Vision-Based Positioning for Internet-of-Vehicles

Kuan-Wen Chen, Chun-Hsin Wang, Xiao Wei, Qiao Liang, Chu-Song Chen, Ming-Hsuan Yang, and Yi-Ping Hung

***Abstract*—This paper presents an algorithm for ego-positioning by using a low-cost monocular camera for systems based on the Internet-of-Vehicles (IoV). To reduce the computational and memory requirements, as well as the communication load, we tackle the model compression task as a weighted k-cover problem for better preserving the critical structures. For real-world vision-based positioning applications, we consider the issue of large scene changes and introduce a model update algorithm to address this problem. A large positioning dataset containing data collected for more than a month, 106 sessions, and 14,275 images is constructed. Extensive experimental results show that sub-meter accuracy can be achieved by the proposed ego-positioning algorithm, which outperforms existing vision-based approaches.**

***Index Terms*—Ego-positioning, model compression, model update, long-term positioning dataset.**

I. Introduction

INTELLIGENT transportation systems have been extensively studied in the last decade to provide innovative and proactive services for traffic management and driving safety issues. Recent advances in driving assistance systems mostly provide stand-alone solutions to these issues by using sensors limited to the line of sight. However, many vehicle accidents occur because of other vehicles or objects obstructing the view of the driver. Had the visual information of all nearby vehicles been available via the Internet-of-Vehicles (IoV), such accidents could have been avoided. To achieve this goal with IoV, one of the main tasks is constructing a neighbor map [1] that estimates the positions of all the surrounding vehicles by integrating the positioning information obtained from the nearby vehicles via vehicle-to-vehicle or vehicle-to-infrastructure communication. Chen *et al.* [1], [2] have proposed a neighbor map construction algorithm, whose accuracy highly depends on the ego-positioning results of each vehicle. While most vehicular communication methods are non-directional, sharing information with other vehicles becomes challenging if their accurate location cannot be detected.

Ego-positioning aims at locating an object in a global coordinate system based on its sensor inputs. With the growth of mobile or wearable devices, accurate positioning has become increasingly important. Unlike indoor positioning [3], [4], [5], [6], considerably less efforts have been put into developing high-accuracy ego-positioning systems for outdoor environments. Global Positioning System (GPS) is the most widely used technology implemented in vehicles. However, the precision of GPS sensors is approximately 3 to 20 meters [7], [8], which is not sufficient for distinguishing the traffic lanes and highway lane levels critical for intelligent vehicles. In addition, existing GPS systems do not work properly in urban areas where signals are obstructed by high rise buildings. Although GPS-based systems using external base stations (e.g., RTK satellite navigation [9]) and positioning methods using expensive sensors (e.g., radar sensors and Velodyne 3D laser scanners) can achieve high positioning accuracy, they are not widely adopted because of cost issues. Hence, it is important to develop accurate ready-to-deploy IoV approaches for outdoor environments.

In this paper, we introduce a vision-based positioning algorithm that utilizes low-cost monocular cameras. It exploits visual information from the images captured by vehicles to achieve sub-meter positioning accuracy even in situations of

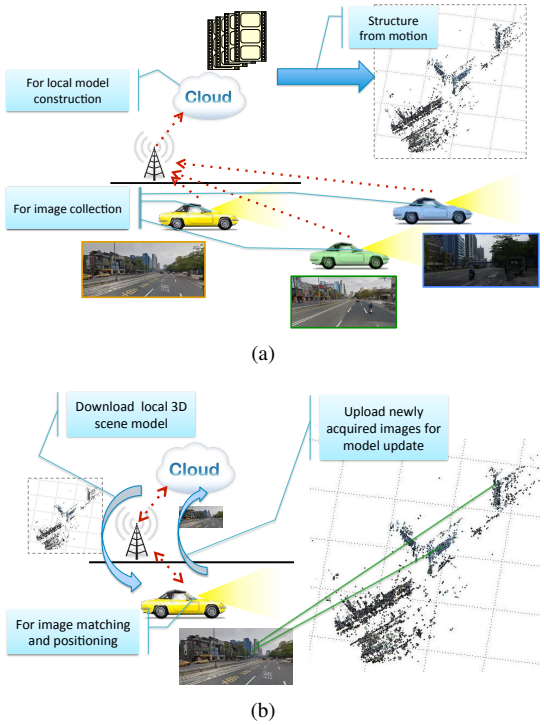


Fig. 1. Overview of the proposed vision-based positioning algorithm: (a) training phase: images from passing vehicles are uploaded to a cloud server for model construction and compression; (b) ego-positioning phase: SIFT features from images acquired on vehicles are matched against 3D models previously constructed for ego-positioning. In addition, the newly acquired images are used to update 3D models.

K.-W. Chen is with the Department of Computer Science, National Chiao Tung University, Taiwan, e-mail: kuanwen@cs.nctu.edu.tw.

C.-H. Wang, X. Wei, and Q. Liang are with Graduate Institute of Networking and Multimedia, National Taiwan University, Taiwan.

C.-S. Chen is with the Institute of Information Science and Research Center for Information Technology Innovation, Academia Sinica, Taiwan and also with the Institute of Networking and Multimedia, National Taiwan University, Taiwan, e-mail: song@iis.sinica.edu.tw.

M.-H. Yang is with the School of Engineering, University of California, Merced, USA, e-mail: mhyang@ucmerced.edu.

Y.-P. Hung is with the Institute of Networking and Multimedia and the Department of Computer Science and Information Engineering, National Taiwan University, Taiwan, e-mail: hung@csie.ntu.edu.tw.

huge traffic jams. An overview of the proposed algorithm is shown in Fig. 1.

The proposed algorithm includes the training and ego-positioning phases. In the training phase, if a vehicle passes an area that can be roughly positioned by GPS, the captured images are uploaded to a cloud server. In the proposed algorithm, GPS is only used for roughly positioning as prior information to narrow down the search region, i.e., we know the local models to be used, matched and updated, rather than matching all images or local models. In our experiments, the range of a local model is at most 800 meters, and thus we only need to have estimation with error less than 400 meters. After numerous vehicles passing through that area, the cloud server collects a sufficient number of images to construct the local 3D point set model of the area by using a structure-from-motion algorithm [10], [11], [12].

In the ego-positioning phase, the vehicle downloads all the local 3D scene models along the route. When driving, the approximate position of the vehicle can be known by using the GPS module, and the current image of the vehicle is matched with the associated local 3D model for ego-positioning. Note that our algorithm only needs a vehicle to download the scene models via I2V communication in the ego-positioning phase, and upload the collected images for model update via V2I communication when the wireless bandwidth is available. Both I2V and V2I wireless communications can be carried out using LTE communication as the bandwidth of 4G technology is sufficient for such tasks. Although there may be hundreds of vehicles in the same area, and the bandwidth would not be enough for all transmissions at the same time. In the proposed algorithm, for I2V communication, the scene models can be preloaded when a fixed driving route is provided. Furthermore the model size of one scene can be compressed significantly in our experiments. For V2I communication to upload images for update, the problem can be alleviated by transmitting a number images from a few vehicles.

Although model-based positioning [13], [14], [15] has been studied in recent years, most of the approaches construct a single city-scale or close-world model and focus on fast correspondence search. As feature matching with large scale models is difficult, the success rate for registered images is under 70% with larger positioning errors (e.g., on average 5.5 meters [13] and 15 meters [14]). Fundamentally, it is difficult to construct and update a large-scale (or world-scale) model for positioning.

The main contributions of this work are summarized as follows. First, we propose a vision-based ego-positioning algorithm within the IoV context. We tackle the above-mentioned issues and practicalize model-based positioning for real-world situations. Our approach involves constructing local and update models, and hence sub-meter positioning accuracy is attainable. Second, a novel model compression algorithm is introduced, which solves the weighted k-cover problem to preserve the key structures informative for ego-positioning. In addition, a model update method is presented. Third, a dataset including more than 14,000 images from 106 sessions over one month (including sunny, cloudy, rainy, night scenes) is constructed and will be made publicly available. To the best of our knowledge, the proposed vision-based ego-positioning system is the first to consider large outdoor scene changes over a long period of time with sub-meter accuracy.

II. Related Work and Problem Context

Vision-based positioning aims to match current images with stored frames or pre-constructed models for relative or global pose estimation. Such algorithms can be categorized into three types according to the registered data for matching. i.e., consecutive frames, images sets, and 3D models. In addition, we discuss related work on compression of 3D models and long-term positioning.

A. Vision-Based Positioning

1) Matching with Consecutive Frames: Positioning methods based on visual odometry, which involves matching the current and previous frames for estimating relative motion, are widely used in robotics [16], [17], [18]. These methods combine the odometry sensor readings and visual data from a monocular or stereo camera for the incremental estimation of local motion. The main drawbacks of these methods are that only the relative motion can be estimated, and the accumulated errors may be large (about 1 meter a movement of 200 meters [18]) with the drifting problems. In [19] Brubaker *et al*. incorporate the road maps to alleviate the problem with accumulated errors. However, the positioning accuracy is low (i.e., localize objects with up to 3 meter accuracy).

