GPS Refinement and Camera Orientation Estimation from a Single Image and a 2D Map

Hang Chu, Andrew Gallagher, Tsuhan Chen
School of Electrical and Computer Engineering, Cornell University
hc772@cornell.edu, andrew.c.gallagher@gmail.com, tsuhan@ece.cornell.edu

Abstract

A framework is presented for refining GPS location and estimate the camera orientation using a single urban building image, a 2D city map with building outlines, given a noisy GPS location. We propose to use tilt-invariant vertical building corner edges extracted from the building image. A location-orientation hypothesis, which we call an LOH, is a proposed map location from which an image of building corners would occur at the observed positions of corner edges in the photo. The noisy GPS location is refined and orientation is estimated using the computed LOHs. Experiments show the framework improves GPS accuracy significantly, generally produces reliable orientation estimation, and is computationally efficient.

1. Introduction

Urban localization and navigation have become an important application for many mobile phones. To accomplish that, many smartphones have an embedded GPS (Global Positioning System) receiver. However, there are several deficiencies with this localization approach. First, GPS is prone to inaccuracy in several situations, including the urban canyons between buildings in cities. The outdoor accuracy of mobile phone GPS is only 12.5 meters [17]. Secondly, GPS does not indicate the direction that the user is facing, even if perfect localization was achieved.

We address these two problems of GPS by an image-based localization approach. In our approach, we first ask the user to take an image of any nearby building. Then we localize the camera position and solve for the camera orientation, with the layout and structure of buildings from a simple 2D plan view city map. 2D maps are widely available in many cities, generally well maintained and updated, and incorporate sufficient building structural information for our task. In summary, the goal of our work is to *refine the position of a cameras GPS location and estimate the camera pose, based on detecting vertical corner edges from a single cuboid building image and matching these to a 2D city map with building outlines*. Our method does not require the overhead of computing or storing appearance descriptors on buildings or image patches. Instead, we find edges in the image that are likely to exhibit themselves as building corners on a 2D map with building outlines. Figure 1 illustrates the inputs and outputs of our system.

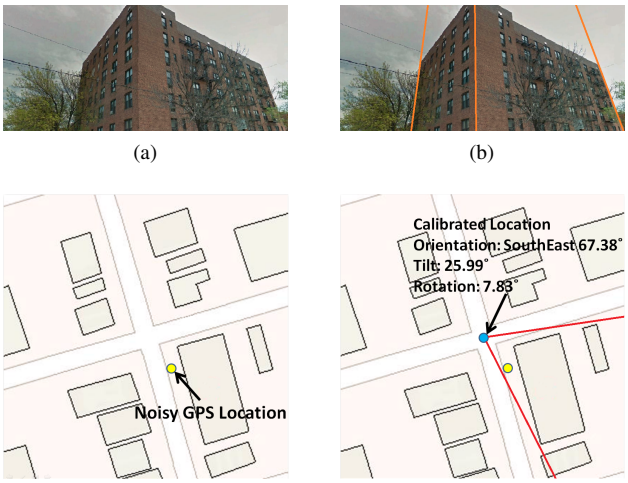


Figure 1: (a) The input building image. (b) Identified vertical corner edges of the building. (c) The 2D region map with building outlines and the noisy GPS location. (d) The calibrated location and estimated camera orientation.

We conduct experiments on 263 street images collected from 11 unique locations. The results show that our framework is able to accurately perform the task and improve GPS accuracy significantly, suggesting potential applications for mobile localization for tourists. We also test our framework on a dataset of images of apartment buildings.

The contributions of this paper are:

