J. Vis. Commun. Image R. 21 (2010) 773–786

Contents lists available at ScienceDirect

J. Vis. Commun. Image R.

journal homepage: www.elsevier.com/locate/jvci

Design and implementation of geo-tagged video search framework

Seon Ho Kim [a,*], Sakire Arslan Ay [a], Roger Zimmermann [b]

[a] University of Southern California, Los Angeles, CA 90089, USA
[b] National University of Singapore, Singapore 117417, Singapore

ARTICLE INFO

Article history:
Available online 2 August 2010

Keywords:
Video search
Geo-tagging
Meta-data
Scalability
Mobile video
Video ranking
GPS
Sensor

ABSTRACT

User generated video content is experiencing significant growth which is expected to continue and further accelerate. As an example, users are currently uploading 20 h of video per minute to YouTube. Making such video archives effectively searchable is one of the most critical challenges of multimedia management. Current search techniques that utilize signal-level content extraction from video struggle to scale.

Here we present a framework based on the complementary idea of acquiring sensor streams automatically in conjunction with video content. Of special interest are geographic properties of mobile videos. The meta-data from sensors can be used to model the coverage area of scenes as spatial objects such that videos can effectively, and on a large scale, be organized, indexed and searched based on their field-of-views. We present an overall framework that is augmented with our design and implementation ideas to illustrate the feasibility of this concept of managing geo-tagged video.

1. Introduction

Camera sensors have become a ubiquitous feature in our environment and more and more video clips are being collected and stored for many purposes such as surveillance, monitoring, reporting, or entertainment. Because of the affordability of video cameras the general public is now generating and sharing their own videos, which are attracting significant interest from users and have resulted in an extensive *user generated online video market* catered to by such sites as YouTube. As of 2010, more than half (55%) of all the video content consumed online in the US is expected to be user generated, representing 44 billion video streams [1]. Cisco predicts that, by 2014, global online video will approach 57% of all consumer Internet traffic [12]. Companies are developing various business models in this emerging market, with one of the more obvious ones being advertising. In 2008, Forrester Research and eMarketer reported that the global online video advertising market will reach more than US$7.2 billion by 2012 [35].

Many of the end-user cameras are mobile, such as the ones embedded in smartphones. The collected video clips contain a tremendous amount of visual and contextual information that makes them unlike any other media type. However, currently it is still very challenging to index and search video data at the high semantic level preferred by humans. Effective video search is becoming a critical problem in the user generated video market. The scope of this issue is illustrated by the fact that video searches on YouTube accounted for 28% of all Google search queries in the US in December of 2009 and that 23% of YouTube's total visits for December originated from Google search [4]. Better video search has the potential to significantly improve the quality and usability of many services and applications that rely on large repositories of video clips.

A significant body of research exists – going back as early as the 1970s – on techniques that extract features based on the visual signals of a video. While progress has been very significant in this area of content-based video retrieval, achieving high accuracy with these approaches is often limited to specific domains (e.g., sports, news), and applying them to large scale video repositories creates significant scalability problems [36,26]. As an alternative, text annotations of video can be used for search, but high-level concepts must often be added manually and hence their use is cumbersome for large video collections. Furthermore, text tags can be ambiguous and subjective.

Recent technological trends have opened another avenue to associate more contextual information with videos: the automatic collection of sensor meta-data. A variety of sensors are now cost-effectively available and their data can be recorded together with a video stream. For example, current smartphones embed GPS, compass, and accelerometer sensors into a small, portable and energy-efficient package. The meta-data generated by such sensors represents a rich source of information that can be mined for relevant search results. A significant benefit is that sensor meta-data can be added automatically and represents objective information (e.g., the position).

* Corresponding author.
E-mail addresses: seonkim@usc.edu (S.H. Kim), arslan@usc.edu (S. Arslan Ay), rogerz@comp.nus.edu.sg (R. Zimmermann).

1047-3203/$ - see front matter © 2010 Elsevier Inc. All rights reserved.
doi:10.1016/j.jvcir.2010.07.004

Some types of video data are naturally tied to geographical locations. For example, video data from traffic monitoring may not have much meaning without its associated location information. Thus, in such applications, one needs a specific location to retrieve the traffic video at that point or in that region. Hence, combining video data with its location information can provide an effective way to index and search videos, especially when a database handles an extensive amount of video data.

Researchers have only recently started to investigate and understand the implications of the trends brought about by technological advances in sensor-rich cameras. We believe that there is tremendous potential that has yet to be explored and consequently that there is a need for an overall framework to facilitate the design and implementation of relevant applications. Here we propose such a framework, based in part on our experiences with preliminary work in geo-tagged video management. Note that this manuscript provides only example solutions for some of the different framework components and is intended to stimulate further work and discussions in these areas.

Our framework is organized around the following key issues that we identified for geo-tagged video management: (1) data acquisition – sensor inputs will be collected while videos are being recorded at mobile devices. The collected data consists of the meta-data for future storage and retrieval of videos; (2) data management (search engine) – the recorded videos are represented as spatial objects using the collected sensor data and videos are indexed and searched mainly based on their geographical properties; and (3) search result presentation – videos are ranked and presented based on their relevance with the query for a fast and effective browsing of results by humans.

In this study, we describe a framework for the handling of geo-tagged video that focuses on the above outlined issues and challenges. The goal is to enhance video search, especially with very large collections of videos, in order to significantly improve the quality of video service applications. The rest of this paper is organized as follows. Related work is surveyed in Section 2. Section 3 presents the design of our framework. Geo-tagged video data collection and search is explained in Sections 4 and 5, respectively, as our own instantiation of an implementation of the framework. Section 6 shows the experimental results of our implementation. Finally, we conclude with Section 7.

2. Related work

There has been significant research on organizing and browsing personal photos according to location and time. Toyama et al. [33] introduced a meta-data powered image search and built a database, also known as World Wide Media eXchange (WWMX), which indexes photographs using location coordinates (latitude/longitude) and time. A number of additional techniques in this direction have been proposed [24,27]. There are also several commercial web sites [2,3,5] that allow the upload and navigation of georeferenced photos. All these techniques use only the camera geo-coordinates as the reference location in describing images. We instead rely on the field-of-view of the camera to describe the scene. More related to our work, Ephstein et al. [13] proposed to relate images with their view frustum (viewable scene) and used a scene-centric ranking to generate a hierarchical organization of images. Several additional methods are proposed for organizing [28] and browsing [32] images based on camera location, direction and additional meta-data. Although these research approaches are similar to ours in using the camera field-of-view to describe the viewable scene, their main contribution is on image browsing and grouping of similar images together. There exist only a few systems that associate videos with their corresponding geo-location. Liu et al. [21] presented a sensor-enhanced video annotation system (referred to as SEVA) which enables searching videos for the appearance of particular objects. SEVA serves as a good example to show how a sensor rich, controlled environment can support interesting applications. However, it does not propose a broadly applicable approach to geo-spatially annotate videos for effective video search. These techniques propose to collect information about the location of the monitored objects only, therefore they do not attempt to describe and record the scene that the video camera captures. In our prior work [8] we have proposed the use of videos' geographical properties (such as camera location and camera heading) to enable effective search of large video collections. We introduced a *viewable scene* model to describe the video content.

Liu et al. [21] propose to record only the identities and locations of the objects within the viewable scene along with visual images. Hwang et al. [15] and Kim et al. [18] provide a mapping between the objects that appear in video and their geographic locations on a map. However, their work neglects to provide any details on how to use the camera location and direction to build links between video frames and real-world objects. Neither of these techniques address the search issues for effective search of large video collections nor do they provide any solutions for analyzing the relevance of search results. Ueda et al. [34] identify the *importance* of the geographic objects based on closeness and orientation of the object with respect to camera. Their main objective is to identify the important objects that appear in the videos. However, we propose to find the most relevant video segments that show a given query object or region. Their work is lacking the details on how they search the video meta-data for the objects that appear in a video. To the best of our knowledge, our framework is the first to provide an effective model to describe the geographic coverage of the video content and to propose an overall structure for managing the geospatial video meta-data for effective and efficient search of large video collections. We further propose video ranking techniques by based on the *viewable scene* cues. Our ranking techniques do not target any specific application domain and therefore can be applied to a variety of applications.

There already exist GPS-enabled digital cameras which can attach the location information while still images and/or videos are being captured (e.g., Ricoh SE-3 camera, Sony HDR-XR520V camcorder). Recently, mobile phones equipped with video camera, GPS and digital compass have been introduced (e.g., Apple iPhone 3GS). As the sensor-equipped video capture devices become popular, more location and direction tagged videos will be produced.

Beyond geographic meta-data, there exist several other automatic video meta-data creation methods using aural, visual and textual processing techniques. Christel [11] provide a survey of automated meta-data creation systems for multimedia systems. There has been considerable work in searching and ranking videos using content-based features. A review of state-of-the-art solutions can be found in the literature [20,31]. The TREC Video Retrieval Evaluation (TRECVID) [29] benchmarking activity has been promoting progress in content-based retrieval of digital video since 2001. Each year, various feature detection methods from dozens of research groups are tested on hundreds of hours of video [30]. Our technique and content-based retrieval methods are orthogonal to each other and could be combined to create powerful solutions, depending on application needs. For example, Zheng et al. [37] propose an earth-scale landmark recognition engine that leverages the multimedia data on the web (i.e., geo-tagged pictures, travel guide articles) and object recognition and clustering techniques. Luo et al. [23] propose an event recognition technique utilizing the extracted visual features from both ground and satellite images. Similarly, the search capabilities of our framework can be advanced by leveraging visual features in addition to geographic properties.

