with the latitude/longitude coordinates on the DEM sampling grid along with an estimation of the camera heading, roll, and FOV. The camera location and heading direction are estimated from the voting process as follows.

Voting. The proposed approach is an extension of the original voting scheme in [2]. Instead of only checking geometric consistency horizontally for the heading direction as in [2], it also takes into account the consistency of relative positions between the query image and references in the vertical direction for the ridge layer ordering.

Specifically, the votes are calculated in four dimensions: the location specified by latitude/longitude coordinates, heading direction, and vertical offset on the normalized image plane. This involves a very large 4D array $\boldsymbol{A}$, but most of the entries are zero or have negligible small values, therefore sparse data structures can be adopted to improve the time and memory efficiency in the voting process. For each contour word from the query image with horizontal angle v_{image} and height h_{image}, we obtain relevant DEM information from the inverted index table in quadruples $(i, j, v_{\text{DEM}}, h_{\text{DEM}})$, where i and j are integers indicating the location on the DEM grid given by the latitude/longitude coordinates, v_{DEM} is the heading direction, and $h_{\text{DEM}})$ is the height. For each entry in the table, we calculate the horizontal offset between the viewing directions d_v and vertical offset between the heights d_h as follows.

$$d_v = v_{\text{image}} - v_{\text{DEM}}, \quad d_h = h_{\text{image}} - h_{\text{DEM}} \tag{1}$$

The offsets are then quantized into disjoint bins

$$b_v = \left\lfloor \frac{1}{q_v} \cdot \bmod \left(d_v + \frac{q_v}{2}, 360 \right) \right\rfloor, \quad b_h = [d_h / q_h] \tag{2}$$

where q_v and q_h are the quantization step sizes for the viewing direction and height bins, respectively, and $\lfloor \cdot \rfloor$ and $[\cdot]$ are the floor and rounding operators, respectively. The appropriate bin in the array $\boldsymbol{A}$ is then updated by the weighted vote:

$$\boldsymbol{A}(i, j, b_v, b_h) = \boldsymbol{A}(i, j, b_v, b_h) + w \tag{3}$$

where w is the tf-idf score of this word. After all image contour words have been processed, we suppress the 4D cube of votes into a 3D one by only keeping the highest votes in the vertical offset dimension:

$$\boldsymbol{A}(i, j, b_v) = \arg\max_{b_h} \boldsymbol{A}(i, j, b_v, b_h). \tag{4}$$

In this way, the proposed multiple-ridge algorithm does not require any accurate skyline and ridge order matching: the order is automatically taken care of by the height-offset bins. This is because the out-of-order matches usually result in large variations in height offsets, therefore the votes from these matches spread into multiple height-offset bins.

Handling camera tilt, FOV, and roll. The camera tilt angle is approximately handled by rotating the skyline vertically to adjust its position such that the mountains occupy the mid-upper part of the query image. The FOV is estimated by non-uniform sampling over typical FOV range (0-70 degrees) as in [2]. As mentioned in Section 1, camera roll also has a significant effect on the matching quality. To overcome the effect of camera roll, we also sample the camera roll over the range of -6 to 6 degrees. The range of roll angle was chosen based on the analysis of typical query images and the tradeoff on computation time and quality of matching.

Fig. 7 illustrates the effect of camera roll. The query image in Fig. 2(b) has a slight roll angle of around 2.3 degrees. If a zero roll angle is assumed (Fig. 7(a)), the query skyline is considerably different from the skyline obtained from synthetic image rendered using the ground truth camera parameters, especially at the curvelet level, leading to different contour words. Fig. 7(c) shows the voting scores obtained from the baseline approach in [2] of the 120 heading direction bins, centered at 0, 3, . . ., 357 degrees, at the ground truth position. We see that although the correct bin of 195 degrees has the highest voting score, the score is not prominent from the other bins. On the other hand, with a 2.3 degree rotation, the query and reference skylines almost perfectly overlap each other (Fig. 7(b)), and the bin at 195 degrees has a significantly higher score than the others (Fig. 7(d)).

Ranking. In the proposed coarse estimation process, both FOVs and rolls are sampled and the results are re-ranked altogether. However voting scores from different FOV assumptions are not compatible to each other. This is because a larger FOV usually leads to more contour words and thus higher scores. Therefore, instead of comparing the raw voting scores from different FOVs as in [2], we use normalized voting scores. The normalization factor is determined empirically by the mean value of the maximum score for a certain FOV across all roll angles. The normalized scores are then re-ranked across all possible locations, headings, FOVs, and rolls to generate the list of coarse camera estimates.

