

LEARNING THE TIME-DELAY MANIFOLD FOR ROBUST SPEAKER LOCALIZATION

Evan Ettinger¹, Shankar Shivappa², Deborah Goshorn¹, and Yoav Freund¹

¹Computer Science and Engineering
²Electrical and Computer Engineering
 University of California at San Diego
 La Jolla, CA 92023

ABSTRACT

We present an algorithm for high dimensional density estimation which is efficient (both computationally and statistically) when the distribution is concentrated close to a low dimensional smooth manifold. The algorithm uses several random projections to generate a hierarchical mixture of Gaussians which rapidly converges to the underlying manifold. We use this algorithm to perform robust estimation of the time delays in an ad-hoc microphone network. We utilize the model to calculate accurate time-delay vectors for two speakers that are talking at the same time.

Index Terms— Delay estimation, Position Measurement, Un-supervised Learning, Density Measurement

1. INTRODUCTION

High dimensional measurements of a physical system often occupy only a tiny fraction of the ambient space. The reason is that the physical system has only a few degrees of freedom, and the dimension of the resulting manifold is how many degrees there are. As measurements are usually noisy, points can lie close to the manifold, rather than exactly on it.

Learning a model of the manifold from sample data is of great value. Such a model exposes the redundancy in the measurements which we can exploit to correct measurement errors and reconstruct missing information. In this paper we present an efficient algorithm for MANifold Density Estimation (MADE). Our algorithm is especially applicable in the domains of sensor networks where manifolds often occur. Our interest in this work is in an ad-hoc microphone network and in time delay of arrival (TDOA) estimation.

Traditionally, microphones are carefully placed so that their physical and relative positions can be known. Because of this, microphone networks are typically either arranged in a line, on a plane, or a combination thereof. However, these traditional arrays are heavily constrained in their localization ability. Linear arrays can localize in one angular dimension, and planar arrays in two. On the other hand, an ad-hoc network that extends into all three spatial dimensions can discriminate between all locations in space as shown in Figure 1.

We choose one microphone as the reference microphone and measure all of the delays relative to it. Given n microphones, there exists a mapping from the location of the source, $\vec{x} \in \mathbb{R}^3$, to the $n-1$ dimensional vector $\vec{\Delta}(\vec{x}) \in \mathbb{R}^{n-1}$, where $\Delta_i(\vec{x})$ is the time delay between microphone i and the reference microphone. The image of this mapping is a smooth three dimensional manifold in \mathbb{R}^{n-1} . This is particularly useful when the microphone network is very large, since the manifold is still three-dimensional. Using MADE we cre-

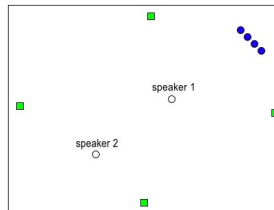


Fig. 1. Top-view layout of a room with two sets of microphones aiming to localize speakers 1 and 2. A linear microphone array (solid circles) fails to localize both speakers since the speakers lie in the same direction from the array. The ad-hoc microphone network (in squares) successfully localizes both speakers.

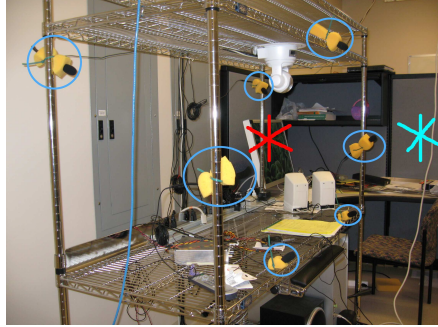
ate a model of this manifold which we then use to remove ambiguities when separating two sound sources.

In this work, we show how to robustly estimate $\vec{\Delta}(\vec{x})$ given knowledge of the underlying manifold structure via MADE. As noted before, this methodology can be easily adapted for many other estimation correction scenarios, especially in sensor networks.

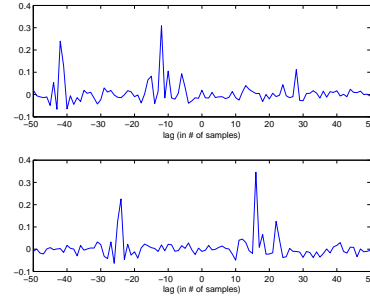
The rest of the paper is organized as follows. In section 2 we discuss the methods behind the TDOA estimation algorithm. Section 3 describes random projection trees, the data structure used for storing the local neighborhoods of the manifold, and describes our TDOA correction algorithm MADE. Section 4 describes the experiment that uses the MADE algorithm to acquire the TDOA estimates for two speakers simultaneously speaking in different locations.

