Hierarchical Semantic Hashing: Visual Localization from Buildings on Maps

Turgay Senlet
Department of Computer Science
Rutgers University
Piscataway, USA
tsenlet@cs.rutgers.edu

Tarek El-Gaaly
Department of Computer Science
Rutgers University
Piscataway, USA
tgaaly@cs.rutgers.edu

Ahmed Elgammal
Department of Computer Science
Rutgers University
Piscataway, USA
elgammal@cs.rutgers.edu

***Abstract*—In this paper we present a vision-based method for instant global localization from a given image. The approach mimics human understanding of maps and aerial images by using *semantic* information in the form of landmarks and their spatial layout on maps. We utilize a hierarchical semantic version of the Geometric Hashing algorithm. Instead of hashing local features, we use more robust semantic feature locations. Unlike other methods that use appearance or feature matching, our method relies on more robust and consistently detectable semantic elements that are invariant to illumination, temporal variations and occlusions. Spatial relations between these elements are efficiently stored and queried under a hashing scheme that inherently introduces translation, rotation and scale invariance. This approach provides a fast and robust localization technique for querying over large-scale areas. We experiment on a large 16.5 km^2 dense map of an urban downtown area with over 7,000 buildings.**

Keywords—visual localization; geometric hashing; satellite map; building detection; geographical information systems

I. INTRODUCTION

Visual localization is the challenging problem of determining the position of a camera with respect to a global or local frame of reference, by using visual sensing of the environment. The presence of camera noise, illumination changes, perspective effects and temporal variations (*e.g.* season changes) adversely affect the accuracy of visual localization thus making it a challenging problem, particularly in the case of localizing within large-scale images.

Given a view captured by a ground-facing camera, it is a challenging process to recognize where this view lies in a large geographic area with missing annotated geo-location information. This single view can either be taken by Unmanned Aerial Vehicles (UAV), surveillance satellites, airplanes or simply downloaded from the Internet. The latter case is very common nowadays with the plethora of images on the Internet and the demand to understand this large amount of data. One specific example of this is in aerial photography, where the photographer or user wants to be able to geo-locate the photographs taken.

In addition, matching between different data modalities, such as, GIS maps, satellite maps and images using conventional vision techniques are difficult and infeasible given the size of the maps. Using local image features, cross-correlation or bag-of-words techniques become impractical in the face of such amounts of data. Such algorithms fail when dealing with large-scale maps of urban/suburban areas because of the sheer number of visual features and amount of similar regions. Rooftops, roads and buildings look alike in these maps. A more semantic-level *understanding* of maps is much more feasible and better mimics the way humans understand and analyze such maps. We claim that a key feature that discriminates regions in these environments is the structure or layout of semantic landmarks, such as buildings.

In this paper, we present an instantaneous visual localization method based on the semantic structures –namely building locations- found on maps. We use a form of geometric hashing that utilizes semantic information. The localization method is able to localize a single view within a large-scale map containing up to 7,000 buildings. The map covers a 6.5 x 2.5 km (16.5km^2) geographic area, which is a portion of downtown Seattle, WA.

Our proposed method to perform localization from aerial images consists of two main stages: hashing and querying (Fig. 1). Hashing stage is the offline pre-process, where the building locations in the map are used to build an efficient hash-table structure for fast and robust querying. In the hashing step, we store the position and area inter-relations between buildings that lie in close proximity to one another. The querying stage is performed online and provides a single-shot global-localization using an observed orthographic image of a smaller area with buildings. Buildings appearing in the query image are extracted by our proposed building detection method. Resulting buildings and their geometric layout are used to obtain queries for the hash-table built in the hashing stage. After the query, we receive votes for possible locations on the map.

The contribution of this paper is that we use semantic features in images to perform fast and efficient localization on large-scale maps. We build a novel hashing scheme and a full pipeline around it to enable instantaneous localization in large city environments. The approach can be extended to work on arbitrary top view projective images, which is very well suited to visual localization systems on micro aerial vehicle (*i.e.*

Preprint submitted to 22nd International Conference on Pattern Recognition.
Received December 21, 2013.

