

Cue-based Networking using Wireless Sensor Networks: A Video-over-IP Application

Yeonsik Jeong ^{†‡}, Sriram Lakshmanan^{*}, Sandeep Kakumanu^{*}, and Raghupathy Sivakumar^{*}

^{*}School of Electrical and Computer Engineering, Georgia Institute of Technology, Atlanta, Georgia

[†]Department of Multimedia System Engineering, SungKongHoe University, Seoul, Korea

Email: {ysjeong, sriram, ksandeep, siva}@ece.gatech.edu, ysjeong@mail.skhu.ac.kr

Abstract—In this paper we present a new approach called *cue-based networking* that uses hints or cues about the physical environment to optimize networked application behavior. We define the notion of cues and describe how cues can be obtained using a wireless sensor network as the underlying platform. We identify both the research and system challenges that need to be addressed to realize benefits of the approach under a target application of video delivery over IP networks. In the process, we identify a key challenge of wireless sensor networks, namely the timeliness-robustness tradeoff. We design an adaptive algorithm that balances this tradeoff satisfying both timeliness and robustness requirements. Through an implementation of the video delivery application using the proposed algorithm in a real home environment, we highlight the practical benefits of the proposed approach.

I. INTRODUCTION

Optimizing the behavior of application and network protocols has been a continuing endeavor since the origins of the Internet. In this paper we present a new approach for networked application optimization called *cue-based networking* (CBN). CBN involves the use of *cues* about the characteristics of the target environment for the application to optimize the network-based delivery of the application.

While we define cues in more specificity later in the paper, briefly cues are signals about target environmental characteristics and examples can range from whether a certain wireless channel is currently utilized to whether or not the application user is present in the target environment. The goal of CBN is to use such cues toward more effectively delivering the application over the network. Examples of such use can include from initiating channel switching based on current channel usage to de-prioritizing content when the target user is not currently actively utilizing the content.

Assuming CBNs can deliver enhanced performance, an important question arises: *how can the cues be generated and converted into an application usable form?* In this paper, we present an approach that uses *wireless sensor networks* (WSNs) as the underlying platform for generating cues and converting them into an application usable form. The basis for using WSNs as the underlying platform stems from the fact

that the basic functionality of WSNs is to sense environmental phenomena which is critical for generating cues. Perhaps, equally importantly, the fact that general purpose WSNs sense low level phenomena makes them a prime target for use with a variety of applications that require different types of cues, albeit complex ones. Thus, while general purpose WSNs can serve multiple applications simultaneously, explicit effort has to be given to translating the low level cues to the complex cues usable by applications.

Since the notion of cues, and how the cues can be leveraged by applications are highly application specific, we present CBN in the context of a specific application: video delivery over IP. Video traffic over IP data networks, including the Internet, is rapidly gaining attention because of the impressive loads it can impose on the network. While there are several dimensions along which the application can be optimized, our goal is to show how CBN can considerably improve the performance of video delivery over IP. Specifically, we tackle two well known problems in the application: poor video quality vis-a-vis throughput performance, and high channel zapping delay. We show how cues can be generated using WSNs to appropriately address both the aforementioned problems. Using a prototype implementation in a real-life *aware* home instrumented with a WSN, we demonstrate both the motivation for the proposed approaches and the performance enhancements achieved in using them.

Thus, the contributions of this work are three-fold:

- We present a new approach called CBN that utilizes cues about the target environment to enhance networked application performance. More importantly, we show how WSNs can be used as the underlying platform to facilitate CBNs.
- We develop the CBN solutions in the specific application context of video delivery over IP and show what capabilities the WSNs need to support in order to be able to interface with a real-time application such as video delivery, and how the simple cues generated by WSNs can be translated into more sophisticated cues usable by the application.
- Finally, we demonstrate CBN and how it helps video delivery over IP using a prototype implementation of both the video delivery application and the WSN in a real *aware* home.

This work was supported in part by the National Science Foundation under grants CNS-0721296, CNS-0519733, and CNS-0519841.

[‡]This work was performed when he was visiting the GNAN research group at Georgia Institute of Technology.

The rest of the paper is organized as follows. Section II introduces the concept of CBN and highlights the challenges in a typical networked application. Section III describes overall system architecture and operations. Section IV describes the research challenges and proposes algorithms to solve those challenges. Section V describes the performance evaluation of the implemented system. Section VI presents related work and Section VII concludes the paper.

II. BACKGROUND

A. CBN

In this paper we introduce the notion of cues. Cues are hints about application behavior or useful information relevant to the application, which is not available in a conventional communication model. Specifically, such cues are important information from external sources which affect application performance but are not perceptible to the core communication network infrastructure. We describe the concept using the following illustrative scenarios.

A CBN is one where networked applications utilize cues to enhance application performance. Cues can be derived actively from the network or passively without significant involvement of the network. Cues can either be used to enhance an existing application or to enable new services. As an instance, consider the problem of channel management in wireless local area networks (WLAN). If additional information about external interference sources like a microwave is present, it can be used to take better channel management decisions than just using the information obtained from the WLAN itself. A CBN can also enable new application services such as targeted delivery of content for different users.

Although the CBN model appears to be related to approaches in the fields of ubiquitous or pervasive computing, it is conceptually and architecturally different. First, the central focus of the CBN model is to improve network performance rather than the design of ubiquitous applications. Thus the focus here is on the network. Further, techniques for enabling a ubiquitous computing experience are aimed at seamless experience for the user and not focused on the optimizations that are needed from a communication network standpoint. Thus, new applications in the context of ubiquitous or pervasive computing can also obtain network level benefits by using the CBN model.

