Chapter 7

Multimodal Tracking for Smart Videoconferencing and Video Surveillance

Dmitry N. Zotkin, Vikas C. Raykar*, Ramani Duraiswami*, and Larry S. Davis*

Perceptual Interfaces and Reality Lab,
Institute for Advanced Computer Studies (UMIACS),
University of Maryland, College Park, MD 20742 USA
**Also at: Department of Computer Science,*
University of Maryland, College Park MD 20742 USA

Abstract

Many applications such as interactive multimedia, videoconferencing, and surveillance require the ability to track the 3-D motion of the subjects. Particle filters represent an attractive solution for the tracking problem because they do not require solution of the inverse problem of obtaining the state from the measurements and because the tracking can naturally integrate multiple modalities. We build a framework for multi-modal tracking using multiple cameras and multiple microphone arrays. In order to calibrate the resulting distributed multi-sensor system, we propose a method to automatically determine the 3-D positions of all microphones in the system using at least five loudspeakers. Our method does not require knowledge of the loudspeaker positions but assumes that for each loudspeaker there exists a microphone very close to it. We derive the Maximum Likelihood estimator, which reduces to the solution of the non-linear least squares problem. A closed-form approximate solution that can be used as an initial guess is derived. We also derive an approximate expression for the estimator covariance using the implicit function theorem and Taylor series expansion. Using the estimator covariance matrix, we analyze the performance of the estimator with respect to the positions of the loudspeakers; in particular, we show that the loudspeakers should be as far away from each other, and the microphones should lie within the convex hull formed by the loudspeakers. We verify the correctness and robustness of the multimodal tracker and of the self-calibration algorithm both with Monte-Carlo simulations and on real data from three experimental setups. We also present practical details of system implementation.

7.1. INTRODUCTION

Our perception of the environment is a strong function of our relative location to objects being perceived. It changes with our orientation, body posture, whether we are seated or standing, etc., and it becomes apparent that multiple modalities are merged in the brain to produce space-time perception [1]. Many applications thus require an ability to track the motion of a person; for example, in surveillance it is necessary to focus the system's attention on the action taking place in the system's field of view; virtual and augmented reality (VR/AR) applications that seek to create convincing experiences need to adapt the presentation according to the person's position and orientation; and applications that seek to provide telepresence (such as videoconferencing) need to determine the spatial distribution of people in an environment. There are multiple modalities that are used to acquire such positional information, including multi-perspective video, active audio transmitters placed on moving objects, speech based audio, magnetic tracking of installed tags, etc. Integrating information obtained from multiple sensors can lead to improvements in both the accuracy and the sampling rate and to a practical design for the development of a surveillance, augmented reality, or smart videoconferencing system. Indeed, the development of multimodal sensor fusion algorithms has seen many applications recently, including multisensor vehicle navigation system where computer vision, sonar, and laser and microwave radar sensors are used together [2]; audio-visual person identification using support vector machine classifier [3]; multimodal speaker detection using Bayesian networks [4]; multimodal tracking using inverse modeling techniques from computer vision, speech recognition, and acoustics [5]; and discourse segmentation using gesture, speech, and gaze cues [6].

One of the problems faced by the developers of the multi-sensor systems is the calibration of sensor's positions. Automatic camera calibration algorithms are relatively well-developed ([7–11], to cite a few); therefore, in this chapter we will focus on presenting a novel algorithm for autocalibration of the audio subsystem of the tracker. Most multi-microphone array processing algorithms need to know the positions of the microphones very precisely (although some algorithms, e.g., blind beamforming, do not need to know the microphone positions); for example, in the case of

source localization, even relatively small uncertainties in sensor location could make substantial, often dominant, contributions to overall localization error [12]. Manual calibration employing a tape or laser range device is error-prone and has to be repeated every time the geometry of the system changes either because of ad-hoc redeployment or accidentally. Thus, automatic position calibration of multiple microphones is essential and has many applications such as sound source localization, audio tracking, hands free voice communication, beamforming, speech enhancement, and speech recognition, where distributed arrays of multiple microphones are widely used.

We propose a method to automatically determine the three dimensional positions of multiple microphones. The approach we follow is to use a few loudspeakers, to play a known calibration signals from each of them, and to measure the time taken by the sound to reach the microphones and thus determine the distances between each loudspeaker and all the microphones. From these distances, the Maximum Likelihood (ML) estimate of the microphone positions can be derived. This approach was taken before by several authors, who assumed known loudspeaker positions (e.g., [13] describes an experimental setup for automatic calibration of a large-aperture microphone array using acoustic signals from transducers whose locations are known). However, we wish to consider the more complex case when the loudspeaker positions are unknown as well and must also be estimated from measurements (related theoretical work can be found in [12, 14, 15]). The ML estimate in this case turns out to be a non-linear optimization problem, which needs a very good initial guess for solution. To obtain it, we derive a novel closed-form solution for the microphones and loudspeakers positions under the assumption that each loudspeaker has a microphone which is very close (i.e., attached) to it. In practice we achieve this by placing a microphone right next to the loudspeaker. (This scenario is also realized practically in the case of mobile phones, handheld computers, and laptops.) The closed-form solution is further refined by the nonlinear minimization procedure. In order to study the ML estimator, we derive the approximate mean and covariance of the implicitly defined estimator using the implicit function theorem and Taylor series expansion and analyze the estimator accuracy with respect to the loudspeaker positions. In particular, we show that the loudspeakers should be placed as far away from each other and all the microphones should be in the convex hull formed by the loudspeakers as opposed to the setup in [13] where all

the loudspeakers are close to each other.

We then propose a multimodal information fusion algorithm for audio and video measurements obtained from multiple calibrated cameras and microphone arrays using sequential Monte-Carlo methods (also known as particle filters [16]). 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 ability to track self-configuration parameters of a changing system (e.g., when a sensor is located on a mobile platform and is moving itself) together and in the same framework as object tracking. This is done by including those parameters into the state vector of the system. We describe a particular setup in which experimental results are obtained, which is an extension of the videoconferencing setup in [17], analyze tracking performance using synthetic data, and show results of several successful tracking experiments in different environments, including a person moving with an ultrasonic sound source in an anechoic room, for an echolocating bat in flight in an anechoic room, and for a speaker moving in a typical office room. We also show that our algorithm is capable of sensor self-motion recovery together with object tracking and of successful handling of temporary absences of some measurements (the target being occluded from one or both cameras, or absence of audio data).

The rest of this chapter is organized as follows. Section 7.2 is devoted to theoretical derivation of the framework for automatic microphone array calibration. We formulate the problem and the Maximum Likelihood (ML) estimator for the microphone and loudspeaker positions and then derive a closed-form approximate solution that can be used as a initial guess for the nonlinear minimization routine. We also derive the theoretical mean and covariance of the estimated parameters, validate those using Monte-Carlo simulations, and analyze the dependency of the estimator covariance on the location of speaker-microphone pairs. In Section 7.3, a discussion of the issues involved in designing a practical system is given and the possible source of error are outlined. Section 7.4 provides a brief introduction to particle filters. Section 7.5 describes the particulars of the multimodal setup used for the experiments as well as mapping of audio and video measurements into the particle filter framework. In Section 7.6, we show the experimental evaluation of the multimodal tracker using synthetic and real input data. Finally, section 7.7 concludes the chapter with

a summary of the work presented.

7.2. AUTOMATIC CALIBRATION OF MULTI-MICROPHONE SETUP

In this section, we present a systematic analysis of the problem of self-calibrating a microphone array of unknown geometry using at least five loudspeakers. We have used the technique to calibrate the microphone array, which was used later in particle filter tracking experiments. However, the method is obviously not restricted to such scenarios and can be used in any application, such as smart conference rooms [18], hands-free voice communication [19], speech enhancement [20], acoustic surveillance [21], and many others. In related work [22], we present an alternative, somewhat more flexible method, which can recover more complex microphone and loudspeaker configurations and can handle the case of no common timebase among all stations (a station is defined as a unit that includes at least one microphone and at least one loudspeaker, such as a mobile phone). However, it is more expensive and requires more measurements than the method described here.

7.2.1 ML estimator

Given a set of M microphones and S loudspeakers in unknown locations, our goal is to estimate their three dimensional coordinates. Each loudspeaker is excited using a known calibration signal (such as maximum length sequence or chirp), and the signal is captured by each of the microphones. The Time of Flight (TOF) is estimated from the captured audio signal. The TOF for a given microphone and speaker pair is defined as the time taken by the acoustic signal to travel from the speaker to the microphone. We assume that the signals emitted from each of the speakers do not interfere with each other (i.e., each signal can be associated with a particular speaker). This can be achieved by confining the signal at each speaker to disjoint frequency bands or time intervals. Alternately, we can use coded sequences so that the signal due to each speaker can be extracted at the microphones and correctly attributed to the corresponding speaker. The $M \times S$ TOF

measurements constitute our observations, based on which we have to estimate the microphone and speaker positions.

7.2.1.1 Formulation

Let $\mathbf{m}_i = [mx_i : my_i : mz_i]^T$ and $\mathbf{s}_j = [sx_i : sy_j : sz_j]^T$ be the three dimensional vectors representing the x, y, and z coordinates of the i^{th} microphone and j^{th} loudspeaker respectively. We excite each of the S speakers one at a time and measure the TOF at each of the M microphones. The TOF_{ij} for the i^{th} microphone and the j^{th} speaker is defined as the time taken for the acoustic signal to travel from the j^{th} speaker to the i^{th} microphone. Let $TOF_{ij}^{estimated}$ and TOF_{ij}^{actual} be the estimated and the actual TOF respectively for the i^{th} microphone and j^{th} speaker. The actual TOF can be written as

$$TOF_{ij}^{actual} = \frac{\parallel \mathbf{m}_i - \mathbf{s}_j \parallel}{c}, \tag{7.1}$$

where $\|\|$ is the Euclidean norm and c is the speed of the sound.

Assuming a Gaussian noise model for our observations, we can derive the Maximum Likelihood (ML) estimator as follows. Let Θ be a vector of length $P \times 1$ representing all the unknown non-random parameters to be estimated (microphone and speaker coordinates), Γ a vector of length $N \times 1$ representing noisy estimated TOF measurements, and $T(\Theta)$ a vector of length $N \times 1$ representing the actual value of the TOF observations. Then our model for the observations is $\Gamma = T(\Theta) + \eta,\ \eta = N(0, \Sigma)$ (i.e., η is the $N \times 1$ Gaussian noise vector with zero mean and covariance matrix Σ). The likelihood function of Γ in vector form can be written as

$$p(\Gamma|\Theta) = (2\pi)^{-\frac{N}{2}} |\Sigma|^{-\frac{1}{2}} \exp\left(-\frac{1}{2}[\Gamma - T(\Theta)]^T \Sigma^{-1} [\Gamma - T(\Theta)]\right). \tag{7.2}$$

The ML estimate $\hat{\Theta}_{ML}$ of Θ is defined as

$$\hat{\Theta}_{ML}(\Gamma) = \arg\{\max_{\Theta} p(\Gamma|\Theta)\}. \tag{7.3}$$

Maximizing $p(\Gamma|\Theta)$ is equivalent to maximizing $\log p(\Gamma|\Theta)$. The log-likelihood function is given by

$$\log p(\Gamma|\Theta) = -\frac{N}{2}\log(2\pi) - \frac{1}{2}\log|\Sigma| - \frac{1}{2}[\Gamma - T(\Theta)]^T \Sigma^{-1} [\Gamma - T(\Theta)]. \tag{7.4}$$

The ML estimator can be written as

$$\hat{\Theta}_{ML}(\Gamma) = \arg\{\max_{\Theta} F(\Theta, \Gamma)\},$$
$$F(\Theta, \Gamma) = -\frac{1}{2}[\Gamma - T(\Theta)]^T \Sigma^{-1}[\Gamma - T(\Theta)]. \tag{7.5}$$

Assuming that each of the TOFs are independently corrupted by zero-mean additive white Gaussian noise of variance σ_{ij}^2, the ML estimate can also be formulated as a nonlinear least squares problem (in which case Σ becomes a diagonal matrix):

$$\hat{\Theta}_{ML} = \arg_{\Theta} \min \sum_{i=1}^{M} \sum_{j=1}^{S} \frac{(TOF_{ij}^{estimated} - TOF_{ij}^{actual})^2}{\sigma_{ij}^2}. \tag{7.6}$$

We estimate the TOF using Generalized Cross Correlation (GCC) [23]. The estimated TOF is corrupted due to ambient noise and room reverberation. For high SNR, the delays estimated by the GCC can be shown to be normally distributed with zero mean [23].

