HUGVid: Handling, Indexing and Querying of Uncertain Geo-Tagged Videos

He Ma
School of Computing, National University of Singapore, Singapore 117417
mahe@comp.nus.edu.sg

Roger Zimmermann
School of Computing, National University of Singapore, Singapore 117417
rogerz@comp.nus.edu.sg

Seon Ho Kim
Integrated Media Systems Center, University of Southern California
Los Angeles, CA 90089
seonkim@usc.edu

ABSTRACT

GIS applications now increasingly make use of geo-located multimedia data such as images and videos. Furthermore, the wide-spread availablity of smartphones allows the acquisition of user-generated videos that are annotated with geo-properties. The sensor meta-data, *e.g.*, GPS and digital compass values, are considerably smaller in size than the visual content and are helpful in effectively and efficiently manage and search through large repositories of videos. However, a major practical issue is the noisy nature of such sensor data. For example, due to sensor data inaccuracies the visual coverage described by the meta-data may not exactly match the actual video scene, which leads to imprecise search results and positional disagreements on map overlays. Obstructions between the camera and its captured objects make these situations worse. Therefore, robust error-tolerance is an essential feature of any geo-tagged video search application.

To this end we introduce HUGVid, a modeling and indexing approach for uncertain geo-tagged videos. We construct an uncertainty model for video frames and segments. Since the frame-by-frame uncertainty model involves high computational complexity, we then propose an approximate modeling method based on a video segmentation algorithm which eliminates costly overlap calculations between the query region and individual frames. Finally, we test the performance of HUGVid with both two real-world and a large-scale synthetic dataset. Experimental results show that our method achieves high precision and good scalability and allows the efficient querying of noisy sensor data. HUGVid also returns confidence probabilities with the results which can then be beneficially used in upstream GIS applications.

Categories and Subject Descriptors

G.3 [**Probability and statistics**]: Probabilistic algorithms; H.2.4 [**Systems**]: Query processing; I.3.5 [**Computational Geometry and Object Modeling**]: Curve, surface, solid, and object representations

ACM SIGSPATIAL GIS '12, November 6-9, 2012. Redondo Beach, CA, USA

General Terms

Algorithms, Measurement, Performance

Keywords

Uncertain data modeling, Geo-tagged video, Video segmentation and indexing

1. INTRODUCTION

With the rapid improvement of video capture technology, user-generated videos (UGV) can now easily be acquired, and smartphones (and increasingly tablets) have emerged as a major source of such UGVs. Moreover, a number of sensors (*e.g.*, GPS and compass units) have been cost-efficiently embedded in such devices in recent years. Consequently, some very useful meta-data, especially geographic properties of video scenes, can easily be collected automatically during video recording. The association of video scenes with their geographic meta-data on commodity hardware has allowed their cost-effective use in consumer and professional GIS applications. This trend has also raised some interesting research issues. For instance, the recorded sensor meta-data can be utilized to model, index and search geo-tagged videos with semantics that are familiar to users.

Some prior research [1, 16] has modeled the coverage region of video frames as a pie-shaped geometric area described by the sensor meta-data, such as the camera location, viewing direction and visible distance. However, an important practical aspect is often neglected in existing solutions: the noisy nature of sensor data. It has long been known (and many users have had first-hand experience) that sensors, especially on consumer-grade electronics, produce sometimes inaccurate and fluctuating values. The surrounding environment can exacerbate the problem. For example, tall buildings and narrow passageways in urban areas can lead to very difficult conditions for obtaining accurate GPS positions [17]. Typically alignment errors, non-orthogonal errors and magnetic deviations can affect the digital compass heading accuracy. Moreover, obstructions (such as buildings, people or vehicles passing by) in front of a camera may in parts of scene result in the recording of objects far away (*e.g.*, 1,500 m) while only close ones in other parts within the same frame. The latter effects influence the captured video scene, but not the measured sensor data.

The above described issues lead to a mismatch between the viewable scenes of video contents and the area represented by the sensor data. Rather than trying to completely avoid or correct such issues (which may be difficult or im-

possible), a well-known approach in the information management community is to design methods that can handle *uncertain data*.

In this study we propose a novel approach which has the objective of modeling the uncertainty of a camera's viewable scene in the presence of noisy, imprecise sensor data. An approximate method is introduced for an efficient, but still effective, estimation of the uncertain data model. Based on the approximate model and geographical properties of a video, a video segmentation method is introduced which helps to describe the probability of a region being captured by a video segment. Finally we introduce efficient methods to perform probabilistic queries. The HUGVid architecture and data flow are presented in Figure 1. The contribution of our work can be summarized as follows:

- **Uncertain data model for individual geo-tagged video frames.** We model the spatial coverage of an individual video frame with multi-sensor meta-data and an approximate method for effectively calculating the probability of any place being captured by this model is introduced.

- **Video segmentation and uncertain data model for segments.** We introduce a video segmentation algorithm with the aim that all frames within the parsed video segment capture a common region. This region is recorded by all frames with a non-zero probability. At the same time, we extend the uncertain data model to video segments.

- **Video indexing with extended R-tree.** For effective and efficient search, an R-tree based method is proposed to index the parsed video segments with geo-properties. Additionally, the statistical information of each video segment is attached to a standard R-tree as a secondary index.

- **Distance estimation.** When videos are uploaded, a centroid point is used to represent all the cameras' locations. We then utilize this point to approximately calculate the distance between the query and the video segment.

The above novel components of HUGVid allow the efficient processing of probabilistic queries over naturally noisy sensor data. Upstream GIS applications can utilize the results and prioritize their processing by concentrating on the most promising candidate video segments first.

We validate our design through extensive experiments with both real and synthetic datasets. The results show that the proposed approach produces high precision, and the approximate distances computed match well with the actual values. The result of a user study shows that our method also satisfies the human perspective. Moreover, the performance on the synthetic dataset indicates that our approach performs efficiently and effectively on large-scale datasets.

The rest of the paper is organized as follows. Section 2 summarizes the related research. Section 3 introduces the uncertain data model for a video viewable scene. Section 4 demonstrates the geo-tagged video indexing method. Section 5 details the query processing. Section 6 reports the results on the evaluations of the proposed method. Section 7 concludes the paper.