B. Use of WSNs for CBN

WSNs consist of a distributed network of sensors which sense various physical phenomena such as light, temperature, humidity, etc. Since they are used to collect information about different events, they can form a natural platform by which one can obtain cues. While one can consider different sensors for sensing different events, a sensible approach here is to design a generic WSN that collects information about various physical phenomena. Also, a single infrastructure can be shared by multiple applications thereby enabling better utilization of information and minimal resource consumption. On the other hand, using a generic sensor network to sense

a variety of events introduces the problem of reliable event detection. So smart techniques must be designed to obtain accurate information from dumb sensors.

Apart from these considerations, the ecosystem of a WSN can consist of two types of sensors, namely *active* and *passive*. Active sensors are independent entities with communication and computing power and spend energy for these operations. On the other hand, passive sensors (such as RFID tags) do not spend energy and do not initiate communication and computing tasks by themselves. In this paper, we consider the use of active sensors only.

C. Video Delivery over IP & Associated Problems

Video delivery over IP networks is becoming an increasingly popular application space, primarily due to the ease with which content can be generated and distributed by anyone without sophisticated help from content providers. Popular instances of such ventures in the industry are YouTube [1] and Joost [2]. With this technology, the choice of the video content is on a per-user device basis rather than an en-masse delivery. While the technology and architecture appear promising, there are several challenges that must be tackled before the vision of Video delivery over IP can be realized. The two main challenges are *bandwidth management* and *channel zapping delay*.

1) *Bandwidth Management*: The path traversed by video content consists of different networks with different underlying technologies and loads. Hence any one or more of these components can cause a bandwidth bottleneck that limits the successful delivery of video. By its nature, video is different from other applications such as the web or email as it consumes significant bandwidth and also demands some level of quality of service support (in the form of data rate, loss rate, and delay jitter). For instance, High Definition TV (HDTV) is envisioned to require around 6 Mbps of sustained bandwidth per stream. Coupled with this, the number of video connections is known to be very high. Statistics in the U.S reveal there are 266 million TVs as of 2005 [3] and the number is growing at the rate of 3.5 Million a year with 3 TVs per house on the average. This means a sustained bandwidth of at least 18 Mbps per home in addition to other applications such as data and voice which share the same links.

2) *Channel Zapping Delay*: It is the time taken for the current video channel stream to end and a new channel to be displayed. This directly impacts user experience. The components that make up the channel zapping delay depend on whether a unicast or multicast video session is used, but consist of mainly I-frame delay and MPEG buffering, and partially stream setup and join time, etc. Threshold values for acceptable channel zapping delay have been identified as around 1 second [4].

III. SYSTEM OVERVIEW AND OPERATIONS

In this section, we discuss the generic system architecture for the target video delivery application that we consider. Further, we also discuss the high-level operations that enable the CBN of the system.

A. System Architecture and Connectivity

Fig. 1 shows a generic architectural topology of video delivery over IP networks and the various components of the associated cue generation network that aids the video delivery network. The various components of the architecture are described below.

1) *Video Delivery Application*: Video is served from several video servers (video head end) to clients at a home through the wired distribution network and the last mile access connection. The last mile connection, which is depicted as an access link in Fig. 1, uses xDSL or cable modem as the technology and possibly Fiber to the Home (FTTH). Inside the home, the distribution to the different TVs can be either through a wired or a WLAN. Each TV has an associated set-top box (STB) that receives the IP packets and converts them into a video stream for the TV. Nowadays more intelligent STBs are being designed to accommodate sophisticated actions such as user preference based content delivery, a popular example of which is Video-on Demand (VoD).

2) *WSNs*: A WSN consists of an ecosystem of active sensors (such as MICA motes [5]) that generate raw sensor information. We assume that the sensors are general purpose sensors and are a part of generic home sensor networks. Different kinds of sensors such as light or accelerometer are mounted on various objects in the home like a remote control, sofas, chairs, and beds to detect light or orientation of an object. All the sensors communicate to a centrally located sink on a wireless channel. The link between an individual sensor and the sink can be either a one hop wireless connection or a multi-hop link through other sensors. The sink sends queries to the different sensors and also collects the information provided by the individual sensors.

3) *Cue Interfaces*: The sink is connected to a base station (BS) that aggregates all the data from the sink. The BS also generates the necessary cues about the user behavior from the raw sensor information. Once the cues are generated they are sent to the STBs associated with the different TVs in different rooms. The STBs make the necessary decisions using the cues to optimize the video delivery network. The STBs also send commands such as play, pause, stop, etc to the video delivery network. Fig. 2 shows how the WSN interfaces with the video delivery network through the BS and sink

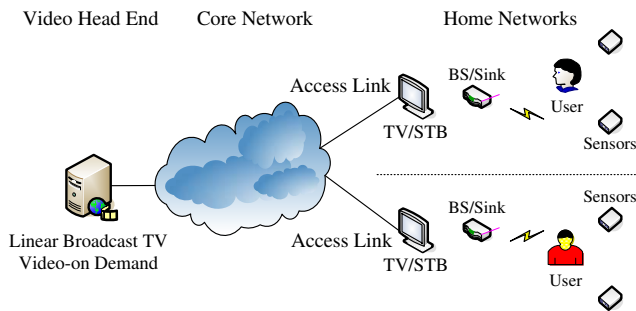


Fig. 1. General system architecture.