7.2.1.2 Reference coordinate system

As the solution depends only on pairwise TOFs, any translation, rotation, and reflection of the global minimum found will also be a global minimum. In order to make the solution invariant to rotation and translation, we select three arbitrary nodes to lie in a plane such that the first is at $(0, 0, 0)$, the second at $(x_1, 0, 0)$, and the third at $(x_2, y_2, 0)$. To eliminate the ambiguity due to reflection along the Z-axis, we specify one more node to lie in the positive-Z half-space. Also the reflections along the X and Y axes can be eliminated by assuming that $x_1 > 0$ and $y_2 > 0$. In practice, microphones belonging to the speaker-microphone pairs are used as the reference nodes. We prefer microphones as reference points because they are usually smaller than loudspeakers and hence the reference coordinate system can be known more precisely. This would be beneficial if we want to subsequently transform this reference coordinate system to the room coordinate system.

7.2.1.3 Non-linear least squares

The ML estimate for the node coordinates of the microphones and loudspeakers is implicitly defined as the minimum of the non-linear function given in Equation 7.5. This function has to be minimized using numerical optimization methods. The Levenberg-Marquardt method [24] is a popular method for solving non-linear least squares problems. It is a compromise between steepest descent and Newton's methods. The steepest descent method potentially has a very slow convergence but can converge from any starting point. Newton's method converges fast but requires a good initial guess and computation of the Hessian matrix inverse. (For more details on nonlinear minimization please refer to [24]). Appendix I gives the non-zero partial derivatives needed for the minimization routines (the actual routines are available in many software products such as MATLAB and Numerical Recipes). A significant problem is that the Levenberg-Marquardt minimization routine will not converge to the global minimum unless we have a very good initial guess. In the next section, we derive an approximate closed-form solution, which can be used as a initial guess for the minimization routine.

Also, the total number of observations should be greater than or equal to the total number of parameters to be estimated. In our case $MS \geq 3(M+S) - 6$. If $M = S = K$ then $K \geq 5$. Thus we need a minimum of five speaker-microphone pairs. We will use this result later to get a closed-form solution for the microphone coordinates.

7.2.2 Closed-form solution

[Figure 7.1]

Given the pairwise Euclidean distances between N nodes, their relative positions can be determined by means of metric or classical Multidimensional Scaling (MDS) [25]. MDS is a popular technique in psychology and denotes a set of data-analysis techniques for the analysis of proximity data on a set of stimuli for revealing the hidden structure underlying the data [26]. The proximity data refers to some measure of pairwise dissimilarity. Given a set of N stimuli along

with their pairwise dissimilarities p_{ij}, MDS arranges them as points in a multidimensional space in a way that the distances between any two points are a monotonic function of the corresponding dissimilarity. MDS is widely used to visually study the structure of proximity data. If proximity data are based on the Euclidean distances, then classical metric MDS [25] can exactly recreate the configuration.

7.2.2.1 Classical MDS

Given a set of N points in 3-D space, let X be a $N \times 3$ matrix where each row represents the 3-D coordinates of each point. Then the $N \times N$ matrix $B = XX^T$ is called the dot product matrix. By definition, B is a symmetric positive definite matrix, so the rank of B (i.e., the number of positive eigenvalues) is equal to the dimensionality of space in which the data points lie (i.e., 3 in this case). Starting with a matrix B (possibly corrupted by noise), it is possible to factor it to get the matrix of coordinates X. One method to factor B is to use singular value decomposition (SVD) [27], that is, decompose $B = U\Sigma U^T$ where Σ is a $N \times N$ diagonal matrix of singular values. The diagonal elements are arranged as $s_1 \geq s_2 \geq s_r > s_{r+1} = ... = s_N = 0$, where r is the rank of the matrix B. The columns of U are the corresponding singular vectors. We can write $X^{'} = U\Sigma^{1/2}$ and take the first three columns of $X^{'}$ to get X. If the elements of B are not corrupted by noise, then all the other columns are zero. It can be shown that SVD factorization minimizes the matrix norm $\parallel B - XX^T \parallel$.

In practice, we can estimate the distance matrix D, where the ij^{th} element is the Euclidean distance between the i^{th} and the j^{th} point. This distance matrix D must be converted into a dot product matrix B before MDS can be applied. We need to choose some point as the origin of our coordinate system in order to form the dot product matrix. Any point can be selected as the origin, but Togerson [25] recommends the choice of the centroid of all the points. If the distances have random errors, then such choice minimizes the errors, as they would tend to cancel each other. We can convert the distance matrix into a dot product matrix using simple geometry; please refer to Appendix II for a derivation.

In our case of M microphones and S speakers, we cannot use MDS directly as we cannot measure some of the pairwise distances (e.g., the distance between two microphones). Figure 7.1 shows an example consisting of 7 microphones and 4 speakers, with each speaker attached to one of the microphones forming in effect 4 speaker-microphone pairs and 3 single microphones. The cells marked 'X' and '?' show available and unavailable measurements, respectively.

7.2.2.2 Forming speaker-microphone pairs

In our practical setup, for every loudspeaker there is a microphone attached to it. Such speaker-microphone pairs are considered as one entity (i.e., we assume that the distance between them is zero). Based on this approximation, the distance d_{ij} between the i^{th} and j^{th} speaker-microphone pair is given by

$$\begin{aligned} d_{ij} &\approx 0 \ : if: \ i = j, \\ d_{ij} &\approx \frac{c(TOF_{ij} + TOF_{ji})}{2} : \ if \ : i \neq j, \end{aligned} \tag{7.7}$$

where c is the speed of the sound. Once all the pairwise distances are obtained, classical MDS is performed to get the approximate positions of the speaker-microphone pairs. The position estimate from MDS is with respect to an arbitrary centroid and orientation and hence it is converted into the reference coordinate system as described in Section 7.2.1.2.

The approximate locations of the speaker-microphone pairs are slightly perturbed to obtain the initial guess for the microphone and speaker locations. We use this as an initial guess for the nonlinear minimization routine and obtain the exact locations of the microphones and loudspeakers in each speaker-microphone pair. As discussed before, for the ML estimation procedure we need a minimum of five speaker-microphone pairs.

7.2.2.3 Closed-form solution for microphone positions

From the previous step we have obtained the locations of five speaker-microphone pairs. If the location of four speakers are known, then by triangulation the positions of remaining (single)

microphones can be determined analytically. (Given its distance from one loudspeaker, the microphone can lie anywhere on a sphere centered at that loudspeaker. With two loudspeakers, the unknown microphone can lie on a circle, as two spheres intersect at a circle. With three, the set of solutions is reduced to two points, and with four loudspeakers a unique location is determined). As the estimated distances are corrupted by noise and further five (instead of four) distances are available, the intersection in general need not to be a unique point. Hence, we solve the problem in a least square sense.

As before, assume that we have S loudspeakers. Let $\mathbf{s}_j = [sx_j : sy_j : sz_j]^T$ be the x, y and z coordinates of the j^{th} speaker. The locations of the loudspeakers are determined as discussed in the previous sections. Let $\mathbf{m}_i = [mx_i : my_i : mz_i]^T$ be the unknown microphone coordinates, which we have to determine. For the i^{th} microphone, we have S TOF measurements

$$c^2TOF_{ij}^2 = \| \mathbf{m}_i - \mathbf{s}_j \|^2 :: j = 1 \ldots S. \tag{7.8}$$

In order to write a closed-form solution for m_i, we take the difference of every pair of equations:

$$\| \mathbf{m}_i - \mathbf{s}_j \|^2 - \| \mathbf{m}_i - \mathbf{s}_k \|^2 = c^2TOF_{ij}^2 - c^2TOF_{ik}^2. \tag{7.9}$$

Expanding, we can write,

$$mx_i(sx_k - sx_j) + my_i(sy_k - sy_j) + mz_i(sz_k - sz_j) = \frac{c^2TOF_{ij}^2 - c^2TOF_{ik}^2 - \| \mathbf{s}_j \|^2 + \| \mathbf{s}_k \|^2}{2}, \tag{7.10}$$

or simply

$$(\mathbf{s}_k - \mathbf{s}_j)^T \mathbf{m}_i = b_{jk}^i, \tag{7.11}$$

where

$$b_{jk}^i = \frac{c^2TOF_{ij}^2 - c^2TOF_{ik}^2 - \| \mathbf{s}_j \|^2 + \| \mathbf{s}_k \|^2}{2}. \tag{7.12}$$

Each pair of speakers generates one equation in 3 unknowns; for S speakers there are $S(S-1)/2$ equations and we need a minimum of 4 speakers to determine the position of one microphone. For

$S > 4$ speakers we define the following matrix A and the vector $\mathbf{b}^i$

$$A = \begin{bmatrix} (\mathbf{s}_1 - \mathbf{s}_2)^T \\ (\mathbf{s}_1 - \mathbf{s}_3)^T \\ \vdots \\ (\mathbf{s}_1 - \mathbf{s}_S)^T \\ (\mathbf{s}_2 - \mathbf{s}_3)^T \\ \vdots \\ (\mathbf{s}_k - \mathbf{s}_j)^T \\ \vdots \\ (\mathbf{s}_{S-1} - \mathbf{s}_S)^T \end{bmatrix} \quad \mathbf{b} = \begin{bmatrix} b^i_{21} \\ b^i_{31} \\ \vdots \\ b^i_{S1} \\ b^i_{32} \\ \vdots \\ b^i_{jk} \\ \vdots \\ b^i_{S(S-1)} \end{bmatrix} \quad A\mathbf{m_i} = \mathbf{b}^i. \tag{7.13}$$

The least squares solution can be written as

$$\mathbf{m_i} = (A^T A)^{-1} A^T \mathbf{b}^i. \tag{7.14}$$

The closed-form solution for the microphone coordinates is further refined via a final ML estimation of all the unknown parameters (i.e., the positions of speaker-microphone pairs and of single microphones).

The following table summarizes the complete algorithm.

ALGORITHM

Assume that we have $M + S$ microphones and $S \geq 5$ loudspeakers. Attach one microphone to each of the loudspeakers so that we have S speaker-microphone pairs and M microphones. Place the speakers such that the microphones are in the convex hull formed by the speakers.

- **STEP 0** : *Select three loudspeakers to form a reference coordinate system: the first as the origin, the second to define the positive X-axis, and the third to form the positive XY-plane. Also select a fourth one to define the positive-Z half-space. The three reference speakers should be chosen such that they are as far away as possible from each other.*

- **STEP 1**: *Measure the $(M+S)\times S\,TOF$ matrix by exciting each of the loudspeakers using an appropriate signal.*

- **STEP 2**:

  a) *Form the approximate distance matrix D between the S speaker-microphone pairs using equation (7.7).*

  b) *Convert the distance matrix D to the dot product matrix B (Appendix II).*

  c) *Get the approximate positions of the speaker-microphone pairs using metric MDS.*

  d) *Slightly perturb the coordinates to get approximate separate initial guesses for the microphones and speakers positions.*

  e) *Minimize the TOF-based error function using the Levenberg-Marquardt method to get the final positions of the S loudspeakers forming the reference coordinate system.*

  f) *Translate, rotate, and mirror the coordinates to the coordinate system specified in STEP 0.*

- **STEP 3**: *Get the closed-form solution for the M microphones using the reference coordinate system formed using equation (7.14).*

- **STEP 4**: *Refine all the values by performing the final ML estimation using the Levenberg-Marquardt method.*

[Figure 7.2]

Figure 7.2 shows an example in two dimensions with 10 microphones (shown as 'x') and 3 speaker-microphone pairs. (In two dimensions we select two nodes to lie on a line, the first at $(0,0)$ and the second at $(x_1,0)$, and specify one more node to lie in the positive-Y area to eliminate the reflection ambiguity). Using MDS, we obtain the approximate locations of the three speaker-microphone pairs, shown as filled squares in the figure. This approximate position is refined using ML estimation procedure to obtain the actual (separate) locations of the microphone and

the loudspeaker in each speaker-microphone pair (no longer assuming that the distance between those is zero). Using the obtained loudspeaker locations, we compute a closed-form solution for the microphone locations, shown as squares in the figure. In the final ML estimation, we refine the closed-form solution to obtain the exact location of the microphones (shown as circles).

