Streaming News Image Summarization

Hao Li
University of Maryland, College Park
haoli@cs.umd.edu

Shangfu Peng
University of Maryland, College Park
shangfu@cs.umd.com

Hanan Samet
University of Maryland, College Park
hjs@cs.umd.edu

***Abstract*—Automatic summarization of streaming news images is critical for efficient news browsing. Although image duplicates are redundant for news reading, the number of duplicates of a news image is a good indicator for its importance. We describe the architecture used in a news aggregation system for online streaming news image summarization. Given a sequence of images for a news topic, we first cluster image duplicates based on a two-stage feature matching process, followed by representative image selection inside each cluster. Images with a high importance score are ranked chronologically to generate a timeline summarization. Our timeline summarization is not limited to a fixed size but enables users to zoom in to see more images with more details based on their interests. Experiments on real-world news data demonstrate that the timelines produced by our method can generate accurate and dynamic timeline summarizations.**

I. Introduction

With the massive news articles published by news media online, news readers are overwhelmed by a large number of news photos, making it hard to get a global picture of a news topic. A visual timeline summarization of a news event is an ideal way to facilitate news reading. The timeline should be generated and updated automatically. Here, we investigate how to select important news photos from multiple news sources and generate a dynamic visual timeline summarization.

The key issue in news image summarization is how to extract a small set of news photos from a large image stream to best summarize the event dynamically. News photos about the same event are often similar due to several factors: 1) Important news events are reported by different news media many times. 2) Different photographers capture the same scene but from different angles. (Figure 1). 3) Different news media use the same "official" photo with modifications (e.g., cropping and rotation). Such actions result in many near-duplicate images.

Although these near-duplicate images are undesirable from the standpoint of news reading (and hence should be filtered), the number of near-duplicates is an implicit indicator of the photo's importance or popularity. Intuitively, the greater the importance of the photo, the higher its frequency and diversity of use. Therefore, the more often a news photo is duplicated, the more important it could be. It is analogous to the act of retweeting or sharing in a social network, which is interpreted as an endorsement of the particular news photo. This leads us to the approach for the streaming news image summarization in a timeline manner.

Given photo streams from multiple news sources, we first cluster exact/near duplicate images and then select the most representative image according to its centrality in each cluster. The importance of an image is modeled by its number of near-duplicates. We rank those images by their timestamp and generate the timeline summarization by optimizing the total information gain. The timeline is dynamically visualized according to a user specified time period. The images shown to the user are the result of maximizing the information gain and minimizing the visual redundancy.

Fig. 1. News photos taken by different media for the same event.

II. Related Work

Previous works on image summarization can be classified into serveral categorizes based on their application scenarios. *Scene image summarization* [1], [2], [3] summarizes large collection of landmark images which are strongly spatially connected for the purpose of scene reconstruction and browsing in 3D space. *Personal photo summarization* [4], [5] aims at selecting high quality and representative images about a trip or an event for a single user. *Image search result summarization* [6] focuses on finding a few representative images that are relevant and diverse. *Collective image summarization* [7], [8] aims at building structured storylines from multiple web image streams about the same topic or event.

News image summarization differs from the above applications in several aspects: 1) News photos are usually not spatially connected but have a with strong temporal correlation. Simple representative image selection works well on queries like celebrities, landmarks and products, but lacks the ability of story telling in a timeline manner. 2) News photos often come with high quality and the number of near-duplicates reflects the collective wisdom of news editors rather than personal preference. The quality of summarization is less subjective-prone. 3) News photos are associated with news articles and captions [9], which provide more semantic information in comparison to web community photos. 4) News photos are updated frequently in a long time peroid in comparison to Flicker and blog posts. The pursuit of freshness requires the summarization to be done online rather than offline.

Another line of related research is multi-document summarization [10], [11] which focuses on cross-modality timeline

generation. A fixed time unit is used for the timeline and the number of images for each unit is limited. We use zoom-able time units and optimize the presented images given the space limit for each time slot.