2. TDOA ESTIMATION

To estimate $\vec{\Delta}(\vec{x})$ we use the popular PHase Transform (PHAT) algorithm that estimates delays between microphone pairs by doing a type of generalized cross-correlation that has shown to give superior performance [1, 2, 3]. However, like most statistical estimation techniques, these TDOA estimates given by PHAT are not always accurate. It has been observed that the errors of these estimates are highly dependent upon the noise present in the room and upon the microphone separation distance [4, 5]. Yu et al. report that the quality of the PHAT estimates severely degrade as the signal to noise ratio (SNR) becomes large. MADE can help by providing a concise summary of the structure of the underlying manifold.



(a)



(b)

Fig. 2. (a) Image of our microphone network. (b) Two PHAT-correlation sequences for two different microphones with the reference microphone. Two maximum peaks are clearly visible.

3. TDOA MANIFOLD AND CORRECTION

In this section we describe a data structure to efficiently store and summarize the local structure of a manifold. We then present our manifold density estimation algorithm that efficiently corrects for noisy TDOA estimates.

3.1. RP-Trees

Random projection trees (RP-Trees) are a type of binary decision tree used to partition data. They are almost identical to the popular kd-tree except that they use a random projection of the data at each level of the tree instead of splitting on a coordinate axis. We utilize these space partitioning structures extensively in MADE, and exploit the fact that they can uncover inherent low-dimensional structure very efficiently. To get partitions that are low-dimensional requires a tree depth that depends only on the dimension of the manifold and not on the dimension of the ambient space. For more about RP-Trees see [6, 7].

3.2. Learning the Manifold Density

Many of the noisy estimates of the TDOA vector in the original space do not lie on the low-dimensional manifold, but they can be denoised by projecting the estimates back onto the manifold. In order to do so smoothly, we estimate the probability density of the manifold, given a collection of data points that lie on it. With this density estimate, we can compute the projection of a noisy data vector modeled by some prior onto the manifold in the maximum-likelihood (ML) sense.

Gaussian mixture models (GMMs) are a flexible and computationally attractive choice of probability density functions. A typical approach for learning a GMM for a density distribution would involve employing the EM algorithm in the original space, but this naive approach does not exploit the low dimensional structure of the manifold. In this work, we propose a MANifold Density Estimation (MADE) algorithm that exploits the manifold structure represented in the RP-Tree by modeling the data points in each node j of the tree by a multivariate Gaussian with mean μ_j and covariance Σ_j . The overall mixture model is what we will call a hierarchical mixture of Gaussians modeling the manifold.

When estimating a mixture of Gaussian from a finite training set there is a tradeoff to consider. When using a fewer number of mixture components, we have fewer model parameters to fit and hence

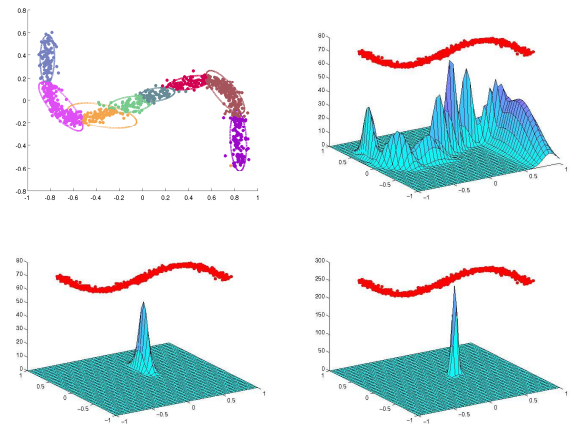


Fig. 3. A toy example S manifold with RP-Tree leaves in different colors (upper-left), the MADE product density distribution for the manifold (upper-right), the density of our prior distribution as input to MADE (lower-left), and the a-posteriori belief (lower-right).

the variance of our estimated model will be lower. The model, however, could have a large bias. By increasing the number of mixture components, we can decrease this bias, but our estimated model has higher variance due to the larger number of parameters. We see the same tradeoff in estimating the parameters of the multivariate Gaussians at each node of the RP-Tree. As we go deeper down the RP-Tree, each node corresponds to a smaller subsection of the manifold and hence has a low bias. On the other hand, it has fewer training points to estimate the density from and hence has greater variance. This means that different prunings of the complete RP-Tree have different bias/variance properties. An effective solution to this problem of finding the best pruning is to use a weighted average over all possible prunings of the tree. MADE estimates this weighted average of the GMM over all prunings of the RP-Tree, \mathcal{T} , along the same lines of the context tree algorithm of Willems et al.[8].