1. A framework for refining GPS location and finding camera orientation with a 2D map and a

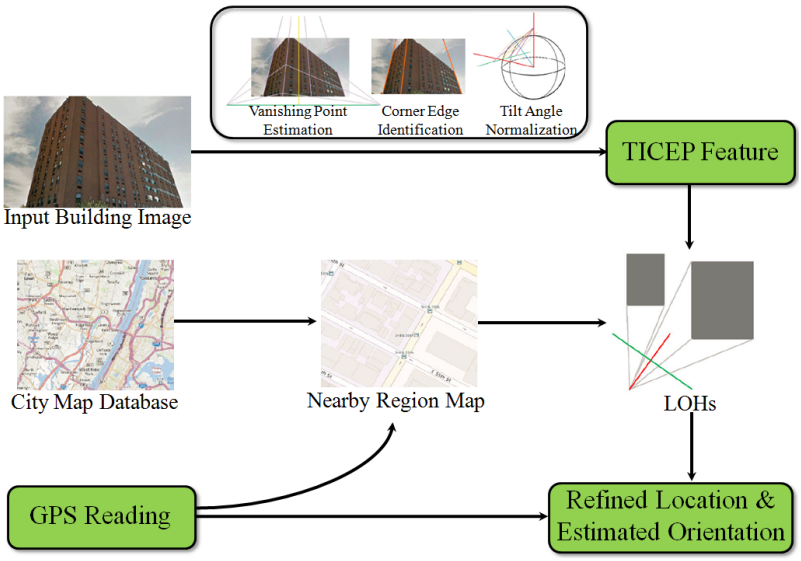


Figure 2: An overview of our framework.

single image.

2. The Tilt-Invariant Corner Edge Position (TICEP) feature that is extracted from building images and is useful in computing high accuracy locations.

3. A method for finding Location Orientation Hypotheses (LOHs) that represent possible solutions for camera location and orientation by finding geometric correspondences between corners on the 2D map and extracted TICEP features. From these LOHs, one is selected based on proximity to the initial GPS estimate as the final result.

2. Related work

Location-related research has long been an important topic in the computer vision community, perhaps beginning with a challenge to the vision community in 2005 [14]. Hays et al. [9] describes estimating GPS locations from images, using nearest neighbour matching of low- and mid- level appearance features to a large geo-tagged image database. In contrast, our work addresses refining the geoposition of an image that has an initial GPS estimate.

Another approach is based on the structure-from-motion (SfM) framework, for instance the method Li et al. proposed in [10]. Although their method shows significant accuracy, it requires a huge amount of images and computational resources, essentially requiring that the recognition database be stored in the cloud. Further, there exist difficulties in keeping the reconstructed model up-to-date. In contrast, our approach does not require a database of feature appearances. In a way, our approach can be seen as an extremely simplified variant of SfM, where we put more value on efficiency because the method is seeded with a GPS positional estimate.

In Babound et al. [2], camera pose estimation is performed by finding matches between mountain outlines and a terrain map. In the work of Schindler et al. [13], image lines are used as a geometric feature to construct building models. The success of these methods suggest the effectiveness of interpreting the environment in view of simple structural lines, which inspires us to use vertical corner edges to represent building structures in images.

Park et al. [11] proposed a method of estimating location and orientation of camera by matching the ground view image with a satellite image. Their work can be seen as an intermediate between feature-based matching and symbolic map matching. The work of Ramalingam et al. [12] can be also categorized similarly as in their framework an omni-skyline image is used in an analogous way as the satellite image of [11].

Other relevant work include the work of Chen et al. [5] and Baatz et al. [1]. Though their works are quite different from ours as they mainly considered image features, they show the effectiveness of using vanishing points and rectification for location recognition in urban area.

The most relevant work is reported in [4] by Cham et al., where vertical and horizontal building edges are extracted from an omnidirectional image comprised of four directional images. From these edges, structural fragments describing a piecewise linear contour of the building are produced and used for searching in a region of 2D map. This inspires us to take a further step: refine the GPS position by

discovering structures from a single image and referring to a map. The framework in [4] cannot solve the GPS refinement problem well. First, in [4] the camera tilt (elevation) is not estimated and considered in the localization process. This is fine with large scale block searching and rough localization tasks as shown in [4], but it is problematic to achieve a precision higher than GPS. Second, four directional images are required in their method, which causes more user operation. In our method, we solve the tilt problem by using the TICEP feature, and we need only one image.

3. Approach

Our algorithm takes a single building image, a 2D city map with building contours, and a noisy GPS reading as inputs. For the building image, we extract Tilt-Invariant Corner Edge Position (TICEP) features by sequentially applying vanish point estimation, corner edge identification, and tilt angle normalization. Next, we retrieve the nearby region map of the GPS reading from the whole city map database. Using the nearby region map and computed TICEP features, we determine multiple Location Orientation Hypotheses (LOHs) in the nearby region. Essentially, an LOH is a geographic position and orientation from which a camera could capture a nearby building that will have corners aligning with the observed edges in the image. Finally, the GPS location is refined using the LOHs, and the orientation of the camera is estimated. Figure 2 shows an overview.