UAV/MAV) or in the case of trying to recognize the location of an aerial view in a large-scale urban map.

Section II describes the related work. Section III describes the Semantic Hashing algorithm and the introduced pipeline: constructing the large-scale geometric hash-table of semantic features, querying stage, building detection and hierarchical querying. Section IV describes the large map that we experimented on. Section V details the experiments and discusses the results.

II. Related Work

The broad category of computer vision methods that provide global localization using only a single observation (*i.e.* instantaneous), use bag-of-words ([1]), map-matching (or template-matching/correlation) [2] and image retrieval algorithms [3]. These algorithms come up with efficient ways to search a vast prebuilt set of features of the environment that are then used to match against query observations. Given a query image, these methods retrieve a probability distribution of possible locations. BoW can be thought of as attempting to capture higher-level groups of features that commonly occur together in the environment. By this, BoW takes a step closer to semantic-level features. In the case of localization on large urban maps there is a large occurrence of self-similar regions. This is a challenge for BoW-type algorithms. In Fig. 2 we can see the marked similarity in the majority of the map. This makes BoW an unsuitable choice for localizing in such urban dense maps filled with similar regions. Map-matching methods such as cross-correlation and template-matching generally suffer with scale due to their computationally heavy operations. Typically cross-correlation methods are applied in a sliding-window fashion. This is computationally expensive and becomes worse when rotation and scale invariance is considered. There is a large body of work on image retrieval algorithms but many rely on the basis of the aforementioned methods.

Semantic Mapping is a term given to mapping an environment using recognized semantic objects [4]. Semantic-level features are more robust against visual uncertainties in uncontrolled environments than local point features. These methods have focused on indoor environments where semantics may be seen as a more concise due to the presence of informative objects. Other approaches have focused on constructing semantic-level information from low-level features [5].

The most relevant work is a method proposed in [6] for visual localization on satellite imagery using image features. This work uses low-level local SIFT feature matching. This work also uses a BoW-type of approach to quantize the features. The matching is done for particular visually discriminative areas of the map. Urban environments require other structural cues to disambiguate between regions. Approaches for vehicle localization using satellite maps and street view images are proposed in [7], [8], respectively. Both use tracking frameworks to iteratively localize their positions.

Map matching based UAV localization in a tracking framework is presented in [9], where image-to-image and image-to-map registration is performed iteratively. SLAM has been affectively applied to the recently growing field of UAV localization [10]. These approaches are used for small-scale environments and do not provide instantaneous global localization on maps and hence susceptible to local drift.

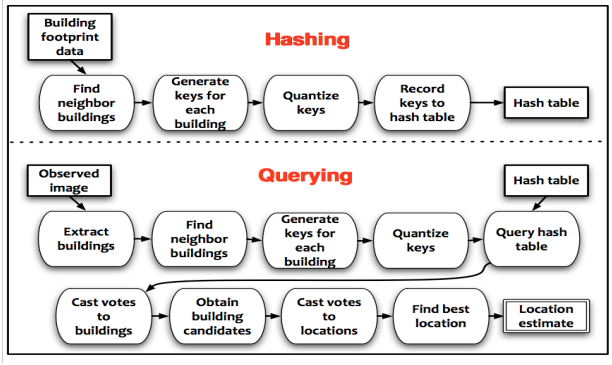

Fig. 1 Algorithm overview. Algorithm consists of two main stages; offline hashing and online image querying.

Although Geometric Hashing [11] has not been directly used for general map based localization method, it has been used as coarse global localization methods by several works in other scenarios: edge based indoor localization, where each room is a model is proposed in [12] and building-facade based 2D localization on maps [13]. The latter approach makes use of hand-made ground truth of building facades on satellite images and images taken from a ground level robot. Unfortunately the method is very limited in scaling to larger areas. Experiments are performed on only 111 buildings. Also the method requires very detailed hand-made façade datasets to be constructed and it accuracy depends on this.

III. Semantic Geometric Hashing for Localization

First we describe the construction of the semantic geometric hash-table that encodes the layout of small neighborhoods of buildings in the map. At query time we are given an observation image and the goal is to localize this by matching it with the closest location using the semantic geometric hash-table.