3. Framework

This section proposes a general framework for geo-tagged video search applications as shown in Fig. 1. We emphasize the mobility of cameras in the framework because of the ubiquity of mobile devices and the prominent importance of geographic properties of moving cameras. We envision that more and more user generated videos are produced from mobile devices such as cellular phones. To address the issues of geo-tagged video search outlined in Section 1, our framework consists of three main parts: the data collection with mobile devices, the search engine to store, index, search and retrieve both the meta-data and video contents, and the user interface to provide web-based video search services. These main parts are communicating through the Internet and/or cellular network.

3.1. Data collecting device

At the mobile device level, the main objective is to capture the sensor inputs and to fuse them with the video for future storage and retrieval of videos. In the framework, a mobile device can be any camera equipped with various sensors and a communication unit. A good example is a smartphone such as Apple's iPhone 3GS that includes GPS, digital compass, accelerometer, 5 mega pixel camera, WiFi/Broadband data connection, and programming capability. The following issues need to be considered at the device level.

First, the *Data Collection Module* in Fig. 1 captures videos with sensor inputs through various sensors including camera, GPS receiver, compass, accelerometer, etc. Sensor signals can be affected by noises so they might be checked and refined in the *Sensor Signal Processing Module*. For example, accelerometer input can be filtered for a clearer signal. Sensor measurement errors can be detected and missing sample values can be estimated here. Then, the sampled sensor data should be synchronized and tagged in accordance with the recorded video frames (*Format Module*). This automatic synchronized annotation forms the basis of the proposed framework. Assuming multiple sensors with different sampling rates and precisions (e.g., for each second 30 frames of video, 1 GPS location coordinate, and 40 direction vectors), values might be manipulated using numerical methods such as interpolating, averaging, etc. The sensor meta-data can be sampled either periodically, or aperiodically by applying adaptive methods. An adaptive method can be more efficient and desirable in a large scale application since it can minimize the amount of the captured data and can support a more scalable system. The synchronization among sensor inputs and video frames should be designed to maximize processing accuracy and to minimize the amount of meta-data.

Another challenge is how the automatic annotation can handle a variety of video technologies (and cameras) and sensors. Without any standard tagging method, the compatibility among various meta-data would be a critical problem for general acceptance in diverse applications. This issue can be more pronounced when video search is extended to user generated videos on public web sites. Therefore, it is desirable to represent the collected meta-data using a standard format, regardless of the number, type, and precision of sensors. The next question is whether the meta-data are either (1) embedded into the video files or (2) handled separately. The embedding granularity can be at the frame, segment, scene or clip level of the video. Embedding requires a standard embedding method based on a specific video technology. The problem is that there are so many different video coding techniques. Separating video and meta-data works independently from the video coding techniques, however it presents a verification problem between a video file and its meta-data.

The next issue is an efficient interaction between the mobile devices and the server. Multimedia data is generally large and may require intense processing (such as compression) and significant bandwidth for its transmission. Therefore, an effective interaction and efficient data transmission among the mobile clients and the

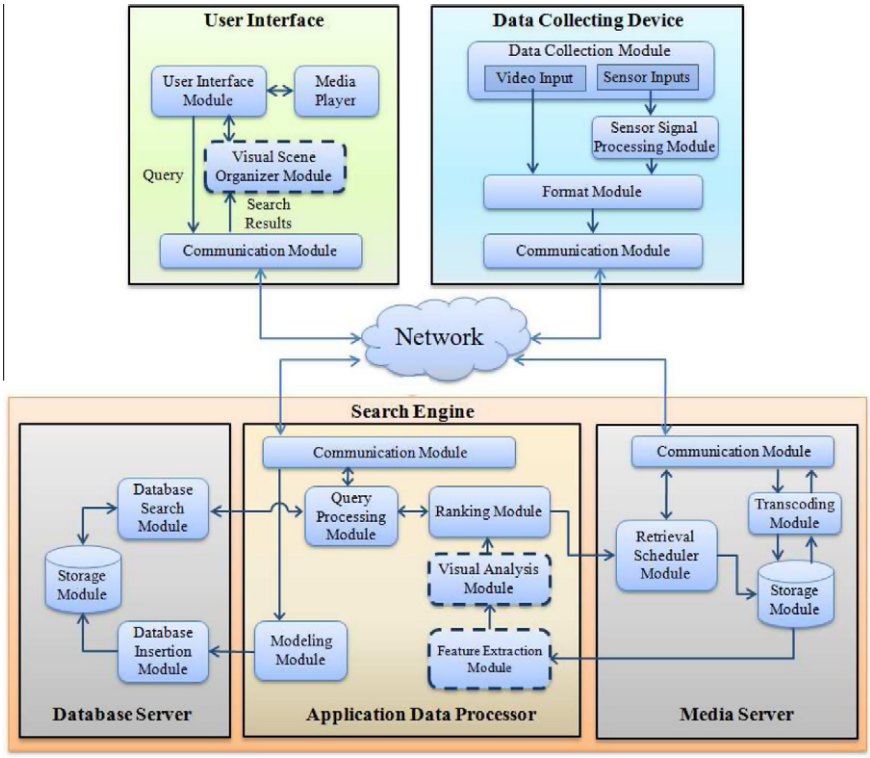


Fig. 1. Framework structure and modules.

server is of much importance. Currently, user generated data is typically sent immediately to a server in their entireties. This approach works well for small size image collections. However, when we consider a large data video transfer from a mobile device which has expensive communication cost or limited communication capability, this approach is not cost-efficient. Furthermore, not all collected videos are considered relevant or with high priority, and thus, are not requested immediately. In general, the collection and consumption of data exist on independent schedules, determined primarily by the convenience for each user, which results in a time gap between the data collection time and the data request time. Considering the bandwidth and power consumption costs of transmitting large amounts of data such as video content, the immediate transmission of potentially irrelevant data is an inefficient use of resources.

There are ways to overcome the drawbacks of the immediate data transmission scheme. For example, we can separate the small amount of text-based geospatial meta-data from the large binary-based video content. This small amount of meta-data would be transmitted to a server in real-time, while the video content would remain on the recording device, creating an extensive, resource efficient catalog of video content searchable by geographical properties established by meta-data associated with each video. Should a particular video be requested, only then it will be transmitted from the camera to the server in an on-demand manner. Otherwise, the delivery to a server can be delayed until a more cost-efficient way is available (e.g., through wired network). Note that this separation of the meta-data from video contents can incur extra processing for the synchronization and verification of the meta-data. The *Communication Module* defines and controls the data transmission between the mobile device and the server based on a predefined protocol. This module needs to provide versatile ways to accommodate diverse applications and service models.

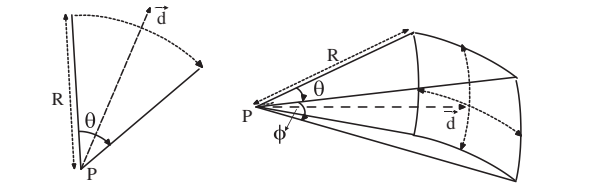
3.2. Search engine

The goal of the search engine is to retrieve more meaningful results for end-users in a highly efficient way that scales to large video archives. In our framework, we define the search engine as the collection of all components that store, index, search, and retrieve both the meta-data and video contents. The search engine consists of three components as shown in Fig. 1: (1) the database server (DB) which manages the spatio-temporal meta-data and performs video search based on them, (2) the media server (MS) which stores and retrieves videos, and (3) the application data processor (AP) which manages the details of application-dependent features such as a spatial model of field-of-view (FOV), query types, etc. The framework does not assume any specific database management system or media server. However, AP can be customized for individual online video applications. Note that, in the most general and largest scale applications, multiple database servers and media servers may be distributed across a wide area and they may collaborate with each other.

3.2.1. Application data processor

To utilize the captured geographic properties of videos for searching, the framework represents the coverage area of video scenes (*FOV*) as spatial objects in a database, i.e., it models the coverage area using the collected meta-data (*Modeling Module*). This modeling effectively converts the problem of video search into the problem of spatial object selection in a database. The effectiveness of such a model depends on the availability of sensor data. For example, in an application with only camera location data from GPS, the potential coverage area of video frames can be represented as a circle centered at the camera location (*CircleScene* in Fig. 2). In another application with extra camera direction data, the coverage area can be more accurately refined like a pie slice shown in Fig. 2 (more details in Fig. 3). Thus, videos represented by the pie model can be searched more effectively. Modeling is essential for indexing and searching of video contents in a database because the query functionality and performance are greatly impacted by the spatial modeling of videos.

The *Query Processing Module* can implement a set of user-defined query types for an application. Note that, for the implementation of the new query types, new indexing techniques or database query functionalities might need to be introduced. Moreover, the evaluation of new query types should be fast enough to be practical, especially for large scale video search. There has been little research on these issues. In Section 5.1, our implementation of new query types will be described in detail.



P : camera location (<longitude,latitude>)
θ : viewable angle
d : camera direction vector
R: visible distance

(a)

P : camera location (<longitude,latitude,altitude>)
θ,ϕ: horizontal and vertical viewable angles
d : camera direction vector (in 3D)
R : visible distance

(b)

Fig. 3. Illustration of camera field-of-view (FOV) (a) in 2D (b) in 3D.