7.2.3 Estimator bias and variance

The properties of the ML estimator can be studied in terms of its bias and error variance. The error variance depends on the noise standard deviation, on the microphone array geometry, and on the positions of the reference speaker-microphone pairs. One way to study it is to perform extensive Monte-Carlo simulations for various geometries and positions of the reference speaker-microphone pairs. However, if we could construct an analytical expression for the estimator bias and variance, then such simulation studies can be carried out much more quickly.

The ML estimate for the microphone and the speaker positions is defined implicitly as the minimum of a certain error function (equation 7.5). Hence it is not possible to obtain exact analytical expressions for its mean and variance. However, by using the implicit function theorem and Taylor series expansion, it is possible to derive approximate expressions for those, similarly to derivations in [28–30]. Also, it is possible to derive the Cramér-Rao lower bound on the error covariance matrix of any unbiased estimator [31]; however, we cannot determine whether our estimator is unbiased.

7.2.3.1 Estimator covariance matrix

The ML estimate $\hat{\Theta}$ of Θ is the one that maximizes the likelihood ratio. In our case, the ML estimator is implicitly defined by equation (7.5). The maximum can be found by setting the first derivative to zero:

$$\nabla_{\Theta} F(\Theta, \Gamma) \mid_{\Theta=\hat{\Theta}} = \mathbf{0}, \tag{7.15}$$

where $\bigtriangledown_\Theta$ is a $P \times 1$ column gradient operator defined as

$$\nabla_\Theta F(\Theta, \Gamma) = [\frac{\partial F(\Theta, \Gamma)}{\partial \theta_1}, \frac{\partial F(\Theta, \Gamma)}{\partial \theta_2}, ..., \frac{\partial F(\Theta, \Gamma)}{\partial \theta_P}]^T. \tag{7.16}$$

As discussed earlier, the function can have multiple global minima. However by defining a reference coordinate system and removing the assumed parameters we can make sure that the function has only one global minimum. The implicit function theorem guarantees that equation (7.15) implicitly defines a vector-valued function

$$\hat{\Theta} = h(\Gamma) = [h_1(\Gamma), h_1(\Gamma), ..., h_P(\Gamma)]^T \tag{7.17}$$

that maps the observation vector Γ to the parameter vector $\hat{\Theta}$. As such, equation (7.15) can be written as

$$\nabla_\Theta F(h(\Gamma), \Gamma) = \mathbf{0}. \tag{7.18}$$

It is not possible to find an analytical expression for $h(\Gamma)$; however, we can approximate the covariance using the first-order Taylor series expansion for $h(\Gamma)$. Let $\bar{\Gamma}$ be the mean of Γ. Expanding $h(\Gamma)$ around $\bar{\Gamma}$ gives

$$h(\Gamma) \approx h(\bar{\Gamma}) + [\nabla_\Gamma h(\Gamma)^T \mid_{\Gamma=\bar{\Gamma}}]^T (\Gamma - \bar{\Gamma}), \tag{7.19}$$

where

$$\nabla_\Gamma = [\frac{\partial}{\partial \gamma_1}, \frac{\partial}{\partial \gamma_2}, \ldots, \frac{\partial}{\partial \gamma_N}]^T \tag{7.20}$$

is a $N \times 1$ column gradient operator. Taking the covariance on both sides of equation (7.19), we obtain

$$Cov[h(\Gamma)] \approx [\nabla_\Gamma h(\Gamma)^T \mid_{\Gamma=\bar{\Gamma}}]^T Cov(\Gamma) [\nabla_\Gamma h(\Gamma)^T \mid_{\Gamma=\bar{\Gamma}}]. \tag{7.21}$$

Note we do not know $h(\Gamma)$, but the dependence is only through the first-order partial derivatives of $h(\Gamma)$. Differentiating equation (7.18) with respect to Γ and evaluating it at $\bar{\Gamma}$ yields

$$\nabla_\Theta \nabla_\Theta F(h(\bar{\Gamma}), \bar{\Gamma}) [\nabla_\Gamma h(\bar{\Gamma})^T]^T + \nabla_\Theta \nabla_\Gamma F(h(\bar{\Gamma}), \bar{\Gamma}) = \mathbf{0}. \tag{7.22}$$

Assuming that $\nabla_\Theta \nabla_\Theta F(h(\bar{\Gamma}), \bar{\Gamma})$ is invertible, we can write

$$[\nabla_\Gamma h(\bar{\Gamma})^T]^T = - [\nabla_\Theta \nabla_\Theta F(h(\bar{\Gamma}), \bar{\Gamma})]^{-1} \nabla_\Theta \nabla_\Gamma F(h(\bar{\Gamma}), \bar{\Gamma}). \tag{7.23}$$

At $\Gamma = \bar{\Gamma}$, the vector derivatives involved can be shown to be

$$\begin{aligned}
\nabla_\Theta \nabla_\Theta F(\Theta, \Gamma) &= -J^T \Sigma^{-1} J, \\
\nabla_\Theta \nabla_\Gamma F(\Theta, \Gamma) &= J^T \Sigma^{-1}, \\
\nabla_\Gamma \nabla_\Theta F(\Theta, \Gamma) &= \Sigma^{-1} J, \\
\nabla_\Gamma \nabla_\Gamma F(\Theta, \Gamma) &= -\Sigma^{-1},
\end{aligned} \tag{7.24}$$

where J is a $N \times P$ Jacobian matrix of partial derivatives of $T(\Theta)$,

$$[J]_{ij} = \frac{\partial T_i(\Theta)}{\partial \Theta_j}. \tag{7.25}$$

Please refer to Appendix I for full derivations. Substituting the vector derivatives, we obtain

$$[\nabla_\Gamma h(\bar{\Gamma})^T]^T = -[-J^T \Sigma^{-1} J]^{-1} J^T \Sigma^{-1}. \tag{7.26}$$

Substituting into the covariance equation (7.21) we finally arrive at the following:

$$Cov\hat{\Theta} \approx Cov[h(\Gamma)] \approx [J^T \Sigma^{-1} J]^{-1}. \tag{7.27}$$

If we assume that all the observation have the same variance σ^2 (i.e., $\Sigma = \sigma^2 I$), we get

$$Cov\hat{\Theta} = \sigma^2 [J^T J]^{-1} = \sigma^2 F^{-1}, \tag{7.28}$$

where $F = J^T J$. If we assume that all the microphone and source locations are unknown, F is rank deficient and hence not invertible. This happens because the solution to the ML estimation problem as formulated is not invariant to rotation and translation. In order to make F invertible, we remove the rows and columns corresponding to the known parameters.

7.2.3.2 Estimator mean

Taking the expectation of the first order Taylor series expansion in equation (7.19), we get

$$E(h(\Gamma)) \approx h(\bar{\Gamma}) = h(T(\Theta)). \tag{7.29}$$

We see that the mean is the value given by the estimation procedure when applied to the actual noise free measurements $T(\Theta)$. A more accurate expression for the mean can be derived using the second order Taylor series expansion. However, it involves third-order vector derivatives and generally cannot be stated in a simple form comparable to equation (7.27).

7.2.3.3 Monte-Carlo simulations

In order to validate the derived expression for the estimator variance, we performed a series of Monte-Carlo simulations with 20 microphones randomly placed in a room of dimensions $4.0m \times 4.0m \times 4.0m$. We placed five speaker-microphone pairs so that all 20 microphones were within the convex hull formed by those pairs. Based on the geometry of the setup, the actual TOF values were calculated and then corrupted with additive white Gaussian noise (mean zero, variance σ^2) in order to model the room ambient noise and reverberation. The Levenberg-Marquardt method was used as the minimization routine. The results were averaged over 200 trials for each noise variance value. Figure 7.4(a) and Figure 7.4(b) show the total estimator variance (sum of estimate variances of each parameter) and the total estimator bias (sum of estimate biases for each parameter) of all unknown microphone coordinates plotted against the noise standard deviation σ. The theoretical estimator variance is also shown. From Figure 7.4(a) we can see that experimentally computed estimator variance closely tracks the theoretical one. Figure 7.4(b) shows that the estimator is unbiased for low noise variance; however, as the noise variance increases, the estimator starts showing an increasing bias.

7.2.3.4 Implications for placement of loudspeakers

In our evaluation of the covariance matrix, we have assumed that we know the positions of some nodes (i.e., we have fixed three loudspeakers to reside in the $z = 0$ plane). The covariance matrix has significant dependence on how those known nodes are arranged. Figure 7.3 shows the 95% uncertainty ellipses for a regular 2-D array containing 25 microphones and 4 loudspeakers

for different positions of the loudspeakers. The microphones are represented as dots (.) and the loudspeakers as crosses ($\times$). The position of one loudspeaker and the x coordinate of another one are assumed to be known (shown in bold). For the second fixed loudspeaker, only the variance in the y direction is shown (as its x coordinate is fixed). For TOF estimation, the noise variance was assumed to be 10^{-9} in order to properly visualize the uncertainty ellipses.

[Figure 7.3]

[Figure 7.4]

In Figure 7.3(a), all four loudspeakers are placed at one corner of the grid. It can be seen that the farther the estimation is performed from the known nodes, the wider is the uncertainty ellipse. The uncertainty in the direction tangential to the line joining the microphone and the loudspeaker cluster is much larger than along the line. The same can be seen in Figure 7.3(b), where all loudspeakers are placed at the grid center. A simple geometric explanation can be provided; assume that we know the locations of two speakers (as shown in Figure 7.3(d)). Each circular band represents the uncertainty in the distance estimation. The intersection of two bands corresponding to two speakers gives the uncertainty region for the microphone position, which widens if bands are intersecting far away from the speakers because of the decrease in curvature. From this reasoning, we can deduce that to minimize the uncertainty ellipse area one should place loudspeakers as far away from each other as possible so that they "enclose" the area containing microphones, as in Figure 7.3(c), where substantially smaller uncertainty ellipses are seen. As such, in order to minimize the error due to Gaussian noise, we should choose the three reference nodes as far apart as possible.

7.3. SYSTEM AUTOCALIBRATION PERFORMANCE

In this section we discuss some of the practical issues of our autocalibration implementation, such as the type of calibration signal, the TOF estimation procedure, and other design choices.

7.3.1 Calibration Signals

In order to measure the TOF accurately, the calibration signal has to be selected and tuned properly for the particular setup. Maximum Length sequences and frequency sweeps (chirps) are the two most popular choices for the calibration signal. A linear chirp signal is a short pulse in which the signal frequency varies linearly between two preset frequencies. The cosine linear chirp signal of duration T with the instantaneous frequency varying linearly between f_0 and f_1 is given by

$$s(t) = A\cos\left(2\pi\left(f_0 + \frac{f_1 - f_0}{T}t\right)t\right), \quad 0 \le t \le T. \tag{7.30}$$

[Figure 7.5]

In our system, we used a chirp signal of 512 samples at 39.0625 kHz as our calibration signal. The instantaneous frequency varied linearly from 5 kHz to 10 kHz. The initial and final frequencies were chosen to lie in the common passband of the microphone and the speaker frequency responses. Convolution of the chirp signal sent by the loudspeaker with the room impulse response results in signal spreadout and distortion. Figure 7.5(a) shows the chirp signal as sent out by a loudspeaker. Figure 7.5(b) shows the signal recorded by the microphone attached directly to the same loudspeaker. Figure 7.5(c) is the same signal recorded by another microphone. Changes in the signal shape are due to the speaker, microphone, and room response.

