IrisNet: An Internet-Scale Architecture for Multimedia Sensors

Jason Campbell[†] Phillip B. Gibbons[†] Suman Nath[‡]
Padmanabhan Pillai[†] Srinivasan Seshan[‡] Rahul Sukthankar[†‡]

[†]Intel Research Pittsburgh [‡]Carnegie Mellon University

{jason.d.campbell, phillip.b.gibbons, padmanabhan.s.pillai}@intel.com, {sknath,rahuls}@cs.cmu.edu, srini@cmu.edu

ABSTRACT

Most current sensor network research explores the use of extremely simple sensors on small devices called motes and focuses on overcoming the resource constraints of these devices. In contrast, our research explores the challenges of multimedia sensors and is motivated by the fact that multimedia devices, such as cameras, are rapidly becoming inexpensive, yet their use in a sensor network presents a number of unique challenges. For example, the data rates involved with multimedia sensors are orders of magnitude greater than those for sensor motes and this data cannot easily be processed by traditional sensor network techniques that focus on scalar data. In addition, the richness of the data generated by multimedia sensors makes them useful for a wide variety of applications. This paper presents an overview of IRISNET, a sensor network architecture that enables the creation of a planetary-scale infrastructure of multimedia sensors that can be shared by a large number of applications. To ensure the efficient collection of sensor readings, IRISNET enables the application-specific processing of sensor feeds on the significant computation resources that are typically attached to multimedia sensors. IRISNET enables the storage of sensor readings close to their source by providing a convenient and extensible distributed XML database infrastructure. Finally, IRISNET provides a number of multimedia processing primitives that enable the effective processing of sensor feeds in-network and at-sensor.

Categories and Subject Descriptors

D.2.11 [**Software Engineering**]: Software Architecture—*Domain-specific Architectures*; I.4.9 [**Computing Methodologies**]: Image Processing and Computer Vision—*Applications*

General Terms

Algorithms, Design

Keywords

Wide-area sensing infrastructure, Multimedia sensors, Sensor networks

MM'05, November 6–12, 2005, Hilton, Singapore.

1. INTRODUCTION

The increasing availability of commodity multimedia sensors, such as microphones and cameras, and the accelerated trend toward ubiquitous Internet connectivity provide new opportunities for Internet-scale applications. These Internet-scale multimedia sensing applications could provide several advantages over existing "mote-based" [24] sensing applications [8,17,26,40,43,46,48], that instrument relatively-small areas with large numbers of resource-constrained scalar sensors. First, in multimedia sensing, a richer set of source data can be considered, including high bit-rate video and audio that cannot be handled by mote devices. Second, multimedia sensors can passively monitor the physical space where it is not possible to closely instrument the physical objects of interest. Third, they are cheap—a single inexpensive digital camera (such as those being developed for cell phones, PDAs, and webcams) can replace dozens of discrete sensor nodes for applications such as vehicle presence detection, velocity/position sensing, and depthmap estimation. Finally, typical multimedia sensor platforms are not subject to the constraints faced by mote-class sensors (e.g., connectivity, processing, memory and power constraints), and can be deployed on a global scale.

However, Internet-scale multimedia sensing applications pose a new set of challenges. Multimedia sensors provide high bit-rate data, requiring compute-intensive processing algorithms. The Internet-scale of the applications requires distributed storage and indexing of the sensor data, along with algorithms for processing expressive queries on that data. The applications frequently require capabilities such as near real-time retrieval of useful information, correlation of data collected from multiple different sensors, triggers associating sensed events with pre-specified tasks, protection of privacy of sensitive information and archival of sensor data. Additional requirements, generic to all distributed systems, include fault-tolerance and load balancing. These requirements make development of a new application a daunting task, evidenced by the current lack of such applications in the real world.

This paper describes IRISNET (Internet-scale, Resource-intensive Sensor Network Services) [3, 18], the first general purpose, shared software infrastructure that simplifies the task of developing new Internet-scale multimedia sensing applications. To address the challenges discussed above, IRISNET implements necessary primitive mechanisms that new applications can use as building blocks. Moreover, it provides a simple way for a developer to customize the use of the generic functionalities (e.g., how data should be indexed in the global storage) and to specify the additional application-specific functionalities for processing sensor feeds. Thus, the application developer needs to write only a small amount of code to create a new application. IRISNET achieves scalability through application-

specific filtering of sensor data near their sources and by organizing the data in hierarchies. It achieves generality by storing data in a distributed XML database and by supporting a standard, yet *extensible*, query processing language. Finally, IRISNET enables good performance by sharing resources among the concurrently-running applications.

1.1 Applications

Rich sensing of the world using multimedia sensors can enable a wide range of applications. One potential application domain includes real-time, physical "congestion avoidance" services that offer traffic routing advice, find destination-proximal available parking spaces, or provide time-in-line indications for popular services (e.g., post office, amusement park ride). Another application area would include "silent witness" devices that can infer and record potentially relevant activities (e.g., car accidents) for future query. Scientific habitat and environmental monitoring may be another area of application, where visual information can often be used both for direct observations as well as to indicate when a resource-constrained observation, such as employing one of a finite supply of sample containers, might be fruitfully-applied. Additional applications include a Bus Alert that notifies users when to leave for the bus stop, a Lost-and-Found service that reports the most recent location recorded for a missing object, and a Family Monitor that watches the user's children or aging relatives.