III. Problem Formulation

Given photo streams from multiple news sources, the incoming images are first clustered into different topics based on corresponding articles, grouping semantically similar and temporally close images into the same topic set [12]. Each news image I is associated with an article a, a timestamp t, a caption d and a topic e. Given a photo stream $\mathcal{I}_e = \{I_1, ..., I_N\}$ on topic e, the goal of summarization is to select a subset of images V to best summarize the topic e in a time interval T that maximizes the information presented to users with temporal and spatial constraints.

The selected news images should have the following properties: 1) *Importance*: The image or its near-duplicate should be widely used by different news media. 2) *Representativeness*: The images should cover the main visual contents among many variations of near-duplicate images.

The timeline should optimize the following properties: 1) *Information Value*: The timeline should present the most valuable information to the user. 2) *Coverage*: The timeline should cover the majority of time slots to keep the completeness of the story. 3) *Diversity*: The selected representative images should cover different perspectives without visual redundancy.

IV. Approach

The pipeline of our approach is illustrated in Figure 2, which consists of the following steps: duplicate image clustering, representative image selection and timeline generation.

A. Online Near-Duplicate Image Clustering

Global features are generally efficient to compute and compare, but have limited robustness against severe cropping and geometric distortions. Local feature matching is more robust to various transformations but is more time consuming for matching. We use a coarse-to-fine strategy to combine the merits of the two methods. Given an incoming image, we use global feature to efficiently match its exact-duplicate image cluster, and then find its near-duplicate image cluster by local feature matching. The clustering result of the two stages is recorded in table $\mathcal{C}$. Each entry of the table contains a posting list of image identifiers and the feature representation of the first arrived image.

Global Feature Clustering We adopt the Hierarchical Color Histogram [12] as the global feature, which is a 512-dimensional feature vector that encodes three levels of a color histogram. The first level is the color histogram of the whole image and for each quadrant from the second level, and finally sixteen histograms from the third level. This structure encodes global features while preserving certain spatial information. Upon receiving a new image I, we first check whether a cluster c exits in $\mathcal{C}$ where its distance to the image is smaller than a threshold. If one or more c exist, add I to it and its corresponding local cluster. Otherwise, a new cluster $c = \{I\}$ is added to $\mathcal{C}$.

Local Feature Clustering When a new cluster c is added to $\mathcal{C}$ in the global feature matching stage, we extract its local features to determine whether it is a near-duplicate of an existing cluster in $\mathcal{C}$. If a matching cluster $c' \in \mathcal{C}$ exists, then add I to c'. Otherwise, add a cluster $c' = I$ to $\mathcal{C}$. We extract SIFT [13] descriptors as local features and perform feature matching with RANSAC [14] verification. Eventually exact and near duplicate images are clustered into the same cluster, with no duplicates existing across clusters. The cluster size $|c|$ is used as the importance/popularity score s.

B. Representative Image Selection

Given a cluster of near-duplicate images c, we choose one image as the representative image for that cluster. We assume that the later images are modified versions of the first arriving image, and thus make its timestamp as the timestamp for the cluster c. However, as illustrated in Figure 3, images added later may have better quality as time evolves. So representative image selection is necessary for better reading experience.

We apply VisualRank [6] to the images of each cluster and choose the image with the highest PageRank value as the representative image for that cluster. VisualRank employs the random walk intuition to rank images by treating visual similarities as probabilistic visual hyperlinks. High quality images will have many other images linking to them and will be visited frequently, which are deemed important. In particular, if image I_u has a visual hyperlink to image I_v, then there is probability that a random walk will jump from I_u to I_v. The transition probability from I_u to I_v is defined as $p(I_u, I_v) = \frac{m(I_u, I_v)}{n(I_u)}$, where $m(I_u, I_v)$ is the number of matched local descriptors and $n(I_u)$ is the number of local descriptors of image I_u.