2) Matching with Image Sets: Methods of this category match images with those in a database to determine the current positions [20], [21]. Such approaches are usually used in multimedia applications where accuracy is not of prime importance, such as non-GPS tagged photo localization and landmark identification. The positioning accuracy is usually low (between 10 and 100 meters).

3) Matching with 3D Models: Methods of this type are based on a constructed 3D model for positioning. Arth *et al*. [22] propose a method to combine a sparse 3D reconstruction scheme and manually determine visibility information to estimate camera poses in an indoor environment. Wendel *et al*. [23] extend this method to an aerial vehicle for localization in the outdoors scenarios. Both these methods are evaluated in constrained spatial domains in a short period of time.

For large-scale localization, one main challenge is matching features efficiently and effectively. Sattler *et al*. [14] propose a direct 2D-to-3D matching method based on quantized visual vocabulary and prioritized correspondence search to quicken the feature matching process. This method is extended to both 2D-to-3D and 3D-to-2D search [15]. Lim *et al*. [24] propose the extraction of more efficient descriptors at every 3D point across multiple scales for feature matching. Li *et al*. [13] deal with a large scale problem including hundreds of thousands of images and tens of millions of 3D points. As these methods focus on efficient correspondence search, the large number of 3D point cloud models results in a registration rate of 70% and mean of localization error of 5.5 meters [13]. Thus, these approaches are not applicable to IoV systems because

of position accuracy, and computational and memory requirements. Recently, methods that use a combination of local visual odometry and model-based positioning are proposed [25], [26]. These systems estimate relative poses of mobile devices and carry out image-based localizations on remote servers to overcome the problem of accumulative errors.

However, none of these model-based positioning approaches considers outdoor scene changes and evaluates images taken in the same session as that of the model being constructed. In this paper, we present a system that deals with large scene changes and updates models for long-term positioning within the IoV context.

B. Model Compression

In addition to feature matching for self-positioning, compression methods have been developed to reduce the computational and memory requirements for large-scale models [27], [28], [29]. Irschara *et al*. [27] and Li *et al*. [28] use greedy algorithms to solve a set cover problem to minimize the number of 3D points while ensuring a high probability of successful registration. Park *et al*. [29] further formulate the set cover problem as a mixed integer quadratic programming problem to obtain an optimal 3D point subset. Cao and Snavely [30] propose a probabilistic approach that considered both coverage and distinctiveness for model compression. However, these approaches only consider point cloud coverage without taking the spatial distribution of 3D points into account. To overcome this problem, a weighted set cover problem is proposed in this work to ensure that the reduced 3D points not only have large coverage, but are also fairly distributed in 3D structures, thereby facilitating feature matching for accurate ego positioning.

C. Long-Term Positioning

Considerably less attention has been paid to long-term positioning. Although a few methods [31], [32] in robotics consider outdoor scene changes for localization and mapping, only 2D image matching is carried out across time spans. In terms of these applications, correct matching is defined as when the distance between a pair of matched images is less than 10 or 20 meters by GPS measurements, which is significantly different from estimating the 3D absolute positions addressed in this work.

III. System Overview

The proposed system consists of two phases, as illustrated in Fig. 2. The training phase is carried out on local machines or cloud systems. It includes three main components, which are processed off-line by using batch processing. First, 3D point cloud models are constructed from the collected images. Second, structure preserving model compression reduces the model size not only for computational and memory requirements, [27], [28], [29] but also for minimizing communication overheads for IoV systems. In addition, the models are updated with newly arrived images. In this study, a model pool is constructed for update. This has been discussed in further detail in Section IV-D.

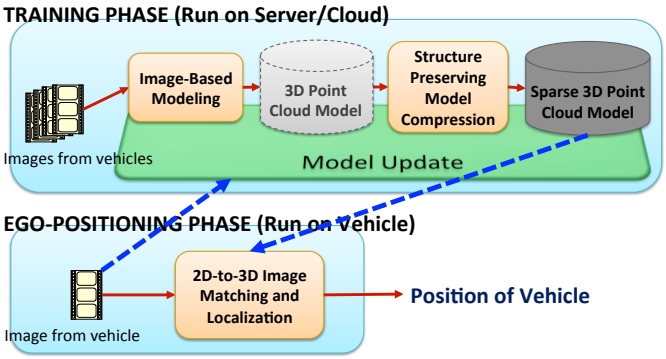


Fig. 2. Overview of the proposed vision-based positioning system. The blue dotted lines denote wireless communication links.

The ego-positioning phase is carried out on vehicles. The 2D-to-3D image matching and localization component matches images from a vehicle with the corresponding 3D point cloud model of an area roughly localized by GPS to estimate 3D positions. There are two communication links in the proposed system. One is to download 3D point cloud models to a vehicle, or preload them when a fixed driving route is provided. However, the size of each uncompressed 3D point model (ranging from 105.1 MB to 12.8 MB in our experiments) is critical as numerous models are required for real-world long-range trips. The other link is to upload images to a cloud server for model update.

IV. Vision-Based Positioning System

There are four main components in the proposed system; they are described in detail in the following sections.

A. Image-based Modeling

Image-based modeling aims to construct a 3D model from a number of input images [33]. A well-known image-based modeling system is the $PhotoTourism$ [10] approach, which estimates camera poses and 3D scene geometry from images by using the structure from motion (SfM) algorithm. Subsequently, a GPU and multicore implementation of SfM and linear-time incremental SfM are developed for constructing 3D point cloud models from images [11], [12].

After images from vehicles in a local area are acquired, the SIFT [34] feature descriptors between each pair of images are matched using an approximate nearest neighbors kd-tree. RANSAC is then used to estimate a fundamental matrix for removing outliers that violate the geometric consistency.

Next, an incremental SfM method [12] is used to avoid poor local minimal solutions and reduce the computational load. It recovers the camera parameters and 3D locations of feature points by minimizing the sum of distances between the projections of 3D feature points and their corresponding image features based on the following objective function:

$$\min_{c_j,P_i} \sum_{i=1}^{n} \sum_{j=1}^{m} v_{ij} d\left(Q\left(c_j, P_i\right), p_{ij}\right), \tag{1}$$

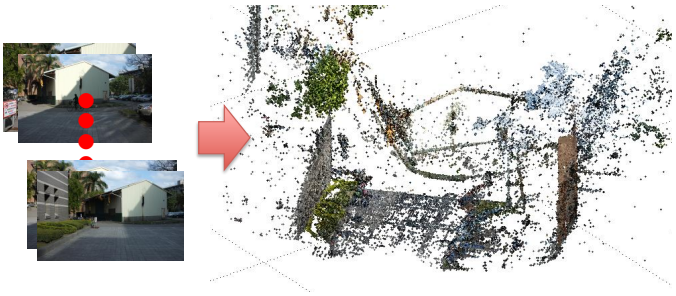
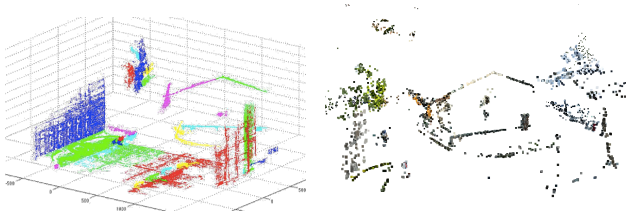


Fig. 3. An example of image-based modeling from 400 images. Given a set of images, the 3D point could model is constructed based on SfM.



(a) (b)

Fig. 4. (a) An example of plane and line detections. (b) An example of point cloud model after model compression.

where c_j denotes the camera parameters of image j, m is the number of images, P_i denotes the 3D coordinates of feature point i, n is the number of feature points, v_{ij} denotes the binary variables that equal 1 if point i is visible in image j and 0 otherwise, $Q(c_j, P_i)$ projects the 3D point i onto the image j, p_{ij} is the corresponding image feature of i on j, and $d(.)$ is the distance function.

This objective function can be solved by using bundle adjustment. A 3D point cloud model **P** is constructed. It includes the positions of 3D feature points and the corresponding SIFT feature descriptor list for each point. An example of the proposed image-based IoV positioning system is shown in Fig. 3.

B. Structure Preserving Model Compression

Compressing the model in a manner that it retains the essential information of an ego-positioning model is a key issue in reducing the computational, memory, and communication loads. A feasible approach to address this issue is to sort the 3D points by their visibility and keep the most observed points. However, it may result in non-uniform spatial distribution of the points, as the points visible to many cameras are likely to be distributed in a small region in the image. Positioning based on these points thus leads to inaccurate estimation of the camera pose. A few methods [30], [27], [28], [29] address this problem as a set k-cover problem to find a minimum set of 3D points that ensures at least k points are visible in each camera view. However, these approaches focus on point cloud coverage and do not consider the non-uniform spatial distribution in 3D structures with each view, as discussed above. The situation deteriorates when we use the local area models constructed in our system. In this work, we address these issues with a weighted set k-cover algorithm where the weights are given to ensure that the selected points are fairly distributed on all planes and lines in the area. The compressed model better preserves the spatial structure of the scenes, which is crucial for our system to achieve sub-meter positioning accuracy.

