Joint Audio-Visual Tracking Using Particle Filters

Dmitry N. Zotkin

Perceptual Interfaces and Reality Laboratory, Department of Computer Science, University of Maryland Institute for Advanced Computer Studies, University of Maryland at College Park, College Park, MD 20742, USA
Email: dz@cs.umd.edu

Ramani Duraiswami

Perceptual Interfaces and Reality Laboratory, Department of Computer Science, University of Maryland Institute for Advanced Computer Studies, University of Maryland at College Park, College Park, MD 20742, USA
Email: ramani@umiacs.umd.edu

Larry S. Davis

Perceptual Interfaces and Reality Laboratory, Department of Computer Science, University of Maryland Institute for Advanced Computer Studies, University of Maryland at College Park, College Park, MD 20742, USA
Email: lsd@umiacs.umd.edu

It is often advantageous to track objects in a scene using multimodal information when such information is available. We use audio as a complementary modality to video data, which, in comparison to vision, can provide faster localization over a wider field of view. We present a particle-filter based tracking framework for performing multimodal sensor fusion for tracking people in a videoconferencing environment using multiple cameras and multiple microphone arrays. One advantage of our proposed tracker is its ability to seamlessly handle temporary absence of some measurements (e.g., camera occlusion or silence). Another advantage is the possibility of self-calibration of the joint system to compensate for imprecision in the knowledge of array or camera parameters by treating them as containing an unknown statistical component that can be determined using the particle filter framework during tracking. We implement the algorithm in the context of a videoconferencing and meeting recording system. The system also performs high-level semantic analysis of the scene by keeping participant tracks, recognizing turn-taking events and recording an annotated transcript of the meeting. Experimental results are presented. Our system operates in real time and is shown to be robust and reliable.

Keywords and phrases: audio-visual tracking, sensor fusion, Monte-Carlo algorithms.

1. INTRODUCTION

The goal of most machine perception systems is to mimic the performance of human and animal systems. A key characteristic of human systems is their multimodality. They rely on information from many modalities, chief among which are vision and audition. It is now apparent that many of the centers in the brain thought to encode space-time are activated by combinations of visual and audio stimuli [1]. However, the problems of computer vision and computer audition have essentially been performed on parallel tracks, with different research communities and problems. Capabilities of computers have now reached such a level that it is now possible to build and develop systems that can combine multiple audio and video sensors and perform meaningful joint-analysis of a scene, such as joint audio-visual speaker localization, tracking, speaker change detection and remote speech acquisition using beamforming techniques, which is necessary for the development of natural, robust and environmentally-independent applications. Applications of such systems include novel human-computer interfaces, robots that sense and perceive their environment, perceptive spaces for applications in immersive virtual or augmented reality, and so forth. In particular, applications such as video gaming, virtual reality, multimodal user interfaces, and video conferencing, require systems that can locate and track persons in a room through a combination of visual and audio cues, enhance the sound that they produce, and perform identification.

In this paper, we describe the development of a system that is able to process input from multiple video and audio sensors. The information gathered is used to perform both lower level analysis (robust object tracking including occlusion handling), higher level scene analysis (providing breakup of the audio meeting recording into pieces corresponding to activity of individual speakers), and speech quality

improvement (simple beamforming-based speech signal enhancement for the speech recognition engine). We present a probabilistic framework for combining results from the two modes and develop a particle filter based joint audio-video tracking algorithm. The availability of independent modalities allows to dynamically adjust the audio and video calibration parameters to achieve consistent tracks. Similarly, our multimodal tracker is able to robustly track through missing features in one of the modalities, and is more robust than trackers relying on one of the modes alone. The system developed is applied for smart videoconferencing and meeting recording, and for animal behavior studies.

The developed algorithm is an application of sequential Monte-Carlo methods (also known as particle filters) to 3D tracking using one or more cameras and one or more microphone arrays. Particle filters were originally introduced in the computer vision area in the form of the CONDENSATION algorithm [2]. Improvements of a technical nature to the condensation algorithm were provided by Isard and Blake [3], MacCormick and Blake [4], Li and Chellappa [5], and Philomin et al. [6]. The algorithm has seen applications to multiple aspects of both computer vision and signal processing. For example, a recent paper by Qian and Chellappa [7] describes a particle filter algorithm for the structure from motion problem using sparse feature correspondence which also performs the estimation of sensor motion from the epipolar constraint, and a recently published book [8] describes many different applications in signal detection and estimation. Overall, it can be said that particle filters provide effective solutions for challenging issues in different areas of computer vision and signal processing.

The development of multimodal sensor fusion algorithms is also an active research area. The applications seen include multisensor vehicle navigation system where computer vision, laser radar, sonar and microwave radar sensors are used together [9], recent papers on audio-visual person identification using support vector machine (SVM) classifier [10], multimodal speaker detection using Bayesian networks [11], multimodal tracking using inverse modeling techniques from computer vision, speech recognition and acoustics [12] and discourse segmentation using gesture, speech and gaze cues [13]. Our algorithm combines multimodality with a particle filter framework, which enables simple and fast implementation and on-the-fly multisensor system self-calibration by tracking the relative positions and orientations of the sensors together with the coordinates of the objects. We present experimental results showing the potential of the developed algorithm.

2. ALGORITHMS

The multimodal tracking system consists of several relatively independent components that produce sensor measurements and perform tracking and camera control. We describe the formulation of the multimodal particle filter, discuss how it can be modified to allow for a dynamic system self-calibration, and show how the measurement vector for the particle filter is obtained. We will also describe the

detection of turn-taking events and separation of the audio recording of the meeting into pieces corresponding to different talkers.

2.1. Particle filter formulation

Several different approaches can be used for multimodal tracking for videoconferencing. Probably the simplest method is a direct object detection in every frame by inverting the measurement equations and obtaining object positions from measurements. Significant drawbacks of this method are slow speed, and more important, the fact that a closed-form inversion of measurement equations may not exist or may not be numerically stable; in addition, the temporal inter-frame relationships between object positions are not exploited. The Kalman filter and the extended Kalman filter provide a statistically optimal tracking solution in the case of a Gaussian probability density function of a process; however, it cannot be used effectively for a process that is not modeled well by the Gaussian distribution. Particle filters address this problem effectively.