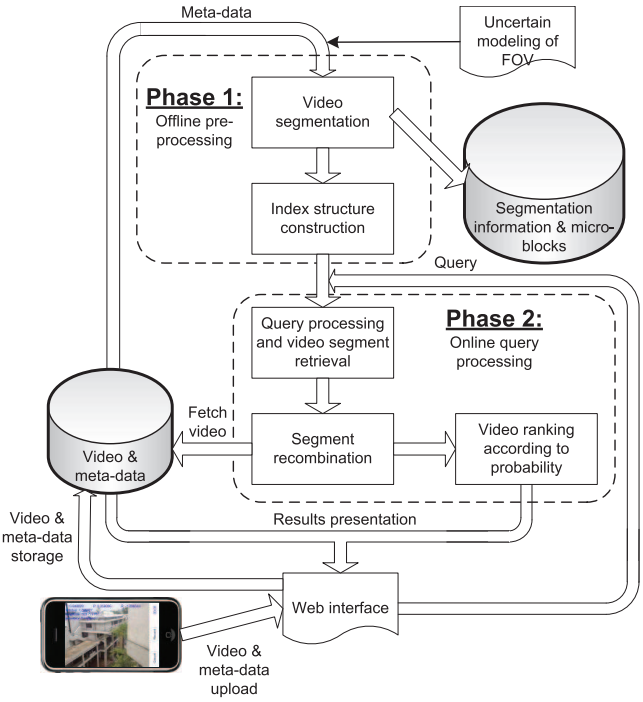


Figure 1: Architecture with uncertain data model and the corresponding tasks for processing queries.

2. RELATED WORK

Associating geographic (*e.g.*, location and orientation) information with a camera's video data for retrieval purposes has become an active research area [27, 20]. Hwang *et al.* [12] and Kim *et al.* [14] proposed a mapping between the 3D world and videos by linking objects to the video frames in which they appear using GPS location and camera orientation. Other authors [1, 16] proposed a model to represent a camera viewable scene with a shape of a pie-slice, based on GPS and digital compass data as well as the visible distance.

When modeling geo-tagged video with meta-data, the uncertainty of the sensor data affects the search results. There exist two categories of techniques for indexing uncertain data with arbitrary probability density functions (PDF). The first type is based on probabilistically constrained regions (PCRs) [5, 25, 7]. It uses a so-called "x-bound" to restrict the probability of a region so that candidates can be pruned when processing probabilistic constrained queries. The second type is based on R-tree index structures [6, 13, 21, 15, 23, 26]. More specifically, the uncertain region of multidimensional uncertain objects is grouped with an R-tree, where each data unit is the minimum bounding rectangle (MBR) of a PDF. Kim *et al.* [13] attached a secondary index for fast access to the leaf nodes. However, none of above methods is appropriate for a camera's viewable scene with a longer visible distance. Since our approach represents each video segment as a spatial object, we extend the standard R-tree [11] by associating additional information to index such videos.

People may capture different places of interest (POI) within the same video but users may only be interested in the parts of the video that show a specific place. Therefore, parsing the video into segments to extract a specific place is essential for geo-relevant applications. The aim of video parsing can be achieved by using content-based methods (*e.g.*, shot

boundary detection [9], hidden Markov models [3], or graph partitioning [4]). Alternatively, Navarrete and Blat [19] utilized geographic information to parse and index video. To the best of our knowledge, there has been little research focus on parsing videos according to geo-properties.

3. PRELIMINARIES

There is an increasing awareness of the naturally occurring uncertainty of some spatial data, for example in the GIS and other communities. The uncertainty falls into two cases: sensor errors (Section 3.2.1 and 3.2.3) and obstacles (Section 3.2.2). In our geo-tagged video search application, the mismatch between video scenes and the associated geographical coverage significantly affects the final results. For example, when users desire to search for videos that capture the Marina Bay Sands (MBS) complex in Singapore, the system may find the videos that can theoretically capture the MBS according to geometric calculations. However, if an object is located between the camera and MBS and hides the building, or if the meta-data are inaccurate, the actual video may not capture the hotel complex. Therefore, an uncertain model for the camera's viewable scene is essential to measure the probability that a location actually appears in video scenes. When taking videos using smartphones, the GPS devices are collecting the location information as well as the accuracy. We manually checked the GPS accuracies and compass readings and found that most errors are within a certain range. Section 3.1 first presents the basic viewable scene model that is based on sensor data which is assumed to be correct. Then the extended, uncertain viewable scene model is introduced in Section 3.2, and finally Section 3.3 presents the simplified, approximate model.

3.1 Field-of-View (FOV) Model

The *camera viewable scene* is what a camera in geo-space captures. This area is referred to as *camera field-of-view* (FOV for short) with a shape of a pie-slice. As shown in Figure 2, the FOV coverage in 2D space can be defined with four parameters: the camera location P, the camera viewing direction vector $\vec{d}$, the viewable angle α, and the maximum visible distance R_V. The camera location P is the $\langle latitude, longitude\rangle$ coordinate read from a positioning device (*e.g.*, GPS and/or Cricket coordinates [22]). The viewing direction vector $\vec{d}$ can be acquired from a digital compass. We use θ to represent its value with respect to the North. The camera's viewable angle α is calculated based on the lens properties for the current zoom level [10]. The visible distance R_V is the maximum distance at which a large object within the camera's FOV can be recognized. Thus, the camera viewable scene at time t is denoted by the tuple $FOV(P\langle lat, lng\rangle, \theta, \alpha, R_V, t)$. The tuple values are acquired from embedded sensors during video recording.

3.2 Uncertain Data Model for FOV

As illustrated in Figure 3, the solid pie-slice shape is formed from the sensor meta-data while the dashed shape might be the actual video scene. For a single FOV, it is commonly assumed that an object has a higher probability of being captured by a camera if it appears close to the camera's location and in the center of the FOV. If not otherwise specified, we refer to the *probability* as the probability of an object being captured by an FOV or a video segment. Next, we describe the uncertainty of meta-data that affects the

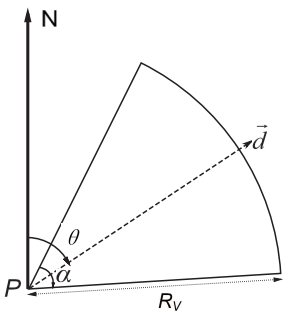


Figure 2: Illustration of the field-of-view (FOV) model in 2D space.

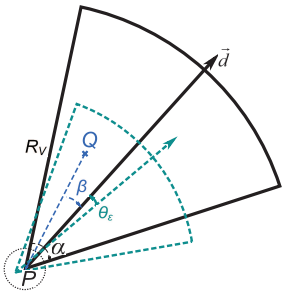


Figure 3: Illustration of the uncertainty of a field-of-view (FOV).

probability based on the FOV model in three independent dimensions: camera viewing direction, visible distance and camera location. Finally we will combine them together.

