Multi-modal Target Tracking using Heterogeneous Sensor Networks

Manish Kushwaha, Isaac Amundson, Peter Volgyesi, Parvez Ahammad†,
Gyula Simon‡, Xenofon Koutsoukos, Akos Ledeczi, Shankar Sastry†

Institute for Software Integrated Systems
Department of Electrical Engineering
and Computer Science
Vanderbilt University
Nashville, TN, USA
Email: manish@isis.vanderbilt.edu

†Department of Electrical Engineering
and Computer Science
University of California - Berkeley
Berkeley, CA, USA

‡Department of Computer Science
University of Pannonia
Veszprem, Hungary

***Abstract*— The paper describes a target tracking system running on a Heterogeneous Sensor Network (HSN) and presents results gathered from a realistic deployment. The system fuses audio direction of arrival data from mote class devices and object detection measurements from embedded PCs equipped with cameras. The acoustic sensor nodes perform beamforming and measure the energy as a function of the angle. The camera nodes detect moving objects and estimate their angle. The sensor detections are sent to a centralized sensor fusion node via a combination of two wireless networks. The novelty of our system is the unique combination of target tracking methods customized for the application at hand and their implementation on an actual HSN platform.**

I. Introduction

Heterogeneous Sensor Networks (HSN) are gaining popularity in diverse fields [23]. They are natural steps in the evolution of Wireless Sensor Networks (WSN) because they can support multiple, not necessarily concurrent, applications that may require diverse resources. Furthermore, as WSNs observe more complex phenomena, multiple sensing modalities become necessary. Different sensors can have different resource requirements in terms of processing, memory, or bandwidth. Instead of using a network of homogeneous devices supporting resource intensive sensors, an HSN can have different nodes for different sensing tasks.

Target tracking is an application that can benefit from multiple sensing modalities [6]. If the moving object emits some kind of sound then both audio and video sensors can be utilized. These modalities can complement each other in the presence of high background noise that impairs the audio or visual clutter affecting the video.

In this paper, we describe our ongoing work in target tracking in urban environments utilizing an HSN of mote class devices equipped with acoustic sensor boards and embedded PCs equipped with web cameras. The moving targets to be tracked are vehicles emitting engine noise. Our system has many components including audio processing, video processing, WSN middleware services, and multi-modal sensor fusion based on a sequential Bayesian estimation framework. While some of these components are not necessarily novel, their composition and implementation on an actual HSN requires addressing a number of significant challenges.

We have implemented audio beamforming on mote class devices utilizing an FPGA-based sensor board and we have evaluated the performance of the audio nodes as well their energy consumption. While we are using a standard object detection algorithm based on video, using post-processing of the measurements, we allow the fusion of audio and video measurements. Further, we have extended time synchronization techniques to HSN consisting of a mote and a PC network. Finally, the main challenge we addressed is system integration as well as making the system work on the actual platform in a realistic deployment scenario. The paper provides results gathered in an uncontrolled urban environment and presents a thorough evaluation including a comparison of different fusion approaches for different combination of sensors.

The rest of the paper is organized as follows. The next section describes the overall system architecture. It is followed by the description of the audio and then the video processing approach. In Section V we present the time synchronization approach for HSNs. Next the multimodal tracking algorithm is presented. The experimental evaluation is described in Section VII followed by a summary of related work. Finally, we discuss the lessons learned and future directions.

II. Architecture

Figure 1 shows the system architecture. The audio sensors, consisting of MICAz motes with acoustic sensor boards equipped with 4-microphone array, form an 802.15.4 network. The video sensors are based on Logitech QuickCam Pro 4000 cameras attached to OpenBrick-E Linux embedded PCs. These video sensors, the mote-PC gateways, the sensor fusion node and the reference broadcaster for time synchronization are all PCs forming a peer-to-peer 802.11b wireless network.

The audio nodes perform beamforming. The detections are sent to the corresponding mote-PC gateway utilizing a multi-hop message routing service that also performs time translation

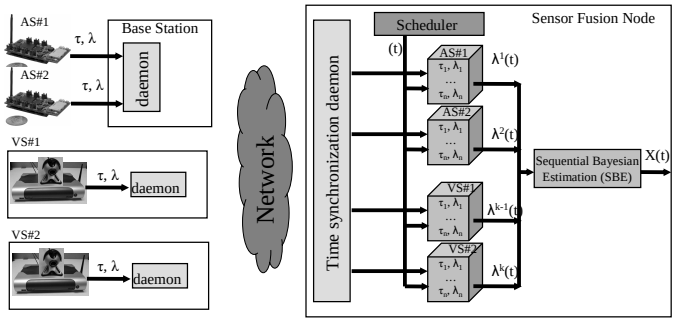

Fig. 1. Multimodal tracking system architecture

of the detection timestamps. The video sensors run a motion detection algorithm and compute timestamped detection functions. Both audio and video detections are routed to the central sensor fusion node and stored in appropriate sensor buffers (one for each sensor). In Figure 1, τ denotes measurement timestamps, λ denotes detection functions described in later sections, and t denotes time. The blocks shown inside the sensor fusion node are circular buffers that store timestamped measurements. A sensor fusion scheduler triggers periodically and generates a fusion timestamp. The trigger is used to retrieve the sensor measurement values from the sensor buffers with timestamps closest to the generated fusion timestamp. The retrieved sensor measurement values are then used for multimodal fusion based on sequential Bayesian estimation.