Given an image I_v with incoming links from other images $I_u \in c$, its weighted PageRank value $PR(I_v)$ is defined as follows:

$$PR(I_v) = \frac{\alpha}{|c|} + (1 - \alpha) \sum_{I_u \in c} PR(I_u) \frac{p(I_u, I_v)}{\sum_{I_x \in c} p(I_u, I_x)}$$

where α is the probability of random jump. The weighted PageRank algorithm assigns different values to other nodes according to the edge weight instead of dividing r evenly among its outlink nodes. The image with highest PageRank value is selected as the representative image. The adjacency graph is updated when a new image is added into the cluster.

C. Dynamic Timeline Generation

To formalize the problem of presenting images on a timeline, we denote each image as a triple $\langle s, t, w \rangle$, representing the popularity score, the timestamp, and the width. The simplest method is to select the top K images for each day while the images are above a threshold ϵ in chronological order:

$$\begin{aligned} &\text{maximize} && \sum_{i \in V_t} s_i \\ &\text{subject to} && |V_t| \leq K \text{ and } \forall s_i > \epsilon \end{aligned}$$

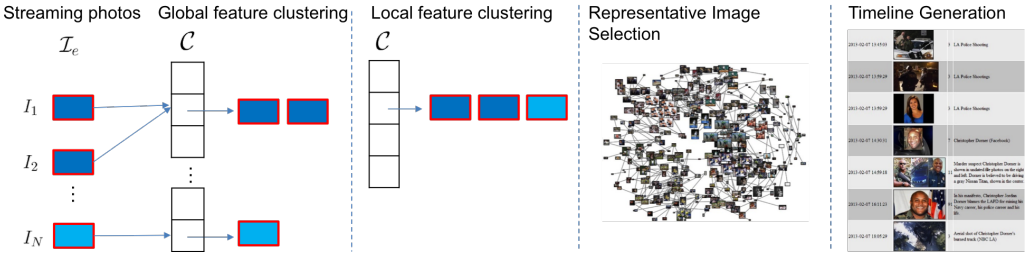


Fig. 2. Pipelines for generating representative summaries of streaming news images.

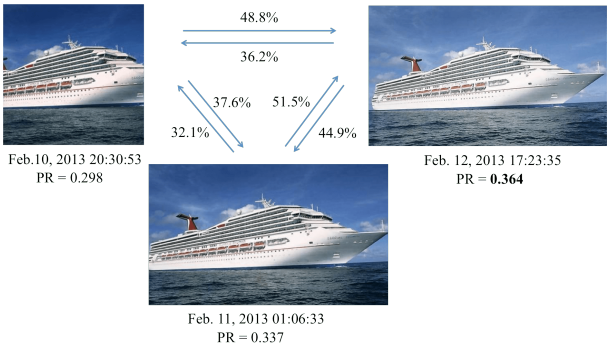


Fig. 3. Similarity graph and PageRank for representative image selection.

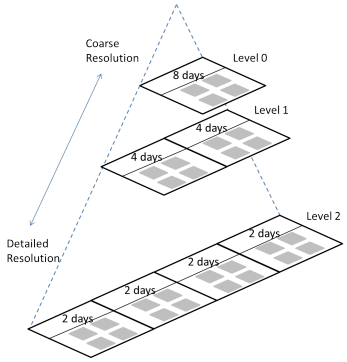
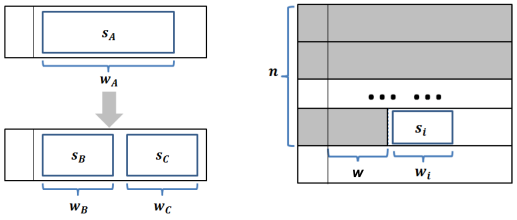


Fig. 4. The timeline with multiple resolutions.



(a) An example where wo images are better than one. (b) State for dynamic programming.