3.2.1 Uncertainty of Viewing Direction

Due to the inaccuracy of the digital compass, the measured viewing direction does sometimes not exactly align which the camera direction. We manually checked the captured video and the associated compass data, and found that the distribution of the compass error satisfies a Gaussian function with a maximum error value. Based on this observation, we assume that the maximum compass error is θ_ε ($\theta_\varepsilon < \alpha$). Thus, the camera can definitely capture the directions ranging from $(\theta - \frac{\alpha}{2} + \theta_\varepsilon)$ to $(\theta + \frac{\alpha}{2} - \theta_\varepsilon)$, while probably record other directions. As shown in Figure 3, the probability that a given location Q (*i.e.*, the query input) is covered by an FOV is presented in Eqn. 1:

$$Prob_Q^\beta = f(\beta) = \begin{cases} 1 & (\beta \le \left|\frac{\alpha}{2} - \theta_\varepsilon\right|) \\ e^{\left(\frac{K_0 \cdot \left|\beta - \frac{\alpha}{2} + \theta_\varepsilon\right|^2}{\left|\frac{\alpha}{2} + \theta_\varepsilon\right|^2}\right)} & (\beta > \left|\frac{\alpha}{2} - \theta_\varepsilon\right|) \end{cases} \quad (1)$$

where β denotes the angle offset between $|PQ|$ and $\vec{d}$, and K_0 is a constant ensuring that it is a small probability event that Q is covered by the FOV when β is larger than $\frac{\alpha}{2} + \theta_\varepsilon$.

3.2.2 Uncertainty of Obstacles

Most current smartphones lack an optical zoom and it is difficult to extract the focal length information. This leads to difficulties when measuring the distance between a camera and the objects it records. Moreover, due to obstructions, it is possible that some objects are hidden by others, or objects at different distances appear in the same scene. This situation has no effects on the sensor data but affects the final search results. Taking the scenes extracted from a video as an example (shown in Figure 4), in this case the camera neither moves nor rotates much during video recording. Due to the bus passing by, the Esplanade building appearing at time 00:01:20 is hidden for a few seconds. Our conjecture is that when an object is closer to the camera, the probability of it being blocked by other objects is lower. Thus, the probability of the object captured by the camera is higher. A Gaussian function is therefore one of the possible ways to represent the probability of objects not obstructed by others (Eqn. 2):

$$Prob_Q^d = g(d) = e^{\left(\frac{K_1 \cdot |d|^2}{|\sigma|^2}\right)} \quad (d \le R_V) \quad (2)$$

Figure 4: Demonstration of the uncertainty of the visible distance. The Esplanade building (highlighted) has been hidden by a bus for a few seconds.

where d is the distance between the camera and the object, and K_1 and σ are the parameters to satisfy that it is a small probability event that any object outside of R_V is captured by the camera.

Hence, we obtain the uncertainty model for the FOV whose camera location is at a specific position (*e.g.*, with geo-coordinates $P(x_0, y_0)$) by combining the above two uncertainty models. Eqn. 3 shows the probability that $Q(x, y)$ is captured by the FOV when the camera is located at $P(x_0, y_0)$. All the parameters used here can be obtained from sensors or the configuration settings of the cameras.

$$Prob_Q^{P(x_0,y_0)} = Prob_Q^{\beta} \cdot Prob_Q^{d}$$
$$= \begin{cases} Prob_Q^{d} & (\beta \leq (\frac{\alpha}{2} - \theta_\varepsilon)) \\ e^{\left(\frac{K_0 \cdot (\frac{\pi}{2} - \gamma - \theta)^2}{(\frac{\alpha}{2} + \theta_\varepsilon)^2}\right)} \cdot Prob_Q^{d} & (x > x_0 \wedge \beta > (\frac{\alpha}{2} - \theta_\varepsilon)) \\ e^{\left(\frac{K_0 \cdot (-\frac{\pi}{2} - \gamma - \theta)^2}{(\frac{\alpha}{2} + \theta_\varepsilon)^2}\right)} \cdot Prob_Q^{d} & (x < x_0 \wedge \beta > (\frac{\alpha}{2} - \theta_\varepsilon)) \end{cases} \quad (3)$$

Here $Prob_Q^d = e^{\left(\frac{K_1 \cdot ((x-x_0)^2 + (y-y_0)^2)}{\sigma^2}\right)}$, $\gamma = \arctan(\frac{y-y_0}{x-x_0})$, and $\beta = |\gamma - \theta|$.

3.2.3 *Uncertainty of Camera Location*

When measuring a location with a GPS device, the position reading is accompanied by an error range d_ε. Therefore, the actual position of the camera is located within a circle around the GPS reading with a radius of d_ε (as shown in Figure 3). Due to various reasons (*i.e.*, a tunnel traversal), the GPS locations are sometimes missing for a while. In this case, we estimate the object location by applying positional interpolation techniques [1]. Sistla *et al.* [24] proposed that the object location follows a Gaussian distribution inside the uncertainty region, as shown in Eqn. 4:

$$Prob_Q^L = h(x_0, y_0) = \frac{e^{-\frac{1}{2(1-\rho^2)} \cdot [\frac{(x_0-\mu_x)^2}{\sigma_x^2} - \frac{2\rho(x_0-\mu_x)(y_0-\mu_y)}{\sigma_x \sigma_y} + \frac{(y_0-\mu_y)^2}{\sigma_y^2}]}}{2\pi\sigma_x\sigma_y\sqrt{1-\rho^2}} \quad (4)$$

where (μ_x, μ_y) denotes the GPS reading, $Prob_Q^L$ denotes the probability that the camera is at location (x_0, y_0), and σ_x, σ_y and ρ are parameters of the probability density function. Finally, the probability of a location to be covered by an FOV is the accumulation of different possible camera positions (see Eqn. 5). The distribution of the probability is illustrated in Figure 5 with a series of parameters.

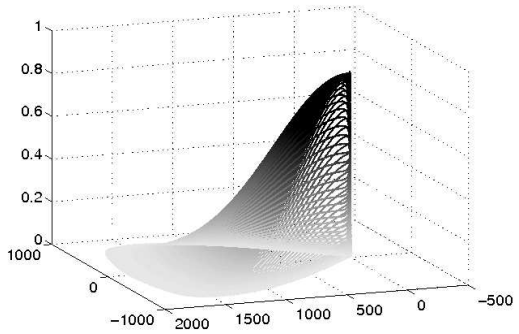


Figure 5: The probabilistic distribution of an area being captured by an FOV at a specific position.

$$Prob_Q = \sum_{d=0}^{d_\varepsilon} Prob_Q^L \cdot Prob_Q^{P(x_0,y_0)} \quad (5)$$

3.3 Approximate Uncertain Data Model

As observed from Figure 5, the coverage region of the probability in 2D space is an irregular shape (the solid shape in Figure 6). Given such an irregular shape, it is computationally expensive to calculate exact probabilistic values, especially for a large set of FOVs. Therefore, we have developed an approximate method to estimate the probability, which can be carried out with a few calculations. As presented in Sections 3.2.1 and 3.2.2, the probability is independent of the direction and the distance domains. For example, if the distance d between the camera and an object is larger than a certain value (*e.g.*, 240 m), the probability affected by distance is smaller than a value, *e.g.*, 50%. Hence, according to Eqn. 3, the probability must be smaller than 50% no matter what value β is. The same situation occurs for the direction domain. Consequently, we can find a border value for each of the two domains, denoted β_b and $dist_b$, respectively, below which the probability is less than a threshold τ. These two border values form a pie-slice shaped region, ensuring that the probability is smaller than τ outside this region. We then use this area to estimate the probability instead of the irregular shape. In Figure 6, P is the camera location according to the GPS reading while P' is a reference point for calculating β_b and $dist_b$. The geo-coordinates of P' can be computed by using R_V, d_ε, α and the location of P. The irregular shapes within the solid lines are the actual probabilistic regions while the pie-slice shapes within the dashed lines are the approximate regions. The exact probability falls on the solid line, *i.e.*, the probability is 50% on the inner solid lines. The regions between the solid lines and the dashed lines are so-called "false positive" regions. For example, the probability of the region between the inner solid shape and the inner dashed lines is actually less than 50% while it is considered greater than or equal to 50% in the approximate model.

