GeoSearch: Georeferenced Video Retrieval System [*]

Youngwoo Kim, Jinha Kim, Hwanjo Yu[†]
Department of Computer Science and Engineering
Pohang University of Science and Technology(POSTECH)
Pohang, South Korea
{ywkim10, goldbar, hwanjoyu}@postech.ac.kr

ABSTRACT

Conventional video search systems, to find relevant videos, rely on textual data such as video titles, annotations, and text around the video. Nowadays, video recording devices such as cameras, smartphones and car blackboxes are equipped with GPS sensors and able to capture videos with spatiotemporal information such as time, location and camera direction. We call such videos *georeferenced videos*. This paper presents a georeferenced video retrieval system, GeoSearch, which efficiently retrieves videos containing a certain point or range in the map. To enable a fast search of georeferenced videos, GeoSearch adopts a novel data structure MB*T*R (Minimum Bounding *Tilted* Rectangle) in the leaf nodes of R-Tree. New algorithms are developed to build MB*T*Rs from georeferenced videos and to efficiently process point and range queries on MB*T*Rs. We demonstrate our system on real georeferenced videos, and show that, compared to previous methods, GeoSearch substantially reduces the index size and also improves the search speed for georeferenced video data. Our online demo is available at "http://dm.hwanjoyu.org/geosearch".

Categories and Subject Descriptors

H.2.4 [**Database Management**]: Systems—*Query processing*; H.2.4 [**Database Management**]: Systems—*Multimedia databases*; H.2.8 [**Database Management**]: Database Applications—*Spatial databases and GIS*

General Terms

Algorithms, Performance

Keywords

Georeferencing, Video search, Spatial Indexing

[*]This work was partially supported by the Brain Korea 21 Project in 2012 and Mid-career Researcher Program through NRF grant funded by the MEST (No. KRF-2011-0016029). This work was also supported by IT Consilience Creative Program of MKE and NIPA (C1515-1121-0003).

[†]corresponding author

1. INTRODUCTION

With the rapid popularization of video recording devices, not only the media made by professionals but also made by ordinary users called UCC (User-Created Contents) are taking a large portion in the web. Since identifying related video requires understanding of video contents, which is a very heavy and complex process, current video search systems mostly rely on textual data such as video titles, annotations, and text around the video.

Nowadays, video capturing devices such as cameras, smartphones and car blackboxes are equipped with GPS sensors and able to capture videos with spatiotemporal information such as time, location and camera direction. We call such videos *georeferenced videos*. For some applications, such spatiotemporal information plays key roles in querying georeferenced videos. For example, some people may want to find videos of specific event that occurred at specific time and location, e.g., videos of a traffic accident captured by car blackboxes, videos of a goal scene in a soccer play captured by users, or videos of a concert with celebrities captured by users.

This paper presents GeoSearch, a georeferenced video retrieval system, which efficiently retrieves videos containing a certain point or range in the map. GeoSearch adopts a novel data structure MB*T*R (Minimum Bounding *Tilted* Rectangle) in the leaf nodes of R-Tree, in order to enable a fast search of georeferenced videos, While traditional MBR (Minimum Bounding Rectangle) is popularly used to describe an area in spatial indexes such as R-Tree, MBR is not suitable for describing the areas of moving scenes where its location and direction are continously changing. MB*T*R is designed to describe an area (or scenes) of moving location and direction. New algorithms are developed to build MB*T*Rs from georeferenced videos and to efficiently process point and range queries on MB*T*Rs. We demonstrate our system on real georeferenced videos, and show that, compared to previous methods, by adopting MB*T*R in the R-Tree, GeoSearch substantially reduces the index size and also improves the search speed for georeferenced video data.

Our demo is available at "http://dm.hwanjoyu.org/geosearch". Due to space limitation, this paper focuses on a high-level overview of techniques and demonstration. Technical details are described in our technical report [10].

