Tracking in Urban Environments Using Sensor Networks Based on Audio-Video Fusion

Manish Kushwaha, Songhwai Oh, Isaac Amundson, Xenofon Koutsoukos, Akos Ledeczi

1 Introduction

Heterogeneous sensor networks (HSNs) are gaining popularity in diverse fields, such as military surveillance, equipment monitoring, and target tracking [41]. They are natural steps in the evolution of wireless sensor networks (WSNs) driven by several factors. Increasingly, WSNs will need to support multiple, although not necessarily concurrent, applications. Different applications may require different resources. Some applications can make use of nodes with different capabilities. As the technology matures, new types of nodes will become available and existing deployments will be refreshed. Diverse nodes will need to coexist and support old and new applications.

Furthermore, as WSNs are deployed for applications that observe more complex phenomena, multiple sensing modalities become necessary. Different sensors usually have different resource requirements in terms of processing power, memory capacity, or communication bandwidth. Instead of using a network of homogeneous devices supporting resource intensive sensors, an HSN can have different nodes for different sensing tasks [22, 34, 10]. For example, at one end of the spectrum low data-rate sensors measuring slowly changing physical parameters such as temperature, humidity or light, require minimal resources; while on the other end even low resolution video sensors require orders of magnitude more resources.

Manish Kushwaha
Institute for Software Integrated Systems, Vanderbilt University, Nashville, TN, USA, e-mail: manish.kushwaha@vanderbilt.edu

Songhwai Oh
Electrical Engineering and Computer Science, University of California at Merced, Merced, CA, USA e-mail: songhwai.oh@ucmerced.edu

Isaac Amundson
Institute for Software Integrated Systems, Vanderbilt University, Nashville, TN, USA, e-mail: isaac.amundson@vanderbilt.edu

Tracking is one such application that can benefit from multiple sensing modalities [9]. If the moving target emits sound signal 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 in the presence of visual clutter that handicap the video. Additionally, tracking based on the fusion of audio-video data can improve the performance of audio-only or video-only approaches. Audio-video tracking can also provide cues for the other modality for actuation. For example, visual steering information from a video sensor may be used to steer the audio sensor (microphone array) toward a moving target. Similarly, information from the audio sensors can be used to steer a pan-tilt-zoom camera toward a speaker. Although, audio-visual tracking has been applied to smart videoconferencing applications [42], it does not use a wide-area distributed platform .

The main challenges in target tracking is to find tracks from noisy observation. This requires solutions to both data association and state estimation problems. Our system employs a Markov Chain Monte Carlo Data Association (MCMCDA) algorithm for tracking. The MCMCDA algorithm is a data-oriented, combinatorial optimization approach to the data association problem that avoids the enumeration of tracks. The MCMCDA algorithm enables us to track an unknown number of targets in noisy urban environment

In this chapter, we describe our ongoing research in multimodal target tracking in urban environments utilizing an HSN of mote class devices equipped with microphone arrays as audio sensors and embedded PCs equipped with web cameras as video sensors. The targets to be tracked are moving vehicles emitting engine noise. Our system has many components including audio processing, video processing, WSN middleware services, multimodal sensor fusion, and target tracking based on sequential Bayesian estimation and MCMCDA. While none of these is necessarily novel, their composition and implementation on an actual HSN requires addressing a number of significant challenges.

Specifically, we have implemented audio beamforming on audio sensors utilizing an FPGA-based sensor board and evaluated its performance as well its energy consumption. While we are using a standard motion detection algorithm on video sensors, we have implemented post-processing filters that represent the video data in a similar format as the audio data, which enables seamless audio-video data fusion. Furthermore, we have extended time synchronization techniques for HSNs consisting of mote and PC networks. Finally, the main challenge we address is system integration including making the system work on an actual platform in a realistic deployment. The paper provides results gathered in an uncontrolled urban environment and presents a thorough evaluation including a comparison of different fusion and tracking approaches.

The rest of the paper is organized as follows. Section 2 presents related work. In Section 3, we describe the overall system architecture. It is followed by the description of the audio processing in Section 4 and then the video processing in Section 5. In Section 6, we present our time synchronization approach for HSNs. Multimodal target tracking is described in Section 7. Section 8 presents the experimental deployment and its evaluation. Finally, we conclude in Section 9.

2 Challenges and Related Work

In this section, we present the challenges for multimodal sensor fusion and tracking. We also present various existing approach for sensor fusion and tracking. In multimodal multisensor fusion, the data may be fused at a variety of levels including the raw data level, where the raw signals are fused, a feature-level, where representative characteristics of the data are extracted and fused, and a decision-level fusion wherein target estimates from each sensor are fused [17]. At successive levels more information may be lost, but in collaborative applications, such as WSN applications, the communication requirement of transmitting large amounts of data is reduced.

Another significant challenge is sensor conflict, when different sensors report conflicting data. When sensor conflict is very high, fusion algorithms produce false or meaningless results [18]. Reasons for sensor conflict may be sensor locality, different sensor modalities, or sensor faults. If a sensor node is far from a target of interest then the data from that sensor will not be useful, and will have higher variance. Different sensor modalities observe different physical phenomena. 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 might produce conflicting data. Also, different modalities are affected by different types of background noise. Finally, poor calibration and sudden change in local conditions can also cause conflicting sensor data.

Another classical problem in multitarget tracking is to find a track of each target from the noisy data. If the association of sequence of data-points with each target is known, multitarget tracking reduces to a set of state estimation problems. The *data association* problem is to find which data-points are generated by which targets, or in other words, associate each data-point with either a target or noise.

Rest of the section presents existing work in acoustic localization, video tracking, time synchronization and multimodal fusion.

Acoustic Localization

An overview of the theoretical aspects of Time Difference of Arrival (TDOA) based acoustic source localization and beamforming is presented in [7]. Beamforming methods have successfully been applied to detect single or even multiple sources in noisy and reverberant environments. A maximum likelihood (ML) method for single target and an approximate maximum likelihood (AML) method for direction of arrival (DOA) based localization in reverberant environments are proposed in [7, 4]. In [1], an empirical study of collaborative acoustic source localization based on an implementation of the AML algorithm is shown. An alternative technique called accumulated correlation combines the speed of time-delay estimation with the accuracy of beamforming [5]. Time of Arrival (TOA) and TDOA based methods are proposed for collaborative source localization for wireless sensor network applications [24]. A particle filtering based approach for acoustic source localization and tracking using beamforming is described in [39].

Video Tracking

A simple approach to motion detection from video data is via frame differencing, which requires a robust background model. There exist a number of challenges for the estimation of robust background models [38], including gradual and sudden illumination changes, vacillating backgrounds, shadows, visual clutter, occlusion, etc. In practice, most of the simple motion detection algorithms have poor performance when faced with these challenges. Many adaptive background-modeling methods have been proposed to deal with these challenges. The work in [13] models each pixel in a camera frame by an adaptive parametric mixture model of three Gaussian distributions. A Kalman filter to track the changes in background illumination for every pixel is used in [21]. An adaptive nonparametric Gaussian mixture model to address background modeling challenges is done in [36]. A kernel estimator for each pixel is proposed in [11] with kernel exemplars from a moving window. Other techniques using high-level processing to assist the background modeling have been proposed; for instance, the Wallflower tracker [38] which circumvents some of these problems using high-level processing rather than tackling the inadequacies of the background model. The algorithm in [19] proposes an adaptive background modeling method based on the framework in [36]. The main differences lie in the update equations, initialization method and the introduction of a shadow detection algorithm.

