Point of Interest Detection and Visual Distance Estimation for Sensor-rich Video

Jia Hao, Guanfeng Wang, Beomjoo Seo, Roger Zimmermann

***Abstract*—Due to technological advances and the popularity of camera sensors it is now straightforward for users to capture and share videos. A large number of geo-tagged photos and videos have been accumulating continuously on the web, posing a challenging problem for mining this type of media data. In one application scenario users might desire to know what the Points of Interest (POI) are which contain important objects or places in a video. Existing solutions attempt to examine the signal content of the videos and recognize objects and events. This is typically time-consuming and computationally expensive and the results can be uneven. Therefore these methods face challenges when applied to large video repositories. We propose a novel technique that leverages sensor-generated meta-data (camera locations and viewing directions) which are automatically acquired as continuous streams together with the video frames. Existing smartphones can easily accommodate such integrated recording tasks. By considering a collective set of videos and leveraging the acquired auxiliary meta-data, our approach is able to detect interesting regions and objects (POIs) and their distances from the camera positions in a fully automated way. Because of its computational efficiency the proposed method scales well and our experiments show very promising results.**

***Index Terms*—Point of Interest, visual distance estimation, sensor-rich video**

I. Introduction

The astounding volume of camera sensors produced for and embedded in cellular phones has led to a rapid advancement in their quality, wide availability and popularity for capturing, uploading and sharing of videos (also referred to user-generated content or UGC). In our work we are interested in mobile videos that have been collected by users during various activities such as vacations, business trips, etc. Specifically, smartphones are carried by millions of people and can now record quite high-quality videos. A recent study by Cisco reported that mobile video already constitutes a large fraction of the overall Internet traffic [1].

With the pervasiveness of these affordable, portable, and networked devices, a large number of geo-tagged photos and videos have been accumulating continuously on the web [2], [3]. However, most applications process this type of information by using only the raw GPS data, such as coordinates and timestamps, without attempting a deeper analysis. They often only allow users to post videos based on a single GPS location, usually the initial camera position. These existing approaches are unsatisfactory under two common conditions: (a) when a user moves during recording the resulting video is taken along a trajectory, hence a single location is insufficient to describe the content, and (b) the location of the most salient object in the video is often not at the position of the camera, but may in fact be quite a distance away. Consider the example of a user recording the pyramids of Giza – he or she would probably need to stand at a considerable distance.

A significant research challenge in recent years has been how to organize large video repositories and make them searchable. This typically requires some kind of understanding of the video content and it has turned out to be a very difficult problem. In our study we propose a method to identify *Points of Interest* (POIs) in videos which contain important objects and places. Such POIs can be part of an attraction, such as the Eiffel Tower, or consist of a more diffuse area that contains no specific physical objects but may be of interest to users, such as the center of an event. Existing approaches [4], [5] often use content-based techniques to extract image features which are then matched to keywords taken from bag-of-words vocabularies. However, due to the overwhelming amount of video material, it is not always realistic to exhaustively process every video segment. Because of the time-consuming procedure of video decoding and the complex feature extraction computations involved, these methods often lack scalability.

Here we present our unique and unconventional solution to address three important challenges in mobile video management: (1) how to find interesting places in user-generated sensor-rich videos, (2) how to leverage the viewing direction together with the GPS location to identify the salient objects in a video, and (3) how to efficiently estimate the visual distance to objects in a video frame. Fig. 1 shows the architecture of the proposed framework. We do not restrict the movement of the camera operator (for example to a road network) and hence assume that mobile videos may be shot along a free-moving trajectory. At first, to obtain a *viewable scene* description, we continuously collect GPS location and viewing direction information (via a compass sensor) together with the video

Manuscript received August 28, 2013; revised December 02, 2013; accepted May 28, 2014. Date of publication 06/12/2014; date of current version 10/13/2014. This research was supported in part by the Singapore National Research Foundation under its International Research Centre @ Singapore Funding Initiative and administered by the IDM Programme Office to the Centre of Social Media Innovations for Communities (COSMIC). The work was also supported in part by the Hongik University new faculty research support fund. This paper was presented in part at 19^{th} ACM International Conference on Multimedia, 2011. The associate editor coordinating the review of this manuscript and approving it for publication was Prof. Kasim Selcuk Candan.

Jia Hao, Guanfeng Wang and Roger Zimmermann are with National University of Singapore, Singapore (e-mail: haojia, wanggf, rogerz@comp.nus.edu.sg).

Beomjoo Seo is with Hongik University in Sejong, Korea (e-mail: bseo@hongik.ac.kr).

Digital Object Identifier: 10.1109/TMM.2014.2330802

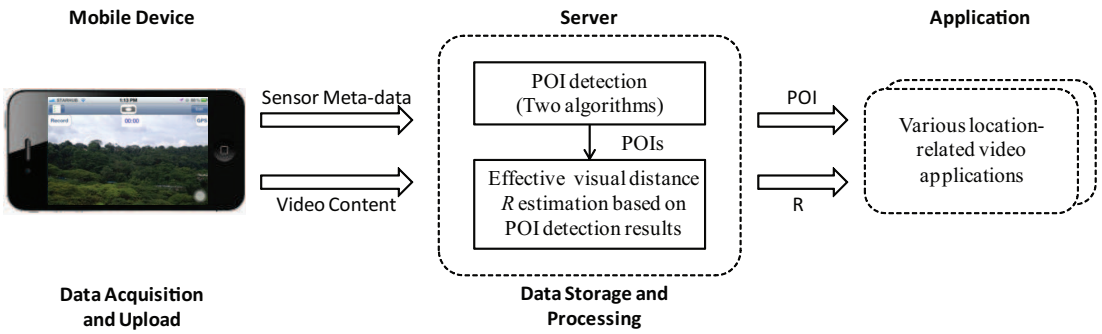


Fig. 1. Architecture of the proposed framework.

frames. Then the collected data are sent via the wireless network to server. This is practically achievable today as smartphones contain all the necessary sensors for recording videos that are annotated with meta-data. On the server side, in the first stage we process the sensor meta-data of a collective set of videos to identify POIs containing important objects or places. The second stage computes a set of visual distances R between the camera locations and the POIs. Finally, the obtained POI and R are ready for other usage.

The overall objective of the proposed work is to enhance the multimedia and mobile web by leveraging the knowledge mined from sensor-rich videos. Manual POI creation, i.e., clicking on a digital map, is possible. However, such manual operations cannot scale to a large spatial dataset. Furthermore, the connection between video content and POI is lost. More importantly, the concept of POI includes not only static objects, but also interesting events, which may dynamically change over time. Relying on manual annotation is not enough to successfully detect all the POIs within a short time period. In contrast, the contributions of our work are as follows:

(1) Two algorithms are proposed to detect POIs (or objects) with high accuracy and fully automatically from large sets of sensor data.

(2) The proposed method identifies salient objects and computes the effective visual distances from the camera location in each video frame.

(3) The approach is scalable to large video repositories as it does not rely on complex video signal analysis, but rather leverages the geographic properties of associated meta-data, which can be done computationally fast and requires no manual intervention.

POI detection can assist in object recognition in videos and landmark detection. It can also be useful in a number of application fields such as providing video summaries for tourists, or as a basis for city planning. Additionally, automatic and detailed video tagging can be done and even simple video search can benefit. This work builds upon our prior work (short paper: [6], demo paper: [7]). The differences and connection between these two works are as follows: (1) The short paper describes a high-level framework for keyframe presentation and the keyframe extraction algorithm from sensor data is its main contribution. (2) In this journal paper we focus on how to accurately detect POIs and their visual distances, and we provide a detailed description and analysis of the two POI detection methods. (3) In the prior work, we made use of the detected POIs and estimated visual distance to identify the keyframes and present the obtained keyframes on a map interface.

The remainder of this paper is organized as follows. Section II describes the related work. Section III details our approach. In Section IV we report on major experimental results and offer some discussions. Finally, in Section V we draw conclusions and present possible future work.

II. Related Work

Our framework draws upon several related areas. Below we provide an overview of some of the existing work.

A. Digital Media with Geo-Locations

Associating GPS coordinates with digital photographs has become an active area of research [8]. There has been work on organizing and browsing personal photos according to location and time. Toyama *et al.* [9] introduced a meta-data powered image search and built a database, also known as the World Wide Media eXchange (WWMX), which indexes photographs using location coordinates and time. A number of additional techniques in this direction have been proposed [10], [11]. There also exist several commercial web sites (e.g., Flickr, Woophy) that allow the upload and navigation of geo-referenced photos. All these techniques use only the camera geo-coordinates as the reference location in describing images. We instead use a multi-sensor approach to describe video scenes.

More related to our work, Ephstein *et al.* [12] proposed to relate images with their view frustum (viewable scene) and used a scene-centric ranking (termed *geo-relevance*) to generate a hierarchical organization of images. The key differences between *geo-relevance* [12] and our grid-based approach lie in three aspects: (1) the geo-relevance method is designed for static images that contain accurate location and orientation information, while our method is working based on a real-world, noisy GPS and compass dataset and is successfully discovering meaningful POIs therefrom. (2) Our method is designed for incremental updates, while it has not been explicitly stated that the geo-relevance approach is suitable for this purpose. (3) The conceptual model of [12] has not been thoroughly validated while we propose two different methodologies (sector-based and center-line-based)

and provide evidence that the center-line-based solution is the most efficient, while achieving comparable accuracy.