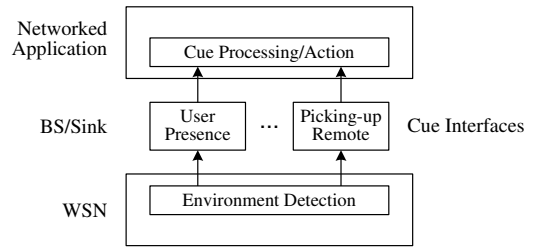


Fig. 2. Cue interface between networked application and WSN.

(cue interfaces). Essentially, the interface can be viewed as a middleware solution for the various problems of the target application that the CBN serves.

B. System Operations

The different components in the system perform different functions. The individual operation performed by each of the components is described below:

1) *WSNs*: The different sensors (light, accelerometer, or temperature) are used to collect raw information about the physical environment. Examples of information that lead to cues are changes in intensity of light, changes in orientation of an accelerometer sensor, and temperature changes. Essentially, this step is used to detect the various physical phenomena that lead to cues. In the current example of video delivery, the different sensors in the room detect the information about the user activities. For example, when a person sits on a sofa, the light intensity at one or more sensors mounted on the sofa changes. Further, the orientation of accelerometer sensors on the sofa also change. The information detected by these sensors is used to generate meaningful cues that can be used by the video delivery application.

2) *Cue Interfaces*: Once the physical phenomena are detected by the sensor devices, the next step is to generate meaningful cues from the information detected. The challenge here is to use a limited number of sensors that provide physical information and take informed decisions about the required application cues. Specifically, we have to represent or obtain information depending on the cue from a combination of multiple sensor values intelligently. This combination is required because a single sensor may not give the complete information required by a cue. This is because the actions that lead to cues are captured through indirect means, using simple (and dumb) sensor devices. The various cues that are used in the video delivery example are: (i) User watching the TV; (ii) Remote control position.

We first discuss the cue about the user watching the TV. For simplicity, let us assume that a user is watching the TV only if he/she is sitting on a sofa. Two different kinds of sensors mounted on the sofa, namely light and accelerometer sensors, provide the information for this cue. Since a single sensor would not reliably capture the event, it is necessary to use more than one sensor. If we suppose that light is used to detect human presence, the ambient lighting conditions will affect the sensor reading. Hence an accelerometer sensor is used to

augment the information provided by the light sensor. Thus for varying ambient conditions, the sensors must be able to provide a fairly accurate detection of user presence. For the cue about the remote control position, an accelerometer sensor mounted on the remote is used to detect if a user is using the remote or not.

While it is true that any generic WSN can be used for generating the cues, there are several key challenges that have to be addressed before the WSN can be integrated into the system. These challenges and their solutions are described in Section IV.

3) *Video Delivery Application*: Once the sensor values are processed to obtain cues, the application must be designed to utilize them properly so that actual benefits can be seen. For instance, consider the user watching the TV. When the presence of user is detected, the TV content should start streaming to the TV. When the absence of user is detected, the streaming should stop (or de-prioritized). This process will allow an efficient management of the network bandwidth, because content is delivered only when actually required. The problem of channel zapping delay was identified earlier. A solution to this problem is to pre-fetch content of adjacent channels before the user actually changes the channel¹. This pre-fetching should be performed only when the user actually wants to switch the channel. We assume that a channel will be changed only when a user picks up the remote. Thus the cue about the remote control position, provided by the accelerometer sensor, can be used to infer the user intent to change channel.

IV. ALGORITHMS AND IMPLEMENTATION

A. *Timeliness and Robustness Tradeoff*

1) *Basic Strategies*: In this section we highlight the important challenges that occur when a WSN is used for realizing the benefits of CBNs. We discuss two commonly used data collection strategies in WSNs namely, *continuous reporting* and *event-driven reporting*, and describe why each of these in isolation is not useful for the CBNs.

In continuous reporting, each sensor transmits the sensed values periodically to the sink. This model has the advantage that the sink has a complete picture of the network at all times. This advantage incurs the cost of all nodes sending messages continuously. In event-driven reporting, sensors transmit information only if they perceive a change in the sensed value. Usually a threshold is set and when the sensed information changes beyond the threshold, the sensor reports the change to sink. The advantage is that information is sent only when required. A hidden requirement of this model is that a clear static threshold must be known a priori. Consequently, if either the threshold varies with different factors or the event of interest is not known a priori, this model cannot be directly applied. Also, from an energy perspective, event-driven reporting consumes lesser energy when compared to continuous reporting, due to the reduced number of transmissions.

¹Designing an intelligent pre-fetching algorithm is beyond the scope of this work. We refer the interested reader to [6] for one such algorithm.

2) *Timeliness*: Timeliness is defined as the property of detecting an event and conveying it to the sink with the minimum delay. The total delay is composed of sensing, processing, and transmission delay. Of these, sensing and processing delay are functions of the sensor hardware. From a communication standpoint, the relevant component is transmission delay. The transmission delay depends on the packet size used, the operating data rate of the node, and especially congestion in the network. For continuous reporting, the event detection time depends on both the reporting period and the time taken for packets to reach the sink. When the reporting rate is low, the dominating factor is reporting period, and when the rate gets high, contention in the network lead to increased packet transmission times. Thus in either case, continuous reporting has problems. On the other hand, event-driven reporting incurs minimum delay owing to the fact that packets are sent if and only if an event occurs. This reduces the contention delays in the network as there are fewer packets.

