Object Browsing and Searching in A Camera Network using Graph Models

Zefeng Ni[†], Jiejun Xu[⋆], B. S. Manjunath[†]
Department of Electrical and Computer Engineering, University of California, Santa Barbara[†]
Department of Computer Science, University of California, Santa Barbara[⋆]
{zefengni, manj}@ece.ucsb.edu[†], jiejun@cs.ucsb.edu[⋆]

Abstract

This paper proposes a novel system to assist human image analysts to effectively browse and search for objects in a camera network. In contrast to the existing approaches that focus on finding global trajectories across cameras, the proposed approach directly models the relationship among raw camera observations. A graph model is proposed to represent detected/tracked objects, their appearance and spatial-temporal relationships. In order to minimize communication requirements, we assume that raw video is processed at camera nodes independently to compute object identities and trajectories at video rate. However, this would result in unreliable object locations and/or trajectories. The proposed graph structure captures the uncertainty in these camera observations by effectively modeling their global relationships, and enables a human analyst to query, browse and search the data collected from the camera network. A novel graph ranking framework is proposed for the search and retrieval task, and the absorbing random walk algorithm is adapted to retrieve a representative and diverse set of video frames from the cameras in response to a user query. Preliminary results on a wide area camera network are presented. [1]

1. Introduction

Wide area video surveillance requires the use of many cameras. Traditional centralized approaches to video analysis do not scale well as the number of cameras in a network increase. In addition, recent technological advances in imaging, embedded computing and communication, have made it possible to consider decentralized processing. In such a set up, raw videos are analyzed at individual sensor nodes and information exchanged between cameras depending on application needs and priorities. At present, there are no good general strategies in such a network that would facilitate easy interaction between human image analysts and the data collected/analyzed at the remote camera nodes. This work is an attempt to fulfill this critical need.

We consider a fixed camera network deployed over a wide area (see Figure 1 and Figure 3). Raw videos are archived at the remote camera nodes and each camera node has limited processing power for simple video analysis such as motion detection and tracking. Given the bandwidth constraint, there is no live video streaming to the distance central node where human analysts are. We envision the following two application scenarios for the interaction between the user and the camera network:

- **Browsing**(see Figure 5 and 7): A user instantiates the interaction with the network by specifying regions on the image plane (cameras, time intervals) of interest. An example query could be "FIND object instances *related* to region A FROM camera 1 OR region B FROM camera 4 between time 9:30am and 9:35am". For each query, the system needs to provide a "smart summarization" with an overview of network activities that satisfy the query criterion. The reason for the "summarization" is to reduce communication cost when accessing the remote videos and alleviate human efforts when interpreting the query results.
- **Searching**(see Figure 9): With the results from the previous scenario, the user could then identify specific objects of interest to initiate further searching for the same or related objects. An example query here could be "FIND all objects *related* to the object instance at region C FROM camera 1 at time 9:32:41.3am". A special case for this query scenario is the classic problem of object re-identification, i.e. find the instances of the same object in all camera views.

Designing a system to address such queries is an interesting and challenging problem, and is the primary motivation for the proposed work. One way to address this problem is to provide human users with an high-level interface, such as dynamic global scene visualization [1, 6]. To achieve this, prior research has focused on methods that can detect and track all observed objects across the entire camera net-

