Ensuring Spatio-Temporal Consistency in Distributed Networks of Smart Cameras

Dilip Sundarraj
Department of Computer Science
Portland State University
dilip@cs.pdx.edu

Phillip B. Gibbons
Intel Research Pittsburgh
Pittsburgh, Pennsylvania
phillip.b.gibbons@intel.com

Padmanabhan S. Pillai
Intel Research Pittsburgh
Pittsburgh, Pennsylvania
padmanabhan.s.pillai@intel.com

Abstract

Spatio-temporal consistency is essential to ensure validity in coordinated replies generated from a wide area distributed sensor system. Consistency requirements are particularly stringent for analyzing composite images from a network of smart cameras, where even modest inconsistency in either space or time often results in misinterpretations of the composite scene. In this paper, we describe techniques to ensure spatio-temporally consistent analysis of composite images from distributed camera networks. We have incorporated our techniques into IrisNet, a prototype wide-area distributed system for cameras and other sensors. We demonstrate how our techniques apply in the context of two real-world applications, nearshore oceanography and parking space finder, which require spatio-temporal consistency.

Keywords: Spatial Consistency, Temporal Consistency, Image Stitching, Distributed Smart Cameras, Automatic Registration

1 Introduction

Wide-area sensor networks often involve remote deployments of sensors. Collective responses are generated from such a system based on coordinated data acquisition from the sensor nodes. Ensuring spatial and temporal consistency in this coordinated data collection is a challenging task in such systems. Spatial inconsistencies occur due to mismatches in the spatial alignment of the constituent sensor nodes, e.g., misalignments among a set of cameras. Temporal inconsistencies can arise due to asynchronous operation, irregular data collection intervals at the sensors, and transient disconnections due to unreliable last links to the sensor nodes.

For scalar sensor nodes, data inconsistencies can often be compensated by using Bayesian estimation techniques [9], which use predicted values based on a priori information about the sensor data values. However, this kind of estimation is unsuitable for cameras because of the complexity of the sensed data (images). Moreover, complex phenomena are often unpredictable, e.g., when monitoring the formation and progression of rip currents. Established techniques in rip current forecasting [1] provide only daily updates, while the more recent information is collected only after confirmation from the lifeguards stationed at the beach.

Sensor networks applications such as rip current tracking need views from several cameras that are spatially correlated, meaning they should have adjacent fields of views with a non-zero overlap. Moreover, in order to track the rip currents consistently with respect to time, the images collected from multiple cameras must be temporally correlated. In general, distributed camera networks pose stringent consistency requirements, in order to accurately analyze composite images collected from multiple cameras.

In this paper, we describe techniques to achieve spatial and temporal consistency in a distributed network of cameras. These techniques enable fast, on-demand composite analysis of current and recent images from multiple cameras, even in challenging settings involving dynamic scenes, pan-tilt-zoom cameras, and irregular image collection intervals. A key algorithmic result is an efficient algorithm for determining the most recent set of images that satisfies a temporal consistency tolerance. We have implemented our techniques in IrisNet [11], a prototype wide-area distributed system for cameras and other sensors, and report on two case studies, nearshore oceanography and parking space finder, requiring spatial and temporal consistency.

The rest of the paper is organized as follows. In Section 2, we describe IrisNet and our two case study applications. In Section 3 we describe our techniques for ensuring spatial and temporal consistency. Section 4 presents related work and Section 5 presents our conclusions.

2 System and Motivating Applications

Our goal is to enable fast, on-demand analysis tasks over the composite (i.e., unified panoramic) view from images collected by multiple, dispersed cameras that overlook portions of a large scene. We assume that the cameras have partially overlapping fields of view, which enable us to generate a smooth merged image from the many discrete views.