The effect of normalization can be illustrated by the following example. Fig. 8 shows the probability heat maps for the camera estimates of the query image in Fig. 4 before and after normalization. We see that before normalization the camera candidates with high scores spread all over the ROI. After normalization, the candidates concentrate within a small region centered at the ground truth location.

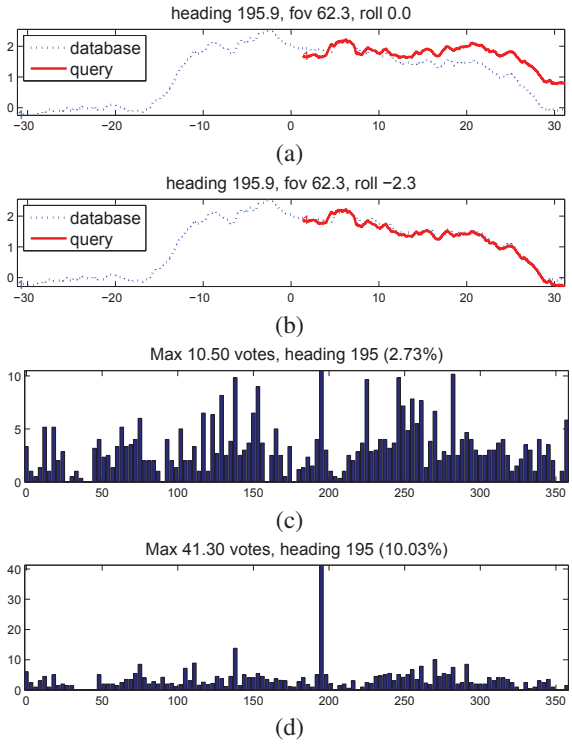


Fig. 7: Effect of camera roll. (a) Query skyline (red solid) directly labelled from image and partial database skyline (blue dashed). (b) Query skyline corrected by camera roll (-2.3 degrees) and database skyline. (c) and (d) Voting scores for the 120 direction bins at the ground truth position for the skylines in (a) and (b), respectively, obtained by the baseline skyline matching algorithm [2]. The y-axis in (a) and (b) is exaggerated to better illustrate the differences between image and synthetic skylines.

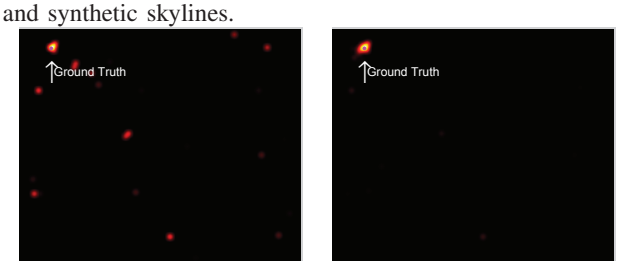


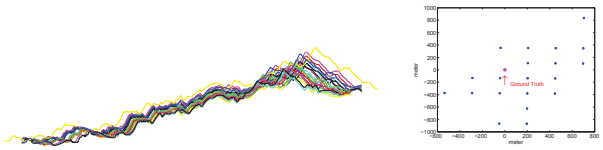
Fig. 8: Probability heat maps for camera estimates (left) before normalization and (right) after normalization.

C. *Refinement of estimated camera parameters*

The refinement process takes the list of coarsely-estimated camera candidates as input. The output of the refinement step is a small list of camera candidates (usually no greater than 20) with estimated parameters (latitude, longitude, altitude, heading, roll, tilt, and FOV). The final output of the refinement step is no longer restricted to a quantized space.

To obtain a pinpoint camera estimation, one of the most important issues is to establish correspondences between the world coordinates (latitude/longitude/altitude) and pixel coordinates ($[i, j]$). Directly obtaining the accurate correspondences from image features with a candidate rendered view is challenging due to the inaccuracies in DEM data, rendering resolution, and noise in image features, etc. However, it is observed that multiple matched rendered views within a small geospatial region can be used to identify stable features for alignment. Fig. 9(a) shows the 20 matching DEM skylines rendered within a 3 sq km region around the ground truth location from the coarse camera estimates. Fig. 9(b) shows the DEM

locations (blue) and ground truth location (magenta). One can observe that the matching skylines have a large variation in terms of the individual pixels, implying that it is unreliable to establish correspondences using a single matching skyline. However, there are stable and common features (such as the maxima of the curve) that can be utilized for camera parameter refinement. The proposed refinement method clusters camera candidates with similar views and exploits them in conjunction to identify stable features and to establish correspondences. A simplified pinhole camera model with intrinsic parameter of focal length, position (latitude/longitude/altitude), and orientation parameters (heading, tilt, and roll) is assumed. The DEM skylines/ridges/peaks mentioned in this section are referring to those extracted from rendered depth maps in Section III-A, but not directly from the DEM dataset.