Fig. 5. Dynamic programming design.

where V_t is the visible images set in the tth time slot and V is the overall visible image set, i.e., $\bigcup V_t = V$. The limitation of this formulation is that it is hard to determine the value of K or the importance threshold ϵ. Simply defining a fixed K value may fail to display other important images. Thresholding ϵ may result in too many (few) images and make the timeline too long (short) to view (users may have to slide the mouse a lot to see the whole timeline). Some approaches [10], [11] use a fixed number of images in each time slot, which hides the contrasting importance of different time slots.

We propose a dynamic timeline generation method that enables users to zoom in or zoom out on the timeline to see the detailed or the general summarization result. The interaction is similar to manipulating maps [15], but in one dimension. In particular, we separate the screen window into several time slots (rows). The height of each time slot (row) is fixed. By default, each time slot corresponds to one day and shows the summary of news images for that day. Figure 4 illustrates the resolution for the slots with zoom level increasing. At zoom level 0, each time slot summarizes the images over a period of 8 days. At each zoom level increment, the number of time slots in a certain time period is increased by a factor of 2. That is, the time slot would present the news images over 12 hours if the user takes a zoom-in action in the default setting, 3. The limit is 1.5 hours for each time slot at the maximum zoom level, 7.

Given a fixed screen size, the time range of the window depends on the zoom level and the start time. In order to show the images for a certain time range (we treat this task as a window query), we first gather all representative images in this time range and assign them to the corresponding time slot according to their timestamps. Showing all images in this time range will convey too much information to the user. Here we consider two constraints for presentation. The first is that we set a limit of M visible images in the window, and each time slot must contain at least one image except that there is no image in the time slot. This constraint ignores the fact that the widths of images are different. For example, it is possible that one image with high importance score has a wide width, in which case it may be better to use two narrower images instead of the wider one. So, the second constraint introduces a width limit W for the time slots, where each visible image consumes some width in its time slot. Figure 5(a) gives an example with three images. Here s_A, s_B and s_C are the importance scores for the images where $s_A > s_B > s_C$ but $s_A < s_B + s_C$; w_A, w_B and w_C are the widths, where $w_A > w_B = w_C$, and $w_A + w_B > W$, $w_A + w_C > W$, but $w_B + w_C < W$. In this example, showing B and C together is better than showing A only.

Thus, our goal is to maximize the sum of importance scores of images in a query window with time period $T = [t_s, t_e]$ and

width W. With the two constraints, we arrive at this solution:

$$\hat{V} = \arg\max_{V} \sum_{i \in V} s_i$$

$$\text{subject to } |V| \leq M, \sum_{i \in V_k} w_i \leq W, t_s \leq t_i \leq t_e$$

where V is the selected visible image set. It becomes a $0-1$ knapsack problem if we discretize the continuous widths to their near integers. In particular, each slot is horizontally divided into W cells, e.g., $W = 100$. The w_i value is the integer representing the number of w_i cells occupied by the image i. We solve this $0-1$ knapsack problem by dynamic programming (DP) because of its optimal substructure. The DP state $f(i, n, m, w)$ stores the maximal sum of importance scores, when we have processed the first $(i-1)$ images and $(n-1)$ slots, and have chosen m images to be visible while the n-th slot has utilized w cells. Figure 5(b) illustrates the state.

V. Experiment

We analyze eight months of real-world streaming news images collected by NewsStand [16] in 2013, a system which aggregates news articles from over 10,000 RSS news feeds. Among the 612,319 news clusters only 0.024% have more than 200 images. We select six news event clusters which have at least 300 images (Table I).

TABLE I
Statistics of the Event Clusters

Event	Articles	Images	Timespan
Disabled Cruise Ship	209	325	Feb 10 - Feb 22
Boston Bombing	835	1540	April 15 - May 12
George Zimmerman	868	1374	Jun 3 - Aug 1
SFO Plan crash	397	788	July 6 - July 22
William and Kate	596	1273	July 8 - July 31
California Kidnapping	308	396	Aug 6 - Aug 26

The timestamp for each image corresponds to the time when the article was downloaded, which is within minutes of the news feed updating. Figure 6 shows the number of hourly incoming news images for each event. The temporal patterns of the events are different from each other. Some events have a sudden burst on a particular date, indicating some important events happening at that time. For example, there are two bursts in the "Boston Bombing" event corresponding to the bombing itself and several days later when the suspects were identified. The number of images needing processing reaches a peak at 150 images/hour in the "SFO Plane Crash" event.