3.1. Computing TICEP features

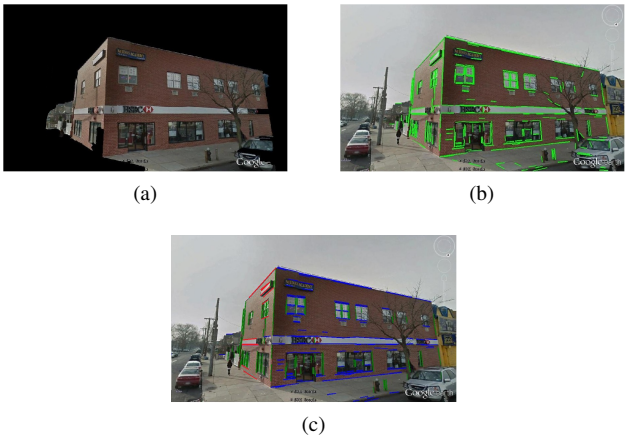
The procedure of computing TICEP features can be divided into three stages: estimating vanishing points, identifying vertical building corner edges, and normalizing the tilt angle.

We first introduce several notations that will be used. For vanishing points (*vp*), we denote the vertical *vp* and the i^{th} horizontal *vp* by $\boldsymbol{v}_v$ and $\boldsymbol{v}_i$. As we are particularly interested in a single Manhattan building, we expect two horizontal vanishing points [6]. We use l_v and l_i to denote line segments labeled to v_v and v_i. I_{ij} is defined to be the set of all intersection points of the extensions of any pair of lines taken from two different horizontal *vp*-labeled line segment sets l_i and l_j, $i \neq j$. S_i denotes one image segment.

3.1.1 Estimating vanishing points

Vanishing points are used for detecting the corner lines, and for estimating the camera focal length. To detect vanishing points of the input image, we perform the following processing steps.

Image segmentation: As in our intended application, we ask the user to take an image of any nearby building, it is reasonable to assume that the building is generally centered within the frame of the image. To reduce the occurrence of false vanishing point detections, we first perform a segmentation with the intent of removing non-building segments from the image periphery. To do this, we perform a standard marker controlled watershed segmentation and select segments near the image center, Figure 3a shows an example.



(a) (b) (c)

Figure 3: (a) An example of selected central segments. (b) An example of detected line segments. (c) An example of *vp* detection and *vp*-labeled line segments, different colors indicate different labels.

Line segment detection (LSD): We use the algorithm introduced by Gioi et al. [16] to detect line segments, denoted by l. N_l denotes total number of line segments. Figure 3b shows an example of LSD.

Vanishing point estimation: We adopt the method in [15] and [18], which are based on the J-Linkage model. Experiments show that this method is highly efficient and accurate for man-made environments [15], which are exactly the properties we want for our algorithm. Figure 3c shows an example of estimated vanishing points for an image.

3.1.2 Identifying vertical building corner edges

We seek lines in the image that correspond to building corners that will correspond to corners on the building footprint of the 2D map, rather than merely co-planar vertical lines of two facades of the same building.

The identification of building boundary corner edges is now described. We expect that boundary corner edges typically exhibit a large gradient in the horizontal direction, and have different colors (building color and background color) on either side of the boundary. To find these boundary corner edges, we first rectify the input image vertically using the estimated *vp*s, and then we detect image columns that are likely to align with boundary corner edges using a score computed as

Figure 4: An example of identified vertical corner edges, blue indicated boundary corner edges, red indicates an intersecting corner edge, and green stands for the horizon.

$$Scr_{BCE}(j) = Scr_c(j) + Scr_p(j) + Scr_l(j) \quad (1)$$

where j is the column coordinate in the vertically rectified trimmed image, Scr_c, Scr_p, and Scr_l are respectively scores of the horizontal pixel color gradient, the number of pixels that change segment label horizontally, and an indication of whether the column contains no horizontal lines and a neighboring column does contain at least one horizontal line. For computing Scr_l, the set of horizontal lines is selected from all horizontal *vp*-labeled line sets that exceed 100 pixels in length. After boundary corner edges are identified, we classify them into left boundary corner edges and right boundary corner edges from the consistency of color, column pixel total number, and coverage of long horizontal lines.