III. Audio Beamforming

Beamforming is a spacetime operation in which a waveform originating from a given source is received at spatially separated sensors and combined in a time-synchronous manner [4]. In a typical beamforming array, each of the spatially separated microphones receives a phase-shifted source signal. The amount of phase-shift at each microphone in the array is dependent on the microphone arrangement and the location of the source. A typical delay-and-sum beamformer divides the sensing region into directions, or *beams*. For each beam, assuming the source is located in that direction, the microphone signals are delayed according to the phase-shift and summed together into a composite signal. The square-sum of the composite signal is the beam energy. Beam energies are computed for each of the beams, and are collectively called the *beamform*. The beam with maximum energy indicates the direction of the acoustic source.

Beamforming Algorithm: The data-flow diagram of our beamformer is shown in Figure 2. The amplified microphone signal is sampled at a high sampling frequency (100 KHz) to provide high resolution for the delay lines, which is required by the closely placed microphones. The raw signals are filtered to remove unwanted noise components and provide band-limited signals for down sampling at a later stage. The signal is then fed to a tapped delay line (TDL), which has M different outputs to provide the required delays for each of the M beams. The delays are set by taking into consideration the exact relative positions of the four microphones so that the resulting beams are steered to angles $\theta_i = i\frac{360}{M}$ degrees, $i = 0, 1, ...M-1$. The signal is downsampled and the M beams

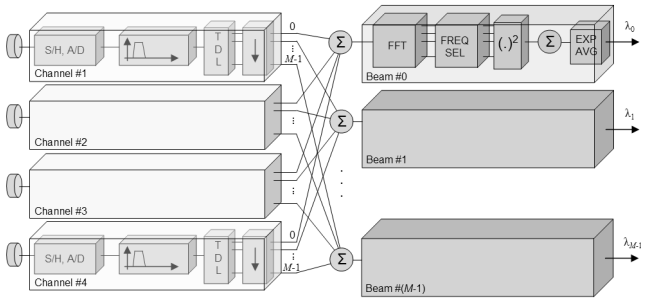

Fig. 2. Data-flow diagram of the real-time beamforming sensor

are formed by adding the four delayed signals together. Data blocks are formed from the data streams (with a typical block length of 5-20ms) and an FFT is computed for each block. A programmable selector selects the required components from the computed frequency lines to compute the total power of the block. The frequency selection procedure provides flexibility for handling different kinds of sources, with its particular frequency spectrum. Note however, that the fixed microphone topology has significant impact on the performance in different frequency bands. The selector can be reprogrammed at runtime to adapt to the nature of the source. The block power values, $\mu(\theta_i)$, are smoothed by exponential averaging into the beam energy:

$$\lambda^t(\theta_i) = \alpha\lambda^{t-1}(\theta_i) + (1-\alpha)\mu(\theta_i) \quad (1)$$

where α is an averaging factor.

Audio Hardware: In our application, the audio sensor node is a MICAz mote with an onboard Xilinx XC3S1000 FPGA chip that is used to implement the beamformer [22]. The onboard Flash (4MB) and PSRAM (8MB) modules allow storing raw samples of several acoustic events. The board supports four independent analog channels, featuring an electret microphone each, sampled at up to 1 MS/s (million samples per second). A small beamforming array of four microphones arranged in a $10cm \times 6cm$ rectangle was placed on the sensor node, as shown in Fig. 3. Since the distances between the microphones are small compared to the possible distances of sources, the sensors perform far-field beamforming. The sources are assumed to be on the same two-dimensional plane as the microphone array, thus it is sufficient to perform planar beamforming by dissecting the angular space into M equal angles, providing a resolution of $360/M$ degrees. In the experiments, the sensor boards were configured to perform simple delay-and-sum-type beamforming in real time with $M = 36$, and an angular resolution of 10 degrees. Finer resolution increases the communication requirements.

Messages containing the audio detection functions require 83 bytes, and include node ID, sequence number, timestamps, and 72 bytes for 36 beam energies. These data are transmitted through the network in a single message. The default TinyOS message size of 36 bytes was changed to 96

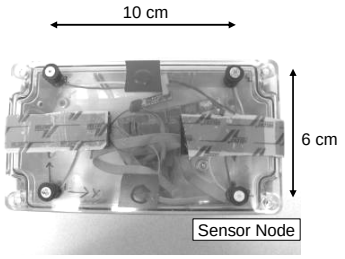

Fig. 3. Sensor Node Showing the Microphones

bytes to accommodate the entire audio message. The current implementation uses less than half of the total resources (logic cells, RAM blocks) of the selected mid-range FPGA device. The application runs at 20 MHz, which is relatively slow in this domain—the inherently parallel processing topology allows this slow speed. Nonetheless, the FPGA approach has a significant impact on the power budget, the sensor draws 130mA current (at 3.3 V) which is nearly a magnitude higher then typical wireless sensor node power currents.

IV. Video Tracking

Video tracking systems seek to automatically detect moving objects and track their movements in a complex environment. Due to the inherent richness of the visual medium, video based tracking typically requires a pre-processing step that focuses attention of the system on regions of interest in order to reduce the complexity of data processing. This step is similar to the visual attention mechanism in human observers. Since the region of interest is primarily characterized by regions containing moving objects, robust motion detection is a key first step in video tracking. A simple approach to motion detection from video data is via frame differencing. It compares each incoming frame with a background model and classifies the pixels of significant variation into the foreground. The foreground pixels are then processed for identification and tracking. The success of frame differencing depends on the robust extraction and maintenance of the background model. Performance of such techniques tends to degrade when there is significant camera motion, or when the scene has significant amount of change.