A. Evaluating the Clustering Accuracy

To evaluate the performance of the clustering, we create the groundtruth for the six events by manually clustering the near-duplicate images, grouping images if they are modified from the same image or if they depict the same scene with moderate differences. We adopt the following metrics to measure the performance of near-duplicate image clustering: 1) **False Match**: The number of near-duplicate images that are incorrectly grouped together as similar. 2) **Miss Match**: The number of near-duplicate images that are spread across multiple clusters.

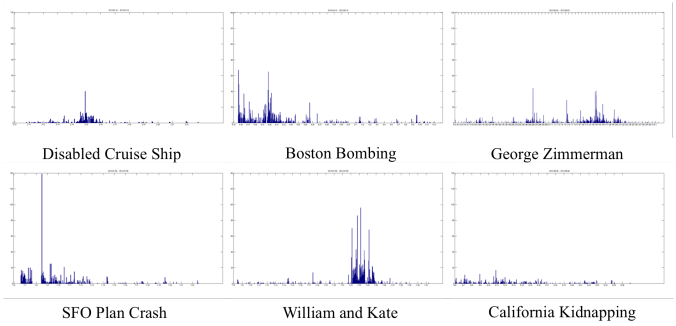


Fig. 6. The number of hourly incoming images for each event.

We compare the two-stage clustering method with each of the single stage clustering methods, i.e., global feature matching and local feature matching. We empirically set the threshold for hierarchical color histogram matching to a high standard to keep its high accuracy. The criterias for SIFT matching are: 1) at least 15 local descriptors are matched and 2) the percentage of matched descriptors compared to the average number of descriptors of two images is greater than 5%. All mages are resized with the maximum width of 240 pixels before feature extraction.

Error rates for near-duplicate image clustering are listed in Table II. Overall, the global feature method has less false matches but more missing matches, while the local feature matching has less missing matches but more false matching. The hybrid approach significantly reduces the false matching rate compared with local feature matching and miss matching rate compared with global feature matching. Since global feature is extracted for each incoming image at the first stage of matching, local feature matching is not necessary if the corresponding cluster event is found by global feature matching. Hence hybrid matching requires less computation.

B. Evaluating the Timeline Generation

The evaluation of an image timeline summarization is a quite subjective task and there are no known metrics to judge the quality of image timeline. Hence we present the summarization result of different methods and analyze them qualitatively. Given a user issued time window query $T = [t_s, t_e]$, we present M images to the user. We compare our dynamic timeline generation method (**Top-M DP**) with the following baseline methods: 1) **Top-M**: M images with highest importance score are shown in the time window. Images are inserted into their corresponding time slots. 2) **Top-M Fill**: First fill each day in the time window with an image if there are images in that timeslot. Next, fill the remaining images into the corresponding slots.

We show a time period of 8 days for all methods. We set $M = 20$, $W = 400$ and the height of each row to be 100 pixels. The generated timeline for the "William and Kate" event are shown in Figure 7. The Top-M strategy will leave empty timeslots on the timeline, while the Top-M Fill strategy

TABLE II
PERFORMANCE OF NEAR-DUPLICATE CLUSTERING

Event	#imgs	$\|\mathcal{C}\|_{GT}$	Dup%	$\max(\|c\|)$	$\|\mathcal{C}\|$			False Match			Miss Match		
					Color	SIFT	Hybrid	Color	SIFT	Hybrid	Color	SIFT	Hybrid
Disabled Cruise	325	175	46.2	44	219	160	170	0	48	15	130	57	45
Boston Bomning	1540	713	53.7	65	963	675	698	4	160	58	661	307	272
George Zimmerman	1374	706	48.6	27	954	608	695	1	253	29	568	255	239
SFO Plane crash	788	305	61.3	25	427	266	301	0	129	23	406	133	144
William and Kate	1273	766	39.8	58	913	726	736	3	91	42	412	212	189
California Kidnapping	396	97	75.5	128	176	91	97	0	24	3	235	54	56