Let a 3D point cloud reconstructed be $\mathbf{P} = \{P_1,...,P_N\}$, $P_i \in \mathbb{R}^3$ and N is the total number of points. First, we detect planes and lines from a 3D point cloud by using the RANSAC algorithm [35]. This method selects points randomly to generate a plane model and evaluate it by counting the number of points consistent to that model (*i.e.* with the point-to-plane distance smaller than a threshold). This procedure is repeated numerous times, and the plane with the most number of consistent points is selected as the detected plane. After a plane is detected via this method, the 3D points belonging to this plane are removed, and the next plane is detected from the remaining points. This process is repeated until there are no more planes with sufficient consistent number of points. Line detection is carried out after plane detection in a similar manner unless two points are selected to generate a line hypothesis in each iteration. An example of plane and line detections from the model in Fig. 3 is shown in Fig. 4(a).

By using the RANSAC with the one-at-a-time strategy (one line or one plane), the detected planes and lines are usually distributed evenly over the scenes because the main structures (in terms of planes or lines) are identified first. This facilitates selecting the non-local common-view points for compressing a 3D point model. The planes and lines detected in the original point cloud model are denoted as r_l $(l = 1 \cdots L)$, where L is the total number of planes and lines. The remaining points that do not belong to any plane or line are grouped into a single category $L+1$. Given a 3D point P_i in the 3D reconstructed model, we initially set its weight w_i as follows:

$$w_i = \begin{cases} \sigma_l, & \text{if } P_i \in r_l, \\ \sigma_{L+1}, & \text{otherwise}, \end{cases} \tag{2}$$

with

$$\sigma_l = |\{P_i | P_i \in r_l, \forall i\}| / N, \tag{3}$$

$$\sigma_{L+1} = |\{P_i | P_i \notin r_l, \forall i, \forall l\}| / N. \tag{4}$$

That is, the plane or line having a larger portion of points is given a higher weight for point selection.

Based on the weights computed by (2), a weighted set k-cover problem is formulated. However, the minimum set k-cover problem is NP-hard. Therefore, we use a greedy algorithm to solve it efficiently. Let $\{c_1,...,c_m\}$ be the m camera views for the 3D model reconstruction in Section IV-A. We aim to select a subset of points from **P** such that there are at least k points viewable for each camera, and the sum of weights of the selected points is maximized.

Following this greedy principle for solving a set cover problem, our method iteratively selects the most visible point until at least k points are selected for every camera. We first

Algorithm 1 Structure preserving model compression

Input: 3D point cloud model $\mathbf{P} = \{P_1,...,P_N\}$, cameras $\{c_1,...,c_m\}$, integer k

Initialize the compressed model $\mathbf{M} \leftarrow \emptyset$, number of covered points $C[j] = 0$, for all camera c_j
Detect planes and lines $r_1,...,r_L$ in $\mathbf{P}$ by RANSAC
Assign weight w_i to each point P_i by (2)
while $C[j] < k$, for all camera c_j **do**
    Select P_s by (5)
    $\mathbf{M} \leftarrow P_s$
    $\mathbf{P} \leftarrow \mathbf{P} \backslash P_s$
    for all cameras c_j **do**
        if $P_s \in c_j$ **then**
            $C[j] = C[j] + 1$
        end if;
    end for;
    for all 3D points P_i **do**
        if $P_s \in r_l$ and $P_i \in r_l$ **then**
            $w_i = w_i/2$
        end if;
        if $P_s \notin c_j$ for all j s.t. $C[j] < k$ **then**
            $w_i = 0$
        end if;
    end for;
    Normalize w_i
end while;
Return: compressed model $\mathbf{M}$

find one point from $\mathbf{P}$. This point (denoted as P_s) satisfies the following criterion:

$$P_s = \underset{P \in \mathbf{P}}{\operatorname{argmax}} \sum_{j=1}^{m} w_i v(P, c_j), \tag{5}$$

with

$$v(P, c_j) = \begin{cases} 1, & \text{if } P \in c_j, \\ 0, & \text{otherwise}, \end{cases} \tag{6}$$

being the visibility of the point P from the j-th camera. Hence, P_s, as shown in (5), is the most commonly visible point in terms of the respective weights. We then remove P_s from $\mathbf{P}$ by using $\mathbf{P} \leftarrow \mathbf{P} \backslash P_s$ in the proposed greedy approach. Our approach continues to identify the most visible point (P_s) based on the new point set $\mathbf{P}$; this procedure is iterated accordingly.

For the method described in the above paragraph, we assume that the weights $\{w_i\}$ are fixed. Although the planes and lines identified by our approach are usually distributed over the scene, the selected points may still be centralized in a local region inside a single plane or line. Thus, we propose to adapt the weights in every iteration of our greedy algorithm. Once the point P_s is selected in an iteration, we reduce the probability of this point being part of a line or a plane as follows. If $P_s \in r_l$, then w_i is divided by 2 for all points $P_i \in r_l$ in the next iteration. In this way, the plane or line from which the points have already been selected are weighted less, and the other planes or lines will have a higher probability of being selected. Algorithm 1 describes the main steps of this method. An example of model compression is shown in Fig. 4(b), where the main structure is maintained as most of the noisy points are removed The model size is significantly reduced from 105.1 MB to 14.4 MB.

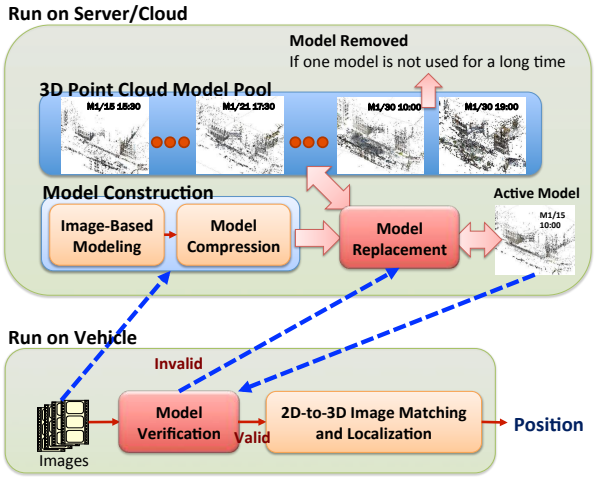
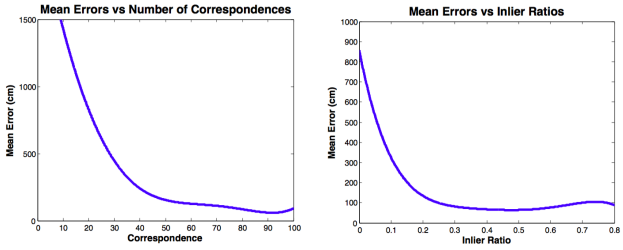


Fig. 5. Overview of our model update algorithm.



(a) (b)

Fig. 6. The distributions of positioning error (a) to number of correspondences and (b) to inlier ratio, respectively.

When a 3D point cloud model is constructed in advance, a visual word tree can be constructed for efficient feature matching in the ego-positioning phase. A visual word is a cluster of similar feature descriptors grouped by the k-means algorithm. In this work, a kd-tree is constructed for fast indexing.

C. 2D-to-3D Image Matching and Localization

Given a test image, the interesting 2D points are detected and their SIFT descriptors are computed. The 2D-to-3D matching determines the correspondence of the 2D points in the test image and the 3D points in the compressed model. The camera position can then be estimated based on the correspondence. In our approach, a 2D-to-3D correspondence is determined if the first and second nearest neighbors in the terms of descriptors pass the ratio test with a threshold (e.g., 0.7 in this work). To speed it up, a prioritized correspondence search [14] is applied. After finding correspondences $\{\{p_1, P_1\}, ..., \{p_{N_c}, P_{N_c}\}\}$, where p is the corresponding 2D feature point of P and N_c is number of correspondences, we