Time Synchronization

Time synchronization in sensor networks has been studied extensively in the literature and several protocols have been proposed. Reference Broadcast Synchronization (RBS) [12] is a protocol for synchronizing a set of receivers to the arrival of a reference beacon. The Timing-sync Protocol for Sensor Networks (TPSN) [**?**] is a sender-receiver protocol that uses the round-trip latency of a message to estimate the time a message takes along the communication pathway from sender to receiver. This time is then added to the sender timestamp and transmitted to the receiver to determine the clock offset between the two nodes. Elapsed Time on Arrival (ETA) [23] provides a set of application programming interfaces for an abstract time synchronization service. RITS [35] is an extension of ETA over multiple hops. It incorporates a set of routing services to efficiently pass sensor data to a network sink node for data fusion. In [15], mote-PC synchronization was achieved by connecting the GPIO ports of a mote and IPAQ PDA using the mote-NIC model. Although using this technique can achieve microsecond-accurate synchronization, it was implemented as a hardware modification rather than in software.

Multimodal Tracking

A thorough introduction to multisensor data fusion, with focus on data fusion applications, process models, and architectures is provided in [17]. The paper surveys a number of related techniques, and reviews standardized terminology, mathematical techniques, and design principles for multisensor data fusion systems. A modular tracking architecture that combines several simple tracking algorithms is presented in [31]. Several simple and rapid algorithms run on different CPUs, and the tracking results from different modules are integrated using a Kalman filter. In [37], two different fusion architectures based on decentralized recursive (extended) Kalman filters are described for fusing outputs from independent audio and video trackers.

Multimodal sensor fusion and tracking have been applied to smart videoconferencing and indoor tracking applications [44, 42, 3, 6]. In [44], data is obtained using multiple cameras and microphone arrays. The video data consist of pairs of image coordinates of features on each tracked object, and the audio data consist of TDOA for each microphone pair in the array. A particle filter is implemented for tracking. In [3], a graphical model based approach is taken using audiovisual data. The graphical model is designed for single target tracking using a single camera and a microphone pair. The audio data consist of the signals received at the microphones, and the video data consist of monochromatic video frames. Generative models are described for the audio-video data that explain the data in terms of the target state. An expectation-maximization (EM) algorithm is described for parameter learning and tracking, which also enables automatic calibration by learning the audio-video calibration parameters during each EM step. In [6], a probabilistic framework is described for multitarget tracking using audio and video data. The video data consist of plan-view images of the foreground. Plan-view images are projections of the images captured from different points-of-view to a 2D plane, usually the ground plane where the targets are moving [8]. Acoustic beamforms for a fixed length signal are taken as the audio data. A particle filter is implemented for tracking.

3 Architecture

Figure 1 shows the system architecture. The audio sensors, consisting of MICAZ motes with acoustic sensor boards equipped with a microphone array, form an IEEE 802.15.4 network. This network does not need to be connected; it can consist of multiple connected components as long as each of these component have a dedicated mote-PC gateway. The video sensors are based on Logitech QuickCam Pro 4000 cameras attached to OpenBrick-E Linux embedded PCs. These video sensors, along with the mote-PC gateways, the sensor fusion node and the reference broadcaster for time synchronization (see Section 6) are all PCs forming a peer-to-peer 802.11b wireless network. Figure 2 shows the conceptual layout of the sensor network.

The audio sensors perform beamforming, and transmit the audio detections to the corresponding mote-PC gateway utilizing a multi-hop message routing service [2].

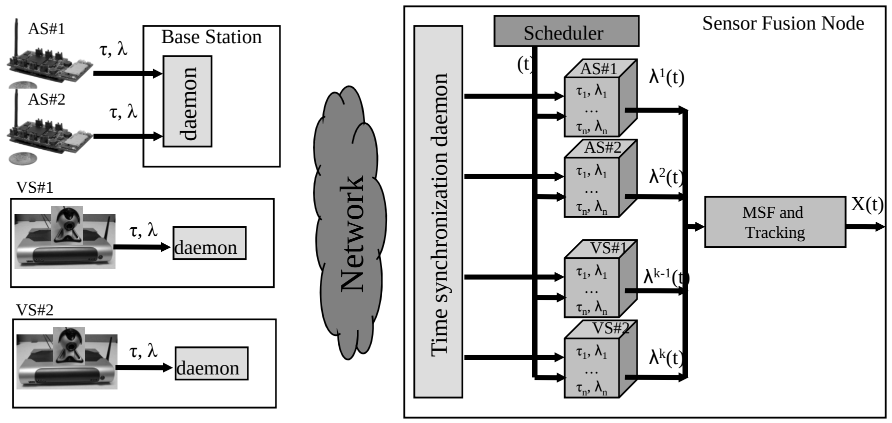

Fig. 1 Architecture for the multimodal target tracking system. Here, τ denotes measurement timestamps, λ denotes sensor measurements (also called detection functions that are described in later sections), and t denotes time. The blocks shown inside the sensor fusion node are circular buffers that store timestamped measurements.

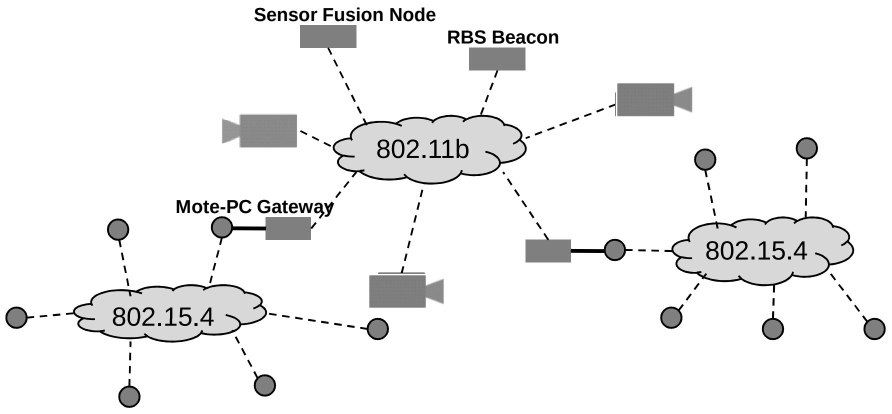

Fig. 2 Conceptual layout of sensor network for multimodal target tracking. Circles represent the audio sensors, camera silhouettes represent the video sensors and rectangles represent PCs.

The routing service also performs time translation of the detection timestamps. The video sensors run a motion detection algorithm and compute timestamped video detection functions. Both audio and video detections are routed to the central sensor fusion node. Time translation is also carried out in the 802.11b network utilizing an RBS-based time synchronization service. This approach ensures that sensor data arrives at the sensor fusion node with timestamps in a common timescale. On the sensor fusion node, the sensor measurements are stored in appropriate sensor buffers, one for each sensor. A sensor fusion scheduler triggers periodically and generates a fusion timestamp. The trigger is used to retrieve the sensor measurements from the sensor buffers with timestamps closest to the generated fusion timestamp. The retrieved sensor measurements are then used for multimodal fusion and target tracking. In this study, we developed and compared target tracking algorithms based on both Sequential Bayesian Estimation (SBE) and MCMCDA. Note that the triggering mechanism of the scheduler decouples the target tracking rate from the audio and video sensing rates, which allows us to control the rate of the tracking application independently of the sensing rate.

4 Audio Beamforming

Beamforming is a signal processing algorithm for DOA estimation of a signal source. In a typical beamforming array, each of the spatially separated microphones receive a time-delayed source signal. The amount of time delay at each microphone in the array depends on the microphone arrangement and the location of the source. A typical delay-and-sum single source beamformer discretizes the sensing region into directions, or *beams*, and computes a beam energy for each of them. The beam energies 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 Fig. 3. The amplified microphone signal is sampled at a high frequency (100 KHz) to provide high resolution for the time delay, which is required for the closely placed microphones. The raw signals are filtered to remove unwanted noise components. 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 microphones so that the resulting beams are steered to the beam angles, $\theta_i = i\frac{360}{M}$ degrees, for $i = 0, 1, ...M-1$. The signal is downsam-

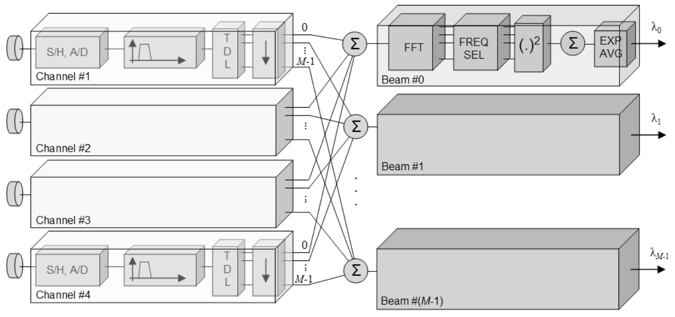

Fig. 3 Data-flow diagram of the real-time beamforming sensor.

pled and the M beams 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. The block power values, $\mu(\theta_i)$, are smoothed by exponential averaging into the beam energy, $\lambda(\theta_i)$

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

where α is an averaging factor.