TABLE III
RUNTIME PERFORMANCE

Event	Feature Extraction		Duplicate Clustering		Image Selection		Total	Average	Latency
	Color	SIFT	Stage 1	Stage 2	Graph Building	PageRank			
Disabled Cruise	1.75	22.58	0.15	33.98	11.81	0.040	70.29	0.22	0.23
Boston Bomning	12.44	121.99	8.33	848.55	58.61	0.210	1050.13	0.68	3.00
George Zimmerman	10.59	91.04	4.74	432.21	7.00	0.167	545.75	0.40	0.53
SFO Plane crash	7.62	64.57	0.90	105.89	7.69	0.106	186.77	0.24	0.23
William and Kate	9.03	137.74	5.09	1149.02	11.97	0.139	1313.00	1.03	3.47
California Kidnapping	3.81	40.27	0.13	11.47	20.57	0.108	76.36	0.19	0.22

(a) Top-M (b) Top-M Fill (c) Top-M DP

Fig. 7. Timeline generated by different layout algorithm with $M = 20$, $W = 400$, $T = [07/18, 07/25]$.

has at least one image at each available time slot, indicating that there is a related story happening on that day. However, it does not meet the width constraints and thus making the timeline cannot fit images in the same event. Our dynamic Top-M method selects images that can optimize the sum of the scores while not violating the width constraints, which is useful when the width of time unit is limited.

Our algorithm provides timeline browsing based on users' interests which were not studied under previous approaches [11], [10]. If users are interested in a certain timeslot, they can zoom in to see more details about that time-unit. More images are shown in the timeslots to fulfill that need. More time summarization results can be seen in Figure 8.

C. Runtime Analysis

The experiments were performed on a server (Dual Xeon L5520 2.27 GHz) using a single thread. The runtime of the pipeline for the six events is shown in Table III. SIFT feature extraction and matching are the most time-consuming parts. The average time to process one image ranges from 0.19s for the "Kidnap" event to 1.03s for the "William and Kate" event. Streaming summarization requires processing data over a long time period with constant time costs, the latency of processing an incoming image is an important criteria for streaming data processing. The maximum latency of the six events is 3.47s for the "William and Kate" cluster event which contains 736 clusters. This speed is acceptable since the number of incoming news is much less at the end stage of the event.

(a) Disabled Cruise (b) Boston Bombing (c) SFO Plane Crash

Fig. 8. Timeline summarization with $M = 30$, $W = 600$ and $|T| = 8$

For breaking news, there are usually much few clusters and the latency is quite small compared to the time when many clusters have been created

D. Discussions

As shown in Figure 9, some near-duplicate images are falsely matched by virtue of having a similar background, while they were actually taken at different times or different events. Simply adding a temporal constraint which only matches duplicates in a short time period is not practical since a near-duplicate image may appear in the news much after its initial appearance. This problem should be solved by utilzing fine-gained image features (e.g., face) and text data.

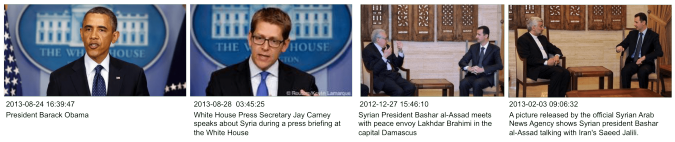


Fig. 9. Falsely matched images.