The particle filter algorithm provides a simple and effective way of modeling a stochastic process with arbitrary probability density function $p(S)$ by approximating it numerically with a cloud of points called particles in a process state space S. (We use S for the state space to avoid confusion with X which we use to denote the geometric coordinates only.) Other components of a particle filter framework are the measurement vector Z, the motion model and the likelihood equation. The measurements depend on the object state, and the object state is statistically derived from them. The motion model $S_{t+1} = F(S_t)$ describes the time evolution of the object state and the conditional posterior probability estimation function $P(Z|S)$ defines the likelihood of the observed measurement for a given point of a state space. (Note that in the particle filter framework, it is never required to invert the measurement equations; only the forward projection from the state space to the measurement space has to be computed, which is usually quite easy.) The update cycle consists of propagation of every particle in the state space according to the motion model, reweighting them in accordance with the obtained measurement vector and resampling the particle set to prevent degeneration and maintain an equiweighted set. The update algorithm is described below and is very similar to the original algorithm.

2.1.1 Update algorithm

Every particle in the set $\{s_i\}$, $i = 1, \ldots, N$, in the state space S has a weight π_i associated with it. This set is called *properly weighted* if it approximates the true PDF $P(s)$, so that for every integrable function $H(s)$,

$$E_{P(s)}\big(H(s)\big) = \lim_{N \to \infty} \frac{\sum_N H(s_i)\pi_i}{\sum_N \pi_i}. \tag{1}$$

Given a properly weighted set of particles at time t with equal weights $1/N$, it is possible to update it to reflect the new measurements obtained at time $t + \delta t$. The update algorithm is as follows.

(1) Propagate each particle s_i in time using the object *motion model* to obtain an updated particle set $\{s_i^*\}$.

(2) Obtain a new measurement vector Z and evaluate the *posterior probability density* π_i^* on $\{s_i^*\}$, $\pi_i^* = p(s_i^*|Z)$, which measures the likelihood of s_i^* given Z. This can be written using Bayes' rule

$$p(s_i^*|Z) = \frac{p(Z|s_i^*)p(s_i^*)}{p(Z)}, \qquad (2)$$

where $p(Z)$ is the prior probability of measurement, which is assumed to be a known constant, and $p(s_i^*) = 1/N$. Thus, $p(s_i^*|Z) = Kp(Z|s_i^*)$ for some constant K, and $p(Z|s_i^*)$ can be computed without inversion of the measurement equations.

(3) Resample from $\{s_i^*\}$ with probabilities π_i^*, and generate a new *properly weighted* set $\{s_i'\}$ with equal weights $1/N$ for each particle.

(4) Repeat steps (1)–(3) for subsequent times.

Several improvements to the original particle filter framework proposed by different researchers are implemented, including importance sampling and quasi-random sampling. They significantly improve the performance of the tracker.

2.2. Self-calibration

The particle filter is usually employed for tracking the motion of an object. However, (and this is one of the contributions of this paper), it can be used equally well to estimate the *intrinsic system parameters* or the sensor ego-motion. In a videoconferencing framework, there often exists uncertainty in the position of the sensors. For example, the position of a microphone array with respect to the camera can be measured with a ruler or determined from a calibrated video sequence. However, both methods are subject to measurement errors. These errors can lead to disagreement in audio and video estimations of the object position and ultimately to tracking loss. In another scenario, a multimodal tracking system with independent motion of a sensor requires estimation of sensor motion, which can be done simultaneously with tracking in the proposed framework. Such a system can include, for example, several moving platforms, each with a camera and a microphone array, or a rotating microphone array.

To perform simultaneous tracking with parameter estimation, we simply include the sensor parameters into the system state space. We should be careful, though, to avoid introducing too many free parameters as this will increase the dimensionality of the state space ("curse of dimensionality") and lead to poor tracking performance. We perform several experiments with synthetic data using one and two planar microphone arrays rotating independently and one and two rotating cameras. In all cases where at least one sensor position is fixed, tracking with simultaneous parameter estimation was successful in recovering both the object and the sensor motion. (When all sensors are free to rotate, there exist configurations in which it is impossible to distinguish between sensor and object motion. Multipoint self-calibration should be used in this case.) We also perform

an experimental study of a self-calibrating videoconferencing system. In our particular experimental setup, two cameras observe the room and a microphone array lies on the room floor. The self-geometry of the array is known with good precision, but the relative position of the array to the cameras is known only approximately and is recovered correctly during tracking.

2.3. Motion model

The motion model describes the temporal update rule for the system state. The tracked object state consists of three coordinates and three velocities of the object $[x \; y \; z \; \dot{x} \; \dot{y} \; \dot{z}]$, thus corresponding to a first-order motion model. To allow changes in the object state, a random excitation force F modeled by Gaussian with zero mean and normal deviation σ is applied to the velocity component. (The value of σ chosen depends on expected acceleration of the tracked object. If it is set too small, tracking can be lost as the tracker cannot follow the object quickly enough; if it is set too large, the predictive value of the model disappears. In our setup, $\sigma = 100 \, \text{m/s}^2$ in the experiments with fast moving free-flying bat which can accelerate quickly and make sharp turns, and $\sigma = 5 \, \text{m/s}^2$ in videoconferencing setup where people are being tracked.) The state update rule is

$$x(t + \delta t) = x(t) + \dot{x}\delta t,$$
$$\dot{x}(t + \delta t) = \dot{x}(t) + F\delta t, \qquad (3)$$

with similar expressions for $y, \dot{y}, z, \dot{z}$. When additional spatial parameters (position or rotation angle) are added for a sensor that is expected to be in motion, both the parameter and its first time derivative (velocity) are added, and the same motion model as in (3) is used. When parameters are added for a static sensor, the velocity is not used and the random excitation applies directly to the parameter. For example, for the case of two rotating arrays being used to track the object, the state vector consists of ten components $[x \; y \; z \; \phi_1 \; \phi_2 \; \dot{x} \; \dot{y} \; \dot{z} \; \dot{\phi}_1 \; \dot{\phi}_2]$, where ϕ_1 and ϕ_2 are the rotation angles of these arrays.

2.4. Video measurements

The video data stream is acquired from two color pan-tilt-zoom cameras. The relationship between image coordinates (u_i, v_i) and world coordinates (X, Y, Z) of the object (the camera projection equations) for the ith camera can be described using the simple direct linear transformation (DLT) model (see [14])

$$u_i = \frac{p_{11}X + p_{12}Y + p_{13}Z + p_{14}}{p_{31}X + p_{32}Y + p_{33}Z + 1}, \qquad (4)$$

$$v_i = \frac{p_{21}X + p_{22}Y + p_{23}Z + p_{24}}{p_{31}X + p_{32}Y + p_{33}Z + 1}. \qquad (5)$$

The matrix P_i has eleven parameters $\{p_{11}, \ldots, p_{14}, p_{21}, \ldots, p_{33}\}$ which in this model are assumed to be independent with $p_{34} = 1$. These parameters are estimated by using a calibration object of a known geometry placed

in the field of view of both cameras with both camera pan and tilt set to zero. The calibration object consists of 25 white balls on black sticks arranged in a regular spatial pattern; the three-dimensional coordinates of the balls are known within 0.5 mm. The image coordinates of every ball is determined manually from the image of the calibration object, thus giving 25 relationships between (X_j, Y_j, Z_j) and (u_{ij}, v_{ij}), $j = 1, \ldots, 25$, for the ith camera of the form (4) with the unknown parameters P. This overdetermined linear system of equations is then solved for P using least squares.