Intersecting corner edges are identified after boundary corner edges. Inspired by [4], we define intersecting corner edges as a vertical *vp*-labeled line segment that intersects with horizontal line segments belonging to 2 different *vp*s. In addition, we require that only one intersecting corner edge exists between a left boundary corner edge and a right boundary corner edge. The longest line is selected if more than one candidate is found. Figure 4 shows an example of identified vertical corner edges. We then compute the corner positions as the intersections of identified corner edge and the horizon (purple circles in Figure 4). Denote the i^{th} corner edge and corner edge position as $\boldsymbol{e}_i$ and $\boldsymbol{p}_i$, we have $\boldsymbol{p}_i = \boldsymbol{e}_i \times (\boldsymbol{v}_1 \times \boldsymbol{v}_2), i = 1, , N_p$.

It should be noticed that as real world images are more complex than ideal examples, sometimes our algorithm fails to identify all the corner edges correctly. However, with the vanishing points that can be estimated by our algorithm accurately, the user can manually correct an identification result by a single tap. We also show in Section 4 that our algorithm is able to identify corner edges with 85.73% accuracy, which means that our algorithm can generally give satisfactory identification results automatically and large amount of user effort can be saved.

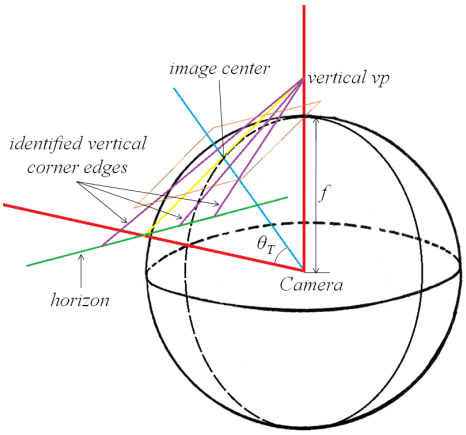


Figure 5: Diagram of computing camera tilt angle (with rotation angle rectified).

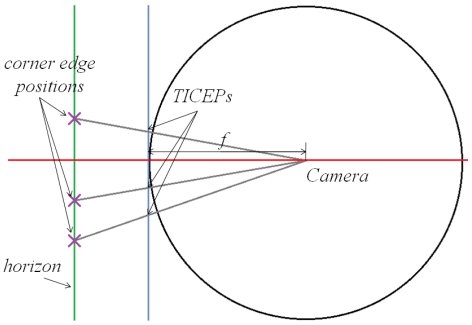


Figure 6: Diagram of normalizing the tilt angle. This is Figure 5 looking from top to bottom.

3.1.3 Normalizing the tilt angle

The computed corner edge positions of a image are variant to the camera tilt angle. We now describe the normalization of tilt angle. The first step is to estimate several camera parameters: rotation angle θ_R, focal length f, and tilt angle θ_T.

We first compute the rotation angle using a procedure similar to the method in [8]. Then we rectify the coordinates of all *vp*s and intersecting points according to the computed rotation angle, so that the vertical *vp* $\boldsymbol{v}_v$ now lies on the images *y*-axis, and the two horizontal *vp*s $\boldsymbol{v}_1$ and $\boldsymbol{v}_2$ lie on a line parallel to the *x*-axis. Also all the intersecting points now have the same *y*-coordinate.

After we neutralize the camera rotation, the focal length can be easily obtained as [3]

$$f = \sqrt{-v_{vy} v_{1y}} \quad (2)$$

It should be noted that when image has no tilt we have $v_{vy} = -\infty$, the focal length cannot be computed and must

be obtained from the image format file. As shown in Figure 5, the tilt angle of the camera can be computed as

$$tan(\theta_T) = \begin{cases} \frac{h-\sqrt{h^2-4f^2}}{2f} & \text{when } |v_{vy}| > |v_{1y}| \\ \frac{h+\sqrt{h^2-4f^2}}{2f} & \text{when } |v_{vy}| \leq |v_{1y}| \\ 0 & \text{when } v_{vy} = -\infty \end{cases} \quad (3)$$

where

$$h = |v_{vy}| + |v_{1y}| \quad (4)$$

As shown in Figure 5, the edge positions p_i are affected by the tilt angle, thus we normalize the tilt angle and compute the TICEP as

$$TICEP_i = p_{ix} \cos(\theta_T) \quad (5)$$

Figure 6 shows a diagram for the tilt angle normalization. We only care about the horizontal coordinates of the intersecting points as they are sufficient to demonstrate the distribution of corner edges.

Now that we have finished all procedures related to the building image, and an image is represented as the focal length and TICEPs, i.e., $Image = \{f, \boldsymbol{TICEP}\}$.