3) *Robustness*: Robustness is defined as the property of detecting an event of interest reliably even in the presence of other sources that affect detection. Robustness is a significant challenge when at least one of the following conditions exist: (i) The ambient conditions vary, thus affecting the phenomenon of interest; (ii) Other undesired sources such as unintended interferences corrupt the measurement; (iii) Sensors do not have sensitivity for all regions of interest. All of these have enormous practical significance and a practical solution must address challenges arising out of the above factors. Continuous reporting achieves high robustness because all data sensed by the sensors is reported to the sink. The sink can use all the information and process them jointly to eliminate/reduce the effects of the above mentioned factors. Event-driven reporting does not have a high robustness. Specifically, each sensor performs a local decision and reports only when a threshold is crossed, whereby the ability to perform a global decision is lost.

4) *Tradeoff Issues*: It can be observed that continuous reporting model unnecessarily increases detection time although it allows achieving of higher robustness. On the other hand, event-driven model achieves significantly reduced detection delay at the cost of unreliable detection. Hence neither of these approaches by themselves are sufficient for solving the problem at hand.

5) *Experimental Results*: To validate the above arguments, we perform measurements in a real-life setting. MICAz motes from Crossbow Technology [5] are used as the sensor nodes. All sensors are directly connected to the sink and the *Surge* application [7] is used for the application. For the continuous reporting, we use the default code by just changing the *initial timer rate* variable. For event-driven reporting, we add the thresholding logic to the code. MICAz datasheets specify a possible minimum reporting interval of 300 ms.

Fig. 3 describes the event detection time as a function of the reporting rate. For this experiment twenty sensor nodes are used. It can be observed that as the reporting rate is increased, the event detection time is reduced. However, as

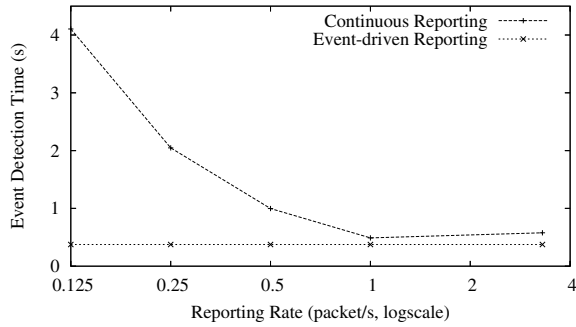


Fig. 3. Event detection time according to reporting rate.

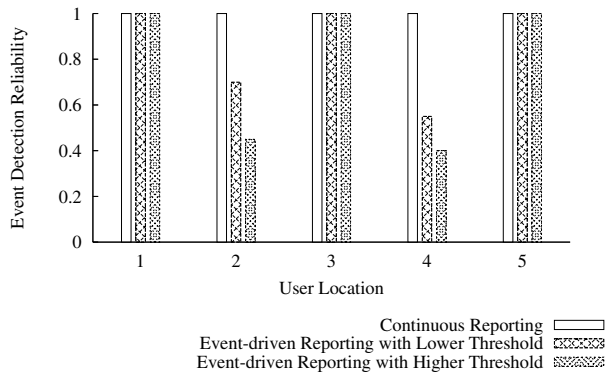


Fig. 4. Event detection reliability according to user location.

we increase the reporting rate to the maximum (i.e. reporting interval of 300 ms), an event detection latency of about 580 ms is observed (which is slightly greater than that of reporting rate of 1). In the worst case, up to 3 second of latency is observed, which is unacceptable for the real-time application. Thus, when the number of nodes is high, packet delivery takes much longer for each node on the average. Further, energy drain is higher with the maximum reporting rate. The event detection time for event-driven is also shown. It can be observed that it has a constant and low detection delay.

Fig. 4 depicts the robustness results. Three MICAz sensors are placed on a sofa at uniform distances. Five positions of person sitting are identified on the sofa. (See Fig. 5.) This experiment is performed on each position for each reporting scheme, and for each trial twenty events are repeated. As indicated, the positions where the sensor location and event location are exactly aligned, the detection is good for all schemes. The difference between the schemes becomes clear from the intermediate positions (for example, location 2 and 4) where using event-driven with two kinds of thresholds causes a very low success rate because each sensor value has not crossed the threshold. On the other hand, with continuous reporting, the sink uses the sensor values of both sensors and uses the topology information to reliably detect that the person is sitting on the sofa. Fig. 4 clearly demonstrates the robustness problem of event-driven schemes and the improved performance of continuous reporting schemes.

With a joint observation of the two graphs, one can observe that the trends for continuous and event-driven schemes are different for the two metrics. This clearly highlights the timeliness-robustness tradeoff. Consequently if each of these techniques is used in isolation, it is ineffective to achieve the goal of cue generation. Hence to simultaneously achieve the best benefits of both techniques, an intelligent scheme is needed, which ensures robust detection while keeping the delay to a minimum. Such an intelligent scheme should use multiple sensor observations when and only when needed. Also the reporting should be only in response to events detected by the sensor.

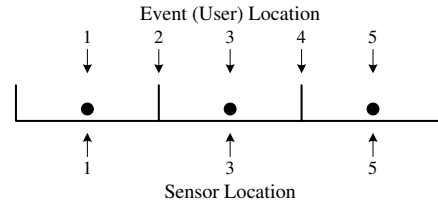


Fig. 5. Illustration of event and sensor location.

B. Adaptive Probabilistic Reporting

1) *Overview:* We now propose and discuss an *adaptive probabilistic reporting* scheme to solve the challenges. The main idea is to adapt transmission decisions based not just on each sensor's perception but also on the information overheard from transmission of other sensors. Depending on the value overheard, each sensor can choose whether to transmit or not. In this way, the sink will receive multiple messages when needed so that robustness is enhanced. Overhearing is used to facilitate more intelligent reporting than separate reporting by each sensor. The probabilistic nature ensures that a distributed mechanism can be designed.