4. OFFLINE UNCERTAIN DATA MODELING OF VIDEO SEGMENTS

The approximate model described in Section 3 applies to a single FOV, indicating that it works for geo-tagged images. Building on this, we represent the coverage of a video clip

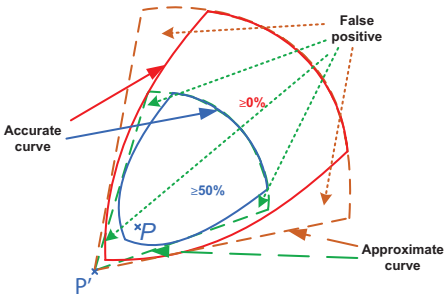


Figure 6: Illustration of the approximate uncertain FOV model in 2D space.

as a spatial object through a series of FOVs. Therefore the next step is to extend our probability model from a single FOV to a sequence, representing a video segment. This is performed by parsing the video into segments according to the associated geo-properties of the FOVs. Additionally, to support large-scale geo-tagged video search applications, we index these spatial objects using an extended R-tree such that they can be effectively and efficiently retrieved. In Section 4.1, the uncertain data model of a video segment is introduced and a video segmentation algorithm is presented. Section 4.2 provides the details of constructing the index structure.

4.1 Uncertain Data Model for Video Segments

Based on the introduced uncertain data model for a single FOV, we now develop the uncertain model of a video segment. One of its benefits is that it can retrieve meaningful results even though there exist sharp jitters among sequential sensor data, which is an essential feature for a robust system. Treating an entire video segment as a spatial object is semantically more meaningful compared to dealing with individual frames. However, treating the whole video as an object may not be a good choice since some long videos or videos captured at high speed may cover a large region. Moreover, processing the entire video results in redundant calculations and increases processing time. Our goal is hence to parse the video into segments and model the probabilistic distribution of each such segment.

Broadly speaking a video segment is represented by its coverage region. When recording videos, people generally point the camera at interesting places even if they are moving along a trajectory. The common, overlapping region among multiple FOVs can hence be considered a POI. For this reason we aim to parse the video such that the FOVs within each parsed segment overlap with each other. This way the likelyhood is high that there exists at least one local POI captured by each video segment. A search will return a video segment without any further processing if users want to search for videos showing a local POI. To parse the video in this manner, the overlapping area between FOVs needs to be calculated. Once a geo-tagged video is uploaded to the server, the video parsing process is activated. It scans the meta-data forward from the beginning and calculates the overlap among the FOVs until a subsequent FOV is identified which does not intersect with the current overlap region. Then the video is partitioned before this FOV and a new segment is started. Figure 7 shows a video clip captured by a camera moving along a trajectory. The video is parsed into three segments illustrated in different line styles. The latticed regions depict the overlap areas, indicating local POIs within the parsed video segment.

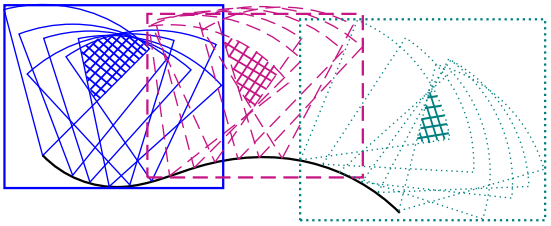


Figure 7: Illustration of the video segmentation algorithm. The video is divided into three segments and the latticed regions illustrate the common overlap within each segment.

Next, we introduce the overlap calculation and design the uncertain data model of a video segment based on the overlap calculation. We define the probability that an object is captured by a video segment as follows:

Definition 1. The probability of a position being captured by a video segment is the average probability of it being captured by all frames within this segment (formalized in Eqn. 6):

$$Prob(P(x,y), VS(0,n-1)) = 1 - \prod_{i=0}^{n-1} (Prob_{cap(i)}\overline{(x,y)}) \quad (6)$$

Here VS denotes the video segment, and $Prob_{cap(i)}(x,y)$ is the probability that an object at location $P(x,y)$ is captured by the i^{th} FOV of the segment. With this definition the probability is high when the query region is close to P and along the camera's viewing direction $\vec{d}$, and when most of the FOVs within the segment point towards P. We subsequently use this probability to help rank the results.

It is computationally complex to find the overlap regions between FOVs with a pie-sliced shape. The Monte Carlo method [18] is one approach for estimating the area of a shape by generating random sampling points. In our geo-tagged video search application, the maximum visible distance R_V is usually hundreds or thousands of meters. Thus, it is impractical to perform the overlap calculation with dense random sampling points. Instead, we divide the MBR of the segment into micro-blocks of size $\delta \times \delta$ ($\delta \ll R_V$). The probability within each micro-block can be treated as constant since δ is much smaller than R_V. We select the center point of each block as the sampling point during the Monte Carlo procedure. All the other points inside a micro-block are treated equivalently to its center point, *i.e.*, the whole micro-block is considered as the overlapping area among FOVs if the center point of this micro-block is covered by all the FOVs. Note that a smaller δ achieves higher accuracy with this method. The current settings used in our experiments work for most cases. The only possible difficulty is an overlap at the border of an FOV, which usually has little or no effect on the final results and may be unimportant to users. The tradeoff is that retrieving more accurate results will cost more processing time and storage space. Since the probability of each micro-block is only slightly different from its adjacent micro-blocks, it is not necessary to maintain the exact value for each one. To simplify the calculation, we quantize the probabilistic range into several levels, which decrease as the probabilistic level falls. Any location out of

the approximate FOV model is assigned zero.

Next, we present the video segmentation algorithm based on the overlap among micro-blocks. When parsing a video clip, the first FOV is selected as a reference. Initially the probabilities of all micro-blocks within this reference FOV are calculated and all blocks are stored in the overlap set *overlapBlocks*. When processing the next FOV, blocks that fall within both FOVs are retained in *overlapBlocks*, while others are moved to the set *non–overlapBlocks*. The probability of each block is updated using Eqn. 6. A video segment is completed once *overlapBlocks* is empty. Note that if the latest inserted FOV has no overlap with the ones ahead, an undo procedure is carried out to guarantee that there exists overlap within a video segment. The detailed process is shown in Algorithm 1. The overlap region is usually very important to the user. Although this parsing algorithm is based on a heuristic and may result in missed global POIs, it still mines the local ones. To overcome this limitation, a post-processing procedure is carried out when retrieving the results during querying.