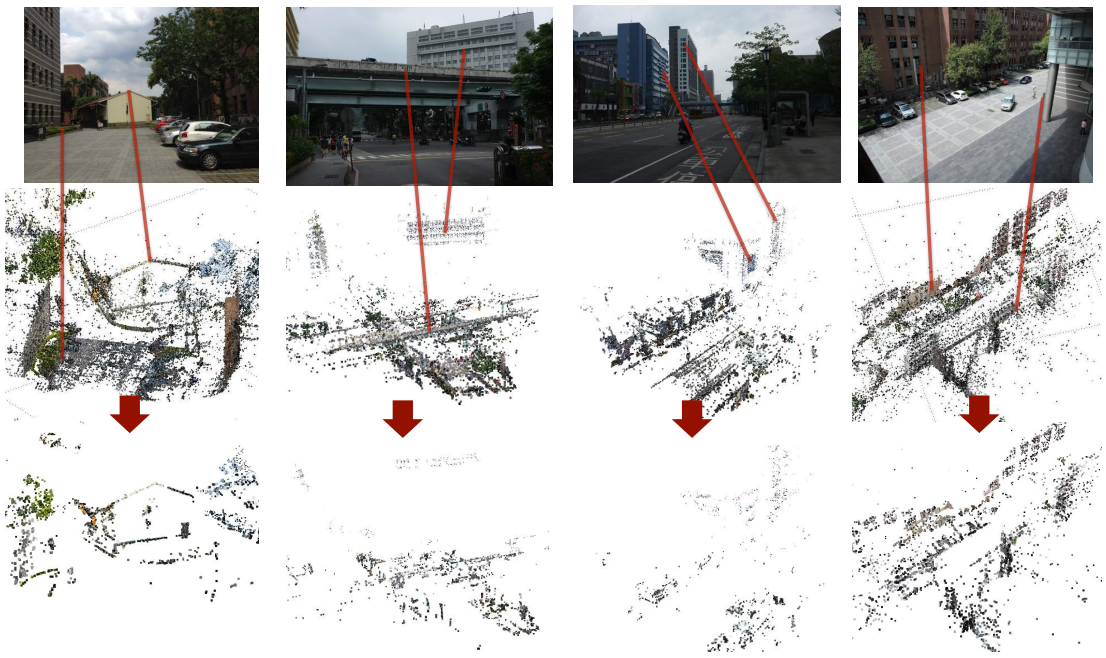


Fig. 7. Four scenes: Scene #1, Scene #2, Scene #3, and Scene #4 from left to right respectively. The first row is the images of the scenes. The second row shows the constructed models, and the compressed models by our method are shown in the third row.

use the 6-point Direct Linear Transform (DLT) algorithm [36] with RANSAC to compute the camera pose.

D. Model Update

As the outdoor scenes are likely to change owing to lighting, occlusion and other factors, it is necessary to update the model such that a test image can be well matched for camera pose estimation. In contrast to the existing methods that assume 3D models are up-to-date, we tackle this problem by presenting an update method for real-world scenarios.

Fig. 5 shows an overview of the proposed model update algorithm. Instead of using only one 3D point cloud model, a pool of them constructed at different scenarios and time is maintained for every local area in our system. Here, we do not combine more observations into a single model as in addition to increasing the size, it might introduce more ambiguities while matching features. Furthermore, only one active model is selected and transmitted in our model update method. There are two main components in the update process: model verification and model replacement. In the ego-positioning phase, the model verification component verifies whether the input image can be registered properly with the model by using the following criterion,

$$(N_c > T_1) \text{ and } (N_I/N_c > T_2), \tag{7}$$

where N_c is number of correspondences, and N_I is the number of inliers determined by the DLT algorithm. Furthermore, T_1 and T_2 are acquired from the training data; in our experiments they are set to 50 and 0.5, respectively. Fig. 6 shows the positioning error with respect to the number of correspondences and the inlier ratio from our experiments. The positioning error decreases when the number of correspondences and inlier ratio are increased up to certain number. The results also show why the model compression works as only a sufficiently large number of point correspondences are used for camera pose estimation.

If multiple observations cannot be matched properly by the current model (i.e., large errors in camera pose estimation), it is replaced by one from the pool. First, all the models in the pool are evaluated by (7) with the test images. Second, the best matched model is selected as the new active one if the number of correspondences is above a threshold. Third, if no model satisfies the criterion, a new model is constructed by using the images collected in this session to replace the current one. Finally, if a model in the pool is not used for a long duration, it is removed so that the pool size does increase too much.

Empirically, we determine that a pool of eight models is sufficient for daytime scenarios. In addition, it is rare for a new model to be created to replace an existing one.

V. Experimental Results

In this section, we first evaluate the positioning of a single still image with both local and up-to-date models in four scenes and compare the proposed algorithm with state-of-the-art methods [28], [30]. Next, image sequences with ground truth positions in a controlled environment are evaluated. Third, the proposed algorithm is evaluated using real-world image sequences. Finally, a dataset with more than 14,000 images of session data with 106 sessions is constructed for evaluation of the proposed algorithm with model update.

The ground truth positions in our experiments are all measured manually in real scenes. For each scene, we define a world coordinate system with x- and y-axes on the ground

Fig. 8. Example test images in four scenes.

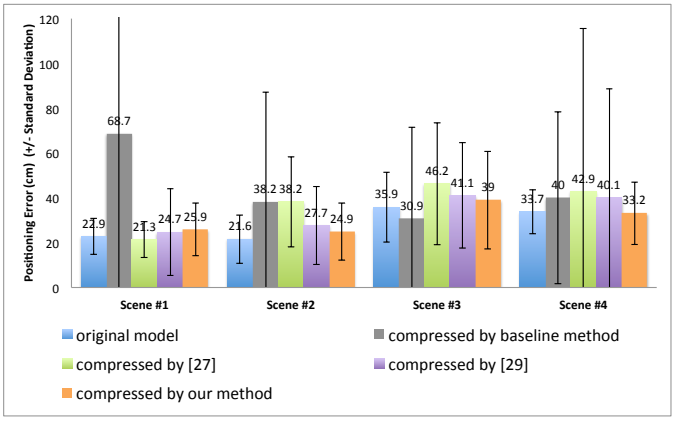


Fig. 9. Positioning error (cm) (+/- standard deviation) of single still image after model compression by different methods.

plane. We measure the 2D coordinates of cameras with a tape measure while acquiring images. As each 3D model is constructed by the SfM algorithm with a monocular camera, the scale of the model compared to real world can be estimated. For each sequence, at least four training images are acquired with manually measured ground truth to estimate the transformation between the image and real world coordinates based on the 3D model. This process can also be carried out by camera auto-calibration algorithms [36].

Unlike the recent methods [24], [25], [26] using the image results generated offline by an SfM algorithm as the ground truth, all test images are registered by using physical measurements. We note that although pose estimations can be assessed accurately in terms of image pixels by using SfM algorithms, these results do not correspond to physical metrics (e.g., meter or foot). As the goal of SfM algorithms is to build the 3D models, images with insufficient feature correspondences are removed. Therefore, only the images which can be matched well during SfM are retained and evaluated. For example, less than 50% images are registered in the night scene after SfM in our experiments. If we use the results of SfM as the ground truth, only the images with features matched are evaluated.

A. Positioning Evaluation of Single Still Images

We demonstrate the ego-positioning performance of the proposed algorithm by evaluating it in four different scenarios. Fig. 7 shows the scenes, uncompressed and compressed 3D point cloud models by our algorithm. The models are reconstructed with 400, 179, 146, and 341 training images, respectively. The test images are acquired in the same session

TABLE I
POSITIONING ERROR (CM) OF SINGLE STILL IMAGE AND NUMBER OF POINTS AND MODEL SIZE AFTER MODEL COMPRESSION. RED AND BLUE FONT SHOW THE MINIMUM AND SECOND MINIMUM MEAN, STANDARD DEVIATION., OR MODEL SIZE OF FOUR COMPRESSION METHODS.

Scene		#1	#2	#3	#4
original model	Mean	22.9	21.6	35.9	33.7
	Stdev.	8.1	10.8	15.6	9.8
	# points	187,572	53,568	33,190	85,447
	Size(MB)	105.1	21.5	12.8	26.4
compressed by baseline method	Mean	68.7	38.2	**30.9**	40.0
	Stdev.	143.5	49.2	40.5	38.3
	# points	8,397	4,329	3,268	8,263
	Size(MB)	**10.2**	2.5	1.5	5.4
compressed by [28]	Mean	**21.3**	38.2	46.2	42.9
	Stdev.	**7.9**	20.1	27.1	72.8
	# points	8,580	5,101	3,345	8,272
	Size(MB)	17.8	3.0	1.6	4.8
compressed by [30]	Mean	24.7	27.7	41.1	40.1
	Stdev.	19.4	17.5	23.5	48.5
	# points	8,519	4,557	3,149	8,166
	Size(MB)	19.9	2.7	1.7	5.0
compressed by our method	Mean	25.9	**24.9**	39.0	**33.2**
	Stdev.	11.7	**12.9**	**21.8**	**14.0**
	# points	7,781	4,351	3,228	8,196
	Size(MB)	14.4	**2.2**	**1.5**	**4.3**

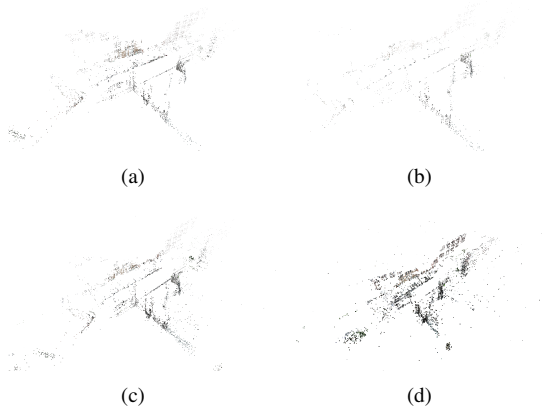


Fig. 10. Model compression results of (a) baseline method, (b) Li's method [28], (c) Cao's method [30] and (d) our method, in scene #4.

of the training images. There are 50, 24, 28, and 88 test images in 4 scenes, and some examples are shown in Fig. 8.