1.1 Related Work

Georeferenced Multimedia Search: There are several approaches to the use of location information in image search. Toyama et al [21] used metadata in image search and developed a database for it. They enabled spatial image search by recording images with longitudinal and latitudinal coordinates and timestamps. Several commercial websites, such as Flickr and Woophy [1], also provide georeferenced image search. They focus on showing static images of the locations in a map and support simple query such as nearest neighbor query. Google Street View [2] provides streams of street view images of their own. We can reference these images by camera's location and viewing direction.

Trajectory indexing methods were developed in order to efficiently store and search moving objects [16, 8]. Our indexing method for supporting moving *scenes* can be considered an extension of trajectory indexing which takes into consideration camera directions as well as the location of the cameras.

Research associated with "trajectory-based video indexing" [7, 17] and "spatial indexing for video" [20] have been published in the computer vision community. However, the problem they address is different from ours; their goal is to index the relative location of objects in the video, not the viewable scenes. They also use video content retrieval methods such as object segmentation and motion tracking to detect objects, while we use metadata to search georeferenced videos.

A viewable scene model for enabling georeferenced video search has been recently proposed by Ay et al. [3, 5]. Based on the viewable scene model, they developed methods for supporting point and range queries using MBR-based filtering. (We compare this method as a baseline method in our experiments in Section 3.) Our method builds an index GeoTree based on their viewable scene model.

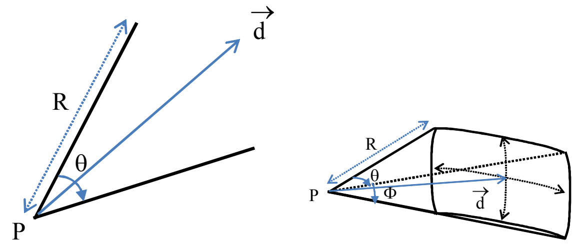
They also developed a *relevance* video search method based on the viewable scene model, and proposed metrics to measure the relevance of video to the given range query [3]. The metrics compute the relevance scores based on the size of the overlapping area. Since computing the exact size of the overlapping area is computationally expensive, they approximate it using grids and histograms. However, their approach requires vast storage to keep all the videos stored in grids, and also the accuracy is compromised due to the running of the query on the gridded data instead of the original data. Moreover, the overlapping area-based relevance may not reflect the true relevance implied in the user's query; user's relevance may be more related to parts that are overlapped and the distance to the overlapping area. Inducing a good relevance function is itself a non-trivial research problem.

Ay et al. also proposed a method for generating synthetic metadata for georeferenced video search [4]. Since it is often difficult to obtain a large set of georeferenced video data, this method can be used for evaluating new methods for georeferenced video search.

Spatial Indexing: There are numerous works on spatial indexing and query processing in the database community. Many of the indexing structures are based on R-Tree [9, 13, 19]. Several works focus on querying location trajectories of moving objects [14, 18]. Also, several effective indexing structures such as STR-Tree and TB-tree were proposed [15]. These works mostly focus on dealing with temporal dimension and complex queries, rather than dealing with complex spatial data such as viewable scenes.

Spatiotemporal trajectory compressions for saving data storage and enabling fast query processing are also being researched [12, 8]. Their methods focus on the data of points or location trajectories that are not directly applicable to viewable scenes. There are also works that focus on indexing range data or region data that consider static rectangular objects [6, 11].

No effective indexing structure has yet been proposed that supports exact queries on camera viewable scenes of moving locations and directions.