The rich data provided by multimedia sensors can be interpreted using different algorithms, enabling the same sensor to be employed for a wide variety of applications. For instance, a camera observing a parking lot could support applications that report parking space occupancy, vehicle velocities, detected collisions or the current weather, depending upon the deployed software. A set of flexible sensor nodes deployed in a wide area can serve a multitude of sensing applications and services. Such a public sensing network could spur the development of a world-wide sensor web in which the resources of a multitude of sensor nodes are available (either on a free or a commercial basis) to many independent entities developing sensor network applications. Just as "network effects" have spurred the deployment of e-mail, Web, and instant messaging by providing increasing returns to scale, so could Internet scale sensing expand the range of sensor network applications. Furthermore, we can imagine a future in which sensor network applications can take advantage of myriad, semantically-rich, real-time and historical data streams using composable sets of services — permitting Web-like services of today to be applied to sensed data streams.

1.2 Design Principles

The design of IRISNET is guided by the following principles.

Shared infrastructure: The most important principle underlying the design of IRISNET is that of infrastructure sharing. There are several implications of this principle: First, the system must provide the functionality required by typical sensing applications such that they can employ IRISNET as a building block (e.g., data collection, storage and query processing). Second, new applications must be able to reuse existing deployed sensor resources and existing application components. Reuse enables developers to focus only on the missing, domain-specific code for the new applications, significantly accelerating the development process. Finally, IRISNET must optimize CPU utilization across applications by detecting the case when two applications request the same sequence of (computationally expensive) multimedia processing steps and caching/reusing those intermediate results.

This emphasis on building a multi-application *shared infrastructure* contrasts with the single-purpose deployments of most "mote-based" sensor networks.

Large collection of high bit-rate sensors: IRISNET targets applications that employ numerous globally-distributed sensors that observe the physical world. Because of the very large volumes of data collected and the potential need for historical as well as current sensor data in some applications, IRISNET stores observations near their sources and transmits them across the Internet only as needed. This results in a dramatic reduction in the global bandwidth requirements of a multimedia sensor network.

Data as a single queriable unit: IRISNET enables users to view a collection of sensors as a single unit that supports a high-level query language. Each query operates on data collected across (potentially) the entire global sensor network, just as a single Google search query encompasses millions of Web pages. But beyond straightforward keyword search, IRISNET supports rich queries including arithmetic, aggregation, and other database operators.

Privacy: Use of multimedia sensors often raises important privacy concerns; e.g., camera-based applications can potentially violate the privacy of the observed individuals. IRISNET aims to provide useful mechanisms for addressing these concerns. IRISNET distinguishes between privileged and non-privileged algorithms in the context of a given sensor feed: privileged algorithms can access the raw sensor data, while unprivileged algorithms can access only the subsidiary dataflows resulting from applying privileged algorithms. The privileged algorithms can use a variety of techniques to block sensitive information (e.g., human faces) or distill the raw sensor feed into a semantically-restricted form that includes only non-sensitive information (e.g., returning a bitmap representing parking space occupancy rather than an image of the parking lot).

Ease of application development: Finally, IRISNET greatly simplifies the task of developing new applications by providing high-level abstractions of the sensing infrastructure that hide the complexities of the underlying distributed-data-collection and query-processing mechanisms. This is crucial to fostering system adoption and proliferation of new applications.

The remainder of the paper is organized as follows. Section 2 provides a general overview of the IRISNET architecture. Section 3 details some multimedia functionalities provided to IRISNET applications. Section 4 describes several applications that have been developed using the IRISNET infrastructure. Section 5 surveys a selection of the related work. Section 6 concludes the paper.

2. SYSTEM ARCHITECTURE

IRISNET is designed to provide a shared, Internet-scale, long-lived software infrastructure that makes it easy to develop and deploy sensor applications. The infrastructure sits on top of a potentially vast collection of shared sensors, which can be added or removed on the fly. The sensors can be a heterogeneous mix of imaging and non-imaging sensors (cameras, microphones, RFID readers, photo detectors, etc.)—the only assumption is that each sensor feed is fed to an Internet-connected machine with sufficient computing power and storage (i.e., PDA-class or better). IRISNET makes it easy for application developers to create sensor applications, by exporting relatively simple interfaces to application developers and by transparently performing a wide variety of challenging tasks on behalf of applications. This section presents an overview of the IRISNET system architecture, the steps to develop an application using IRISNET, and the tasks IRISNET performs on behalf of applications.

2.1 SAs and OAs

IRISNET provides a two-tier architecture of *Sensing Agents (SAs)* for data collection and filtering and *Organizing Agents (OAs)* for data storage and querying, as depicted in Figure 1.

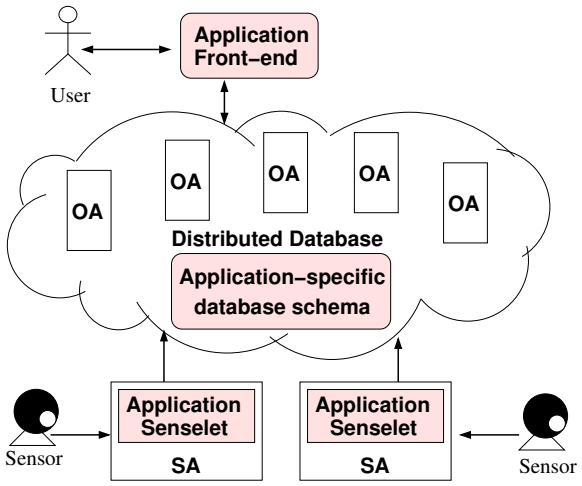

Figure 1: IRISNET architecture. The shaded regions denote application-specific components.