We evaluate the k-cover method [28] (k = 720, 500, 250, and 200 for each scene), the probabilistic k-cover approach [30] (k = 430, 300, 250, and 200 for each scene), and the proposed weighted set k-cover algorithm (k = 500, 300, 370, and 185 for each scene) where we select a proper value of k for each method to ensure that the number of points after model compression is similar for all methods for fair comparisons. More details about the number of points and model sizes are shown in Table I. Fig. 9 shows the positioning results with comparisons to different compression methods where the baseline scheme is to set to be 5% (for scene #1) or 10% (for other scenes) mostly seen points on each plane or line.

As shown in Fig. 9, high positioning accuracy can be achieved when model compression is performed for a local

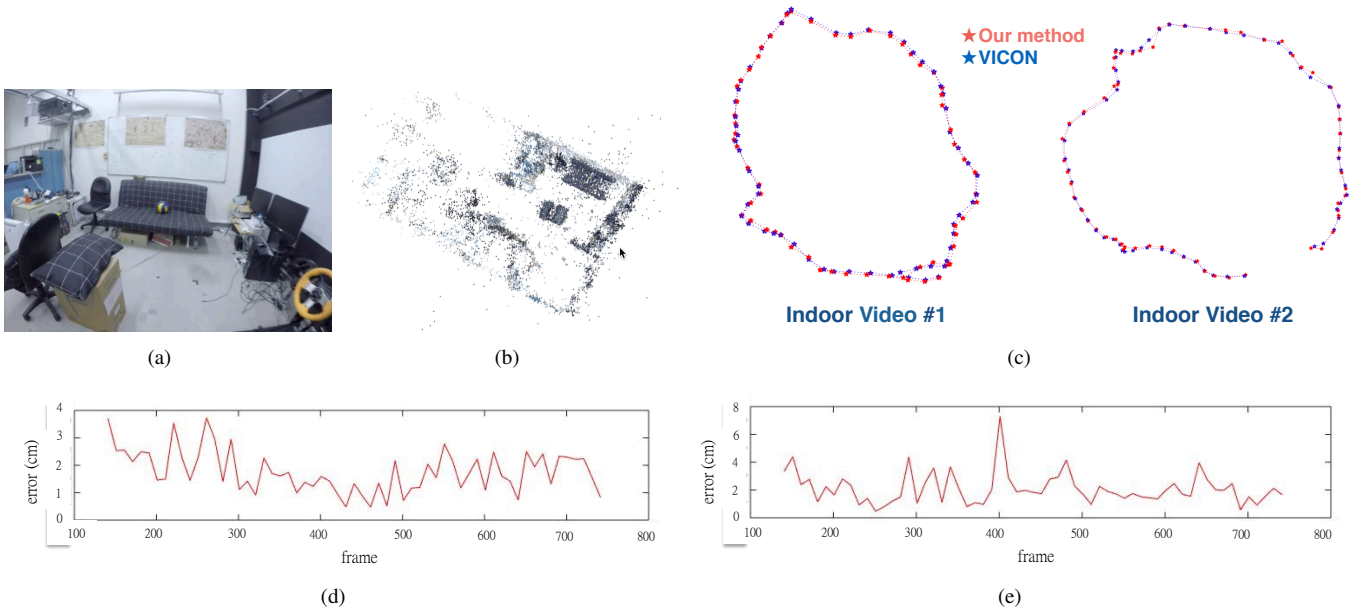


Fig. 11. Evaluation results using image sequences from an indoor environment. (a) Sample image acquired in the environment. (b) Constructed model from two training image sequences. (c) Results of our method compared with those estimated by Vicon. (d)-(e) Error distribution of each frame for indoor video #1 and #2.

and up-to-date model by the proposed system. Furthermore, the proposed system performs better in terms of positioning accuracy and model size reduction, when compared to other methods; thus, more stable positioning results can be obtained when an essential structure is preserved from model compression (or equivalently noisy features are removed). The proposed system efficiently performs large reduction of model size with compression ratio of 86%, 90%, 88%, and 83% for four scenes, respectively. In addition, the standard deviation of the positioning error by the proposed system is smaller than that of the other methods. Fig. 10 shows the compressed models for scene #4, and our system maintains more structural information than other methods.

As the main goal of this study is to achieve sub-meter accuracy for vision-based ego-positioning, real-time processing is not the prime concern. The average execution time of an image with 900 $\times$ 600 pixels on a desktop computer with Intel Core i7-4770K processor and 16G ram is 1.4089 seconds (including SIFT feature extraction 1.274 seconds, feature matching 0.1319 seconds, and RANSAC pose estimation 0.0003 seconds). Most of the computational load is from extraction of SIFT features, which can be improved by GPU or multicore implementations. For example, Wu [37] implements SiftGPU to construct Gaussian pyramids and detect DoG keypoint for SIFT with GPU, and shows about 20 fps can be achieved for images with 800 $\times$ 600 pixels. In addition, other features may be tried to improve system efficiency such as ORB features [38] which have achieved similar performance but much faster than SIFT in certain tasks.

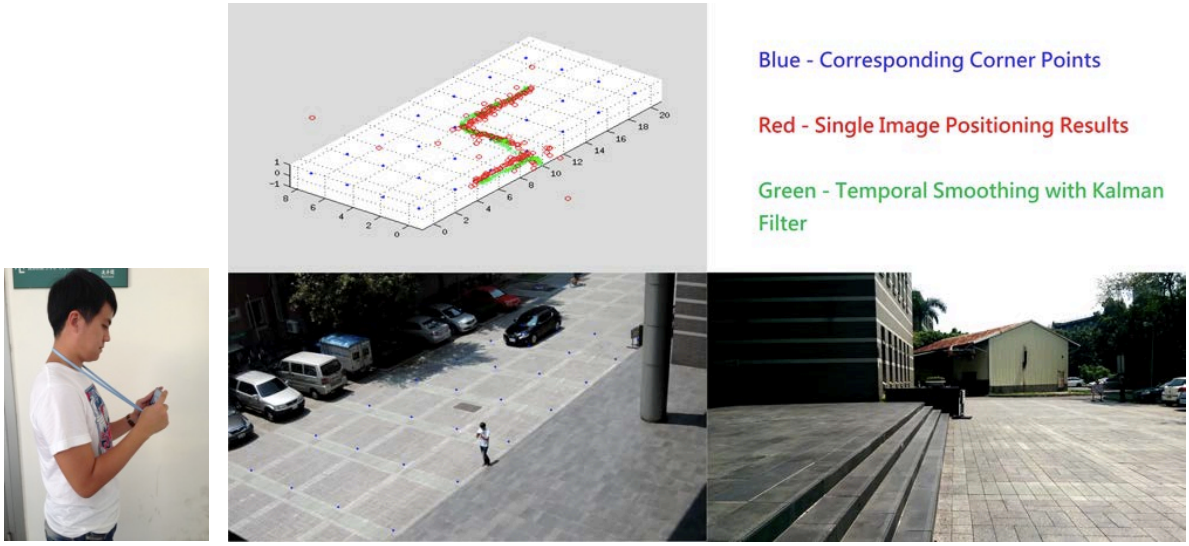
B. Positioning Evaluation of Image Sequences

We evaluate two image sequences taken by a smartphone in an indoor environment where we install four Vicon cameras to provide ground truth positions. The error of the Vicon motion capture system is less than 1 mm, and we use it as the ground truth for higher precision evaluation. Fig. 11(a) shows the environment, and Fig. 11(b) shows the constructed model by two training image sequences. We test two image sequences taken by a person walking along a loop where the camera is pointed either at the chair in the center or forward. Fig. 11(c) shows the estimated trajectories by our method (red points) and Vicon (blue points). Fig. 11(d) and (e) shows the frame-by-frame error of our method for both image sequences. The mean and standard deviation of positioning error are 1.78 cm and 0.77 cm, and 2.09 cm and 1.14 cm, respectively. The test image sequences and positioning results of our method are available at https://youtu.be/iLJXzCCllCQ (video #1).

TABLE II
POSITIONING ERROR (CM) OF IMAGE SEQUENCES.