[1]This work was supported by ONR grant # N00014-10-1-0478

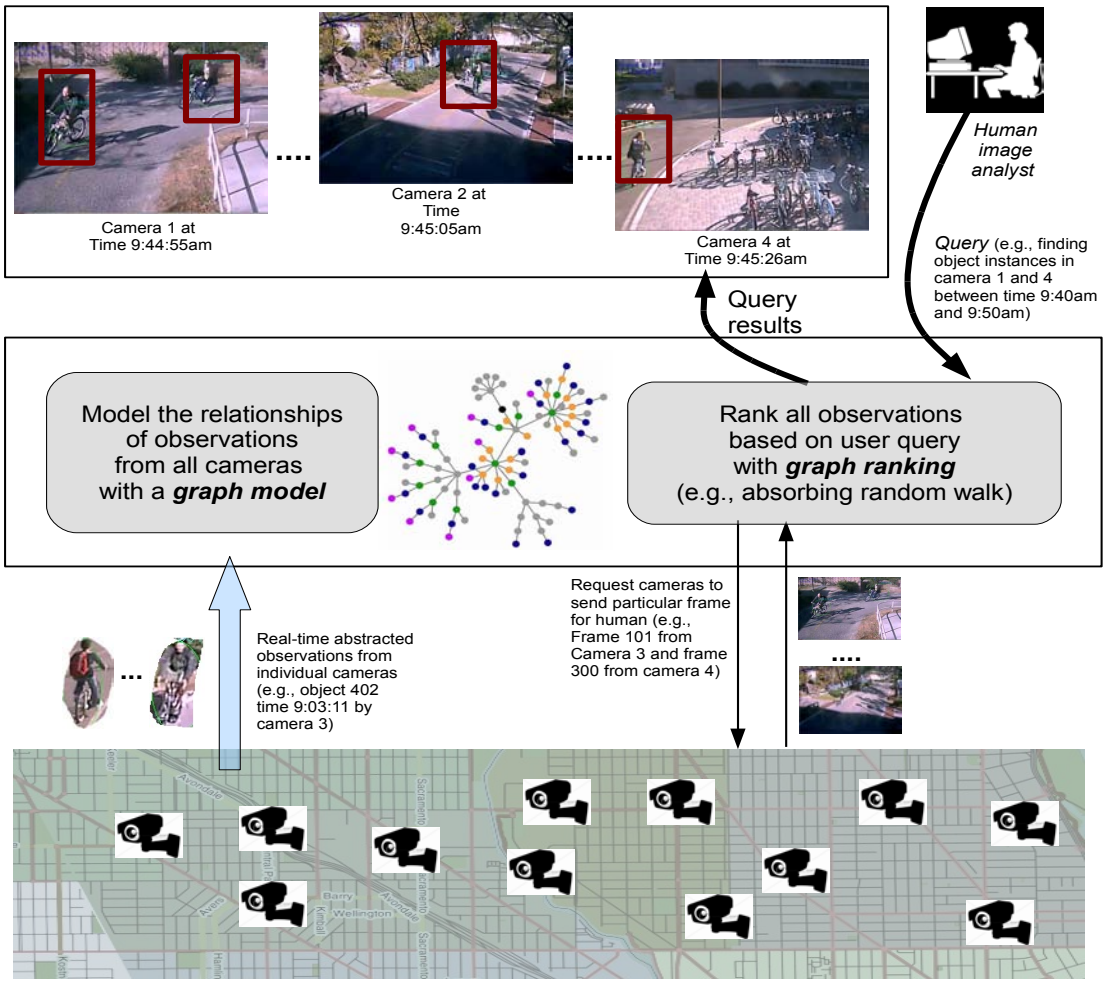

Figure 1. The proposed system to facilitate human image analysts to efficiently browse and search objects in a camera network

work [7, 8, 10]. While this could be an ultimate goal for an ideal surveillance system, it is a difficult task to achieve with existing state of the art in computer vision. For example, with limited on-board processing and low-quality image sensors at remote camera nodes, it would be difficult to detect and track objects in a consistent manner.

In this work, instead of trying to find global trajectories for every object visible in the network, we propose to compute *representative* raw video frames (snap shots) from individual cameras. These frames are likely to contain events or objects of interest (see Figure 1) requested in the user query. In particular, the proposed system acts as an intermediate agent between distributed camera nodes and human image analysts, and provides recommendations to the user with a concise and authoritative set of frames captured by the camera network. The goal is to help the image analysts to browse, search and identify objects of interest by giving a canonical overview of the entire set of visual observations in the network.

The proposed system contains two essential parts: 1) real-time object detection and tracking at the remote camera nodes; and 2) modeling of relationships among camera observations with a graph at a central node. The key contributions of this paper include constructing a time-evolving graph based on remote camera observations and serving various user queries by ranking graph nodes (raw video frames with observations) and recommending the high-ranked frames to user. To prevent redundant items from receiving high ranks, we utilize the absorbing random walk [18] to improve searching diversity and present the user with a diverse and representative visual summarization. To the best of our knowledge, this proposed system is the first attempt to allow user interaction with distributed camera network by utilizing graph modeling and ranking to facilitate effective object browsing and searching. This system is successfully demonstrated with an 11-camera outdoor network.