Sensing Agents (SAs) are software modules that collect and filter sensor feeds. These modules run on the machines connected to the raw sensor feeds, which we term *SA hosts*. Multiple sensors of various types may feed into the same SA host; these feeds are buffered within the SA. Because multimedia sensors can produce vast quantities of data in a short period of time, IRISNET enables running application-specific filtering modules, called *senselets*, at the sensor node itself. A senselet is typically used to distill raw data to the semantic information needed by a specific application. For example, our parking space finder (PSF) application (see Section 4) uses a PSF-specific senselet that reduces the rich video feed overlooking parking spaces down to a few bytes of per-space availability data per time period. This significantly reduces network bandwidth consumption because only the occupancy information is sent on the network. SAs provide a common runtime environment for senselets, across the entire sensor network. An application developer can thus write a *single* senselet (e.g., the PSF senselet) that is uploaded and executed on all of the desired sensor feeds (e.g., all cameras overlooking parking spaces). Because a separate instance of the senselet is concurrently run on each machine receiving a raw sensor feed, senselet processing scales with the number of such machines.

Figure 2 depicts the execution environment of an SA running on an SA host. In the figure, the webcam feed is first placed in the raw sensor buffer, where it can only be read by "privileged" senselets. Privacy filters are also run, with the output placed in the processed sensor buffer (privacy filters are discussed in Section 3). Application senselets (such as the PSF senselet) can access the privacy-processed feeds. A shared memory pool is provided for use by application senselets as working space for their processing. To ensure that a malicious or buggy application does not use excessive resources or compromise an SA host, senselets are sandboxed within a virtual machine on the host [36]. Multiple application senselets can process the same feed concurrently, e.g., the same video feed can be used by both a PSF application and a Lost-and-Found application. To help alleviate the load on the SA host, the SA identifies opportunities to share common processing steps across senselets, in order to eliminate duplicate computation and to avoid storing multiple copies of the same partial results. In particular, each SA includes the OpenCV computer vision library [11]. The SA uses a memoization process that enables it to detect cases where an identical sequence of OpenCV functions has already been

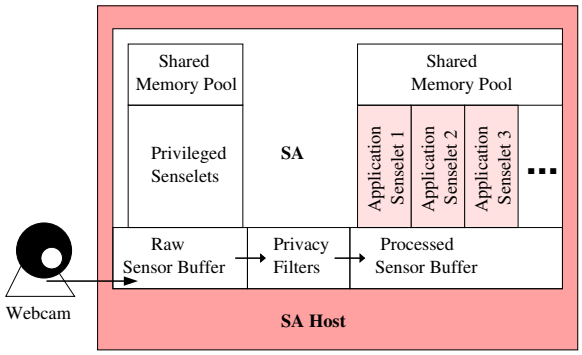

Figure 2: SA architecture

applied to a given image frame (e.g., background subtraction, edge detection) [36]. In such cases, the existing result (residing in the shared memory pool) is used in lieu of recomputing the result. To keep network bandwidth in check, each senselet sends only the distilled information to the OAs.

Organizing Agents (OAs) are software modules that collectively provide a distributed database for each application. These modules run on Internet-connected machines. The mapping of OAs to machines is transparent to IRISNET application developers and users, and can be adjusted by IRISNET to achieve various system performance and availability goals. Each OA has a local database for storing the distilled information produced by individual senselets, as well as any aggregated information on the data, as desired. The set of local databases for an application combine to constitute an overall database for the application. Unlike many previous solutions, IRISNET supports widely-distributed databases, to ensure scalability despite the high sensor feed volumes (as discussed in the introduction). One of the key challenges is to divide the responsibility for the overall database among the participating OAs. IRISNET relies on a hierarchically organized database schema (using the XML data representation) and on corresponding hierarchical partitions of the overall database, in order to define the responsibility of any particular OA [16]. Applications can use XPath [5], a standard XML query language, to query the sensor database.

2.2 Developing an application using IrisNet

Given a deployment of IRISNET on a (vast) collection of available sensors, there are just three steps required to develop a new sensor-based application. First, the application developer creates the sensor database (XML) schema that defines the attributes, tags and hierarchies used to describe and organize distilled sensor data. For example, in our IRISLOG application (discussed in Section 4), the database hierarchy consists of a root node, with two children (USA and non-USA), each with multiple regions, each with multiple sites, etc., as depicted in Figure 3. Associated with each node in this hierarchy are attributes and fields for storing distilled data pertinent to the node (e.g., storing the current memory usage by user-1 on CMU-2). Second, the application developer writes senselet code for the SAs, for converting raw sensor feeds into updates on the database defined by the schema. Third, the application developer provides an application-specific front end for end users to access the application. The front end converts user requests into XPath queries on the database defined by the schema. These three application-specific components are highlighted in Figure 1.

In this way, IRISNET makes it easy to create and deploy new sensor-based applications. IRISNET seamlessly handles many of the common tasks within such applications, as discussed below.

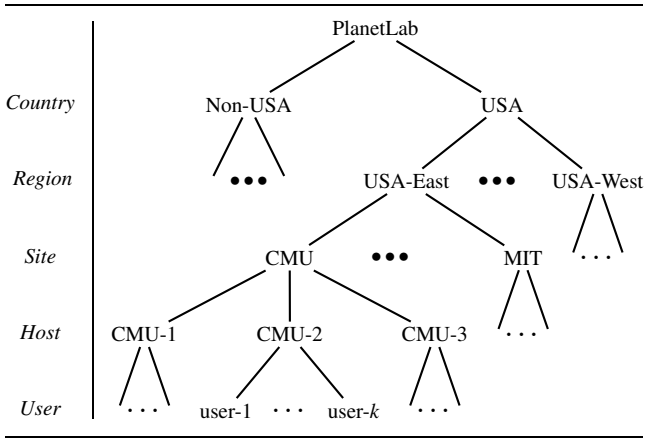

Figure 3: XML hierarchy used by IRISLOG

Note that the development process as described above permits application-specific code within individual SAs but not within the OAs. For applications that require special processing within the OAs (e.g., specialized aggregation or fusion across geographically dispersed sensor feeds), IRISNET provides an extensibility mechanism for defining such functions and seamlessly incorporating them within the system [13]. An example, for stitching images, is given in Section 3.