(a) FOVscene in 2D (b) FOVscene in 3D

Figure 1: Examples of FOVscene in 2D and 3D spaces

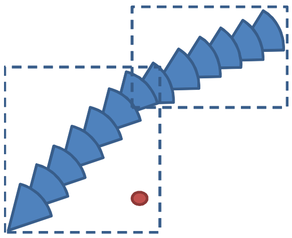
2. GEOTREE: SPATIAL INDEXING FOR GEO-REFERENCED VIDEO SEARCH

Preliminary: Viewable Scene Model: The viewable scene changes, as the camera moves or rotates. The area captured by the camera is referred to as the Field-Of-View scene (FOVscene) [5]. As illustrated in Figure 1(a), in a 2D space, four parameters are required to express a single FOVscene: (1) location of the camera P, (2) direction of the camera $\vec{d}$, (3) viewable angle of the camera θ, and (4) visible distance R. Camera location P is represented as a (latitude, longitude) pair that can be obtained using a GPS sensor. Camera direction $\vec{d}$ can be obtained using a compass sensor. Camera viewable angle θ and visible distance R can be obtained from the camera lens' property and zoom level. As assumed in [5, 3], we also fixed θ and R by assuming the use of static camera lens. (Some video recording devices such as smartphones and car blackboxes use static camera lenses.)

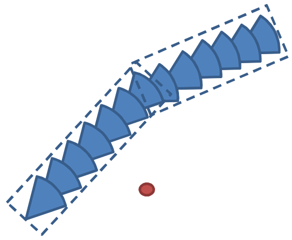
The camera view or the viewable scene information at a particular moment is entirely expressed by an FOVscene. An FOVscene corresponds to a frame in the video. A georeferenced video can be represented by a sequence of FOVscenes. We use the term FOVstream to denote a sequence of FOVscenes. As done in [5, 3], we restrict FOVscene to a 2D space, however it can be extended to a 3D space in a straightforward manner. Altitude and pitch should be considered in the 3D model (Figure 1(b)). but they do not alter the nature of the FOVscene model.

Index Construction and Query Processing: To enable an efficient indexing of georeferenced video search, videos must be recorded with the metadata of $(P, \vec{d}, \theta, R)$ for each FOVscene. Video recording devices such as car blackboxes and smartphones are nowadays equipped with sensors that have the ability to record metadata with videos. We construct a spatial index called GeoTree using the metadata. When a point or range query is submitted by users, it is processed on the GeoTree by finding the FOVstreams that overlap the query area.

GeoTree adopts a novel structure called Minimum Bounding *Tilted* Rectangle (MB*T*R). GeoTree uses MB*T*R instead of MBR (Minimum Bounding Rectangle) to represent an area of moving scenes in the index. GeoTree is a kind of R-tree using MBR in the nonleaf nodes and MB*T*R in the leaf nodes. Figure 2 depicts examples of MBR and MB*T*R representing areas of moving scenes. MB*T*R is more suitable for representing an area of moving scenes; as video recording devices such as car blackboxes are often moving piece-wise linearly, MB*T*R produces less false positive areas. For example, when a query, i.e., the red circle in the figure, is submitted, the MBR in Figure 2(a) includes the query area, thereby incurring a large area of false scanning of videos while the MB*T*R in Figure 2(b) efficiently prunes the videos.



(a) MBR-based filtering



(b) MBTR-based filtering

Figure 2: Examples of MBR (Minimum Bounding Rectangle) and MBTR (Minimum Bounding *Tilted* Rectangle). Red circle is the querying area. Blue circular sector is a viewable scene

	MBR-Filtering	R-Tree	GeoTree
Range Query (ms)	297,800	6,039.4	4,554.2
Point Query (ms)	68,297	1,302.0	1,264.5
Memory Use (MB)	16.6	27.3	2.83

Table 1: Comparison of Query Processing Time

Building MBTRs requires modeling moving scenes piece-wise linearly such that each MBTR covers as large an area as possible while producing as small a number of false positive areas as possible. Therefore, we first identify *markup* FOVscene*s* from moving scenes, while considering the changing patterns of P and $\vec{d}$. Markup FOVscenes are those at the edge of an MBTR, such that the FOVscenes between markup FOVscenes are moving linearly in terms of P and constant in terms of $\vec{d}$, and thus any FOVscenes can be estimated from the markup FOVscenes. Once a set of markup FOVscenes are found, MBTRs from pairs of adjacent markup FOVscenes are constructed.