There exist a number of challenges for the estimation of robust background model including gradual illumination changes (e.g. sunlight), sudden illumination changes (e.g. lights switched on), vacillating backgrounds (e.g. swaying trees), shadows, sleeping person phenomenon, waking person phenomenon, visual clutter, and occlusion [21]. Because there is no single background model that can address all these challenges, the model must be selected based on application requirements.

Algorithm: The dataflow in Figure 4 shows the motion detection algorithm and its components used in our tracking application. The first component is background-foreground segmentation of the currently captured frame (I_t) from the camera. We use the algorithm described in [12] for background-foreground segmentation. This algorithm uses an adaptive background mixture model for real-time background and foreground estimation. The mixture method models each background pixel as a mixture of K Gaussian distributions. The algorithm provides multiple tunable parameters for desired performance. In order to reduce speckle noise and smooth the estimated foreground (F_t), the foreground is passed through a median filter. In our experiments, we used a median filter of size 3×3.

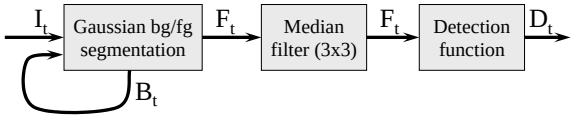

Fig. 4. Data-flow diagram of real-time motion detection algorithm

Since our sensor fusion algorithm (Section VI) utilizes only the angle of moving objects, we use video sensors as directional devices. We essentially convert the 2D image frame to a 1D detection function that captures the angle information of moving objects. This approach provides us with a 30 times reduction in communication bandwidth usage, while on the downside, we lose a dimension of information that can potentially be used in fusion.

Similar to the beam angle concept in audio beamforming (Section III), the field-of-view of the camera is divided into M equally-spaced angles

$$\theta_i = \theta_{min} + (i-1)\frac{\theta_{max} - \theta_{min}}{M} : i = 1, 2, ..., M$$

where θ_{min} and θ_{max} are the minimum and maximum field-of-view angles for the camera. The detection function value for each beam direction is simply the number of foreground pixels in that direction. Formally, the detection function for the video sensors can be defined as

$$\lambda(\theta_i) = \sum_{j \in \theta_i}^{W} \sum_{k=1}^{H} F(j,k) : i = 1, 2, ..., M \qquad (2)$$

where F is the binary foreground image, H,W are the vertical and horizontal resolutions in pixels, and $j \in \theta_i$ indicates columns in the frame that fall within angle θ_i.

Video Post-processing: In our experiments, we gathered video data of vehicles from multiple video sensors from an urban street setting. The data contained a number of real-life artifacts such as vacillating backgrounds, shadows, sunlight reflections and glint. The algorithm described above was not able to filter out such artifacts from the detections. We implemented two post-processing filters to improve the detection performance. The first filter removes any undesirable persistent background. The second filter removes any sharp spikes (typically caused by sunlight reflections and glint). For this we convolved the detection function with a small linear kernel to add a blurring effect.

We implemented the motion detection algorithm using OpenCV (open source computer vision) library. Our motion

detection algorithm implementation runs at 4 frames-per-second and 320×240 pixel resolution. The number of beam angles is $M = 160$.

V. Time Synchronization

In order to seamlessly fuse time-dependent audio and video sensor data for tracking moving objects, participating nodes must have a common notion of time. Although several microsecond-accurate synchronization protocols have emerged for wireless sensor networks (e.g. [7], [9], [16], [18]), achieving accurate synchronization in a *heterogeneous* sensor network is not a trivial task. We employ a hybrid approach, which pairs a specific network with the synchronization protocol that provides the most accuracy with the least amount of overhead.

Mote Network: We used Elapsed Time on Arrival (ETA) [13] to synchronize the mote network. ETA timestamps messages at transmit and receive time, thus removing the largest amount of nondeterministic message delay from the communication pathway. We evaluated synchronization accuracy in the mote network with the pairwise difference method. Two nodes simultaneously timestamped the arrival of an event beacon, then forwarded the timestamps to a sink node two hops away. At each hop, the timestamps are converted to the local timescale. The synchronization error is the difference between the timestamps at the sink node. For 100 synchronizations, the average error was $5.04\mu s$, with a maximum of $9\mu s$.

PC Network: We used RBS [7] to synchronize the PC network. RBS synchronizes a set of nodes to the arrival of a reference beacon. Participating nodes timestamp the arrival of a message broadcast over the network, and by exchanging these timestamps, neighboring nodes are able to maintain reference tables for timescale transformation. We used a separate RBS transmitter to broadcast a reference beacon every ten seconds over 100 iterations. Synchronization error, determined using the pairwise difference method, was as low as $17.51\mu s$ on average, and $2050.16\mu s$ maximum. The worst-case error is significantly higher than reported in [7] because the OpenBrick-E wireless network interface controllers in our experimental setup are connected via USB, which has a default polling frequency of 1 kHz.