The rest of the paper is organized as follows. Section 2 describes related works on object re-identification and graph-based method for visual searching. Section 3

describes the methodology in details. Finally, Section 4 presents experimental results on a real 11-camera outdoor camera network and Section 5 concludes this paper.

2. Related Work

The proposed system is related to the problem of object reacquisition or re-identification in multiple cameras. In [1, 6], similar systems with distributed cameras are proposed, with a server collecting camera observations and assigning unique global object ID based on object's estimated location and/or color appearance. To deal with appearance variations across views, much work has been done on finding the best matching criterion, for example the joint motion and appearance model in [7], low-dimension subspace learning of brightness transfer functions in [8], symmetry-driven accumulation of local features in [5], probabilistic relative distance comparison in [17], and the shared set of haar-feature in [14]. All these methods share one common property, that is the pair-wise comparison of measurements from different camera views. This way of direct comparison might suffer when the measurements (object detection and tracking) from the individual cameras are noisy. A more effective way of relating observations from different cameras is to treat them collectively, instead of doing pair-wise similarity comparison, such as the method in [10], which finds optimum paths (maximum a posterior estimates) over all camera observations. However, their proposed solution of linear programming still requires the perfect detection and tracking from individual cameras. In this paper, we propose to utilize a graph to represent the underlying relationships among camera observations and cast the problem of user interaction as an unified graph ranking problem by identifying *representative* snap shots that could contain the observations requested by the user query.

The proposed system adapts concepts from content-based image retrieval (CBIR), especially graph modeling in large-scale image databases. In [9] a visual ranking algorithm was proposed to apply graph-based PageRank for image search. However, visual features alone might not be sufficient to convey the semantics in the images. Researchers in the CBIR community have tried to exploit multiple information cues to alleviate this problem. For example, a graph framework was utilized in [16] to fuse information from multiple sources (e.g. image feature and text annotation). The utilization of graph model introduces structures to the data to capture their global inter-relationship and exploit the mutual reinforcement among different modalities. Similar ideas were used in [12, 15]. In summary, graph modeling has been proven to be an efficient method to combine multiple cues, especially for large databases.

3. Proposed Method

Figure 1 gives an overview for the proposed system. Assuming a network of N distributed static cameras with embedded storage and computing power, each camera node independently detects and tracks moving objects in real time. For each frame with detected objects, the camera sends an abstracted record, including object's spatial, temporal and appearance information, to a central node. At the central node, a time-evolving graph is incrementally built to model the relationships among the camera observations based on the received records.

Given a user's query, e.g., "FIND observations related to region A of camera 2 between time t_1 and t_2", the central node performs ranking on the graph to identify a representative and concise set of frames and then requests the remote cameras to deliver the corresponding snap shots over the network. In this way, the system avoids the need of any real-time video streaming, which could be prohibitively expensive.

3.1. Real-time Distributed Detection and Tracking

At each camera node, the system detects interesting objects and tracks them on the image plane. Assuming a static camera network, objects can be detected by modeling background and identifying moving foreground. In particular, foreground pixels are identified using background subtraction [11]. Connected foreground pixels are combined together to form foreground blobs, which are then tracked by a mean shift algorithm [3]. In the current set up, each object is represented with a rectangular blob. To address the problem of scale variations, we utilize the the general mean-shift blob tracking algorithm proposed in [2].

For each tracked object, a unique object ID is assigned [2]. For each frame processed by the camera, a record is generated for each detected/tracked object and sent to the central node over the network. Each observation record includes information such as camera ID, time, object's position on the image plane and a 16-bin Hue histogram as appearance representation.