A point or range query is processed in a similar fashion to R-Tree in the nonleaf nodes, as the nonleaf nodes of GeoTree are also described by MBRs. Once a query reaches a leaf node in the GeoTree, it is processed in two steps – (1) MBTR filtering and (2) MBTR lookup. MBTR filtering ascertains if the query can overlap the MBTR. If MBTR filtering returns *false*, this means that there is no FOVscene in the MBTR that overlaps the query. However, if MBTR filtering returns *true*, MBTR lookup is called to compute and return an expected subsequence of FOVscenes that overlap the query. Since an MBTR typically contains numerous FOVscenes, it would be inefficient to scan all the FOVscenes in the MBTR to find the overlapping FOVscenes. We directly compute, using the MBTR structure, a sequence of FOVscenes that is expected to contain the query point.

3. EXPERIMENTS

We compare the performance of GeoTree to R-Tree and MBR-Filter, i.e., the MBR-based filtering proposed in [3]. We evaluate the performance of processing point queries and range queries in terms of query processing time and also memory usage. Our experiments were done using a machine with the following specifications: dualcore CPU (2.4GHz) and 4GB of Memory.

DataSet: Experiments were performed on real FOVstream data sets. We captured real scenes using an Android mobile phone equipped with a GPS receiver and a 3D compass. We ignored altitude and pitch angles from the 3D compass, as we are only interested in 2D. Thirty FOVscenes were captured every second. Latitudinal and longitudinal data were converted into meter metrics in the 2D plane. No zoom level change was simulated, thus viewable angle θ and visible distance R were kept constant and thus fixed to $\theta = 55^\circ$ and $R = 50m$.

FOVscenes ranged in the area of 3.3km long in the north-south and 3.8km long in east-west. Test data corresponded to 11 streams with a total duration of 180 minutes. The number of FOVscenes was 325,313 in total. This number of streams corresponded to approximately 4.4GB of typical video data. The sum of all trajectories' length was approximately 68km. For the leaf nodes of GeoTree with less than 5 FOVscenes, we simply used simple one-by-one search and did not perform MBTR filtering and lookup, because, when an MBTR contains just a few FOVscenes, scanning FOVscenes one-by-one runs faster due to some overhead of MBTR filtering and lookup.

Result Summary: Table 1 summarizes the results of the query processing time for each method. For the range query test, we randomly generated 10,000 queries and measured the accumulated processing time for the queries. For the point query test, we randomly generated 100,000 queries and also measured the accumulated processing time for the queries. The processing times are averaged by 30 runs. For GeoTree, we first tuned the error threshold parameters on the dataset, and fixed the parameter values for all runs, these were. $\epsilon_P = 1m$ and $\epsilon_\theta = 10^\circ$

MBR-Filter was one hundred times slower than the other tree-based searches. We omitted the naive sequential search results because it was even much slower than MBR-Filter. GeoTree gave a consistently faster query processing time than R-Tree. Although the query processing times of R-Tree and GeoTree were not substantially different, their memory usages were an order of magnitude different. As Figure 3 shows, the performance difference became more significant, as data size increased.

4. DEMONSTRATION PLAN

We plan to demonstrate our system on a real map of the campus of POSTECH[1]. We captured various places of the campus and uploaded the videos to YouTube. We then built an online website, "http://dm.hwanjoyu.org/geosearch/", which can search corresponding video frames using point or range queries. In our online search website, a user submits a point or range query by mouse clicks on the map, and our system returns corresponding video frames using Tubechop[2]. Figure 4(a) illustrates an example of a range query in our website. Figure 4(b) shows the result in text and links to video views. Once the user clicks a link, our system plays corresponding video frames as shown in Figure 4(c).

5. REFERENCES

[1] Woophy. http://www.woophy.com.

[2] D. Anguelov, C. Dulong, D. Filip, C. Frueh, S. Lafon, R. Lyon, A. Ogale, L. Vincent, and J. Weaver. Google street view: Capturing the world at street level. *Computer*, 43:32–38, June 2010.