Some methods [13], [14] use location and other meta-data, as well as text tags associated with images and the images' visual features, to generate representative candidates within image clusters. Geo-location is often used as a filtering step. Our work considers a much more comprehensive scenario that is concerned with continuous sensor-streams of mobile videos, which are dynamically changing over time.

B. Location Mining History

During the past years, trajectory mining of moving objects and location history mining have attracted significant research efforts. An area of investigation has considered the prediction of the movement of mobile users by mining trajectories [15], [16]. Location-based recommendation systems [17], [18] have also been investigated to provide navigation information. Giannotti *et al.* [19] mined similar sequences from user's moving trajectories, and Hariharan *et al.* [20] presented a framework to parse and model location histories. Li *et al.* [21] mined user similarities based on location histories, and Zheng *et al.* [22] have investigated mining correlations among locations and between interesting locations and travel sequences. The main differentiation between these prior methods and our proposed approach is their isolated use of the location information while we focus on a multi-sensor, multi-video approach that enables the computation of more comprehensive result information.

C. Landmarks Mining from Social Sharing Websites

Some related work focuses on landmark mining using user-generated texts or photos. Ji *et al.* [23] mined city landmarks from blogs by exploiting graphic models. Kennedy *et al.* [24] adopted a tag-location-vision strategy to group city views from geo-tagged Flickr photos of San Francisco. Zheng *et al.* [25] investigated finding highly photographed landmarks automatically from a large collection of geotagged photos. Liu *et al.* [26] presented a filter-refinement framework to discover hot topics corresponding to geographical dense regions. The key difference from prior work is that we identify the location of places (or buildings, etc.) that are of interest to people, not the camera location (which could be far away). Thus the video can be associated with the object(s) that are the focus of attention in a video.

III. Approach Design

In this section we present the details of our approach. As key features we use statistical knowledge to infer POI locations from collected sensor measurements without any a priori information and then estimate the visual distance R between camera positions and those POIs.

Conceptually, the estimation of the effective visual distance R is the process of determining the distance to object or scene locations where one or many camera views are pointing. In the rest of this paper we term such a location (or its grid cell representation on a 2D map) as POI. Such POIs can be part of an attraction or landmark or consist of a more diffuse area that contains no specific physical objects but may be of interest to users (*e.g.*, a beautiful lake or valley).

Our approach leverages a two-stage process which is outlined in Fig. 1. First, POIs are detected from profiling of data collected from a large set of sensor recordings and then second, the visual distances R are estimated between camera and POI locations. Our framework utilizes the meta-data which is concurrently collected with video frames to describe the geographic properties related to the camera view. We start in Section III-A by describing the viewable scene model used and present how we collected videos and their sensor measurements. Next, Section III-B introduces two methods for POI detection. Cluster-based method computes the intersection points of all the camera views and then identifies the clusters inferred from this point cloud as POIs. This is an intuitive design based on the concept of POI. Grid-based method generates a popularity map based on how often an area appears in different camera views and then identifies the most popular places. This method quantizes the target space into a finite number of cells, and fast processing time which depends on number of cells in each dimension in quantized space is its advantage. Finally we estimate the effective visual distance R by calculating the distance between camera locations and the closest POIs in Section III-C.

A summary of the terms and a definition of the symbols appearing in this paper are shown in Table I.

TABLE I
Terms and definition of symbols.

Notation	Definition
α	Camera orientation
$\mathbf{d}_i$	Multi-dimensional sensor data observation (FOV)
$\mathbf{D}$	Set of sensor measurement $\mathbf{d}_i$ (FOVs)
FOV	Field-of-View
$\mathcal{G}$	Target geographic map area
g	Square grid cell
h	POI grid cell
H	Set of h
$\mathbf{l}_i$	Center line of FOV $\mathbf{d}_i$
θ	Viewable angle
P	Camera location consists of the latitude and longitude
R^{max}	Maximum visible distance
R	Effective visual distance
$\mathbf{x}_i$	Measurement vector consists of the subtended angle a and distance d to a POI
X	Set of $\mathbf{x}_i$

A. Viewable Scene Model

A camera positioned at a given point P in geo-space captures a scene whose covered area is referred to as the camera *field-of-view* (FOV, also called the *viewable scene*). We adapt the FOV model introduced in our prior work [27], which describes a camera's viewable scene in 2D space by using four parameters: the camera location P, the camera orientation α, the viewable angle θ and the maximum visual distance R^{max} (see Fig. 2).

The camera position P consists of the latitude and longitude coordinates read from a positioning device (e.g., GPS sensor) and the camera direction α is obtained based on the orientation angle provided by a digital compass. The orientation obtained here refers to the shift from the geographic north, not the

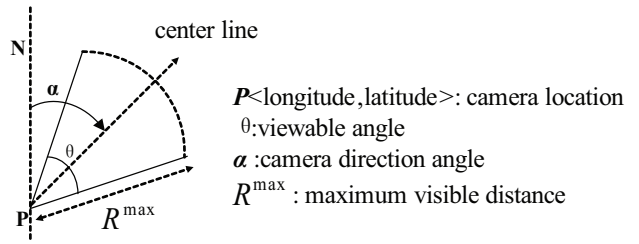


Fig. 2. Illustration of the camera field-of-view (FOV) in 2D space.

magnetic north. R^{max} is the maximum visual distance from P at which a large object within the camera's field-of-view can be recognized [27]. The angle θ is calculated based on the camera and lens properties for the current zoom level [28]. The collected meta-data streams consist of sequences of $\mathbf{d}_i = \langle nid, vid, t_{FOV}, t_f, P, \alpha, \theta \rangle$ tuples, where nid represents the ID of the mobile device, vid is the ID of the video file and t_{FOV} indicates the time instant at which the FOV tuple was recorded. The timecode associated with each video frame is denoted by t_f. In 2D space, the field-of-view of the camera at time t_{FOV} forms a pie-slice-shaped area as illustrated in Fig. 2.

To acquire sensor-annotated videos we have developed two custom recording apps for Android- and iOS-based smartphones and tablets. When a mobile device begins to capture video, the GPS and compass sensors are concurrently enabled to record the location and orientation of the camera. Our data-acquisition software fetches such sensor values whenever new values are available. Video data are processed in real-time to extract frame timecodes (t_f). All collected meta-data (*i.e.*, location, direction, frame timecode and video ID) are combined as a tuple and stored for uploading to a server.

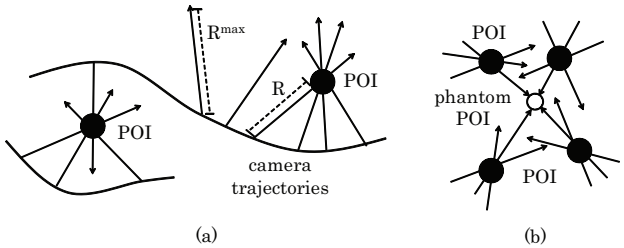


Fig. 3. (a) Conceptual illustration of visual distance estimation. (b) Illustration of the detection of a non-existent "phantom" POI.

Fig. 3(a) illustrates the concept of visual distance R estimation in conjunction with the identification of POIs. Along the camera trajectory (curves), the camera views (arrows) tend to point to some areas (solid circles) more frequently, and R can be determined as the distance between such popular areas, *i.e.*, POIs, and the camera locations.

To explain our POI detection and visual distance estimation better, we now provide some descriptions of two terms which will be used later.

Description *(POI)*: POI is a point of interest in a video which contains an important object, place or event. When mapping to geographic space, the term POI refers to a poplular area to which camera views tend to point more frequently. We define a POI as an area encompassing a set of points (square grid cells).

Description *(phantom POI)*: A phantom POI is not a real point of interest. It is generated due to the interference of nearby POIs (as shown in Fig. 3(b)), i.e., multiple cross-intersections from at least two lines of two independent POIs. A phantom POI should not be included in the set of detected POIs.

B. POI Detection

We introduce two approaches for detecting POIs. The *cluster-based POI detection* method (Section III-B1) applies a clustering algorithm on the cloud of intersection points of the FOV center lines. The second, *grid-based POI detection* method (Section III-B2) is utilizing a grid counting approach to obtain a popularity distribution of FOVs. The target space is assumed to be a 2D geographical map (*i.e.*, $\mathcal{G} = \mathbb{R}^2$). Let the sensor measurement results be expressed as sets of tuples of $\mathbf{D} = \mathbf{d}_1, \cdots, \mathbf{d}_n$, where each $\mathbf{d}_i$ is a multi-dimensional observation consisting of GPS coordinates, compass direction angles, etc., as described in the previous section.

1) Cluster-based POI Detection: As is illustrated in Fig. 3(a), an intuitive way to detect POIs is to use the intersection points of the center vector of FOVs. All the intersection points together form a point cloud on which a clustering algorithm can be applied to obtain clusters as the detected POIs. However, as we found from our experiments, this method suffers from a major performance issue due to the generally large number of intersection points. Hence, we propose a pre-processing step to reduce the number of input points for the clustering algorithm.