CircleScene Coverage

FOV Coverage

Fig. 2. Comparison of two coverage models.

In a large collection of videos, a search usually results in multiple video matches. The challenge is that human visual verification of the video results may take a significant amount of time due to the overall length of videos. To enhance the effectiveness of the result presentation, an approach is to quantify the relevance of resulting videos with respect to the given query and to present the results based on their relevance ranking. The difficulty lies in the fact that the human appreciation process of relevance is very subjective and so it is challenging to be quantified. In our framework, the *Ranking Module* harnesses objective measurements to quantify the relevance between a query and videos in two different ways: (1) spatial relevance: overlapping coverage area between query range and a video, and (2) temporal relevance: overlapping covered time.

The main objective of the framework is to search videos using their spatio-temporal meta-data. However, it is also well known that video search can further be improved by leveraging the features extracted from the visual video content. As a complementary approach to enhance the searchability, the framework can be combined with optional video processing modules. For example, the *Visual Analysis Module* can synergistically enhance the accuracy of the ranking process by accommodating visual features extracted by the *Feature Extraction Module*. To calculate the relevance based on the visual content, existing video ranking techniques can be adopted [22]. However, considering that most techniques are proven to work effectively on specific domains, it remains uncertain how well these techniques can perform with unconstrained video datasets in a general video search framework such as the one presented. Some recent work [17] proposed to analyze the visual similarities among resulting images to choose the representative images that answer the search keywords well. The well-connected images that are found to be similar to a majority of the resulting images are returned as the most relevant. Such an approach can be applied in our framework for content-based ranking of videos.

Since our framework targets general video search while most content-based search techniques are domain-specific, any discussion of combining our approach and a content-based technique may not be meaningful without a specific application. Consequently, we do not provide further details here.

3.2.2. Database server

The database server stores the coverage areas of video scenes as spatial objects in a conventional database management system such as MySQL. When a user query is received from AP and translated into a spatio-temporal selective query according to the video scene model, the database is searched using conventional query processing techniques.

The database server unit can consist of the following modules:

Database Insertion Module: Inserts the spatio-temporal video scene descriptions into the database.
Database Search Module: Searches the database based on the query specifications received from the application data processor unit.
Storage Module: The video scene meta-data (based on the model) are stored using appropriate data structures and indexes.

3.2.3. Media server

The role of media server is to store actual video contents and to provide streaming service to users. In general, the media server obtains a list of video segments as query results and transmits them to the user interface in a predefined way. Different user interfaces can present the search results in different ways so the media server corresponds to the requests of the user interface.

In a large collection of videos with heterogenous coding techniques, a video might need to be transcoded by the *Transcoding Module* when it arrives at the server. For example, a user can collect videos in any format but the application might require certain predefined formats for service. Similarly, when users request the display of the query results, videos can be transcoded to accommodate the different supported video formats between the server and user devices. The *Storage Module* stores videos based on the underlying storage system and the media server technology. The query results from the application data processor are analyzed by the *Retrieval Scheduler Module* to provide the most efficient way to retrieve the requested data. One critical functionality of the media server in our framework is the ability to randomly access any portion of stored videos in a fast way. Since the amount of data in an entire video clip can be very large and the user might be interested in watching only a portion of the video where the query overlaps, random access is very important for humans to verify the results.

3.3. User Interface

The role of the *User Interface* unit is to provide users with methods to communicate with the search engine from both wired computers and mobile devices. Then, depending on the type of devices, the user interface can be designed in different ways. Our framework assumes a web-based user interface to communicate with the search engine. Depending on the computing power of the user's machine and the availability of other supporting software (e.g., Google Maps, media player, web browser), the features of the video search applications can be significantly affected.

Users can search videos in multiple ways (*User Interface Module*). One intuitive method is submitting a map-based query when users are familiar with the interested area. Drawing a query region directly on any kind of map (see our implementation examples in Figs. 11 and 12) might provide the most human-friendly and effective interface paradigm. Alternatively, a text-based query can also be effective when users are searching for a known place or object. For example, the user interface can maintain a local database of the mapping between places and their geo-coordinates. Then, the textual query can be converted into a spatial query with exact longitude and latitude input. External geo-coding services can also provide this translation.

The user interface module receives the ranked query results from the search engine. In addition to the ranking method, the presentation style or format of the results also greatly affects the effectiveness of the presentation. Thus, human-friendly presentation methods should be considered such as using key frames, thumbnail images, any textual descriptions, etc. The optional *Visual Scene Organizer Module* can organize video search results based on the spatial and visual scene similarity. Then, the user interface module may implement more effective video browsing tools based on the scene similarity. A map-based user interface for both query input and video output can also be an effective tool by coordinating the vector map and actual video display. Note that relevance ranking and presentation style are not just technical issues, but may require an intensive user study.

The *Media Player* software plays out the query results. One important aspect of video presentation is the capability to display only relevant segments of videos at the user side avoiding unnecessary transmission of video data. Thus, the media player at the user side and the streaming media server are expected to closely collaborate.

4. Data collection with viewable scene model

Our implementation focuses on the very essential geographic properties of the video contents captured from a video camera, a

3D digital compass, and a GPS receiver. We assume that the optical properties of the camera are known. The digital compass mounted on the camera heads straight forward as the camera lens. The compass periodically reports the direction in which the camera is pointing with the current heading angle (with respect to North) and the current pitch and roll values. The GPS receiver, also mounted on the camera, reports the current latitude, longitude, and altitude of the camera. Video can be captured with various camera models. Our custom-written recording software receives direction and location updates from the GPS and compass devices as soon as new values are available and records the updates along with the current computer time and coordinated universal (UTC) time.

A camera positioned at a given point p in geo-space captures a scene whose covered area is referred to as camera field-of-view (FOV, also called a viewable scene), see Fig. 3. The meta-data related to the geographic properties of a camera and its captured scenes are as follows: (1) the camera position p is the latitude, longitude coordinates read from GPS, (2) the camera direction α is obtained based on the orientation angle ($0° \leqslant \alpha < 360°$) provided by a digital compass, (3) the maximum visible distance from p is R at which objects in the image can be recognized by observers [8] – since no camera can capture meaningful images at an indefinite distance, R is bounded by M which is the maximum distance set by an application, and (4) the camera view angle θ describes the angular extent of the scene imaged by the camera. The angle θ is calculated based on the camera and lens properties for the current zoom level [14]. The above geo-properties are captured from a sensor-equipped camera while video is recorded. Note that some commercial cameras are already equipped with those sensors or expected to be equipped in the very near future.

Based on the collected meta-data, we model the viewable area in 2D space, which is represented as a circular sector as shown in Fig. 3(a). For a 3D representation shown in Fig. 3(b), we would need the altitude of the camera location point and the pitch and roll values to describe the camera heading on the zx and zy planes (i.e., whether the camera is directed upwards or downwards). We believe that the extension to 3D is straightforward, especially since we already acquire the altitude level from the GPS and the pitch and roll values from the compass. Thus, for a focused discussion, we will represent the FOV in 2D space in this paper.

One important point in data collection is the difference in the data sampling frequencies. The GPS location updates are available every 1 s whereas compass can produce 40 direction updates per second. And for a 30 fps video stream there will be 30 frame timecodes for every 1 s video. An intuitive way is to create the combined dataset as the sensor data is received from the devices and use a common timestamp for the combined tuple. Such a tuple will include the last received updates for the location and direction values. Because of the heterogeneity in data frequencies, it is possible to match data items which are not temporally closest. A better way is to create separate datasets for GPS updates, compass readings and frame timecodes, and later combine the data items from each data set that has the closest time match. Since all sensor values will be refreshed at most every second, intuitively, the data frequency for the combined dataset will be a second.

We used a periodic data collection in our implementation, i.e., an update per one second. Thus, in the implementation, a n second long video is represented with n FOVs each representing the geographic properties of one second long video frames. Table 1 shows examples of the collected geo-tagged meta-data for a 5 s long video, where each row (i.e., tuple) corresponds to an FOV. In Table 1, *Timestamp* is the computer time when the meta-data is recorded and *Timecode* is the corresponding frame timecode in the actual video.

5. Geo-tagged video search

5.1. Considerations

Video searching should be able to fully take advantage of the collected meta-data for various requests by applications. Beyond the importance of the geographic information where a video is taken, there are other obvious advantages in exploiting the spatial properties of video because the operation of a camera is fundamentally related to geometry. When a user wants to find images of an object captured from a certain viewpoint and from a certain distance, these semantics can be interpreted as geometric relations between the camera and the object, such as the Euclidean distance between them and the directional vector from the camera to the object. Thus, more meaningful and recognizable results can be achieved by using spatial queries on geo-tagged videos.

Search types exploiting the geographic properties of the video contents may include not only conventional point and range queries (i.e., overlap between the covered area of video and the query range), but also new types of video specific queries. For example, one might want to retrieve only frames where a certain small object at a specific location appears within a video scene, but with a given minimum size for better visual perception. Usually, when the camera is closer to the query object, the object appears larger in the frame. Thus, we can devise a new search type with a range restriction for the distance of the camera location from the query point; we term this a *distance query*. Similarly, the camera view direction can be an important factor for the image perception of an observer. Consider the case where a video search application would like to exploit the collected camera directions for querying, representing a *directional query*.