(a) (b)

Fig. 9: 20 matching database skylines for the same query image rendered at locations around the camera ground truth.

Clustering of camera candidates. The coarsely-estimated camera candidates are first re-ranked by a 2D Iterative Closest Point (ICP)-like fine alignment step [4], [17]. This is similar to the geometric verification process in [2]. However we employ both skyline and ridges. Moreover, the alignment errors are normalized to alleviate the effects from different vertical FOVs and are calculated by

$$e_n = \frac{1}{N \cdot f_v^2} \sum_{k=1}^{N} \left(i_{\text{image},k} - i_{\text{DEM},k}\right)^2 + \left(j_{\text{image},k} - j_{\text{DEM},k}\right)^2 \quad (5)$$

where $\left(i_{\text{image},k}, j_{\text{image},k}\right)$ and $\left(i_{\text{DEM},k}, j_{\text{DEM},k}\right)$ are the coordinates of the kth image and DEM points in the aligned image plane, respectively, and f_v is the vertical FOV. The camera candidates are then re-ranked according to the alignment errors e_n and then grouped into several clusters, as shown in the example in Fig. 10. A camera is estimated for each cluster separately.

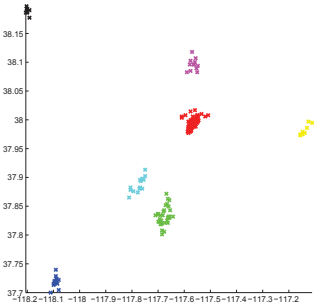


Fig. 10: Clustering the re-ranked camera estimates. Seven clusters are shown in different colors.

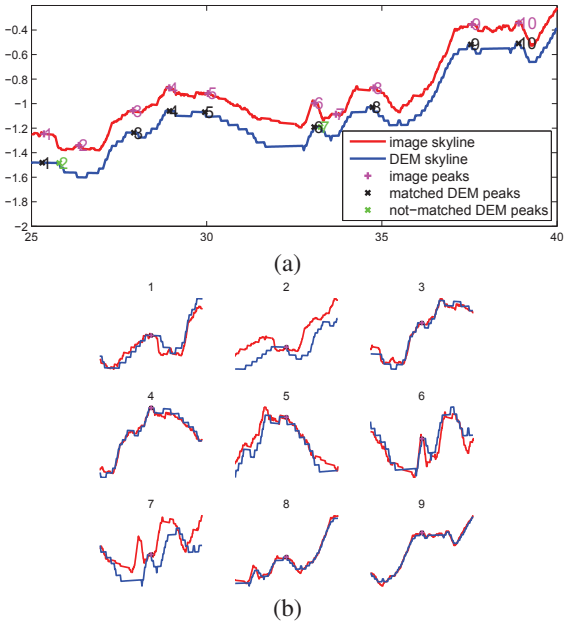
Establishing correspondences from a cluster of camera candidates. Given a cluster of camera estimates, we first get world coordinates of peaks on image skyline and ridges from all camera estimates in the cluster. The peaks are usually considered to be the most robust terrain features [15]. The pixel coordinates of image and DEM peaks are extracted as the local maxima of the skylines and ridges. The local windows centered the image and DEM peaks are then aligned. Fig. 11(a) shows peaks extracted from a partial query image skyline (red), roughly aligned with the DEM skyline (blue), and Fig. 11(b) displays the local windows centered at the corresponding peaks. It can be easily seen that the 2nd and 7th image peaks are not matched to any DEM peaks, while all others align very well locally.

Let $e_{i,j}$ denote the local alignment error between the ith image peak and the jth DEM peak, Coord_j be the world coordinates of the jth DEM peak, and S_i be the set of world coordinates associated with the ith image peak with initial value $\emptyset$, then S_i is updated by

$$S_i = S_i \bigcup \text{Coord}_j, \quad \text{if } e_{i,j} < \xi \quad (6)$$

where the threshold ξ is calculated by $\xi = 0.8 \times \text{median}\left\{e_{i,k}, \forall k\right\}$. After all DEM peaks from this cluster have been processed, the centroid of S_i is then used as the corresponding world coordinates for the ith image peak

$$\text{image peak}_i \Leftrightarrow \text{centroid}\left(S_i\right).$$



(a)

(b)

Fig. 11: (a) Matching query image skyline peaks to DEM skyline peaks (only partial skyline is shown for better clarity). (b) Aligned local windows. The 2nd and 7th peaks are not considered as matched.

In some cases, the peaks on the skyline and ridges are limited and may be insufficient for solving all unknown camera

parameters. Therefore, more corresponding features are often needed in addition to the peaks. We sample the image skyline and ridges and adopt a similar local approach to associate world coordinates obtained from DEM features to the sampled points in image.