As a base system for our techniques, we use IrisNet [11, 7], a distributed, wide-area sensing prototype we developed. IrisNet has a two-tier architecture, with the upper layer providing an Internet-scale, hierarchically-organized, distributed database for efficient querying of data collected from the lower layer consisting of wide-area rich media sensors. IrisNet is intended to make the development and deployment of wide-area sensing applications easier, by taking care of the many complexities of distributed system implementations, and providing many optimizations, such as routing queries to only the relevant portions of the distributed database. IrisNet enables application-specific code to run on

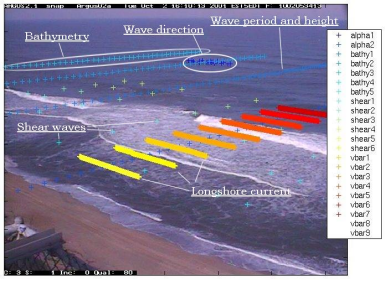

Figure 1. Virtual "Pixel Stack Instruments" used by oceanographers to study various nearshore phenomena *[Source: The Argus Project]* [2]

the sensor nodes in the lower tier, so that high-bandwidth sensor feeds can be converted to smaller data products at the leaves of the hierarchy. IrisNet also performs in-network fusion (e.g., aggregation), enabling sensor data and/or data products from the leaves to be combined at their common parent, and so on up the hierarchy.

We consider two deployment scenarios where unified images are useful to address real-world applications and needs. First, we look at a prototype deployment of high resolution cameras along the Newport coast in Oregon. These cameras have been deployed to capture images of nearshore phenomenon for the ocean near Agate Beach. Oceanographers at Oregon State [2] use these images to understand nearshore phenomenon such as creation/erosion of sandbars, bathymetry, longshore current and wave height/direction (Figure 1). In addition to the scientific merit of the images collected from these cameras, the images can also be used to track the progression of rip currents. In the United States alone, rip currents were responsible for over 22,000 lifeguard rescues and 9 drowning deaths in 2005 [3]. We believe that these numbers can be reduced if frequent updates are provided in a publicly accessible way (such as on the web) about the presence and progression of rip currents along beaches. Deploying a large number of cameras that collectively cover the vast coastal regions, combined with an automated querying system that can retrieve and analyze rectified, virtual overhead images (merging data from multiple cameras when necessary), would provide effective, universal tracking of rip currents.

A second, currently hypothetical deployment involves repurposing images from an urban surveillance and security infrastructure to provide novel services, such as a parking space finder.[1] Here, images from multiple traffic and security cameras that happen to overlook portions of a parking lot are combined and fed into a vision-based detection algorithm that determines the number of free spaces remaining. This distilled information is inserted into a database that can be accessed through the finder service, directing drivers to available spaces near their desired destinations. This scenario presents several interesting challenges, because neither the orientation of the cameras nor the acquisition of images is under direct control.

[1] In our test parking space finder deployment, we positioned cameras looking out our office windows.

3 Design

Motivated by the example applications described above, in this section we describe techniques for generating merged, temporally and spatially consistent images from multiple, disparate cameras.

3.1 Spatial Consistency via Landmarks

Imagery for oceanographic and other scientific monitoring generally requires precise knowledge of the placement and orientation of the cameras. Before the cameras are deployed along the coast, the oceanographers survey the area in order to identify the best possible field of view for each camera for the best overall coverage. During this survey, they obtain the world coordinates, e.g., through differential GPS, of certain landmark points in the area and map those points to the appropriate pixel locations in the images from the cameras. These landmark points help describe the image captured by the camera and provide reference points with known world coordinates. Over time, the cameras may become displaced due to wind, and the pixel locations will no longer correspond to the surveyed world coordinates. In this case the oceanographers carefully measure and record the displacement of landmark points from their original pixel locations to keep the calibration up to date.

For our implementation, these carefully surveyed and calibrated landmark points facilitate the stitching together of multiple camera views while preserving spatial consistency. Given at least 4 landmark points, roughly at sea level, in a single camera's view, we can compute the 2-D image-to-world *homography*, or matrix that transforms pixel coordinates of the camera image to world coordinates [12]. Note that such homographies can only transform one set of planar coordinates to another. Fortunately, given cameras that do not significantly distort the image and the limited range a single camera's view, both the surface of the ocean and global latitude and longitude are sufficiently close to planar for homographies to work well. Given an image from camera a with image-to-world homography H_a, and another from camera b with image-to-world homography H_b, we can produce a spatially consistent stitched image in say b's image coordinates, by applying the transform $H_b^{-1}H_a$ to the pixels of a and then stitching the result with the pixels of b.