3.2. Modeling Camera Observations with a Graph

Given a user query, the system aims to find the observations with the following two properties, centrality (i.e., representative ones which are closely related to the query and many other observations and hence considered important) and diversity (i.e., covering as many distinct groups as possible). In a browsing scenario, there is no live video for the user to monitor in real time. The system should provide a smart "summarization" from all the cameras. A frame with more detected objects is considered more important to

[2] Unique for the particular camera but not across cameras, therefore no cross camera collaboration/communication is required.

the human analysts. Similarly, an object observed by multiple cameras over a longer duration is more important than an object just appearing in a single camera. On the other hand, in a searching scenario, a user might be interested in a particular object(s). Instead of showing near-identical observations, it would be more interesting to display the observations with different properties, e.g., different visual appearance and from different cameras etc.

With a graph framework, we can easily address the above requirements utilizing effective graph ranking algorithms, e.g., absorbing random walks [18] and manifold ranking with stopping points [19]. Further more, the graph framework presents a principled formulation to answer different queries. In such a graph framework, individual camera observations (i.e., frames with detected objects) form the vertices V in a simple graph $G(V, \mathbf{W})$. The weight matrix $\mathbf{W}$ defines the strength of connectivity between camera observations (e.g., the same object at different views). Note that the graph G is built at the central server incrementally as the new records are received in real time from the cameras.

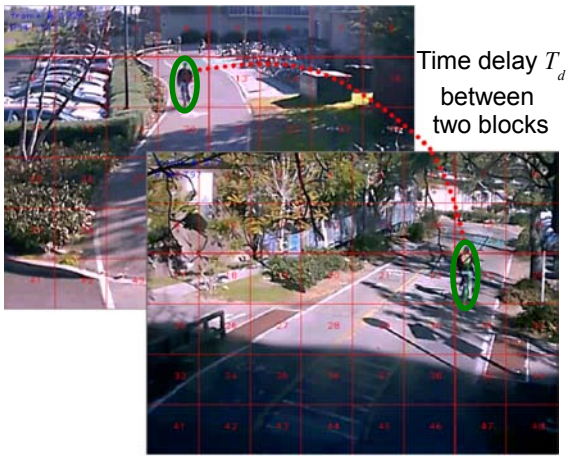

Figure 2. Spatial-temporal topology across cameras

In our current implementation, $\mathbf{W}$ is estimated according to objects' visual appearance and spatial-temporal topology in the camera network. Given two vertices X_i and X_j, their edge weight $\mathbf{W}_{ij}$ is calculated using Algorithm 1. If the two vertices are from the same camera, $\mathbf{W}_{ij}$ is set to $k\omega$ where k is the number of common objects (records with same local object ID) in X_i and X_j and ω here is a constant. If frame X_i and X_j are from different cameras, we first check whether it is likely that the two observations are caused by the same object based on the network's spatial-temporal topology (see Figure 2).

To model the spatial-temporal topology across cameras, the image plane from each camera view is divided into 8x6 blocks. We assume the time delay T_d for an object to travel across any two blocks follows a Gaussian distribution with known mean μ and variance δ^2. With this topology model and two observation records R^i and R^j, from X_i and X_j respectively, we can calculate $P^{ST}(R^i, R^j)$, the likelihood that R^i and R^j belongs to the same object, based on the time delay between $Block(R^i)$ and $Block(R^j)$. If $P^{ST}(R^i, R^j)$ is larger than a threshold [3], the weight $\mathbf{W}_{ij}$ is increased by $p^A(R^i, R^j) * p^{ST}(R^i, R^j)$, where $p^A(R^i, R^j)$ models two record's similarity in visual appearance (correlation between the Hue-histograms of record R^i and R^j).

Algorithm 1 Weight computation between two vertices

Input: Two vertices X_i and X_j in the graph
Output: Edge weight between $\mathbf{W}_{ij}$ between X_i and X_j

```
Initialization: W_ij = 0, i.e., no connection.
for Each each object record R^i in X_i do
    for Each each object record R^j in X_j do
        if R^i and R^j are from the same camera and share the same object ID then
            W_ij = W_ij + ω
        else
            if R^i and R^j are from different cameras and p^ST(R^i, R^j) > Threshold then
                W_ij = W_ij + p^A(R^i, R^j) * p^ST(R^i, R^j)
            end if
        end if
    end for
end for
```