In the course of tracking, the video processing subsystem analyzes the acquired video frames and computes the likelihood of an observed video frame (measurement) given a system state. This can be done in two ways. One possible way is to first extract the object coordinates from the image by template matching over the whole image and finding the best match; then see how well the image object coordinates match the coordinates obtained by projecting the system state onto the measurement space. Another, more promising, approach is to take the whole image as a measurement, perform template matching at the image point to which the system state projects and report the matching score as a likelihood measure; this has the advantage of performing matching only at points where it is likely to find a match—that is around the true object position—and has the ability to handle multiple objects in the same frame. We use a simple face detection algorithm based on skin color and template matching for the initial detection and then perform head tracking based on shape matching and color histograms [15] after the detection is done.

We denote the image coordinates of the object as $(\tilde{u}_i, \tilde{v}_i)$ (the tilde denotes measured values). Object localization is described in a later subsection. Given the system state S (and the object coordinates (x, y, z) as part of S), the data likelihood estimation $P_v(Z_v|S)$ is computed as follows. First, we need to account for the (known) current camera pan and tilt angle. To do that, we simply rotate the world around the camera origin using the same pan and tilt angles obtaining a source position (x', y', z') in the coordinate system of the rotated camera. Then, these coordinates are plugged into the DLT equations (4) to obtain the corresponding image object position (u_i, v_i). The error measure ε_v for the video localization is given by a sum over N cameras

$$\varepsilon_v^2 = \frac{1}{N} \sum_{i=1,\ldots,N} \left[(u_i - \tilde{u}_i)^2 + (v_i - \tilde{v}_i)^2 \right], \quad (6)$$

and the data likelihood estimation $P_v(Z_v|S)$ as

$$P_v(Z_v|S) = \frac{1}{\sqrt{2\pi}\sigma_v} e^{-\varepsilon_v^2/2\sigma_v^2}, \quad (7)$$

where σ_v is the width of a corresponding Gaussian reflecting the level of confidence in the video measurements. (We introduce the notion of the error measure exclusively to split the complicated formula into two parts; the data likelihood can be easily expressed directly over measurements as well.)

2.5. Audio measurements

The audio localization is based on computing the time differences of arrivals (TDOA) between channels of the microphone array. TDOA values are computed by a generalized cross-correlation algorithm [16]. Denote the signal at the ith microphone as $h_i(t)$ and its Fourier transform $H_i(\omega)$, then, the time difference $\hat{\tau}_{ij}$ that maximizes the value of the generalized cross-correlation between channels i and j can be computed quickly

$$\hat{\tau}_{ij} = \arg \max_t R_{ij}(t), \quad R_{ij}(t) \overset{\text{FFT}}{\Longleftrightarrow} R_{ij}(\omega), \quad (8)$$

$$R_{ij}(\omega) = W(\omega)H_i(\omega)H_j^*(\omega), \quad (9)$$

where $W(\omega)$ is a weighting function and is equivalent to the inverse noise spectrum power $|N(\omega)|^{-2}$, and $H_j^*(\omega)$ denotes the complex conjugate of $H_j(\omega)$. The noise power spectrum is estimated during silence periods.

To be able to use these measurements in the filtering framework, we have to define the likelihood of observing an audio measurement vector Z_a consisting of particular measurements $\{\tilde{\tau}_{ij}\}$, $i, j = 1, \ldots, N$, for a given system state S. It is easy to do that. Assume that the state S corresponds to the source position (x_s, y_s, z_s) and microphone positions (x_i, y_i, z_i), $i = 1, \ldots, N$. (In case of moving sensors, the microphone positions may change over time.) Then, define the distance χ_i from the source to the ith microphone as $\chi_i^2 = (x_s - x_i)^2 + (y_s - y_i)^2 + (z_s - z_i)^2$. The TDOA set for this system state is simply $\tau_{ij} = (\chi_j - \chi_i)/c$, where c is the sound speed. Now, we define the audio error measure ε_a between TDOAs for the state S and the observed set of TDOAs as

$$\varepsilon_a^2 = \binom{N}{2}^{-1} \sum_{1 \le i < j \le N} (\tau_{ij} - \tilde{\tau}_{ij})^2 \quad (10)$$

and the data likelihood estimation $P_a(Z_a|S)$ as

$$P_a(Z_a|S) = \frac{1}{\sqrt{2\pi}\sigma_a} e^{-\varepsilon_a^2/2\sigma_a^2}. \quad (11)$$

(On a side note, a probabilistic audio source localization algorithm, similar to the one described here, is computationally more expensive but is superior to the algorithms that use pairs of cross-correlation values and perform intersection of multiple cones of equivalent time delays, since one invalid cross-correlation can throw the resulting intersection vastly off position. In contrast, a probabilistic approach does not require unstable inverse calculations and is shown to be more robust—see, for example, [17].)

2.6. Occlusion handling

The combined audio-video data likelihood estimation for the multimodal particle filter $P(Z|S)$ is obtained by multiplication of the corresponding audio and video parts, such that, $P(Z|S) = P_v(Z_v|S)P_a(Z_a|S)$. Note that the final formula consists simply of a product of multiple Gaussians, one

per component of the measurement vector. This property allows the tracker to handle partial measurements, which can be due to occlusion of the tracked object from one of the cameras, or due to missing values for some of the TDOA estimations due to noisy or weak audio channels. In these cases, the part of the product that corresponds to the missing measurement is simply set to a constant value, meaning that the missing measurement does not give any information whatsoever. The tracking can still be performed as long as there is sufficient enough information to localize the object, no matter which particular sensor it is coming from. This allows the tracker to perform well when separate audio and video trackers would fail. We performed some experiments with real data and show in a later section the recovered track of the person through occlusion in one camera.

Occlusion handling and misdetection handling are also simplified by the underlying mechanisms of the particle filter. The PDF of the process is concentrated around the area in the system state space which the system is predicted to occupy at the next time instant, thus vastly decreasing the probability of misdetection since only the space near the predicted system state is densely sampled. If there is insufficient information available to perform tracking due to full or partial occlusion, the PDF of the process begins to widen over time, reflecting an uncertainty in the determination of the system state. The PDF still continues to be clustered around the point in the state space where the object is likely to reappear again, greatly improving the chances of successfully reacquiring the object track after the occlusion clears. If the object is not detected for such a long time that the width of the PDF reaches a certain threshold, the tracker is reinitialized using a separate detection algorithm (described below) and tracking is started over.