Audio Hardware

In our application, the audio sensor is a MICAz mote with an onboard Xilinx XC3S1000 FPGA chip that is used to implement the beamformer [40]. 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 is placed on the sensor node, as shown in Fig. 4(a). 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 are 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.

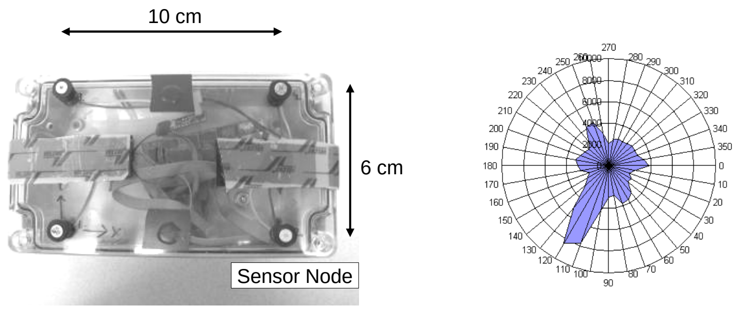

Fig. 4 (a) Sensor Node Showing the Microphones, (b) Beamform of acoustic source at a distance of 50 feet and an angle of 120 degrees.

Evaluation

We test the beamformer node outdoors using recorded speech as the acoustic source. Measurements are taken by placing the source at distances of 3, 5, 10, 25, 50, 75, 100, and 150 feet from the sensor, and at an angle from -180° to +180° in 5° increments. Figure 4(b) shows the beamforming result for a single audio sensor when the source was at a distance of 50 feet. Mean DOA measurement error for 1800 human speech experiments is shown in Figure 5. The smallest error was recorded when the source was at a distance of 25 feet from the sensor. The error increases as the source moves closer to the sensor. This is because the beamforming algorithm assumes a planar wavefront, which holds for far-field sources but breaks down when the source is close to the sensor. However, as the distance between the source and sensor grows, error begins accumulating again as the signal-to-noise ratio decreases.

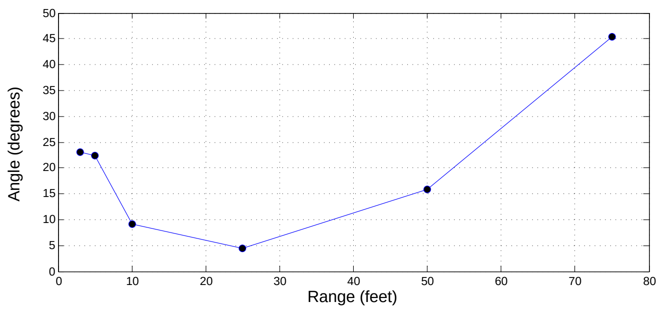

Fig. 5 Human speech DOA error for distances of 3, 5, 10, 25, 50, 75 feet between acoustic source and beamformer node.

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 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 inherent 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. Flash-based FPGAs and smart duty cycling techniques are promising new directions in our research project for reducing the power requirements.

5 Video Tracking

Video tracking systems seek to automatically detect moving targets and track their movement 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 targets (in the context of target tracking), robust motion detection is the 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.

Algorithm

The dataflow in Figure 6 shows the motion detection algorithm and its components used in our tracking application. The first component is background-foreground seg-

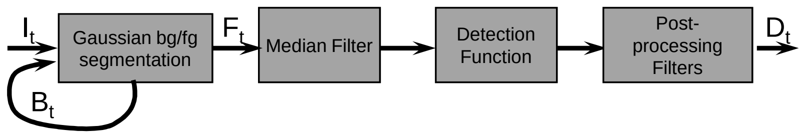

Fig. 6 Data-flow diagram of real-time motion detection algorithm

mentation of the currently captured frame (I_t) from the camera. We use the algorithm described in [19]. 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 use a median filter of size 3×3.

Since our sensor fusion algorithm (Section 7) utilizes only the angle of moving targets, it is desirable and sufficient to represent the foreground in a simpler detection function. Similar to the beam angle concept in audio beamforming (Section 4), the field-of-view of the camera is divided into M equally-spaced angles

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

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

$$\lambda(\theta_i) = \sum_{j \in \theta_i}^{W} \sum_{k=1}^{H} F(j,k) : i = 1,2,...,M \tag{3}$$

where F is the binary foreground image, H and W are the vertical and horizontal resolutions in pixels, respectively and $j \in \theta_i$ indicates columns in the frame that fall within angle θ_i. Figure 7 shows a snapshot of motion detection.

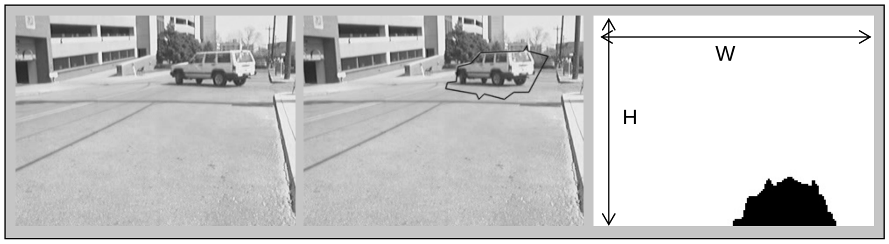

Fig. 7 Video detection. The frame on the left shows the input image, the frame in the middle shows the foreground and the frame on the right shows the video detection function.

Video Post-processing

In our experiments, we gathered video data of vehicles from multiple 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 is 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. For this purpose we keep a background of moving averages which was removed from each detection. The background update and filter equations are

$$b^t(\theta) = \alpha b^{t-1}(\theta) + (1-\alpha)\lambda^t(\theta) \tag{4}$$
$$\lambda^t(\theta) = \lambda^t(\theta) - b^{t-1}(\theta) \tag{5}$$

where $b^t(\theta)$ is the moving average background, and $\lambda^t(\theta)$ is the detection function at time t. 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. This essentially reduces the effect of any sharp spikes in detection function due to glints. The equation for this filter is

$$\lambda^t(\theta) = \lambda^t(\theta) * k \tag{6}$$

where k is a 7×1 vector of equal weights, and $*$ denotes convolution.

We implemented the motion detection algorithm using OpenCV (open source computer vision) library. We use Linux PCs equipped with the QuickCam Pro 4000 as video sensors. The OpenBrick-E has 533 MHz CPU, 128 MB RAM, and a 802.11b wireless adapter. The QuickCam Pro supports up to 640×480 pixel resolution and up to 30 frames-per-second frame rate. Our motion detection algorithm implementation runs at 4 frames-per-second and 320×240 pixel resolution. The number of angles in Equation (2) is $M = 160$.

6 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. [12, 14, 27, 35]), achieving accurate synchronization in a *heterogeneous* sensor network is not a trivial task.

6.1 Synchronization Methodology

Attempting to synchronize the entire network using a single protocol will introduce a large amount of error. For example, TPSN [14], FTSP [27], and RITS [35] were all designed to run on the Berkeley Motes, and assume the operating system is tightly integrated with the radio stack. Attempting to use such protocols on an 802.11 PC network will result in poor synchronization accuracy because the necessary low-level hardware control is difficult to achieve. Reference Broadcast Synchronization (RBS) [12], although flexible when it comes to operating system and computing platform, is accurate only when all nodes have access to a common network medium. A combined network of motes and PCs will be difficult to synchronize using RBS because each platform uses different communication protocols and wireless frequencies. Instead, we adopted a hybrid approach [2], which pairs a specific network with the synchronization protocol that provides the most accuracy with the least amount of overhead. To synchronize the entire network, it is necessary for gateway nodes (i.e., nodes connecting multiple networks,) to handle several synchronization mechanisms.