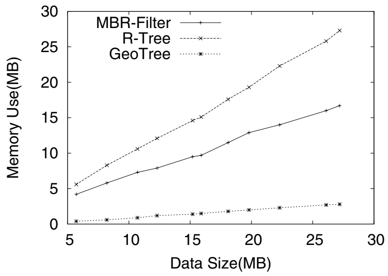
[3] S. Arslan Ay, R. Zimmermann, and S. Kim. Relevance ranking in georeferenced video search. *Multimedia Systems*, 16:105–125, 2010. 10.1007/s00530-009-0177-x.

[4] S. A. Ay, S. H. Kim, and R. Zimmermann. Generating synthetic meta-data for georeferenced video management. In *Proceedings of the 18th SIGSPATIAL International Conference on Advances in Geographic Information Systems*, GIS '10, pages 280–289, New York, NY, USA, 2010. ACM.

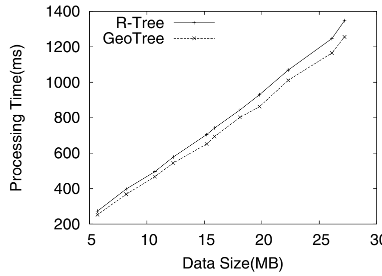
[5] S. A. Ay, R. Zimmermann, and S. H. Kim. Viewable scene modeling for geospatial video search. In *Proceedings of the 16th ACM*

[1] http://www.postech.ac.kr

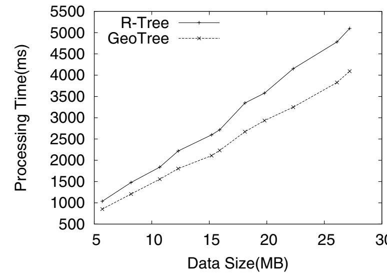
[2] www.tubechop.com: a site that shows a segment of YouTube videos