A. Constructing the Semantic Geometric Hash-table

Geometric hashing [17] has been proposed as a model-based efficient object recognition scheme that has later been used for many different pattern recognition tasks including 3D shape recognition [18], pose estimation, hand-written image matching; mainly using point or line features [19], [20].

Unlike the traditional geometric hashing applications that use low-level point or line features in an image, we present an algorithm that makes use of higher-level semantic features in the scene. We hash the building centroid locations and building features such as the area ratios. This approach is more robust against noise or detection errors when compared to using low-level visual features in the image. Furthermore using semantic features enables us to match across different data modalities, *e.g.* matching satellite images to GIS building contour polygon data.

Preprint submitted to 22nd International Conference on Pattern Recognition.
Received December 21, 2013.

In geometric hashing, a model consists of a set of feature points that belong to a single entity to be detected in the scene. The performance of hashing degrades when the feature points to be hashed in a model increase too much. Hence we need to define our models to be small subsets of the whole building dataset. We define each model to be the K nearest neighbor buildings around each building in the map. For each building we find at most K nearest neighbors inside a neighborhood R to be the model for that particular building. The hashing step is the step where we store the geometric inter-relations (*i.e.* layout/structure) of the elements of these models. In the query stage, detection of a model from hash-table will suggest localizing that particular model and localizing its main building.

Hashing algorithm steps are given in Algorithm 1, where for each building to be hashed (main building), we select each of its neighbors to be the base neighbor to form a baseline with the main building location, to normalize the coordinates of the rest of the buildings (Fig. 2). Normalization is performed by considering the new coordinate system where the main building center is the origin and base neighbor center is the point (1,0). These normalized coordinates along with the ratio of areas of the buildings are quantized to generate keys for hash-table entries for the model. We use area ratios within neighbors to create more unique hash-keys. Using the ratio between building areas is a discriminative feature, which is also scale and rotation invariant. Corresponding to each key we note down the main building number and base neighbor number in the hash-table. The K nearest neighbor approach and the selected baseline reduces the per model geometric hashing complexity to $O(K^2)$. In the original geometric hashing algorithm, all possible point pairs in a model are used as a basis, which leads to $O(K^3)$ complexity. With our approach, the complete hashing time complexity becomes $O(nK^2)$, where n is the number of buildings to be hashed in the whole map.

Geometric hashing is inherently invariant to scale and rotation due to the normalized coordinate system of each hashed local neighborhood and the normalization with respect to the basis, which are taken as each pair of points within the local neighborhood. Geometric hashing by its construction is also robust to occlusion and missing data.

B. *Query Stage*

1) Automatic Building Detection from Satellite Images

In order to be able to query the location of a given image, we need to extract the semantic features to build our semantic hashing queries. By processing a given query image, we automatically determine the coarse building locations and sizes by employing a combination of machine learning and image processing techniques. Although in the scenarios we are dealing with, the query images can come from a variety of different sources like UAV's or aerial imagery, for our testing purposes, because of their availability and ease of access, we chose to use satellite images (Fig. 3 (a)) to test our queries.

Automatic and semi-automatic building footprint detection from aerial imagery is a well studied area in the Geographic Information Systems and Remote Sensing (GIS-RS) literature [21]-[24], even though usually building detection in Remote Sensing is performed using the help of other sensors and geographic information like LIDAR, Digital Elevation Maps (DEM) and multi-spectral imagery.

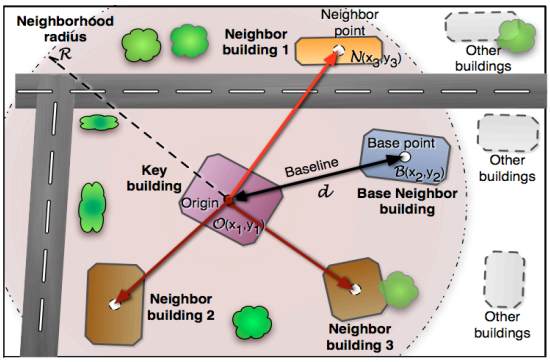

Fig. 2 Building hashing scheme. Hashing baseline is formed by the main building and base neigbor centers.