The collected meta-data are stored in a database so that videos can be searched based on the alpha-numeric meta-data. Specifically, FOVs are stored as pie-shaped spatial objects and searched only by their spatial and temporal properties. Videos are separately stored in a media server. Our implementation utilizes an existing database management system and conventional query techniques because (1) the development of new indexing or query optimization is beyond the scope of this paper and (2) we wanted to demonstrate that the framework could be implemented without extra effort of developing new database features. We focused on how the proposed FOV model can be effectively and efficiently used in a large scale video search on a conventional database such as MySQL which supports spatial constructs (i.e., data types, indices).

When a large collection of videos is stored in a database, the cost of processing spatial queries may be significant because of

Table 1
Example georeferenced meta-data tuples ($\theta = 60°$).

FOV	*Vid*	p ⟨*lat.* − *lon.*⟩	$\alpha(°)$	$R(m)$	*Timestamp*	*Timecode*
522	22	46.741548–116.998496	257.4	259	2008/03/30 19:22:13.37	0:53:46:24
523	22	46.741548–116.998498	6.2	259	2008/03/30 19:22:14.84	0:53:47:24
524	22	46.741547–116.998490	4.3	259	2008/03/30 19:22:15.37	0:53:48:24
525	22	46.741547–116.998488	359.5	259	2008/03/30 19:22:16.37	0:53:49:24
526	22	46.741547–116.998485	3.2	259	2008/03/30 19:22:17.37	0:53:50:24

the computational complexity of the operations involved, for example, determining the overlap between a circular sector shaped FOV and a polygon-shaped query region. Therefore, such queries are typically executed in two steps: a *filter* step followed by a *refinement* step [25,10]. The idea behind the filter step is to approximate the complex spatial shapes with simpler outlines (e.g., a minimum bounding rectangle, MBR [9]) so that a large number of unrelated objects can be dismissed very quickly based on their simplified shapes at the earlier stage of searching. The resulting candidate set from the filter step is then further processed during the refinement step to determine the exact results based on the exact geometric shapes. The rationale of the two step process is that the filter step is computationally far cheaper than the refinement step due to the simple approximations. Overall, the cost of spatial queries is determined by the efficiency of the filter step (many objects, but simple shapes) and the complexity of the refinement step (few objects with complex shapes).

Additionally, in video search applications, the refinement step can be very expensive due to the nature of the processing. Depending on the application, various computer vision and content-based ranking techniques can be applied before presenting the search results. For example, some specific shapes or colors of objects might be analyzed for more accurate results. Such extra processing is in general performed frame by frame, and it may significantly increase the time and execution cost of the refinement step. It is thus critical to minimize the amount of refinement processing for large scale video searches. This, in turn, motivates the use of effective and efficient filtering algorithms which minimize the number of frames that need to be considered in the refinement step.

5.2. Implementation of filter step

In conventional spatial data processing, MBR approximations are very effective for the filter step. However, with a bounding rectangle, some key properties that are useful in video search applications may be lost. For example, MBRs retain no notion of directionality which is a critical factor in searching relevant images. Thus, we need to introduce a new approximation that can provide similar efficiency and low processing cost as MBR-based methods, but can additionally provide a better support for the type of searches that a video database may encounter. We devised a novel vector-based approximation of FOVs for the filter step. The contents of a video are represented by a series of FOVs. In the vector model, the spatial property of an FOV is represented by the camera position p and the center vector $\mathbf{V}$ (Fig. 4). The magnitude of $\mathbf{V}$ is the viewable distance from p, i.e., R and the direction of $\mathbf{V}$ is α. When we project the FOV onto the x- and y-axes of the 2D coordinate system, a point p is divided into p_x and p_y, and $\mathbf{V}$ is divided into V_X and V_Y along the x- and y-axes, respectively. Then, an FOV denoted by a point and vector can be represented by a quadruple $\langle p_x, p_y, V_X, V_Y\rangle$ which can be interpreted as a point in four-dimensional space.

In conjunction with the vector estimation, we introduced a space transformation of the spatial meta-data to provide new query functionalities. In mathematics, space transformation is an approach to simplify the study of multidimensional problems by reducing them to lower dimensions or by converting them into some other multidimensional space. Using a space transformation, an FOV $\langle p_x, p_y, V_X, V_Y\rangle$ can be divided and represented in two 2D subspaces, i.e., $p_x - V_X$ and $p_y - V_Y$. Then, an FOV can be represented as two points, each in its own 2D space. For example, Fig. 4 shows the mapping between an FOV represented by $p1$ and **V1** in geo-space and two points in two transformed spaces without loss of information. To define the vector direction, let any vector heading towards the right (East in the northern hemisphere) on the x-axis have a positive V_X value, and a negative V_X value for the other direction (West). Similarly, any vector heading up (North) on y-axis has a positive V_Y value, and a negative V_Y value for the other direction (South). Using the proposed model, any single FOV can be represented as a point in a $p - V$ space. As a result, the problem of searching for FOV *areas in the original space* can be converted to the problem of finding FOV *points in the transformed subspace*. Thus, FOVs can be indexed and searched in a simpler way while keeping the directionality data.

5.3. Implementation of query

The following subsections briefly describe how the filter step can be effectively performed by using the vector model for four spatial query types. More query types and the technical details of the filtering can be found in [19].

5.3.1. Point query (PQ)

The assumed query is, "for a given query point $q\ \langle x, y\rangle$ in 2D geo-space, find all video frames that overlap with q". The filter step can be performed in $p - V$ space by identifying all possible points of FOVs that have a potential to overlap with the query point.

Recall that the maximum magnitude of any vector is limited to M, and hence any vector outside of a circle centered at the query point q with a radius M cannot reach q in geo-space. This means that any FOV whose camera location is farther than its maximum viewable distance M from the query point cannot contain the image of the query point; see Fig. 5 for an illustration. Because a query point is not a vector, it is mapped only to the p-axis. First, let us consider only the x components of all vectors. In $p_x - V_X$ space, the possible vectors that can cross (or touch) q_x should be in the range $[q_x - M, q_x + M]$. That is, any vector at p_x is first filtered out if $|p_x - q_x| > M$. Next, even though a vector is within the circle, it cannot reach q_x if its magnitude is too small. Thus, $|p_x - q_x| \leqslant |V_X|$ must be satisfied for V_X to reach q_x. At the same time the vector direction should be towards q_x. For example, when $p_x > q_x$, any vector with a positive V_X value cannot meet q_x. Hence, in $p - V$ spaces as shown in Fig. 5, all points (i.e., all vectors) outside of the shaded isosceles right triangle areas will be excluded in the filter step. For example, vector V_1 in geo-space is represented as a point v_1 in $p - V$ space. Now consider all vectors starting from a point on the circumference of the circle towards the center with the maximum magnitude M. All such vectors moving from V_1 to V_4 in a

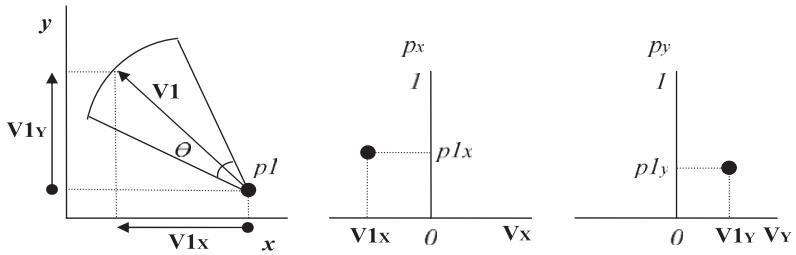


Fig. 4. FOV representation in different spaces.

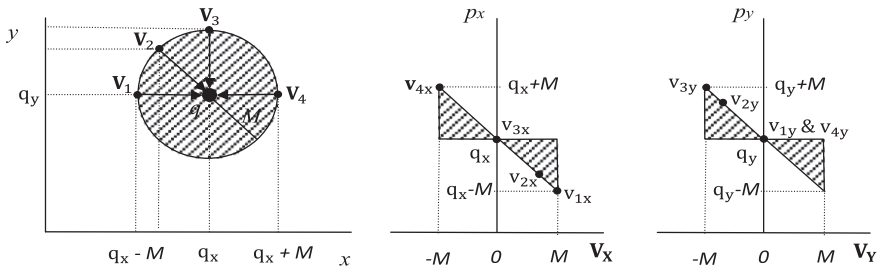


Fig. 5. Illustration of filter step in point query processing.

clockwise direction map to the diagonal line starting from v_1 to v_4 in $p - V$ space. The same can be observed for the y components of vectors, i.e., the same shape appears in $p_y - V_Y$ space. The resulting vectors from the filter step should be included in the shaded areas of both $p_x - V_X$ and $p_y - V_Y$ space. Formally, a vector at p that satisfies the conditions in column A of Table 2 can be selected in the filter step.

5.3.2. Point query with bounded distance r

Unlike a general spatial query, video search may enforce application specific search parameters. For example, one might want to retrieve only frames where a certain small object at a specific location appears within a video scene, but with a given minimum size for better visual perception. Usually, when the camera is closer to the query object, the object appears larger in the frame. Thus, we can devise a search with a range restriction for the distance of the camera locations from the query point such as "for a given query point $q\ \langle x, y\rangle$ in 2D geo-space, find all video frames that overlap with q and that were taken within the distance r from q". Because of the distance requirement r, the position of the camera in an FOV cannot be located outside of the circle centered at q with radius r, where $r < M$. Thus, the search space can be reduced so that the resulting vectors should satisfy the conditions in column B of Table 2.