Except for the orientation angle θ_O, all the parameters of camera pose have been estimated. θ_O will be estimated in Section 3.2 together with the location.

3.2. Refining GPS by LOH

For the entire city map, we first extract the region map as the *200m-by-200m* square region centered at the noisy GPS location. The range of the region map is generally far larger than the noise of GPS so that the correct location is included in the region map with high confidence.

We now describe the computing of an LOH. Assume we know the correspondence of the computed $\boldsymbol{TICEP} = \{TICEP_i\}$ to the corners in the map. The locations of the corresponding map corners are $\boldsymbol{c}_h = \{\boldsymbol{c}_{hi}\}$. An LOH is defined as the particular location and orientation in the map, from where the corners in c_h can be seen in the way that $\boldsymbol{TICEP}$ are distributed. To describe an LOH, its location and orientation are needed: $LOH = (\boldsymbol{x}_{LOH}, \boldsymbol{n}_{LOH})$, where $\boldsymbol{x}_{LOH}$ is the position on the map, and $\boldsymbol{n}_{LOH}$ is a normalized orientation vector.

To compute the parameters of an LOH associated with corners of a building footprint, we minimize the total deviation between the positions where corners would be seen in the image plane from the view of a potential LOH and TICEPs, i.e., the summation of distances between the intersection of LOH-corner line and image plane and location of the corresponding corner edges on the image plane:

$$(\boldsymbol{x}_{LOH}, \boldsymbol{n}_{LOH}) = \underset{(\boldsymbol{x},\boldsymbol{n})}{\arg\min} \sum_{1 \leq i \leq N_p} ||\boldsymbol{q}_i - \boldsymbol{inter}_i||^2 \quad (6)$$

where

$$\boldsymbol{inter}_i = (\boldsymbol{x} \times \boldsymbol{c}_{hi}) \times (\boldsymbol{q}_1 \times \boldsymbol{q}_2) \quad (7)$$

$$\boldsymbol{q}_i = \boldsymbol{x} + f\boldsymbol{n} + TICEP_i \boldsymbol{n}_\perp \quad (8)$$

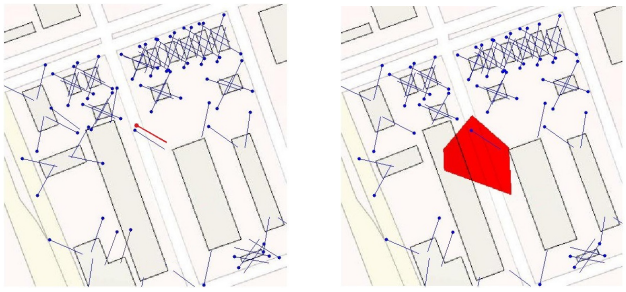
The minimization problem is not linear. As one LOH has three degree of freedom, when $N_p = 3$ a precise multinomial approximation can be found for the two coordinates and one orientation angle by taking a Taylor expansion and solving closed-form equations. When $N_p > 3$, as we already have an efficient solution for three corners, RANSAC [7] could be applied to solve the problem, although this will be implemented in the future. At present, all possible sets of three adjacent corners from each building outline are selected to solve for a candidate LOH. We find, in general, the strategy of using 3 corners is reliable, and results are given in Section 4. Figure 7a shows an example of the set of candidate LOHs that our algorithm finds.

From Figure 7a, it can be seen that not all LOHs are reasonable: some LOHs are indoor, some LOHs have their visibility blocked by another building so they are not able to have visual of the corners they are matching with. To eliminate those bad LOHs, we perform a visibility check. We exam the visibility of a LOH by checking if its sightlines to the matched corners intersect with any building walls, and, if so, that LOH is eliminated.

As the last step, we take the visible LOH that is nearest to the noisy GPS location. In practice we find the correctly corresponded LOH is often selected when the noise of GPS is not too large. A major cause of deviation from the correctly corresponded LOH to the correct location is the accuracy of vertical corner edge positions in the building image, and that accuracy can be achieved with fairly small error. Thus our algorithm is able to produce results where the correctly corresponded LOH deviates from the correct location by only a few meters. Figure 7b shows visible LOHs and the red color indicates the area that if the noisy GPS falls within it, the correctly corresponded LOH will be selected.