Mote Network

We used Elapsed Time on Arrival (ETA) [23] to synchronize the mote network. ETA timestamps synchronization messages at transmit and receive time, thus removing the largest amount of nondeterministic message delay from the communication pathway. On the sender side, a timestamp is taken upon transmission of a designated synchronization byte, and placed at the end of the outgoing message. On the receiver side, a timestamp is taken upon arrival of the synchronization byte. The difference between these two timestamps is generally deterministic, and is based primarily on the bitrate of the transceivers and the size of the message. We selected ETA for the mote network due to its accuracy and low message overhead.

PC Network

We used RBS 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. RBS minimizes synchronization error by taking advantage of the broadcast channel of wireless networks.

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.

6.2 Evaluation of HSN Time Synchronization

Our HSN contains three communication pathways: mote-mote, mote-PC, and PC-PC. We first examine the synchronization accuracy of each of these paths, then present the synchronization results for the entire network.

Mote Network

We evaluated synchronization accuracy in the mote network with the pairwise difference method. Two nodes simultaneously timestamp the arrival of an event beacon, then forward 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 is $5.04\mu s$, with a maximum of $9\mu s$.

PC Network

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 [12] 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. For our tracking application, this is acceptable because we use a sampling rate of 4 Hz. Note that the reason for selecting such a low sampling rate is due to bandwidth constraints and interference and not because of synchronization error.

Mote-PC Connection

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 $10\mu s$. The majority of the error is due to jitter, both in the UART and the CPU. A technique to compensate for such jitter on the motes is presented in [27], however, we did not attempt it on the PCs.

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 audio and video sensor fusion at 4Hz.

6.3 Synchronization Service

The implementation used in these experiments was bundled into a time synchronization and routing service for sensor fusion applications. Figure 8 illustrates the interaction of each component within the service. The service interacts with sensing applications that run on the local PC, as well as other service instances running on remote PCs. The service accepts timestamped event messages on a specific port, converts the embedded timestamp to the local timescale, and forwards the message toward the sensor-fusion node. The service uses reference broadcasts to maintain synchronization with the rest of the PC network. In addition, the service accepts mote-based event messages, and converts the embedded timestamps using the ETA-based serial timestamp synchronization method outlined above. Kernel modifications in the serial and wireless drivers were required in order to take accurate timestamps. Upon receipt of a designated synchronization byte, the time is recorded and passed up to the synchronization service in the user space. The modifications are unobtrusive to other applications using the drivers, and the modules are easy to load into the kernel. The mote implementation uses the TimeStamping interface, provided with the TinyOS distribution [25]. A modification was made to the UART interface to insert a transmission timestamp into the event message as it is being transmitted between the mote and PC. The timestamp is taken immediately before a synchronization byte is transmitted, then inserted at the end of the message.

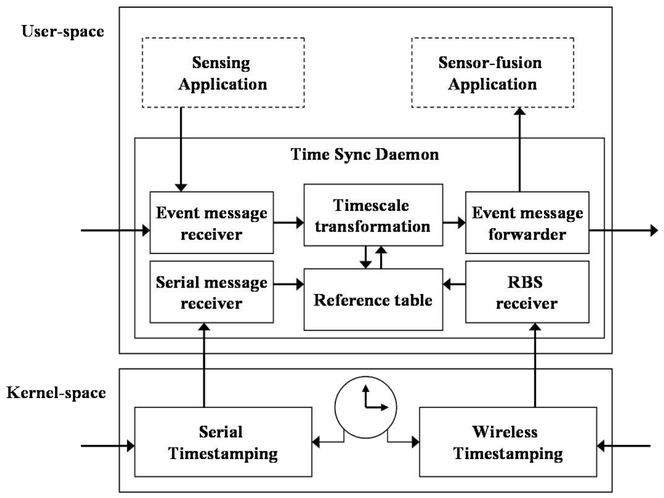

Fig. 8 PC-based time synchronization service.

7 Multimodal Target Tracking

This section describes the tracking algorithm and the approach for fusing the audio and video data based on sequential Bayesian estimation. 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 data at time t is denoted as $z^{(t)}$.

7.1 Sequential Bayesian Estimation

We use sequential Bayesian estimation to estimate the target state $x^{(t)}$ at time t, similar to the approach presented in [26]. In sequential Bayesian estimation, the target state is estimated by computing the posterior probability density $p(x^{(t+1)}|z^{(t+1)})$ using a Bayesian filter described by

$$p(x^{(t+1)}|z^{(t+1)}) \propto p(z^{(t+1)}|x^{(t+1)}) \int p(x^{(t+1)}|x^{(t)}) p(x^{(t)}|z^{(t)}) dx^{(t)} \quad (7)$$

where $p(x^{(t)}|z^{(t)})$ is the prior density 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 state $x^{(t+1)}$ given the current state $x^{(t)}$ according to a target motion model. Since we are tracking moving vehicles it is reasonable to use a directional motion model based on the target velocity. The directional motion model is described by

$$x^{(t+1)} = x^{(t)} + v + \mathscr{U}[-\delta, +\delta] \quad (8)$$

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

Because the sensor models (described later in Subsection 7.2) are nonlinear, the probability densities cannot be represented in a closed form. It is, therefore, reasonable to use a nonparametric representation for the probability densities. The nonparametric densities are represented as discrete grids in 2D space, similar to [26]. For nonparametric representation, the integration term in Equation (7) 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 resource constrained mote class sensors, centralized Bayesian estimation is a reasonable approach because of its modest computational requirements. The likelihood function in Equation (7) can be calculated either as a product or weighted summation of the individual likelihood functions. We describe the two methods next.

Product of Likelihood functions

Let $p_k(z^{(t)}|x^{(t)})$ denote the likelihood function from sensor k. If the sensor observations are mutually independent conditioned on the target state, the likelihood functions from multiple sensors are combined as

$$p(z^{(t)}|x^{(t)}) = \prod_{k=1,\dots,K} p_k(z^{(t)}|x^{(t)}) \tag{9}$$

Weighted-Sum of Likelihood functions

An alternative approach to combine the likelihood functions is to compute their weighted-sum. This approach allows us to give different weights to different sensor data. These weights can be used to incorporate sensor reliability and quality of sensor data. We define a quality index $q_k^{(t)}$ for each sensor k as

$$q_k^{(t)} = r_k \max_{\theta} (\lambda_k^{(t)}(\theta))$$

where r_k is a measure of sensor reliability and $\lambda_k^{(t)}(\theta)$ is the sensor detection function. The combined likelihood function is given by

$$p(z^{(t)}|x^{(t)}) = \frac{\sum_{k=1,\dots,K} q_k^{(t)} p_k(z^{(t)}|x^{(t)})}{\sum_{k=1,\dots,K} q_k^{(t)}} \tag{10}$$

We experimented with both methods in our evaluation. The product method produces more accurate results with low uncertainty in target state. The weighted-sum method performs better in cases with high sensor conflict, though it suffers from high uncertainty. The results are presented in Section 8.

Hybrid Bayesian Estimation

In sensor fusion a big challenge is to account for conflicting sensor data. When sensor conflict is very high, sensor fusion algorithms produce false or meaningless fusion results [18]. Reasons for sensor conflict are sensor locality, different sensor modalities, and sensor faults. Selecting and clustering the sensors in different groups based on locality or modality can mitigate poor performance due to sensor conflict. For example, clustering the sensors close to the target and fusing the data from only the sensors in the cluster would remove the conflict caused by distant sensors.