Let $D = \{x_1, x_2, \dots, x_n\}$ be a set of training points lying on the manifold and D_j be the subset that falls in node j of the RP-Tree. We define the quantity $\mathcal{L}_n(j)$ to be the likelihood of D_j under the

Gaussian, $N(\mu_j, \Sigma_j)$, corresponding to node j . Since the data is iid,

$$\mathcal{L}_n(j) = \prod_{x_j \in D_j} N(\mu_j, \Sigma_j)(x_j)$$

We also define the quantity $\mathcal{L}_s(j)$ to be the average likelihood of D_j over all prunings of the subtree rooted at j . If j is a leaf node, $\mathcal{L}_s(j) = \mathcal{L}_n(j)$. Otherwise, $\mathcal{L}_s(j) = \frac{1}{2}\mathcal{L}_n(j) + \frac{1}{2}\mathcal{L}_s(l(j))\mathcal{L}_s(r(j))$ where $l(j)$ and $r(j)$ correspond to the left and right children of node j . We define β_j as a ratio of the likelihood of D_j under the Gaussian corresponding to j to the likelihood of D_j under all prunings of the subtrees rooted at $l(j)$ and $r(j)$.

$$\beta_j = \frac{\mathcal{L}_n(j)}{\mathcal{L}_s(l(j))\mathcal{L}_s(r(j))}$$

Intuitively a large value of β_j corresponds to overfitting in the subtrees below node j . Conversely, a small value corresponds to much better fits in the subtrees. If β_j is larger than 1, we set it to 1, since this means that node j fits the data better than its subtrees and that they should be pruned. Note that for leaf nodes, β_j is set to 1. Once the $\vec{\beta}$ vector is computed, we can represent the weighted average probability distribution as follows

$$P(x) = \sum_{i \in \mathcal{T}} \alpha_i N(\mu_i, \Sigma_i)(x)$$

$$\alpha_i = \frac{(1 - \beta_{\pi(i)})\beta_i}{\beta_{\pi(i)}} \alpha_{\pi(i)}$$

where $\pi(i)$ denotes the parent of node i and $\alpha_{root} = \beta_{root}$. The α_i equation serves as a normalization term for the weighted average.

Returning to the problem of denoising a TDOA estimate, given a prior distribution about our confidence in the noisy TDOA, we can compute the product of the manifold’s density from MADE with this prior. We can then find the maximum of this a-posteriori distribution resulting in a de-noised estimate of the TDOA vector in the ML sense. Figure 3 depicts the entire process on a toy S-manifold. First an RP-Tree is built on the training data and Gaussians are fit at each node of the tree (only the leaves are depicted in the figure). On a new training set we calculate each β_j . Finally, given a prior distribution for a noisy estimate, we can calculate the product distribution which results in a peaked distribution centered on the manifold.

4. EXPERIMENTAL RESULTS

We have at our disposal a small network of 7 CM100 CAD unidirectional microphones connected to a MOTU 896HD audio digitizer and then to an Apple G5. A picture of our setup is given in Figure 2(a). We denote one microphone as the reference (the middle upper one) and only calculate the PHAT estimates between each microphone and the reference. As a consequence each $\vec{\Delta}(\vec{x})$ lies in \mathbb{R}^6 , however, as discussed earlier, the region of feasible TDOAs for our network only fills a very small portion of this space and lies on a low dimensional manifold. The power in using MADE is that it would still be efficient even when each $\vec{\Delta}(\vec{x})$ lie in \mathbb{R}^{60} or \mathbb{R}^{600} or more, because the underlying manifold is still 3-dimensional. The complexity of how large the RP-Tree need be relies only on this fact.