The above method obtains world coordinates corresponding to pixels on image skylines and ridges. Therefore it cannot utilize informative world coordinates if the corresponding pixels are not labelled in the query image. To obtain the image pixel coordinates of distinctive world coordinates, we consider peaks extracted from DEM ridges. The DEM ridges containing such peaks are transformed to the image coordinates and then matched to the edge map of the query image derived from a compass edge detector [18]. A template matching approach [16] is applied to maximize the correlation between DEM ridge and image edges. Fig. 12 shows a matching pair of transformed DEM ridge and image edge. The world coordinates of the DEM ridge peak (blue circle) and the pixel coordinates of the matched image point (red circle) are then considered a pair of correspondence.

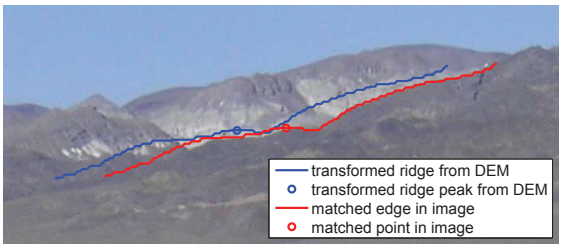


Fig. 12: Matching the transformed DEM ridge to image edge. The transformed DEM ridge has a small offset due to coarsely estimated camera parameters.

Optimization of camera parameters. The camera position is obtained by minimizing a weighted summation of errors between the image pixel coordinates and the projected world coordinates, with constraint on camera altitude being around 1.6 meters relative to ground. In optimization, the RANSAC algorithm [6] is used to handle outliers. Within each RANSAC iteration, we use the iteratively re-weighted least squares (IRSL) algorithm [5] which assigns higher weights for better inliers (i.e., with smaller alignment errors):

$$\hat{\boldsymbol{a}} = \arg\min_{\boldsymbol{a}} \sum_i w_i(\boldsymbol{a}) \left\| y_i - P_{\boldsymbol{a}}(x_i) \right\|^2 \tag{7}$$

where y_i are the pixel coordinates and x_i are the world coordinate, $P_{\boldsymbol{a}}$ is a function that projects the world coordinates to the pixel coordinates given camera parameter $\boldsymbol{a}$, w_i is the weight of the ith correspondence which is updated at each IRSL iteration by

$$w_i^{(t+1)} = \begin{cases} w_i^{(t)} / e_i^{(t)} & e_i^{(t)} \geq \varepsilon \\ w_i^{(t)} / \varepsilon & \text{otherwise,} \end{cases} \tag{8}$$

where $e_i^{(t)} = \left\| y_i - P_{\boldsymbol{a}}^{(t)}(x_i) \right\|$ is the error of the ith correspondence in the tth IRSL iteration, $\varepsilon = \min(1, 0.005 \times \max(W,H))$ is a small value used to bound the weights, and W and H are the width and height of the query image, respectively. The weights are then normalized by $w_i^{(t+1)} = w_i^{(t+1)} / \sum_k w_k^{(t+1)}$.

After all of the clusters have been processed, the camera candidates are then re-ranked based on number of inliers and the ratio between the number of inliers and correspondences.

IV. Experimental Results and Discussions

In this section, we demonstrate the effectiveness of the proposed algorithm on five 10,000 sq. km-sized ROIs (Fig. 13) located in North America, South America, and Asia. The ROI in North America is covered by USGS 10m-resolution NED (National Elevation Dataset) [7], [8]. The other four ROIs are covered by the 30m-resolution ASTER-GDEM (Advanced Spaceborne Thermal Emission and Reflection Radiometer-Global Digital Elevation Model) data [14].

Fig. 13: The five ROIs considered in this paper, denoted by the red polygons.

The algorithm is tested on query images from a real-world system where more than 40% of the query images suffer from adverse conditions including blurred or occluded skylines, and non-negligible camera roll angles. Some challenging examples are shown in Figs. 1 and 2. The exact camera ground truth locations (latitude/longitude) of 50 query images are known, and about 40% of these query images have known camera parameters. The skyline and ridges in the query images are labelled by a user through a semi-automatic interface. Though automated skyline detection algorithms can be used, the system has not yet been tested against the output of such algorithms. Fully automated skyline/ridge detection can be quite challenging in noisy images with occlusions, and is out of the scope of this paper. In our system, skyline/ridge extraction from the query image is a semi-automated process. The user selects the region of interest in the image which contains the mountain regions, and clicks on a few difficult or confusing points (e.g., points occluded by trees or partially overlapping with cloud boundaries). The system then automatically extracts the complete skyline/ridge by fitting a line along the detected edges in the region of interest and the user-provided points, while minimizing an energy function.