In addition to stitching two images together, the homographies can be used to produce a virtual overhead (rectified) view by projecting all of the images into world coordinates, and rendering a composite image directly from these coordinates (Figure 2). As the sea surface is roughly planar, assuming one is careful to account for the aspect ratio of pixels and degrees of latitude and longitude at the particular location, certain key traits are preserved in this virtual overhead view: straight lines are maintained, and distances and intersection angles are preserved. This rectified stitched image is a very useful construct to quickly survey large coastal regions covered by multiple cameras in a unified "bird's eye" view, without regard to individual camera orientations. Furthermore, this view is amenable to overlays of map information that can provide further details on political boundaries and labels for geographic features and regions.

Figure 2. Spatially-consistent stitching of three images into a virtual overhead view. GPS information at the ends of selected yard lines are used to compute homographies. The stitched view of the field (essentially planar) preserves straight lines, angles, and distances. Nonplanar parts (bleachers, etc.) are highly distorted and have been cropped.

3.2 Spatial Consistency via Keypoints

In our second scenario that repurposes surveillance camera images for a parking space finder service, we have no way to control the orientations of the cameras. Unlike the scientific monitoring scenario, there is no need for precise geographic calibration of the images, and the expense of localizing and maintaining calibration of a sufficient number of landmarks in each camera's view cannot be justified. For this case, we have developed a mechanism that automatically generates a set of correspondences between two overlapping images, and constructs the transformation that permits the stitching of the images, all without any extrinsic calibration of the scene with landmarks.

In contrast to previous approaches, our mechanism identifies *keypoints* in images using PCA-SIFT features [15]. These features, developed by our group building on the work by Lowe [18], map regions of an image into a high-dimensional feature vector space in a manner that is robust to differences in scale and rotation, as well as moderate changes in perspective. As the same region in a scene is likely to map to a nearby point in the feature space even when imaged by cameras with different viewpoints, the PCA-SIFT features can be used to find correspondences between two overlapping images of a scene. Likewise, as the dimensionality of the feature vector is very high, unrelated elements of the scene are highly unlikely to be near each other in the feature space. Figure 3 shows an example of automatically stitching two images of a scene. Many keypoints are identified in each of the source images. With just a few of the closest matching keypoints in the feature space, the homography transforming image coordinates between the two camera views can be automatically generated, and the images stitched together in a spatially consistent manner. Figure 4 shows an example of analyzing the stitched image for parking space availability, where the red and green triangles in the upper image indicate full and empty parking spaces, respectively.

Figure 3. Automatic stitching of multiple camera images for the Parking Space Finder using keypoints. The circles mark PCA-SIFT interest points in the lower two source images. A few closely matching keypoints suffice to generate a transform mapping one image's coordinates into the other's and to enable fully automated stitching.

Figure 4. A temporally inconsistent stitched image

3.3 Temporal Consistency

A complication that arises from the above image stitching algorithm occurs when the images corresponding to the different cameras have been captured at different times. This temporal inconsistency can cause errors in the computations of pixel stack instruments that are configured to use data spanning multiple cameras. This also results in obvious visual errors, as in Figure 4, where the same man appears twice, and half of a car appears in the stitched image. (Although in this particular example, the parking space availability information happens to be correct.)

Temporal inconsistency may occur for many reasons. First, for a distributed set of cameras, unless the system has been designed to operate synchronously, it is very unlikely that the cameras will capture images closely spaced in time. This is especially true when using a surveillance network of cameras not under our control, as in the parking space finder application scenario. Second, the cameras may not be capturing images at consistent intervals. Even with video cameras with fixed frame rates, there are good reasons for not keeping all of the frames. In particular, locally adapting the interval of capture based on the rate of scene change can help significantly reduce storage and network bandwidth overheads. Finally, transient failures of the cameras and dis-

Input:
ε: Upper bound on the temporal divergence
t_{LB}: Oldest acceptable timestamp
$\{I_{i,j}\}, \{t_{i,j}\}$: All images and corresponding timestamps, where $i \in [1,.n]$ is the camera id, $j \in [1..m_i]$ is the image index, m_i is the number of images from camera i, and $t_{i,j} < t_{i,j+1}$
Output:
Temporally Consistent Set (**TCS**) of images