4.1. Learning the TDOA Manifold

Since Gaussian white noise is entirely uncorrelated with shifts of itself, the expected correlation of incorrect shifts of such a signal with

delayed versions of itself is zero. This makes it ideal for getting accurate $\vec{\Delta}(\vec{x})$ estimates for a particular location since only the true TDOA should exhibit a strong peak. The PHAT correlations for such signals are extremely peaky and stable for a fixed source location. With this in mind, we took a speaker playing white noise and calculated the $\vec{\Delta}(\vec{x})$ estimates of each 250ms window from the network at a 16kHz sampling rate. We move the speaker around the desired region of interest for approximately 15 minutes resulting in close to 7,000 $\vec{\Delta}(\vec{x})$ vectors representing various locations in the room. We then removed those points that were more than three standard deviations from the mean and the remaining $\sim 6,000$ were used to train an RP-Tree with 16 leaves (deep enough to capture the low dimensionality of the TDOA manifold). The projections of these points onto their principal eigenvectors are shown in Figure 4 with different colored points representing the different leaves of the RP-Tree. The top two PCA eigenvectors explain 95% of the variance and the top three explain 99.8%. The manifold in this data is quite clear from these projections. Some landmarks are pointed to on the 2-d manifold to give a sense of the orientation of the speaker relative to the network that resulted in a particular $\vec{\Delta}(\vec{x})$. We also show the 3-d projection to give a sense of the slightly curved nature of our manifold. It is worth noting that this training process need be done only once for a particular microphone arrangement, and is extremely simple to do.

4.1.1. Intrinsic Dimensionality

Figure 4 explicitly illustrates the 3-d resolution capabilities of our microphone network. In this experiment, we moved a source of white noise along a 3-dimensional cross, once in front of the network (cyan points) and once inside the enclosure (red points). This path is illustrated in Figure 2(a). In the region enclosed by the microphones we can localize the source in all three spatial dimensions. This is a significant improvement over linear arrays, which can at best localize angular positions (planar arrays can localize two angular dimensions). However, in the regions outside the enclosure our network is effectively planar and hence we can only resolve in two dimensions.

4.2. Separating Two Point Sources

Having a notion of the distribution of $\vec{\Delta}(\vec{x})$ that we would expect on the TDOA manifold can help for correcting erroneous $\vec{\Delta}(\vec{x})$ estimates by finding the most likely place on the manifold. With this in mind, we tested how robust our method is in locating the appropriate delays for two sources playing simultaneously. Our sources were either white noise or a recording of a human speaker. Running PHAT based correlation on these simultaneous two-source signals results in sequences that contain two distinct peaks at the appropriate delays for each source as seen in Figure 2(b). The problem now lies in the combinatorial explosion of potential $\vec{\Delta}(\vec{x})$ estimates that can be constructed for a single source. For each PHAT correlation sequence we do not know which peak corresponds to which source. However, since the TDOA manifold is low dimensional, it is unlikely for incorrect $\vec{\Delta}(\vec{x})$ combinations to result in points that fall on the manifold. We will exploit the fact that the TDOA manifold only occupies a very small portion of its ambient space. We use MADE to evaluate how likely it is for each combination to lie on the TDOA manifold.

After learning the $\vec{\beta}$ vector on our training set, we noticed that nearly the entirety of the weights were being placed on the leaves of the RP-Tree. Moreover, the tree was able to split the manifold in such a way that each region only spanned contiguous regions on

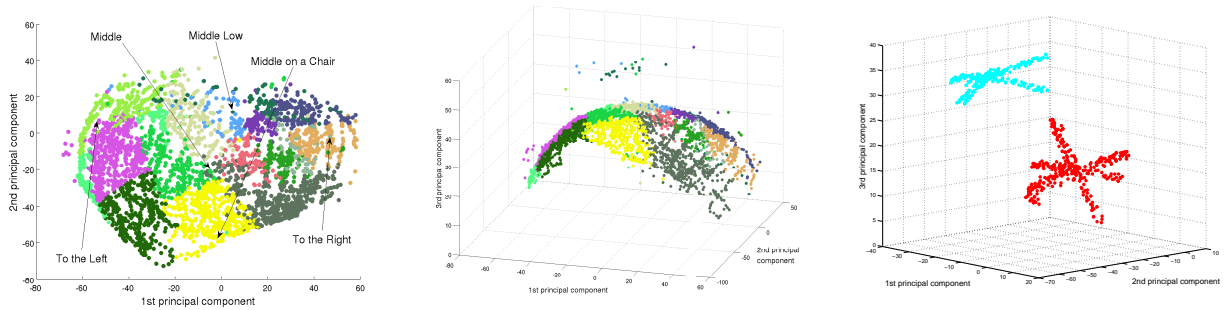


Fig. 4. Projection of the TDOA manifold on its top 2 principal components with landmarks labeled (left). First 3 eigenprojections (middle). Showing the 3-dimensional structure of the TDOA manifold as well as the 2-dimensional structure at different locations (right).