To clarify the idea further, consider the Fig. 5 again. When a person occupies an intermediate position where neither of these sensors can determine user presence individually, the transmission of both sensors would be needed (assuming a certain light threshold to determine if a person is sitting on a sofa or not). We need both the sensors adjoining the event to report. Thus, for cases when the event is clearly detected by one sensor, it is sufficient for that sensor alone to transmit. However, for events that are only partially detected, we need multiple sensors to transmit with increasing probabilities. The increasing probability is because when a single sensor reports a partial detection, it is likely that the other sensor values are necessary to determine if it is truly an actual event (such as user sitting) or a stray incident (such as a book thrown on the sofa). In this way, false positives are minimized. Similarly, the improved reliability provided by multiple sensor readings helps to minimize false negatives.

2) *Algorithm Description:* The reporting condition can be formally described as follows. Each sensor transmits a message with a probability proportional to the intensity of the event to be detected. Also, this probability is updated when

Constants/Variables:

```

1 MAX_DELTA: maximum difference between sensor values
2 MAX_RAND: maximum random number
3 sensorStatus: current sensor status, S_Init or S_Pending
4 curMaxDelta: current maximum delta for probabilistic transmission
5 curVal, prevVal: current and previous sensor value in sensor
6 deltaVal: difference between the current and previous sensor values
7 overheardDeltaVal: delta value overheard from other sensors

```

Functions:

```

8 s_decideTransmission() {
9   get a random number, randNum;
10  if randNum/MAX_RAND < min(1, deltaVal/curMaxDelta)
11    sensorStatus ← S_Init;
12    prevVal ← curVal;
13    transmit curVal and deltaVal to the sink;
14  else
15    reset the timer if there is a timer set before;
16    sensorStatus ← S_Pending;
17    set a timer;
18 } // probabilistic transmission function in sensor

```

```

19 s_overhearHandler() {
20   if sensorStatus == S_Pending
21     curMaxDelta ← curMaxDelta - overheardDeltaVal;
22   s_decideTransmission();
23 } // interrupt handler for packet overhearing in sensor

```

```

24 s_timerHandler() {
25   sensorStatus ← S_Init;
26   prevVal ← curVal;
27 } // interrupt handler for the timer set in sensor

```

Main Procedure:

```

// interrupted by s_overhearHandler() and s_timerHandler()
28 sensorStatus ← S_Init;
29 while reading a sensor value
30   if sensorStatus == S_Init
31     curMaxDelta ← MAX_DELTA;
32     deltaVal ← curVal - prevVal;
33   s_decideTransmission();

```

Fig. 6. Adaptive probabilistic reporting algorithm at sensor.

other nodes transmit. Thus if ΔS_i is the change in sensed value observed by sensor node i and there are totally n sensors which detect this event, the probability of transmission is given as

$$P_i = \begin{cases} \min(1, \frac{\Delta S_i}{\text{TH} - \sum_{k=1, k \neq i}^n \Delta S_k}) & \text{if } \sum_{k=1, k \neq i}^n \Delta S_k < \text{TH}, \\ 0 & \text{else,} \end{cases} \quad (1)$$

where TH is a threshold value set once when the WSN is designed. This reporting condition ensures probabilistic transmission such that multiple sensors transmit when the event is not detected strongly by any of the sensors. If the event is detected strongly by a single sensor, only that sensor reports. (i.e. If $\sum \Delta S_k \geq \text{TH}$, then $P_i = 0$.) But as long as one sensor detects partially, the other overhearing sensors transmit with increased probability.

The proposed technique has the following major differences compared to alternate techniques such as event-driven reporting using a low threshold. When using a low threshold for detection, the sensitivity of the system is increased thereby leading to transmission of much more packets than required. Transmission of a large number of packets affects energy and lifetime of the nodes apart from increasing contention and associated medium access delay. This could be severe when the deployment consists of a large number of sensors forming a generic sensor network. Further, the use of distributed collaboration is a distinguishing feature of the technique. Consequently, extensive calibration for getting exact value of threshold is not required by the proposed scheme.

The aim of the proposed algorithm is to perform an intelligent reporting that provides energy and delay benefits close to that of the event-driven model but also provides significant reliability using multiple sensor views. It also addresses the practical challenge of cumbersome calibration by the user by intelligently adapting transmissions based on observed sensor values. The central idea is to utilize sensor pattern information

along with the topological information to make intelligent decisions on the occurrence or non-occurrence of an event. The algorithm is described in detail in Fig. 6.

3) *Extensions:* The algorithm defined above can be extended in the following ways.

- *Aggregation Function:* In the algorithm, a linear aggregation function is assumed for the sensor pattern. The sensed values are such that for different positions of the event, the aggregated value crosses a threshold for legitimate events and lies below for illegitimate events. The aggregation function of linear summation is verified to work for the present case of using light sensors in Section V. However, for different sensors this law could be different.
- *Virtual Sensors:* Here the transmission probability proportional to the change in sensor values is used. This is found to be satisfactory, when the number of sensors is large. However, when the number of sensors is small, the probability of not transmitting in response to an event might still be non-negligible. In this case, each sensor could be considered to have a number of virtual sensors. The idea of virtual sensing is to consider many virtual sensors in place of a single physical sensor with the sensed value equally divided between them. Now the probability of transmission if carried over these virtual sensors will be high, thereby reducing missed detections. In Section V-C we call this scheme *enhanced adaptive probabilistic reporting*.
- *Correlation Awareness:* The sink needs to be aware of the correlation in data sent by different sensors, which are nearby and sensing the same events. For instance, if two sensors are deployed physically close to each other, there might not be new information gained by using both observations than a single observation. This way it can prevent a wrong decision by only considering those

sensors that have independent views of the event.