Algorithm:
initialize: $p_i := m_i, \forall i \in \{1,2,\ldots,n\}$
loop:
$t_{max} := \max_{i=1}^{n}\{t_{i,p_i}\}$; $t_{min} := \min_{i=1}^{n}\{t_{i,p_i}\}$
if $(t_{max} - t_{min} \leq \varepsilon)$ **return** TCS $:= \{I_{1,p_1}, I_{2,p_2}, \ldots, I_{n,p_n}\}$
find an i such that $t_{i,p_i} = t_{max}$; $p_i := p_i - 1$
if $((p_i < 1)$ **or** $(t_{i,p_i} < t_{LB}))$ **return** *NULL*

Figure 5. Centralized algorithm for finding a temporally consistent set of images from multiple cameras

connections in the network can result in missing data from some of the cameras. The larger the system, the greater the likelihood that images from one or more cameras are not available at any given time.

Constructing a temporally consistent response to a query of the distributed system requires the identification of a set of images acquired at approximately the same time [21]. Missing data due to disconnections makes this a nontrivial task, as the system will have to look back into historical data to find a set of images with comparable acquisition times from all relevant sensors. The semantics of this operation is to return the most recent, complete set of images with acquisition timestamps that differ by no more than some small, specified bound. Here, the consistency of the timestamps takes priority over the recency of the images (within the limits of a specified oldest acceptable timestamp).

A brute-force algorithm for constructing a temporally consistent response would be to compare all possible sets of images and return the best. Consider a camera network with n cameras and let m_i be the number of images from camera i, for $i = 1,\ldots,n$. Then the brute-force algorithm would compare $\prod_{i=1}^{n} m_i$ possible sets, i.e., an exponential number of sets (e.g., m^n sets when $m_i = m$ for all i).

Instead, we present an efficient algorithm to determine a temporally consistent set of images, as summarized in Figure 5. The algorithm runs in only $O((\sum_{i=1}^{n} m_i)\log n)$ time, nearly linear in the number of images on all the cameras.

The user specifies a desired bound, ε, on the temporal divergence and a bound, t_{LB}, on the maximum age/staleness to be considered. For each relevant camera i, we start with a set of m_i images and corresponding acquisition timestamps, $t_{i,1}, t_{i,2}, \ldots, t_{i,m_i}$, in chronological order. We have a set of n pointers, $p_1, \ldots, p_n$, one for each of the n cameras, each initialized to m_i.

At each iteration of the algorithm, we find the minimum and maximum timestamps, t_{min} and t_{max}, from the set $\{t_{1,p_1}, \ldots, t_{n,p_n}\}$. If $t_{max} - t_{min} \leq \varepsilon$, we have found a temporally consistent set of updates, and can construct the reply based on the data corresponding to the selected set of pointers. Otherwise, we decrement the pointer p_j corresponding to the t_{max} element. We repeat this process, unless $p_i < 1$ or $t_{i,p_i} < t_{LB}$, in which case we have failed to find a consistent set, and return an error code as appropriate. By maintaining the timestamps for the current set of n pointers in a simple heap data structure, each loop iteration takes at worst $O(\log n)$ time. Because each non-returning loop iteration decreases a pointer, there are fewer than $\sum_{i=1}^{n} m_i$ loop iterations.

Input:
c: number of children of this node
ε: Upper bound on the temporal divergence
t_{LB}: Oldest acceptable timestamp
$\{[l_{i,j}, u_{i,j}]\}$: Ordered set of timestamp ranges from its children, where $i \in [1..c]$ is the child id, $j \in [1..r_i]$ is the range index, r_i is the number of ranges from child i's subtree, $l_{i,j-1} < l_{i,j} \leq u_{i,j} < u_{i,j+1}$, and $l_{i,1} \geq t_{LB}$
Output:
Ordered set of candidate ranges (**CR**) for this node's subtree