2.7.   Face detection and tracking

To initially locate people in the scene, we use a template matching algorithm on a skin color image which works sufficiently well in the videoconferencing environment. The assumption for the method to work is that people are facing the camera, which is usually true for videoconferencing. Our face detection algorithm is described in [18]; here, we give only a brief outline of the processing. The skin color is detected using R/B and G/B color intensity ratios γ_{rb} and γ_{gb} for a pixel with intensity $I = (R, G, B)$. These are compared to the "correct" values which correspond to the skin color $\hat{\gamma}_{rb}(I)$ and $\hat{\gamma}_{gb}(I)$ which are acquired by hand-localization of the face area in several sample images. Due to nonlinearity of the camera CCDs, these reference values depend on the brightness of the pixel in the scene; the functions $\hat{\gamma}_{rb}(I)$ and $\hat{\gamma}_{gb}(I)$ are obtained by sampling sample face images at pixels with different intensities. Then, if the following three conditions are satisfied, the pixel is assumed to have the skin color:

$$\hat{I}_l < I < \hat{I}_h,$$
$$\hat{\gamma}_{rb} - \zeta < \gamma_{rb} < \hat{\gamma}_{rb} + \zeta, \tag{12}$$
$$\hat{\gamma}_{gb} - \zeta < \gamma_{gb} < \hat{\gamma}_{gb} + \zeta.$$

The first condition rejects too dark or too bright pixels since they are often mis-recognized as skin color pixels due to nonlinearity of the camera CCD. The second and third conditions perform actual testing for the skin color. In our implementation, $\hat{I}_l = 0.1$, $\hat{I}_h = 0.9$, and $\zeta = 0.12$.

Then, the image is divided into blocks of 8×8 pixels. These blocks are classified according to the number of skin color pixels inside, and a connected components algorithm is executed on the blocks to find skin color blobs. For every blob found, template matching is performed with a simple oval-shaped template with different template center positions and template sizes. If the best score is less than a certain threshold, the skin color blob is rejected. Otherwise, some heuristic features that are characteristic to the face image are tested (eyes, lips, nose, and forehead areas). If these features are present, the algorithm decides that a face image is found. Experimental results show that the algorithm is sufficiently fast to operate in real-time, robust to illumination changes, and capable of detecting multiple faces.

After successful localization, the head tracking algorithm described in [6] is invoked on an image sequence, and the output of this subtracker constitutes video measurements. The tracking algorithm is based on the head tracking using shape matching and object color histogram. (In principle, it can be incorporated directly into the main tracker.) The head is modelled by an ellipse with a fixed vertical orientation and a fixed aspect ratio of 1.2 similar to [15]. The ellipse state is given by $\mathbf{s} = (x, y, \sigma)$, where (x, y) is the center of the ellipse and σ is the minor axis length of the ellipse. We use quasi-random points for sampling instead of the standard pseudo-random points since these points improve the asymptotic complexity of the search (number of points required to achieve a certain sampling error), can be efficiently generated and are well spread in multiple dimensions (see [6] for details). For a given tolerance to tracking error, the quasi-random sampling needs a significantly lower number of sampling points (about 1/2) as compared to pseudo-random sampling, thereby speeding up the execution of the algorithm significantly. Our measurement model is a combination of two complementary modules (see [15] for why this is good), one that makes measurements based on the object's boundary and the other that focuses on the object's interior (color histograms [19]). Figure 1 shows a sample screenshot from the face detection algorithm and three frames from the head tracking sequence in the case where two persons are presented. The tracker is able to tolerate temporary occlusions and switches back to the correct target after the occlusion is cleared.

2.8.   Turn-taking detection

For applications in videoconferencing, meeting recording, or a surveillance system, it is often desirable to know the high-level semantic structure of the scene and to provide an annotated transcript of the meeting. This information can be later used for content-based retrieval purposes. Our system can create such an annotated transcript. Currently, no speech recognition is performed; the only information available is
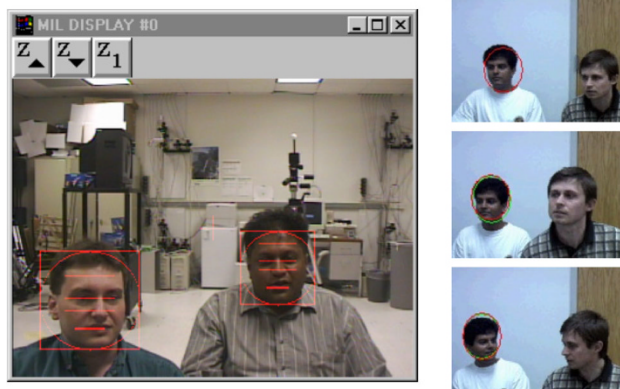
FIGURE 1: Sample screenshot from the face detection algorithm and three frames from a sequence of head tracking.

the set of associations between the segments of an audio recording and the thumbnails of a corresponding speaker.

To perform audio annotation, we detect the speaker change event during the tracking. The speaker change event is deemed to have occurred when (a) the audio localization data significantly disagree with the position of current tracked speaker, and (b) the face is recognized at or near the position of a new sound source. The color histogram of the image of the speaker is used to maintain identity of speakers. The recorded turn-taking sequence is used to segment the audio data into parts corresponding to individual speakers. Examples of such recordings are given later in the experimental results section.

We also optionally perform acoustic beamforming using the determined position of the speaker, as provided by the tracking algorithm. Simple delay-and-sum beamforming is used, achieving SNR gain of about 7 dB. The beamforming algorithm removes noise and interference from the recorded voice, allowing a speech recognition engine to be used on the recorded audio portions [18].

3. SYSTEM SETUP

To evaluate the suitability and performance of the developed tracking and event detection algorithms, we have built an experimental system which includes two cameras and two microphone arrays. We use two different setups, one of which is targeted for videoconferencing applications and the other for ultrasonic sound localization. In this section, we briefly describe these setups.