We separate the center line $\mathbf{l}_i$ of FOV (illustrated in Fig. 2) $\mathbf{d}_i$ into m segments $(\mathbf{l}_i^1, \cdots, \mathbf{l}_i^m)$ of equal length R^{max}/m. Next we calculate the intersection points of segment $\mathbf{l}_i$ with all the other center lines. The algorithm maintains a data structure for each segment containing a monotonically increasing counter of intersection points. Subseqently, for each center line we represent it as a curve to describe the distribution of the intersection points. To increase the signal-to-noise ratio (SNR), we smooth the curve with a moving-average filter. Among all the segments of $\mathbf{l}_i$ we compute the local maxima of the curve and then identify their positions as the points of interest of $\mathbf{l}_i$. The key rationale behind this operation is that the intersection points tend to crowd around the interesting points where the FOVs really focus on.

After collecting the interesting points for each center line we apply a density-based clustering algorithm, DBSCAN [29], to detect POI regions. A number of clustering algorithms are available and the reasons we selected DBSCAN are as follows. (1) DBSCAN does not require the number of clusters to be selected in advance. This is important as we do not know how many POIs may exist. (2) DBSCAN can find arbitrarily shaped clusters so that we can identify irregularly shaped POIs. (3) DBSCAN has a notion of noise suppression so that unpopular areas can be pruned. Finally, (4) DBSCAN is insensitive to the ordering of the input points.

2) Grid-based POI Detection: The target space is first partitioned into equally-spaced square grid cells g. Assuming

$H \subset \mathcal{G}$ is the set of POI grid cells given $\mathbf{D}$, then the probability that a grid cell g belongs to a POI given the sensor observations is expressed as $p(g \in H|\mathbf{D})$, or simply $p(g|\mathbf{D})$.

To obtain the posterior probability $p(g|\mathbf{D})$ we use a grid counting-based popularity method. Unlike existing popularity estimation methodologies for geo-tagging [30], [31], where GPS locations are typically used as the popularity measure, we leverage the visual coverage model conveyed by the field-of-view of a camera, because the camera position may be displaced from the area or location in which users are actually interested in when recording video. The key rationale behind our approach is that if an area is pointed to by cameras more often, it will more likely be a POI. This reasoning is analogous to the *PageRank* [32] algorithm where a web page is considered more important if it is linked to by many other web pages. As illustrated in Fig. 3(a), POIs tend to be pointed to, or experience overlap from, many camera FOVs. One situation needs to be specially handled, namely when important objects are located across from each other and hence camera direction vectors coincidentally intersect in an area between the actual POIs – see the example in Fig. 3(b). In the rest of this paper we term such areas "phantom POIs" and they will be considered later.

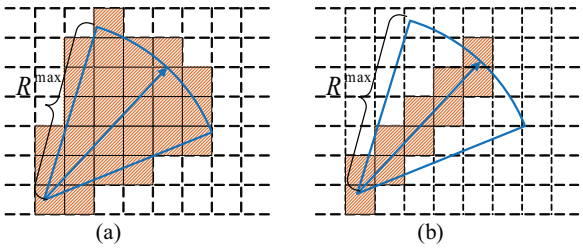


Fig. 4. (a) Sector-based coverage model. (b) Center-line-based coverage model.

Our algorithm maintains a 2-dimensional data structure representing every map grid cell and containing a monotonically increasing counter of interest. Without prior knowledge of the underlying landmarks or attractions, the counter is incremented whenever its grid cell is visually covered by an FOV. We investigate two visual coverage models: a *sector-based* coverage model and a *line-based* model. The sector-based coverage model uses an FOV that is abstracted as a sector whose maximal visual distance is R^{max} (say, 1 km). As illustrated in Fig. 4(a), we increase the counter of all the grid cells that overlap partially or fully with the sector shape. Since this exhaustive coverage is time-consuming to process, we introduce a lightweight solution, namely a *center-line-based* coverage model. It uses a line vector with length R^{max}, whose origin coincides with the GPS location and whose heading is the camera direction – see Fig. 4(b). With this model we increase only the counters of the grid cells that intersect with the center vector. The intuition for this approach is that the main focus of interest in videos is often on objects located near the center of the frame or the FOV as shown in Fig. 5.

Using either of these two coverage models we generate a posterior popularity probability for every grid cell, $p(g|\mathbf{D})$, by

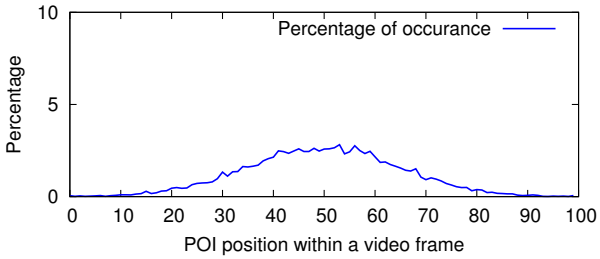


Fig. 5. Distribution of horizontal POI position within a video frame for two videos V_{8636} and V_{1477} in Fig. 13 (0 – left margin, 50 – center, 100 – right margin).

normalizing the counters as follows

$$p(g|\mathbf{D}) = \frac{c_g}{\sum_{\mathbf{d}_j \in \mathbf{D}} s_j}, \tag{1}$$

where c_g is the counter of grid cell g and s_j is the number of grid cells affected by each coverage model generated by the sensor measurement $\mathbf{d}_j$. Without loss of generality we use the posterior probability as the counter value interchangeably.

Among all the grid cells, we then compute the local maxima across the map and identify them as POIs if their probability is higher than those of all their neighboring cells and the difference exceeds a certain threshold (K)

$$p(g = h|\mathbf{D}) \geq K + p(i|\mathbf{D}), i \in N_g, \tag{2}$$

where N_g is the set of g's neighboring cells.

One important aspect of large-scale systems is the efficiency of the employed algorithms. Next we will perform a complexity analysis of the two POI detection algorithms.

3) Computational Complexity Analysis: We consider that there are N_f FOVs (sensor measurement results) in the target 2D space.

For the cluster-based method, computing the intersection points of N_f center lines of FOVs results in time complexity $O(N_f^2)$. Next we compute the local maxima along the m segments of each center line, and the complexity for this operation is $O(N_f \times m)$. Assuming that for each center line we get k ($k < m$) interesting points on average as input to the DBSCAN algorithm, then the total number of input points is $O(k \times N_f)$ and the run time complexity for clustering is $O((k \times N_f)^2)$. Therefore, the overall time complexity for the clustering-based method is

$$O(N_f^2) + O(N_f \times m) + O((k \times N_f)^2) = O(N_f^2)$$

For the grid-based method, the time complexity of computing the counters for N_c grid cells covered by N_f FOVs is $O(N_f)$. The complexity of computing the local maxima of the map is $O(N_c)$. Hence, the overall time complexity for the gird-based method is

$$O(N_f) + O(N_c)$$

From the above analysis we observe that when N_f^2 is much larger than N_c, the grid-based approach is much more efficient than the clustering-based method. On the other hand, when N_f^2 is much smaller than N_c, the cluster-based method is more efficient. We conclude that the grid-based approach is more

suitably applicable to dense areas with a lot of FOV data (which is the more frequently case), while the cluster-based method is best applied to sparse data areas.

C. Effective Visual Distance Estimation

Let H be a set of grid cells from the estimated POIs computed from the previous stage. The sensor values related to a camera view are transformed into a measurement vector $\mathbf{X} = \{\mathbf{x}_1, \ldots, \mathbf{x}_{|H|}\}$, where $\mathbf{x}_i$ consists of the subtended angle a of a compass direction to the center of POI grid cell $i \in H$ and the Euclidean distance d to the center of i.

Our effective visual distance estimation is based on the assumption and observation that a camera view tends to point to an interesting place more often than to an unattractive area. Therefore, we can leverage this information to estimate the effective visual distance by the closeness – in terms of lower angular disparity and higher popularity rank – to a POI and select the closest one. Eqn. 3 expresses the effective visual distance as the distance to the POI which is most likely pointed to by a camera view, by choosing the minimum subtended angle along with a higher posterior popularity probability, *i.e.*, 1 - $p(i|\mathbf{D})$.

$$\{\underset{\mathbf{x}_i}{\operatorname{argmin}} \ (\mathbf{x}_i.a) \cdot (1 - p(i|\mathbf{D}))\}.d \tag{3}$$

This computation, however, may result in an incorrect decision because of the existence of *phantom* POIs. As seen in Fig. 3(b), camera views crowd around real POIs, but occasionally four or more camera views point to the same area by-passing the real POIs. Such an area is detected as a POI by our algorithm. However, a phantom POI should not be chosen as the closest POI, since it may be erroneous. We may exclude such non-existing POIs with information from third-party landmark databases such as Google Maps or OpenStreetMap. However, such landmark information is generally only available for some highly popular areas.

To eliminate such undesirable phantom POIs when no landmark databases are available, we propose a cross-intersection elimination heuristic. This exhaustive technique is based on the observation that a phantom POI is generated by multiple cross-intersections from at least two lines of two independent POIs. This implies that some candidate POIs near the intersection points are highly likely to be phantom POIs. Algorithm 1 outlines our method. It first creates all possible intersection points of two disjoint lines from a set of candidate POIs (lines 2–9). If the intersection points and their corresponding POIs are too close, we discard them (lines 4–5). Next, we compute how far each POI is located from the intersection points and select some within a given threshold distance (lines 11–15). Finally, we recover the POIs that contribute to the elimination of other POIs and return the remaining POIs without any cross-intersections (lines 16–21). The setting of threshold Th relates to the size of grid cells. In our experiment, we set Th as the length of diagonal of a grid cell.