4. Experiments

To evaluate our framework, we first collect 390 images using Google Street View from 11 unique locations in New York City to simulate user input images. We apply our TICEP feature extraction procedure on each image. We define a successful detection as all detection results deviate less than 20 pixels from the ground truth. Our algorithm identifies 1003 corner edges successfully out of all 1170 corner edges (85.73%), also in 263 images (67.44%)



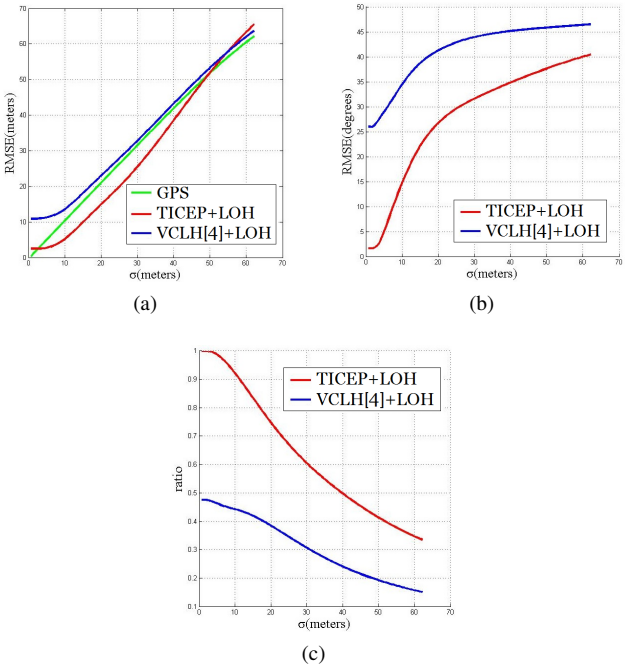
(a) (b)

Figure 7: (a) An example of solved LOHs (blue), correct location and orientation (red). (b) Example of visible LOHs, and area when GPS falls in the correctly corresponded LOH can be found.

all the edges are correctly identified. It should be noted that we detect three corner edges in each image, in practical applications the detection can be improved significantly with very little user aid. To measure the performance of GPS refinement and orientation estimation using TICEP+LOH, we use the 263 images with successful detection for the next experiment. The 2D region maps with building outlines are collected from *here.com*. We implement the framework using a mixture of c and Matlab, all the experiments are tested on an Intel Core i5 2.40GHz PC. The average run time of our whole framework is 1.7 seconds per image. Finally the dataset and code are available at *chuhang.github.io/vision.html*.

For each image and its corresponding region map, we first test the location and orientation error of the correctly corresponded LOH (i.e., the LOH computed with correct correspondence between building map corners and detected TICEPs) to the ground truth location and orientation provided by Google Street View. We compare our method with the method using the VCLH feature in [4] (corner edge positions without considering the influence of tilt angle) instead of our proposed TICEP feature. We compute the Root Mean Square Error (RMSE) of all 263 images, as listed in Table 1. Our method outperforms the compared method in both location and orientation. The RMSE of location of the correct LOH in our method is significantly smaller than the accuracy of a common mobile phone GPS in outdoor urban area (12.5 meters according to [17]). That explains why our method is able to improve the accuracy of GPS. In the first and second row of Figure 9, we show some examples of this experiment.

We have shown why our method is able to refine GPS, we now conduct the main experiment where we solve all LOHs among the map and combine our method with simulated noisy GPS to find the final refined location and orientation. To simulate the noise of GPS, we use a gaus-



(a) (b) (c)

Figure 8: Localization and orientation estimation results of different methods. (a) shows location error. (b) shows orientation error. (c) shows the ratio of samples where the correctly corresponded LOH is selected. σ measures the noise of GPS.

Table 1: RMSE of location and orientation of the correctly corresponded LOH of different methods.

	Proposed Method	Using VCLH in [4]
Location	$2.48m$	$18.68m$
Orientation	$1.6°$	$5.9°$

sian distributed noise with different standard deviations as suggested in [17]. We also compare our method with the method using VCLH feature in [4]. We experiment with different GPS noise standard deviation σ and 2000 noisy GPS locations are simulated for each image, so for each value σ we have 263×2000 test samples. Figure 8 shows the results.