The videoconferencing setup includes two cameras and two microphone arrays. A single high-end office PC (dual PIII-933 MHZ Dell PC under WinNT) is used. The video data is acquired using two Sony EVI-D30 color pan-tilt-zoom cameras that are mounted on two tripods to form a wide-baseline stereo pair. Pan, tilt, and zoom of these cameras are controlled by software through a computer serial port for videoconferencing translation. The video stream is captured using two Matrox Meteor II cards. Two microphone arrays are attached to the room wall above the cameras. Each

array consists of 7 small button Panasonic microphones in a circular arrangement. The signal is digitized using a 12-bit PowerDAQ ADC board at 22.05 KHz per channel. Parallel programming is used to utilize both processors effectively, achieving a frame rate of a combined audio-visual tracking system of approximately 8 frames per second. Much higher frame rates can be achieved by performing audio and video analysis on separate networked machines.

The ultrasonic tracking system that is used for more precise localization experiments is set up in a partially anechoic room that is used for bat behavioral studies. The video data is acquired using two digital Kodak MotionCorder infrared cameras at a frame rate of 240 frames per second. (The room is illuminated only by infrared light during the experiments to ensure that the bat uses exclusively acoustic information for navigation.) The video stream is recorded on a digital tape and later digitized using a video capture card. The audio stream is captured using seven Knowles FG3329 ultrasonic microphones arranged in an L-shaped pattern on the room floor. The bat ultrasonic chirps consist of downward sweeping frequency-modulated signals ranging from 20 to 50 KHz. The microphone output is digitized at 140 KHz per channel and captured using an IoTech Wavebook ADC board. Joint audio-visual bat tracking is performed using the described algorithms. The results show that the self-calibration indeed allows for automatic compensation of inaccuracies in knowledge of sensor positions.

4. RESULTS

We perform several experiments with synthetic and real data in both operating environments to test the performance and robustness of the tracking algorithms. First, we evaluate algorithm performance on synthetic data using fixed cameras and fixed microphone array positions. Then, we test the self-calibration ability of the algorithm by introducing an error in the microphone array position. The third experiment deals with the case when both the object and the sensors are in motion; we show that for the case of two independently rotating microphone arrays, the system can recover both the object motion and the array rotations.

Then, we performed experiments with real data for the sound-emitting object tracking in both setups. We show that the algorithm tracks real objects well, the self-calibration is performed along with the tracking to bring audio and video tracks in agreement, and the algorithm is capable of tracking through occlusions.

4.1. Synthetic data

First, we test algorithm performance in the case when the ground truth is available. Using the ultrasonic tracking system setup, we synthesize the track of an object moving along a spiral trajectory for one second. The trajectory $(X(t), Y(t), Z(t))$ is given by

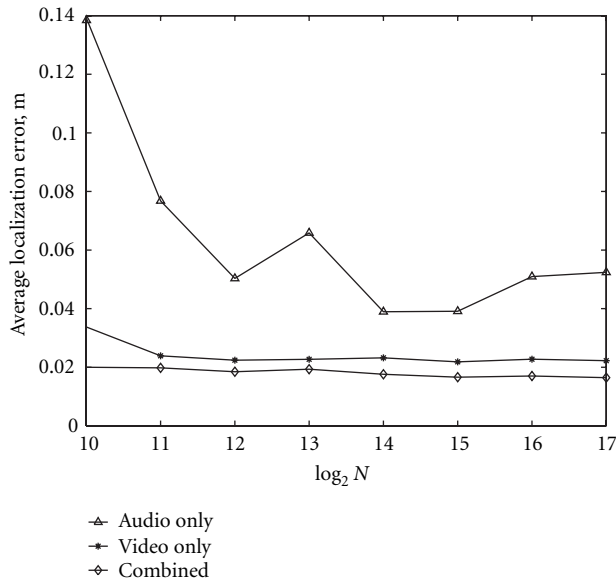$$X(t) = \sin(2\pi t), \quad Y(t) = 2 - t, \quad Z(t) = \cos(2\pi t), \quad t \in [0, 1].$$
$$(13)$$

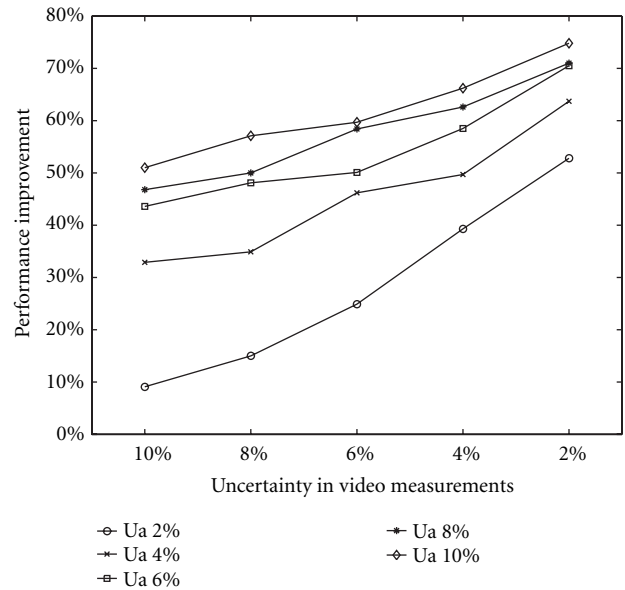FIGURE 2: Unimodal and multimodal tracking performance.

FIGURE 3: Plot of the percentage improvement in the performance of audio-video tracker versus the performance of audio-only tracker for different combinations of audio and video measurement uncertainty.

All parameters of the system are taken from the real setup. The frame rate is set to the 240 frames per second corresponding to the real data. At every frame, the measurement vector corresponding to the true object position is computed. Then, a random Gaussian noise with zero mean and deviation of $\sigma_v = 3\%$ for video measurements and $\sigma_a = 8\%$ for audio measurements is added to every component of the vector. The tracker is run on an obtained synthetic data trace and average tracking error is computed over 128 runs for different number of particles. In Figure 2, the average tracking error for video-only based tracking (with all acoustic measurements omitted), for audio-only based tracking and for multimodal tracking is plotted versus log-number of particles. Note that the horizontal axis is logarithmic and the number of particles ranges from 1024 to 131072. It can be seen that the performance of the combined tracker is better than for both unimodal cases, and the performance increases as the number of particles grows. The smallest tracking error obtained is approximately 16.5 mm; this is an almost threefold improvement over a pure object detection in every frame without tracking, which gives an error of approximately 38.3 mm.