Algorithm 1 Cross-Intersection Elimination Algorithm

Require: H: set of candidate POIs, Th: a threshold
1: $P \leftarrow \emptyset$
2: **for all** different $i, j, k, l \in H$ **do**
3:

$$\begin{aligned} p\{i,j,k,l\} &= \texttt{intersection}(\overline{\{i,j\}}, \overline{\{k,l\}}) \\ &\cup \ \texttt{intersection}(\overline{\{i,k\}}, \overline{\{j,l\}}) \\ &\cup \ \texttt{intersection}(\overline{\{i,l\}}, \overline{\{j,k\}}) \end{aligned}$$

4: **if** $\exists\ p \in p\{i,j,k,l\}, q \in p\{i,j,k,l\}$ and $||p,q|| \leq Th$ **then**
5: **continue**
6: **else**
7: $P \cup = p\{i,j,k,l\}$
8: **end if**
9: **end for**
10: $C \leftarrow \emptyset$
11: **for all** $i \in H, p \in P$ **do**
12: **if** $||i,p|| \leq Th$ **then**
13: $C = C + i$, **break**
14: **end if**
15: **end for**
16: **for all** $c \in C$ **do**
17: **if** $\exists\ p\{\cdots, c, \cdots\} \in P$ **then**
18: $C = C - c$
19: **end if**
20: **end for**
21: **return** $H - C$

IV. Experiments

We first describe how we collected the test data for our experiments. Second, we report on the experimental results and provide some discussions.

A. Data Collection

1) Recording Hardware and Software: Fig. 6 shows the screenshots of the acquisition software we used to collect our dataset. The software apps provide automated annotation of captured videos with their respective field-of-views (FOV). The resolution of the collected videos is 640×480 pixels (Android) or 720×480 (iOS). The frame rate is 24 frames per second, and the sampling rate for the location and orientation sensor information is 1 sample per second.

2) Video and Sensor Dataset: To evaluate the performance of our framework for POI detection and effective visual distance estimation, we prepared two video and sensor datasets.

(1) Singapore dataset

The videos were collected in the Marina Bay area in Singapore. We recorded 71 video sequences of sensor-annotated, mobile videos over a three months period (Dec. 2010 – March 2011) in a 2km-by-2km area, with most of the videos taken in open space. The trajectories of these test videos are shown in Fig. 9(c) (bright green lines). The total length of the test videos is 9,718 s. Different users were capturing the videos and they were not told to deliberately capture landmarks. Therefore, the videos were taken in various locations within the area and not all landmarks appeared in every video. All FOVs of the test videos are contained within a rectangular area with the top left corner at (lat/long 1.29328497, 103.8481559) and bottom right corner at (lat/long 1.27728497, 103.8661559). We partitioned the area into 100×100 grid cells. The size of each cell was about 20×18 m^2.

(2) Chicago dataset

This dataset was collected during the period of NATO

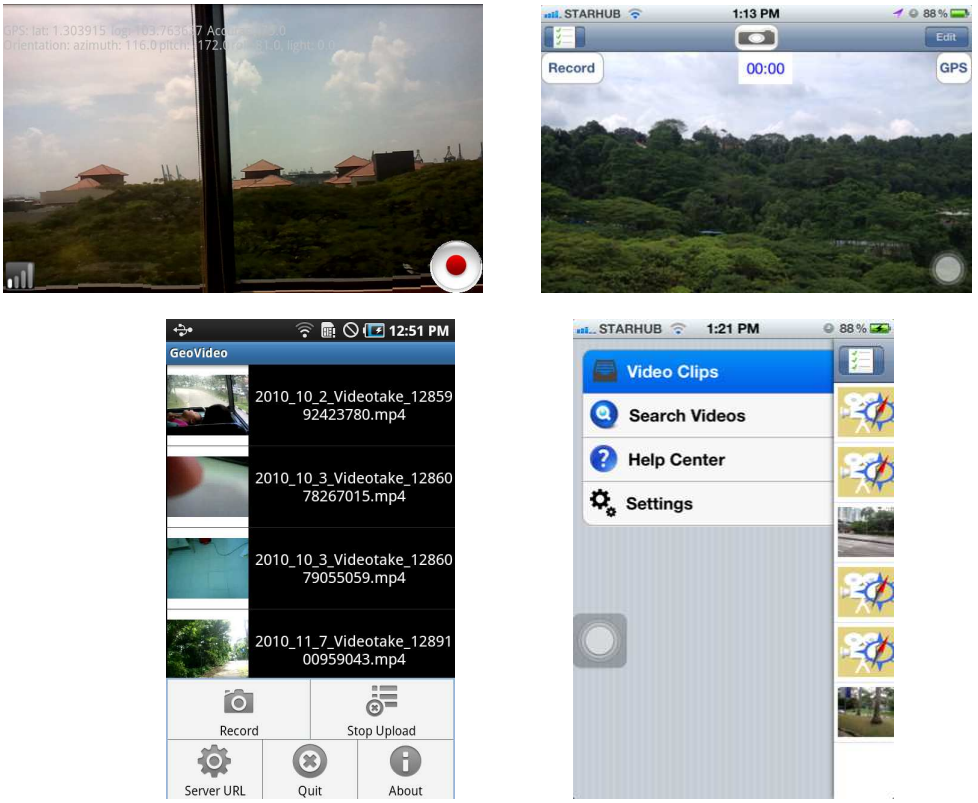


Fig. 6. Screenshots of acquisition software for Android-based (left) and iOS-based (right) smartphones used in the experiments.

Summit (18 – 19 May 2012) held in Chicago. 284 videos were recorded by college students in the downtown area. We choose 228 videos in a 3km-by-3km area as our test dataset. The trajectories of these test videos are shown in Fig. 10 (bright green lines). The total length of the test videos is 9,634 s. All FOVs of the test videos are contained within a rectangular area with the top left corner at (lat/long 41.893993, -87.649107) and bottom right corner at (lat/long 41.865440, -87.611713).

More details about the two datasets are listed in Table II.

TABLE II
STATISTICS OF THE TWO DATASETS.

Dataset	# of photographer	Recording period	Total length of videos	Total length of trajectories
Singapore	5	Dec. 10 – Mar. 11	9,718 s	8,022 m
Chicago	40	18 – 19 May 12	9,634 s	11,520 m

3) Sensor Data Error Filtering: State-of-the-art mobile devices report the GPS accuracy of the received GPS signal in addition to the location. We found that the two different mobile devices we tested had two different minimum GPS error bounds: 6 meters and 2 meters. The GPS error distribution for Singapore dataset is shown in Fig. 7. One can see that most of the GPS errors are below 20 meters.

Based on this observation we filtered out GPS values with GPS error values higher than 20 meters. Hence, because some sample points were discarded, there existed a few gaps between some consecutive GPS measurements and we linearly interpolated those values. The above method is introduced by Hakeem *et al.* [33].

Even after filtering, we still cannot guarantee that all the errors are corrected. Since our method is based on the FOVs constructed by continuous sensor data sampling, sensor data errors can counterbalance each other to a certain degree. For example, for a specific point, the GPS data sampled at one time may be located on one side of the accurate location while for another sampling it may be located on the opposite side. By accumulating these measurements, the randomness of the errors provides compensation and should not significantly affect the accuracy of POI detection and visual distance estimation.

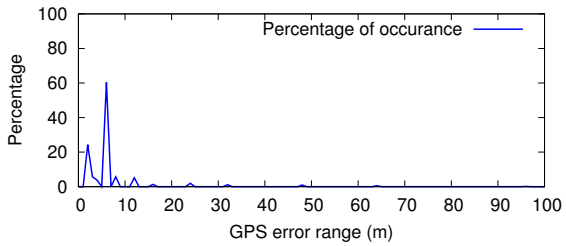


Fig. 7. GPS error distribution for Singapore dataset.

Orientation data collected from the digital compass is not accurate enough either. To improve the accuracy of noisy orientation sensor measurements generated by mobile devices, we use the OSCOR system [34], [35]. The system collects visual landmark information and matches it against GIS data sources to infer a target landmark's real geo-location. By knowing the geographic coordinates of the captured landmark and the camera, we are able to calculate corrected orientation data.

B. Results

Two POI detection algorithms are written in C and Matlab. We used C program to get the intersection points info (Cluster-based) and the grid counting results (Grid-based), after which we ran DBSCAN algorithm (Cluster-based) and computed the local maxima (Grid-based) in Matlab.

1) POI Detection Results:

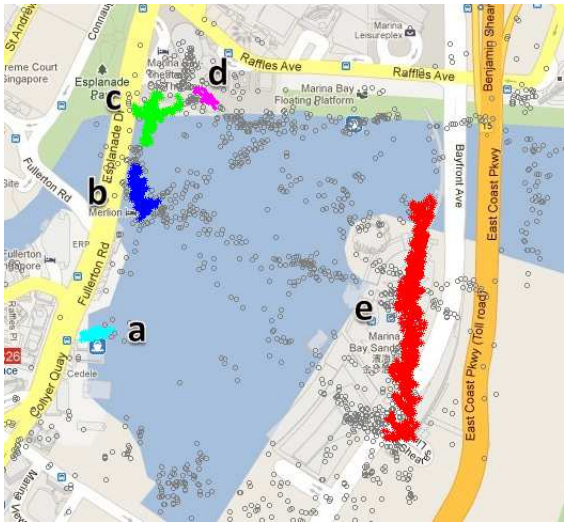


Fig. 8. POI detection results of the cluster-based method (Singapore).

a) POI detection from Singapore Dataset: Fig. 8 shows POI detection results from the **cluster-based method**. The labeled POIs a, b, c, d, and e all represent the locations of interesting areas (a: Clifford Pier; b: Merlion; c, d: Esplanade; e: Marina Bay Sands). The gray circles correspond to noise which was separated from the POIs by DBSCAN. In this method, we use the Euclidean distance as the distance metric. From the figure, we can see that the cluster-based method can successfully identify a set of POIs. However, DBSCAN does not cluster data sets well with large differences in densities. Therefore, some areas that are more dense than the surrounding regions but still have an overall quite low density will not be selected even though they should be identified as POIs.