TABLE I: Performance of coarse estimation on 50 queries

	# geolocated	# in R_1
Multi-ridge with roll	42	35
Multi-ridge without roll	26	23
Skyline with roll	34	31
Skyline without roll	25	23

To quantify the coarse estimation performance, we use the measure of geolocation area (GA) and total area (TA). Specifically, a probability heat map is generated from the ranked list of camera estimates, and then the highly probable regions are selected and sorted according to the probability. Suppose there are K camera candidate regions $\{R_i\}_{i=1,2,...,K}$ each with area A_i km^2, sorted from the most probable to the least. If the camera ground truth is within the tth region R_t, then GA is calculated as $GA = \sum_{i=1}^{t} A_i$, which represents the search area a user will have to go through to geolocate the query image. The TA measure is the total retrieved area calculated by $TA = \sum_{i=1}^{K} A_i$, which implies the precision. A query image is considered as "geolocated" if the ground truth is within any of the K camera candidate regions. The GA and TA curves show the percentage of query images geolocated within certain GA/TA thresholds.

We first examine the performance of coarse estimation. Fig. 14 shows the GA and TA curves using three algorithms with DEM resolution 250 meters. The algorithm "skyline without roll" is a direct implementation of the coarse matcher in [2]. The proposed coarse estimation algorithm ("multi-ridge with roll") found around 60% of all query images within a small region of 10 km^2 (0.1% of the considered ROI with a total area of 10,000 km^2) and outperforms the algorithms based purely on the skyline by geolocating 23% more queries, demonstrating the effectiveness of exploiting non-skyline ridges. Moreover, the incorporation of roll angles also helps to significantly improve the geolocation ability, as seen in the performance curves of the two skyline-only algorithms, "skyline without roll" in black and "skyline with roll" in green in Fig. 14. These two algorithms do not involve the height offsets in the voting process.

Table I shows the total number of geolocated images and the number of images located in the first candidate region R_1. With the proposed algorithm, a total of 42 query images have been geolocated, and 35 out of the 42 have ground truth inside R_1. Fig. 15 shows the probability heat maps of the coarse camera estimates using the proposed approach (multi-ridge with roll, top row), skyline only with roll (middle row), and the baseline algorithm (skyline only without roll). These probability heat maps were obtained for four query images, one query for each column in Fig. 15. The first two columns are heat maps for the top two images in Fig. 1 exhibiting unreliable skylines and the last two columns for Figs. 2(a) and (b) exhibiting non-negligible camera rolls. We can see that exploiting both multiple ridges and camera roll estimation leads to significant performance improvement. The coarse estimates from our proposed algorithm concentrate at a small region centered at the ground truth camera location with a high probability.

It is clear from the coarse estimation results that the

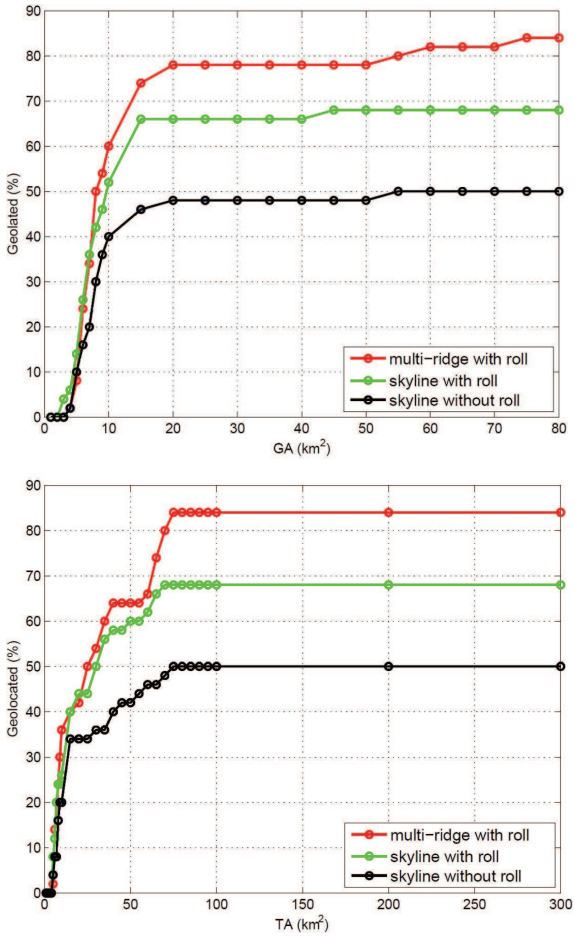


Fig. 14: Percentage of query images geolocated within GA (top) and TA (bottom) thresholds obtained using different coarse algorithms.