V. PERFORMANCE EVALUATION

In this section, we discuss the testbed setup for our implementation and discuss the performance evaluation of the proposed algorithms using the testbed.

A. Prototyping Testbed Setup

All our implementations have been made in the Aware Home at Georgia Institute of Technology. The Aware Home Research Initiative (AHRI) [8] is an interdisciplinary research endeavor at Georgia Tech aimed at addressing the fundamental technical, design, and social challenges in a home environment that is aware of its occupants whereabouts and activities. The physical deployment of the various devices, used in the testbed, is shown in the floorplan diagram of the Aware Home in Fig. 7. Three TVs are used in three different rooms (living room, Bedroom 1, and Bedroom 2). Three laptops, running Fedora Core 7 Linux, are used as the STBs serving these TVs. Another desktop, running Fedora Core 7, is used as the video streaming server. A Network NightMare network emulator [9] is used to emulate the Internet. The video content is streamed to the STBs in the different rooms using an 802.11g AP. A fourth laptop, running Windows XP, works as the BS (described in Section III-A). MICAz motes are used as the wireless sensor devices and operate on a frequency band of 2.4 GHz. Light and accelerometer sensors are used for the generating the necessary cues. They are mounted on the sofa in the living room, on a chair in Bedroom 1, and on the remote for the TV in the living room. The optimizations for bandwidth management and zapping delay are performed only for the TVs in the living room and Bedroom 1. The TV in the bedroom 2 is used only for limiting bandwidth of the 802.11g wireless channel. MPEG-2 is used to encode raw video files and MPEG-4 file format is used to stream MPEG-2 encoded videos.

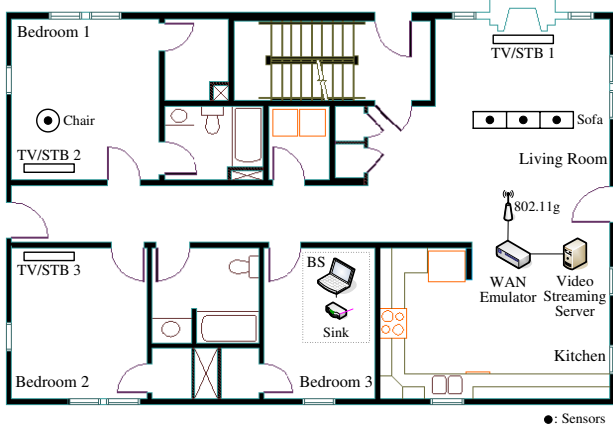


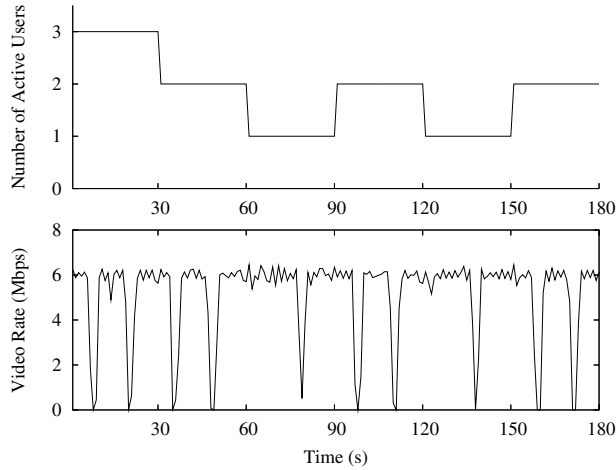
Fig. 7. Physical deployment of prototyping testbed in Aware Home.

B. Macroscopic Results

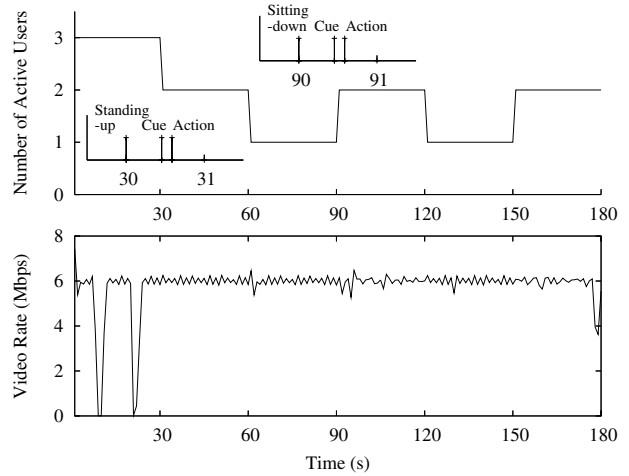
In this section we discuss the various macroscopic results that are indicative of the optimization benefits of CBN for the video delivery application.

1) *Bandwidth Management*: We study the bandwidth management problem at the wireless leg of the video delivery. We have three TVs in the house, and 6 Mbps HDTV video streams are sent to each TV over a wireless 802.11g channel. Fig. 8(a) shows the number of active TVs (that is, TVs with people watching content), as function of time and also the corresponding video display rate of one active TV without the optimizations. We assume that a user does not switch off a TV when he/she is not watching the TV. Thus without the optimization, packets belonging to content for all the three TVs are sent over the wireless channel. The capacity of the wireless channels is not sufficient for all three TVs. This is evident from the fact that data rate drops every few seconds. Fig. 8(b) shows the same result with CBN turned on. We see that when the number of active TVs is less than three, the video rate drops are eliminated. This is because, only the required number of streams are sent over the wireless channel and the capacity is sufficient for one or two streams. Fig. 8(b) also shows the various events in the CBN when the person sits on or stands up from the sofa. These events are shown in the magnification graph inset in the top part of the figure. The event shown as cue is the actual cue generation process at the base station and the event denoted as action is the application preprocessing event.