Since the plots in Figure 2 represents only one combination of σ_v and σ_a, we also tested the performance of the tracking algorithm for different combinations of σ_v and σ_a to see if a consistent performance improvement is obtained with a second modality. Figure 3 shows the improvement of the performance of the combined audio-video tracker relative to the performance of audio-only tracker (i.e., the effect of adding the video modality to the tracker). The performance improvement is defined as a percentage decrease of the tracking error (if the error is halved, the performance improvement is 50%). Every point in the plot is computed by averaging results from 128 runs; 4096 particles were used

in the simulations. Five curves are plotted for different levels of noise contamination of the audio-only (base) tracker. The values of the standard deviation of the audio measurement noise, σ_a, are shown as "Ua" in the legend, and each curve shows the dependence of the improvement on σ_v. For example, the bottom curve reflects the addition of video modality with different degree of contamination (σ_v varying from 2% to 10%) to a case where the audio modality is quite accurate ($\sigma_a = 2\%$). Note that for the abscissa the measurements are cleaner towards the right edge of the plot. It can be seen that addition of noisy video modality to clean audio ($\sigma_a = 2\%$, $\sigma_v = 10\%$, left end of the bottom curve) improves the performance only slightly (by about 10%), as can be reasonably expected, and addition of clean video modality to clean audio modality ($\sigma_a = \sigma_v = 2\%$) improves the performance by about 50%, which is also reasonable. The top curve represents the opposite case when the audio modality is contaminated significantly ($\sigma_a = 10\%$); when clean video is added (right point of the top curve), the tracking error decreases by 75%, and when noisy video is added to noisy audio the improvement is again about 50%. Indeed, it can be seen from the plots that the performance improvement is about 50% when $\sigma_v = \sigma_a$. The performance gain is small when a noisy modality is added to a cleaner one and larger in the opposite case, but the gain is always present. Figure 4 represents the complementary case when audio modality is added to the video-only tracker. Five curves for different levels of noise contamination of the video-only (base) tracker are plotted, and the same trends can be observed. The important results shown by this experiment is that the performance improvement is consistent and systematic, that the modalities have the same relative importance and that the addition of even a
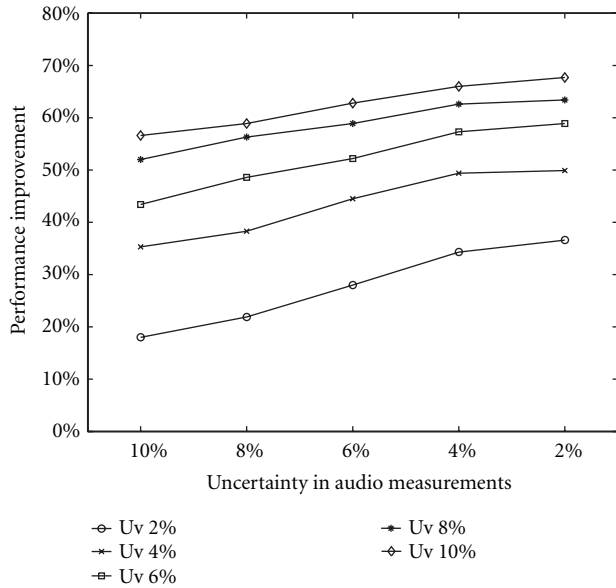
FIGURE 4: Plot of the percentage improvement in the performance of audio-video tracker versus the performance of video-only tracker for different combinations of audio and video measurement uncertainty.

FIGURE 5: Rotating sensor motion estimation.

seriously contaminated modality to a clean one produces noticeable performance gain, when both are present, and provides tracker robustness when one of the modalities is absent.

Then, the sensor motion recovery capability of the algorithm was tested. We used two L-shaped microphone arrays placed on the ground, rotating with different speeds of 0.5 and 0.25 radians per second in opposite directions. The object is moving along the same spiral trajectory as before. The rotation was modeled by adding two rotation angles and two rotational velocities into the state of the system. The measurement vector was computed using true microphone coordinates and the object position. Then, random Gaussian noise with the same parameters as before was added to the measurement vector. Due to lack of space, we show only one result here, which corresponds to the simultaneous tracking and sensor motion recovery using only one fixed camera. The algorithm succeeds in tracking, despite the fact that using any sensor alone is not sufficient to recover full object motion and the sensor's relative geometry is constantly changing. We show the plot of recovered sensor motion in Figure 5; the solid lines correspond to the true sensor rotation angles and the dashed lines are the estimations computed by the tracking algorithm. The object tracking error for this set of experiments is only slightly increased (approximately 21.4 mm) compared to the case of two static arrays and two static cameras (16.5 mm). The same results were obtained for the case of two rotating cameras and one fixed microphone array.

4.2. Real data

We use the developed algorithms to perform tracking of the echolocating bat in a quiet room. The bat is allowed to fly freely in 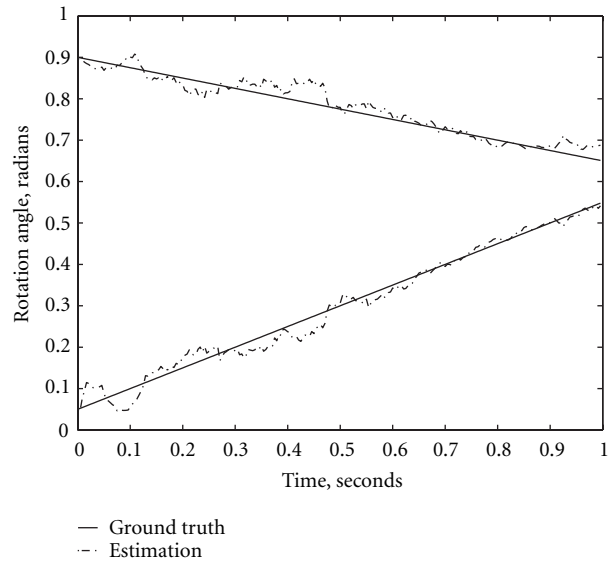the flight area and to hunt for a food item (a mealworm) suspended from the ceiling. In earlier experiments, we noticed that there was disagreement between bat trajectories recovered by audio and video means, although their shapes were similar. This was attributed to the fact that microphone coordinates were determined using the image of the microphone array from two calibrated cameras, which is not very accurate for the points far from the area where the calibration object was located. That led to the idea to perform adjustment of the microphone array position and orientation as tracking progresses. The array is built of a long L-shaped tube with microphones attached to it, so the relative positions of sensors within the array are known exactly. Therefore, we introduce three additional parameters—(x_a, y_a) positions of the array center and the rotation angle around the center θ_a—into the state vector of the system. Since the array lies on the floor, these parameters fully describe possible inaccuracy of the array placement. The tracking is performed in the nine-dimensional space $[x \; y \; z \; \dot{x} \; \dot{y} \; \dot{z} \; x_a \; y_a \; \theta_a]$ to simultaneously estimate the bat trajectory and the array position.