Algorithm 1 – Hashing

```
Input: buildings
hash_table ← []
for all main_building in buildings do
    (x1,y1) ← centroid(main_building)
    a1 ← area(main_building)
    N ← neighbors(main_building, max_neighbors, R)
    for all base_neighbor in N do
        (x2,y2) ← centroid(base_neighbor)
        a2 ← area(base_neighbor)
        baseline ← vector((x2, y2), (x1, y1))
        other_neighbors ← N – {base_neighbor}
        for all neighbor in other_neighbors do
            (x3, y3) ← centroid(neighbor)
            a3 ← area(neighbor)
            (fx, fy) ← normalize((x3,y3), baseline)
            (qx, qy) ← quantize(fx, fy)
            qa13 ← quantize(a1 / a3),   qa23 ← quantize(a2 / a3)
            key ← [qx, qy, qa13, qa23]
            value ← [main_building, base_neighbor]
            hash_table[key].append(value)
return: hash_table
```

We are proposing a building detection algorithm to extract building footprints from a given image (*e.g.* Fig. 4 (a)) even in the presence of occlusion from vegetation and high similarities in the appearances of roads and building roofs. As a first step, we use an image classifier to obtain pixel-wise classification probabilities for buildings (Fig. 4 (c)). The classifier is a three-class Random Forest classifier trained over training samples of vegetation, road and building regions. Building classification results are binarized and noise reduction filters are applied to remove unwanted salt and pepper noise and small components. Each connected component in the resulting image that has a large enough area and that is structurally not too thin for a building is considered as a building seed region. Holes are filled and building seeds are then dilated outwards based on color similarity to complete missing part of the buildings. Detection results are shown in Fig. 4 (d). Missed detections and false detections are highlighted in Fig. 4 (f).

Preprint submitted to 22nd International Conference on Pattern Recognition.
Received December 21, 2013.

Most of the missed detections are due to tree occlusions and buildings with similar appearance to roads. False detections occur when isolated road patches are detected as buildings. Building detection accuracy is listed in the results section.

2) Hierarchical Querying

We localize the query image by employing a hierarchical query approach similar to the original geometric hashing based model-detection. We use the buildings extracted from the query image and the previously built hash-table. As detailed in Algorithm 3, we first independently localize every detected building in the image. Then we combine the single building localization results that agree on image center locations to obtain final image localization (Algorithm 2). This hierarchy enables our approach to be robust against missing or false building detections and incorrect localizations of individual buildings. The output of each single building localization step is the index of the best matching building on the map as well as scale and orientation hypotheses for the building.

In the query step, building neighborhoods are generated from the detected buildings. A similar hash-table baseline and key generation procedure is followed to generate a query key to the hash-table. All the values in the hash-table that correspond to this key are the initial candidate buildings for the query building location. Using the main building and baseline neighbor numbers in each value, probable orientation and scales votes for each candidate are generated. The single building localization result is the highest voted building candidate number with its specific orientation and scale.

The hierarchical approach we propose for the problem is the succeeding step where we take the building localization results for all the buildings in the query image and cast votes for the orientation and scale of that image and location of the image center. The votes are quantized in order to compensate for mismatches due to individual localization errors and quantization artifacts. The highest voted area is the localization estimation result for the center of the query image. An improvement step can also be applied to the quantized localization estimate to further fine-tune it and get more precise localization results.

IV. Large-Scale Dataset of Buildings

The buildings dataset we used to build the semantic hash-table is a part of the Seattle's building outline data provided by Seattle City GIS Program [16]. In the provided GIS dataset, outlines of buildings in Seattle were extracted based upon imagery acquired in 2009 and further hand processed for higher precision (in the order of centimeters). Data provided is composed of individual building contour polygons with WGS84 and coordinates in feet.

The part of the data we used covers a 16.5km^2 area that contains 7,000 buildings with their geographic coordinates and detailed outline contours. Fig. 3 (a) shows the satellite image of the area of interest and (b) GIS building outline data overlaid on top of the Google Maps. We chose this part of the city to include both the downtown area (Fig. 3 West) containing large and sparse buildings and the residential Central District area (Fig. 3 East) containing neighborhoods of small and dense buildings.