2) *Channel Zapping Delay*: Fig. 9(a) shows two events of changing channels at two different time instants and their corresponding video rate of the TV. When the user changes the channels, it takes about 3.5 seconds to display the new channel. As discussed before, the major components of this zapping delay are the video buffering delay at the TV and the various connection establishment times. An accelerometer sensor, mounted on the remote for the TV in the living room, is used for generating the cue that aids in reducing the zapping delay between channels. Assuming that when a user picks up the remote, the other channel is pre-fetched. Fig. 9(b) shows the time taken to switch the channel, when the cue is used. The results indicate that the CBN reduces the zapping delay considerably. The various events that occur during the channel zapping are indicated as different events in Fig. 9(b). Correspondingly the video rate is shown in the bottom part of the same figure. When the remote is picked up, the other channel is pre-fetched in the background. This is evident from the increased video rate at the TV. When the channel is actually switched, it happens seamlessly, and the time required to change the channel is about 800 ms. When the remote is put down, pre-fetching stops. A small figure in the bottom part of Fig. 9(b) also shows the network rate when the different events occur. We can see that during pre-fetching the network rate doubles because two streams are fetched at the same time.

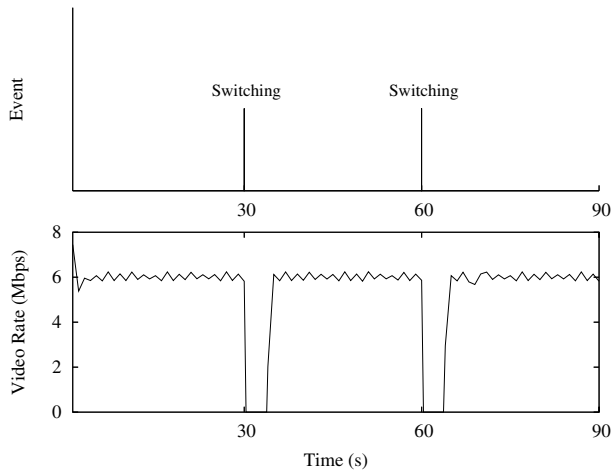


(a) Without optimization

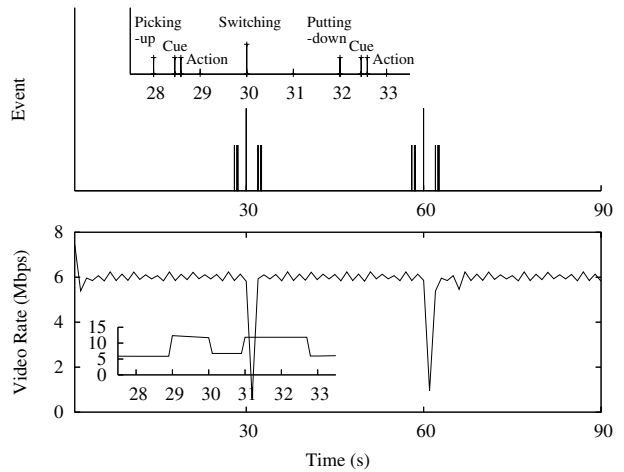


(b) With optimization

Fig. 8. Video rate of one specific TV vs the number of users, with three TVs on.



(a) Without optimization



(b) With optimization

Fig. 9. Video rate of one specific TV with channel zapping.

C. Microscopic Results

We discuss microscopic results of the system solution and present insights that enable us to gain a deeper understanding of system internals and also practical issues. The focus of this section is on the cue generator (sensor) network and primarily concerns the robustness and timeliness of detection and how the proposed algorithm addresses the challenges. The same setup of the sensors as in Section IV-A5 is used for showing the benefits of the proposed adaptive probabilistic reporting algorithm.

1) *Timeliness*: Fig. 10 adds the time taken to report an event using the proposed algorithm to the previous results of the continuous and event-driven reporting. We can see that the time taken is very similar to the case of simple event-driven approach. The adaptive algorithm requires one of the sensor to overhear a packet from the neighboring sensors and send an

additional packet. The results indicate that this overhearing is not time consuming. This is evidenced by the fact that the pure event-driven reporting that does not need overhearing requires almost the same amount of time for the event detection.

2) *Robustness*: Fig. 11 adds the reliability results of the proposed algorithm to the reliability results of continuous and event-driven reporting. The event detection reliability of the adaptive probabilistic reporting scheme does not have 100% reliability when the user sits in one of the intermediate positions. We observe that the reliability of the enhanced adaptive probabilistic algorithm is always 1 for all locations of the user. Note that the enhanced adaptive probabilistic reporting is based on the use of virtual sensors described in Section IV-B3. Thus we conclude that the adaptive algorithm has the reliability of the continuous reporting.

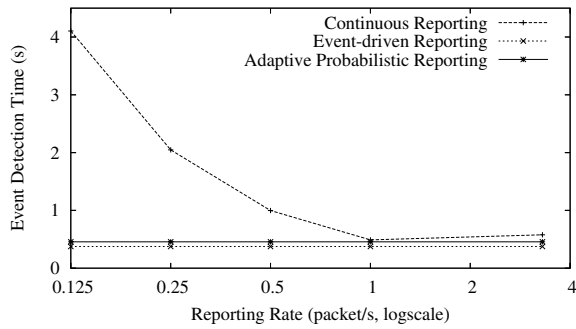


Fig. 10. Event detection time according to reporting rate.