VI. Conclusions

We described automatic generation of an image-based timeline overview for news browsing. This is in contrast to our prior work on spatial browsing [17], [18] The more valuable an image, the more times it could have exact or near duplicates. On this basis, we applied near-duplicate detection on streaming news images to find the most representative and important images. The dynamically generated timeline summarization provides a clear line of photos for continuously developing news topics enabling quick glimpse of new topics for readers. Experiments on six streaming news image clusters showed that our method can generate accurate interactive timeline summarization. Future work involves devlopment in a distributed sptially-referenced environment such as in a peer-to-peer setting (e.g., [19]).

Acknowledgment

This work was supported in part by the National Science Foundation under Grants IIS-12-19023 and IIS-13-20791.

References

[1] A. Jaffe, M. Naaman, T. Tassa, and M. Davis, "Generating summaries for large collections of geo-referenced photographs," in *WWW*, 2006.
[2] I. Simon, N. Snavely, and S. M. Seitz, "Scene summarization for online image collections," in *ICCV*, 2007.
[3] L. S. Kennedy and M. Naaman, "Generating diverse and representative image search results for landmarks," in *WWW*, 2008.
[4] J. Yang, J. Luo, J. Yu, and T. Huang, "Photo stream alignment and summarization for collaborative photo collection and sharing," *IEEE T-MM*, vol. 14, no. 6, pp. 1642–1651, 2012.
[5] P. Sinha, S. Mehrotra, and R. Jain, "Summarization of personal photologs using multidimensional content and context," in *ICMR*, 2011.
[6] Y. Jing and S. Baluja, "Visualrank: Applying pagerank to large-scale image search," *IEEE T-PAMI*, vol. 30, no. 11, pp. 1877–1890, 2008.
[7] G. Kim and E. Xing, "Reconstructing storyline graphs for image recommendation from web community photos," in *CVPR*, 2014.
[8] G. Kim, S. Moon, and L. Sigal, "Joint photo stream and blog post summarization and exploration," in *CVPR*, 2015.
[9] J. Sankaranarayanan and H. Samet, "Images in news," in *ICPR*, 2010.
[10] R. Yan, X. Wan, M. Lapata, W. X. Zhao, P.-J. Cheng, and X. Li, "Visualizing timelines: Evolutionary summarization via iterative reinforcement between text and image streams," in *ACM CIKM*, 2012.
[11] S. Xu, L. Kong, and Y. Zhang, "A cross-media evolutionary timeline generation framework based on iterative recommendation," in *MM*, 2013.
[12] H. Samet, M. D. Adelfio, B. C. Fruin, M. D. Lieberman, and J. Sankaranarayanan, "Photostand: A map query interface for a database of news photos," *PVLDB*, vol. 6, no. 12, 2013.
[13] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *IJCV*, vol. 60, no. 2, 2004.
[14] M. Brown and D. G. Lowe, "Automatic panoramic image stitching using invariant features," *IJCV*, vol. 74, no. 1, 2007.
[15] S. Peng, H. Samet, and M. D. Adelfio, "Viewing streaming spatially-referenced data at interactive rates," in *SIGSPATIAL*, 2014.
[16] H. Samet, J. Sankaranarayanan, M. D. Lieberman, M. D. Adelfio, B. C. Fruin, J. M. Lotkowski, D. Panozzo, J. Sperling, and B. E. Teitler, "Reading news with maps by exploiting spatial synonyms," *Communications of the ACM*, vol. 57, no. 10, pp. 64–77, 2014.
[17] C. Esperança and H. Samet, "Experience with SAND/Tcl: a scripting tool for spatial databases," *JVLC*, vol. 13, no. 2, pp. 229–255, Apr. 2002.
[18] H. Samet, H. Alborzi, F. Brabec, C. Esperança, G. R. Hjaltason, F. Morgan, and E. Tanin, "Use of the SAND spatial browser for digital government applications," *Comm. ACM*, vol. 46, no. 1, pp. 63–66, Jan. 2003.
[19] E. Tanin, A. Harwood, and H. Samet, "A distributed quadtree index for peer-to-peer settings," in *ICDE*, 2005.