3.3. Query Serving with Graph-based Ranking

With the graph modeling of camera observations, we can utilize off-the-shelf graph ranking methods to answer different user queries. Among them, VisualRank [9] is probably the most related to our scenario. Essentially a similarity graph is constructed based on image visual similarity, and the PageRank algorithm [13] is applied to re-rank the initial text-based searching results. However PageRank does not ensure diversity at all, i.e. if two images are both very similar to many other images, they will have similar (high) ranks. Thus redundant information is being kept. In order to deliver more diverse ranking results, recently methods such as absorbing random walks [18], decayed DivRank [4], and manifold ranking with stop points [19] have been proposed. These methods perform quite similarly, this paper adapts the absorbing random walk approach since it is easy to implement. The main idea is to let a high ranked node to transform into an "absorbing" state during the random walk on the graph. This node will then "drag down" the importance value of other similar unranked nodes, thus encouraging diversity. The algorithm consists

[3] This threshold helps to remove edges with negligible weights, which simplifies the graph model and improves ranking speed significantly with little effect on the final results.

of two parts. The first part is to find the overall top ranked node. Assuming an $n \times n$ weight/affinity matrix $\mathbf{W}$, a raw transition matrix $\widetilde{\mathbf{P}}$ is defined by row-normalizing $\mathbf{W}$, i.e. $\widetilde{\mathbf{P}}_{ij} = \mathbf{W}_{i,j} / \sum_{k=1}^{n} \mathbf{W}_{ik}$, such that $\widetilde{\mathbf{P}}_{ij}$ is the probability that the random walker moves from vertex i to j. Then a teleporting random walk $\mathbf{P}$ is defined by adding each row with the user-supplied initial preference vector $\mathbf{r}$,

$$\mathbf{P} = \lambda \widetilde{\mathbf{P}} + (1 - \lambda)\mathbf{e}\mathbf{r}^{\mathrm{T}}, \tag{1}$$

where $\mathbf{e}$ is an all-1 vector. The $\mathbf{r}$ is determined accordingly to the particular query scenario. The final ranking vector π is the stationary distribution of the random walk, i.e., the solution for equation $\pi = \mathbf{P}^{\mathrm{T}}\pi$. The vertex with the largest stationary probability is the overall top ranked observation, i.e.,$g_1 = \mathrm{argmax}_{i=1}^{n} \pi_i$

The second part of the absorbing random walk is a series of ranking iterations to pick the remaining vertices in the graph. Suppose a group of top-ranked vertices $\mathcal{G} = \{g_i\}$ have been selected, they are turned into absorbing states by setting $\mathbf{P}_{gg} = 1$ and $\mathbf{P}_{gi} = 0, \forall i \neq g$, which is essentially adding a self-edge to those vertices and making them into sinking/stopping states. If we arrange vertices such that ranked ones are listed before unranked ones, we modify transition matrix $\mathbf{P}$ to

$$\mathbf{P} = \begin{bmatrix} \mathbf{I}_{\mathcal{G}} & \mathbf{0} \\ \mathbf{R} & \mathbf{Q} \end{bmatrix}. \tag{2}$$

Here $\mathbf{I}_{\mathcal{G}}$ is the identify matrix on $\mathcal{G}$. Submatrices $\mathbf{R}$ and $\mathbf{Q}$ correspond to the rows of unranked items from (1).

Based on the above matrix, we can compute the expected number of visits to each remaining nodes before reaching any absorption by $v = (\mathbf{N}^{\mathrm{T}}\mathbf{e})/(n - |\mathcal{G}|)$, where $\mathbf{N}$ is known as the fundamental matrix: $(\mathbf{I} - \mathbf{Q})^{-1}$. Again, we can select the vertex with the largest expected number of visits as the next item in ranking: $g_{|\mathcal{G}|+1} = \mathrm{argmax}_{|\mathcal{G}|+1}^{n} v_i$. The main steps to compute the diverse ranked list is summarized in **Algorithm** 2.

Algorithm 2 Serve user query by ranking camera observations with absorbing random walk

Input: Graph weight matrix $\mathbf{W}$ and preference vector $\mathbf{r}$.
Output: Top-ranked vertices $\{g_1, g_2, g_3, \ldots\}$.