Algorithm:
initialize: $p_i := r_i, \forall i \in \{1,2,\ldots,c\}$; $CR := \emptyset$
loop:
$t_{max} := \max_{i=1}^{c}\{u_{i,p_i}\}$; $t_{min} := \min_{i=1}^{c}\{l_{i,p_i}\}$
if $(t_{max} - t_{min} \leq \varepsilon)$
if (is_root) **return** $\{[t_{min}, t_{min} + \varepsilon]\}$ as the final range
else
let $[l^*, u^*]$ be the range most recently added to CR
if $(t_{max} < u^*)$
add $[t_{min}, t_{max}]$ to CR
if $(t_{min} = l^*)$ remove $[l^*, u^*]$ from CR
find an i such that $u_{i,p_i} = t_{max}$; $p_i := p_i - 1$
if $(p_i < 1)$
if (is_root) **return** *NULL* **else return** CR

Figure 6. Distributed algorithm for finding a timestamp range corresponding to a temporally consistent set of images in a hierarchical distributed system

3.4 Scalable, Distributed Consistency

We have implemented our consistency-preserving image stitching techniques into our IrisNet prototype. We make use of an IrisNet facility called *stored procedures* that allows arbitrary processing and aggregation of retrieved data. By implementing our stitching algorithms as stored procedures, we take advantage of distributed, in-network merging of images as they are passed up the database hierarchy.

To make best use of in-network stitching, we have also developed a distributed version of our temporal consistency algorithm, outlined in Figure 6. (Developing distributed versions of the two spatial consistency algorithms is straightforward.) Here, the algorithm is run on each node in the queried database subtree, starting from the parents of the leaves, and repeating the algorithm up the tree. To start the process, the leaf nodes first pass their set of image timestamps to their parents, where each (distinct) timestamp $t_{i,j} \geq t_{LB}$ at a leaf is encoded as the range $[t_{i,j}, t_{i,j}]$. By summarizing the timestamps in each of its child's subtree by a collection of candidate ranges, the parent node can efficiently determine the set of candidate ranges in its subtree. Once the final range is determined at the root, a query can be initiated to retrieve and stitch a consistent set of images, using the most recent image at each camera that falls within this range.

4 Related Work

In the past multiresolution spline techniques [6] have been used to combine multiple images into a single large mosaic. More recently, Eden et al. [10] present techniques to produce

composite mosaics from images collected from several different cameras that have diverse calibration settings. Combining images from multiple cameras to generate panoramas has also been studied in the context of video sequences, as described in [8, 13, 19, 20]. The approaches in these papers take advantage of the spatial correlation between cameras with overlapping fields of view and the temporal correlation between successive frames of a video sequence, in order to generate smooth and highly informative panoramic video sequences. In [19], the spatial and temporal misalignments between multiple camera captures of the same dynamic scene are used to provide more detailed information on the scene. Steedly et al. [20] present optimal techniques that automatically stitch successive key frames of multiple video sequences. Kang et al. [13] present techniques for concise video summarization with maximum information content from multiple input videos. Constructing larger image mosaics from smaller images has also been studied in the context of texture synthesis [16]. In the above papers, spatial consistency is ensured by using graph cut optimizations or similar techniques. Smooth continuity in the resultant output video is achieved by using the spatial and temporal correlations between temporally adjacent frames in the input video sequences. The most related work on spatial consistency via keypoints is by Brown and Lowe [5]; their paper uses Lowe's SIFT algorithm [18] to find keypoint correspondences among stored images, in order to automatically construct panoramas offline. One of the motivations for developing the faster PCA-SIFT algorithm was to enable online stitching of live camera images.

In the context of real-time databases, a key issue is maintaining temporal consistency while satisfying the timing constraints [21]. More recent work deals with providing temporally consistent write ahead logging in sensor database systems [14] and trading off the QoS of temporal consistency for the number of transactions processed within the limited time [17]. Finally, Ahmed and Vrbsky [4] describe a trigger-based updating mechanism for out-of-date derived objects.

5 Conclusion

Many interesting applications that analyze composite images from distributed camera networks pose stringent spatial and temporal consistency requirements. In this paper, we have described techniques enabling fast, on-demand composite analysis of current and recent images from multiple cameras, which meet these consistency requirements. We implemented our techniques in a prototype IrisNet system running over a distributed camera network, and reported on two real-world applications. This work on multi-camera image stitching complements IrisNet's existing features for distributed query processing over a collection of single-camera data products, extending the generality of the system.