Algorithm 1: VideoSegmentation()

Input: a video clip $V_i\{FOV_{i0}, FOV_{i1}, ..., FOV_{im}\}$
Output: video segments $VS\{V_{i0}, V_{i1}, ..., V_{in}\}$ $(m \geq n)$

```
1  overlapBlocks{} ← ∅, non−overlapBlocks{} ← ∅
   /* FOV_i0 stands for the 0th FOV in Video V_i          */
2  for all blocks within FOV_i0 do
       /* B_j stands for the jth block in space          */
3      overlapBlocks.append(B_j);
4  end
5  videoCount = 0;
6  for f=1 to m do
7      for all blocks B_t within FOV_if do
8          if B_j==overlapBlocks.find(B_t) then
               /* B_j is called a hit block              */
9              Prob(B_j).update();
10         end
11         else
12             non−overlapBlocks.append(B_t);
13         end
14         move all non-hit blocks from overlapBlocks{} to
           non−overlapBlocks{};
15         if overlapBlocks{} is not empty then
16             VS_(videoCount).append(F_if);
17         end
18         else
19             calculate MBR of the segment;
20             videoCount+=1;
21             undo for the last inserted FOV;
22             VS_(videoCount).append(F_if);
23             reset all sets;
24             for all blocks B_t within FOV_if do
25                 overlapBlocks.append(B_j);
26             end
27         end
28     end
29 end
30 return VS{V_i0, V_i1, ..., V_i(videoCount)};
```

4.2 Video Segment Index Structure

Since we represent the coverage of a video segment as a spatial object we can utilize a popular spatial index structure such as the R-tree to help effectively and efficiently search for requested videos. As described in Algorithm 1, the MBR of a video segment is obtained once the segment is partitioned from the original video. In our geo-tagged video search application, we construct an extended R-tree that regards each parsed video segment as an entry. The difference from a standard R-tree is that the original video ID, the anchor of the starting frame and the segment length are attached to the leaf nodes. In order to quickly process kNN queries, we also store the camera location information in the leaf nodes. However, it is usually impractical and unnecessary to store all the camera locations within a segment. We instead use a centroid point to represent all the camera positions. The geo-coordinates of this centroid represent the average of all the camera locations in both latitude and longitude and we store the micro-block information in which the centroid is located. This information is utilized to estimate the average distance between the query and camera locations and is treated as the distance between the query and a video segment. Additionally, the corresponding probability map of each segment is attached to the leaf nodes as a secondary index.

An alternative way to store these maps is to compress them utilizing a discrete cosine transform (DCT). We experimented with compressing the maps to a quarter of their original size and found that the probability level error rose to at least 10%. Furthermore, a DCT calculation needs to be performed for each of the candidate maps, which is inappropriate for online queries due to the increased processing time. Consequently, we decided to store the probability maps directly on disk.

5. ONLINE QUERY PROCESSING

Given the offline pre-processing calculations described in Section 4, the online query procedure is performed in two major steps. First, the result video segments (referred to as *candidates*) are retrieved by searching through the R-tree. In the second step, the final segments are reconstructed by combining candidates to preserve video continuity. Note that our approach can retrieve videos without overlap calculations between the query region and the individual video frames. Also, the probability is updated after recombination. Importantly, when presenting the results the video segments are sorted according to their probability.

5.1 Searching for Segment Candidates

Utilizing the R-tree index we process both typical spatial queries (including point, range and kNN queries) and queries with a probabilistic threshold to search for videos in large-scale datasets. Since we use an MBR to represent the video coverage, all the queries are processed with overlap calculations between query locations and MBRs. Generally, for all query types, we need to search from the root to the leaf nodes of the R-tree to find the video segments whose MBRs overlap with the query. Once the candidates are obtained, the corresponding probability maps are fetched.

The basic query procedure is described in the above paragraph, while the differences among various types of queries are as follows:

- **Point Query:** Once the leaf nodes containing candidates are retrieved from the R-tree, the corresponding probability maps attached to the leaf nodes are loaded into memory. Then the micro-block is identified in which the query point is located and the probability of that micro-block is obtained.
- **Range Query:** The difference to the point query is that the query rectangle may cover more than one

micro-block. Hence, the maximum probability among all overlapping micro-blocks is selected as the result.

- **kNN Query:** The distance between the query and a video segment is defined as the average distance between the query and all its frames. When processing a kNN query, we approximately calculate the distance between the query point and the centroid point of the micro-block whose information is stored in the leaf nodes of the R-tree. Thus, we treat this as the distance between the query and the video segment and use it to rank the results.
- **Query with Probabilistic Threshold:** The basic processing of a query with a probabilistic threshold is the same as with the above typical queries. The difference lies in filtering out the candidates whose probabilities are smaller than the threshold and then reconstruct the candidates.

5.2 Candidate Recombination

As presented in Section 4.1, the video segmentation algorithm may result in missed POIs. In that case, frames that capture the same POI may be parsed in two consecutive segments, which is not what users expect. Considering the example in Figure 7, the last two FOVs of the first segment and the first three FOVs of the second segment cover a common region. When the query is located in these regions, the segments indexed by the R-tree may not be the optimal representation of the results. Consequently, when the results include segments that are contiguous in the original video, a recombination operation needs to be carried out as detailed in Algorithm 2. Consecutive segments in the results are re-combined into new segments. The procedure terminates when there are no more such segments. During this process, the probability is also updated using Eqn. 7. The videos are then ranked in descending order according to their probabilities. Alternatively, for kNN queries, the videos are ranked according to their distance from nearest to farthest without recombination.

$$Prob(V_{ij}) = 1 - (1 - Prob(V_i)) \cdot (1 - Prob(V_j)) \quad (7)$$

Here V_i and V_j are two consecutive segments and V_{ij} denotes the combination of these two segments.

Algorithm 2: Recombination()

```
Input: Candidate set vSet{V_0, V_1, ..., V_n}
Output: Results rSet{V_r0, V_r1, ..., V_rm} (m ≤ n)
for all segments V_i in vSet{} do
    if V_i.findAjacent() == V_j then
        V_ij = combine(V_i, V_j);
        vSet.remove(V_i); vSet.remove(V_j);
        vSet.append(V_ij);
        prob(V_ij).update();
    end
    else
        vSet.remove(V_i); rSet.append(V_i);
    end
end
return rSet{V_r0, V_r1, ..., V_rm};
```

6. EXPERIMENTAL EVALUATION

We performed our experiments on three datasets: two small sets of real-world videos and sensor meta-data, and one large synthetic dataset. We used the real datasets to test the functionality of HUGVid and the synthetic dataset to demonstrate its scalability for large-scale applications. For all the experiments we constructed a local MySQL database in which we stored the FOV meta-data. We inserted the MBRs of all the parsed video segments and the relevant statistical information into our extended R-tree and processed all types of queries based on the R-tree [8] implemention by Greg Douglas. We found this implementation to be very mature and achieve excellent performance. Additionally, we treated each FOV as an object and indexed it using the same structure to form a *baseline method* (BM) for comparison.