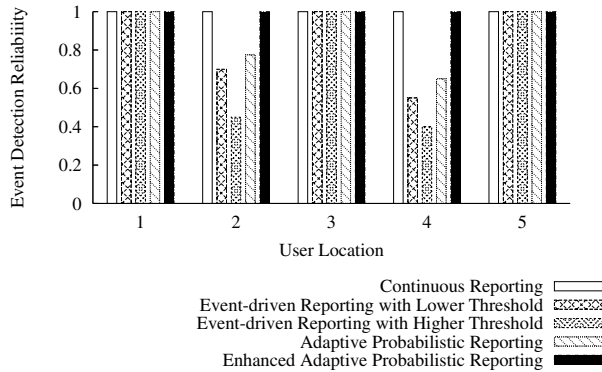


Fig. 11. Event detection reliability according to user location.

VI. RELATED WORK

A few representative works in the field of ubiquitous computing are [10], [11], [12], [13]. In almost all the works, context or sensor data is often used as input for applications running as a layer on top of the network stack. Thus the network is used only as a transport medium for the sensed data. However, in the CBN approach, we argue that a communication network can itself benefit from intrinsic processing of context information, provided as cues by the sensors. The work that is closest to the CBN approach is [14], where the authors discuss context aware networking. However, the work only uses contexts within the network such as available energy resources, processor load, and link quality.

Video delivery over wireless has received a lot of attention of late. A significant amount of work looks at optimizing video delivery at the application layer itself by adapting the video encoding rate [15]. Some works also look optimizing the lower layers of the network stack to aid video delivery [16], [17]. In particular, [16] discusses the various optimizations at the transport layer and routing layers, to optimize video performance over wireless networks. In [17], the authors discuss a cross layered signaling approach for optimized video streaming over 802.11 networks.

We assume that a generic sensor network in the home can help several CBN applications. To support several applications on a such a single deployed sensor network a middleware solution is required. Research towards such a goal has been

conducted in works such as [18], [19].

VII. CONCLUSION

In this paper, we have introduced a new model for networked application performance enhancement called *cue-based networking*. In this model, network applications use external hints to significantly enhance performance and enable new services. We have described an architecture where a wireless home sensor network is used to generate cues to a video delivery application. We have also discussed the design of algorithms and their implementation to demonstrate the feasibility of applying the proposed model. Finally, our goal has been to introduce this new approach of networked application performance enhancement and demonstrate the benefits in a real-life setting.

REFERENCES

- [1] YouTube. [Online]. Available: <http://www.youtube.com>
- [2] Joost: Free Online TV. [Online]. Available: <http://www.joost.com>
- [3] [Online]. Available: <http://www.csmonitor.com/2005/0616/p13s02-stct.html>
- [4] "How network conditions impact IPTV QoE," Agilent Technologies, Tech. Rep. [Online]. Available: <http://cp.literature.agilent.com/litweb/pdf/5989-6143EN.pdf>
- [5] Crossbow Technology. [Online]. Available: <http://www.xbow.com>
- [6] J. Caja, "Optimization of IPTV multicast traffic transport over next generation metro networks," in *Proc. Telecommunications Network Strategy and Planning Symposium*, Nov. 2006.
- [7] Surge Application. [Online]. Available: http://www.tinyos.net/tinyos-1.x/doc/multihop/multihop_routing.html
- [8] Aware Home Research Institute. [Online]. Available: <http://awarehome.imtc.gatech.edu>
- [9] Network Nightmare: Network Emulator. [Online]. Available: <http://www.networknightmare.net>
- [10] M. J. Dong, G. Yung, and W. J. Kaiser, "Low power signal processing architectures for network microsensors," in *Proc. ISLPED*, Aug. 1997, pp. 173–177.
- [11] W. Su and I. F. Akyildiz, "A stream enabled routing (SER) protocol for sensor networks," in *Proc. Med-hoc-Net*, Sep. 2002.
- [12] J. Su, J. Scott, P. Hui, E. Upton, M. H. Lim, C. Diot, J. Crowcroft, A. Goel, and E. d. Lara, "Haggle: Seamless networking for mobile applications," in *Proc. UbiComp*, Sep. 2007.
- [13] S. Davidoff, M. K. Lee, C. Yiu, J. Zimmerman, and A. Dey, "Principles of smart home control," in *Proc. UbiComp*, Sep. 2006.
- [14] M. Beigl, A. Krohn, T. Zimmer, C. Decker, and P. Robinson, "AwareCon: Situation aware context communication," in *Proc. UbiComp*, Oct. 2003.
- [15] P. V. Beek and M. U. Demircin, "Delay-constrained rate adaptation for robust video transmission over home networks," in *Proc. IEEE ICIP'05*, Sep. 2005, pp. 173–176.
- [16] X. Zhu and B. Girod, "Video streaming over wireless networks," in *Proc. EUSIPCO'07*, Sep. 2007.
- [17] L. Haratcherev, J. Taal, K. Langendoen, R. Lagendijk, and H. Sips, "Optimized video streaming over 802.11 by cross-layer signaling," *IEEE Commun. Mag.*, vol. 44, no. 1, pp. 115–121, Jan. 2006.
- [18] Y. Yu, B. Krishnamachari, and V. K. Prasanna, "Issues in designing middleware for wireless sensor networks," *IEEE Netw.*, pp. 15–21, Jan. 2004.
- [19] Y. Yu, L. J. Rittle, V. Bhandari, and J. B. LeBrun, "Supporting concurrent applications in wireless sensor networks," in *Proc. ACM SenSys'06*, Nov. 2006, pp. 139–152.