2.3 IrisNet features

IRISNET transparently performs a wide variety of common yet challenging tasks on behalf of applications. For scalability, IRISNET supports distributed data collection, processing and storage. IRISNET shields application developers from the many challenging complexities of distributed systems. Consider each of the application-specific components described above. The database schema is a *logical* hierarchy, independent of any actual distribution of the data. In fact, from the application's perspective, it can be viewed as a monolithic centralized database. The front end queries the database as if it were centralized, using a standard XML query language (XPath) that assumes the database is centralized. Similarly, senselets and any application-specific aggregation or fusion functions are tailored to the logical hierarchy, independent of the actual distribution of the data.

Some of the key innovations within IRISNET are in its techniques to support a centralized database abstraction. Previous work did not support the richness of XML in such a transparent way (see Section 5). IRISNET transparently manages the distributed database for each application, including:

- partitioning the database among the OAs, and adjusting this partitioning on-the-fly to balance load among the OAs,
- replicating the data across multiple OAs (for fault tolerance) and keeping the replicas up-to-date,
- pushing queries into the network and out to the edges where the data resides,
- routing queries directly to the relevant OAs,
- caching data at the OAs to opportunistically alleviate hot spots and avoid network latencies,
- handling all networking aspects, and
- managing continuous queries and triggers.

Additional details are available in [13, 16, 37]. IRISNET also provides a variety of features particularly relevant to multimedia sensors, as described in the next section.

3. MULTIMEDIA COMPONENTS

As highlighted above, IRISNET offers a variety of technical components that simplify the development and deployment of sensor network applications. This section focuses on those that have particular relevance to multimedia systems, such as image and video sensors.

3.1 Camera calibration

Sensor calibration refers to the problem of mapping measurements in sensor space to the real world, such as determining the region in the world corresponding to a particular pixel in a camera's image. For cameras, calibration is traditionally decomposed into two aspects: (1) *intrinsic* calibration, which is concerned with modeling transforms that are camera-specific, such as the effects of lens distortion; and (2) *extrinsic* calibration, which is concerned with the physical position and orientation of the camera with respect to the scene. The former is independent of camera location while the latter is independent of the camera internals. In IRISNET, the intrinsic calibration for each camera is independently performed (in advance of deployment) using the routines provided in OpenCV [10,11]. Several images of a checkerboard calibration target in various orientations are captured using the camera, and the intrinsic parameters are recovered automatically.

Extrinsic calibration can only be performed on site, once the cameras have been deployed. In the simplest case, the user can manually specify correspondences between known global landmarks and points in a camera image. Correspondences between objects visible in different camera views can also enable IRISNET to infer the relative mapping between cameras. Manual calibration of such systems clearly does not scale, particularly because, in real deployments, camera pose can shift over time. IRISNET employs techniques derived from recent work in multi-view geometry [22] to automatically calibrate camera networks. For instance, when the scene of interest can be approximated by a planar surface (e.g., a typical outdoor parking lot or an ocean surface), the relationship between the image coordinates of a scene point viewed by multiple cameras is concisely described by the class of projective transforms (also termed a homography). The homography between two images can be recovered from as few as four point correspondences using standard techniques. Scenes with significant 3-D structure require more complicated methods for wide-baseline stereo. For all of these algorithms, an important challenge is to automatically identify correspondences between images, despite changes in appearance due to differing viewpoint. For example, given two cameras watching a coastline, IRISNET needs to determine that the lighthouse visible in the corner of one image is the same lighthouse that is in the center of the second image. We describe our approach to this problem in Section 3.2.

3.2 Keypoints

IRISNET employs distinctive interest points or *keypoints* to automatically identify correspondences between images. These have been an active area of recent research, and keypoints have been successfully employed in a variety of applications including object recognition [32], image retrieval [28], and automatic panorama generation [12]. The basic idea can be summarized as follows. Each image is processed using an interest operator to identify regions that are highly-distinctive and stable to local changes. IRISNET employs the scale-invariant feature transform (SIFT) interest

Figure 4: Finding correspondences using PCA-SIFT. Correct correspondences are denoted by white lines and incorrect ones with black dotted lines. 9/10 of the correspondences in the left scene are correct, while 10/10 of the ones in the right scene are correct.

operator [32] to find stable local optima in scale-space of the image (intuitively, these are "blobs" that are somewhat robust to viewpoint and illumination changes). The local image neighborhood around each interest point is represented using the PCA-SIFT [27] descriptor that we developed, generating a 36-dimensional feature vector. Feature vectors corresponding to the same region in the scene tend to map to nearby points in feature space, even if they were imaged at different scales and viewed from slightly different camera positions. The PCA-SIFT representation is somewhat robust to illumination effects and perspective distortion, making it well-suited as a means for finding correspondence points between images. Figure 4 shows two examples of PCA-SIFT on an indoor and an outdoor scene. In each figure, ten point correspondences were matched between the two views; PCA-SIFT only made one error.