7.3.2 Time Delay Estimation

[Figure 7.6]

TOF estimation is the most crucial part of the algorithm and a potential source of error as well. Hence a lot of care is required to obtain the TOF accurately in noisy and/or reverberant environments. The time delay between two signals can be found by locating the peak in the cross-correlation of those; however, this method is not robust to noise and reverberations. Knapp and Carter [23] developed a ML estimator for determining the time delay between signals received at two spatially separated sensors in the presence of uncorrelated noise. They introduce the Gener-

alized Cross Correlation (GCC) function, which is the cross-correlation of the filtered versions of the received signals; the delay is still estimated by locating a peak in the GCC. The GCC function $R_{x_1x_2}(\tau)$ is computed as

$$R_{x_1x_2}(\tau) = \int_{-\infty}^{\infty} W(\omega) X_1(\omega) X_2^*(\omega) e^{j\omega\tau} d\omega, \tag{7.31}$$

where $X_1(\omega)$ and $X_2(\omega)$ are the Fourier transforms of the microphone signals $x_1(t)$ and $x_2(t)$ respectively and $W(\omega)$ is the weighting function. Two most commonly using weighting functions are the ML and the Phase Transform (PHAT) weighting. The ML weighting function accentuates the signal passed to the correlator at frequencies for which the signal-to-noise ratio is the highest and simultaneously suppresses the noise power [23]. It performs well for low room reverberation levels. As the room reverberation increases, this method shows severe performance degradations [32]. Due to the spectral characteristics of the received signal being modified by the multipath propagation in a room, the GCC function can be made more robust by deemphasizing the frequency dependence. The PHAT weighting takes this idea to the extreme and flattens the magnitude spectrum. It is given by

$$W_{PHAT}(\omega) = \frac{1}{|X_1(\omega)X_2^*(\omega)|}. \tag{7.32}$$

As a result, the GCC peak corresponds to the dominant delay. The disadvantage of the PHAT weighting is that it equally emphasizes both the low and high SNR regions and hence works well only when the noise level is low. However, it is generally observed that in typical in-room microphone array scenarios reverberation is more detrimental to TOF determination than noise [32–35]; therefore, we use PHAT weighting for our experiments in real setup. Figure 7.6 shows the calibration signal, the received signal, and the corresponding GCC-PHAT function. The TOF is the position of the peak in the correlation function.

7.3.3 Speed of sound

Inaccuracy in knowledge of the speed of sound in the air can lead to errors in microphone position estimations. The speed of sound depends on the air temperature and is given by $c =$

$(331 + 0.6T)$ m/s, where T is the temperature in degrees Celsius. In practice, we assume that c is known and constant. However, we can also estimate the speed of the sound along with the positions of the microphones and loudspeakers [13].

7.3.4 Synchronization error

All the derivations in this chapter assume that all the microphones and loudspeakers are sharing a common clock source (e.g., are interfaced via a single or multiple but synchronized data acquisition boards). If that is not the case, TOFs will obviously contain errors due to lack of common time base. The method described in this chapter can be extended to handle such a case and recover time shifts between multiple sound production and acquisition units, but becomes significantly more complicated and requires more measurements; for full treatment, please refer to our related work in [22].

7.3.5 Testbed Setup and Results

[Figure 7.7]

We have set up a microphone array with 32 elements (Knowles Electronics model FG-3629). The array, shown in Figure 7.7(a), was built and is currently used for fast measurement of the head-related transfer function (HRTF) of human subjects [36]. A microphone is placed at each node of the structure. In order to calibrate this array, we placed five micro-loudspeakers (Knowles Electronics model ED-9689) next to five microphones of the array to form the speaker-microphone pairs. The first four speakers define the coordinate system, and all positions are computed with respect to that coordinate system. Figure 7.7(b) shows the results obtained by the proposed algorithm for our microphone array; asterisks indicate the speaker positions and circles indicate the microphone positions. In order to validate our results, we have measured the actual microphone coordinates with a Polhemus tracker and found that results are in good agreement.

7.4. THE TRACKING ALGORITHM

The particle filtering tracker, known also as a CONDENSATION tracker, was first introduced in the computer vision area in the work of Isard and Blake [16]. Various improvements of the technical nature were provided by Isard and Blake [37], Carpenter et al. [38], MacCormick and Blake [39], Li and Chellappa [40], Philomin et al. [41], and Qian and Chellappa [42]. The algorithm has seen application to multiple aspects of both computer vision and signal processing and was extended to track multiple objects, and a book on this topic [43] describes many different applications in signal detection and estimation. The mathematical framework of the tracker assumes that there exists a state vector X_s that describes the state of the tracked object and includes parameters of interest (coordinates, velocities, Euler angles, color histogram, etc.). There also exists a measurement vector (sometimes called the observation vector) X_m that consists of the measurement values obtained from the sensors, which are related to and carry some information about the underlying state of the object. The (unknown) true state vector for any given time corresponds to point in a *state space*. The *probability distribution function* is defined on the state space and represents the uncertainty in knowledge of the state vector. For example, the extended Kalman filter assumes that the distribution is Gaussian and thus implicitly creates a probability distribution function (PDF) on the state space by keeping its mean and variance, which is sufficient to define a Gaussian. In contrast, the CONDENSATION tracker maintains an explicit approximate PDF by computing the PDF value at a set of randomly selected sample points (called *particles*) in the state space, which allows it to work well when Kalman filtering fails due to the underlying PDF being non-Gaussian. A further practical advantage of the technique is that it allows one to mix modalities and measurements during the tracking relatively easily, as it is not necessary to construct inverse solutions explicitly. Such multiple-modality tracking is the focus of the current chapter.

7.4.1 Algorithm overview

The particle set update algorithm used in this paper is very similar to the original algorithm [16]. In the simple algorithm used in this paper, each particle $\{x_i\}$, $i = 1...N$, in the state space X has a weight π_i associated with it. This set of particles along with their weights is called *properly weighted* if it approximates the true PDF $P(x)$ correctly; this means that for every integrable function $H(x)$

$$E_x(H(x)P(x)) = \lim_{N\to\infty} \frac{\sum_N H(x_i)\pi_i}{\sum_N \pi_i}. \tag{7.33}$$

Given a properly weighted set of particles at time t with $\pi_i = \frac{1}{N}$ for every particle, it is possible to update the set reflecting the new measurements obtained at time $t + \delta t$ and end up, again, with a properly weighted set. The update algorithm is as follows:

1) Propagate each particle x_i in time using the object *motion model* which captures physical or any other knowledge about the relationship between the object's current state and that at subsequent time steps, obtaining particle set $\{x_i^*\}$.

2) Obtain a new measurement vector X_m and perform an evaluation of the *posterior probability density* π_i^* on set $\{x_i^*\}$ using the posterior probability estimation function $\pi_i^* = p(x_i^*|X_m)$, which essentially answers the question "how likely is it that the object state is x_i^*, given that the current measurement vector is X_m". In this way, the hidden internal state of the object is recovered from the observations. This probability cannot be computed in closed form as it would require one to invert the measurement equations, often resulting in a non-linear iterative process. Therefore, it is instead expanded using Bayes' rule:

$$p(x_i^*|X_m) = \frac{p(X_m|x_i^*)p(x_i^*)}{p(X_m)}, \tag{7.34}$$

in which $p(X_m)$ is the prior probability of measurement which is assumed to be known (e.g., assumed constant) and $p(x_i^*) = \frac{1}{N}$ is the weight of each particle at time t. Thus, $p(x_i^*|X_m) = Kp(X_m|x_i^*)$ for some constant K. Thus the expression is obtained for π_i^* using only $p(X_m|x_i^*)$, which *can* be computed without inversion of the measurement equations.

3) Resample from set $\{x_i^*\}$ with probabilities π_i^*, generating a new *properly weighted* set $\{x_i'\}$

with equal weights $\frac{1}{N}$ associated with every particle. The algorithm can be repeated now for the next time step and so on.

The algorithm does not keep an explicit representation of the currently most probable object state; however, it can be computed at any time by performing the numerical integration of a desired function of state variables over the state PDF approximately represented by the particle set.

7.4.2 Instantiation of the particle filter

The particle filter framework as described above is very general; in this section, we describe the state vector, the audio and the video measurement vectors, the motion model, and the posterior probability estimation functions used in this work. The state vector X_s for the system consists of the coordinates and the velocities of the tracked object. Thus, $X_s = [x\ y\ z\ \dot{x}\ \dot{y}\ \dot{z}]$. The motion model that is used to propagate object state in time is given by

$$\begin{aligned} x(t+\delta t) &= x(t) + \dot{x}(t)\delta t, \\ \dot{x}(t+\delta t) &= \dot{x}(t) + F\delta t. \end{aligned} \tag{7.35}$$

and similar expressions for $y, z, \dot{y}, \dot{z}$. F is the random excitation force applied to the particle. It generally depends on the expected range of object velocity and was set to an empirical value of 10 ms^{-2} for all test runs.

The observation vector X_m is built from audio and video measurements. The video part consists of the pairs $(\hat{u}_i, \hat{v}_i)$ of *image coordinates* of feature points on the tracked object for every camera in the system. Thus, N video cameras produce $2N$ components of the observation vector for each feature point. The transformation $(u_i, v_i) = \Psi_i(X_s) = \Psi_i(x, y, z)$ that converts the world coordinates into the image coordinates is pre-computed during a camera calibration procedure (described later). This transformation is used to "project" the state vector to the image coordinate (observation) space and compute the posterior probability of a state vector given the observation vector by measuring the distance between the projection of a state vector and the observed image coordinates. The audio part of the observation vector consists of the values of *time differences*

of arrivals (TDOA) $\hat{\tau}_{ij}$ of the acoustic source signal between different microphone pairs in the microphone array; for M microphones, the number of such observations is equal to C_2^M. The corresponding transformation from the state space to the observation space $\{\tau_{ij}\} = \Phi(X_s) = \Phi(x, y, z),\ i, j = 1...M$ is easy to compute and is also described later.

The final component of the tracker is the specific form of the posterior probability function, which is used to measure how likely is that the particular observation vector X_m at time t is caused by some candidate object state X_s. We define the video and audio error distances and the posterior probability as

$$\begin{aligned}
\epsilon_v^2(X_s, X_m) &= \frac{1}{N}\sum_{i=1}^{N}[(u_i - \hat{u}_i)^2 + (v_i - \hat{v}_i)^2] \\
(u_i, v_i) &= \Psi_i(X_s),\ (\hat{u}_i, \hat{v}_i) \in X_m \\
\epsilon_a^2(X_s, X_m) &= \frac{1}{C_2^M}\sum_{i,j}(\tau_{ij} - \hat{\tau}_{ij})^2,\ \tau_{ij} = \Phi(X_s),\ \hat{\tau}_{ij} \in X_m \\
p(X_m|X_s) &= \frac{\exp\left(-\frac{1}{2}\frac{\epsilon_v^2(X_s,X_m)}{\sigma_v^2}\right)}{\sqrt{2\pi\sigma_v}}\frac{\exp\left(-\frac{1}{2}\frac{\epsilon_a^2(X_s,X_m)}{\sigma_a^2}\right)}{\sqrt{2\pi\sigma_a}}
\end{aligned} \tag{7.36}$$

The parameters σ_v and σ_a in the PDF correspond to the standard deviation of the corresponding Gaussian. Loosely speaking, they play the same role as the variance of the measurement noise in the EKF and define how much trust is put on every individual measurement. Variation of these parameters affects the filter behavior. If the measurements are known to be inaccurate, larger values of σ should be used. On the other hand, if the value used is too large, the filter would be slow to learn the correct object motion.

Note that the $p(X_m|X_s)$ is a product of Gaussians formed from individual measurements. If some audio or video measurement is unavailable or unreliable at some time instant, then the part of the $p(X_m|X_s)$ corresponding to this measurement is simply set to a constant value and the particle set update is performed using the marginalized values, and when the measurement becomes available again, it is put back into the framework.

7.4.3 Self-calibration within the particle filter framework

The particle filter is usually employed for tracking the motion of an object. However, it can be used equally well to estimate the *intrinsic system parameters* or the sensor ego-motion. For example, in a videoconferencing framework there often exists an uncertainty in the position of the sensors. 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 some sensors moving independently (either by requirements or by design) 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 motion parameters into the state space. (Note that it is not correct to talk about object state space now, since the state space includes also the system parameters. In some sense, the parameters of a whole system including the object and the tracking system itself are estimated). One should be careful, though, to avoid introducing too many free parameters as this will boost the dimensionality of the state space ("curse of dimensionality") and lead to poor tracking performance.