The results from one of the cases are shown in Figure 6. The bat flies from right to left, and the plot shows a plan view of the room. The solid line corresponds to the bat position estimated by video means only. The crosses are the audio estimations; they are discrete because the bat emits echolocation calls only intermittently. The bat behavioral pattern can be seen in the picture as infrequent vocalizations in the beginning of trajectory (search stage), a series of frequent calls in the middle (target approach stage) and the following silence (target capture stage); after that, the bat is again in the search mode. It can be seen from the track that there is a disagreement of about 0.2 meters between video and audio position estimations. The multimodal tracker with a fixed microphone array position estimated from video is run first. The output is shown in the plot with a dashed line. The
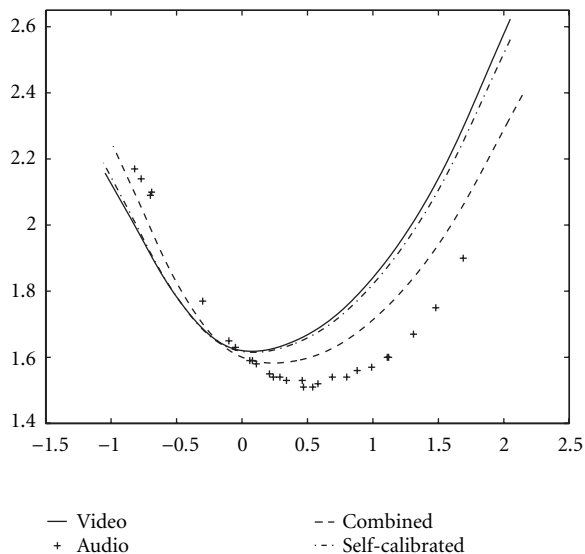
— Video　　　　　 − − Combined
  + Audio　　　　　 ·−· Self-calibrated

FIGURE 6: An object track recovered with and without self-calibration.

— Video
  + Audio
 ·−· Combined

FIGURE 7: Track of the (X, Y)-coordinates of a person through a simulated occlusion.

combined trajectory correctly lies in-between the audio and video tracks. Still, it is desirable to eliminate misalignment between modalities; to perform that, we run a self-adjusting tracker. The output is shown with a dotted line. The new trajectory lies substantially closer to the video estimation, and after a while the parameters describing array shift (x_a, y_a, θ_a) stabilize around values $(-0.22, -0.17, 0.067)$ which presumably correspond to the error in the array placement. The experiment shows that the tracker successfully recovers both the bat trajectory and the error in the sensor placement.

4.3. Occlusion handling

Another advantage of the proposed multimodal tracking algorithm is its ability to handle temporary absence of some measurements. As described before, this is done by setting the members of the cumulative data likelihood that correspond to the missing measurements to constant values. For the video measurement, the measurement is marked as missing if the face detector was not able to find a face in an image. For the audio TDOA values, the measurement is not used if it does not pass certain consistency checks (more details in [18]). To demonstrate the possibility of tracking through occlusion, in Figure 7 we show a case of a speaking person tracked in a videoconferencing setup.

The plot shows coordinates of a speaking person moving from left to right and going down and up in the meantime. The video-only based trajectory estimation is shown as a solid line and is obtained using the face detector described previously. The crosses show the successive audio estimations of the speaker position. The audio localization is less accurate in the videoconferencing setup since the array baseline and the discretization frequency are substantially less than in the anechoic room setup. Still, the audio estimations follows the video track pretty well (note that the whole vertical axis span is only 0.5 meters). We simulate the face occlusion
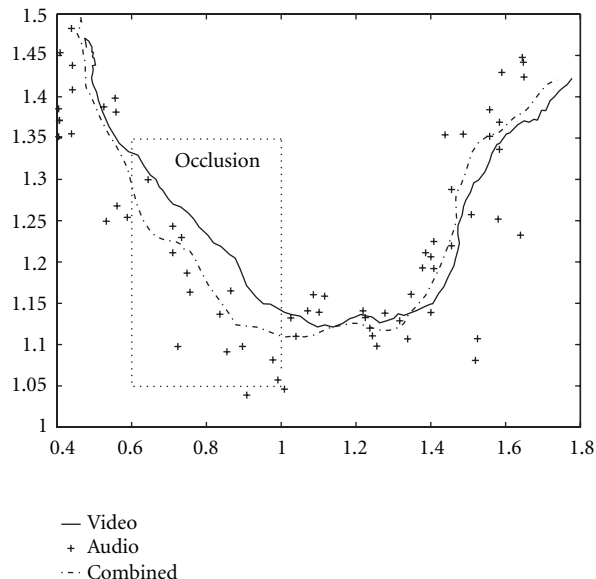
in one camera field of view by omitting the measurements from one camera when the person is within a marked rectangular area on the plot. The output of the tracker is shown as the dotted line. Tracking is performed successfully using partial measurements; the tracker output deviates from the video trajectory during occlusion since the audio information gets higher relative weight now, but still stays close to the correct trajectory. The tracker recovers quickly once the occlusion is cleared.

4.4. Annotated meeting recording

The developed multimodal tracking system has the ability to detect change in an active speaker and to rotate the active videoconferencing camera to point to the currently active speaker. In addition, the algorithm segments the audio recording of a meeting into pieces corresponding to the activity of individual speakers. We collected multimodal data during three simulated meetings of different types (lecture-type meeting where there is one primary speaker and occasional short interruptions occur, seminar-type meeting where speaker roles are equal and typical length of a speech segment by one person is significant, and informal talk or chat between participants where speaker changes and interruptions are quite frequent). Figure 8 shows the sequence of speaker changes for those three sequences. The time axis is horizontal and covers 80 seconds of meeting time. The bold line in the plot indicates the active speaker. Small icons attached to the tracks show the identities of individual speakers automatically captured and stored by the system. An audio recording of the meeting, enhanced by beamforming, is subdivided according to the turn-taking sequence and later is used to select parts corresponding to activities of individual speakers. A separate graphical user interface can be used later to retrieve several such recordings at once and selectively play
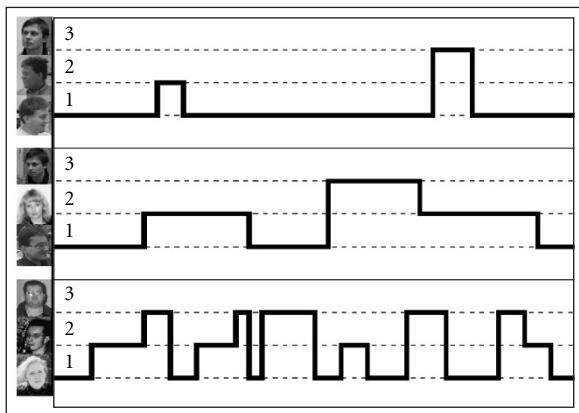
FIGURE 8: Three samples of turn-taking sequences (speaker versus time). Speaker icons show the identity of each speaker as automatically captured by system.

back recordings or parts of recordings containing only the speaker(s) of interest.