5.3.3. Directional point query

The camera view direction can be an important factor for the image perception by an observer. Consider the case where a video search application would like to exploit the collected camera directions for querying. An example search is, "for a given query point $q\ \langle x, y\rangle$ in geo-space, find all video frames taken with the camera pointing in the Northwest direction and overlapping with q". The view direction β can be defined as a line of sight from the camera to the query point (i.e., an object or place pictured in the frame). The line of sight can be defined using an angle at the camera location similar to the camera direction α. Note that the camera orientation is always pointing to the center of an FOV scene while the view direction can point to any locations or objects within the scene. An important observation is that all FOVs that cover the query point have their starting points along the same line of sight in order to point towards the requested direction. Thus, the filter step needs to narrow the search to the vectors that satisfy the following conditions: (1) their starting points are on the line of sight, (2) their vector directions are heading towards q, and (3) their vector magnitudes are long enough to reach q.

For a given view direction angle β, we can calculate the maximum possible displacement of a vector starting point from the query point. Because the largest magnitude of any vector is M, the maximum displacement between the query point and the starting point of any possible overlapping vector is $-M\sin\beta$ on the x-axis and $-M\cos\beta$ on the y-axis (note that the sign is naturally decided by β, e.g., $\sin 315^\circ = -0.71$ and $\cos 315^\circ = 0.71$ where the Northwest direction is equivalent to 315°, so $\beta = 315$). As shown in Fig. 6, any vector starting at a point greater than $q_x + (-M\sin\beta)$ on the x-axis or less than $q_y + (-M\cos\beta)$ on the y-axis cannot touch or cross the query point with the given angle β. Thus, the search area for such vectors can be reduced as illustrated in Fig. 6. To meet the view direction request (say, 315° line of sight), no vector with a positive V_X value can reach q. Therefore, in the filter step the entire search space (i.e., the triangle shape) on the positive V_X side is excluded in the $p_x - V_X$ space. Similarly, no vector with a negative V_Y value can reach q, so the entire search space (the triangle shape) on the negative V_Y side is excluded in the $p_x - V_Y$ space. Again, the resulting vectors should satisfy the conditions in column C of Table 2.

5.3.4. Rectangular range query

The assumed query is, "for a given rectangular query range in geo-space, find all the video frames that overlap with this region". Suppose that the rectangular query region q is a collection of points. When we apply the same space transformation, all points in the query region can be represented as a line interval on the p_x and p_y-axes. Then, when any vector's starting point falls inside the query region, the vector clearly overlaps with q so it should be included in the result of the filter step. Next, when we assume that any location along the perimeter of q is an independent query point as in Section 5.3.1, the starting points of vectors that can reach the query point is bounded by a circle with radius M. Drawing circles along all the points on the perimeter forms the boundary of the search space for the range query.

Table 2
Summary of search space (PQ: point query).

		A. PQ	B. PQ with r	C. Directional PQ
Boundary condition		$\lvert p - q\rvert \leqslant M$	$\lvert p - q\rvert \leqslant r$ where $r < M$	$\lvert p_x - q_x\rvert \leqslant \lvert M\sin\beta\rvert$ $\lvert p_y - q_y\rvert \leqslant \lvert M\cos\beta\rvert$
Overlap condition	if $p_x > q_x$	$p_x - q_x \leqslant -V_X$	$p_x - q_x \leqslant -V_X$	$q_x + (-M\sin\beta) \leqslant -V_X$
	if $p_y > q_y$	$p_y - q_y \leqslant -V_Y$	$p_y - q_y \leqslant -V_Y$	$q_y + (-M\cos\beta) \leqslant -V_Y$
	if $q_x > p_x$	$q_x - p_x \leqslant V_X$	$q_x - p_x \leqslant V_X$	$q_x - (-M\sin\beta) \leqslant V_X$
	if $q_y > p_y$	$q_y - p_y \leqslant V_Y$	$q_y - p_y \leqslant V_Y$	$q_y - (-M\cos\beta) \leqslant V_Y$
	if $p_x = q_x$	any V_X	any V_X	any V_X
	if $p_y = q_y$	any V_Y	any V_Y	any V_Y

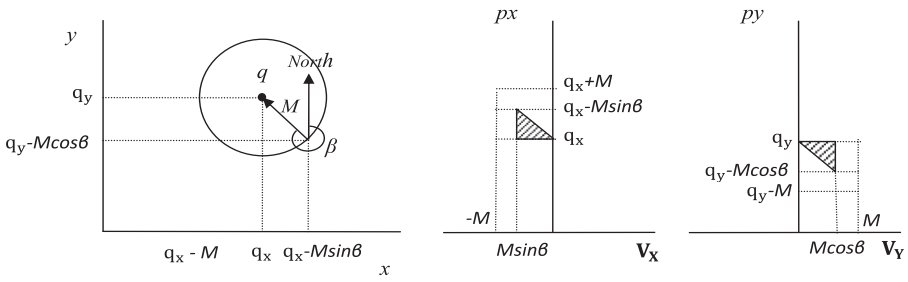


Fig. 6. Illustration of filter step in directional point query with angle β.

5.4. Ranking search results

In video search, when results are returned to a user, it is critical to present the most related videos first since manual verification (watching videos) can be very time-consuming. This can be accomplished by creating an order which will rank the videos from the most relevant to the least relevant. Otherwise, although a video clip completely captures the query region, it may be listed last within query results. It is essential to question the relevance of each video with respect to the user query and to provide an ordering based on estimated relevance. Our framework implements three ranking methods in the following subsections based on two relevant dimensions to calculate video relevance with respect to a query, i.e., its *spatial* and *temporal* overlap.

Analyzing how the FOV descriptions of a video overlap with a query region gives clues on calculating its relevance with respect to the given query. A natural and intuitive metric to measure spatial relevance is the extent of region overlap. The greater the overlap between FOVs and the query region, the higher the video relevance. To measure the temporal relevance, the time duration overlapping with the query region can be used. A video which captures the query region for a longer period will probably include more information about the region of interest and therefore can be more interesting to the user. Note that during the overlap period the amount of overlap at each time instant changes dynamically for each video. For example, among two videos whose total overlap amounts are comparable, one may cover a small portion of the query region for a long time and the rest of the overlap area only for a short time, whereas another video may cover a large portion of the query region for a longer time period. Fig. 7(a) and (b) illustrate the overlap between the rectangular query region and the videos V_1 and V_2, respectively. Although the actual overlapped area of the query is similar for both videos, the coverage by V_2 is much denser. Consequently, among the two videos V_2's relevance is higher.

Let Q be an arbitrary polygon-shaped query region. Suppose that a video clip V_k consists of n FOVs and t_s and t_e are the start time and end time for video V_k, respectively. The sampling time of the ith FOV is denoted as t_i. Note that FOVs can be collected at any time and timestamped. The starting time of a video t_s is defined as t_1. The ith FOV represents the video segment between t_i and t_{i+1} and the nth FOV, which is the last FOV, represents the segment between t_n and t_e (for convenience, say $t_e = t_{n+1}$). The set of all FOV descriptions for V_k is given by $V_k^F = \left\{FOV^{V_k}(t_i) | 1 \leqslant i \leqslant n\right\}$. Similarly, the FOV at time t_i is denoted as $V_k^F(t_i)$.

If Q is viewable by V_k, then the set of FOVs that capture Q is given by

$$OverlapSet(V_k^F, Q) = \left\{V_k^F(t_i) | \text{ for all } i \text{ where } V_k^F(t_i) \text{ overlaps with } Q\right\} \quad (1)$$

The overlap between V_k^F and Q at time t_i, forms a region as illustrated in Fig. 8. Let $O(V_k^F(t_i), Q)$ denote the overlapping area between video V_k^F and query Q at time t_i. More detailed descriptions and the exact calculations can be found in [7].

5.4.1. Ranking based on total overlap area

The total overlap area of $O(V_k^F, Q)$ covers all overlap regions formed between V_k^F and Q, i.e., the area of Q covered at least once by FOVs. Subsequently, the *relevance using total overlap area* (R_{TA}) is given by the area of $O(V_k^F, Q)$. A higher R_{TA} value implies that a video captures a larger portion of the query region Q and therefore

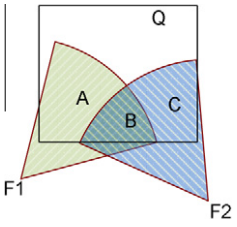


Fig. 8. Example of spatial overlap among query region Q and two FOVs.

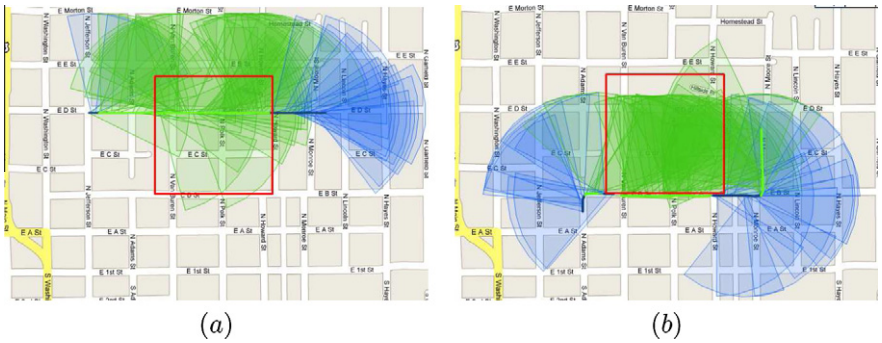


Fig. 7. Visualization of the overlap regions between query and videos (a) V_1 and (b) V_2.

its relevance with respect to Q can be higher. For example, two FOVs F1 and F2 overlap with Q in Fig. 8. The area A denotes the overlapped area between Q and F1 and C is the overlapped area between Q and F2. The area B is the intersection between A and C. Then, $R_{TA} = A + C - B$.