Mote-PC Network: To synchronize a mote with a PC in software, we adopted the underlying methodology of ETA and applied it to serial communication. On the mote, a timestamp is taken upon transfer of a synchronization byte and inserted into the outgoing message. On the PC, a timestamp is taken immediately after the UART issues the interrupt, and the PC regards the difference between these two timestamps as the PC-mote offset. Serial communication bit rate between the mote and PC is 57600 baud, which approximately amounts to a transfer time of 139 microseconds per byte. However, the UART will not issue an interrupt to the CPU until its 16-byte buffer nears capacity or a timeout occurs. Because the synchronization message is six bytes, reception time in this case will consist of the transfer time of the entire message in addition to the timeout time and the time it takes to transfer the date from the UART buffer into main memory by the CPU. This time is compensated for by the receiver, and the clock offset between the two devices is determined as the difference between the PC receive time and the mote transmit time.

GPIO pins on the mote and PC were connected to an oscilloscope, and set high upon timestamping. The resulting output signals were captured and measured. The test was performed over 100 synchronizations, and the resulting error was $7.32\mu s$ on average, and did not exceed $10\mu s$.

HSN: We evaluated synchronization accuracy across the entire network using the pairwise difference method. Two motes timestamped the arrival of an event beacon, and forwarded the timestamp to the network sink, via one mote and two PCs. RBS beacons were broadcast at four-second intervals, and therefore clock skew compensation was unnecessary, because synchronization error due to clock skew would be insignificant compared with offset error. The average error over the 3-hop network was $101.52\mu s$, with a maximum of $1709\mu s$. The majority of this error is due to the polling delay from the USB wireless network controller. However, synchronization accuracy is still sufficient for our application. The implementation used in these experiments was bundled into a time synchronization service for sensor fusion applications.

VI. Multimodal Target Tracking

This section describes the tracking algorithm and the approach for fusing the audio and video measurements based on a sequential Bayesian estimation framework. We use following notation: Superscript t denotes discrete time ($t \in \mathbb{Z}^+$), subscript $k \in \{1, ..., K\}$ denotes the sensor index, where K is the total number of sensors in the network, the target state at time t is denoted as $x^{(t)}$, and the sensor measurement at time t is denoted as $z^{(t)}$.

A. Sequential Bayesian Estimation

We use a sequential Bayesian estimation framework to estimate the target location $x^{(t)}$ at time t similar to the approach presented in [15]. Sequential Bayesian estimation is a framework to estimate the probability distribution of the target state, $p(x^{(t+1)}|z^{(t+1)})$ using a Bayesian filter described by

$$p(x^{(t+1)}|z^{(t+1)}) \propto p(z^{(t+1)}|x^{(t+1)}) \cdot \int p(x^{(t+1)}|x^{(t)}) \cdot p(x^{(t)}|z^{(t)}) dx^{(t)} \quad (3)$$

where $p(x^{(t)}|z^{(t)})$ is the prior distribution from the previous step, $p(z^{(t+1)}|x^{(t+1)})$ is the likelihood given the target state, and $p(x^{(t+1)}|x^{(t)})$ is the prediction for the target location $x^{(t+1)}$ given the current location $x^{(t)}$ according to a target motion model. Since we are tracking moving vehicles it is reasonable to use a directional motion model based on the vehicle velocity. The directional motion model is described by

$$x^{(t+1)} = x^{(t)} + v + \mathcal{U}[-\delta, +\delta] \quad (4)$$

where $x^{(t)}$ is the target location at time t, $x^{(t+1)}$ is the predicted location, v is the target velocity, and $\mathcal{U}[-\delta, +\delta]$ is a uniform random variable.

Since the sensor models (described later in subsection VI-B) are nonlinear, We use a nonparametric representation for the probability distributions which are represented as discrete grids in 2D space similar to [15]. For nonparametric representation, the integration term in equation (3) becomes a convolution operation between the motion kernel and the prior distribution. The resolution of the grid representation is a trade-off between tracking resolution and computational capacity.

Centralized Bayesian Estimation: Since we use mote class audio nodes that are resource-constrained, centralized Bayesian estimation is a reasonable approach due to the limited computational resources. The likelihood function in equation (3) can be calculated either as a product or weighted summation of the individual likelihood functions.

Hybrid Bayesian Estimation: In sensor fusion a big challenge is to account for conflicting sensor measurements. When sensor conflict is very high, sensor fusion algorithms produce false or meaningless fusion results [11]. Reasons for sensor conflict are sensor locality, different sensor modalities, and sensor faults. If a sensor node is far from a target of interest then the measurements from that sensor will not be useful. Different sensor modalities observe different physical phenomenons. For example, audio and video sensors observe sound sources and moving objects respectively. If a sound source is stationary or a moving target is silent, the two modalities will be in conflict. Also, different modalities are affected by different types of background noise. Finally, poor calibration, sudden change in local conditions can also cause conflicting sensor measurements.

Selecting and clustering the sensor nodes in different groups based on locality or modality can mitigate poor performance due to sensor conflict. For example, clustering the nodes close to the target location and fusing only the nodes in the cluster would remove the conflict due to distant nodes.

The sensor network deployment in this paper is small and the sensing region is comparable to the sensing ranges of the audio and video sensors. For this reason, we do not use locality based clustering. However, we want to evaluate the tracking performance of the audio and video sensors. Hence, we developed a hybrid Bayesian estimation framework by clustering sensor nodes based on modalities and compare it with the centralized approach. Figure 5 illustrates the framework. The

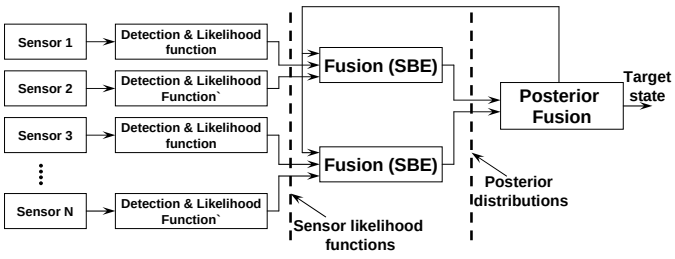

Fig. 5. Hybrid Sequential Estimation

likelihood function from each of the sensor in a cluster is fused together using product or weighted sum. The combined likelihood is then used in equation (3) to calculate the posterior distribution for that cluster. The posteriors from all the clusters are then combined together to estimate the target state.