7.5. SETUP AND MEASUREMENTS

[Figure 7.8]

[Figure 7.9]

We have implemented the tracking algorithms in two different naturally multimodal setups. Setup 1 was constructed in a large anechoic room ("flight room") and was used for studying the hunting behavior of the *Eptesicus Fuscus* echolocating bat. The flight room is pictured in Figure 7.8. Setup 2 is a typical acoustically untreated office environment with two cameras and

two microphone arrays, shown in Figure 7.9. Below we describe details of the audio and video hardware used in these setups.

7.5.1 Video modality

For the flight room setup, the video hardware consisted of two Kodak MotionCorder digital infrared cameras used at the resolution of 640 by 480 pixels and a frame rate of 240 Hz, placed at two corners of the experimental room. There were no sources of visible light in the room during the recording to ensure that the bat navigates using echolocation calls only. The video stream was recorded at a digital video recorder with embedded timestamps and then the parts of the recording corresponding to the audio activity were extracted. For the office setup, two color Sony EVI-D30 active cameras were used at a resolution of 320 by 240 pixels and the frame rate determined by the performance limit of the single computer doing both audio and video acquisition and data processing in real-time. The recorded video shows a frame rate of approximately 7 fps. The frame grabbers used were Matrox Meteor II.

[Figure 7.10]

To convert the world coordinates to the image coordinates, we use the *Direct Linear Transformation* (DLT) [44]. The DLT is defined by a 3x4 camera calibration matrix P which has 11 free parameters (the transformation is invariant to matrix scaling) and relates the world point coordinates (x, y, z) to the image coordinates (u, v) by the transformation

$$\begin{aligned} u &= \frac{p_{11}x + p_{12}y + p_{13}z + p_{14}}{p_{31}x + p_{32}y + p_{33}z + 1}, \\ v &= \frac{p_{21}x + p_{22}y + p_{23}z + p_{24}}{p_{31}x + p_{32}y + p_{33}z + 1}. \end{aligned} \tag{7.37}$$

(This is the transformation $(u_i, v_i) = \Psi_i(X_s) = \Psi_i(x, y, z)$ involved in the computation of the posterior probability). The matrix P_i for the i^{th} camera has eleven parameters $\{p_{11,}..., p_{14}, p_{21,}..., p_{33}\}$ which in this model are assumed to be independent with $p_{34} = 1$. These parameters were estimated by using a calibration object of known geometry (Figure 7.10) 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 1 mm. The image coordinates of every ball is determined manually from an image of the calibration object, thus giving 25 relationships between (x_j, y_j, z_j) and (u_{ij}, v_{ij}), $j = 1...25$ for the i^{th} camera of the form above with the unknown parameters P. This overdetermined linear system of equations is then solved for P using least squares for $i = 1$ and $i = 2$, providing the DLT parameters for both cameras of the system.

To obtain the feature-point locations coordinates from the video streams captured by the cameras, we use different techniques for different setups. In the flight room, infrared light is used for imaging and the room walls are covered by black audio-absorbing material; because of that, only a few bright spots can be seen in an infrared image, and a simple background subtraction technique works well to detect the bat as the only moving spot. The tracking is more complicated in case of a person moving in an office environment. The size of the image of head of a person standing 3 or 4 meters from the camera is at most 15 by 15 pixels, so the level of detail is insufficient to find facial features. We used a simple automatic tracker based also on background subtraction to roughly locate the head of the person and then refined the results manually by hand-clicking on the person's nose (it is easier to identify in a low-resolution picture than the lip area) to provide video coordinates for the tracker. We were able to achieve the agreement of separately computed video and audio trajectories within 50 mm error using this hand-tracking mode.

7.5.2 Audio modality

The audio tracking setup used is also different for the flight room setup and the office room setup because the nature of the sound signals are different. The bat produces ultrasonic echolocation chirps with the duration ranging from 2 to 20 ms and with the signal frequency decreasing from 50 to 20 kHz during each chirp. To capture the signal, seven Knowles Electronics FG-3329 microphones were used, with the microphones arranged on a horizontal plane in an L-shaped frame with the arms of the frame along two adjacent room walls. The microphones were connected to a custom-made preamplifier and digitized at 140 kHz per channel using a IoTech Wavebook board.

The positions of the individual microphones for audio processing were obtained from the video images (the microphones are visible in the image as small bright dots).

The office room audio setup consists of two microphone arrays mounted on the wall. Two groups of seven Panasonic WM-60A button microphones are used; each group includes one microphone at the center and six at the circumference of a circle of 0.30 m diameter. The microphones are connected to a custom-made low-noise low-distortion preamplifier based on a AD797 chip, and the signal is digitized at 22.05 kHz per channel using a PowerDAQ board. To ensure a good match between the coordinate systems used in audio and video processing, the calibration frame is set up with its axes parallel to the room walls, and the location of the origin of the audio coordinate system with respect to the central ball of calibration frame is obtained using a measuring tape.

The audio algorithms for estimating the TDOAs are based on generalized cross-correlation method [23] and have been described in our prior work [17] [45]. The TDOAs obtained by the cross-correlation constitute the audio measurements. The projection function from the state space to the measurement space is trivially obtained as

$$\tau_{ij} = \Phi(X_s) = \Phi(x, y, z) = (\chi_j - \chi_i)/c, \tag{7.38}$$

where χ_i is the distance between i^{th} microphone and the sound source and c is the sound speed. To recover the audio trajectory from set of TDOAs for tracker verification, the set of algorithms described in [17] and [46] were used.

7.6. TRACKING PERFORMANCE

We performed evaluation of the developed multimodal tracker on several sets of synthetic and real data obtained in different conditions. The synthetic data were used to verify the algorithm performance when the ground truth data is available. The real data include the tracking of an artificial moving sound source, a bat in the flight, and the speaking person in the room. Tracking in the flight room setup was done offline, and the real-time tracking system for the office environment

was implemented on a Dell Workstation with dual PIII-933MHz PC using Windows NT 4.0 and MSVC++ 6.0, working at approximately 7 fps. We are able to show that the performance of the multi-modal tracker is better that the performance of both the audio and the video trackers taken separately.

7.6.1 Synthetic data

We created a set of synthetic data by simulating an object moving in the spiral motion over the trajectory given by $x = \sin(2\pi t),\ y = 2.0 - t,\ z = \cos(2\pi t), t \in [0, 1]$. The frame rate was set to 240 fps, the discretization frequency to 140 kHz, and all geometric parameters of the system for the run were kept the same as the parameters for the real setup in the flight room. In every frame, we obtained true object coordinates in image frames and TDOA values. Then, the image coordinates and TDOA values were perturbed by Gaussian noise with zero mean and variances of 3 pixels and 10 samples, respectively. The tracker was initialized by the correct source position at $t = 0$ and zero velocity; the initial distribution of particles in the state space was chosen to be Gaussian around the correct source position with $\sigma_{init} = 0.2$ m. The σ_v and σ_a for the tracker were set to 3 pixels and 10 samples, corresponding to the true value of measurement noise. Any change to those values resulted in increase of the track estimation error, as can be expected. We performed several runs of the tracker with different number of particles measuring the average distance between the estimated and the true object position at every time step; the results are shown in Figure 7.11.

[Figure 7.11]

It can be seen that the performance improves with the number of particles. The audio tracker performance alone is not very good; this can be attributed to the fact that all the microphones lie in the same horizontal plane, which decreases the accuracy of object height determination. The performance improves with the number of particles. The performance of the video tracker alone is better, and the performance for the combined tracker is improved even more (approximately by 15%). For a large number of particles, the average tracking error is approximately 16.5 mm, which is 2.5 times less than the error obtained by pure object detection in every frame (about 38.3 mm).

This shows the effect of learning the object motion parameters by the filter.

[Figure 7.12]

The sensor motion recovery capability of the algorithm was also tested. We perform several experiments with synthetic data using one and two simulated planar microphone arrays rotating independently and one and two rotating cameras. 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. The case we show here corresponds to the simultaneous tracking and sensor motion recovery using only one fixed camera. The algorithm succeeds in tracking, despite the fact that the 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 7.12; the solid lines correspond to the true sensor rotation angles, and the dashed lines are the estimates 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, and in all cases where at least one sensor position is fixed, tracking with simultaneous parameter estimation succeeded in recovering both the object motion and the sensor motion. When all sensors are free to rotate, it is impossible to distinguish between sensor and object motion. Multi-point self-calibration should be used in this case.

7.6.2 Ultrasonic sounds in anechoic room

[Figure 7.13]

[Figure 7.14]

To verify algorithm performance in controlled environment on real data, we acquired the multi-

channel audio and video recordings of an echolocating bat hunting for a mealworm prey [46] in an anechoic environment (the flight room). In addition, we performed several experiments with the person-carried sound source in the flight room. TDOA values for all microphone pairs and the object position in the frame for both cameras were used as an input to our multi-modal CONDENSATION tracker. As TDOA values are not available in every frame (due to the bat being acoustically active only intermittently), linear interpolation is used to obtain missing values. (If audio information is used only when it is available, the trajectory essentially jumps back and forth between the video trajectory and audio points). In addition, the video trajectory and audio data points was determined from video and audio data independently. In Figures 7.13 and 7.14, the video-determined trajectory, audio data points corresponding to the individual echolocating bat calls, and the output of the tracker are plotted for two experiments.

The pictures show that the independently obtained video and audio trajectories are in fairly good agreement. The misalignment between them is likely to be due to the bias in determination of the microphone coordinates, which is done by video and can be inaccurate because the microphones lie far from the well-calibrated area where the calibration object was placed. The output of the multi-modal tracker integrates the audio and video information and lies between the audio and video tracks, as expected. No ground truth data is available for these runs, so the comparison with ground truth cannot be done.

7.6.3 Occlusion handling

We tested occlusion handling ability in the office environment setup. It is much more noisy and reverberant than the flight room. Using the setup described above, we performed real-time tracking of a single speaker moving in the field of view of a tracking system. The data was then processed off-line as well to recover the speaker trajectory. Low discretization frequency, relatively small intermicrophone distance within an audio subarray, and large distance from the microphone array to the speaker all contribute to relatively low accuracy of audio data, so the video data is the primary source of information for the run. The audio error is though about 10

cm on average, which is sufficient to roughly localize the speaker (e.g, for camera pointing) using audio data alone.

[Figure 7.15]

We also tested the algorithm's robustness to occlusion. Normally, the speaker is visible to both of the tracking cameras. When only one camera can see the speaker due to occlusion or being out of the field of view, video information alone cannot be used to recover the speaker coordinates. The multimodal tracker, however, can continue to track the speaker because of the audio constraints. In the Figure 7.15, we show the tracking results with a simulated occlusion. For the marked part of track, the video data from one of the cameras were omitted. Still, the track stays near the video trajectory, although it is influenced more by audio data now. When the video data is available again, the tracking error decreases back to the original value.

7.7. CONCLUSIONS

The described multimodal tracking algorithm provides a natural framework to integrate multimodal information and is robust to partial unavailability of input measurements (e.g., video occlusion or audio noise). The audio subsystem of the tracker is calibrated using the novel automatic microphone position calibration algorithm. The algorithm does not require precise placement of the calibration loudspeakers, and the only constraint we impose is that each loudspeaker is placed at or very close to one of the array microphones. We derive a closed-form approximate solution and further refine it by nonlinear minimization. We also derive and verify the expression for the variance of our estimator. The tracking and calibration algorithms are extensively validated on simulated and real data. We anticipate continued development of the system and achievement of real-time multimodal tracking ability on common hardware.

ACKNOWLEDGEMENTS

This chapter is an extended version of conference papers [47] and [48]. Partial support of NSF awards 0086075 and 0205271 and of ONR grant N00014951021 is gratefully acknowledged. We would also like to thank Dr. Cynthia F. Moss and Kaushik Ghose (Auditory Neuroethology Laboratory, Neuroscience and Cognitive Science Program, Department of Psychology, University of Maryland, College Park) for providing us with the flight room experimental data used in this chapter.