Results from our **grid-based POI detection method** on Singapore dataset are presented in Fig. 9. Fig. 9(a) and 9(b) show the contour plots for two popularity maps obtained from the sector-based and the center-line-based coverage models, respectively. As more points are sampled with the sector-based method, the corresponding FOV popularity density map (Fig. 9(a)) is more smooth in appearance. On the other hand, the FOV density popularity map we obtained from the center-line-based coverage model (Fig. 9(b)) appears more jagged and a greater number of local maxima can be observed.

Fig. 9(c) and 9(d) show two snapshots from Google Maps with superimposed POIs detected from the popularity maps of Fig. 9(a) and 9(b). The bright green lines are the trajectories of the 71 videos and the red points represent the POIs that were detected. In Fig. 9(c) we can observe that the POIs around the Merlion and Marina Bay Sands complex were successfully detected. However, compared with Fig. 9(d), several POIs are missing in Fig. 9(c). In Fig. 9(d), the POIs labeled A, B, C, D, E, H, G all represent the locations of landmarks or locations near landmarks (A: Clifford Pier; B: The Fullerton Hotel Singapore; C: Merlion; D, E: Esplanade; G: The Float; H: Marina Bay Sands). However, point F is located in the water in between those landmarks. F is an example of a phantom POI that was detected because of the intersection of center lines targeting surrounding landmarks. With our cross-intersection elimination heuristic (Algorithm 1), F can be removed.

We observe from the above figures that the sector-based coverage model obtains a subset of all the POIs detected by the center-line-based model. The reason is that the most interesting objects in a video frame often appear in the middle of the frame. Hence the sector-based model actually has a more diffuse focus, resulting in POI regions that are less clearly distinguishable. As an additional benefit, the center-line-based model is more computationally lightweight, therefore we select it for our visual distance estimation process. We can observe that the POI distribution in the center-line-based model is very similar to the results achieved by the cluster-based method (Fig. 8).

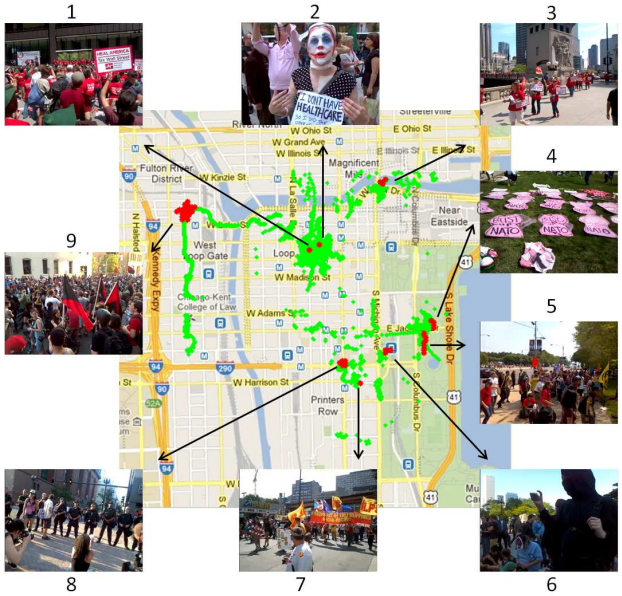
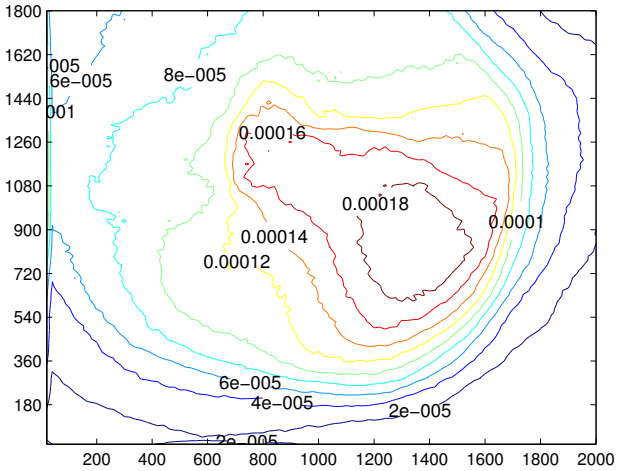
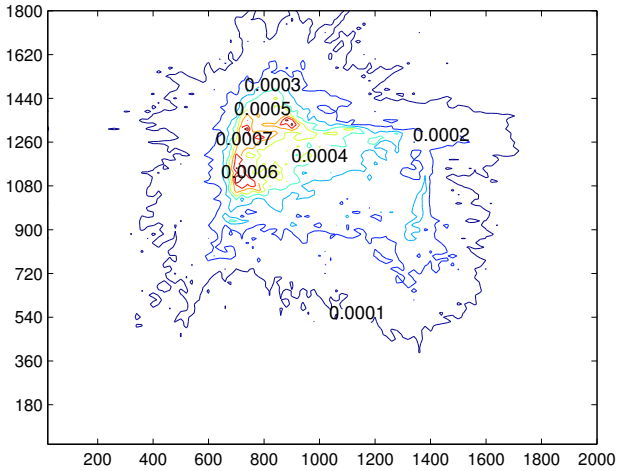


Fig. 10. POI detection results of the grid-based method (Chicago).

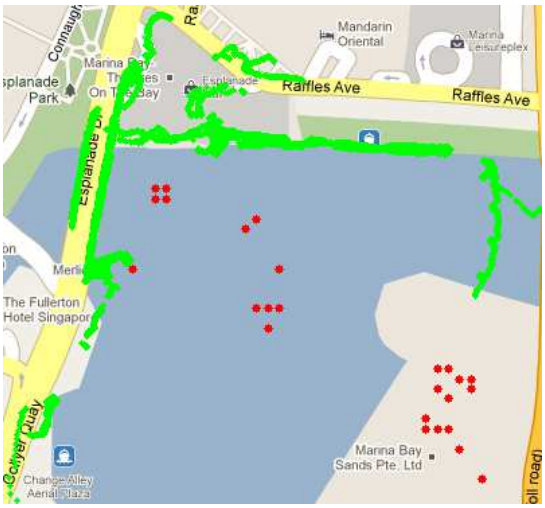
b) POI detection from Chicago Dataset: Fig. 10 shows POI detection results obtained from the center-line-based coverage model for **grid-based method** based on Chicago dataset. The bright green lines are the trajectories of the 228 videos and the red points represent the POIs that were detected. The pictures pointed by individual black arrows are the keyframes which represent the important events occurred in the POI areas. As the dataset was collected during the NATO Summit, most of the POIs detected are crowded by agitated protestors who were waving placard and shouting slogan. For instance, picture 1, 3, 5, 7, 9 show marching demonstrators, while picture 2, 6 are interviews with the citizens. Obviously, the property of POIs obtained from two datasets are quite different. POIs from Singapore dataset represent famous landmark and interesting physical object, while POIs from Chicago



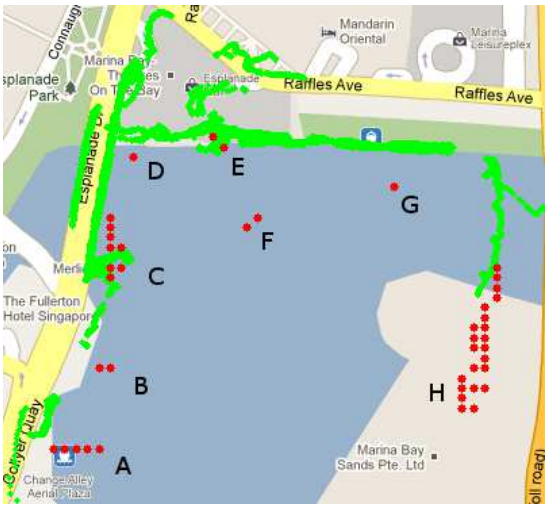
(a) Sector-based FOV density popularity map.



(b) Center-line-based FOV density popularity map.



(c) POIs estimated from sector-based FOV density popularity map.



(d) POIs estimated from center-line-based FOV density popularity map.

Fig. 9. FOV density popularity and POI detection results for two coverage models with the grid-based method (Singapore).

dataset represent events which have a specific timeline. Both sets of POIs are mined from a certain number of sensor-rich videos within a geographical area, and they indicate the popular location or the interesting object around that area.