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 have multimodal sensors that can report conflicting data. Hence, we developed a hybrid Bayesian estimation framework [17] by clustering sensors based on modalities, and compare it with the centralized approach. Figure 9 illustrates the framework. The likelihood function

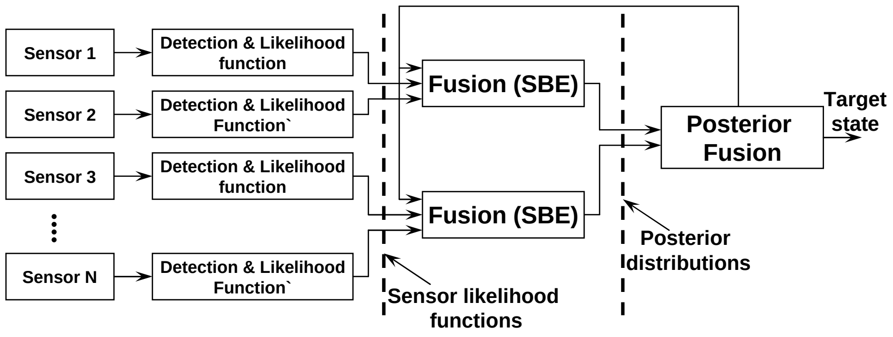
Sensor 1
Sensor 2
Sensor 3
Sensor N
Detection & Likelihood function
Detection & Likelihood Function`
Detection & Likelihood function
Detection & Likelihood Function`
Fusion (SBE)
Fusion (SBE)
Posterior Fusion
Target state
Posterior distributions
Sensor likelihood functions

Fig. 9 Hybrid Bayesian estimation framework.

from each sensor in a cluster is fused together using Equation (9) or (10). The combined likelihood is then used in Equation (7) to calculate the posterior density for that cluster. The posterior densities from all the clusters are then combined together to estimate the target state.

For hybrid Bayesian estimation with audio-video clustering, we compute the likelihood functions using Equation (9) or (10). The audio posterior density is calculated using

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

while the video posterior density is calculated as

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

The two posterior densities are combined either as (product fusion)

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

or (weighted-sum fusion)

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

where α is a weighing factor.

7.2 Sensor Models

We use a nonparametric model for the audio sensors, while a parametric mixture-of-Gaussian model for the video sensors is used 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

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

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 (3) 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, and a_i is the mixing proportion for component i. Each component is a Gaussian density given by $f_i(\theta) = \mathcal{N}(\theta|\mu_i, \sigma_i^2)$, where the component parameters μ_i, σ_i^2 and a_i are calculated from the detection function.

Likelihood Function

The 2D search space is divided into N rectangular cells with center points at (x_i, y_i), for $i = 1, 2, ..., N$ as illustrated in Figure 10. The likelihood function value for k^{th} sensor at i^{th} cell is the average value of the detection function in that cell, given by

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

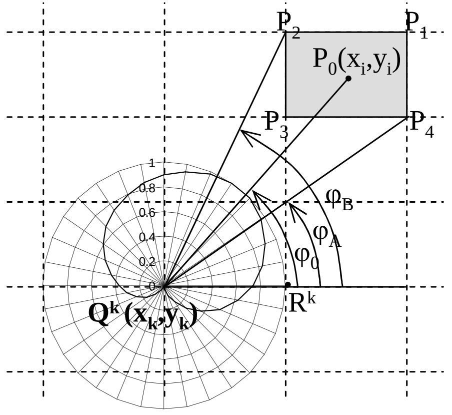

Fig. 10 Computation of the likelihood value for k^{th} sensor at i^{th} cell, (x_i, y_i). The cell is centered at P_0 with vertices at P_1, P_2, P_3, and P_4. The angular interval subtended at the sensor due to the i^{th} cell is $\theta \in [\varphi_A^{(k,i)}, \varphi_B^{(k,i)}]$, or $\theta \in [0, 2\pi]$ if the sensor is inside the cell.

7.3 Multiple-Target Tracking

The essence of the multi-target tracking problem is to find a track of each object from noisy measurements. If the sequence of measurements associated with each object is known, multi-target tracking reduces to a set of state estimation problems for which many efficient algorithms are available. Unfortunately, the association between measurements and objects is unknown. The *data association* problem is to work out which measurements were generated by which objects; more precisely, we require a partition of measurements such that each element of a partition is a collection of measurements generated by a single object or noise. Due to this data association problem, the complexity of the posterior distribution of the states of objects grows exponentially as time progresses. It is well-known that the data association problem is NP-hard [32], so we do not expect to find efficient, exact algorithms for solving this problem.

In order to handle highly nonlinear and non-Gaussian dynamics and observations, a number of methods based on particle filters has recently been developed to track multiple objects in video [30, 20]. Although particle filters are highly effective in single-target tracking, it is reported that they provide poor performance in multi-target tracking [20]. This is because a fixed number of particles is insufficient to represent the posterior distribution with the exponentially increasing complexity (due to the data association problem). As shown in [20, 43], an efficient alternative is to use Markov chain Monte Carlo (MCMC) to handle the data association problem in multi-target tracking.

For our problem, there is an additional complexity. We do not assume the number of objects is known. A *single-scan* approach, which updates the posterior based only on the current scan of measurements, can be used to track an unknown number of targets with the help of trans-dimensional MCMC [43, 20] or a detection algorithm [30]. But a single-scan approach cannot maintain tracks over long periods because it cannot revisit previous, possibly incorrect, association decisions in the light of new evidence. This issue can be addressed by using a *multi-scan* approach, which updates the posterior based on both current and past scans of measurements. The well-known *multiple hypothesis tracking* (MHT) [33] is a multi-scan tracker, however, it is not widely used due to its high computational complexity.

A newly developed algorithm, called Markov chain Monte Carlo data association (MCMCDA), provides a computationally desirable alternative to MHT [29]. The simulation study in [29] showed that MCMCDA was computationally efficient compared to MHT with heuristics (*i.e.*, pruning, gating, clustering, N-scan-back logic and k-best hypotheses). In this chapter, we use the online version of MCMCDA to track multiple objects in a 2-D plane. Due to the page limitation, we omit the description of the algorithm in this paper and refer interested readers to [29].

8 Evaluation

In this section, we evaluate target tracking algorithms based on the sequential Bayesian estimation and MCMCDA.

8.1 Sequential Bayesian Estimation

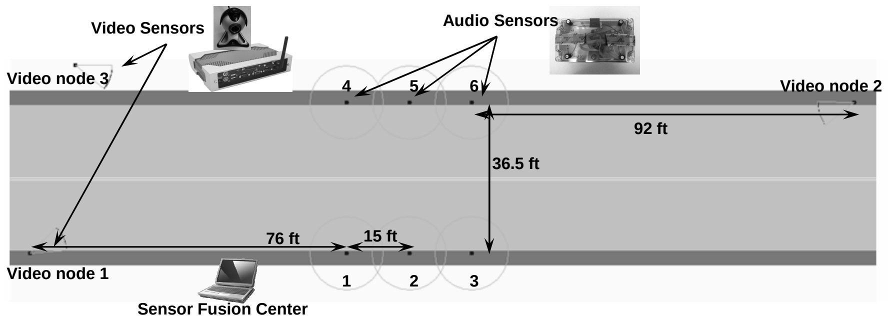

Fig. 11 Experimental setup

The deployment of our multimodal target tracking system is shown in Figure 11. We employ 6 audio sensors and 3 video sensors deployed on either side of a road. The complex urban street environment presents many challenges including gradual change of illumination, sunlight reflections from windows, glints due to cars, high visual clutter due to swaying trees, high background acoustic noise due to construction and acoustic multipath effects due to tall buildings. 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 are required. In our experimental setup, the sensor nodes are manually placed at marked locations and orientations. The audio sensors are placed on 1 meter high tripods to minimize audio clutter near the ground. An accurate self-calibration technique, e.g. [16, 28], is desirable for a target tracking system. Our experimental setup consists of two wireless networks as described in Section 3. The mote network is operating on channel 26 (2.480 GHz) while the 802.11b network is operating on channel 6 (2.437 GHz). Both the channels are non-overlapping and different from the infrastructure wireless network, which operates on channel 11 (2.462 GHz). We choose these non-overlapping channels to minimize interference and are able to achieve less than 2% packet loss.