5.4.2. Ranking based on total overlap duration

The relevance using overlap duration (R_D) is given by the total time in seconds that V_k^F overlaps with query Q. Eq. (2) formulates the computation of R_D. R_D is obtained by summing the overlap time for each FOV in V_k^F with Q. We estimate the overlap time for each FOV as the difference between timestamps of two sequential FOVs.

$$R_D = \sum_{i=1}^{n} (t_{i+1} - t_i) \text{for } i \text{ when } O\left(V_k^F(t_i), Q\right) \neq \varnothing \tag{2}$$

When the duration of overlap is long, the video will capture more of the query region and therefore its relevance will be higher.

5.4.3. Ranking based on summed area of overlap regions

R_{TA} and R_D capture the spatial and temporal extent of the overlap, respectively. However both relevance metrics express only the properties of overall overlap and do not describe how each individual FOV overlaps with the query region. For example, in Fig. 7, for videos V_1 and V_2, although $R_{TA}(V_1^F, Q) \cong R_{TA}(V_2^F, Q)$ and $R_D(V_1^F, Q) \cong R_D(V_2^F, Q)$, V_2^F overlaps with around 80% of the query region Q during the whole overlap interval, whereas V_1^F overlaps with only 25% of Q for most of its overlap interval and overlaps with 80% of Q only for the last few FOVs. In order to differentiate between such videos, we propose the *relevance using summed overlap area* (R_{SA}) as the summation of areas of all overlap regions during the overlap interval. Eq. (3) formalizes the computation of R_{SA} for video V_k^F and query Q.

$$R_{SA}\left(V_k^F, Q\right) = \sum_{i=1}^{n} \left(O\left(V_k^F(t_i), Q\right) * (t_{i+1} - t_i)\right) \tag{3}$$

Using the example in Fig. 8, $R_{SA} = A + C$. Note that the area B was included twice in the calculation of R_{SA} but only once in R_{TA}.

6. Experiments

6.1. Prototype implementation and experimental methodology

To collect geo-tagged video data, we have constructed a prototype system which includes: (1) Canon VIXIA HV30 and JVC JY-HD10U cameras, with 1920×1080 and 1280×720 pixels at 30 frames per second, which produce MPEG-2 HD video streams around 20 Mb/s; (2) OS5000-US Solid State Tilt Compensated 3 Axis Digital Compass, which provides precise tilt compensated headings with roll and pitch data; and (3) Pharos iGPS-500 GPS receiver. A program was developed to acquire, process, and record the georeferences along with the MPEG-2 HD video streams. The system can process MPEG-2 video in real-time (without decoding the stream) and each video frame is associated with its meta-data. In all of our experiments, an FOV was constructed every second, i.e., one FOV per 30 frames of video.

Each video data packet received from the camera is processed in real time to extract frame timecodes. Extracted timecodes are recorded along with the local computer time when the frame was received. Since video data is received from the camera as data packet blocks, all frames within a video packet will initially have the same local timestamp. Creating a frame level time index for the video stream will minimize the synchronization errors that might occur due to clock skew between the camera clock and the computer clock. In addition, such a temporal video index, whose timing is compatible with other datasets, enables easy and accurate integration with the GPS and compass data. We also keep track of the size of the video data captured since the beginning of the capture and record the byte offset for each video frame.

In any actual video capturing, the behavior of a camera (i.e., its movements and rotations) depends on the occasion and purpose of the video recording. Such camera behaviors can be described by the pattern of camera movements and rotations. For example, a camera can be mounted on a vehicle and move along road networks such as equipped on city buses, police cars or ambulances. The camera direction is usually fixed or rotates within a predefined angle. Another example is a pedestrian camera. A walking tourist holds a hand-held camera to capture tourist attractions, landmarks, and special events and follows a random trajectory, analogous to a walking path. Then, the tourist can freely change the camera angle.

In collecting the real data for the experiments, we simulated the camera behavior as if videos were captured on a tourist bus, where the camera moves along a road network and it casually captures the street scenes. We mounted the recording system setup on a vehicle and captured video driving along streets at different speeds (max. 25 MPH). During video capture, we frequently changed the camera view direction. The recorded videos covered a 5.5 km by 4.6 km region (our entire search region, i.e., "the universe") quite uniformly. However, for a few popular locations we shot several videos, each viewing the same location from different directions. The total captured data includes 134 video clips, ranging from 60 to 240 s in duration. Each second, an FOV was collected, resulting in 10,652 FOVs in total.

As the user interface in our implementation, we developed a web-based search system [6]. The implemented query interface[1] allows users to visually draw the query region and view direction on the map. The result of a query contains a list of the overlapping video segments that show the query region from the query viewpoint. For each returned video segment, we display the corresponding FOVs on the map, and during video playback we highlight the FOV region whose timecode is closest to the current video frame. We constructed a MySQL database that stored all the FOV meta-data and created MySQL user-defined functions (UDFs) using the proposed vector model to search through the FOVs in the database. For media management and streaming we adopted the Wowza Media Server (http://www.wowzamedia.com/), and for video playback we used the Flowplayer (http://flowplayer.org/), an open source flash media player. In Section 6.2.2, we will demonstrate some example queries which we ran on our web-based search interface based on our real-world dataset. Figs. 11 and 12 are screenshots of our search interface.

To evaluate the implementation of our framework, we performed some experiments on the collected video meta-data. We generated 250 random query regions, of size 300 m by 300 m, within the 6 km by 5 km area of total video coverage. We then searched for the videos that captured at least one of the query regions and extracted the video segments that show these regions, i.e., where FOVs and the query region intersect. A detailed description of the FOV-based search algorithm can be found in our prior work [8].

The first group of experiments includes a user study where we compare the results obtained from our search system to user provided feedbacks. Our main intention in performing the user study is to check whether the results of the proposed search technique are plausible to humans. Our methodology therefore does not live up to the rigorous process usually attributed to scientific user studies. We next evaluated the accuracy of the FOV-based search and compared the search results to those obtained from other

[1] http://eiger.ddns.comp.nus.edu.sg/geo/Query_idaho.html.

approaches. We also performed analysis on the performance of the proposed ranking algorithms.

The second group of experiments includes examples for the new directional and bounded distance query types described in Section 5.1. We demonstrate that our implementation successfully retrieves the relevant video results, based on the specifications of the proposed new query types. We provide some screenshots for the example queries executed on our web-based search interface.

6.2. Results

6.2.1. Feasibility analysis

We evaluated the feasibility and adequacy of our FOV-based video search with a specific focus on the completeness and accuracy of search results.

The completeness of results is hard to verify, since there is no easy way to get the *ground truth* for the query result set. One possible way is to have a human subject watch through all videos and confirm whether a query region is visible within a video. Such manual verification is prone to errors, however, human feedback is still the most reliable source to determine whether an object is visible within a video. For the user study, we randomly chose 40 videos and manually verified the overlap among FOVs and query regions (details can be found in [8]).

To evaluate the accuracy of the query result set, we compared our approach to two other video scene description models: (1) the *CircleScene* model – the camera viewable scene is described as a circular region around the camera location with the assumption that the view direction is not known. A query is visible if its region intersects with the circular viewable scene (see Fig. 2), (2) the *PointScene* model – the camera viewable scene is the camera location point. A query is visible if the camera point resides within the query's region.

6.2.1.1. Completeness of search results. In this set of experiments we compared algorithmic results with user provided feedback (*ManualCheck*). Given 250 queries, through manual scan, we created the list of query regions that are visible within each video in the dataset of 40 videos. We then executed the same set of queries using the proposed FOV model, the CircleScene model and the PointScene model on the same 40 videos. Fig. 9 shows the number of queries marked as visible for each video file by manual verification and the three search algorithms mentioned above. Note that we used the same far visible distance (R) value for both the CircleScene and the FOV models.

Results show that FOV-based search completely returns all videos marked visible through ManualCheck. However, the query result also includes a small number of false positives (i.e., returned as an intersecting query but the scene does not actually show the query region), which might occur due to the following reasons: when the camera view is occluded with big structures or when the camera viewable scene intersects only a small percentage of the query region, the human subject might not include the query in the manual results. As expected, the CircleScene model returns an excessive percentage of extra irrelevant videos and overestimates the manual search results while the PointScene model underestimates the manual search results by returning only a subset of the visible queries.

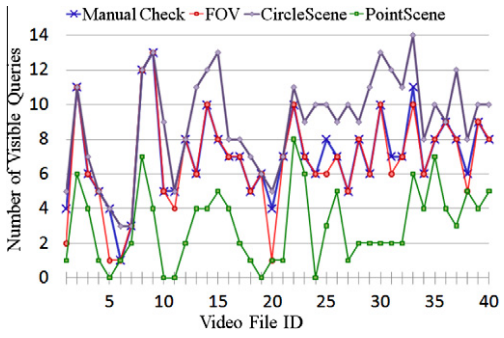


Fig. 9. Number of visible queries per video file.