3.3 Image stitching

Fusing information collected from multiple sensors, such as cameras with partially-overlapping fields of view, is an important component of large-scale sensor networks. The keypoint technique described in Section 3.2 enables IRISNET to automatically detect overlapping regions and to generate panoramic views from multiple cameras. As is well-known [22], such panoramas are possible for arbitrary scenes only if all of the camera centers are collocated, or when the scene is intrinsically planar. Fortunately, for many common IRISNET applications (e.g., parking lot or coastline monitoring), the latter is approximately true.

Figure 5 shows two camera views of a parking lot and IRISNET's automatically-generated panorama. Several keypoints in the overlap region are automatically identified as being common, and are used to generate a transform between the two views. In regions where multiple cameras view the same scene, IRISNET uses pixels from the camera that has a higher resolution view of that scene. Regions that are far from a camera are imaged at lower resolution. For images related by a homography, the relationship between areas in the overhead view and areas in a particular source image is given by $A(x,y) = |J(X,Y)|\, a(X,Y)$, where J is the Jacobean of the transformation and $A(x,y)$ and $a(X,Y)$ are the areas of pixels in

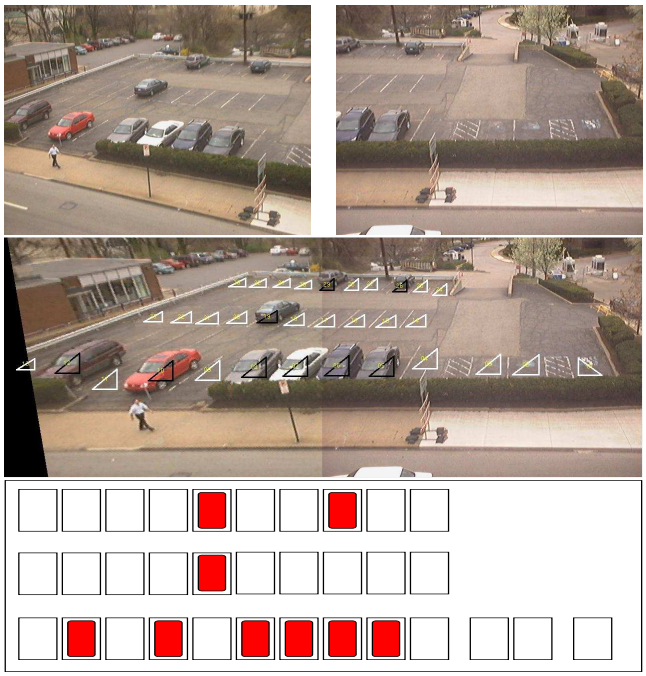

Figure 5: Views from two cameras (top) are automatically merged into a single panoramic view (middle). Keypoints in the overlap region are matched to generate the appropriate image warp to fuse the two images. The fused image is used by Parking Space Finder to detect unoccupied and occupied spaces (denoted by light and dark triangles respectively). The distilled data (bottom) is sent to the OAs.

the source image and the overhead view, respectively. The parking space finder (PSF) application (see Section 4) analyzes these panoramas to determine empty and occupied parking spaces.

Similar image warping techniques can also be used to generate overhead (rectified) views of planar scenes, such as a bird's eye view of a coastline. These transforms are particularly useful in the context of oceanography applications, as discussed in Section 4.

3.4 Techniques for privacy protection

There is increasing concern about the potential for privacy violation generated by a wide-scale deployment of sensor network technology, particularly in the case of images and video. IRISNET eschews prescribing particular privacy policies. Rather, the infrastructure is designed to support (and help enforce) privacy policies as they are developed. Policies for privacy enforcement can be categorized as those based on technological mechanisms or social contracts. While the IRISNET infrastructure cannot guarantee that sensor data cannot be misused by malicious senselets (or malicious operators), it reduces the risk for privacy violation in the following ways.

First, where possible, IRISNET encourages the use of senselets that digest the sensor data into symbolic form, at the collection point—eliminating the propagation and storage of raw images and video where possible. For instance, in the PSF application, the senselets process the video stream and output only a binary vector of filled and empty parking spaces. However, this by itself cannot prevent senselets from sending raw images or (more deviously) steganographically-encoding sensitive image content in the senselet's output.

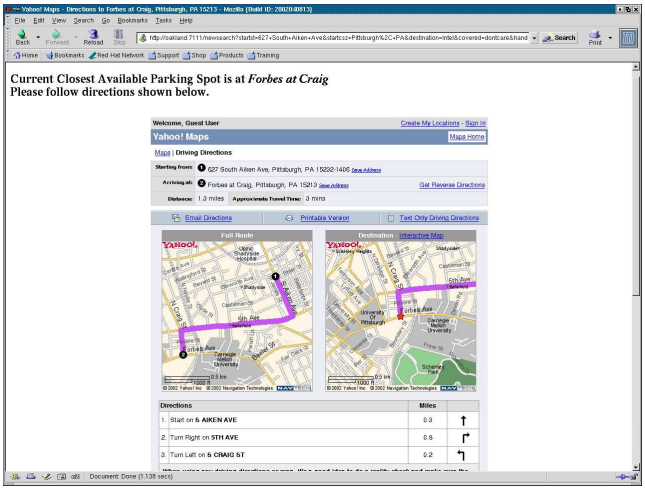

Figure 6: Parking Space Finder application front-end showing directions to nearest available parking space.

Second, IRISNET supports a set of privacy-preserving filters that process the sensor data before it is made available to application senselets. For instance, for the PSF application, IRISNET currently pre-processes camera images by finding human faces [47] and replacing each face with a black box—making it more difficult to identify pedestrians in these images. A license plate detector could be similarly employed to anonymize vehicles. Such privacy-preserving filters will become more powerful as automatic techniques for more sophisticated object detection and scene analysis are developed. The architecture for privacy-preserving filters is shown in Figure 2. Raw video data is available only to a set of *privileged senselets*, such as those that the owner of a camera may deploy for personal use. The anonymized data is shared with standard IRISNET application senselets. The SA runtime employs cryptographic signatures to verify the identity of privileged senselets before execution.