For hybrid Bayesian estimation with audio-video clustering, the audio posterior is calculated using

$$p_{audio}(x^{(t+1)}|z^{(t+1)}) \propto p_{audio}(z^{(t+1)}|x^{(t+1)}) \cdot \int p(x^{(t+1)}|x^{(t)}) \cdot p(x^{(t)}|z^{(t)}) dx^{(t)}$$

while the video posterior is calculated as

$$p_{video}(x^{(t+1)}|z^{(t+1)}) \propto p_{video}(z^{(t+1)}|x^{(t+1)}) \cdot \int p(x^{(t+1)}|x^{(t)}) \cdot p(x^{(t)}|z^{(t)}) dx^{(t)}$$

The two posteriors are combined either as (product fusion)

$$p(x^{(t+1)}|z^{(t+1)}) \propto p_{audio}(z^{(t+1)}|x^{(t+1)}) \cdot p_{video}(z^{(t+1)}|x^{(t+1)})$$

or (weighted-sum fusion)

$$p(x^{(t+1)}|z^{(t+1)}) \propto \alpha \cdot p_{audio}(z^{(t+1)}|x^{(t+1)}) \cdot (1-\alpha) \cdot p_{video}(z^{(t+1)}|x^{(t+1)})$$

where α is a weighing factor.

B. Sensor Models

We use a nonparametric model for the audio sensors, while a parametric mixture-of-Gaussian model for the video sensors to mitigate the effect of sensor conflict in object detection.

Audio Sensor Model: The nonparametric DOA sensor model for a single audio sensor is the piecewise linear interpolation of the audio detection function, i.e.

$$\lambda(\theta) = w\lambda(\theta_{i-1}) + (1-w)\lambda(\theta_i), \text{ if } \theta \in [\theta_{i-1}, \theta_i]$$

where $w = (\theta_i - \theta)/(\theta_i - \theta_{i-1})$.

Video Sensor Model: The video detection algorithm captures the angle of one or more moving objects. The detection function from equation (2) can be parametrized as a mixture-of-Gaussian

$$\lambda(\theta) = \sum_{i=1}^{n} a_i f_i(\theta)$$

where n is the number of components, $f_i(\theta)$ is the probability density function, and a_i is the mixing proportion for component i. Each component is a Gaussian density function parametrized by μ_i and σ_i^2.

Likelihood Function: Next we present the computation of the likelihood function for a single sensor given the sensor model. The 2D search space is divided into N rectangular regions with center points (x_i, y_i) and side lengths $(2\delta_x, 2\delta_y)$, $i = 1, 2, ..., N$ as illustrated in Figure 6.

The angular interval subtended at the sensor node location Q^k due to region i is $[\varphi_A^{(k,i)}, \varphi_B^{(k,i)}]$. This angular interval is defined as

$$\underbrace{\begin{aligned}\varphi_A^{(k,i)} &= \varphi_0^{(k,i)} + \min_j(\angle P_0^i Q^k P_j^i), j = 1,2,3,4 \\ \varphi_B^{(k,i)} &= \varphi_0^{(k,i)} + \max_j(\angle P_0^i Q^k P_j^i), j = 1,2,3,4\end{aligned}}_{\text{if sensor } k \text{ is } \textbf{not} \text{ in region } i}$$

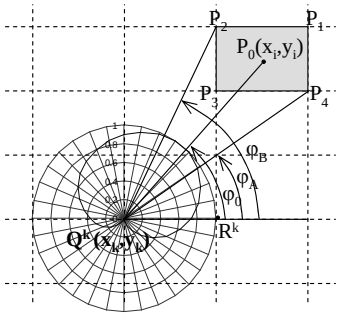

Fig. 6. Computing likelihood function for single sensor k and cell (x_i, y_i)

$$\underbrace{\begin{array}{c}\varphi_A^{(k,i)} = 0 \\ \varphi_B^{(k,i)} = 2\pi\end{array}}_{\text{if sensor } k \text{ is in region } i}$$

where $\varphi_0^{k,i} = \angle R^k Q^k P_0^i$, and the points Q^k and R^k are $Q^k = (x_k, y_k)$ and $R^k = (x_k + 1, y_k)$. The sensor likelihood function value for sensor k at region i is the average detection function value in that region, i.e.

$$p_k(z|x) = p_k(x_i, y_i) = \frac{1}{(\varphi_B^{(k,i)} - \varphi_A^{(k,i)})} \sum_{\varphi_A^{(k,i)} \leq \theta \leq \varphi_B^{(k,i)}} \lambda_k(\theta)$$

VII. Evaluation

The deployment of the multi-modal target tracking system is shown in Figure 7. We deploy 6 audio sensors and 3 video sensors on either side of a road. The objective of the system is to detect and track vehicles using both audio and video under these conditions. Sensor localization and calibration for both audio and video sensors is required. In our experimental setup, we manually placed the sensor nodes at marked locations and orientations. The audio sensors were placed on 1 meter high tripods to minimize audio clutter near the ground.