proposed method based on multi-ridge matching and camera roll angle estimation significantly outperforms the baseline algorithm in [2]. A close examination reveals that the baseline algorithm achieves lower performances compared to those reported in [2], which is mainly due to the difference in training and query data. In our experiments, more than 40% of the query images have artifacts, which present significant challenge to the baseline approach. Furthermore, the ground sampling resolutions of the DEM data used in our experiment are 10m or 30m per sample, which are much lower than the resolution of the DEM data used in [2] (one sample per 2 square meters). We believe using DEM data with higher resolution would benefit both baseline and proposed algorithms. In addition, the sample distance of the grid used for building visual feature database used in our experiment is 250m, which is over twice as large as that in [2]. Such increase in grid sample distance may also contribute to the deteriorated performance of the baseline algorithm on our dataset. In our research, we experimented with multiple values for the grid sample distance from 125m to 2000m. Fig. 16 shows the GA and TA curves for a subset of 30 query images

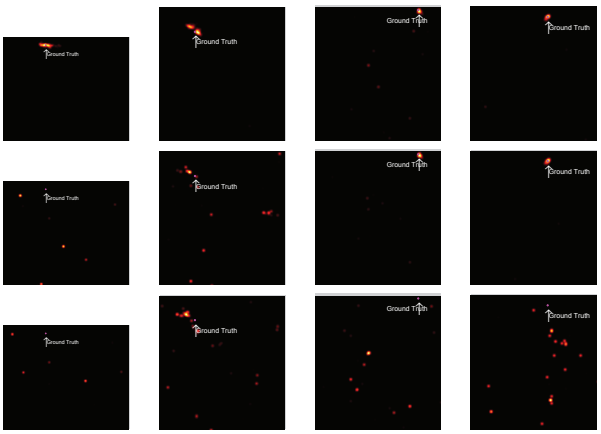


Fig. 15: Probability heat maps on four queries images, one for each column, obtained from three coarse estimation algorithms: multi-ridge with roll (top row), skyline with roll (middle row), and skyline without roll (bottom row)

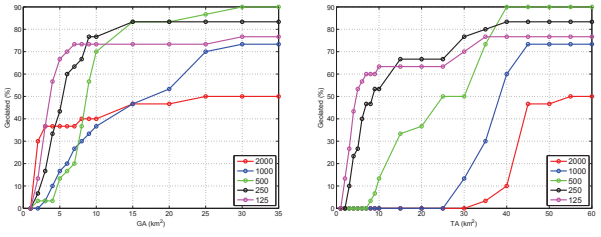


Fig. 16: Percentages of queries geolocated within GA (left) and TA (right) thresholds using different ground sampling distance when the visual feature database was built

with grid sampling distances ranging from 125m to 2000m. It can be observed from Fig. 16 that the performance, especially the precision, generally improves with finer sampling grid. The 250m grid sampling distance was selected as a tradeoff between the desired accuracy and affordable computational complexity for both database building and query processing.

We further examine the performance of the refinement process. Note that the refinement step depends on the coarse camera estimates, therefore we only consider the 42 query images that have been geolocated. The measures are the total number of output camera estimates, distance from the estimated camera position to the ground truth, and heading offset. Table II shows the minimal, maximal, and median values of these measures on the query images. Among these 42 query images, 34 have the best estimate (in terms of the distance to ground truth) as the first candidate, and 6 others as the second. This indicates that the user only need to go through a very small number of candidates to justify the results. Fig. 17 shows the percentage of query images that are located within certain distances to the ground truth. We see that around 80% are within 1000 meters. It is worth noting that the heading estimation is very accurate (median offset < 0.5 degrees) and the ground truth camera position is usually along the heading direction of the estimated camera.

TABLE II: Performance of refinement on 42 query images.

	min	max	median
number of output camera estimates	1	10	6
distance (m)	8.92	5750.22	341.64
heading offset[a] (°)	0.001	10.402	0.270

[a]computed using ground truth headings from 13 queries

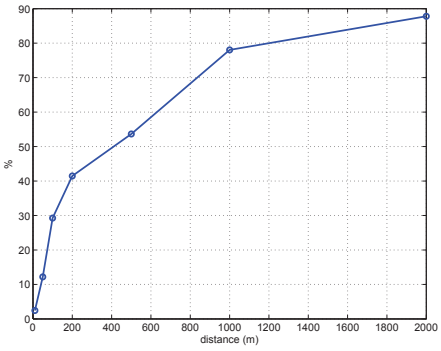


Fig. 17: Percentage of query images located within certain distances.

Finally, we visually inspect the geolocalization performance by comparing the query images against synthetic views rendered through Google Earth using the optimized camera parameters. Fig. 18 shows examples of query images and the corresponding synthetic views. For these query images, the distance from the estimated camera location to the ground truth ranges from 188.17m to 372.88m. It can be seen that the synthetic views are nearly identical to the query images.