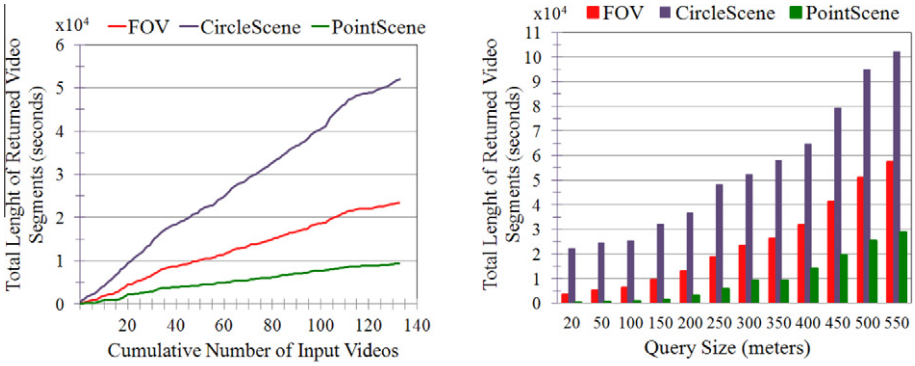
6.2.1.2. Accuracy of search results. We next compared the effectiveness of the three search algorithms in identifying the relevant videos. Fig. 10(a) shows the total length of all video segments identified by each search algorithm while varying the number of input video files. We have used the cumulative sums to show the overall difference as the input data size grows. The graph clearly shows the superiority of the FOV model over the CircleScene and PointScene models. It is important to note that, although a query region is marked as visible by both the FOV and CircleScene models, the video segments they report for the appearance of the query region in a single video can be different. For example, as shown in Fig. 2, when the camera is rotated, although the query region is not visible anymore in the video, CircleScene will still report that the query intersects with its viewable scene for the following frames. Therefore the FOV model eliminates the frames that do not show the query region and returns more precise results with less false positives. Considering the huge size of video data and time-consuming human verification process for the final result, this significant reduction of false positives can greatly enhance the performance of video search.

To analyze the effect of the query size over the total length of the returned video segments, we repeated the same experiments shown in Fig. 10(a) while varying the size of the query regions. Fig. 10(b) reports the total length of the returned video segments by all three approaches for query region sizes ranging from 20 m $\times$ 20 m to 550 m $\times$ 550 m. For smaller query regions we observed bigger differences between the three approaches, i.e., the superiority of FOV model is maximized for small query regions. The performance gap among the three approaches reduced as the query size increased. For sizes bigger than 550 m $\times$ 550 m, we have not observed dramatic changes in the results.

6.2.1.3. Accuracy of ranking. We rank the search results obtained from the 250 queries based on the three metrics proposed in Section 5.4. The rank lists RL_{TA}, RL_{SA} and RL_D are constructed from the relevance metrics R_{TA}, R_{SA} and R_D, respectively.

Similar to the previous completeness issue, it is not easy to define a single best rank. Thus, we introduced another user study to obtain the rank lists of search results based on human judgement. We used an evaluation metric termed Discounted Cumulated Gain (DCG) [16], which systematically combines the video rank order and degree of relevance. The Normalized-DCG (NDCG) is the final DCG sum normalized by the DCG of the ideal ordering (i.e., rank from human judgement). The higher the NDCG of a given ranking the more accurate it is. Next, the NDCG scores with respect to the user results were calculated. The NDCG scores of RL_{TA}, RL_{SA} and RL_D were 0.975, 0.951 and 0.921, respectively. All scores are close to 1, which implies that all three are highly successful in ranking the most relevant videos at the top, similar to human judgement. The high NDCG scores further lend credibility to the claim that the proposed ranking methods successfully identify the most relevant videos.

The three ranking techniques R_{TA}, R_{SA} and R_D interpret the relevance in geo-tagged video search by somewhat different means. Therefore they are not expected to produce an identical ranking order across all schemes. However, we conjecture that they all should contain similar sets of video clips within the top N of their rank lists (for some N). A similar result from all three ranking algo-



(a) Cumulative sum of returned video segment lengths as a function of number of input videos (whole dataset)

(b) Effect of query size.

Fig. 10. Comparison of the results of the three search algorithms (using FOV, CircleScene and PointScene models).

rithms would indicate that the resulting videos are most interesting to the user. To compare the accuracy of the results, we adopt the *Precision at N* ($P(N)$) metric, which is a popular method that describes the fraction of relevant videos ranked in the top N results. We redefine $P(N)$ as the fraction of common videos ranked within the top N results of more than one rank list. $P(N)$ only shows the precision of a single query therefore, to measure the average precision over multiple queries, we use the *Mean Average Precision (MAP)*, which is the mean of several $P(N)$ from multiple queries.

We compare the ranking accuracy of RL_{TA}, RL_{SA} and RL_D using MAP scores. In Table 3, the first row calculates the MAP values as the average ratio of the videos that are common to all three rank lists within the top 1, 2, 5, 10 and 20 ranked results for all 250 queries, respectively. The second, third and fourth rows display the MAP scores pair-wise for two methods each: R_{TA} and R_{SA}, R_{TA} and R_D, and R_D and R_{SA}. The results show that the precision increases as N grows and achieves a close to perfect score beyond $N = 10$. Note that the precision is very high even at $N = 5$. This implies that all three proposed schemes similarly identify the most relevant videos.

Even though the results among the ranking methods vary somewhat, at this point we do not favor any specific approach. We believe that each ranking scheme emphasizes a different aspect of relevance, therefore query results should be customized based on user preferences and application requirements.

6.2.2. Functionality illustration

In Section 5.1 we introduced some of the spatial query types that can be applied in geo-tagged video search to enforce application specific search information. For example, one might specify the query position, the view direction from the camera, and the distance between the location of the query and the camera. In this section, we provide examples of such queries and demonstrate the new functionality of our geo-tagged video search system. All results were facilitated by the implementation of the vector model in Section 5.1 and examples are illustrated through the screen shots from our web-based video search interface. Details of the implementation and experiments can be found in [19].

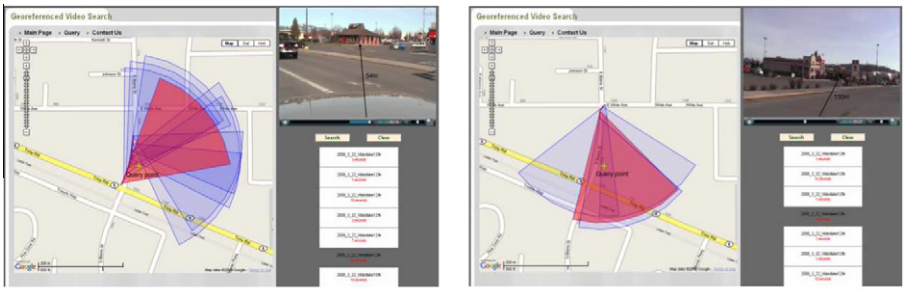
6.2.2.1. Query with bounded distance. Fig. 11 illustrates the results of a bounded distance query on our real-world video data. We searched for the video segments that show the *Pizza Hut* building in the scenes. The query returns 12 video segments (total 120 s of video). Two of the resulting video segments are shown in Fig. 11(a) and (b). The *Pizza Hut* building appears very small (and is difficult to be recognized by humans) in the second figure since it was located far from the camera. Note that the same building is easily recognizable in the first figure when the camera was closer to the object. We can effectively exclude the video segment shown in Fig. 11(b) using an appropriate bounded distance value (e.g., 100 m) in the query. The camera FOVs for the video segments are illustrated on the map. The FOVs that corresponds to the current video frames are highlighted in both images.

6.2.2.2. Directional query. In Fig. 12, we illustrate an example of a directional query. We would like to retrieve the video segments that overlap with the given query region (the *University of Idaho Kibbie Dome* in the scenes) while the camera was pointing in the *North* direction. Fig. 12(a) shows the video segments returned from the range query without the notion of directionality. And Fig. 12(b) shows the results of the directional range query with input direction 0° (*i.e., North*). Without the direction condition the *Kibbie Dome* query returns a total of 250 s of video whereas the directional query returns only 65 s of video. As shown in Fig. 12(b), the directional query precisely returns the related video segments and eliminates the unwanted videos and video sections.

For a specific application, the bounded distance query, or the directional query, or a combination of the two can be used to effectively retrieve the related video segments. Based on the

Table 3
Comparison of proposed ranking methods: RL_{TA}, RL_{SA} and RL_D.