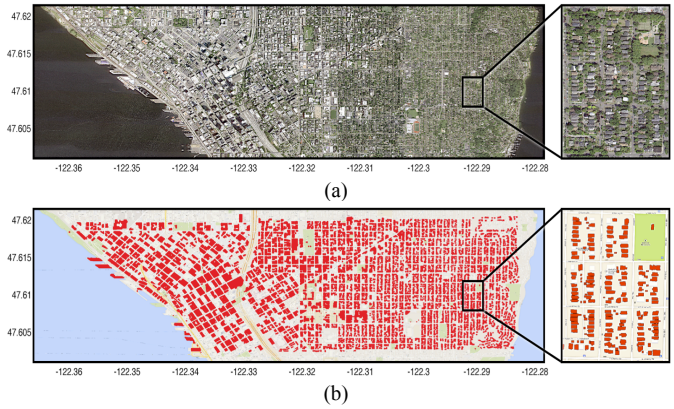

Fig. 3 (a) Satellite map of the area that has been used in the localization queries. (b) Same area with ground truth GIS building outlines ovelayed on top of the Google Maps image of the area. All 7,000 buildings with their outlines are shown on the map. Best viewed on computer screen.

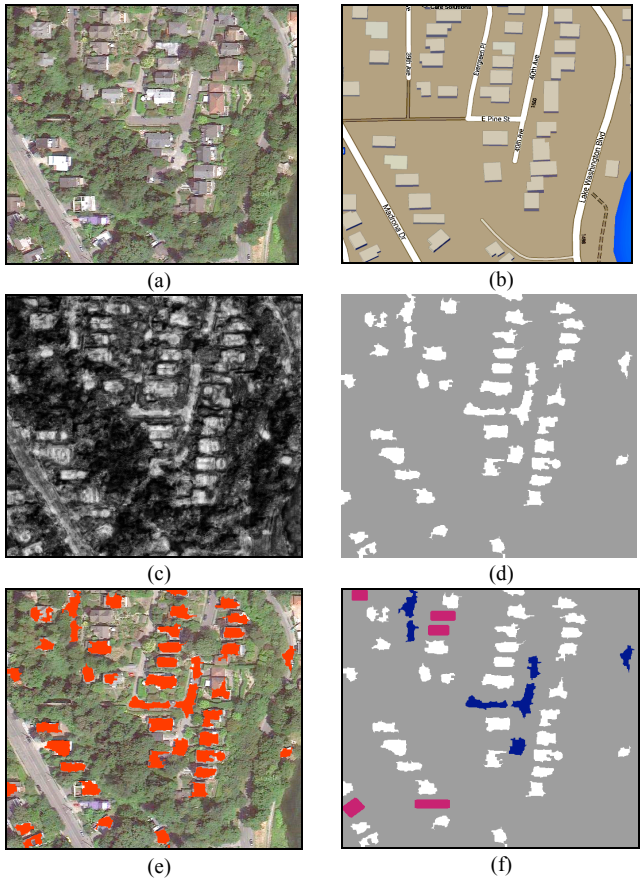

Fig. 4 Building detection results on a sample tile. (a) Google Maps Satellite image tile, (b) Google Maps tile, (c) Pixel-based building classification results from satellite image, brighter values are higher building probability areas, (d) Final building detection results, (e) Building detection results overlayed on satellite image, (f) Building detection showing correct (white), missing (magenta) and false (dark blue) detections. Best viewed in color.

Preprint submitted to 22nd International Conference on Pattern Recognition.
Received December 21, 2013.