Figure 8a show that when GPS has low noise, doing nothing (i.e., using the GPS estimate) produces the most accurate estimate. When the noise of GPS becomes larger, our method begins to outperform pure GPS by refining the noisy GPS to the correctly corresponding LOH. The average error of our method also increases as GPS noise increases, because as GPS uncertainty increases, it is more likely that the wrong LOH is closest to the GPS estimate (see the third row of Figure 9). This is also described in Figure 8c. When

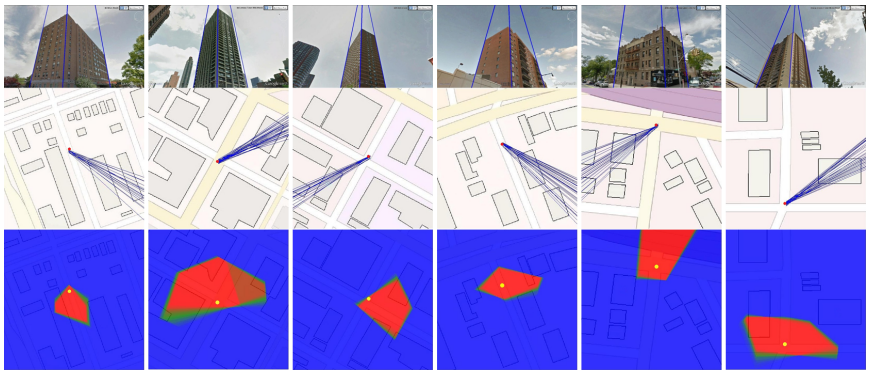


Figure 9: Top row: example building images with identified vertical corner edges. Middle row: 2D map with correctly corresponded LOH of images from same location (blue), ground truth location (red). Different images from the same location vary in ground truth orientation. Bottom row: The yellow dot shows the ground truth location. The red pixels show the locations to which the nearest LOH is the correctly corresponded LOH, so when noisy GPS falls in the red area the correctly corresponded LOH is selected (which we term as *refinable area*). These maps are averaged across all images of the same location.

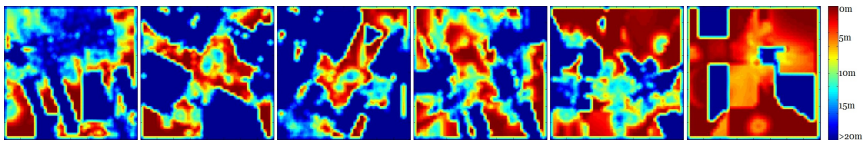


Figure 10: Heat maps of RMSE for every possible location on the map, sampled every 5 meters. We consider only outdoor locations. Pure blue means this location is either an indoor location or a location without any building (in any direction) that has at least three visible corners. When the user is at the blue area our method does not work, when the user is at the red area our method works well and produces small RMSE for location. The heat maps correspond to the maps in Figure 9.

using the VCLH feature in [4] where the influence of tilt angle is omitted, performance degrades and an accuracy that is higher than pure GPS cannot be achieved (as shown in Figure 8a). According to [17], the general RMSE of mobile phone GPS is 12.5 meters. Under such a noisy condition, our method is able to reduce the RMSE to 6.89 meters and the orientation estimate has average error $17.96°$.

Figure 8b also indicates a drawback of both methods. When GPS is accurate, the correctly corresponded LOH is selected so the orientation estimation is fairly accurate. However, in cases such as the first and third column of Figure 9, due to the layout of buildings, the correct location can be near to the border of the refinable area. That means there exists an incorrectly corresponded LOH near the correct location. This does not bring too much trouble to location refinement because that incorrectly corresponded LOH is not far from the correct location, but the estimated orientation can be affected significantly because incorrect corner matches are used. It should be noted that this problem is unavoidable unless increase the number of corner edges that are considered or use other features.

To further measure the generality of our framework, we conduct another experiment that estimates an upper bound on the performance of our method. For every outdoor location in the region map, we assume we have a pseudo image taken at that location of the nearest building with at least 3 visible corner edges, and measure the location RMSE of our refined result using 2000 simulated noisy GPS readings from a $12.5m$ gaussian noise distribution. In Figure 10 we show heat maps of RMSE. Figure 12 shows the distribution of RMSE errors for location estimation. For all outdoor locations of all our collected region maps, the percentage of area with at least one building with three visible corners is 80.6%, and the mean RMSE for all qualified locations is 6.70 meters. This indicates that for a large portion of urban environment, our method can be applied to refine noisy GPS location.

As the last experiment and a full demonstration of our intended application, we take building images in an apartment area using a mobile phone. Then we apply our framework with user aid in correcting mistakes in the corner edge identification step. We simulate 2000 noisy GPS locations for each image using a gaussian distribution with the standard 12.5 meter RMSE. Figure 11 shows the results and Table 2 lists statistics.