We gathered audio and video detection data for a total duration of 43 minutes. Table I presents the parameter values that we use in our tracking system. We ran our sensor fusion and tracking system online using centralized sequential Bayesian estimation based on the product of likelihood functions. We also collected all the audio and video detection data for offline evaluation. This way we were able to experiment with different fusion approaches on the same data set. We shortlisted 10 vehicle tracks where there was only a single target in the sensing region. The average duration of tracks was 4.25 sec with 3.0 sec minimum and 5.5 sec maximum. The tracked vehicles were part of an uncontrolled experiment. The vehicles were traveling on road at 20-30 mph speed. The ground truth is estimated post-facto based on the video recording by a separate camera. For evaluation of tracking accuracy, the center of mass of the vehicle is considered to be the true location. Sequential Bayesian estimation requires a prior distribution of the target state. We initialized the prior using a simple detection algorithm based on audio measurements. If the maximum of the combined audio detection functions exceeds a threshold and is within the sensing region, we initialize the prior distribution.

Number of beams in audio beamforming, M_{audio}	36
Number of angles in video detection M_{video}	160
Sensing region (meters)	35×20
Cell size (meters)	0.5×0.5
Interval for uniform random variable in Equation 4 (δ)	$1.2\ v$

TABLE I
PARAMETERS USED IN EXPERIMENTAL SETUP

We experimented with eight different approaches. We used audio-only, video-only and audio-video sensor measurements for sensor fusion. For each of these data sets, the combined likelihood was computed either as the weighted-sum or product of individual sensor likelihood functions. For the audio-video data, we used centralized and hybrid fusion. The list of different approaches is.:

1) audio-only, weighted-sum (AS)
2) video-only, weighted-sum (VS)
3) audio-video, centralized, weighted-sum (AVCS)
4) audio-video, hybrid, weighted-sum (AVHS)
5) audio-only, likelihood product (AP)
6) video-only, likelihood product (VP)
7) audio-video, centralized, likelihood product (AVCP)
8) audio-video, hybrid, likelihood product (AVHP)

Figure 8 shows the tracking error for a representative vehicle track. The tracking error for tracking using audio data is consistently lower than that for the video data. When we use both audio and video data, the tracking error is lower than either of those considered alone. Figure 9 shows the determinant of the covariance of the target state for the same vehicle track. The covariance, which is an indicator of uncertainty in target state is significantly lower for product fusion than weighted-sum fusion. In general, covariance for audio-only is higher than video-only, while using both modalities lowers the uncertainty.

Figure 10 shows average tracking errors and Figure 11 shows the determinant of the covariance for all ten vehicle tracks for all target tracking approaches mentioned above. Audio and video modalities are able track vehicles successfully, though they suffer from poor performance in presence of high background noise and clutter. In general, audio sensors are able to track vehicles with good accuracy, but they suffer from high uncertainty and poor sensing range. Video tracking is not very robust on multiple objects and noise. As expected, fusing the two modalities consistently gives better performance. There are some cases where audio tracking performance is better than fusion. This is due to poor performance of video tracking.

Fusion based on product of likelihood functions gives better performance but it is more vulnerable to sensor conflict and errors in sensor calibration, etc. The weighted-sum approach is more robust to conflicts and sensor errors, but it suffers from high uncertainty. Centralized estimation framework consistently performed better than the hybrid framework.