K_0	α	θ_ε	K_1	σ	d_ε	δ	R_V
-23.237	60 °	10 °	-0.137	300m	10m	20m	2km

Table 1: The parameters used in the construction of the uncertain data model.

In the following experiments, if not otherwise specified, the micro-block size δ was set to 20 meters, which is small compared to the camera's maximum visible distance R_V. R_V may vary due to different devices and resolutions. For example, many smartphones now can record 1080p HD video. According to existing methods [1], the maximum visible distance of, for example, an iPhone 4S can be estimated as 1470 meters since its CMOS sensor size is 1/3.2 inches and its focal length is 35 mm. To make our approach robust with videos captured from different devices, we set the value of R_V as 2 km. The angle α may be calculated with the image sensor size and the camera focal length of the lens [10]. However, it is difficult to obtain the focal length of the camera on a mobile device and hence to calculate the precise value of α. Due to videos being captured with different smartphones, α might vary among different devices. Therefore, we use a large, practical value for α, which is set to 60 °. The values of θ_ε and d_ε are obtained from real-world data. We manually checked GPS accuracies in our data and found that over 86% of GPS errors fall within 10 m, and that over 90% of the compass reading errors are less than 10 °. Other parameters are set to satisfy a small probability event when constructing the uncertain data model (presented in Section 3.2). The detailed parameters are summarized in Table 1.

6.1 Experiments with Real-world Dataset

We collected 71 geo-tagged videos (representing about 190 minutes) around the Marina Bay Reservoir in Singapore, and 247 videos (about 400 minutes) in Chicago, recording the event of the NATO Summit 2012 using smartphones. The lengths of videos vary from less than one minute to 14 minutes. Since the range and kNN queries are representative among the query types, we used the collected data to process these queries and demonstrate the functionality of HUGVid. We selected five landmark places in Singapore (the Marina Bay Sands, the Merlion, the Esplanade, the Singapore Flyer, and the One Marina Boulevard) and two in Chicago (Chicago City Hall and Bulter Field) for range queries, which we refer to as *Q1* to *Q7*. Within each query region, one representative point is selected for the kNN queries. Figure 8 shows a sampling of frames for *Q1* in the Marina Bay Sands region. The query area is the solid rectangle shown on Google Maps and the 3D image extracted from Google Earth illustrates what the videos are assumed to capture. The ten surrounding images are sampling snapshots from the result video segments with their respective probabilities, pinned to their camera locations.

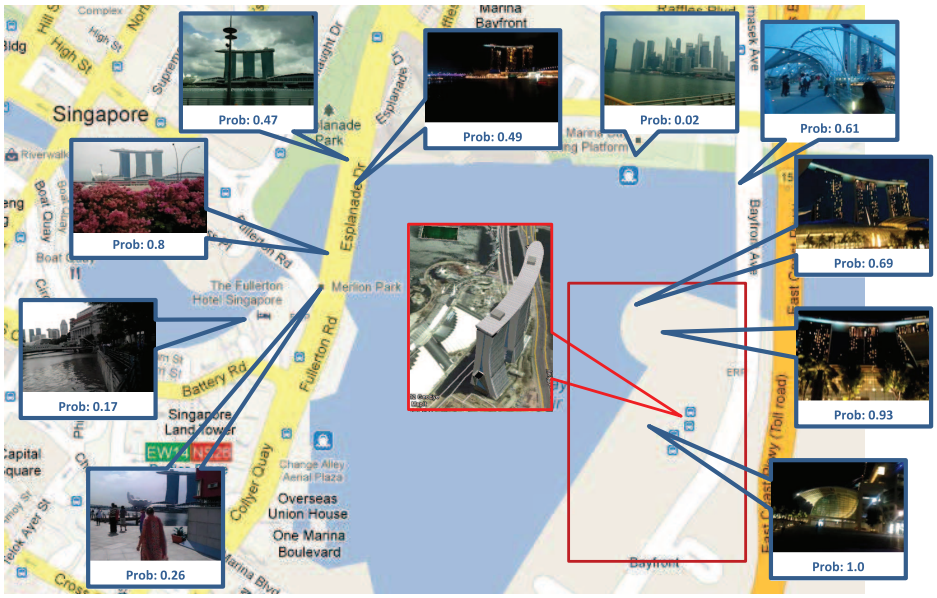


Figure 8: Sample frames for query *Q1* in the Marina Bay Sands region. Ten snapshots are chosen from resulting videos with different probabilities.

Query	Precision	Recall
Q1	0.9976	0.7675
Q2	0.9948	0.4821
Q3	0.9971	0.6968
Q4	0.9967	0.6973
Q5	0.9959	0.5644
Q6	0.9742	0.6749
Q7	0.9655	0.6518

Table 2: The precision and recall of HUGVid with different queries.

Q1	V1	V2	V3	V4	V5
Algorithm score	5	4	3	2	1
Average score	3.05	4.24	3.33	2.38	2.00
Standard deviation	1.717	1.091	0.966	0.973	1.140
Q2	V1	V2	V3	V4	V5
Algorithm score	5	4	3	2	1
Average score	4.86	4.00	3.00	1.90	1.24
Standard deviation	0.359	0.632	0.447	0.539	0.539

Table 3: Scores from ranking by the HUGVid algorithm and by the users (1 – least, 5 – most relevant).

The figure nicely illustrates how frames with higher probabilities capture the target landmark well and at close range while the ones with lower probabilities only capture parts or none of the target, or at a far distance.

Precision and recall. Next, we studied the accuracy and redundancy of HUGVid. We manually watched all the videos and recorded the IDs of video frames which showed the query location. This was considered as the ground-truth (GT for short). We then compared the video segments retrieved using HUGVid with the GT. The precision and recall of HUGVid is presented in Table 2. The high precision shows that HUGVid retrieves almost all the video scenes in the GT. Conversely, it also includes some FOVs not in the GT, which leads to the low value for recall. The reason is two-fold: first, the probabilistic method finds more possible FOVs using the uncertain data model, and second, extra FOVs are included during video segmentation. Although more FOVs are found by HUGVid, it returns only half of the video segments after segment recombination.

User study. It is difficult to find an objective method to evaluate the query results of searching visual content and perform ranking. Therefore, a user study is an appropriate methodology to evaluate how well our approach satisfies the user perspective. Our study involved 21 persons (11 females and 10 males). The participants, which included students and professionals working in different fields (*e.g.*, computer science, biological engineering, and public services), were familiar with the query region. We processed *Q1* to *Q5* and then selected five different video segments (overall about 30 minutes) according to their probabilities from each query result. We chose segments of different probabilistic levels, *e.g.*, one segment with a probability higher than 0.8, one with a probability higher than 0.6 and lower than 0.8, and so on. This made it easy for the users to differentiate. We ranked these five segments according to their probabilities and scored them in descending order. The participants were then asked to watch these videos and rank the segments according to the time duration, the position, and the integrality of the queried place appearing in the scene, while ignoring the video quality, the weather and the time. The HUGVid ranking was then compared with the user ranking.