We gather audio and video detection data for a total duration of 43 minutes. Table 1 presents the parameter values that we use in our tracking system. We run our

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

Table 1 Parameters used in experimental setup

sensor fusion and tracking system online using centralized sequential Bayesian estimation based on the product of likelihood functions. We also collect all the audio and video detection data for offline evaluation. This way we are able to experiment with different fusion approaches on the same data set. For offline evaluations, we shortlist 10 vehicle tracks where there is only a single target in the sensing region. The average duration of tracks is 4.25 sec with 3.0 sec minimum and 5.5 sec maximum. The tracked vehicles are part of an uncontrolled experiment. The vehicles are traveling on road at a speed of 20-30 mph speed.

Sequential Bayesian estimation requires a prior density of the target state. We initialize the prior density using a simple detection algorithm based on audio data. If the maximum of the audio detection functions exceeds a threshold, we initialize the prior density based on the audio detection.

In our simulations, we experiment with eight different approaches. We use audio-only, video-only and audio-video sensor data for sensor fusion. For each of these data sets, the likelihood is computed either as the weighted-sum or product of the likelihood function for the individual sensors. For the audio-video data, we use centralized and hybrid fusion. Following is the list of different target tracking approaches.

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)

The ground truth is estimated post-facto based on the video recording by a separate camera. The standalone ground truth camera is not part of any network, and have the sole responsibility of recording ground truth video. For evaluation of tracking accuracy, the center of mass of the vehicle is considered to be the true location.

Figure 12 shows the tracking error for a representative vehicle track. The tracking

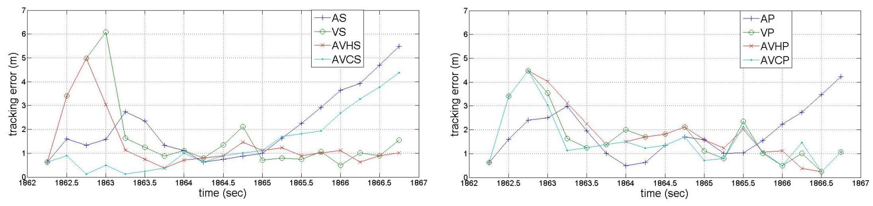

Fig. 12 Tracking error (a) weighted sum, (b) product

error when audio data is used is consistently lower than the case when the video data is used. When we use both audio and video data, the tracking error is lower than either of those considered alone. Figure 13 shows the determinant of the covariance of the target state for the same vehicle track. The covariance, which is an indicator of

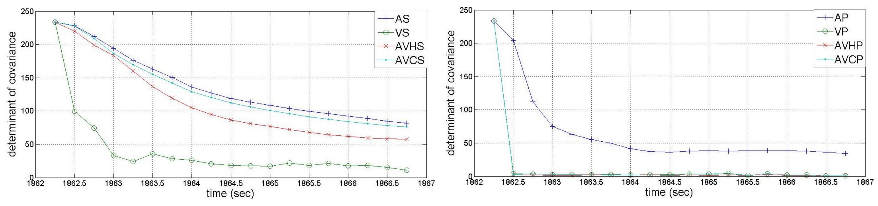

Fig. 13 Tracking variance (a) weighted sum, (b) product

uncertainty in the target state is significantly lower for product fusion than weighted-sum fusion. In general, covariance for audio-only tracking is higher than video-only tracking, while using both modalities lowers the uncertainty.

Figure 14(a) shows the tracking error in the case of fusion based on weighted-sum. The tracking error when using only video data shows a large value at time $t = 1067$ second. In this case, the video data has false peaks not corresponding to the target. The audio fusion works fine for this track. As expected, when we use both

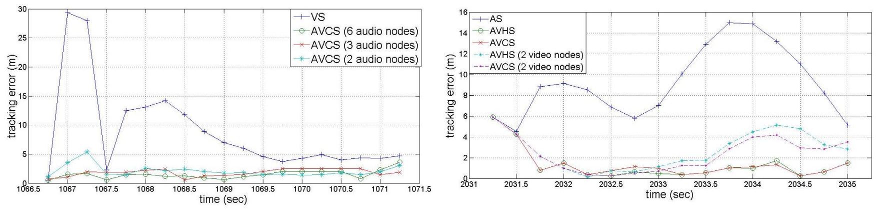

Fig. 14 Tracking error (a) poor video tracking, (b) poor audio tracking

audio and video data together, the tracking error is decreased. Further, Figure 14(a) shows the tracking error when using 6, 3 and 2 audio sensors for the centralized audio-video fusion. Using as few as two audio sensors can assist video tracking to disambiguate and remove false peaks. Similarly, when there is an ambiguity in the audio data, the video sensors can assist and improve tracking performance. Figure 14(b) shows the tracking error for a track where audio data is poor due to multiple sound sources. When fused with video data the tracking performance is improved. The figure shows tracking performance when using two and three video sensors for the centralized and hybrid fusion.

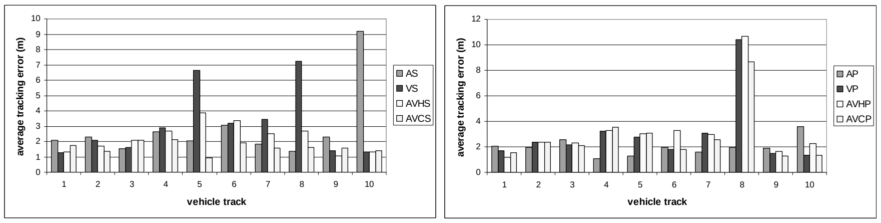

Fig. 15 Tracking errors (a) weighted sum, (b) product

Figure 15 shows average tracking errors for all ten vehicle tracks for all target tracking approaches mentioned above. Figure 16(a) averages tracking errors for all

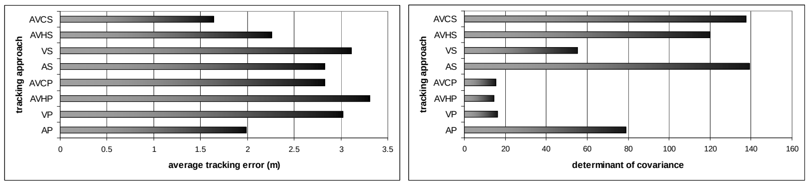

Fig. 16 (a) Average tracking errors and (b) average of determinant of covariance for all tracks

the tracks to compare different tracking approaches. Audio and video modalities are able to track vehicles successfully, though they suffer from poor performance in the 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 in the presence of multiple targets 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 the product of likelihood functions gives better performance but it is more vulnerable to sensor conflict and errors in sensor calibration. The weighted-sum approach is more robust to conflicts and sensor errors, but it suffers from high uncertainty. The centralized estimation framework consistently performs better than the hybrid framework.

Figure 17 shows the determinant of the covariance for all tracks and all approaches. Figure 16(b) presents averages of covariance measure for all tracks to

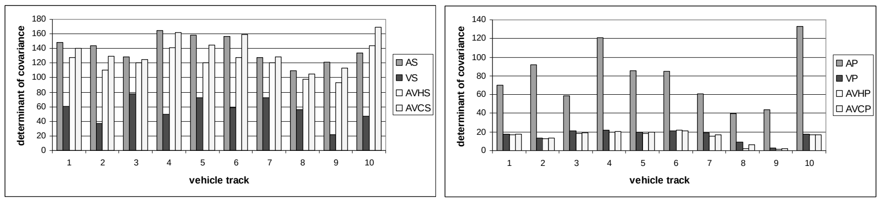

Fig. 17 Determinant of covariance (a) weighted sum, (b) product

compare the performance of tracking approaches. Among modalities, video sensors have lower uncertainty than audio sensors. Among the fusion techniques, product fusion produces lower uncertainty, as expected. There was no definite comparison between the centralized and hybrid approach, though the latter seems to produce lower uncertainty in the case of weighted-sum fusion.