Results from our **cluster-based POI detection method** on Chicago dataset are presented in Fig. 11. The POIs detected are identified by different color. The gray circles correspond to noise which was separated from the POIs by DBSCAN. Compared with the results from grid-based method (see Fig. 10), there is only some small offset on the location of the POIs, while the size of some POIs do change a lot. Currently we have not investigated the problem of how to get a precise size of POI, and for these videos with events as their content, we can not simply decide which method is better than the other by the shape of POI. Therefore further efforts are still needed to provide a reasonable benchmark to measure the accuracy of POI detected.

c) Comparison between two POI detection methods: We executed the algorithms on a PC with 1 dual-core 3GHz CPU and 4GB of main memory. Table III provides the statistics about the POI detection results. For the Singapore dataset, it costs 28.83 s to process the metadata from the 71 videos and

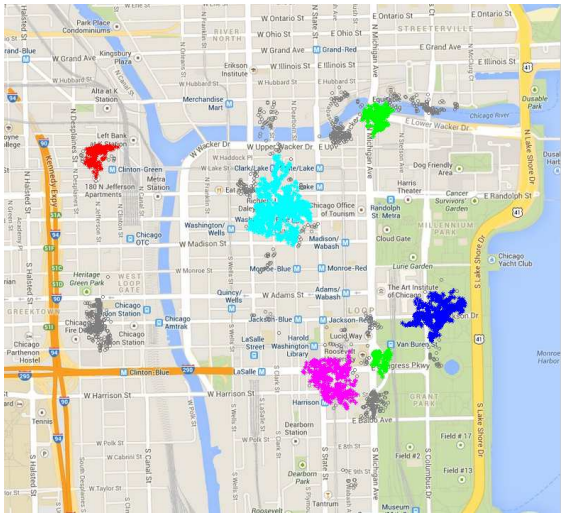


Fig. 11. POI detection results of the cluster-based method (Chicago).

TABLE III
COMPARISON BETWEEN TWO POI DETECTION METHODS.

	Singapore dataset		Chicago dataset	
	Run time	# of POIs	Run time	# of POIs
Cluster-based method	28.83 s	5	41.48 s	8
Grid-based method	4.33 s	8	8.80 s	9
Average R difference	29.85 m		33.33 m	

get 5 POIs for the cluster-based method while it costs 4.33 s to obtain 8 POIs with the grid-based method. For the Chicago dataset, it costs 8.8 s to process the metadata from the 228 videos and get 9 POIs for the grid-based method, and it costs 41.48 s to obtain 8 POIs with the cluster-based method. For both datasets, the grid-based method are much more efficient than the cluster-based method. This is due to the large amount of FOVs. To find out when the cluster-based method can run faster, we randomly discarded FOVs from Chicago dataset and ran the algorithms with the left FOVs again. As shown in Fig. 12, after the number of FOVs is reduced from 9,634 to 98, the cluster-based method can be more efficient than the grid-based method.

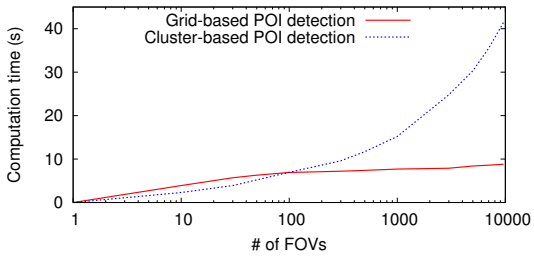


Fig. 12. Computation time of two methods with varying number of FOVs.

To compare the obtained POIs from the two methods, we compute "Average Distance Difference" between Cluster-based and Grid-based method. First, for each method, we calculated the center of each POI by using Method C introduced from [36]. Second, we found the corresponding POI pairs from the two sets of POIs and computed the distance difference between the center of the POIs. Obtained all the distance differences, the average number was calculated to reflect the difference between the two methods. From the last row of Table III, we can see that for the two datasets, the average distance differences are both within 35 m, which means that the results from two methods are quite similar.

However, for the cluster-based method, due to the limitation of the clustering algorithm, some of the POIs detected by the grid-based method are not selected as POIs (i.e., A, G in Fig. 9(d)). Despite this, the cluster-based method still has its own strength – detecting the shape of a POI more accurately (i.e., e in Fig. 8 compared to H in Fig. 9(d), e represents a better shape for the Marina Bay Sands building).

2) Effective Visual Distance Evaluation: To evaluate our estimation of the effective visual distance R we manually collected ground truth data. For each video frame represented by a corresponding FOV, we found the most important object in the frame based on human perception, located the object on Google Earth, and then obtained its latitude and longitude. After that, the distance between the camera location and the object was calculated as the ground truth of the effective visual distance. The frames in transition or containing ambiguous content (*i.e.*, when even users could not identify the most important object) were discarded.

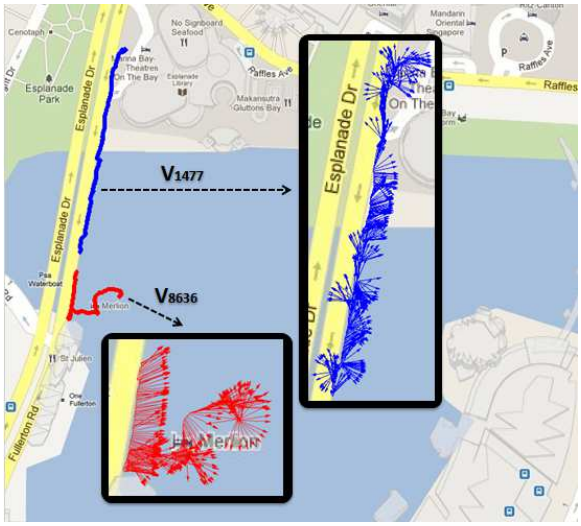


Fig. 13. Center line vector sequences for videos V_{8636} and V_{1477}.

For the R estimation we present the detailed results for two videos from Singapore dataset, V_{8636} (length 00:05:07) and V_{1477} (00:08:44). Both were captured along a trajectory near the Merlion statue. We found the content of videos V_{8636} and V_{1477} to be representative as they include most of the landmarks in the Marina Bay region. The center line vector sequences for the two videos are shown in Fig. 13. The trajectory of V_{8636} is presented in red, while V_{1477} is presented in blue. The sensor data sampling rate here is 1 per 5 seconds. Note that the actual visual distance is much larger than the length of the trajectory, hence we set the length of the vectors to 20 m.

Fig. 14 shows the comparison between the ground truth and the estimated effective visual distance for video V_{8636}. We can see that the estimated distances match excellently with the ground truth data. For instance, the estimated distance R is 712 m for frame 40. Correspondingly, we observe from the actual image that the focus of the frame is Marina Bay Sands, which is very far away. The actual distance from the camera location to Marina Bay Sands is 720 m. For frame 120, the focus is on the Merlion, which is nearby (the actual distance is 9 m). Accordingly, the estimated distance R is 20 m. However, as seen in other frames, there sometimes exists quite a substantial difference between the ground truth and the estimated R. This may be due to the fact that we selected only one object as the ground truth from each frame and hence sometimes our estimated R cannot completely match the ground truth well when there are multiple POI candidates within one image. With frame 65, for example, the ground truth is the distance to the Merlion while the estimated R is very different because it is based on a different POI (Fig. 9(d) G). The average distance error (|Estimated R - Ground truth R|) in Fig. 14 is 77.7 m.

Fig. 15 shows similar results as Fig. 14, with the average distance error Fig. 15 being 69.6 m. However, some interesting details are revealed. In Fig. 15, frame 1 captures the front

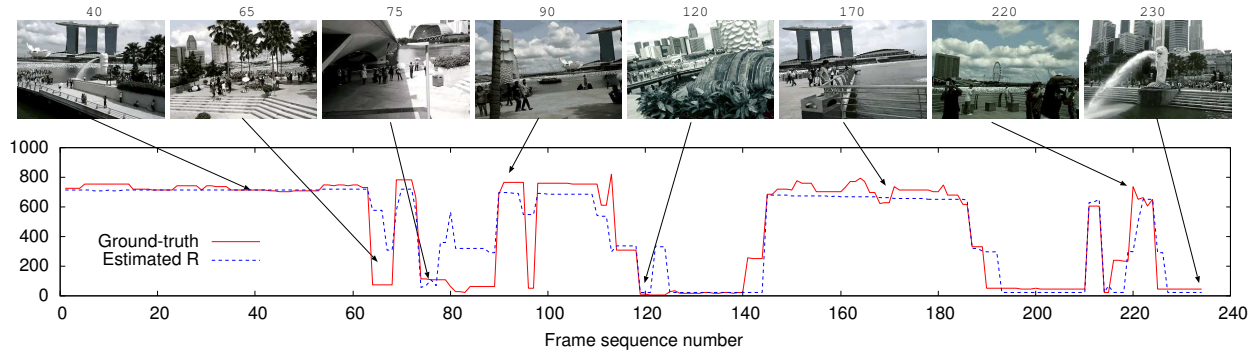


Fig. 14. Comparison between the ground truth and the estimated visual distance R for video V_{8636} (the frame sequence number is labeled on top of the selected frames).