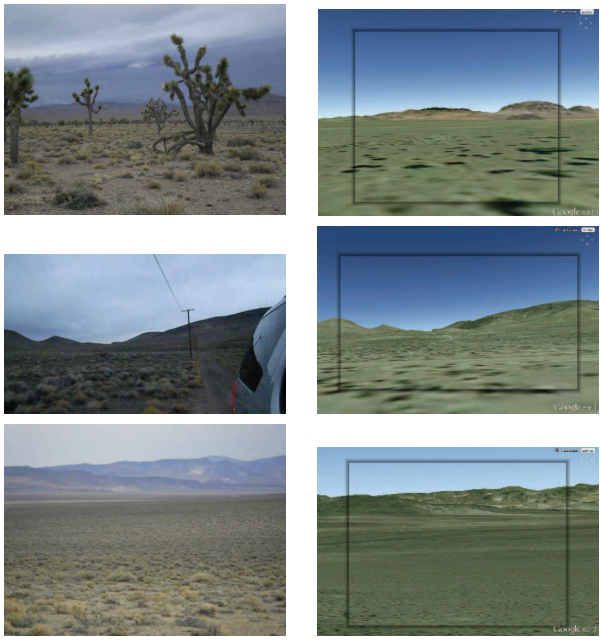


Fig. 18: Left column: query images. Right column: synthetic views rendered in Google Earth using the optimized camera parameters. The distances from the estimated camera locations to the ground truth are (from top down): 372.88m, 368.52m, and 188.17m.

Our system implementation is written in a combination of Matlab, Java, and C/C++ languages. It takes several seconds to one minute for the voting stage in the coarse estimation process per roll per FOV, and one to ten minutes to optimize the camera parameters in refinement process. It is more computationally intensive than the baseline approach (the authors reported 10 seconds per image in [2], and our own implementation takes seconds to tens of seconds per roll per FOV). The processing efficiency can be significantly improved with software optimization and parallel processing.

V. Conclusions and Future Work

Query images with poor skylines or non-negligible camera rolls present a significant challenge to mount view-based image geolocation. This paper presents a novel camera geolocation approach capable of geolocating such challenging query images by using additional multi-ridge features and performing camera roll estimation. Experimental results on real-world query images demonstrate the efficacy of our proposed approach. It is shown that using visual features extracted from multiple mountain ridges is critical to reliably geolocating challenging real-world query images with blurred or invisible mountain skylines, and that estimating camera roll angles is the key to handle query images with non-negligible camera rolls. In our future work, we will adopt an adaptive grid sampling scheme [24] for visual feature database construction. Adaptive grid sampling leads to improved computational efficiency by allowing the sampling density to adjust adaptively to the terrain complexity in the ROI. Furthermore, we will extend our current approach to handle query videos, e.g., by exploiting the 3D scene geometry recovered from the video using structure from motion techniques [1].

VI. Acknowledgments

This work is supported by the Intelligence Advanced Research Projects Activity (IARPA) via Air Force Research Laboratory (AFRL), contract FA8650-12-C-7212. The U.S. Government is authorized to reproduce and distribute reprints for Governmental purposes notwithstanding any copyright annotation thereon. Disclaimer: The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of IARPA, AFRL, or the U.S. Government.

References

[1] S. Agarwal, N. Snavely, I. Simon, S. M. Seitz, and R. Szeliski, "Building Rome in a day," in *Proc. of IEEE International Conference on Computer Vision*, Sep. 2009, pp. 72–79.

[2] G. Baatz, O. Saurer, K. Köser, and M. Pollefeys, "Large scale visual geo-localization of images in mountainous terrain," in *Proc. 12th European Conference on Computer Vision, Part II*, Florence, Italy, Oct. 2012, pp. 517–530.

[3] L. Baboud, M. Cadik, E. Eisemann, and H.-P. Seidel, "Automatic photo-to-terrain alignment for the annotation of mountain pictures," in *Proc. of IEEE Conference on Computer Vision and Pattern Recognition*, Providence, RI, Jun. 2011, pp. 41–48.

[4] P. J. Besl and N. McKay, "A method for registration of 3-d shapes," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 14, no. 2, pp. 239–256, Feb. 1992.

[5] I. Daubechies, R. DeVore, M. Fornasier, and C. S. Güntür, "Iteratively reweighted least squares minimization for sparse recovery," *Communications on Pure and Applied Mathematics*, vol. 63, no. 1, pp. 1–38, Jan. 2010.

[6] M. A. Fischler and R. C. Bolles, "Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography," *Communications of the ACM*, vol. 24, no. 6, pp. 381–395, Jun. 1981.