The average tracking error of 2 meters is reasonable considering the fact that a vehicle is not a point source, and the cell size used in fusion is 0.5 meters.

8.2 MCMCDA

The audio and video data gathered for target tracking based on sequential Bayesian estimation is reused to evaluate target tracking based on MCMCDA. For MCMCDA evaluation, we experiment with six different approaches. We use audio-only (A), video-only (V) and audio-video (AV) sensor data for sensor fusion. For each of these data sets, the likelihood is computed either as the weighted-sum or product of the likelihood functions for individual sensors.

8.2.1 Single Target

We shortlist 9 vehicle tracks with a single target in the sensing region. The average duration of tracks is 3.75 sec with 2.75 sec minimum and 4.5 sec maximum. Figure 18 shows the target tracking result for two different representative vehicle tracks. The figure also shows the raw observations obtained from the multimodal sensor fusion and peak detection algorithms. Figure 19 shows average tracking errors for

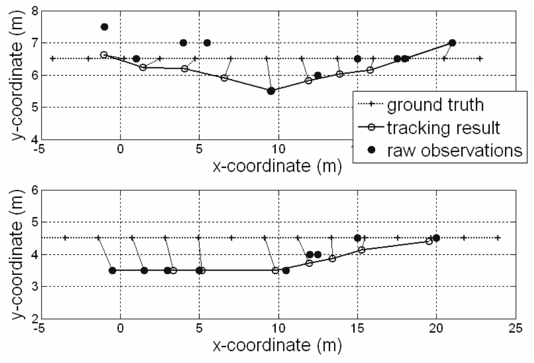

Fig. 18 Target Tracking (a) no missed detection (b) with missed detections

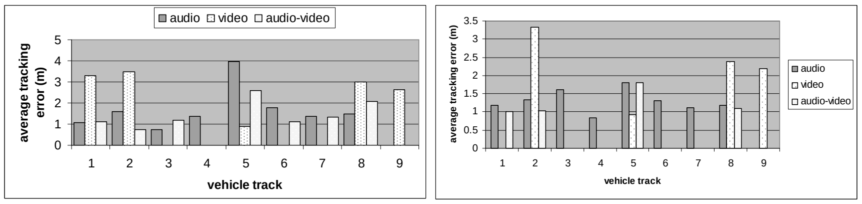

Fig. 19 Tracking errors (a) weighted sum fusion, and (b) product fusion

all vehicle tracks for the weighted-sum fusion and product fusion approaches. The missing bars indicate that the data association algorithm is not able to successfully estimate a track for the target. Figure 20 averages tracking errors for all the tracks to compare different tracking approaches. The figure also shows the comparison of the performance of tracking based on sequential Bayesian estimation to MCMCDA based tracking. The performance of MCMCDA is consistently better than sequential Bayesian estimation. Table 2 compares average tracking errors and tracking success across likelihood fusion and sensor modality. Tracking success is defined as the percentage of correct tracks that the algorithm is successfully able to estimate. Table 3 shows the reduction in tracking error for audio-video fusion over audio-only and video-only approaches. For summation fusion, the audio-video fusion is able to

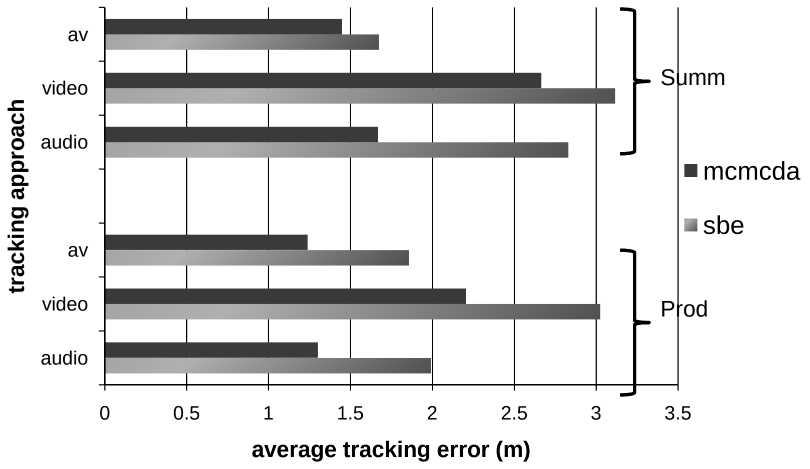

Fig. 20 Average tracking errors for all estimated tracks. A comparison with sequential Bayesian estimation based tracking is also shown.

		Average error (m)	Tracking success
Fusion	Summ	1.93	74%
	Prod	1.58	59%
Modality	Audio	1.49	89%
	Video	2.44	50%
	AV	1.34	61%

Table 2 Average tracking error and tracking success

reduce tracking error by an average of 0.26 m and 1.04 m for audio and video approaches, respectively. The audio-video fusion improves accuracy for 57% and 75% of the tracks for audio and video approaches, respectively. For the rest of the tracks, the tracking error either increased or remained same. Similar results are presented for product fusion in Table 3. In general, audio-video fusion improves over either audio or video or both approaches. Video cameras were placed at an angle along the

	Summ		Prod	
	Average error reduction (m)	Tracks improved	Average error reduction (m)	Tracks improved
Audio	0.26	57%	0.14	100%
Video	1.04	75%	0.90	67%

Table 3 Average reduction in tracking error for AV over audio and video-only for all estimated tracks

road to maximize coverage of the road. This makes video tracking very sensitive to camera calibration errors and camera placement. Also, an occasional obstruction in front of a camera confused the tracking algorithm which took a while to recover. An accurate self-calibration technique, e.g. [16, 28], is desirable for better performance of a target tracking system.

8.2.2 Multiple Targets

Many tracks with multiple moving vehicles in the sensing region were recorded during the experiment. Most of them have vehicles moving in the same direction. Only a few tracks include multiple vehicles crossing each other. Figure 21 shows

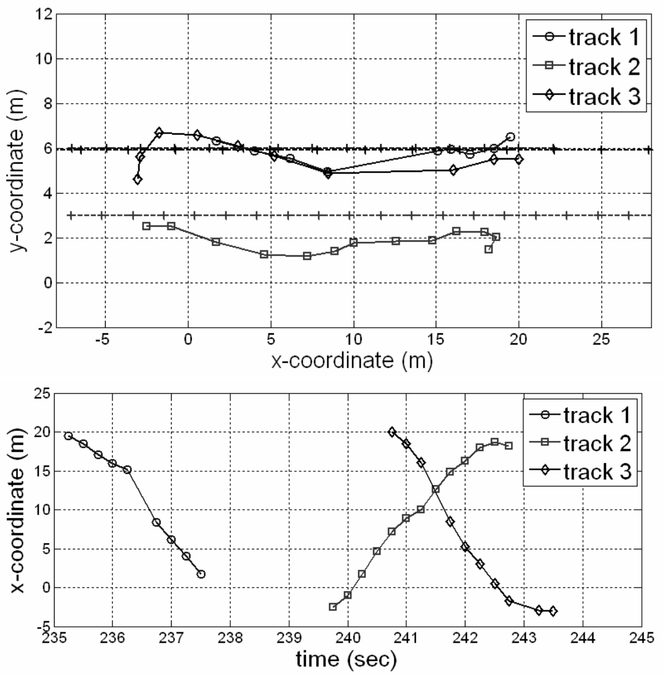

Fig. 21 Multiple Target Tracking (a) XY plot (b) X-Coordinate with time

the multiple target tracking result for three vehicles where two of them are crossing each other. Figure 21(a) shows the three tracks with the ground truth, while Figure 21(b) shows the x-coordinate of the tracks with time. The average tracking errors for the three tracks are 1.29m, 1.60m and 2.20m. Fig. 21 shows the result when only video data from the three video sensors is used. Multiple target tracking with audio data could not distinguish between targets when they cross each other. This is due to the fact that beamforming is done assuming acoustic signals are generated from a single source. Acoustic beamforming methods exist for detecting and estimating multiple targets [7].