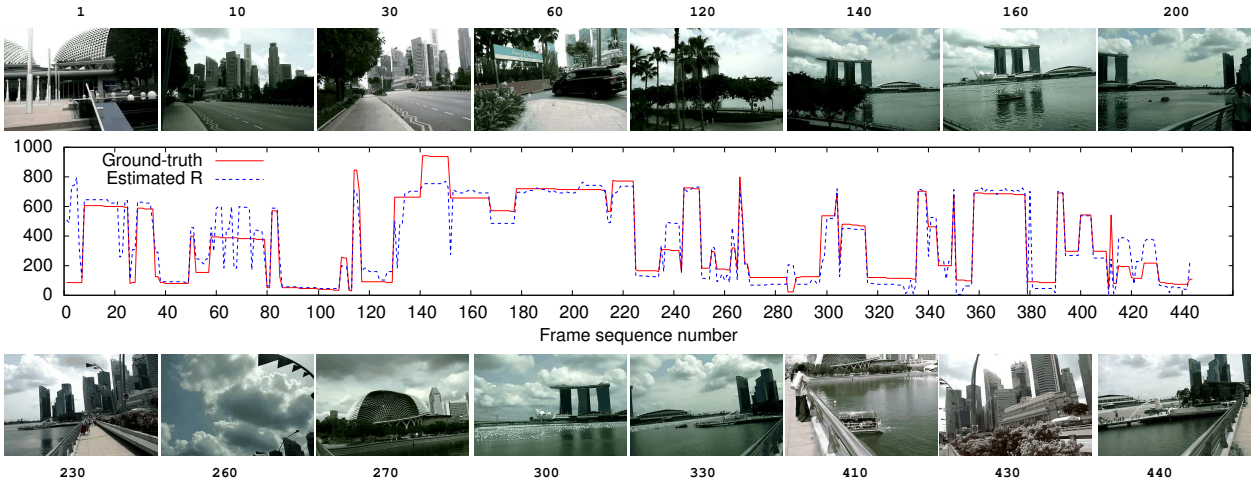


Fig. 15. Comparison between the ground truth and the estimated visual distance R for video V_{1477} (the frame sequence number is labeled on the selected frames).

side of the Esplanade theater, where the estimated distance R does not match with the ground truth. This is due to the failure of our method to accurately describe the shape of the Esplanade POI (Fig. 9(d) D and E), which reinforces the notion that more precise POI detection results lead to better visual estimations. In frame 260, the user was pointing the camera up and recording a flag waving in the sky. This operation caused an erroneous R estimation because our method is currently limited to 2D space and therefore unaware of the altitude and elevation angle of the camera. The estimation process always tries to find POIs on the ground plane. A future solution to this problem is to construct a field-of-view model in 3D space so that the height of a POI can be considered.

To understand the accuracy of the estimated distance R, we collected ground truth data from 1,000 video frames, which were randomly selected from our video dataset. In Table IV, the first column shows the error range calculated from the difference between the estimated visual distance R and the ground truth data. The second column shows the percentage of the corresponding error range. We note that more than 70% of the errors in the estimated distance are below 100 m and errors above 300 m occur very rarely. Columns three and four list the relative error values. We used the following error metric e to calculate the relative error:

$$e = \left| \frac{\text{Estimated } R \text{ - Ground Truth } R}{\text{Ground Truth } R} \right| \tag{4}$$

The table demonstrates that R values with a relative error of 30% or less represent the majority. The above two error distribution trends clearly indicate the usability of our visual distance estimation method.

TABLE IV
ABSOLUTE AND RELATIVE ERROR DISTRIBUTION OF THE ESTIMATED VISUAL DISTANCE R.

Absolute Error	Percentage of Occurance	Relative Error	Percentage of Occurance
0 m $\sim$ 100 m	73.0%	0 $\sim$ 10%	38.1%
100 m $\sim$ 200 m	12.8%	10 $\sim$ 20%	18.9%
200 m $\sim$ 300 m	11.9%	20 $\sim$ 30%	7.9%
300 m $\sim$ 400 m	0.8%	30 $\sim$ 40%	10.1%
400 m $\sim$ 500 m	0.3%	40 $\sim$ 50%	6.5%
500 m $\sim$ 600 m	0.2%	50 $\sim$ 60%	5.9%
600 m $\sim$ 700 m	0.7%	60 $\sim$ 70%	0.9%
700 m $\sim$ 800 m	0.1%	70 $\sim$ 80%	2.0%
800 m $\sim$ 900 m	0.2%	80 $\sim$ 90%	2.4%
900 m $\sim$ 1,000 m	0%	90 $\sim$ 100%	0.3%
		>100%	7.0%

3) Experiments with Synthetic Dataset: Due to the difficulties of collecting a very large set of real-world videos, a synthetic dataset of moving cameras with positions inside a 75 km $\times$ 75 km region was used to test the performance of our

algorithm with large-scale data. We generated metadata using the Georeferenced Synthetic Metadata Generator [37]. We predefined 1000 POIs which are randomly distributed inside the target region, and generated 5,500 moving cameras with trajectories near these POIs. The POIs can be in arbitrary shape and the size is not fixed. Each camera was traced for 1,000 seconds, with a sampling rate of 1/s for the GPS and compass. Thus, the resulting dataset contained about 5.4 million FOVs. In the original generator, the camera rotation can be set at a fixed speed. To simulate a real-world case, we modified the generator in order to make the cameras record the nearby POI areas more frequently.

We applied the grid-based POI detection to this synthetic dataset, and 1,183 POIs were detected. Then we matched the detected POIs to the predefined POIs. We found 942 matching pairs. Hence the precision and recall of the grid-based POI detection is 94.2% (942/1000) and 79.6% (942/1183), respectively, and the number of phantom POIs is 241 (1183-942).

To evaluate our phantom POI elimination algorithm, we applied Algorithm 1 to the detected POI set. The algorithm was able to eliminate 221 POIs. Then we matched the eliminated POIs to the 241 phantom POIs. We found 216 matching pairs. Hence the precision and recall of Algorithm 1 is 97.7% (216/221) and 89.6% (216/241), respectively.

C. Discussion

1) Robustness of Approach: In our experiments we did not make any specific assumptions about the users' video recording style. The robustness of our approach with respect to various anomalies is of course of significant importance. We have some preliminary indication that our method is quite robust and can be further enhanced with existing approaches (*e.g.*, GPS stabilization, content-based processing, etc.). However, we will have a clearer understanding of these issues once we collect a much larger repository of data. Acquiring sensor-rich videos is becoming increasingly easy. Thus, we expect there will be a growing trend for users to capture more videos. We currently have smartphone apps publicly available and are collecting more user-generated content. We plan to do an in-depth analysis of a very large set of data in the future.

With any grid-based algorithm the size of the cells can influence performance. For our grid-based POI detection we have tested different grid sizes from 5×5 m^2 to 30×30 m^2, and we found that the resulting POI distributions are similar across these different settings.

2) Comparison with Alternative Approaches: Our method is complementary to other approaches while it also has some specific strengths. Methods that use content-based analysis, such as Google Goggles [38], require distinctive features of known landmarks (*i.e.*, structures). For example, Goggles may not be able to recognize a famous lake because of a lack of unique features. Our approach crowd-sources "interesting" spots automatically. It is of course possible to enhance our approach by combining it with content-based analysis and thus achieving the best of both worlds. This will require careful study and we plan to explore such an integration in the future.

Our POI estimation is not solely designed to be a standalone method. We take advantage of using existing landmark databases if available. There exists considerable research literature on detecting landmark places from photos. The main difference of our method from existing approaches is that we identify the location of interesting places that appear in users' videos, rather than the location where the user was standing, holding the camera.

V. Conclusions and Future Work

Capturing video in conjunction with descriptive sensor meta-data provides a comprehensive foundation to model the scene that a camera is acquiring. There exists a growing corpus of videos that are annotated with comprehensive geographic sensor information, aided by the convenient availability of smartphone recording apps.

In our study we presented an approach to detect POIs and their distances from the camera location in a fully automated way. We provided two algorithms for POI identification and also a method to estimate the effective visual distance without examining the actual video content, purely based on associated sensor information. In addition, we designed a cross-intersection elimination method to remove non-existing phantom POIs. The experimental results show that our technique is very effective in detecting POIs and estimating their visual distance from the camera location. In our future work we plan to extend our approach in the following aspects:

(i) For the POI detection our two proposed methods each have their own benefits. When targeting large scale applications, we may consider a hybrid strategy to combine the two methods to achieve overall better performance.

(ii) Currently, our visual distance R estimation algorithm only works when there exists one or more than one POIs within the field-of-view. For frames with ambiguous content a user feedback mechanism may be able to help improve the R estimation results.

(iii) Given the estimated distance R, we may use it to adjust the center vector length of the stored field-of-view slices and hence obtain a continuous stream of precise viewable scene descriptions corresponding to the video frames. We plan to utilize such data to facilitate many types of video applications such as video search and presentation.

References

[1] Cisco Systems, Inc., "Cisco Visual Networking Index: Forecast and Methodology, 2010-2015," White Paper, 2011.

[2] "Flickr," http://www.flickr.com.

[3] "Woophy," http://www.woophy.com.

[4] P. Duygulu, K. Barnard, J. De Freitas, and D. Forsyth, "Object Recognition as Machine Translation: Learning a Lexicon for a Fixed Image Vocabulary," in *7^{th} European Conference on Computer Vision*, 2002, pp. 97–112.

[5] J. Sivic, B. Russell, A. Efros, A. Zisserman, and W. Freeman, "Discovering Objects and Their Location in Images," in *10^{th} IEEE Intl. Conference on Computer Vision*, 2005, pp. 370–377.