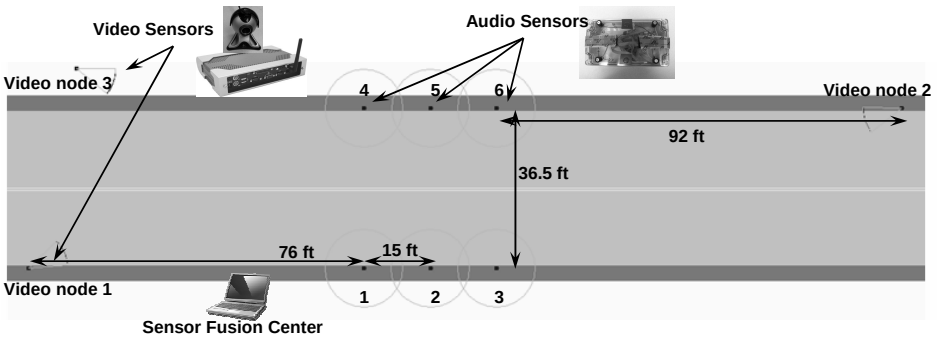

Fig. 7. Experimental setup

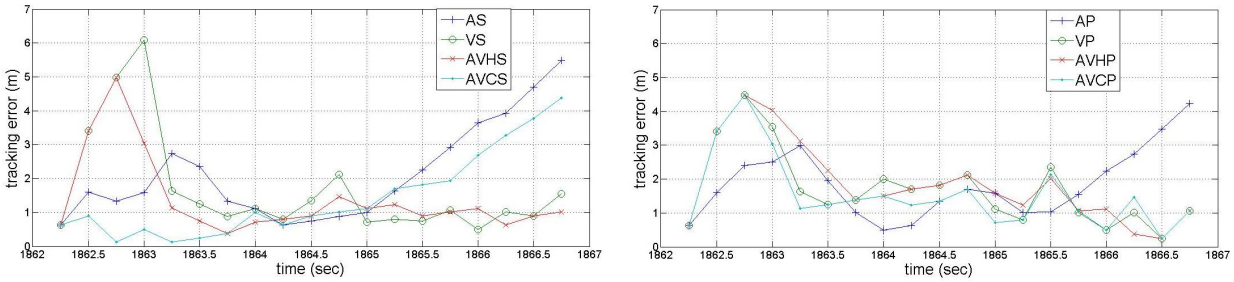

Fig. 8. Tracking error (a) weighted sum, (b) product

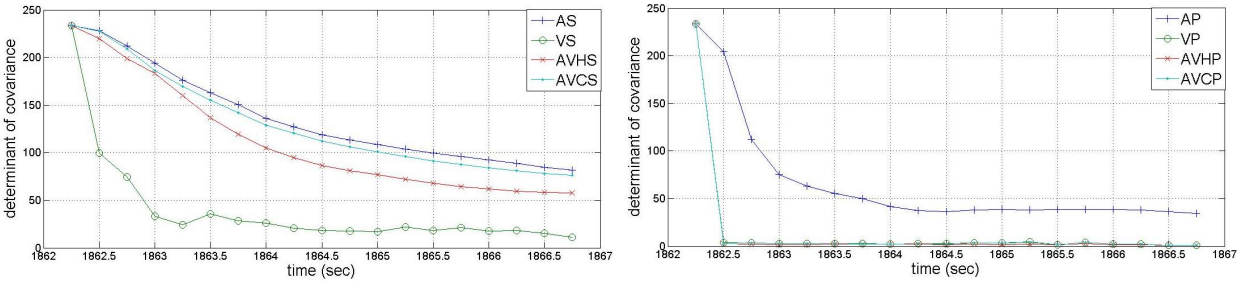

Fig. 9. Tracking variance (a) weighted sum, (b) product

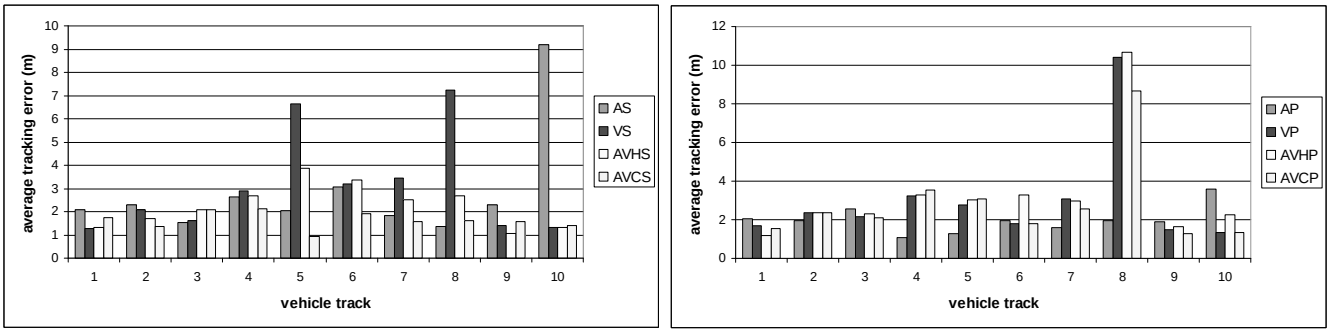

Fig. 10. Tracking errors (a) weighted sum, (b) product

VIII. Related Work

Audio Beamforming: An overview on beamforming and its application for localization in sensor networks can be found in [5]. Beamforming methods have successfully been applied to detect single or even multiple sources in noisy and reverberant environments [4], [1], [14].

Video Tracking: Many adaptive background-modeling methods have been proposed. The work in [8] modeled each pixel in a camera scene by an adaptive parametric mixture model of three Gaussian distributions. An adaptive

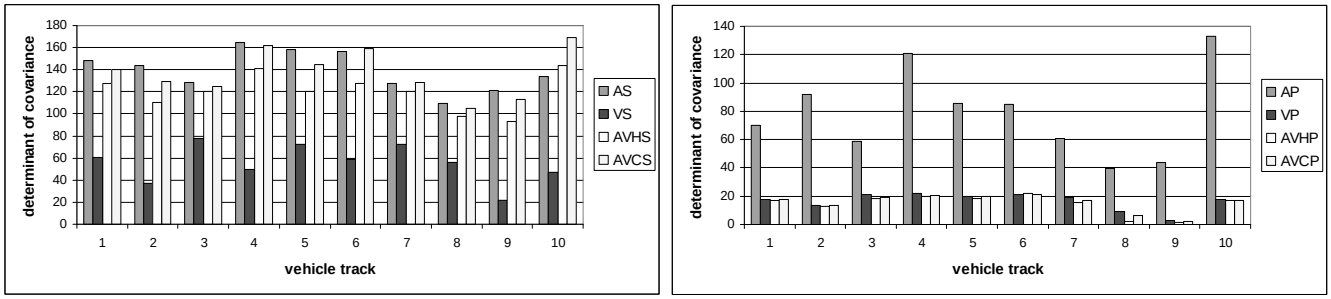

Fig. 11. Determinant of covariance (a) weighted sum, (b) product

nonparametric Gaussian mixture model to address background modeling challenges is presented in [19]. Other techniques using high-level processing to assist the background modeling also have been proposed [21], [12]

Time Synchronization: Time synchronization in sensor networks has been studied extensively in the literature and several protocols have been proposed [7], [9], [16], [17], [13], [18]. Mote-PC synchronization was achieved in [10] by connecting the GPIO ports of a mote and IPAQ PDA.