Algorithm 2 – Image Query

Input: query_image, hash_table
image_center_votes ← []
query_buildings ← detect_buildings(*query_image*)
for all *main_building* in *query_buildings* **do**
  match ← **LocateBuilding**(*main_building, hash_table*)
  image_center_vote ← calculate_image_center(*match*)
  q_{center} ← quantize(*image_center_vote*)
  image_center_votes[q_{center}]++
best_center ← x,y of max(*image_center_votes*)
return: *best_pose_no*

Algorithm 3 – Locate Building

Input: main_building, hash_table
(x_1,y_1) ← centroid(*main_building*)
a_1 ← area(*main_building*)
N ← neighbors(*main_building, max_neighbors*, ∞)
query_hash_table ← []
for all *base_neighbor* in N **do**
  (x_2,y_2) ← centroid(*base_neighbor*)
  a_2 ← area(*base_neighbor*)
  baseline ← vector((x_2, y_2), (x_1, y_1))
  other_neighbors ← N – {*base_neighbor*}
  for all *neighbor* in *other_neighbors* **do**
    (x_3, y_3) ← centroid(*neighbor*)
    a_3 ← area(*neighbor*)
    p ← [(x_1, y_1) , (x_2, y_2) , (x_3, y_3)]
    (f_x, f_y) ← normalize((x_3,y_3), *baseline*)
    (q_x, q_y) ← quantize(f_x, f_y)
    q_{a13} ← quantize(a_1 / a_3), q_{a23} ← quantize(a_2 / a_3)
    key ← [$q_x, q_y, q_{a13}, q_{a23}$]
    values ← *hash_table*[*key*]
    for all *value* in *values* **do**
      baseline ← *value*
      building_id ← *value*[0]
      building_theta ← calculate_theta(*baseline,p*)
      building_scale ← calculate_scale(*baseline,p*)
      q_{theta} ← quantize(*building_theta*)
      q_{scale} ← quantize(*building_scale*)
      building_key ← [*building_id*, q_{theta}, q_{scale}]
      query_hash_table[*building_key*]++
match ← {scale, theta, id} of max(*query_hash_table*)
return: *match*

V. Experiments and Results

To test the semantic geometric hashing method presented we perform the task of localization of query satellite images on the large-scale dataset. The building contour data from the GIS Seattle building data is used to construct the hash-table. Non-overlapping image segments of size 1024x1024px from Google Earth satellite images (40cm/px resolution) are used for querying and evaluating our localization method. We report on the accuracy of the building detection and localization stages and analyze the robustness of each under different parameter configurations.

A. Building Detection

We report the *precision* and *recall* for the building detection step in Table I. Precision and recall are not based on pixel matching criteria but instead by comparison of location and area of the buildings against the ground-truth within a threshold. Our method detects ~80% buildings correctly, which leads to a high recall, but suffers from false positives that decrease the precision. The majority of false positives come from roads having similar appearance to residential building rooftops where tree-occlusions and complex building structures lead to the false negatives.

TABLE I. Building Detection Results

Query Image Size	Precision	Recall
1024x1024 pixel images	65.27%	80.51%

B. Large-Scale Localization

In Fig. 5 we report the localization accuracy for map areas divided into *denseness* categories; each category representing the denseness (*i.e.* number of buildings per unit area – downtown areas tend to be less dense than residential due to larger buildings). It can be seen that the more semantic information in the form of buildings, the better the localization accuracy. This is seen by the large increase in localization accuracy (up to 100%) in areas with 200+ buildings. The overall average accuracy for localization is ~90%. In Fig. 5 we show the accuracy for the original query images as well as for rotated query images. We rotate the images 90 and 180 degrees to show the invariance to rotation. The accuracy can be seen to be comparable in the 3 rotation configurations. Scale is inherently handled by the normalization of the coordinate system when computing the geometric hashing.

There are a couple of ways to explain why dense regions provide better localization accuracy. Firstly, dense regions provide more semantic information in the form of buildings. Secondly, sparse regions tend to have larger buildings and therefore in both hashing and query, the number of close neighbors inside the neighborhood radius may be less than in denser regions. Finally, the building detection algorithm may perform worse in downtown areas where more complex appearance buildings exist.

Due to the robustness of the hierarchical localization algorithm, even with imperfect building segmentation results we still perform relatively good localization.

During testing we take the area with the maximum number of votes as the estimated location of the query image. This is a hard localization decision. It is also possible to include a soft localization decision, which would examine the top k voted locations within the large-scale map. Later, these regions can then be examined more closely.