Image sequence		#1	#2
Frame number		200	281
Single still image	Mean	60.1	88.4
	Stdev.	53.8	116.7
Temporal smoothing	Mean	37.2	41.8
	Stdev.	18.3	26.3

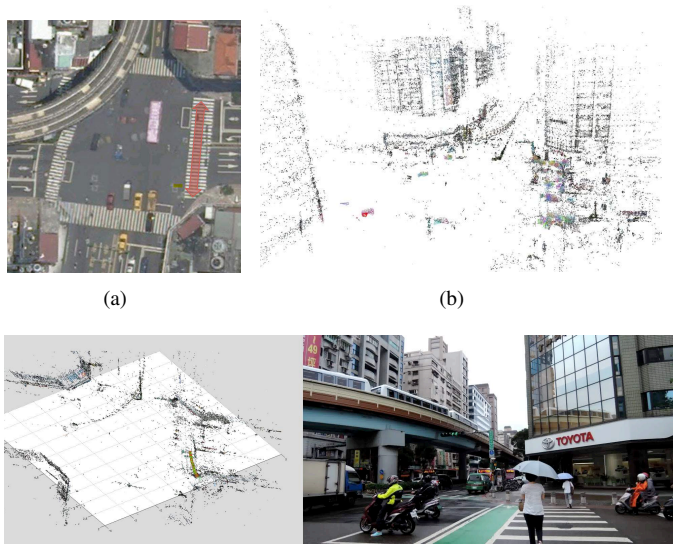
We then evaluate two image sequences in scene #4. For each experiment, we record the sequences from two cameras at 10 frames per second. One sequence is acquired by a camera placed on the third floor of a nearby building, which gives a bird's-eye view, as shown in the bottom left panel of Fig. 12(b), We manually label the coordinates of the person's foot on the ground plane in each image and transform the coordinates to the ground plane as the ground truth positions. The other sequence is acquired by a smartphone, and it gives



(a) (b)

Fig. 12. (a) Setup for acquiring image sequences. (b) An example of results of video evaluation. Bottom-right: image from smartphone; bottom-left: image from camera on third floor; upper-left: positioning results.

Fig. 13. Evaluation results using image sequences. Right column: an image from a dash camera. Left column: an image from a camera on third floor where red and green circles are the positioning results with still images and the results after temporal smoothing by the Kalman filter.



(a) (b)

(c)

Fig. 14. Evaluation results using image sequences from urban scenes. (a) The aerial view and the red arrow shows the test routes. (b) The constructed model from 235 images. (c) An example of positioning results. The image on the right hand side is the image taken by a smartphone, and the image on the left hand side shows the estimated positions projected onto the model, where the red and green circles are the positioning results with still images and the results after temporal smoothing by the Kalman filter, respectively.

the first-person view (bottom right panel of Fig. 12(b)), which is used for vision-based ego-positioning. The upper left panel of Fig. 12(b) shows the estimation results, where the red and green circles are the positioning results with still images and the results after temporal smoothing by the Kalman filter [39], respectively. The positioning results can be found at https://youtu.be/iLJXzCCllCQ (video #2 and #3) and Table II shows the quantitative results. Due to occlusions or motion blurs in the videos, there are outliers or noisy estimations by positioning using still images, and they can be smoothed out temporally using sequences.

In addition, we evaluate the proposed algorithm on image sequences acquired from a dash camera on a moving vehicle. Fig. 13 shows sample results; additional results can be seen in the demo video (video #4) on the above mentioned website. Although we do not have quantitative results for these image sequences as it is difficult to determine the ground truth positions by using a dash camera, experimental results show that high accuracy is achieved as the estimated positions match the that of vehicle well. Please note that the test video is recorded on different days and the proposed model update algorithm is used for feature matching. More results of model update are shown in Section V-E.

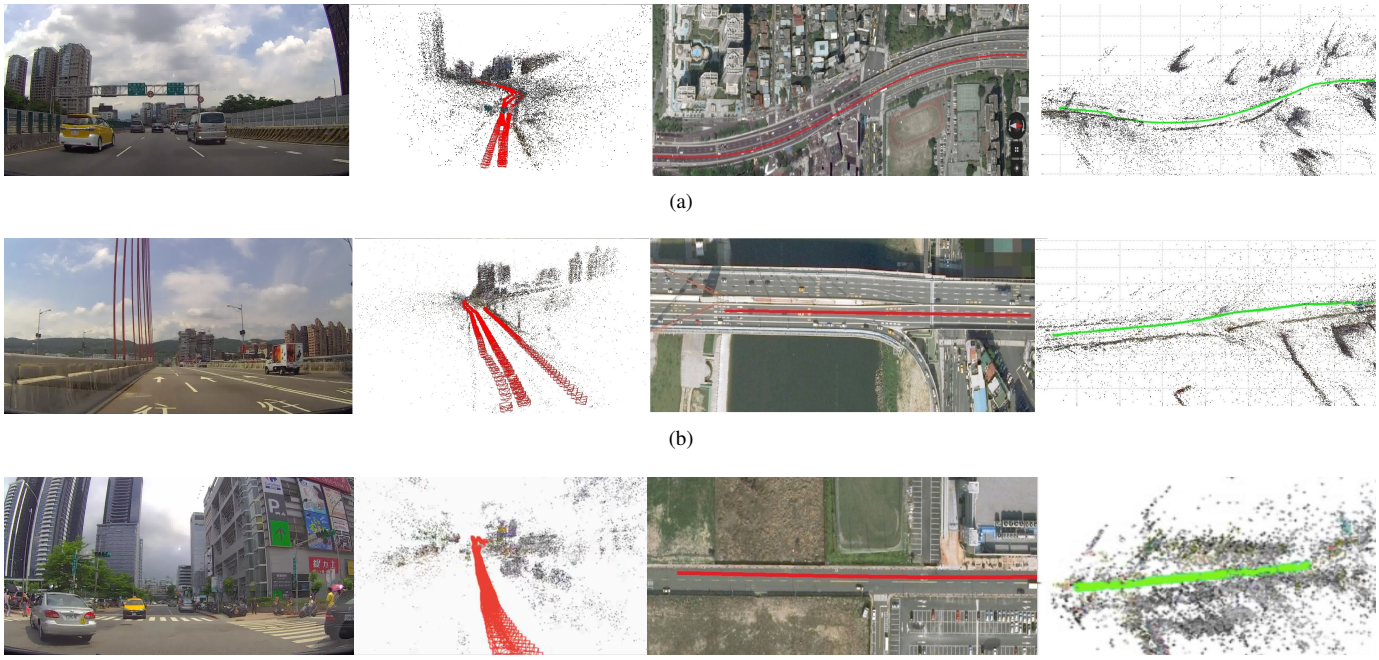


Fig. 15. Evaluation results using sequences at two traffic scenes: (a) expressway, (b) bridge, (c) downtown. First column: sample images used for training. Second column: the reconstructed 3D models, where the red lines indicate the motion paths of training videos. Third column: aerial views of each scene. Fourth column: ego-positioning results where the green lines are the estimated positions with temporal smoothing applied.

TABLE III
POSITIONING ERROR (CM) OF THE SESSION DATA IN OTHER MONTHS.

Test date & time		M2/3 10:30	M2/4 17:30	M2/5 15:00	M2/6 12:00	M2/7 14:00	**M2/8 22:30**	**M2/9 18:30**	M2/10 10:00	**M3/4 14:30**
Weather		Cloudy	Dusk	Sunny	Sunny	Cloudy	**Night**	**Night**	Cloudy	**Rainy**
Fixed model (M1/20 15:30)	Mean	72.4	NaN	74.2	171.4	91.6	**1136**	**NaN**	135.9	**4107.8**
	Stdev.	40.4	NaN	42.5	318.6	174.7	**850**	**NaN**	180.2	**1264.7**
Model update applied	Mean	30.6	39.3	33.7	28.2	30.8	**441.5**	**34.2**	41.5	**36.2**
	Stdev.	12.5	16.8	12.8	11.7	14.3	**352.1**	**28**	13.3	**15.8**
	Selected model & weather	M1/27 16:00 Cloudy	M1/30 19:00 Night	M1/30 14:30 Sunny	M1/16 11:00 Sunny	M1/27 16:00 Cloudy	**M1/30 19:00 Night**	**M1/30 19:00 Night**	M1/27 16:00 Cloudy	**M1/27 16:00 Cloudy**

C. Positioning Evaluation in Real Traffic Scenes

We evaluate the proposed system on seven image sequences acquired in four real traffic scenes. The first test contains images collected at an intersection in an urban area. Fig. 14(a) shows its aerial view of the scenes. We use 235 images for training and two image sequences acquired from a smartphone on the routes shown by the red arrow in Fig. 14(a) for evaluation. Fig. 14(b) and (c) show the constructed model and sample positioning results. In the other traffic scenes (expressway, bridge, and downtown), the training (three for each scene) and test (two for expressway, two for bridge, and one for downtown) image sequences are acquired from a dash camera. During the test phase, the car is allowed to change lanes and our method performs well as shown in the first two columns of Fig. 15. Thus, for vehicles traveling on multi-lane highways, it is feasible to use a model with training data covering these multiple lanes. Once 3D models are constructed, the corresponding vehicle positions from different lanes can be estimated as long as features are extracted and matched. While the view angles of acquired images change to certain extent when vehicles change lanes, SIFT features are known to be robust in accounting for large pose variation (e.g., view angle change from different cameras within 20 degrees). Although the ground truth of the test videos is not available, the positioning results qualitatively match the scenes well as shown at https://youtu.be/iLJXzCCllCQ (video #5, #6, #7 and #8).