1: Compute the initial transition matrix P from (1).
2: Compute stationary distribution π.
3: Pick the top ranked item $g_1 = \mathrm{argmax}_i \pi_i$.
4: **while** Need to look for enough high ranked vertices **do**
5: Convert ranked vertices into absorbing states (2).
6: Compute the expected number of visits for all remaining vertices before reaching any absorption.
7: Pick the next vertex $g_{|G|+1} = \mathrm{argmax}_i \pi_i$.
8: **end while**

The preference vector $\mathbf{r}$ in (1) is a n-dim vector representing the user query. The entries of the vector are mostly zeros, except for the ones that correspond directly to the vertices (i.e., camera observations) carrying initial query intention. For instance, suppose an image analyst is interested in objects related to "region B of camera c between time t_1 and t_2". The system will first identify all frames with records that match this criteria and then mark the corresponding m vertices $\{\mathcal{G}_q\}$ as the query vertices. Then, a uniform score is given the vertices in this query set $\{\mathcal{G}_q\}$, i.e., $\mathbf{r}_i = 1/m$ if $i \in \{\mathcal{G}_q\}$, and $\mathbf{r}_i = 0$ otherwise. Here we can consider $\mathbf{r}$ as an initial ranking vector that kick-starts the absorbing random walk. For the searching query, it is more straightforward. For example, to search for a particular object instance at time t of camera 3, which corresponds the vertex j in the graph, preference vector is set as $\mathbf{r}_j = 1$ with all other entries as 0.

4. Experiments

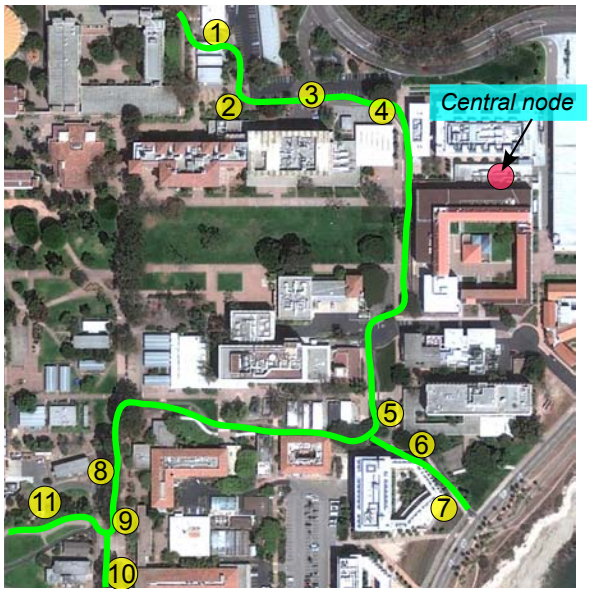

Figure 3. Experimental setup: an outdoor network with 11 camera nodes observing bike paths (shown in green line, the area is approximately 600 meters in width and length).

To demonstrate the proposed system, an outdoor network of 11 cameras is deployed along bike paths in an urban environment (see Figure 3). In particular, our "smart camera node" consists of two parts: a Cisco WVC2300 wireless-G Internet video camera and a nearby dedicated computer. The local computer achieves and processes the live streamed video (640x480, about 20fps) from the Cisco camera. The computer and the wireless camera together simulate a distributed smart camera node in a camera network. These "smart camera nodes" communicate with a distance central server node, where the human user locates.

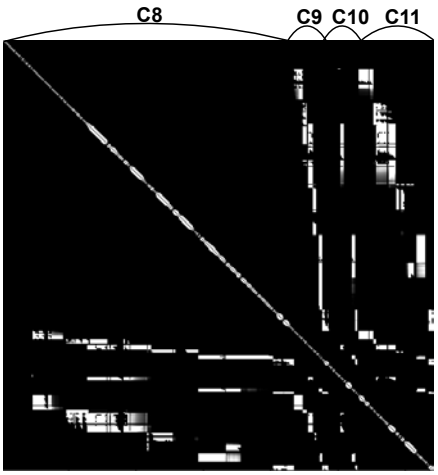

Figure 4. Example graph weight matrix $\mathbf{W}$ with observations from camera 8-11 in 60 seconds. Brightness indicates edge weight.