APPENDIX I: JACOBIAN COMPUTATIONS

The following are the derivatives necessary for the minimization routine. These derivatives constitute the non-zero elements of the Jacobian matrix.

$$\begin{aligned}
\frac{\partial TOF_{ij}^{actual}}{\partial mx_i} &= -\frac{\partial TOF_{ij}^{actual}}{\partial sx_j} = \frac{mx_i - sx_j}{c\|m_i - s_j\|} \\
\frac{\partial TOF_{ij}^{actual}}{\partial my_i} &= -\frac{\partial TOF_{ij}^{actual}}{\partial sy_j} = \frac{my_i - sy_j}{c\|m_i - s_j\|} \\
\frac{\partial TOF_{ij}^{actual}}{\partial mz_i} &= -\frac{\partial TOF_{ij}^{actual}}{\partial sz_j} = \frac{mz_i - sz_j}{c\|m_i - s_j\|}
\end{aligned} \tag{7.39}$$

APPENDIX II: CONVERTING THE DISTANCE MATRIX TO A DOT PRODUCT MATRIX

[Figure 7.16]

Assume that we chose the k^{th} point as the origin of our coordinate system. Let d_{ij} be the distance between i^{th} and j^{th} point. Using the cosine law (Figure 7.16), one can write

$$d_{ij}^2 = d_{ki}^2 + d_{kj}^2 - 2d_{ki}d_{kj}cos(\alpha). \tag{7.40}$$

The dot product b_{ij} is further defined as

$$b_{ij} = d_{ki}d_{kj}cos(\alpha). \tag{7.41}$$

Combining the above two equations,

$$b_{ij} = \frac{1}{2}(d_{ki}^2 + d_{kj}^2 - d_{ij}^2). \tag{7.42}$$

However, this equation assumes that the k^{th} point is the coordinate system origin, and we need to get the dot product matrix with the centroid as the origin. Let B and B^* be the dot product matrices with respect to the k^{th} point and the centroid as the origin, respectively. Let X^* be the matrix of coordinates with the origin shifted to the centroid.

$$X^* = X - \frac{1}{N}\mathbf{I}^* X, \tag{7.43}$$

where $\mathbf{I}^*$ is an $N \times N$ matrix with all elements equal to one (not simply an identity matrix). Now B^* can be written in terms of B as follows:

$$\begin{aligned} B^* &= X^* X^{*T} \\ &= B - \frac{1}{N}B\mathbf{I}^* - \frac{1}{N}\mathbf{I}^* B + \frac{1}{N^2}\mathbf{I}^* B \mathbf{I}^*. \end{aligned}$$

Hence the ij^{th} element in B^* is given by

$$b_{ij}^* = b_{ij} - \frac{1}{N}\sum_{l=1}^{N} b_{il} - \frac{1}{N}\sum_{m=1}^{N} b_{mj} + \frac{1}{N^2}\sum_{o=1}^{N}\sum_{p=1}^{N} b_{op}. \tag{7.44}$$

Using equation (7.42), we obtain that

$$b_{ij}^* = -\frac{1}{2}\left[d_{ij}^2 - \frac{1}{N}\sum_{l=1}^{N} d_{il}^2 - \frac{1}{N}\sum_{m=1}^{N} d_{mj}^2 + \frac{1}{N^2}\sum_{o=1}^{N}\sum_{p=1}^{N} d_{op}^2 \right]. \tag{7.45}$$

This operation is also known as double centering (i.e., subtract the row and the column means from its elements, add the grand mean, and then multiply by $-\frac{1}{2}$).

REFERENCES

[1] B. Stein, P. Laurienti, T. Stanford, and M. Wallace (2000). "Neural mechanisms for integrating information from multiple senses", Proc. IEEE ICME 2000, New York, NY, pp. 567-570, July 2000.

[2] K. C. Cheok, G. E. Smid, and D. J. McCune (2000). “A multisensor-based collision avoidance system with application to a military HMMWV”, Proc. IEEE Conf. Intelligent Transportation Systems, Dearborn, MI, pp. 288-292, October 2000.

[3] S. Ben-Yacoub, J. Luettin, K. Jonsson, J. Matas, and J. Kittler (1999). “Audio-visual person verification”, Proc. CVPR 1999, Fort Collins, CO, vol. 1, pp. 1580-1585, June 1999.

[4] V. Pavlovic, A. Garg, J. M. Rehg, and T. S. Huang (2000). “Multimodal speaker detection using error feedback dynamic Bayesian networks”, Proc. CVPR 2000, Hilton Head, SC, vol. 2. pp. 34-41, June 2000.

[5] G. Pingali, G. Tunali, and I. Carlbom (1999). “Audio-visual tracking for natural interactivity”, Proc. ACM Multimedia 1999, Orlando, FL, vol. 1, pp. 373-382, October 1999.

[6] F. Quek, D. McNeill, R. Bryll, C. Kirbas, H. Arslan, K. E. McCullough, N. Furuyama, and R. Ansari (2000). “Gesture, speech and gaze cues for discourse segmentation”, Proc. CVPR 2000, Hilton Head, SC, vol. 2, pp. 247-254, June 2000.

[7] O. Faugeras (1993). “Three-Dimensional Computer Vision: A Geometric Viewpoint”, MIT Press, Cambridge, MA, 1993.

[8] G. Wei amd S. Ma (1994). “Implicit and explicit camera calibration: Theory and experiments”, IEEE Trans. Pattern Analysis and Machine Intelligence, vol. 16, no. 5, pp. 469-480.

[9] D. Liebowitz and A. Zisserman (1998). “Metric rectification for perspective images of planes”, Proc. IEEE CVPR 1998, Santa Barbara, CA, pp. 482-488, June 1998.

[10] B. Triggs (1998). “Autocalibration from planar scenes”, Proc. ECCV 1998, Freiburg, Germany, pp. 89-105, June 1998.

[11] Z. Zhang (2000). “A flexible new technique for camera calibration”, IEEE Trans. Pattern Analysis and Machine Intelligence, vol. 22, no. 11, pp. 1330-1334.

[12] Y. Rockah and P. M. Schultheiss (1987). ”Array shape calibration using sources in unknown locations. Part II: Near-field sources and estimator implementation”, IEEE Trans. Acoustics, Speech, and Signal Processing, vol. ASSP-35, no. 6, pp. 724-735.

[13] J. M. Sachar, H. F. Silverman, and W. R. Patterson III (2002). “Position calibration of large-aperture microphone arrays”, Proc. IEEE ICASSP 2002, Orlando, FL, vol. 2, pp. 1797-1800,

May 2002.

[14] A. J. Weiss and B. Friedlander (1989). “Array shape calibration using sources in unknown locations – a maximum-likelihood approach”, IEEE Trans. Acoustics, Speech, and Signal Processing, vol. ASSP-37, no. 12, pp. 1958-1966.

[15] B. C. Ng and C. M. S. See (1996). “Sensor-array calibration using a maximum-likelihood approach”, IEEE Trans. Acoustics, Speech, and Signal Processing, vol. ASSP-44, no. 6, pp. 827-835.

[16] M. Isard and A. Blake (1996). “CONDENSATION conditional density propagation for visual tracking”, Intl. J. Computer Vision, vol. 29, no. 1, pp. 5-28.

[17] D. N. Zotkin, R. Duraiswami, V. Philomin, and L. S. Davis (2000). “Smart videoconferencing”, Proc. IEEE ICME 2000, New York, NY, pp. 1597-1600, August 2000.

[18] C. Wang, S. Griebel, M. Brandstein, and P. Hsu (2001). “Real-time automated video and audio capture with multiple cameras and microphones”, J. VLSI Signal Processing Systems, vol. 29, no. 1/2, pp. 81-99.

[19] S. Oh and V. Viswanathan (1992). “Hands-free voice communication in an automobile with a microphone array”, Proc. IEEE ICASSP 1992, San Francisco, CA, pp. 281-284, April 1992.

[20] M. Omologo, M. Matassoni, and P. Svaizer (2001). “Speech recognition with microphone arrays”, in Microphone Arrays: Signal Processing Techniques and Applications, ed. by M. S. Brandstein and D. B. Ward, Springer-Verlag, Berlin, Germany, pp. 331-353.

[21] J. D. de Jesus, J. J. V. Calvo, and A. I. Fuente (2000). “Surveillance system based on data fusion from image and acoustic array sensors”, IEEE Aerospace and Electronic Systems Magazine, vol. 15, no. 2, pp. 9-16.

[22] V. C. Raykar, I. V. Kozintsev, and R. Lienhart (2004). “Position calibration of microphones and loudspeakers in distributed computing platforms”, IEEE Trans. Speech and Audio Processing, vol. 13, no. 1, pp. 70-83.

[23] C. H. Knapp and G. C. Carter (1976). “The generalized correlation method for estimation of time delay”, IEEE Trans. Acoustics, Speech, and Signal Processing, vol. ASSP-24, no. 4, pp. 320-327.

[24] P. E. Gill, W. Murray, and M. H. Wright (1982). “Practical Optimization”, Elsevier Science, Netherlands.

[25] W. S. Torgerson (1952). “Multidimensional scaling. Part I: Theory and method”, Psychometrika, vol. 17, pp. 401-419.

[26] M. Steyvers (2002). “Multidimensional Scaling”, in Encyclopedia of Cognitive Science, Nature Publishing Group, London, UK.

[27] H. P. Press, S. A. Teukolsky, W. T. Vettring, and B. P. Flannery (1995). “Numerical Recipes in C: The Art of Scientific Computing”, Cambridge University Press, Cambridge, UK.

[28] T. J. Broida and R. Chellappa (1991). “Estimating the kinematics and structure of a rigid object from a sequence of monocular images”, IEEE Trans. Pattern Analysis and Machine Intelligence, vol. 13, no. 6, pp. 497-513.

[29] J. A. Fessler (1996). “Mean and variance of implicitly defined biased estimators (such as penalized maximum likelihood): Applications to tomography”, IEEE Trans. Image Processing, vol. 5, no. 3, pp. 493-506.

[30] A. K. Roy Chowdhury and R. Chellappa (2003). “Stochastic approximation and rate distortion analysis for robust structure and motion estimation”, Intl. J. Computer Vision, vol. 55, no. 1, pp. 27-53.

[31] H. L. van Trees (2001). “Detection, estimation, and modulation theory”, vol. 1, John Wiley and Sons Inc., Hoboken, NJ.

[32] J. H. DiBiase, H. F. Silverman, and M. S. Brandstein (2001). “Robust localization in reverberant rooms”, in Microphone Arrays: Signal Processing Techniques and Applications, ed. by M. S. Brandstein and D. B. Ward, Springer-Verlag, Berlin, Germany, pp. 157-180.

[33] M. S. Brandstein and H. F. Silverman (1997). “A robust method for speech signal time-delay estimation in reverberant rooms”, Proc. IEEE ICASSP 1997, Munich, Germany, pp. 375-378.

[34] M. Omologo and P. Svaizer (1997). “Use of the crosspower-spectrum phase in acoustic event location”, IEEE Trans. Speech and Audio Processing, vol. 5, no. 3, pp. 288-292.

[35] H. Wang and P. Chu (1997). “Voice source localization for automatic camera pointing system in videoconferencing”, Proc. IEEE ICASSP 1997, Munich, Germany, pp. 187-190.

[36] D. N. Zotkin, R. Duraiswami, E. Grassi, and N. A. Gumerov (2006). “Fast head-related transfer function measurement via reciprocity”, J. Acoustical Society of America, vol. 120, no. 4, pp. 2202-2215.

[37] M. Isard and A. Blake (1997). “ICONDENSATION: Unifying low-level and high-level tracking in a stochastic framework”, Proc. ECCV 1998, Freiburg, Germany, pp. 893-908, June 1998.

[38] J. Carpenter, P. Clifford, and P. Fearnhead (1999). ”An improved particle filter for non-linear problems”, IEEE Proc. Radar, Sonar, and Navigation, vol. 146, pp. 2-7.

[39] J. MacCormick and A. Blake (2000). “Probabilistic exclusion and partitioned sampling for multiple object tracking”, Intl. J. Computer Vision, vol. 39, no. 1, pp. 57-71.

[40] B. Li and R. Chellappa (2000). “Simultaneous tracking and verification via sequential posterior estimation”, Proc. CVPR 2000, Hilton Head, SC, vol. 2, pp. 110-117, June 2000.

[41] V. Philomin, R. Duraiswami, and L. S. Davis (2000). “Quasi-random sampling for CONDENSATION”, Proc. ECCV 2000, Dublin, Ireland, pp. 134-149, June 2000.

[42] G. Qian and R. Chellappa (2001). “Structure from motion using sequential Monte-Carlo methods”, Proc. ICCV 2001, Vancouver, Canada, pp. 614-621, July 2001.