D. Long-Term Positioning Dataset

To evaluate the performance of ego-positioning methods over a long duration of time, a large amount of data is collected over two months. It contains videos collected from 106 sessions with 14,275 images and 9,720 of them are manually measured with ground truth positions. To the best of our knowledge, this is by far the largest dataset that contains videos acquired in different lighting conditions (e.g., sunny,

date	Month 1																	Month 2										Month 3
time	8	9	13	14	15	16	17	20	21	22	23	24	26	27	28	29	30	31	2	3	4	5	6	7	8	9	10	4
9												108			108													
10					108			108	108		108			108		108	277			108		301					108	
11				108		373									108	108					108		108					
12		108	108							108		108								108		108	108				108	
13						108					108					108	108		108		108	108		237	108		108	
14	108					334		108		108				108	255		373	108		281	108		108	108			108	
15				108	108	108		288	108				108		108							108						108
16							108	108		108		108		376	108	108	108	292		108	108		108	230			108	
17		108	108						108	108	108			108			108				108	108						
18				108					263									284							108	108	108	
19			108						108			108					391			108	108							
20										108							108		108									
21				108																								
22								108	108		108	108						108							108			
23																							108					
24					108																							

Sunny Cloudy Rainy Dusk Night

(a)

(b)

Fig. 16. Long-term positioning dataset. (a) Session distribution with the number of images. (b) Example images.

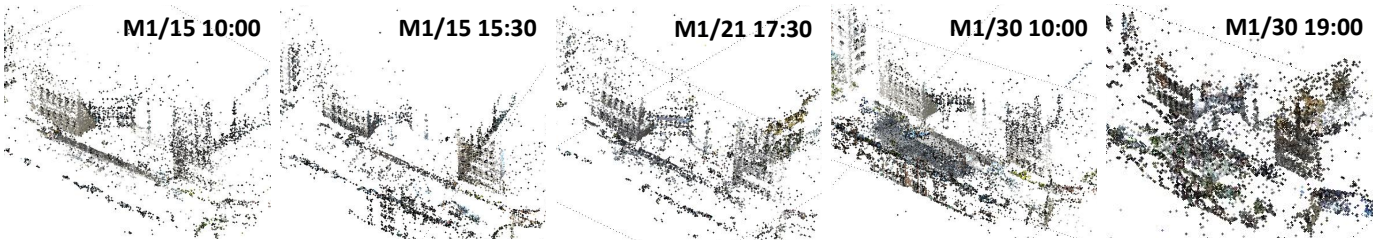


Fig. 17. Constructed 3D point models using images acquired at different sessions.

rainy, day, and night scenes). The session distribution and some sample images are shown in Fig. 16, and more details can be seen at http://imlab.tw/positioning/dataset/.

E. Positioning Evaluation with Model Update

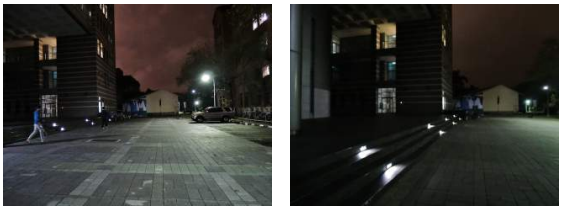
We demonstrate the effectiveness of the proposed model update algorithm by using a long-term dataset collected over the span of two months. At the end of the first month, there are 21 models in the model pool, of which, 8 models are for daytime, and the others for night scenes. Fig. 17 shows some of constructed 3D models. It is clear that a few models are sufficient for accurate ego-positioning in the daytime but it is difficult to register images at night. Table III presents evaluation results using test images acquired in other months. Note that all models are selected automatically with the first ten images in a session by using the proposed update algorithm discussed in Section IV-D. Experimental results in Table III show that our method can select a suitable model, and higher positioning accuracy can be achieved using the model update approach, e.g., the improvement at M3/4 14:30 (the rightmost column in the table) is hundredfold. The positioning accuracy with a static model is significantly inferior and cannot be registered in some sessions (e.g. M2/4 17:30). Furthermore, we find positioning in the night scenes even more challenging. The proposed algorithm performs well on night scenes (columns in blue) with sufficient lighting, e.g., M2/9 18:30 shown in Fig. 18(a). However, it does not perform well for scenes under poor lighting conditions (e.g. M2/8 22:30 shown in Fig. 18(b)).

VI. Conclusions

In this paper, we propose a vision-based ego-positioning system within the IoV framework. To achieve higher positioning accuracy with lower computational load, we compress 3D point cloud models by using the proposed weighted k-cover method, which preserves the essential structural information. To account for large scene changes over time, we present a model update algorithm. In addition, a large long-term positioning dataset is developed for performance evaluation. Experimental results show that sub-meter positioning accuracy can be achieved based on the compressed models for both still images and videos over long durations of time.

Acknowledgments

This work is supported in part by the Ministry of Science and Technology and National Taiwan University under Grants MOST 105-2633-E-002-001, MOST 104-2622-



(a) (b)

Fig. 18. Examples of test images in the night scene at: (a) M2/9 18:30 and (b) M2/8 22:30.

8-002-002, MOST 104-2627-E-002-001 and NTU-ICRP-105R104045. The work of M.-H. Yang is supported in part by the NSF CAREER Grant (No. 1149783).

REFERENCES

[1] K. W. Chen, H. M. Tsai, C. H. Hsieh, S. D. Lin, C. C. Wang, S. W. Yang, S. Y. Chien, C. H. Lee, Y. C. Su, C. T. Chou, Y. J. Lee, H. K. Pao, R. S. Guo, C. J. Chen, M. H. Yang, B. Y. Chen, and Y. P. Hung, "Connected vehicle safety - science, system, and framework," *IEEE World Forum on Internet of Things*, 2014. 1

[2] K. W. Chen, S. C. Chen, K. Lin, M. H. Yang, C. S. Chen, and Y. P. Hung, "Object detection for neighbor map construction in an iov system," *IEEE International Conference on Internet of Things*, 2014. 1

[3] Google, "Google project tango," https://www.google.com/atap/project-tango/. 1

[4] Y. Gu, A. Lo, and I. Niemegeers, "A survey of indoor positioning systems for wireless personal networks," *IEEE Communications Survey and Tutorials*, vol. 11, no. 1, pp. 13–32, 2009. 1

[5] H. Koyuncu and S. H. Yang, "A survey of indoor positioning and object locating systems," *International Journal of Computer Science and Network Security*, vol. 10, no. 5, pp. 121–128, 2010. 1

[6] H. Liu, H. Darabi, P. Banerjee, and J. Liu, "Survey of wireless indoor positioning techniques and systems," *IEEE Transactions on Systems, Man, and Cybernetics - Part C*, vol. 37, no. 6, pp. 1067–1080, 2007. 1

[7] T. Driver, "Long-term prediction of gps accuracy: Understanding the fundamentals," *ION GNSS International Technical Meeting of the Satellite Division*, 2007. 1

[8] M. Modsching, R. Kramer, and K. Hagen, "Field trial on gps accuracy in a medium size city: The influence of built-up," *Workshop on Positioning, Navigation and Communication*, 2006. 1

[9] A. Rietdorf, C. Daub, and P. Loef, "Precise positioning in real-time using navigation satellites and telecommunication," *Workshop on Positioning, Navigation and Communication*, 2006. 1

[10] N. Snavely, S. Seitz, and R. Szeliski, "Photo tourism: Exploring photo collections in 3d," *ACM Transactions on Graphics*, vol. 25, no. 3, pp. 835–846, 2006. 2, 3

[11] C. Wu, S. Agarwal, B.Curless, and S. M. Seitz, "Multicore bundle adjustment," *IEEE Conference on Computer Vision and Pattern Recognition*, 2011. 2, 3

[12] C. Wu, "Towards linear-time incremental structure from motion," *International Conference on 3D Vision*, 2013. 2, 3

[13] Y. Li, N. Snavely, D. Huttenlocher, and P. Fua, "Worldwide pose estimation using 3d point clouds," *European Conference on Computer Vision*, 2012. 2

[14] T. Sattler, B. Leibe, and L. Kobbelt, "Fast image-based localization using direct 2d-to-3d matching," *International Conference on Computer Vision*, 2011. 2, 5

[15] ——, "Improving image-based localization by active correspondence search," *European Conference on Computer Vision*, 2012. 2