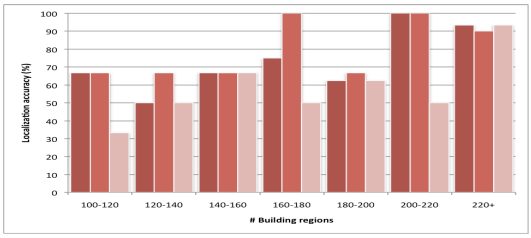

Fig. 5 Localization accuracy vs number of buildings in the query image. Accuracy increases as the number of buildings increase. 3 color bars show the accuracy for 3 different angle rotations: 0,90,180 degrees

C. Computational Performance

To evaluate how well the presented approach does for the large-scale dataset we have experimented on, and to motivate the approach's ability to cope with even larger datasets, we report on average hashing time, query time and hash-table size with respect to the number of buildings. As shown in Fig. 6, query time and hash-table size scale linearly with increasing number of buildings. Hashing time is observed to be almost constant and the small increase in the hashing time are believed to be coming from practical implementation issues like increasing memory requirements.

We also show the average hashing time, query time and hash-table size against the changes in the number of local neighborhood buildings. Query time and hash-table size increase quadratically with increasing number of neighbors. Hashing increases almost linearly. Deviation in query time last data point is due to insufficient memory and disk swapping.

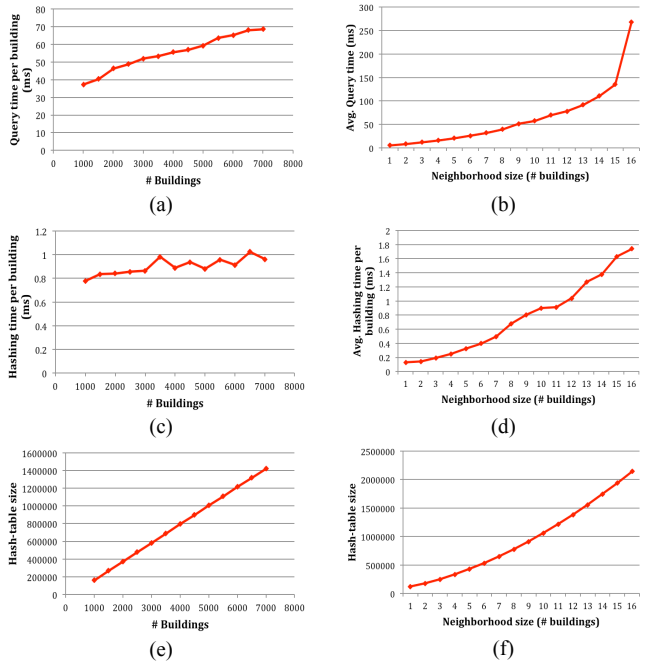

Fig. 6 (a) Query time for each building, (c) Hashing time for each building, (e) Total hash-table entries for each building. (b) Query time vs. neighborhood size, (d) Hashing time vs. neighborhood size, (f) Total hash-table entries vs. neighborhood size.

VI. Conclusion

We demonstrate a complete vision-based instantaneous localization pipeline that has no manual steps. We present a novel semantic hashing scheme invariant to scale, rotation, translation and in addition robust to occlusions. The method is also scalable with increasing number of buildings stored in the database. We use this to perform efficient localization of aerial images on large-scale maps. Our approach can be modified to work with arbitrary top view images and other semantic landmark information. We performed experiments on a large urban area of Seattle with a large number of buildings and self-similar regions and validated that the presented localization method generates good localization results provided that sufficient buildings exist in the query images.

References

[1] D. Filliat, "A visual bag of words method for interactive qualitative localization and mapping," presented at the Robotics and Automation, 2007 IEEE International Conference on, 2007, pp. 3921–3926.

[2] J. P. Lewis, "Fast normalized cross-correlation," *Vision interface*, 1995.

[3] R. Datta, D. Joshi, J. Li, and J. Z. Wang, "Image retrieval: Ideas, influences, and trends of the new age," *ACM Computing Surveys (CSUR)*, vol. 40, no. 2, pp. 1–60, Apr. 2008.

[4] A. Pronobis, "Semantic Mapping with Mobile Robots," KTH Royal Institute of Technology, Stockholm, Sweden.