We present two representative query results: *Q1* targets a tall and wide landmark while *Q2* targets at a small statue. Table 3 presents the comparison between ranking by the algorithm and ranking by the users for *Q1* and *Q2*. The first row shows the score assigned by HUGVid while the last two rows are statistics from the user ranking. The ranking between HUGVid and the users for *Q1* does not exactly match, especially for the resulting Video 1, even though Video 1 captures *Q1* from a close location. Even users disagreed on Video 1: some chose it as their favourite while others disliked it. The reason is that for a large building, some users desire to watch a panoramic view while others wish to view an up close shot with details instead. Conversely, for targets that are not so large (*e.g.*, *Q2*), HUGVid shows consistent results. The scores given by the users are almost the same as those by the algorithm. Moreover, the low standard deviation indicates that most users agree with the manner in which HUGVid ranks videos.

Approximate distance. We also evaluated HUGVid on its ability to estimate the distance between the query and the video segments in a kNN query. BM shown in Figure 9 represents the average distance calculated using the

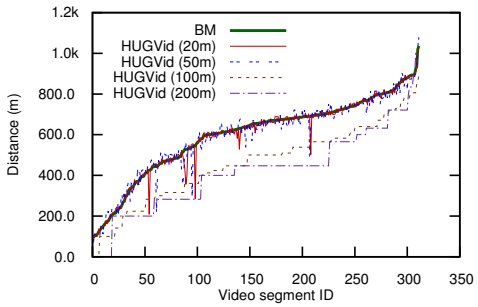


Figure 9: Comparison between the distance from BM and HUGVid with different micro-block sizes.

geo-coordinates of the query point and all the camera locations. The resulting 311 video segments are sorted by ascending distance from BM. Comparing the four approximate distances estimated by HUGVid with different micro-block sizes, the most accurate results are obtained from HUGVid with the smallest micro-block size. Since we utilized the position of a local POI to help estimating the distance, the errors are proportional to the block-size. In order to achieve accurate results, we chose to use 20 m as the default micro-block size (δ). There exist a few outliers where the distance from HUGVid significantly differs from that of BM when δ is no larger than 50 m. We manually checked those videos and found that all the segments with outliers are from the same unparsed video. In that video, the GPS raw data is extremely inaccurate, jumping from one location to another about 1 km away and then jumping back to its previous location. This situation is very rare and outside of the common GPS error range. Moreover, the reason that the outliers with different micro-block sizes appear in different segments is that different micro-block sizes lead to different video segmentation in some situations.

6.2 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 × 75 km region was used to test the performance of our algorithm with large-scale data. We generated camera trajectories using the Georeferenced Synthetic Meta-data Generator [2]. The produced synthetic meta-data exhibits equivalent characteristics to real-world data. We selected 100 randomly distributed center points within the test area and generated 5,500 moving cameras with trajectories near these center points. 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. To simulate a real-world case, we set the maximum speed of the moving cameras to 60 km/h, with an average speed of 20 km/h. We also set the maximum camera rotation speed to 30 ° per second, ensuring that the camera rotates smoothly and does not jump from one direction to another, hence emulating real user behaviour. The parameters used for generating the synthetic dataset are summarized in Table 4. The experiments were then conducted on a server with two quad core Intel® Xeon® $X5450$ 3.0 GHz CPUs and 32 GB memory running Linux 2.6.18.

Among all the query types that our approach supports, the system workload for range queries is the heaviest. Therefore, we use range queries to present the performance of HUGVid on the large-scale dataset. In this experiment, the

Parameter	Value
# of Center points	100
Speed limit	60 km/h
Average speed	20 km/h
Rotation limit	30 °/s
# of cameras	5,500
# of snapshots	1,000
# of FOVs	5,405,051
Viewable angle of FOV (α)	60 °
Visible distance of FOV (R_V)	2,000 m

Table 4: The main parameters of the large-scale, synthetic dataset.

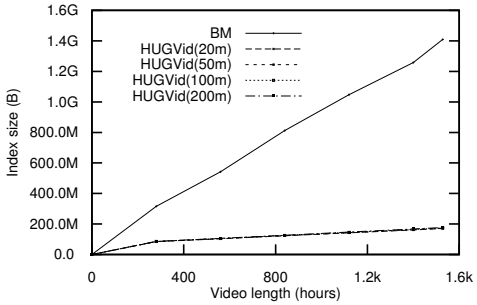


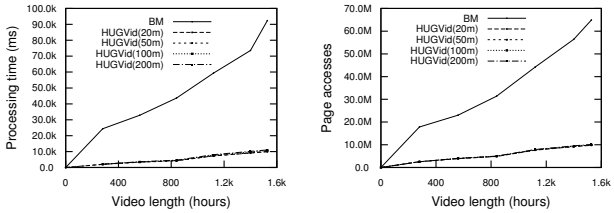
Figure 10: Comparison of the index sizes between BM and HUGVid with different micro-block sizes.

page and cache sizes are set to 4 kB, and we store one R-tree node per page. We generated 10,000 range queries of 500 m × 500 m rectangles within the 75 km × 75 km test region and counted the cumulative processing time and the overall number of page accesses for answering 10,000 queries.

We conducted experiments with different video lengths. Figure 10 shows the in-memory index size of different methods with different test sets. The index size of all the methods grows linearly, but the rate for HUGVid is much smaller than for BM. Although the video segmentation is carried out using the Monte-Carlo method with different micro-block sizes, the segmentation is still mostly related to the spatial properties of the video itself. Hence the in-memory index sizes of HUGVid with different micro-block sizes are almost the same. The main difference is in the storage of the secondary index: it occupies more disk space when the micro-block size is smaller. However, as stated in Section 6.1, one extra benefit is that it achieves more accurate results. As shown in Figure 11, HUGVid with different micro-block sizes performs faster and accesses fewer pages than BM. Even when HUGVid involves video recombination and video ranking, it still answers the queries quickly, with an execution time of only about 12% of BM. We conclude from these experiments that HUGVid performs well on this large-scale dataset. Moreover, it is beneficial to select a relatively small micro-block size while the exact value may depend on different applications.

7. CONCLUSIONS AND FUTURE WORK

We explored the challenge introduced by naturally noisy data from GPS and compass sensors which can result in inaccurate geo-descriptions of video scenes. This, in turn, may lead to undesirable query results for geo-tagged video searches. To address this issue, we proposed an uncertain data model to represent individual and sequences of field-of-views and finally constructed a light-weight approximate



(a) Processing time. (b) Page accesses.