[6] J. Hao, G. Wang, B. Seo, and R. Zimmermann, "Keyframe Presentation for Browsing of User-generated Videos on Map Interfaces," in *19^{th} ACM Intl. Conference on Multimedia*, 2011, pp. 1013–1016.

[7] B. Seo, J. Hao, and G. Wang, "Sensor-rich Video Exploration on a Map Interface," in *19^{th} ACM Intl. Conference on Multimedia*, 2011, pp. 791–792.

[8] K. Rodden and K. R. Wood, "How do People Manage their Digital Photographs?" in *SIGCHI Conference on Human Factors in Computing Systems*, 2003, pp. 409–416.

[9] K. Toyama, R. Logan, and A. Roseway, "Geographic Location Tags on Digital Images," in 11^{th} *ACM Intl. Conference on Multimedia*, 2003, pp. 156–166.

[10] M. Naaman, Y. J. Song, A. Paepcke, and H. Garcia-Molina, "Automatic Organization for Digital Photographs with Geographic Coordinates," in 4^{th} *ACM/IEEE-CS Joint Conference on Digital Libraries*, 2004, pp. 53–62.

[11] A. Pigeau and M. Gelgon, "Building and Tracking Hierarchical Geographical & Temporal Partitions for Image Collection Management on Mobile Devices," in 13^{th} *ACM Intl. Conference on Multimedia*, 2005.

[12] B. Epshtein, E. Ofek, Y. Wexler, and P. Zhang, "Hierarchical Photo Organization Using Geo-Relevance," in 15^{th} *ACM Intl. Symposium on Advances in Geographic Information Systems*, 2007, pp. 1–7.

[13] C. Torniai, S. Battle, and S. Cayzer, "Sharing, Discovering and Browsing Geotagged Pictures on the World Wide Web," in *The Geospatial Web*. Springer, 2007, pp. 159–170.

[14] L. S. Kennedy and M. Naaman, "Generating Diverse and Representative Image Search Results for Landmarks," in 17^{th} *Intl. Conference on the World Wide Web*, 2008, pp. 297–306.

[15] G. Yavas, D. Katsaros, O. Ulusoy, and Y. Manolopoulos, "A Data Mining Approach for Location Prediction in Mobile Environments," *Data & Knowledge Engineering*, vol. 54, no. 2, pp. 121–146, 2005.

[16] M. Morzy, "Mining Frequent Trajectories of Moving Objects for Location Prediction," *Machine Learning and Data Mining in Pattern Recognition*, pp. 667–680, 2007.

[17] M. Park, J. Hong, and S. Cho, "Location-based Recommendation System using Bayesian User's Preference Model in Mobile Devices," *Ubiquitous Intelligence and Computing*, pp. 1130–1139, 2007.

[18] Y. Takeuchi and M. Sugimoto, "An Outdoor Recommendation System based on User Location History," in *1^{st} Intl. Workshop on Personalized Context Modeling and Management for UbiComp Applications*, 2005, pp. 91–100.

[19] F. Giannotti, M. Nanni, F. Pinelli, and D. Pedreschi, "Trajectory Pattern Mining," in *13^{th} ACM Intl. Conference on Knowledge Discovery and Data Mining*, 2007, pp. 330–339.

[20] R. Hariharan and K. Toyama, "Project Lachesis: Parsing and Modeling Location Histories," *Geographic Information Science*, pp. 106–124, 2004.

[21] Q. Li, Y. Zheng, X. Xie, Y. Chen, W. Liu, and W. Ma, "Mining User Similarity based on Location History," in *16^{th} ACM Intl. Conference on Advances in Geographic Information Systems*, 2008, p. 34.

[22] Y. Zheng, L. Zhang, X. Xie, and W. Ma, "Mining Interesting Locations and Travel Sequences from GPS Trajectories," in *18^{th} Intl. Conference on World Wide Web)*, 2009, pp. 791–800.

[23] R. Ji, X. Xie, H. Yao, and W. Ma, "Mining City Landmarks from Blogs by Graph Modeling," in 17^{th} *ACM Intl. Conference on Multimedia*, 2009, pp. 105–114.

[24] L. Kennedy, M. Naaman, S. Ahern, R. Nair, and T. Rattenbury, "How Flickr Helps Us Make Sense of the World: Context and Content in Community-contributed Media Collections," in 15^{th} *ACM Intl. Conference on Multimedia*. ACM, 2007, pp. 631–640.

[25] Y. Zheng, M. Zhao, Y. Song, H. Adam, U. Buddemeier, A. Bissacco, F. Brucher, T. Chua, and H. Neven, "Tour the World: Building a Web-scale Landmark Recognition Engine," in 22^{nd} *IEEE Conference on Computer Vision and Pattern Recognition*, 2009, pp. 1085–1092.

[26] K. Liu, J. Xu, L. Zhang, Z. Ding, and M. Li, "Discovering Hot Topics from Geo-tagged Video," *Neurocomputing*, vol. 105, pp. 90–99, 2013.

[27] S. Arslan Ay, R. Zimmermann, and S. H. Kim, "Viewable Scene Modeling for Geospatial Video Search," in 16^{th} *ACM Intl. Conference on Multimedia*, 2008, pp. 309–318.

[28] C. H. Graham, N. R. Bartlett, J. L. Brown, Y. Hsia, C. C. Mueller, and L. A. Riggs, *Vision and Visual Perception*. John Wiley & Sons, Inc., 1965.

[29] M. Ester, H. Kriegel, J. Sander, and X. Xu, "A Density-based Algorithm for Discovering Clusters in Large Spatial Databases with Noise," in 2^{nd} *Intl. Conference on Knowledge Discovery and Data Mining*, vol. 96, 1996, pp. 226–231.

[30] Y. Arase, X. Xie, T. Hara, and S. Nishio, "Mining People's Trips from Large Scale Geo-tagged Photos," in 18^{th} *ACM Intl. Conference on Multimedia*, 2010, pp. 133–142.

[31] X. Lu, C. Wang, J.-M. Yang, Y. Pang, and L. Zhang, "Photo2Trip: Generating Travel Routes from Geo-tagged Photos for Trip Planning," in 18^{th} *ACM Intl. Conference on Multimedia*, 2010, pp. 143–152.

[32] L. Page, S. Brin, R. Motwani, and T. Winograd, "The PageRank Citation Ranking: Bringing Order to the Web," Stanford InfoLab, Technical Report 1999-66, November 1999.

[33] A. Hakeem, R. Vezzani, M. Shah, and R. Cucchiara, "Estimating Geospatial Trajectory of a Moving Camera," in 18^{th} *IEEE Intl. Conference on Pattern Recognition*, vol. 2, 2006, pp. 82–87.

[34] G. Wang, B. Seo, Y. Yin, R. Zimmermann, and Z. Shen, "OSCOR: an Orientation Sensor Data Correction System for Mobile Generated Contents," in 21^{st} *ACM international conference on Multimedia*, 2013, pp. 439–440.

[35] G. Wang, B. Seo, Y. Yin, and R. Zimmermann, "Orientation Data Correction with Georeferenced Mobile Videos," in 21^{st} *ACM SIGSPATIAL Intl. Conference on Advances in Geographic Information Systems*, 2013, pp. 390–393.

[36] "Geographic Midpoint Calculator," geomidpoint.com/calculation.html.

[37] S. A. Ay, S. H. Kim, and R. Zimmermann, "Generating Synthetic Metadata for Georeferenced Video Management."

[38] "Google Goggles," http://www.google.com/mobile/goggles/.

Jia Hao is a research fellow at School of Computing, National University of Singapore (NUS). She received the B.E. degree at the Department of Computer Science from Harbin Institute of Technology, Harbin, China, in 2008, and the Ph.D. degree at the School of Computing from National University of Singapore in 2013. Her research interests include sensor-rich video management, mobile and location-based multimedia, streaming media architectures and large scale multimedia systems.

Guanfeng Wang received the B.E. degree in Software Engineering from Zhejiang University, China, in 2010. He is currently a Ph.D. candidate in School of Computing at National University of Singapore. His research interests include mobile media analysis, mobile sensing, geostatistics and multimedia system.

Beomjoo Seo is an Assistant Professor at the School of Games, Hongik University in Sejong, Korea. He received the B.S. degree and M.S. degrees at the Department of Computer Engineering from Seoul National University, Korea in 1994 and 1996, respectively. After working 5 years as a research engineer at LG Electronics, he started his Ph.D. study in Computer Science and received the degree in 2008 from the University of Southern California in Log Angeles, CA. His research was supervised by Dr. Roger Zimmermann. Before joining the Hongik University, he worked as a research fellow at the School of Computing, National University of Singapore in Singapore. His research includes distributed storage model, distributed spatial audio streaming for virtual worlds, sensor-rich mobile video acquisition, annotation and its application.

Roger Zimmermann (S'93-M'99-SM'07) received the M.S. and Ph.D. degrees from the University of Southern California (USC) in 1994 and 1998. He is currently an associate professor in the Department of Computer Science at the National University of Singapore (NUS). He is also a deputy director with the Interactive and Digital Media Institute at NUS and a co-director of the Centre of Social Media Innovations for Communities (COSMIC). His research interests are in the areas of streaming media architectures, distributed and peer-to-peer systems, mobile and geo-referenced video management, collaborative environments, spatio-temporal information management, and mobile location-based services. He has coauthored a book, six patents, and more than 170 conference publications, journal articles, and book chapters. He is a senior member of the IEEE and a member of ACM.