4. PROTOTYPE DEPLOYMENTS

The IRISNET framework has been applied to and tested in several different application domains, including image-based oceanography and network monitoring. These applications illustrate the flexible nature of the IRISNET system, in particular its ability to work with both imaging and non-imaging sensors. Three such prototype applications are outlined below.

4.1 Parking Space Finder

The Parking Space Finder (PSF) is intended to provide the useful service of locating available parking spaces near a desired destination and directing the driver to such a space (see Figure 6). The system utilizes a set of cameras connected to IRISNET SAs running senselets to (1) detect the presence of cars in spaces and (2) update the distributed database with this high-level semantic information. The database itself is organized according to a geographic hierarchy, and is logically divided by region, city, neighborhood, block, etc. This hierarchy fits well with the application, as any update from a given camera or query from a given driver is likely to touch only small subtrees of the database, improving the scalability of the distributed system.

The PSF front-end is a web-based interface that takes as input the desired destination and current location. It queries IRISNET for the closest spot to the destination that is not occupied, and that matches other user-specified parameters, e.g., whether covered, if a permit is required, maximum hourly rate, etc. The front-end then uses Yahoo!® Maps online service to generate driving directions to the available parking spot. We imagine that in the future, this front-end can be integrated into a car's navigation system, and would be able to get current location and destination directly from the system, and make use of the built-in mapping to generate driving directions.

The PSF is able to handle some real-world constraints on deployed camera systems. For example, a single camera may not be able to cover a particular parking lot. Our current PSF senselet is able to use feeds from multiple, oblique camera views and stitch them together to produce an image that covers the entire lot (as in Figure 5), before running car detection routines. The detector uses variance of pixel intensity in image regions to determine whether a parking space is occupied. A more sophisticated detector could employ machine learning techniques to acquire visual models of empty spaces, or employ techniques to directly determine the presence of cars.

Figure 7: Images from the IRISNET Coastal Imaging prototype. On the left, raw video frames. On the right, temporally smoothed images revealing the sand bars.

4.2 Coastal Imaging

In collaboration with oceanographers of the Argus project [1] at Oregon State University, we have developed a coastal imaging application on IRISNET. SAs connected to cameras deployed along the Oregon coastline run senselets to detect and monitor near-shore phenomena, such as riptides and the formation of sandbars. The system can capture and store still and temporally smoothed images, essentially raw data, and also distilled, high-level information processed through senselets. See Figure 7. Using IRISNET senselets, the application allows oceanographers to run their detection algorithms at the remote site. This permits a greater fidelity of observation (more images can be examined) than is possible through previous efforts that collected raw data over low-bandwidth modem and long-range radio links for centralized processing. Users can dynamically change parameters of the senselets, vary data sampling rates, and even install new processing algorithms to the remote camera sites, without interrupting service or making a trip to the coast.

One important type of oceanographic image-based sensor is the pixel stack [25]. These detectors track time-varying intensities of series of small regions of interest (ROIs) in coastal images, and correlate these changes with various phenomena. An important aspect of using these detectors is the correct association of regions in the images with real-world locations and coordinates. Extrinsic calibration techniques applied in IRISNET (Section 3), along with a few known ground truth points, can be used to accurately determine the correspondences between image regions and global coor-

dinates. Furthermore, we have implemented a technique to provide a composite overhead view of the coastline, by projecting multiple camera images onto global coordinates and stitching them together (Section 3.3). Such a rectified, composite image is designed to assist in the instrumenting of pixel sensors, permitting the user to select locations based on a world or map coordinate frame, which the system can automatically convert to ROIs in the source images.

4.3 IrisLog

IRISLOG [2] is a distributed infrastructure monitor that demonstrates the scalability of IRISNET, as well as its use outside the context of image-based sensing. IRISLOG is deployed in PlanetLab [4,38], an open, globally distributed infrastructure for Internet-scale network services. Currently, IRISLOG monitors over 500 PlanetLab hosts at 270 sites spanning 5 continents, and is the largest IRISNET deployment to date. Rather than using physical sensors, this application uses machine statistics, e.g., CPU load or network bandwidth consumption, and system logs as sensor inputs. It allows efficient querying of both individual and aggregate machine statistics and resource utilization across the PlanetLab infrastructure.

The database schema used in IRISLOG employs a geographic XML hierarchy, a slice of which is shown in Figure 3. At each host machine, a set of monitoring tools is executed periodically to log machine and user statistics, which are used to update locally-hosted fragments of the distributed database corresponding to the machine. Fragments corresponding to higher levels of the XML tree are automatically distributed among various machines, based on query load and performance, and replicated for fault tolerance. Compared to a system that streams log information from each host to a centralized monitoring station, IRISLOG both distributes the processing load for handling user queries, and reduces total bandwidth for handling statistics updates.

5. RELATED WORK

In this section, we explore related efforts in the following areas of work: sensor networks, multi-camera systems, and distributed databases. Note that while each of these related efforts addresses a subset of the issues in creating sensor-based applications, only IRISNET provides a complete solution for enabling a wide variety of such applications.