5. SUMMARY AND CONCLUSIONS

We have developed a multimodal sensor fusion tracking algorithm based on particle filtering. The posterior distribution of the system intrinsic parameters and the tracked object position are approximated with a set of points in combined system-object state space. Experimental results from a developed real-time system are presented showing that the tracker is able to seamlessly integrate multiple modalities, cope with temporary absence of some measurements and perform self-calibration of a multisensor system simultaneously with object tracking.

REFERENCES

[1] B. Stein, P. Laurienti, T. Stanford, and M. Wallace, "Neural mechanisms for integrating information from multiple senses," in *Proc. IEEE International Conference on Multimedia and Expo*, New York, NY, USA, July 2000.

[2] M. Isard and A. Blake, "CONDENSATION: Conditional density propagation for visual tracking," *International Journal of Computer Vision*, vol. 29, no. 1, pp. 5–28, 1998.

[3] M. Isard and A. Blake, "ICONDENSATION: Unifying low-level and high-level tracking in a stochastic framework," in *Proc. 5th European Conference on Computer Vision*, vol. 1, pp. 893–908, Freiburg, Germany, June 1998.

[4] J. MacCormick and A. Blake, "Probabilistic exclusion and partitioned sampling for multiple object tracking," *International Journal of Computer Vision*, vol. 39, no. 1, pp. 57–71, 2000.

[5] B. Li and R. Chellappa, "Simultaneous tracking and verification via sequential posterior estimation," in *Proc. IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, Hilton Head, SC, USA, June 2000.

[6] V. Philomin, R. Duraiswami, and L. Davis, "Quasi-random sampling for condensation," in *Proc. 6th European Conference on Computer Vision*, vol. 2, pp. 134–149, Dublin, Ireland, June 2000.

[7] G. Qian and R. Chellappa, "Structure from motion using sequential Monte-Carlo methods," in *Proc. International Conf. on Computer Vision*, vol. 2, pp. 614–621, Vancouver, Canada, July 2001.

[8] A. Doucet, N. Freitas, and N. Gordon, Eds., *Sequential Monte-Carlo Methods in Practice*, Springer-Verlag, New York, NY, USA, 2001.

[9] K. Cheok, K. Smid, and D. McCune, "A multisensor-based collision avoidance system with application to a military HMMWV," in *Proc. IEEE Conf. on Intelligent Transportation Systems*, Dearborn, Mich, USA, October 2000.

[10] S. Ben-Yacoub, J. Luttin, K. Jonsson, J. Matas, and J. Kittler, "Audio-visual person verification," in *Proc. IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, Fort Collins, Col, USA, June 1999.

[11] V. Pavlovic, A. Garg, J. Regh, and T. Huang, "Multimodal speaker detection using error feedback dynamic Bayesian networks," in *Proc. IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, Hilton Head, SC, USA, June 2000.

[12] G. Pingali, G. Tunali, and I. Carlborn, "Audio-visual tracking for natural interactivity," in *Proc. 7th ACM International Conference on Multimedia*, vol. 1, pp. 373–382, Orlando, Fla, USA, 30 October–5 November 1999.

[13] F. Quek, D. McNeill, R. Bryll, et al., "Gesture, speech and gaze cues for discourse segmentation," in *Proc. IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, vol. 2, pp. 247–254, Hilton Head, SC, USA, June 2000.

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

[15] S. Birchfield, "Elliptical head tracking using intensity gradients and color histograms," in *Proc. IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp. 232–237, Santa Barbara, Calif, USA, June 1998.

[16] M. Brandstein and H. Silverman, "A robust method for speech signal time-delay estimation in reverberant room," in *Proc. IEEE Int. Conf. Acoustics, Speech, Signal Processing*, Atlanta, Ga, USA, May 1996.

[17] R. Duraiswami, D. Zotkin, and L. Davis, "Active speech source localization by a dual coarse-to-fine search," in *Proc. IEEE Int. Conf. Acoustics, Speech, Signal Processing*, Salt Lake City, Utah, USA, May 2001.

[18] D. Zotkin, R. Duraiswami, L. Davis, and I. Haritaoglu, "An audio-video front end for multimedia applications," in *Proc. IEEE Int. Conf. on Systems, Man, and Cybernetics*, Nashville, Tenn, USA, October 2000.

[19] M. Swain and D. Ballard, "Color indexing," *International Journal of Computer Vision*, vol. 7, no. 1, pp. 11–32, 1991.

Dmitry N. Zotkin was born in Moscow, Russia in 1973. He received combined B.S./M.S. degree in applied mathematics and physics from the Moscow Institute of Physics and Technology, Dolgoprudny, Moscow, Russia, in 1996, and M.S. and Ph.D. degrees in Computer Science from University of Maryland at College Park, USA, in 1999 and 2002, respectively. He is currently a postdoctoral student at the University of Maryland. His current research interests are in multimodal signal processing for tracking, multimedia and virtual reality, and in particular in algorithms and methods for information presentation using customizable virtual auditory displays and perceptual processing interfaces.

Ramani Duraiswami was born in Madras, India in 1963. He received the B.Tech degree from Indian Institute of Technology, Bombay in 1985 and the Ph.D. degree from The Johns Hopkins University in 1991. Dr. Duraiswami is a research scientist at the Institute for Advanced Computer Studies, University of Maryland. He is also director of the Perceptual Interfaces and Reality Laboratory at the Institute, where multidisciplinary research in perceptual interfaces and virtual reality is conducted. He has broad research interests in the areas of virtual reality, computer vision, scientific computing, modeling human audition, computational acoustics, applied mathematics, and fluid dynamics.

Larry S. Davis received his B.A. from Colgate University in 1970 and his M.S. and Ph.D. in Computer Science from the University of Maryland in 1974 and 1976, respectively. From 1977–1981 he was an Assistant Professor in the Department of Computer Science at the University of Texas, Austin. He returned to the University of Maryland as an Associate Professor in 1981. From 1985–1994 he was the Director of the University of Maryland Institute for Advanced Computer Studies. He is currently a Professor in the Institute and the Computer Science Department, as well as Chair of the Computer Science Department. He was named a Fellow of the IEEE in 1997. Prof. Davis is known for his research in computer vision and high performance computing. He has published over 75 papers in journals and has supervised over 12 Ph.D. students. He is an Associate Editor of the International Journal of Computer Vision and an area editor for Computer Models for Image Processor: Image Understanding. He has served as program or general chair for most of the field's major conferences and workshops, including the 5th International Conference on Computer Vision, the field's leading international conference.