Figure 5. Browsing scenario 1 with regions of interest indicated by the rectangles in camera C8 and C9.

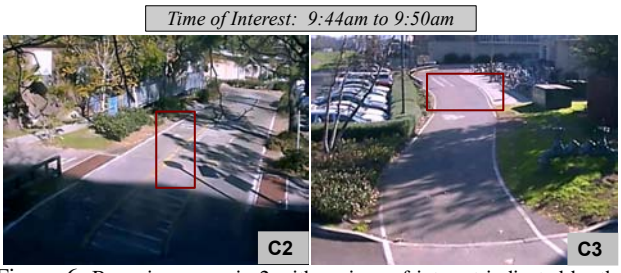

Figure 6. Browsing scenario 2 with regions of interest indicated by the rectangles in camera C2 and C3.

It is a challenging task to reliably detect and track cyclists and pedestrians observed in the scene, especially with low quality video sent from the wireless cameras. Hence this experimental set up serves as a good test bed for the proposed system.

Due to the nature of the problem, there is no off-the-shelf metric to perform large-scale quantitative evaluation. In this paper, we demonstrate the effectiveness of the system with a few application scenarios. Figure 5 and 6 show two "browsing" examples and Figure 7 and 8 show the corresponding top ranked camera observations. The system allows regions of interest from any set of cameras, which simply specify the preference vector $\mathbf{r}$ for the absorbing random walk algorithm. The anticipated result is that top ranked frames should contain majority of the objects (diversity) related to the browsing query (i.e., the object has passed through the regions of interest of all the queried cameras within the specified time). This does not mean the returned frames must be from these regions of interests, as other frames might contain the same objects with more information and hence more representative (centrality). The following table shows the error (wrong objects) and recall (number of matched objects which have been identified) for the two scenarios.

	Browsing 1 (Top 10)	Browsing 2 (Top 15)
Recall	10 out of 10 objects	17 out of 22 objects
Error	1 out 10 frames	0 out 15 frames

With results from browsing, a user can further initiate searching of a particular object instance. Figure 9 shows the searching result when querying with the 5th ranked frame of Figure 7 (starred). Figure 10 shows the results when searching for the 11th ranked frame of Figure 8 (starred). For both cases, the top ranked frames contain a diverse set of objects that is spatially or temporally close to the query object. Collectively, these frames tell a summarized "story" for the object of interest. The red cyclist in Figure 9 travels along the bike path alone all the time, thus the system returns snapshots of him passing different camera views. In addition, the top ranked frames contain other cyclists who are temporally nearby. In Figure 10, the pink cyclist travels along the bike path while occasionally passed by other cyclists. As a result, the system finds those frames which contain other cyclists which are spatially nearby.

5. Discussions

This paper proposes a novel system to assist human image analysts to effectively browse and search for objects in a large distributed camera network for visual surveillance. In particular, the proposed approach directly models the relationship among raw camera observations with a graph. All frames with detected/tracked objects are treated as vertices in a graph, with edges determined by spatial-temporal topology and visual appearance. With the proposed approach, reliable detection and tracking from local cameras is not required, as there is no need for cross camera object association. The graph structure naturally captures the global relationship of camera observations, and enables the system to answer various human queries through a unified ranking framework. The system utilizes absorbing random walk algorithm to retrieve a representative and diverse set of video frames based on the human queries. The effectiveness of the system is demonstrated with a 11-node outdoor camera network. For future work, we would like to utilize the similar graph model here for other applications such as event recognitions by clustering graph vertices. In addi-

Figure 7. Top 10 ranked frames for browsing scenario 1 (Decreasing order: left to right, top to down. The green ellipses are the blobs detected by the remote camera nodes). There are a total of 10 distinct objects satisfying the criterion in Figure 5. All of them have been identified (labeled in yellow). The 8th ranked frame is a "false positive" (it has not passed the queried regions within the specified time interval).

Figure 8. Top 15 ranked frames for browsing scenario 2 (Decreasing order: left to right, top to down. The green ellipses are the blobs detected by the remote camera nodes). There are a total of 22 distinct objects satisfying the criterion in Figure 6. 17 of them have been identified (labeled in yellow).

tion, we plan to prepare manually labeled object trajectories to facilitate large-scale quantitative performance evaluation. The presented data set in this paper will also be released to the research community in the near future.