Sensor Networks. Sensor networks and IRISNET share the goal of making real world measurements accessible by applications. Viewed broadly, specialized sensor networks have been used for years for home security, building monitoring, etc. Networks of sensor "motes", small nodes containing a simple processor, a little memory, a wireless network connection and a sensing device, have been used for studying habitat and environment (e.g., behavior of birds at Great Duck Island [46], population of CaneToads in Australia [26], impact of climate on ecosystems at UC James Reserve [8]), for monitoring structural integrity of buildings [48], and for addressing several problems in the area of agriculture [17], health [40], education [43], etc. In contrast with these previous deployments, we believe that IRISNET is the first work that considers sensor networks with intelligent sensor nodes, high bit-rate sensor feeds, global scale, and infrastructure support.

Earlier key contributions in the area of sensor networks include designs for tiny operating systems [24] and low-power network protocols [30]. Existing systems have relied on techniques such as directed diffusion [23] to direct sensor readings to interested parties or long-running queries [9] to retrieve the needed sensor data to a front-end database. Other groups have explored using query techniques for streaming data and using sensor proxies to coordinate queries [33–35]. These designs are most relevant to sensor networks that use resource-constrained sensor motes. As a result, many of the proposed techniques are not applicable to the environment that IRISNET considers. Recent efforts have begun to explore sensor networks that mix motes with more powerful microservers [19]. However, these efforts still do not address other aspects of IRISNET's target environment.

Multi-Camera Systems and Algorithms. There has been significant research in the area of networked multi-camera systems, such as Video Surveillance and Monitoring (VSAM) [15]. Much of that work has centered on the development of algorithms for detecting objects of interest and tracking them within and between cameras. The automatic calibration of large-scale camera networks has also been the subject of recent research, such as Lee *et al.* [31] and Stauffer and Tieu [44]. These efforts are complementary to IRISNET's focus on wide-area scaling and application development tools.

Distributed Databases. The distributed database component of IRISNET shares much in common with a variety of large-scale distributed databases. Harren *et al.* [21] have investigated peer-to-peer databases that use enhanced distributed hash tables (DHTs) to provide a rich querying model. Distributed databases supporting a full query processing language such as SQL are a well-studied topic [42], with the focus on supporting distributed transactions or other consistency guarantees (*c.f.* [6, 7, 14, 20, 29, 39, 41]). None of the previous work addresses the difficulties in distributed query processing over a hierarchically organized, XML document. Suciu presented efficient query evaluation algorithms for a collection of XML documents on different sites that are connected through links [45]. Unlike this work, IRISNET maintains a single logical XML document, keeping the underlying fragmentation and distribution of the document transparent from the users.

6. CONCLUSION

Despite the availability of low-cost multimedia sensors, wired and wireless networking hardware, and cheap computers, there has been a dearth of real-world, wide-area multimedia sensing applications. This is due in large part to the difficulties in building large-scale distributed systems and the challenges of dealing with large volumes of data. IRISNET has been designed to enable such a class of Internet-scale applications, which utilize large collections of high bit-rate sensors. The system provides core services that permit efficient processing and filtering of data at the source, the sharing and reuse of large sensor deployments, efficient querying and updates of collected data, replication and fault tolerance, and multi-sensor calibration. Overall, the IRISNET framework eases the development of Internet-scale sensing applications by shielding developers from many of the complexities of implementing an efficient distributed system, thus enabling them to focus on the domain-specific aspects of their task.

Acknowledgements

We thank Shimin Chen, Amol Deshpande, Brad Karp, Yan Ke, Mahadev Satyanarayanan, Dilip Sundarraj and Haifeng Yu for their significant contributions to IRISNET, and Larry Huston for feedback on the paper. We also thank Mark Abbott, Ganesh Gopalan, Rob Holman, Chuck Sears, John Stanley and Curt Vandetta from the Argus team.

7. REFERENCES

[1] The Argus program. http://cil-www.oce.orst.edu:8080/.

[2] IrisLog: A distributed syslog. http://www.intel-iris.net/irislog.php.

[3] IrisNet: Internet-scale resource-intensive sensor network services. http://www.intel-iris.net/.

[4] Planetlab. http://www.planet-lab.net/.

[5] XML path language (XPATH). http://www.w3.org/TR/xpath.

[6] D. Agrawal and S. Sengupta. Modular synchronization in distributed, multi-version databases: Version control and concurrency control. *IEEE TKDE*, 5(1), 1993.

[7] R. Alonso, D. Barbara, and H. Garcia-Molina. Data caching issues in an information retrieval system. *ACM TODS*, 15(3), 1990.

[8] M. A. Batalin, M. Rahimi, Y. Yu, D. Liu, A. Kansal, G. S. Sukhatme, W. J. Kaiser, M. Hansen, G. J. Pottie, M. Srivastava, and D. Estrin. Call and response: experiments in sampling the environment. In *ACM SenSys*, 2004.

[9] P. Bonnet, J. E. Gehrke, and P. Seshadri. Towards sensor database systems. In *Intl. Conf. Mobile Data Management*, 2001.

[10] J.-Y. Bouguet. Camera calibration toolbox. http://www.vision.caltech.edu/bouguetj/calib_doc/.

[11] G. Bradski. Programmer's tool chest: The OpenCV library. *Dr. Dobbs Journal*, November 2000.

[12] M. Brown and D. Lowe. Recognizing panoramas. In *IEEE ICCV*, 2003.

[13] S. Chen, P. B. Gibbons, and S. Nath. Database-centric programming for wide-area sensor systems. In *IEEE Intl. Conf. Distributed Computing in Sensor Systems*, 2005.

[14] S. W. Chen and C. Pu. A structural classification of integrated replica control mechanisms. Technical Report CUCS-006-92, Columbia University, New York, NY, 1992.