9 Conclusions

We have developed a multimodal tracking system for an HSN consisting of audio and video sensors. We presented various approaches for multimodal sensor fusion and two approaches for target tracking, which are based on sequential Bayesian estimation and MCMCDA algorithm. We have evaluated the performance of the tracking system using an HSN of six mote-based audio sensors and three PC webcamera-based video sensors. We evaluated and compared the performance for both the tracking approaches. Time synchronization across the HSN allows the fusion of the sensor data. We have deployed the HSN and evaluated the performance by tracking moving vehicles in an uncontrolled urban environment. We have shown that, in general, fusion of audio and video data can improve the tracking performance. Currently, our system is not robust to multiple acoustic sources or multiple moving targets. An accurate self-calibration technique and robust audio and video sensing algorithms for multiple targets are required for better performance. A related challenge is sensor conflict that can degrade the performance of any fusion method and needs to be carefully considered. As in all sensor network applications, scalability is an important aspect that has to be considered as well.

References

1. A. M. Ali, K. Yao, T. C. Collier, C. E. Taylor, D. T. Blumstein, and L. Girod. An empirical study of collaborative acoustic source localization. In *IPSN '07: Proceedings of the 6th international conference on Information processing in sensor networks*, pages 41–50, 2007.
2. I. Amundson, B. Kusy, P. Volgyesi, X. Koutsoukos, and A. Ledeczi. Time synchronization in heterogeneous sensor networks. In *International Conference on Distributed Computing in Sensor Networks (DCOSS 2008)*, 2008.
3. M. J. Beal, N. Jojic, and H. Attias. A graphical model for audiovisual object tracking. volume 25, pages 828–836, 2003.
4. P. Bergamo, S. Asgari, H. Wang, D. Maniezzo, L. Yip, R. E. Hudson, K. Yao, and D. Estrin. Collaborative sensor networking towards real-time acoustical beamforming in free-space and limited reverberence. In *IEEE Transactions On Mobile Computing*, volume 3, pages 211–224, 2004.
5. S. T. Birchfield. A unifying framework for acoustic localization. In *Proceedings of the 12th European Signal Processing Conference (EUSIPCO)*, Vienna, Austria, September 2004.
6. 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.
7. 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.
8. T. Darrell, D. Demirdjian, N. Checka, and P. Felzenszwalb. Plan-view trajectory estimation with dense stereo background models. In *Proceedings. Eighth IEEE International Conference on Computer Vision (ICCV 2001)*, 2001.

9. M. Ding, A. Terzis, I.-J. Wang, and D. Lucarelli. Multi-modal calibration of surveillance sensor networks. In *Military Communications Conference, MILCOM 2006*, 2006.
10. E. Duarte-Melo and M. Liu. Analysis of energy consumption and lifetime of heterogeneous wireless sensor networks. In *IEEE Globecom*, 2002.
11. A. Elgammal, D. Harwood, and L. Davis. Non-parametric model for background subtraction. In *IEEE ICCV'99 Frame-Rate workshop*, 1999.
12. J. Elson, L. Girod, and D. Estrin. Fine-grained network time synchronization using reference broadcasts. In *Operating Systems Design and Implementation (OSDI)*, 2002.
13. N. Friedman and S. Russell. Image segmentation in video sequences: A probabilistic approach. In *Conference on Uncertainty in Artificial Intelligence*, 1997.
14. S. Ganeriwal, R. Kumar, and M. B. Srivastava. Timing-sync protocol for sensor networks. In *ACM SenSys*, 2003.
15. L. Girod, V. Bychkovsky, J. Elson, and D. Estrin. Locating tiny sensors in time and space: A case study. In *ICCD*, 2002.
16. L. Girod, M. Lukac, V. Trifa, and D. Estrin. The design and implementation of a self-calibrating distributed acoustic sensing platform. In *SenSys '06*, 2006.
17. D. Hall and J. Llinas. An introduction to multisensor data fusion. In *Proceedings of the IEEE*, volume 85, pages 6–23, 1997.
18. 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.
19. 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.
20. Z. Khan, T. Balch, and F. Dellaert. MCMC-based particle filtering for tracking a variable number of interacting targets. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27(11):1805–1918, Nov. 2005.
21. D. Koller, J. Weber, T. Huang, J. Malik, G. Ogasawara, B. Rao, and S. Russell. Towards robust automatic traffic scene analysis in realtime. In *IEEE Conference on Decision and Control*, 1994.
22. R. Kumar, V. Tsiatsis, and M. B. Srivastava. Computation hierarchy for in-network processing. 2003.
23. 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.
24. 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.
25. P. Levis, S. Madden, D. Gay, J. Polastre, R. Szewczyk, A. Woo, E. Brewer, and D. Culler. The emergence of networking abstractions and techniques in TinyOS. In *NSDI*, 2004.
26. J. Liu, J. Reich, and F. Zhao. Collaborative in-network processing for target tracking. In *EURASIP, Journal on Applied Signal Processing*, 2002.
27. M. Maroti, B. Kusy, G. Simon, and A. Ledeczi. The flooding time synchronization protocol. In *ACM SenSys*, 2004.
28. M. Meingast, M. Kushwaha, S. Oh, X. Koutsoukos, and S. S. Akos Ledeczi. Heterogeneous camera network localization using data fusion. In *In ACM/IEEE International Conference on Distributed Smart Cameras (ICDSC-08)*, 2008.
29. S. Oh, S. Russell, and S. Sastry. Markov chain Monte Carlo data association for general multiple-target tracking problems. In *Proc. of the IEEE Conference on Decision and Control*, Paradise Island, Bahamas, Dec. 2004.
30. K. Okuma, A. Taleghani, N. de Freitas, J. Little, and D. Lowe. A boosted particle filter: Multitarget detection and tracking. In *European Conference on Computer Vision*, 2004.
31. J. Piater and J. Crowley. Multi-modal tracking of interacting targets using Gaussian approximations. In *IEEE International Workshop on Performance Evaluation of Tracking and Surveillance*, 2001.

32. A. Poore. Multidimensional assignment and multitarget tracking. In I. J. Cox, P. Hansen, and B. Julesz, editors, *Partitioning Data Sets*, pages 169–196. American Mathematical Society, 1995.
33. D. Reid. An algorithm for tracking multiple targets. *IEEE Trans. Automatic Control*, 24(6):843–854, December 1979.
34. S. Rhee, D. Seetharam, and S. Liu. Techniques for minimizing power consumption in low data-rate wireless sensor networks. 2004.
35. 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.
36. C. Stauffer and W. Grimson. Learning patterns of activity using real-time tracking. In *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2000.
37. N. Strobel, S. Spors, and R. Rabenstein. Joint audio video object localization and tracking. In *IEEE Signal Processing Magazine*, 2001.
38. K. Toyama, J. Krumm, B. Brumitt, and B. Meyers. Wallflower: Principles and practice of background maintenance. In *IEEE International Conference on Computer Vision*, 1999.
39. J. Valin, F. Michaud, and J. Rouat. Robust localization and tracking of simultaneous moving sound sources using beamforming and particle filtering. In *Robot. Auton. Syst.*, volume 55, pages 216–228, March 2007.
40. P. Volgyesi, G. Balogh, A. Nadas, C. Nash, and A. Ledeczi. Shooter localization and weapon classification with soldier-wearable networked sensors. In *Mobisys*, 2007.
41. M. Yarvis, N. Kushalnagar, H. Singh, A. Rangarajan, Y. Liu, and S. Singh. Exploiting heterogeneity in sensor networks. In *IEEE INFOCOM*, 2005.
42. B. H. Yoshimi and G. S. Pingali. A multimodal speaker detection and tracking system for teleconferencing. In *ACM Multimedia '02*, 2002.
43. T. Zhao and R. Nevatia. Tracking multiple humans in crowded environment. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2004.
44. D. Zotkin, R. Duraiswami, H. Nanda, and L. S. Davis. Multimodal tracking for smart videoconferencing. In *In Proc. 2nd Int. Conference on Multimedia and Expo (ICME'01)*, 2001.