References

[1] R. Collins, A. Lipton, H. Fujiyoshi, and T. Kanade. Algorithms for cooperative multisensor surveillance. *Proceedings of the IEEE*, 89(10):1456–1477, 2001. 1, 3

[2] R. T. Collins. Mean-shift blob tracking through scale space. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2003. 3

[3] D. Comaniciu and P. Meer. Mean shift: a robust approach toward feature space analysis. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(4):603–619, 2002. 3

[4] P. Du, J. Guo, and X.-Q. Cheng. Decayed divrank : Capturing relevance, diversity and prestige in information networks. In *International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 1239–1240, 2011. 4

[5] M. Farenzena, L. Bazzani, A. Perina, V. Murino, and M. Cristani. Person re-identification by symmetry-driven accumulation of local features. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 2360–2367, 2010. 3

[6] O. Javed, Z. Rasheed, O. Alatas, and M. Shah. Knightˆm: A real time surveillance system for multiple overlapping and non-overlapping cameras. In *IEEE Conference on Multi media and Expo*, pages 6–9, 2003. 1, 3

Figure 9. Results when searching for object P6 from browsing scenario 1 (the starred frame in Figure 7 as the query frame; 2nd to 11th ranked frames are displayed here. Top ranked frame is omitted as it is the same as the query frame.

Figure 10. Results when searching for object P6 from browsing scenario 2 (the starred frame in Figure 8 as the query frame; 2nd to 11th ranked frames are displayed here. Top ranked frame is omitted as it is the same as the query frame).

[7] O. Javed, Z. Rasheed, K. Shafique, and M. Shah. Tracking across multiple cameras with disjoint views. In *International Conference on Computer Vision*, pages 952–960. IEEE Computer Society, 2003. 2, 3

[8] O. Javed, K. Shafique, and M. Shah. Appearance modeling for tracking in multiple non-overlapping cameras. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 26–33, 2005. 2, 3

[9] Y. Jing and S. Baluja. Visualrank: Applying pagerank to large-scale image search. *IEEE Transaction Pattern Analysis and Machine Intelligence*, 30(11):1877–1890, 2008. 3, 4

[10] V. Kettnaker and R. Zabih. Bayesian multi-camera surveillance. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 252–259, 1999. 2, 3

[11] L. Li, W. Huang, I. Y. H. Gu, and Q. Tian. An improved adaptive background mixture model for real-time tracking with shadow detection. In *ACM International Conference on Multimedia*, pages 2–10, 2003. 3

[12] E. Moxley, T. Mei, and B. S. Manjunath. Video annotation through search and graph reinforcement mining. *IEEE Transactions on Multimedia*, 12(3):184–193, 2010. 3

[13] L. Page, S. Brin, R. Motwani, and T. Winograd. The pagerank citation ranking: Bringing order to the web. Technical Report 1999-66, Stanford InfoLab, November 1999. 4

[14] R. Rios Cabrera, T. Tuytelaars, and L. Van Gool. Efficient multi-camera detection, tracking, and identification using a shared set of haar-features. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 65–71, 2011. 3

[15] H.-K. Tan and C.-W. Ngo. Fusing heterogeneous modalities for video and image re-ranking. In *Proceedings of the 1st ACM International Conference on Multimedia Retrieval*, pages 1–8, 2011. 3

[16] H. Tong, J. He, M. Li, C. Zhang, and W.-Y. Ma. Graph based multi-modality learning. In *ACM International Conference on Multimedia*, 2005. 3

[17] W.-S. Zheng, S. Gong, and T. Xiang. Person re-identification by probabilistic relative distance comparison. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 649–656, 2011. 3

[18] X. Zhu, A. B. Goldberg, J. Van Gael, and V. G. Andrzejewski. Improving diversity in ranking using absorbing random walks. In *The Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 97–104, 2007. 2, 4

[19] X. Zhu, J. Guo, X. Cheng, P. Du, and H.-W. Shen. A unified framework for recommending diverse and relevant queries. In *International conference on World Wide Web*, pages 37–46, 2011. 4