Figure 11: Left: Building images and their correctly corresponded LOHs in the 2D map, numbers show the RMSE of location and orientation. Right: Images, refinable areas (red), 38%, 68%, 95% of noisy GPS samples (concentric circles).

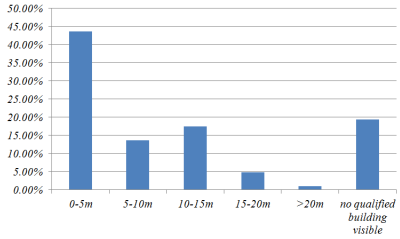


Figure 12: Histogram of RMSE of all outdoor locations.

Table 2: Statistics for the last experiment.

% of selecting correctly corresponded LOH	Location RMSE	Orientation RMSE
71.52%	$6.55m$	$12.7°$

5. Conclusion

We have presented a framework for refining a noisy GPS location and estimating the camera orientation using a building image, and a 2D map. We extract Tilt-Invariant Corner Edge Position features from the image, and identify plausible camera locations and orientations on the map that would result in images having lines at the observed positions. A set of Location-Orientation Hypotheses are proposed to describe the interaction between extracted features and the map effectively. Experiments show that our framework is able to improve accuracy of GPS, and determine the cameras orientation on the map. This framework could be useful for tourist navigation in an urban environment.

References

[1] G. Baatz, K. Köser, D. Chen, R. Grzeszczuk, and M. Pollefeys. Handling urban location recognition as a 2d homothetic problem. In *ECCV*, 2010. 2

[2] L. Baboud, M. Cadík, E. Eisemann, and H.-P. Seidel. Automatic photo-to-terrain alignment for the annotation of mountain pictures. In *CVPR*, 2011. 2

[3] B. Caprile and V. Torre. Using vanishing points for camera calibration. *IJCV*, 4(2):127–139, 1990. 4

[4] T.-J. Cham, A. Ciptadi, W.-C. Tan, M.-T. Pham, and L.-T. Chia. Estimating camera pose from a single urban ground-view omnidirectional image and a 2d building outline map. In *CVPR*, 2010. 2, 3, 4, 6, 7

[5] D. M. Chen, G. Baatz, K. Koser, S. S. Tsai, R. Vedantham, T. Pylvanainen, K. Roimela, X. Chen, J. Bach, M. Pollefeys, et al. City-scale landmark identification on mobile devices. In *CVPR*, 2011. 2

[6] J. M. Coughlan and A. L. Yuille. Manhattan world: Compass direction from a single image by bayesian inference. In *ICCV*, 1999. 3

[7] 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*, 24(6):381–395, 1981. 5

[8] A. C. Gallagher. Using vanishing points to correct camera rotation in images. In *CRV*, 2005. 4

[9] J. Hays and A. A. Efros. Im2gps: estimating geographic information from a single image. In *CVPR*, 2008. 2

[10] Y. Li, N. Snavely, D. Huttenlocher, and P. Fua. Worldwide pose estimation using 3d point clouds. In *ECCV*. 2012. 2

[11] M. Park, J. Luo, R. T. Collins, and Y. Liu. Beyond gps: determining the camera viewing direction of a geotagged image. In *ACM MM*, 2010. 2

[12] S. Ramalingam, S. Bouaziz, P. Sturm, and M. Brand. Skyline2gps: Localization in urban canyons using omni-skylines. In *IROS*, 2010. 2

[13] G. Schindler, P. Krishnamurthy, and F. Dellaert. Line-based structure from motion for urban environments. In *3DPVT*, pages 846–853, 2006. 2

[14] R. Szeliski. Where am i? In *ICCV*, 2005. 2

[15] J.-P. Tardif. Non-iterative approach for fast and accurate vanishing point detection. In *ICCV*, 2009. 3

[16] R. von Gioi, J. Jakubowicz, J.-M. Morel, and G. Randall. Lsd: A fast line segment detector with a false detection control. *PAMI*, 32(4):722–732, 2010. 3

[17] P. A. Zandbergen and S. J. Barbeau. Positional accuracy of assisted gps data from high-sensitivity gps-enabled mobile phones. *Journal of Navigation*, 64(03):381–399, 2011. 1, 6, 7

[18] B. Zhong, D. Xu, and J. Yang. Vertical corner line detection on buildings in quasi-manhattan world. In *ICIP*, 2013. 3