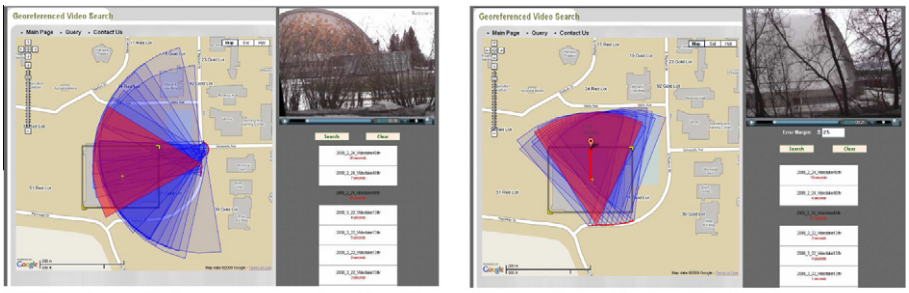
	Top N results	MAP at $N = 1$	MAP at $N = 2$	MAP at $N = 5$	MAP at $N = 10$	MAP at $N = 20$
Compare all	$\frac{N(RL_{TA}) \bigcap N(RL_D) \bigcap N(RL_{SA})}{N}$	0.60	0.789	0.918	0.993	1.0
Compare RL_{TA} and RL_{SA}	$\frac{N(RL_{TA}) \bigcap N(RL_{SA})}{N}$	0.727	0.839	0.961	0.993	1.0
Compare RL_{TA} and RL_D	$\frac{N(RL_{TA}) \bigcap N(RL_D)}{N}$	0.677	0.842	0.933	0.987	1.0
Compare RL_{SA} and RL_D	$\frac{N(RL_{SA}) \bigcap N(RL_D)}{N}$	0.745	0.885	0.947	0.987	1.0



(a) The object is close to the camera (30m away), therefore can be easily recognized in video

(b) The object is far from the camera (130m away), therefore appears very small in video.

Fig. 11. Impacts of bounding distance in video search. The FOVs for the current video frames are highlighted on the map.



(a) Search results for range query (no direction specified).

(b) Search results for directional range query (viewing direction 0°)

Fig. 12. Illustration of directional range query results. The FOVs for the current video frames are highlighted on the map.

application's requirements the query location can be specified as a point or a range. In these scenarios, our search mechanism effectively and efficiently reduces the amount of data returned to user, therefore minimizing the user browsing time.

7. Conclusion

In this study, we advocated geo-tagged video acquisition as a means to manage large scale video content. We proposed a framework in support of various applications such as spatio-temporal search. We further described a design and various implementation details of the framework to demonstrate its real-world feasibility and adequacy. Our results show that many of the fundamental aspects of our proposed framework can be effectively instantiated.

Interesting challenges remain. While our framework is designed from the ground up for scalability, its performance on a large scale will need to be further investigated and validated. The extensibility of the framework also presents an interesting aspect as additional sensors, such as accelerometers, are added.

Acknowledgment

We would like to acknowledge the support of the Centre of Social Media Innovations for Communities (CoSMIC), sponsored by the Media Development Authority (MDA) of Singapore.

References

[1] Centernetwork. <http://www.centernetworks.com/user-generated-video-market-size-2008>.
[2] Flickr. <http://www.flickr.com>.
[3] Geobloggers. <http://www.geobloggers.com>.
[4] Reelseo. <http://www.reelseo.com/youtube-search-december-2009/>.
[5] Woophy. <http://www.woophy.com>.
[6] Sakire Arslan Ay, Lingyan Zhang, Seon Ho Kim, Ma He, Roger Zimmermann, GRVS: a georeferenced video search engine, in: ACM International Conference on Multimedia, 2009, pp. 977–978.
[7] Sakire Arslan Ay, Seon Ho Kim, Roger Zimmermann, Relevance ranking in georeferenced video search, Multimedia Syst. 16 (2) (2010) 105–125.
[8] Sakire Arslan Ay, Roger Zimmermann, Seon Ho Kim, Viewable scene modeling for geospatial video search, in: ACM International Conference on Multimedia, 2008, pp. 309–318.
[9] N. Beckmann, H.-P. Kriegel, R. Schneider, B. Seeger, The r*-tree: an efficient and robust access method for points and rectangles, in: ACM SIGMOD International Conference on Management of Data, 1990.
[10] T. Brinkhoff, H.-P. Kriegel, R. Schneider, B. Seeger, Multi-step processing of spatial joins, in: ACM SIGMOD International Conference on Management of Data, 1994.
[11] Michael Christel, Automated Metadata in Multimedia Information Systems: Creation, Refinement, Use in Surrogates, and Evaluation, Morgan and Claypool Publishers, 2009.
[12] Cisco Systems, Inc., Cisco Visual Networking Index: Forecast and Methodology, 2009–2014, White Paper, 2010.
[13] Boris Epshtein, Eyal Ofek, Yonatan Wexler, Pusheng Zhang, Hierarchical photo organization using geo-relevance, in: 15th ACM International Symposium on Advances in Geographic Information Systems (GIS), 2007, pp. 1–7.
[14] Clarence H. Graham, Neil R. Bartlett, John Lott Brown, Yun Hsia, Conrad C. Mueller, Lorrin A. Riggs, Vision and Visual Perception, John Wiley & Sons, Inc., 1965.
[15] Tae-Hyun Hwang, Kyoung-Ho Choi, In-Hak Joo, Jong-Hun Lee, MPEG-7 metadata for video-based GIS applications, in: Geoscience and Remote Sensing Symposium, vol. 6, 2003, pp. 3641–3643.
[16] Kalervo Järvelin, Jaana Kekäläinen, Cumulated gain-based evaluation of IR techniques, ACM Trans. Inform. Syst. 20 (4) (2002) 422–446.
[17] Yushi Jing, Shumeet Baluja, Visualrank: applying pagerank to large-scale image search, IEEE Trans. Pattern Anal. Mach. Intell. 30 (2008) 1877–1890.
[18] Kyong-Ho Kim, Sung-Soo Kim, Sung-Ho Lee, Jong-Hyun Park, Jong-Hyun Lee, The interactive geographic video, in: Geoscience and Remote Sensing Symposium, vol.1, 2003, pp. 59–61.
[19] Seon Ho Kim, Sakire Arslan Ay, Byunggu Yu, Roger Zimmermann, Vector model in support of versatile georeferenced video search, in: ACM Multimedia Systems Conference, 2010, pp. 235–246.
[20] Michael S. Lew, Nicu Sebe, Chabane Djeraba, Ramesh Jain, Content-based multimedia information retrieval: state of the art and challenges, ACM Trans. Multimedia Comput. Commun. Appl. 2 (1) (2006) 1–19.
[21] Xiaotao Liu, Mark Corner, Prashant Shenoy, SEVA: sensor-enhanced video annotation, in: ACM International Conference on Multimedia, 2005, pp. 618–627.

[22] Ying Liu, Dengsheng Zhang, Guojun Lu, Wei-Ying Ma, A survey of content-based image retrieval with high-level semantics, Pattern Recogn. 40 (1) (2007) 262–282.
[23] Jiebo Luo, Jie Yu, Dhiraj Joshi, Wei Hao, Event recognition: viewing the world with a third eye, in: ACM International Conference on Multimedia, 2008, pp. 1071–1080.
[24] Mor Naaman, Yee Jiun Song, Andreas Paepcke, Hector Garcia-Molina, Automatic organization for digital photographs with geographic coordinates, in: 4th ACM/IEEE-CS Joint Conference on Digital Libraries, 2004, pp. 53–62.
[25] A. Orenstein, Spatial query processing in an object-oriented database system, in: ACM SIGMOD International Conference on Management of Data, 1986.
[26] Theo Pavlidis, Why meaningful automatic tagging of images is very hard, in: IEEE ICME 2009, 2009, pp. 1432–1435.
[27] A. Pigeau, M. Gelgon, Building and tracking hierarchical geographical & temporal partitions for image collection management on mobile devices, in: ACM International Conference on Multimedia, 2005.
[28] Ian Simon, Steven M. Seitz, Scene Segmentation Using the Wisdom of Crowds, in: Proc. ECCV, 2008.
[29] Alan F. Smeaton, Paul Over, Wessel Kraaij, Evaluation campaigns and TRECVid, in: MIR '06: Proceedings of the 8th ACM International Workshop on Multimedia Information Retrieval, 2006, pp. 321–330.
[30] Alan F. Smeaton, Paul Over, Wessel Kraaij, High-level feature detection from video in TRECVid: a 5-year retrospective of achievements, in: Ajay Divakaran (Ed.), Multimedia Content Analysis, Theory and Applications, Springer-Verlag, 2009, pp. 151–174.
[31] Cees G.M. Snoek, Marcel Worring, Concept-based video retrieval, Found. Trends Inf. Retr. 2 (4) (2009) 215–322.
[32] Carlo Torniai, Steve Battle, Steve Cayzer, Sharing, Discovering and Browsing Geotagged Pictures on the Web, Springer, 2006.
[33] Kentaro Toyama, Ron Logan, Asta Roseway, Geographic location tags on digital images, in: ACM International Conference on Multimedia, 2003, pp. 156–166.
[34] Takamasa Ueda, Toshiyuki Amagasa, Masatoshi Yoshikawa, Shunsuke Uemura, A system for retrieval and digest creation of video data based on geographic objects, in: DEXA '02: Proceedings of the 13th International Conference on Database and Expert Systems Applications, Springer-Verlag, 2002, pp. 768–778.
[35] Richard Wray, Online video ads put message into the medium, in: The Guardian, 2008. <http://www.guardian.co.uk/media/2008/dec/29/blinkx-internet-video-advertisin>.
[36] Hongjiang Zhang, Multimedia content analysis and search: new perspectives and approaches, in: ACM International Conference on Multimedia, ACM, 2009, pp. 1–2.
[37] Yan-Tao Zheng, Ming Zhao Yang, Song H. Adam, U. Buddemeier, A. Bissacco, F. Brucher, Tat-Seng Chua, H. Neven, Tour the world: building a web-scale landmark recognition engine, in: IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2009, pp. 1085–1092.