[16] L. Kneip, M. Chli, R. Siegwart, R. Siegwart, and R. Siegwart, "Robust real-time visual odometry with a single camera and an imu," *British Machine Vision Conference*, 2011. 2

[17] H. Lategahn and C. Stiller, "Vision-only localization," *IEEE Transactions on Intelligent Transportation Systems*, vol. 15, no. 3, pp. 1246–1257, 2014. 2

[18] K. Schmid and H. Hirschmuller, "Stereo vision and imu based real-time ego-motion and depth image computation on a handheld device," *International Conference on Robotics and Automation*, 2013. 2

[19] M. Brubaker, A. Geiger, and R. Urtasun, "Lost! leveraging the crowd for probabilistic visual self-localization," *IEEE Conference on Computer Vision and Pattern Recognition*, 2013. 2

[20] D. M. Chen, G. Baatz, K. Koser, S. S. Tsai, R. Vedantham, T. Pylvanainen, K. Roimela, X. Chen, J. Bach, M. Pollefeys, B. Girod, and R. Grzeszczuk, "City-scale landmark identification on mobile devices," *IEEE Conference on Computer Vision and Pattern Recognition*, 2011. 2

[21] A. Zamir and M. Shah, "Accurate image localization based on google maps street view," *European Conference on Computer Vision*, 2010. 2

[22] C. Arth, D. Wagner, M. Klopschitz, A. Irschara, and D. Schmalstieg, "Wide area localization on mobile phones," *International Symposium on Mixed and Augmented Reality*, 2009. 2

[23] A. Wendel, A. Irschara, and H. Bischof, "Natural landmark-based monocular localization for mavs," *International Conference on Robotics and Automation*, 2011. 2

[24] H. Lim, S. Sinha, M. Cohen, and M. Uyttendaele, "Real-time image-based 6-dof localization in large-scale environments," *International Symposium on Mixed and Augmented Reality*, 2012. 2, 7

[25] S. Middelberg, T. Sattler, O. Untzelmann, and L. Kobbelt, "Scalable 6-dof localization on mobile devices," *European Conference on Computer Vision*, 2014. 3, 7

[26] J. Ventura and T. Hollerer, "Wide-area scene mapping for mobile visual tracking," *International Symposium on Mixed and Augmented Reality*, 2012. 3, 7

[27] A. Irschara, C. Zach, J. Frahm, and H. Bischof, "From structure-from-motion point clouds to fast location recognition," *IEEE Conference on Computer Vision and Pattern Recognition*, 2009. 3, 4

[28] Y. Li, N. Snavely, and D. Huttenlocher, "Location recognition using prioritized feature matching," *European Conference on Computer Vision*, 2010. 3, 4, 6, 7

[29] H. S. Park, Y. Wang, E. Nurvitadhi, J. C. Hoe, Y. Sheikh, and M. Chen, "3d point cloud reduction using mixed-integer quadratic programming," *Computer Vision and Pattern Recognition Workshops*, 2013. 3, 4

[30] S. Cao and N. Snavely, "Minimal scene descriptions from structure from motion models," *IEEE Conference on Computer Vision and Pattern Recognition*, 2014. 3, 4, 6, 7

[31] E. Johns and G. Z. Yang, "Dynamic scene models for incremental, long-term, appearance-based localisation," *International Conference on Robotics and Automation*, 2013. 3

[32] ——, "Feature co-occurrence maps: Appearance-based localisa- tion throughout the day," *International Conference on Robotics and Automation*, 2013. 3

[33] P. Debevec, C. Taylor, and J. Malik, "Modeling and rendering architecture from photographs: a hybrid geometry- and image-based approach," *ACM SIGGRAPH*, 1996. 3

[34] D. Lowe, "Distinctive image features from scale-invariant keypoints," *International Journal of Computer Vision*, vol. 60, no. 2, pp. 91–110, 2004. 3

[35] M. Y. Yang and W. Forstner, "Plane detection in point cloud data," *International Conference on Machine Control Guidance*, 2010. 4

[36] R. Hartley and A. Zisserman, "Multiple view geometry in computer vision," *Cambridge University Press*, 2004. 5, 7

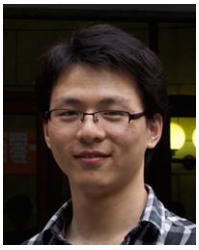
[37] C. Wu, "Siftgpu: A gpu implementation of scale invariant feature transform (sift)," http://www.cs.unc.edu/~ccwu/siftgpu/. 8

[38] E. Rublee, V. Rabaud, K. Konolige, and G. Bradski, "Orb: an efficient alternative to sift or surf," *International Conference on Computer Vision*, 2011. 8

[39] R. Kalman, "A new approach to linear filtering and prediction problems," *Journal of Basic Engineering*, vol. 82, no. 1, pp. 35–45, 1960. 9

Kuan-Wen Chen is an assistant professor in Department of Computer Science at National Chiao Tung University, Hsinchu, Taiwan. He received the B.S. degree in computer and information science from National Chiao Tung University, Hsinchu, Taiwan, in 2004, and the Ph.D. degree in computer science and information engineering from National Taiwan University, Taipei, Taiwan, in 2011. He was an postdoc researcher (from 2012 to 2014) and an assistant research fellow (from 2014 to 2015) in Intel-NTU Connected Context Computing Center at National Taiwan University, Taipei, Taiwan. His current research interests include computer vision, pattern recognition, multimedia, and internet-of-vehicles.

Chun-Hsin Wang received the B.S. degree in electrical engineering from University of Washington, USA, in 2012, and the master degree in Graduate Institute of Networking and Multimedia from National Taiwan University, Taipei, Taiwan, in 2015.

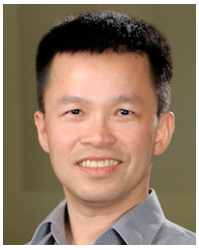


Xiao Wei received the B.S. degree in information display from University of Electronic Science and Technology of China, China, in 2012, and the master degree in Graduate Institute of Networking and Multimedia from National Taiwan University, Taipei, Taiwan, in 2014.

Qiao Liang is a master student in Graduate Institute of Networking and Multimedia at National Taiwan University, Taipei, Taiwan. He received the B.S. degree in information and communication engineering from Beijing Jiaotong University, China, in 2014.

Chu-Song Chen is currently a Research Fellow with the Institute of Information Science and the Research Center for IT Innovation, Academia Sinica, Taiwan. He is an Adjunct Professor with the Graduate Institute of Networking and Multimedia, National Taiwan University. His research interests include computer vision, signal/image processing, and pattern recognition. He is on the Governing Board of the Image Processing and Pattern Recognition Society, Taiwan. He served as an Area Chair of ACCV'10 and NBiS'10, the Program Chair of IMV'12 and IMV'13, the Tutorial Chair of ACCV'14, and the General Chair of IMEV'14, and will be the Workshop Chair of ACCV'16. He is on the Editorial Board of the Journal of Multimedia (Academy Publisher), Machine Vision and Applications (Springer), and the Journal of Information Science and Engineering.

Ming-Hsuan Yang is an associate professor in Electrical Engineering and Computer Science at University of California, Merced. He received the PhD degree in computer science from the University of Illinois at Urbana-Champaign in 2000. Prior to joining UC Merced in 2008, he was a senior research scientist at the Honda Research Institute. Yang served as an associate editor of the IEEE Transactions on Pattern Analysis and Machine Intelligence from 2007 to 2011, and is an associate editor of the International Journal of Computer Vision, Computer Vision and Image Understanding, Image and Vision Computing and Journal of Artificial Intelligence Research. He received the NSF CAREER award in 2012, Senate Award for Distinguished Early Career Research at UC Merced in 2011, and Google Faculty Award in 2009. He is a senior member of the IEEE and the ACM.



Yi-Ping Hung received the B.S. degree in electrical engineering from National Taiwan University, Taipei, Taiwan, in 1982, the M.S. degree from the Division of Engineering, Brown University, Providence, RI, in 1987, the M.S. degree from the Division of Applied Mathematics, Brown University, in 1988, and the Ph.D. degree from the Division of Engineering, Brown University, in 1990. From 1990 to 2002, he was with the Institute of Information Science, Academia Sinica, Taipei, where he became a Tenured Research Fellow in 1997, and is currently a Joint Research Fellow. He served as the Deputy Director of the Institute of Information Science from 1996 to 1997, and the Director of the Graduate Institute of Networking and Multimedia with National Taiwan University from 2007 to 2013. He is currently a Professor with the Graduate Institute of Networking and Multimedia, and the Department of Computer Science and Information Engineering, National Taiwan University. His current research interests include computer vision, pattern recognition, image processing, virtual reality, multimedia, and human-computer interaction. He was the Program Cochair of ACCV'00 and ICAT'00, and the Workshop Cochair of ICCV'03. He has been an Editorial Board Member of the International Journal of Computer Vision since 2004. He will be the General Chair of ACCV'16.