[43] A. Doucet, N. de Freitas, and N. Gordons (eds.) (2001). “Sequential Monte-Carlo Methods in Practice”, Springer, New York, NY.

[44] R. Hartley and A. Zisserman (2000). “Multiple View Geometry in Computer Vision”, Cambridge University Press, Cambridge, UK.

[45] D. N. Zotkin, R. Duraiswami, L. S. Davis, and I. Haritaoglu (2000). “An audio-video front end for multimedia applications”, Proc. IEEE SMC 2000, Nashville, TN, pp. 786-791.

[46] K. Ghose, D. N. Zotkin, R. Duraiswami, and C. F. Moss (2001). “Multimodal localization of a flying bat”, Proc. IEEE ICASSP 2001, Salt Lake City, UT, pp. 3057-3060, May 2001.

[47] D. N. Zotkin, R. Duraiswami, H. Nanda, and L. S. Davis (2001). “Multimodal tracking for smart videoconferencing”, Proc. IEEE ICME 2001, Tokyo, Japan, pp. 37-40, August 2001.

[48] V. C. Raykar and R. Duraiswami (2004). “Automatic position calibration of multiple microphones”, Proc. IEEE ICASSP 2004, Montreal, QC, Canada, vol. 4, pp. 69-72, May 2004.

FIGURES

	s1	s2	s3	s4	m1	m2	m3	m4	m5	m6	m7
s1	?	?	?	?	X	X	X	X	X	X	X
s2	?	?	?	?	X	X	X	X	X	X	X
s3	?	?	?	?	X	X	X	X	X	X	X
s4	?	?	?	?	X	X	X	X	X	X	X
m1	X	X	X	X	?	?	?	?	?	?	?
m2	X	X	X	X	?	?	?	?	?	?	?
m3	X	X	X	X	?	?	?	?	?	?	?
m4	X	X	X	X	?	?	?	?	?	?	?
m5	X	X	X	X	?	?	?	?	?	?	?
m6	X	X	X	X	?	?	?	?	?	?	?
m7	X	X	X	X	?	?	?	?	?	?	?

FIG. 7.1: Pairwise distance matrix for four loudspeakers and seven microphones. Four microphones are attached to the loudspeaker forming four speaker-microphone pairs. The measured quantities are shown as 'X' and the unknown distances are shown as '?'.

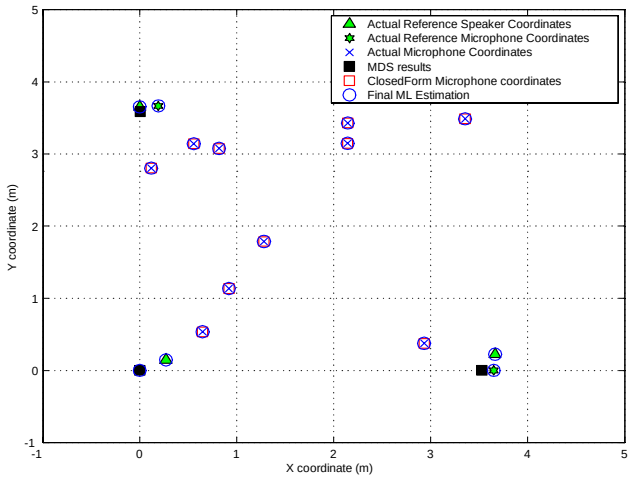


FIG. 7.2: Result of the proposed algorithm in two dimensions consisting of ten microphones and three speaker-microphone pairs.

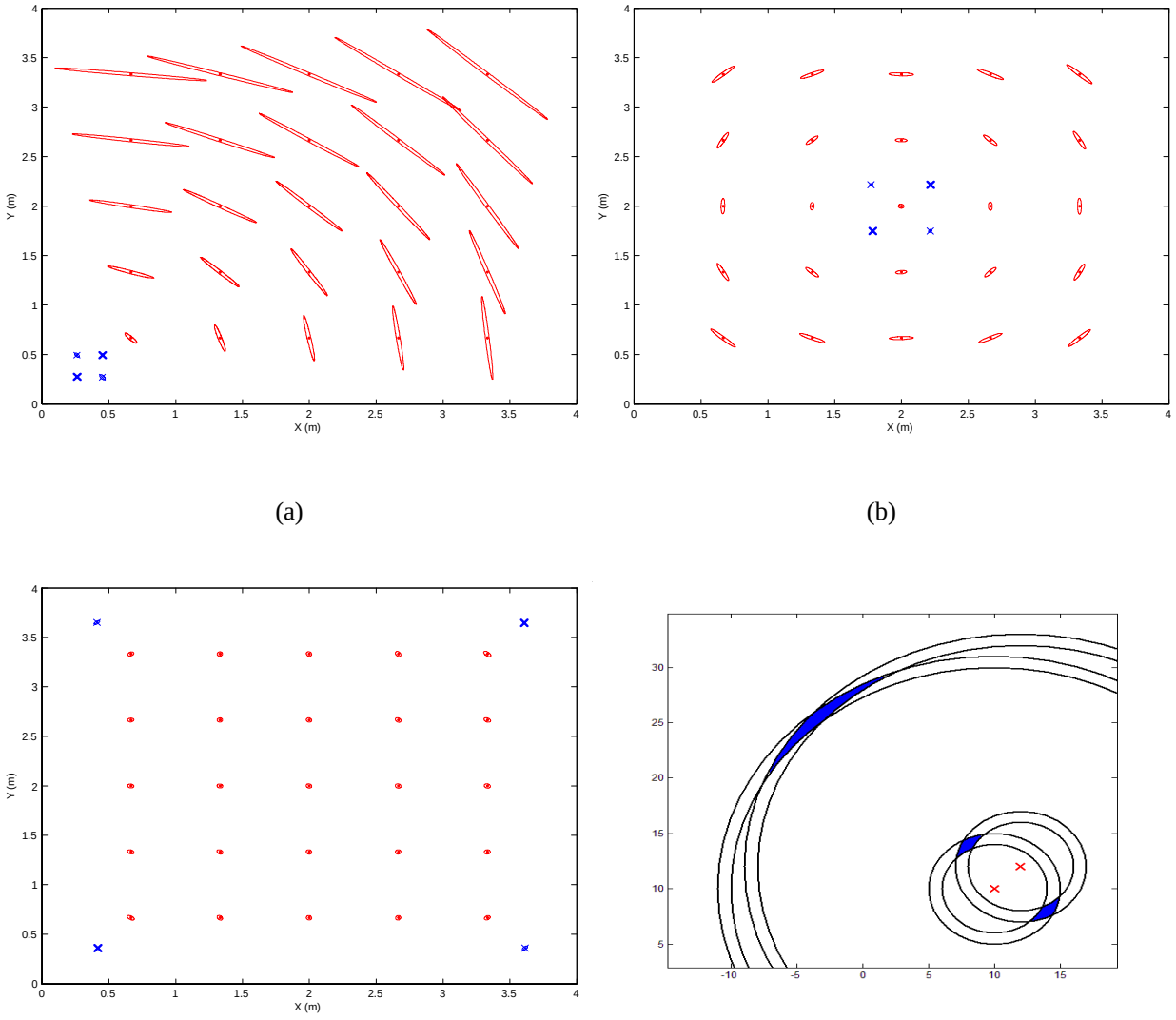
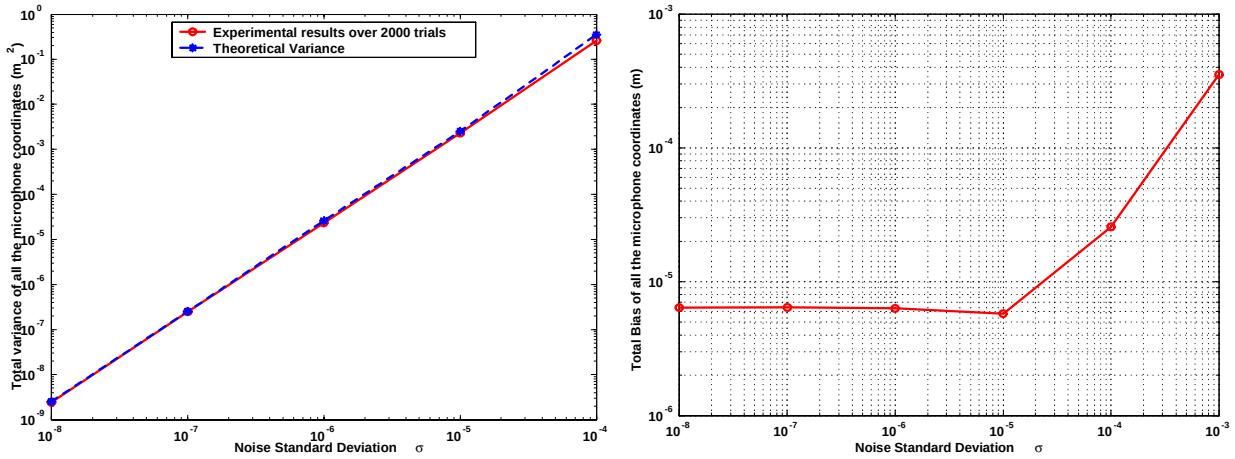


FIG. 7.3: 95% uncertainty ellipses for a regular 2-D array of 25 microphones and 4 loudspeakers. Noise variance for all cases is $\sigma^2 = 10^{-9}$. The microphones are represented as dots (.) and the loudspeakers as crosses ($\times$). The position of one loudspeaker and the x coordinate of another is assumed to be known. In (a) and (b) the loudspeakers are close to each other and in (c) they are spread out one at each corner of the grid. Drawing (d) explains the shape of the uncertainty ellipses.



(a) (b)

FIG. 7.4: (a) The total variance and (b) the total bias of all the microphone coordinates for increasing noise standard deviation σ. The network consists of 20 microphones and 5 speaker-microphone pairs. The theoretical variance is also shown.

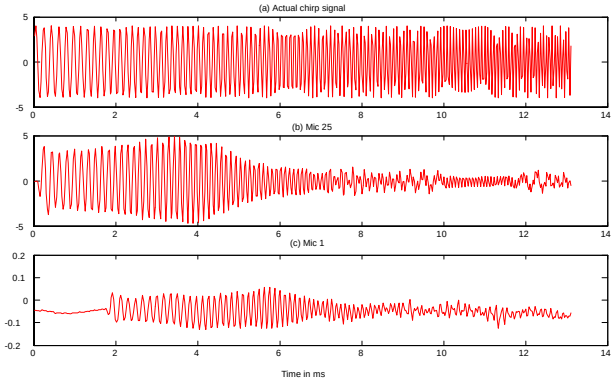


FIG. 7.5: (a) The actual chirp signal used in our setup. (b) The chirp signal received by the microphone directly attached to the speaker. (c) The delayed chirp signal received by another microphone.