(a) Memory Use

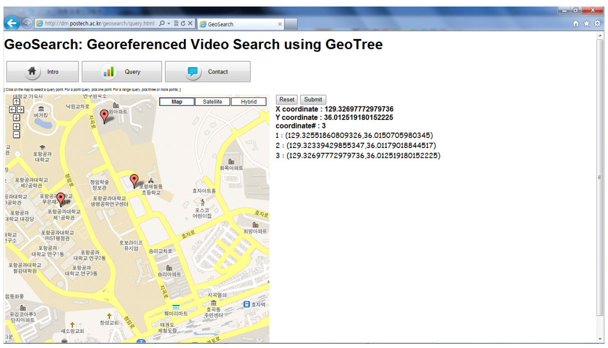


(b) Point Query Processing Time

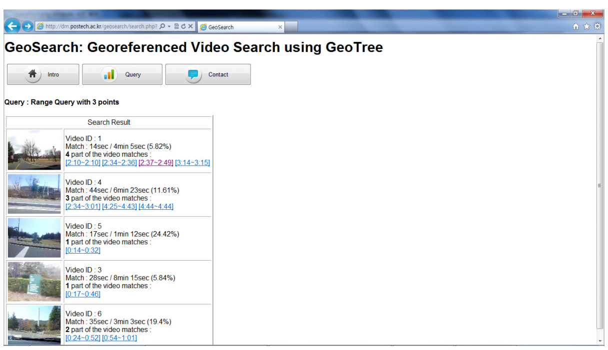


(c) Range Query Processing Time

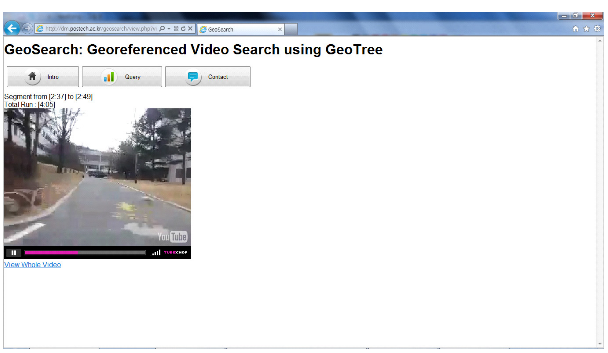
Figure 3: Result of Varying Data Sizes



(a) Point Query



(b) Query Result



(c) Video view on YouTube

Figure 4: Demonstration Site

international conference on Multimedia, MM '08, pages 309–318, New York, NY, USA, 2008. ACM.

[6] M. Chaabouni and S. M. Chung. The point-range tree: a data structure for indexing intervals. In *Proceedings of the 1993 ACM conference on Computer science*, CSC '93, pages 453–460, New York, NY, USA, 1993. ACM.

[7] S. Dagtas, W. Al-Khatib, A. Ghafoor, and R. Kashyap. Models for motion-based video indexing and retrieval. *Image Processing, IEEE Transactions on*, 9(1):88 –101, jan 2000.

[8] E. Frentzos and Y. Theodoridis. On the effect of trajectory compression in spatiotemporal querying. In *Proc. of East European Conf. Advances in Databases and Information Systems*. Springer Berlin / Heidelberg, 2007.

[9] A. Guttman. R-trees: A dynamic index structure for spatial searching. In *INTERNATIONAL CONFERENCE ON MANAGEMENT OF DATA*, pages 47–57. ACM, 1984.

[10] Y. Kim and H. Yu. Geotree: Using spatial information for georeferenced video search. Technical report, POSTECH-CSE-2012-03, http://dm.postech.ac.kr/techreport/TechReport-POSTECH-CSE-2012-03-GeoTree.pdf, 2012.

[11] H.-P. Kriegel, M. PÃűtke, and T. Seidl. Interval sequences: An object-relational approach to manage spatial data. In C. Jensen, M. Schneider, B. Seeger, and V. Tsotras, editors, *Advances in Spatial and Temporal Databases*, volume 2121 of *Lecture Notes in Computer Science*, pages 481–501. Springer Berlin / Heidelberg, 2001.

[12] N. Meratnia and R. de By. Spatiotemporal compression techniques for moving point objects. In *Proc. of Int. Conf. on Extending Database Technology (EDBT 2004)*, 2004.

[13] M. F. Mokbel, T. M. Ghanem, and W. G. Aref. Spatio-temporal access methods. *IEEE Data Engineering Bulletin*, 26:40–49, 2003.

[14] D. Pfoser. Indexing the trajectories of moving objects. *IEEE Data Engineering Bulletin*, 25:3–9, 2002.

[15] D. Pfoser, C. S. Jensen, and Y. Theodoridis. Novel approaches to the indexing of moving object trajectories. pages 395–406, 2000.

[16] M. Potamias, K. Patroumpas, and T. Sellis. Sampling trajectory streams with spatiotemporal criteria. In *Scientific and Statistical Database Management, 2006. 18th International Conference on*, pages 275 –284, 0-0 2006.

[17] E. Sahouria and A. Zakhor. A trajectory based video indexing system for street surveillance. In *IEEE Int. Conf. on Image Processing*, pages 24–28, 1999.

[18] S. Saltenis, C. S. Jensen, S. T. Leutenegger, and M. A. Lopez. Indexing the positions of continuously moving objects, 2000.

[19] H. Samet. Spatial data structures. Addison-Wesley, 1995.

[20] K. Shearer, S. Venkatesh, and D. Kieronska. Spatial indexing for video databases. *Journal of Visual Communication and Image Representation*, 7(4):325 – 335, 1996.

[21] K. Toyama, R. Logan, and A. Roseway. Geographic location tags on digital images. In *Proceedings of the eleventh ACM international conference on Multimedia*, MULTIMEDIA '03, pages 156–166, New York, NY, USA, 2003. ACM.