Acknowledgements. We thank our colleagues Rahul Sukthankar and Yan Ke for their help with PCA-SIFT and homographies, and for developing an early prototype of image stitching using PCA-SIFT on IrisNet. We thank Rob Holman, John Stanley, Curt Vandetta and the members of the CIL lab at OSU for their help in collecting GPS measurements for the OSU football field. We thank Suresh Singh for providing financial support for the first author during part of this work from the grant NSF (ITR Medium) 0325014. Much of this work was done while the first author was an intern at Intel Research Pittsburgh.

6 References

[1] National Weather Service, Rip Current Forecasts. http://www.ripcurrents.noaa.gov/forecasts.shtml.

[2] The Argus Project, College of Oceanic and Atmospheric Sciences, Oregon State University. http://cil-www.coas.oregonstate.edu/.

[3] United States Life Saving Association. http://www.usla.org/Statistics/public.asp.

[4] Q. N. Ahmed and S. V. Vrbsky. Triggered updates for temporal consistency in real-time databases. *Real-Time Systems*, 19(3):209–243, 2000.

[5] M. Brown and D. Lowe. Recognising panoramas. In *ICCV*, pages 1218–1225, 2003.

[6] P. J. Burt and E. H. Adelson. A multiresolution spline with application to image mosaics. *ACM Transactions on Graphics*, 2(4):217–236, 1983.

[7] J. Campbell, P. B. Gibbons, S. Nath, P. Pillai, S. Seshan, and R. Sukthankar. IrisNet: An internet-scale architecture for multimedia sensors. In *ACM Multimedia*, pages 81–88, 2005.

[8] Y. Caspi and M. Irani. Spatio-temporal alignment of sequences. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(11):1409–1424, 2002.

[9] A. Deshpande, C. Guestrin, S. Madden, J. M. Hellerstein, and W. Hong. Model-driven data acquisition in sensor networks. In *VLDB*, pages 588–599, 2004.

[10] A. Eden, M. Uyttendaele, and R. Szeliski. Seamless image stitching of scenes with large motions and exposure differences. In *IEEE CVPR*, pages 2498–2505, 2006.

[11] P. B. Gibbons, B. Karp, Y. Ke, S. Nath, and S. Seshan. IrisNet: An architecture for a worldwide sensor web. *IEEE Pervasive Computing*, 2(4):22–33, 2003.

[12] R. Hartley and A. Zisserman. *Multiple View Geometry in Computer Vision*. Cambridge University Press, 2000.

[13] H.-W. Kang, Y. Matsushita, X. Tang, and X.-Q. Chen. Space-time video montage. In *IEEE CVPR*, pages 1331–1338, 2006.

[14] H. Kawashima, M. Imai, M. Toyama, and Y. Anzai. Providing persistence for sensor data streams with temporal consistency conscious WAL. In *Information Systems and Databases*, pages 13–18, 2002.

[15] Y. Ke and R. Sukthankar. PCA-SIFT: A more distinctive representation for local image descriptors. In *IEEE CVPR*, pages 506–513, 2004.

[16] V. Kwatra, A. Schodl, I. Essa, G. Turk, and A. Bobick. Graphcut textures: Image and video synthesis using graph cuts. *ACM Transactions on Graphics*, 22(3):277–286, 2003.

[17] K.-Y. Lam, M. Xiong, B. Y. Liang, and Y. Guo. Statistical quality of service guarantee for temporal consistency of real-time data objects. In *IEEE RTSS*, pages 276–285, 2004.

[18] D. Lowe. Distinctive image features from scale-invariant keypoints. *Intl. Journal Computer Vision*, 60(2):91–110, 2004.

[19] E. Shechtman, Y. Caspi, and M. Irani. Space-time super-resolution. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27(4):531–545, 2005.

[20] D. Steedly, C. Pal, and R. Szeliski. Efficiently registering video into panoramic mosaics. In *ICCV*, pages 1300–1307, 2005.

[21] M. Xiong, J. A. Stankovic, K. Ramamritham, D. Towsley, and R. Sivasankaran. Maintaining temporal consistency: Issues and algorithm. In *RTDB*, pages 1–6, 1996.