[7] D. Gesch, M. Oimoen, S. Greenlee, C. Nelson, M. Steuck, and D. Tyler, "The national elevation dataset," *Photogrammetric Engineering and Remote Sensing*, vol. 68, no. 1, pp. 5–11, 2002.

[8] D. B. Gesch, "The national elevation dataset," in *Digital Elevation Model Technologies and Applications: The DEM Users Manual*, 2nd ed., D. Maune, Ed. Bethesda MD: American Society for Photogrammetry and Remote Sensing, 2007.

[9] R. I. Hammoud, S. A. Kuzdeba, B. Berard, V. Tom, R. Ivey, R. Bostwick, J. HandUber, L. Vinciguerra, N. Shnidman, and B. Smiley, "Overhead-based image and video geo-localization framework," in *Proc. of IEEE Conference on Computer Vision and Pattern Recognition Workshops*, Jun. 2013, pp. 320–327.

[10] J. Hays and A. Efros, "IM2GPS: estimating geographic information from a single image," in *Proc. of IEEE Conference on Computer Vision and Pattern Recognition*, Anchorage, AK, Jun. 2008, pp. 1–8.

[11] A. Irschara, C. Zach, J.-M. Frahm, and H. Bischof, "From structure-from-motion point clouds to fast location recognition," in *Proc. of IEEE Conference on Computer Vision and Pattern Recognition*, Jun. 2009, pp. 2599–2606.

[12] Y. Li, N. Snavely, and D. P. Huttenlocher, "Location recognition using prioritized feature matching," in *Proc. 11th European Conference on Computer Vision, Part II*, Heraklion, Crete, Greece, Sep. 2010, pp. 791–804.

[13] T.-Y. Lin, S. Belongie, and J. Hays, "Cross-view image geolocalization," in *Proc. of IEEE Conference on Computer Vision and Pattern Recognition*, Portland, OR, Jun. 2013, pp. 891–898.

[14] NASA Land Processes Distributed Active Archive Center (LP DAAC), "ASTER L1B," Sioux Falls, SD, 2001, sGS/Earth Resources Observation and Science (EROS) Center.

[15] P. C. Naval Jr., M. Mukunoki, M. Minoh, and K. Ikeda, "Estimating camera position and orientation from geographical map and mountain image," in *38th Research Meeting of the Pattern Sensing Group, Society of Instrument and Control Engineers*, 1997, pp. 9–16.

[16] W. K. Pratt, *Digital Image Processing: PIKS Scientific inside*, 4th ed. Hoboken, NJ: John Wiley & Sons, Inc., 2007.

[17] B. C. Russell, J. Sivic, J. Ponce, and H. Dessales, "Automatic alignment of paintings and photographs depicting a 3D scene," in *3rd International IEEE Workshop on 3D Representation for Recognition (3dRR-11), associated with ICCV*, Nov. 2011.

[18] M. A. Ruzon and C. Tomasi, "Edge, junction, and corner detection using color distributions," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 23, no. 11, pp. 1281–1295, Nov. 2001.

[19] T. Sattler, B. Leibe, and L. Kobbelt, "Fast image-based localization using direct 2D-to-3D matching," in *Proc. of IEEE International Conference on Computer Vision*, Nov. 2011, pp. 667–674.

[20] W. B. Thompson, T. C. Henderson, T. L. Colvin, L. B. Dick, and C. M. Valiquette, "Vision-based localization," in *Proceedings of the 1993 Image Understanding Workshop*, 1993.

[21] E. Tzeng, A. Zhai, M. Clements, R. Townshend, and A. Zakhor, "User-driven geolocation of untagged desert imagery using digital elevation models," in *Proc. of IEEE Conference on Computer Vision and Pattern Recognition Workshops*, Jun. 2013, pp. 237–244.

[22] A. R. Zamir and M. Shah, "Accurate image localization based on Google maps street view," in *Proc. 11th European Conference on Computer Vision, Part IV*, Heraklion, Crete, Greece, Sep. 2010, pp. 255–268.

[23] Y. Zheng, M. Zhao, Y. Song, H. Adam, U. Buddemeier, A. Bissacco, F. Brucher, T. Chua, H. Neven, and J. Yagnik, "Tour the world: building a web-scale landmark recognition engine," in *Proc. of IEEE Conference on Computer Vision and Pattern Recognition*, Miami, FL, Jun. 2009, pp. 1085–1092.

[24] J. Zhu, M. Bansal, N. V. Valk, and H. Cheng, "Adaptive rendering for large-scale skyline characterization and matching," in *Proc. 12th European Conference on Computer Vision Workshops and Demonstrations, Part I*, Florence, Italy, Oct. 2012, pp. 163–174.