Multimodal Tracking: Previous work in multimodal target tracking using audio-video data object localization and tracking based on Kalman filtering [20] as well as particle filtering approaches [3], [2].

IX. Conclusions

We have developed a multimodal tracking system using an HSN consisting of six mote audio nodes and 3 PC camera nodes. Our system employs a sequential Bayesian estimation framework which integrates audio beamforming with video object detection. Time synchronization across the HSN allows the fusion of the sensor measurements. We have deployed the HSN and evaluated the performance by tracking moving vehicles in an uncontrolled urban environment. Our evaluation in this paper is limited to single targets and we have shown that, in general, fusion of audio and video measurements improves the tracking performance. Currently, our system is not robust to multiple acoustic sources or multiple moving objects and this is the main direction of our future work. As in all sensor network applications, scalability is an important aspect that has to be addressed, and we plan to expand our HSN using additional mote class devices equipped with cameras.

Acknowledgement: This work is partially supported by ARO MURI W911NF-06-1-0076.

References

[1] S. T. Birchfield. A unifying framework for acoustic localization. In *Proceedings of the 12th European Signal Processing Conference (EUSIPCO)*, Vienna, Austria, September 2004.

[2] V. Cevher, A. Sankaranarayanan, J. H. McClellan, and R. Chellappa. Target tracking using a joint acoustic video system. *IEEE Transactions on Multimedia*, 9(4):715–727, June 2007.

[3] N. Checka, K. Wilson, V. Rangarajan, and T. Darrell. A probabilistic framework for multi-modal multi-person tracking. In *IEEE Workshop on Multi-Object Tracking*, 2003.

[4] J. Chen, L. Yip, J. Elson, H. Wang, D. Maniezzo, R. Hudson, K. Yao, and D. Estrin. Coherent acoustic array processing and localization on wireless sensor networks. In *Proceedings of the IEEE*, volume 91, pages 1154–1162, August 2003.

[5] J. C. Chen, K. Yao, and R. E. Hudson. Acoustic source localization and beamforming: theory and practice. In *EURASIP Journal on Applied Signal Processing*, pages 359–370, April 2003.

[6] M. Ding, A. Terzis, I.-J. Wang, and D. Lucarelli. Multi-modal calibration of surveillance sensor networks. In *Military Communications Conference, MILCOM 2006*, 2006.

[7] J. Elson, L. Girod, and D. Estrin. Fine-grained network time synchronization using reference broadcasts. In *Operating Systems Design and Implementation (OSDI)*, 2002.

[8] N. Friedman and S. Russell. Image segmentation in video sequences: A probabilistic approach. In *Conference on Uncertainty in Artificial Intelligence*, 1997.

[9] S. Ganeriwal, R. Kumar, and M. B. Srivastava. Timing-sync protocol for sensor networks. In *ACM SenSys*, 2003.

[10] L. Girod, V. Bychkovsky, J. Elson, and D. Estrin. Locating tiny sensors in time and space: A case study. In *ICCD*, 2002.

[11] H. Y. Hau and R. L. Kashyap. On the robustness of Dempster's rule of combination. In *IEEE International Workshop on Tools for Artificial Intelligence*, 1989.

[12] P. KaewTraKulPong and R. B. Jeremy. An improved adaptive background mixture model for realtime tracking with shadow detection. In *Workshop on Advanced Video Based Surveillance Systems (AVBS)*, 2001.

[13] B. Kusy, P. Dutta, P. Levis, M. Maroti, A. Ledeczi, and D. Culler. Elapsed time on arrival: A simple and versatile primitive for time synchronization services. *International Journal of Ad hoc and Ubiquitous Computing*, 2(1), January 2006.

[14] A. Ledeczi, A. Nadas, P. Volgyesi, G. Balogh, B. Kusy, J. Sallai, G. Pap, S. Dora, K. Molnar, M. Maroti, and G. Simon. Countersniper system for urban warfare. *ACM Trans. Sensor Networks*, 1(2), 2005.

[15] J. Liu, J. Reich, and F. Zhao. Collaborative in-network processing for target tracking. In *EURASIP, Journal on Applied Signal Processing*, 2002.

[16] M. Maroti, B. Kusy, G. Simon, and A. Ledeczi. The flooding time synchronization protocol. In *ACM SenSys*, 2004.

[17] K. Romer. Time synchronization in ad hoc networks. In *ACM Symposium on Mobile Ad-Hoc Networking and Computing*, 2001.

[18] J. Sallai, B. Kusy, A. Ledeczi, and P. Dutta. On the scalability of routing integrated time synchronization. In *Workshop on Wireless Sensor Networks (EWSN)*, 2006.

[19] C. Stauffer and W. Grimson. Learning patterns of activity using real-time tracking. In *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2000.

[20] N. Strobel, S. Spors, and R. Rabenstein. Joint audio video object localization and tracking. In *IEEE Signal Processing Magazine*, 2001.

[21] K. Toyama, J. Krumm, B. Brumitt, and B. Meyers. Wallflower: Principles and practice of background maintenance. In *IEEE International Conference on Computer Vision*, 1999.

[22] P. Volgyesi, G. Balogh, A. Nadas, C. Nash, and A. Ledeczi. Shooter localization and weapon classification with soldier-wearable networked sensors. In *Mobisys*, 2007.

[23] M. Yarvis, N. Kushalnagar, H. Singh, A. Rangarajan, Y. Liu, and S. Singh. Exploiting heterogeneity in sensor networks. In *IEEE INFOCOM*, 2005.