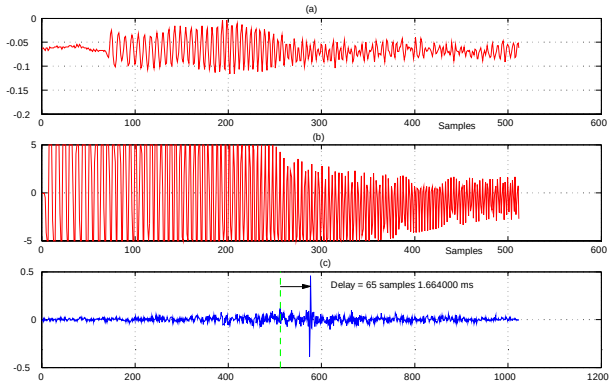
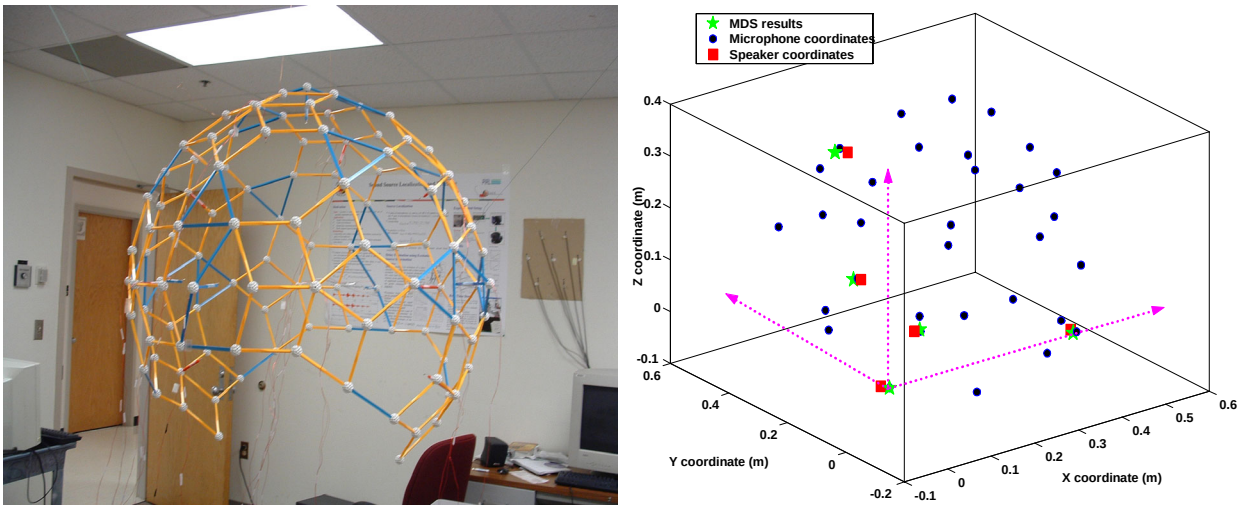


FIG. 7.6: (a) The delayed chirp signal received by a microphone. (b) The chirp signal received by the microphone attached directly to the speaker. (c) The GCC-PHAT function.



(a) (b)

FIG. 7.7: (a) The 32 element microphone array. (b) The results obtained from our algorithm.

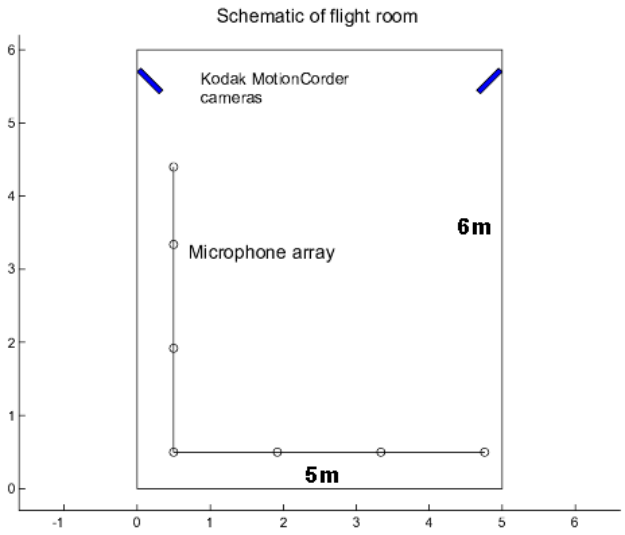


FIG. 7.8: Schematic of flight room experimental setup.

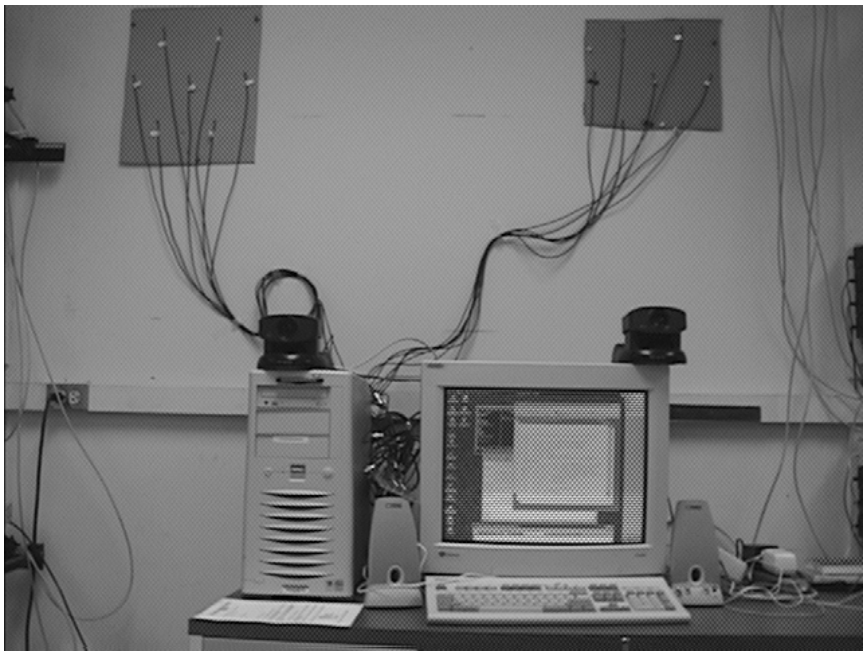


FIG. 7.9: A two-camera, two-array setup used in office room experiments.

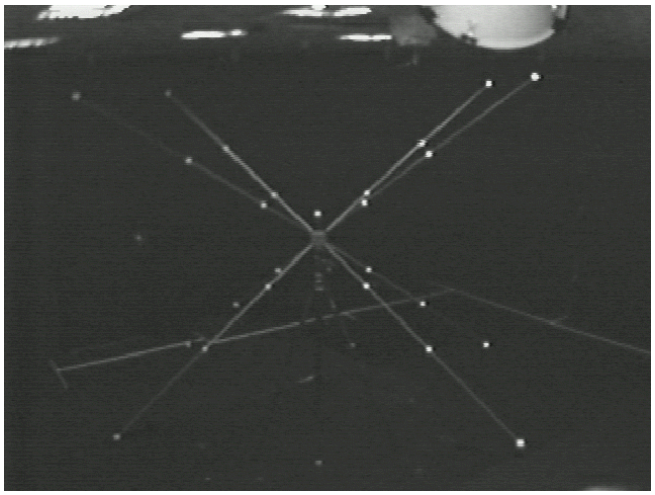


FIG. 7.10: Video calibration object.

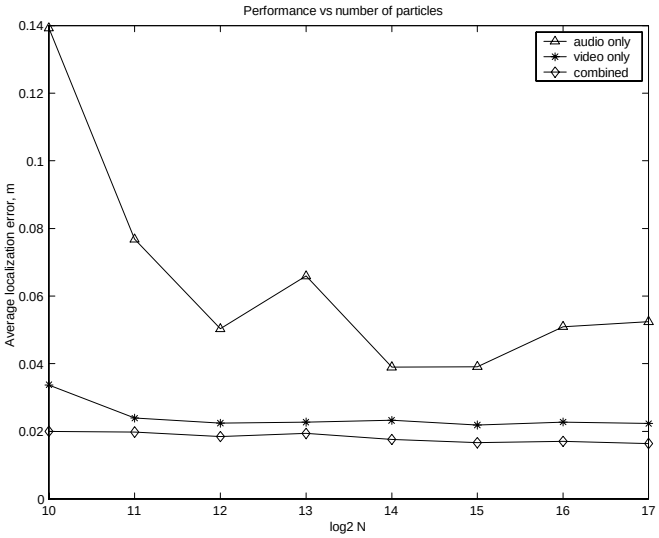


FIG. 7.11: Performance vs log number of particles.

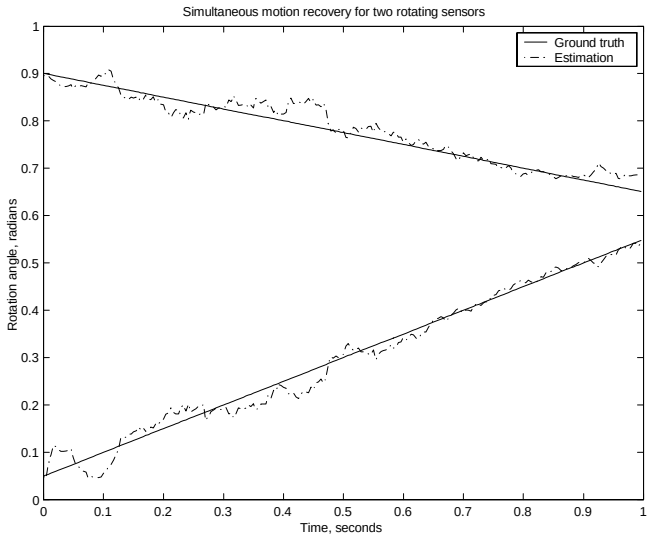


FIG. 7.12: Rotating sensor motion estimation.

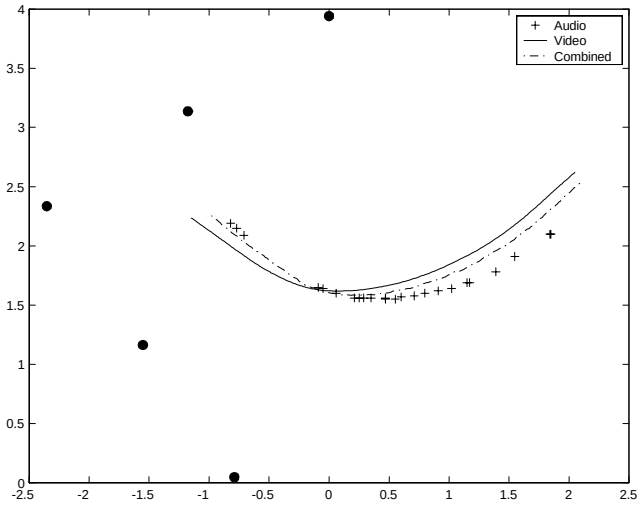


FIG. 7.13: Bat flight (0.95 seconds) and the microphone setup (large black dots).

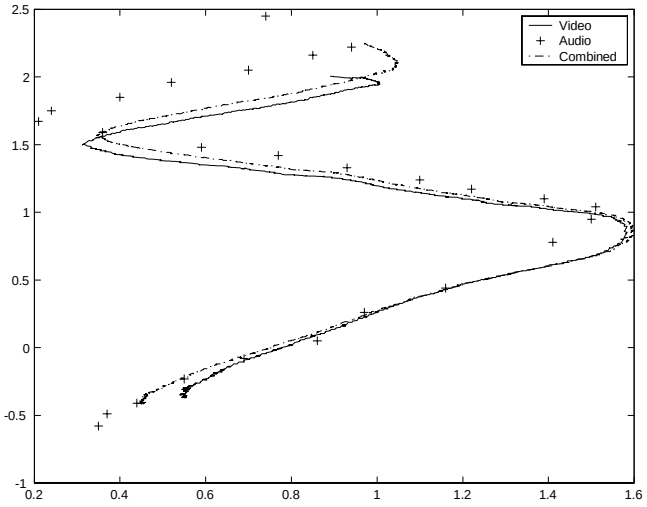


FIG. 7.14: A 6.5 seconds walk with the bat trainer in hand.

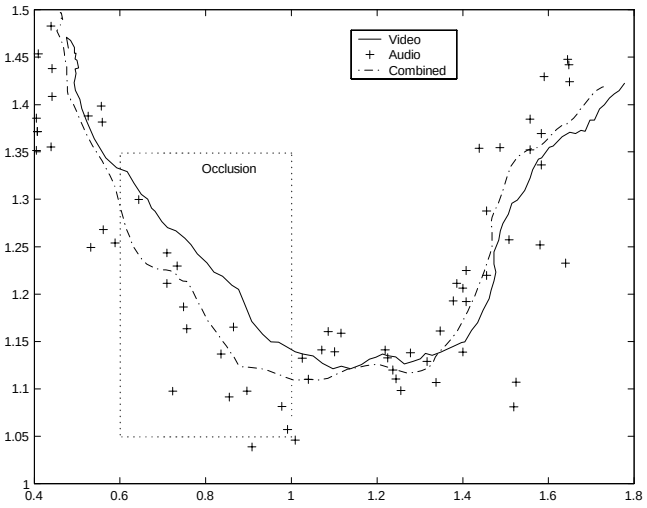


FIG. 7.15: Multimodal speaker tracking with occlusions.

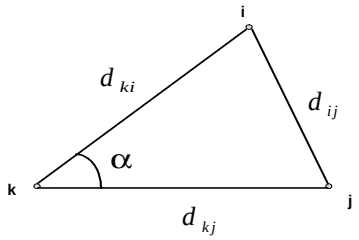


FIG. 7.16: Law of cosines.