[15] R. Collins, A. Lipton, H. Fujiyoshi, and T. Kanade. Algorithms for cooperative multisensor surveillance. *Proc. IEEE*, 89(10), Oct. 2001.

[16] A. Deshpande, S. Nath, P. B. Gibbons, and S. Seshan. Cache-and-query for wide area sensor databases. In *ACM SIGMOD*, 2003.

[17] N. Fauchald. Wi-fi hits the vineyeard. Wine Spectator online. http://www.winespectator.com/Wine/Free/Sign_In_redirect?cds_misc_1=/Win%e/Daily/News_Print/0,,2539,00.html, July 2004.

[18] 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), 2003.

[19] L. Girod, T. Stathopoulos, N. Ramanathan, J. Elson, D. Estrin, E. Osterweil, and T. Schoellhammer. A system for simulation, emulation, and deployment of heterogeneous sensor networks. In *ACM SenSys*, 2004.

[20] J. Gray, P. Helland, P. O'Neil, and D. Shasha. The dangers of replication and a solution. In *ACM SIGMOD*, 1996.

[21] M. Harren, J. Hellerstein, R. Huebsch, B. Loo, S. Shenker, and I. Stoica. Complex queries in DHT-based peer-to-peer networks. In *Intl. Workshop on Peer-To-Peer Systems (IPTPS)*, 2001.

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

[23] J. Heidemann et al. Building efficient wireless sensor networks with low-level naming. In *ACM SOSP*, 2001.

[24] J. Hill, R. Szewczyk, A. Woo, S. Hollar, D. E. Culler, and K. S. J. Pister. System architecture directions for networked sensors. In *ACM ASPLOS*, 2000.

[25] R. Holman, J. Stanley, and T. Ozkan-Haller. The application of video sensor networks to the study of nearshore oceanography. *IEEE Pervasive Computing*, 2(4), 2003.

[26] W. Hu, V. N. Tran, N. Bulusu, C.-T. Chou, S. Jha, and A. Taylor. The design and evaluation of a hybrid sensor network for cane-toad monitoring. In *IEEE IPSN*, 2005.

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

[28] Y. Ke, R. Sukthankar, and L. Huston. Efficient near-duplicate and sub-image retrieval. In *ACM Multimedia*, 2004.

[29] N. Krishnakumar and A. Bernstein. Bounded ignorance in replicated systems. In *ACM PODS*, 1991.

[30] J. Kulik, W. Rabiner, and H. Balakrishnan. Adaptive protocols for information dissemination in wireless sensor networks. In *ACM MOBICOM*, 1999.

[31] L. Lee, R. Romano, and G. Stein. Monitoring activities from multiple video streams: Establishing a common coordinate frame. *IEEE Trans. PAMI*, 22(8), 2000.

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

[33] S. Madden and M. J. Franklin. Fjording the stream: An architecture for queries over streaming sensor data. In *IEEE ICDE*, 2002.

[34] S. Madden, M. J. Franklin, J. M. Hellerstein, and W. Hong. TAG: A tiny aggregation service for ad-hoc sensor networks. In *USENIX OSDI*, 2002.

[35] S. Madden, M. J. Franklin, J. M. Hellerstein, and W. Hong. The design of an acquisitional query processor for sensor networks. In *ACM SIGMOD*, 2003.

[36] S. Nath, Y. Ke, P. B. Gibbons, B. Karp, and S. Seshan. A distributed filtering architecture for multimedia sensors. In *IEEE Int'l Workshop on Broadband Advanced Sensor Networks*, 2004. Extended version appears as Technical Report, Intel Research Pittsburgh, 2005.

[37] S. Nath, H. Yu, P. B. Gibbons, and S. Seshan. Tolerating correlated failures in wide-area monitoring services. Technical report, Intel Research Pittsburgh, 2004.

[38] L. Peterson, T. Anderson, D. Culler, and T. Roscoe. A blueprint for introducing disruptive technology into the internet. In *ACM Hotnets-I*, 2002.

[39] C. Pu and A. Leff. Replica control in distributed system: An asynchronous approach. In *ACM SIGMOD*, 1991.

[40] L. Schwiebert, S. K. Gupta, and J. Weinmann. Research challenges in wireless networks of biomedical sensors. In *ACM MOBICOM*, 2001.

[41] J. Sidell, J. Sidell, P. M. Aoki, S. Barr, A. Sah, C. Staelin, M. Stonebraker, and A. Yu. Data replication in Mariposa. In *IEEE ICDE*, 1996.

[42] A. Silberschatz, H. F. Korth, and S. Sudarshan. *Database Systems Concepts*. McGraw Hill, 2002.

[43] M. Srivastava, R. Muntz, and M. Potkonjak. Smart kindergarten: sensor-based wireless networks for smart developmental problem-solving enviroments. In *ACM MOBICOM*, 2001.

[44] C. Stauffer and K. Tieu. Automated multi-camera planar tracking correspondence modeling. In *IEEE CVPR*, 2003.

[45] D. Suciu. Distributed query evaluation on semistructured data. *ACM TODS*, 27(1), 2002.

[46] R. Szewczyk, A. Mainwaring, J. Polastre, J. Anderson, and D. Culler. An analysis of a large scale habitat monitoring application. In *ACM SenSys*, 2004.

[47] P. Viola and M. Jones. Rapid object detection using a boosted cascade of simple features. In *IEEE CVPR*, 2001.

[48] N. Xu, S. Rangwala, K. K. Chintalapudi, D. Ganesan, A. Broad, R. Govindan, and D. Estrin. A wireless sensor network for structural monitoring. In *ACM SenSys*, 2004.