Figure 11: Comparison of processing time and page accesses between BM and HUGVid with different micro-block sizes.

model for video segments based on sensor meta-data. With this architecture, probabilistic queries can be executed and upstream GIS tasks prioritized based on the most promising results. Experiments show that HUGVid achieves high precision and can be deployed in large-scale applications. For our future research, we plan to utilize other information in the query process, such as landmark databases.

Acknowledgments

This research has been supported in part by the Singapore National Research Foundation under its International Research Centre @ Singapore Funding Initiative and administered by the IDM Programme Office and by Award No. 2011-IJ-CX-K054 from National Institute of Justice, Office of Justice Programs, U.S. Department of Justice. The opinions, findings, and conclusions or recommendations expressed in this publication are those of the authors and do not necessarily reflect those of the Department of Justice.

8. REFERENCES

[1] S. Arslan Ay, R. Zimmermann, and S. H. Kim. Viewable Scene Modeling for Geospatial Video Search. In *SIGMM, ACM International Conference on Multimedia*, pages 309–318, 2008.

[2] S. Arslan Ay, R. Zimmermann, and S. H. Kim. Generating Synthetic Meta-data for Georeferenced Video Management. In *ACM SIGSPATIAL GIS International Conference on Advances in Geographic Information Systems*, pages 280–289, 2010.

[3] J. Boreczky and L. Wilcox. A Hidden Markov Model Framework for Video Segmentation Using Audio and Image Features. In *Acoustics, IEEE International Conference on Speech and Signal Processing*, volume 6, pages 3741–3744, 1998.

[4] Z. Cernekova, N. Nikolaidis, and I. Pitas. Temporal Video Segmentation by Graph Partitioning. In *Acoustics, IEEE International Conference on Speech and Signal Processing*, volume 2, pages 209–212, 2006.

[5] J. Chen and R. Cheng. Efficient Evaluation of Imprecise Location-Dependent Queries. In *ICDE, IEEE International Conference on Data Engineering*, pages 586–595, 2007.

[6] R. Cheng, S. Prabhakar, and D. V. Kalashnikov. Querying Imprecise Data in Moving Object Environments. In *TKDE, IEEE Transactions on Knowledge and Data Engineering*, volume 16, pages 1112–1127, 2004.

[7] R. Cheng, Y. Xia, S. Prabhakar, R. Shah, and J. S. Vitter. Efficient Indexing Methods for Probabilistic Threshold Queries over Uncertain Data. In *VLDB, International Conference on Very Large Data Bases*, pages 876–887, 2004.

[8] G. Douglas. R-Tree, Templated C++ Implementation, 2010. URL: `http://superliminal.com/sources/RTreeTemplate.zip`.

[9] X. Gao, B. Han, and H. Ji. A Shot Boundary Detection Method for News Video Based on Rough Sets and Fuzzy Clustering. In *Image Analysis and Recognition*, volume 3656, pages 231–238. Springer, 2005.

[10] C. H. Graham, N. R. Bartlett, J. L. Brown, Y. Hsia, C. C. Mueller, and L. A. Riggs. Vision and visual perception. 1965.

[11] A. Guttman. R-Trees: A Dynamic Index Structure for Spatial Searching. In *SIGMOD, ACM International Conference on Management of Data*, pages 47–57, 1984.

[12] T. H. Hwang, K. H. Choi, I. H. Joo, and J. H. Lee. MPEG-7 Metadata for Video-based GIS Applications. In *IEEE International Geoscience and Remote Sensing Symposium*, volume 6, pages 3641–3643, 2003.

[13] D. O. Kim, D. S. Hong, H. K. Kang, and K. J. Han. UR-Tree: An Efficient Index for Uncertain Data in Ubiquitous Sensor Networks. In *GPC, International Conference on Advances in Grid and Pervasive Computing*, pages 603–613, 2007.

[14] K. H. Kim, S. S. Kim, S. H. Lee, J. H. Park, and J. H. Lee. The Interactive Geographic Video. In *IGARSS, IEEE International Geoscience and Remote Sensing Symposium*, volume 1, pages 59–61, 2003.

[15] R. Li, B. Bhanu, C. V. Ravishankar, M. Kurth, and J. Ni. Uncertain Spatial Data Handling: Modeling, Indexing and Query. In *Computers & Geosciences*, volume 33, pages 42–61, 2007.

[16] X. Liu, M. Corner, and P. Shenoy. SEVA: Sensor-Enhanced Video Annotation. In *SIGMM, ACM International Conference on Multimedia*, pages 618–627, 2005.

[17] J. D. Martin, J. Krosche, and S. Boll. Dynamic GPS-position Correction for Mobile Pedestrian Navigation and Orientation. In *WPNC, Workshop on Positioning, Navigation and Communication*, pages 199–208, 2006.

[18] N. Metropolis and S. Ulam. The Monte Carlo Method. In *Journal of the American Statistical Association*, volume 44, pages 335–341, 1949.

[19] T. Navarrete and J. Blat. VideoGIS: Segmenting and Indexing Video Based on Geographic Information. In *AGILE, Conference on Geographic Information Science*, pages 1–9, 2002.

[20] N. O'Connor, T. Duffy, C. Gurrin, H. Lee, D. Sadlier, A. Smeaton, and K. Zhang. A Content-Based Retrieval System for UAV-like Video and Associated Metadata. In *Airborne Intelligence, Surveillance, Reconnaissance Systems and Applications*, 2008.

[21] H. peter Kriegel, P. Kunath, M. Pfeifle, and M. Renz. Probabilistic Similarity Join on Uncertain Data. In *DASFAA, International Conference on Database Systems for Advanced Applications*, pages 295–309, 2006.

[22] N. B. Priyantha, A. Chakraborty, and H. Balakrishnan. The Cricket Location-Support System. In *MobiCom, ACM International Conference on Mobile Computing and Networking*, pages 32–43, 2000.

[23] S. Singh, C. Mayfield, S. Prabhakar, R. Shah, and S. E. Hambrusch. Indexing Uncertain Categorical Data. In *ICDE, IEEE International Conference on Data Engineering*, pages 616–625, 2007.

[24] P. Sistla, O. Wolfson, S. Chamberlain, and S. Dao. Querying the Uncertain Position of Moving Objects. In *Temporal Databases: Research and Practice*, pages 310–337. Springer, 1998.

[25] Y. Tao, X. Xiao, and R. Cheng. Range Search on Multidimensional Uncertain Data. In *ACM Transactions on Database Systems*, volume 32, pages 15–68, 2007.

[26] Y. Zhang, X. Lin, W. Zhang, J. Wang, and Q. Lin. Effectively Indexing the Uncertain Space. In *TKDE, IEEE Transactions on Knowledge and Data Engineering.*, volume 22, pages 1247–1261, 2010.

[27] Z. Zhu, E. Riseman, A. Hanson, and H. Schultz. An Efficient Method for Geo-referenced Video Mosaicing for Environmental Monitoring. In *Machine Vision and Applications*, volume 16, pages 203–216. Springer, 2005.