[5] W. Jiang, K. L. Chan, M. Li, and H. Zhang, "Mapping low-level features to high-level semantic concepts in region-based image retrieval," presented at the Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on, 2005, vol. 2, pp. 244–249.

[6] C. Wu, F. Fraundorfer, J.-M. Frahm, and J. Snoeyink, "Image localization in satellite imagery with feature-based indexing," presented at the Proceedings of the International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences, Beijing, 2008, pp. 197–202.

[7] G. Vaca-Castano, A. R. Zamir, and M. Shah, "City scale geo-spatial trajectory estimation of a moving camera," presented at the Computer Vision and Pattern Recognition, IEEE Conference on, 2012, pp. 1186–1193.

[8] T. Senlet and A. Elgammal, "A framework for global vehicle localization using stereo images and satellite and road maps," presented at the Computer Vision Workshops, IEEE International Conference on, 2011, pp. 2034–2041.

[9] Y. Lin and G. Medioni, "Map-enhanced UAV image sequence registration and synchronization of multiple image sequences," presented at the Computer Vision and Pattern Recognition, IEEE Conference on, 2007, pp. 1–7.

[10] M. W. Achtelik, S. Lynen, S. Weiss, L. Kneip, M. Chli, and R. Siegwart, "Visual-inertial SLAM for a small helicopter in large outdoor environments," presented at the Intelligent Robots and Systems (IROS), 2012 IEEE/RSJ International Conference on, 2012, pp. 2651–2652.

[11] I. Rigoutsos and H. J. Wolfson, "Geometric Hashing," *Computational Science & Engineering, IEEE*, vol. 4, no. 4, p. 9, 1997.

[12] Y. B. Yang and H. T. Tsui, "Mobile robot localization by geometric hashing and model-based scene matching," presented at the Pattern Recognition, 1996., Proceedings of the 13th International Conference on, 1996, vol. 1, pp. 181–185.

[13] T.-J. Cham, A. Ciptadi, W.-C. Tan, M.-T. Pham, and 2. I. C. O. Liang-Tien Chia Computer Vision and Pattern Recognition CVPR, "Estimating camera pose from a single urban ground-view omnidirectional image and a 2D building outline map," presented at the Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on, 2010.

[16] *Seattle's Data Site*. [Online]. Available: https://data.seattle.gov/dataset/2009-Building-Outlines/y7u8-vad7. [Accessed: 19-Dec-2013].

[17] H. J. Wolfson and I. Rigoutsos, "Geometric hashing: an overview," *Computational Science & Engineering, IEEE*, vol. 4, no. 4, pp. 10–21, 1997.

[18] H. Van Dijck, M. Korsten, and F. Van Der Heijden, "Robust 3-dimensional object recognition using stereo vision and geometric hashing," presented at the Image Processing, 1996. Proceedings., International Conference on, 1996, vol. 1, pp. 329–332.

[19] J.-J. Liu and R. Hummel, "Geometric hashing with attributed features," presented at the CAD-Based Vision Workshop, 1994., Proceedings of the 1994 Second, 1994, pp. 9–16.

[20] F. C. Tsai, "A probabilistic approach to geometric hashing using line features," *Computer Vision and Image Understanding*, vol. 63, no. 1, pp. 182–195, 1996.

[21] W. Willuhn and L. Van Gool, "Building reconstruction from aerial images using efficient semi-automatic building detection," 2005.

[22] B. P. Olsen, "Automatic change detection for validation of digital map databases," *International Archives of Photogrammetry, Remote Sensing and Spatial Information Sciences*, vol. 34, pp. 569–574, 2004.

[23] N. Ekhtari, M. R. Sahebi, M. V. Zoej, and A. Mohammadzadeh, "Automatic building detection from Lidar point cloud data," 2008.

[24] N. Shorter and T. Kasparis, "Automatic vegetation identification and building detection from a single nadir aerial image," *Remote Sensing*, vol. 1, no. 4, pp. 731–757, 2009.

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

Preprint submitted to 22nd International Conference on Pattern Recognition.
Received December 21, 2013.