the manifold. This allowed us to make the simplifying algorithmic assumptions that the ML estimate can simply be made by finding the leaf that gives us the largest product distribution with MADE. In this application, we assumed a narrow spherical Gaussian prior around each of the 2^6 possible $\vec{\Delta}(\vec{x})$ combinations calculated from the two largest peaks in the PHAT correlation sequences.

We considered three cases. In the first case, we played different white noise out of each of two speakers simultaneously placed in different locations. We then played white noise with a sports radio announcer, and then two different segments of the sports radio announcer simultaneously. We calculated $\vec{\Delta}(\vec{x})$ for frames of the signal 250ms long. We determined ground truth for each of the two sources by using the PHAT results when only one of the two sources was played, which was very stable over the interval. Figure 5 shows the results of the first experiment. The large blue points represent the potential 2^6 $\vec{\Delta}(\vec{x})$ combinations. The largest point colored in black is the ML point recovered by our algorithm and it is *exactly* the true $\vec{\Delta}(\vec{x})$ of one of the sources (the remaining peaks not used corresponds *exactly* to the other source). Identical results held for the white noise with speech and the speech with speech trials.

After recovering each source's $\vec{\Delta}(\vec{x})$ vector we used a delay-and-sum (DEAS) filter to focus the network to each source. We observed an almost constant gain of 6.2 dB in SNRs from the DEAS over a range of -20 dB to 0 dB SNR which coincides with the theory of DEAS beamformers presented in [9]. We achieved an additional 2.5 dB improvement in the S/N by using DEAS to reconstruct the **noise** signal and subtract it from the microphone signals before performing the DEAS for the speech. Recordings of these results are available in the supporting material¹.

5. REFERENCES

- [1] C.H. Knapp and G.C. Carter, "The generalized correlation method for estimation of time delay," *IEEE Trans. on Acoustics, Speech and Signal Processing*, vol. 27, no. 4, pp. 320–327, 1976.
- [2] Ehud Ben-Reuven and Yoram Singer, "Discriminative binaural sound localization.," in *NIPS*, 2002, pp. 1229–1236.
- [3] J. H. DiBiase, H. F. Silverman, and M. S. Brandstein, *Robust localization in reverberant rooms in Microphone arrays: signal processing techniques and applications*, Berlin: Springer-Verlag, 2001.

¹At <http://coldcall.ucsd.edu/~eettinger/micarray.html>

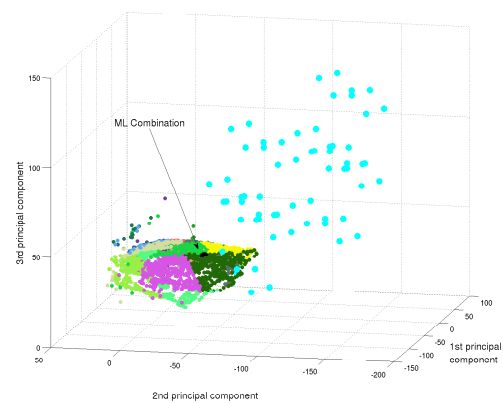


Fig. 5. The 64 potential $\vec{\Delta}(\vec{x})$ vectors (larger points in cyan) and the ML winner from the RP-Tree Context algorithm (large point depicted in black)

- [4] Ying Yu and H.F. Silverman, "An improved tdoa-based location estimation algorithm for large aperture microphone arrays," in *Acoustics, Speech, and Signal Processing:2004*, 2004, pp. iv–77–iv80.
- [5] D. Bechler and K. Kroschel, "Reliability criteria evaluation for tdoa estimates in a variety of real environments," in *Acoustics, Speech, and Signal Processing:2005*, 2005.
- [6] Yoav Freund, Sanjoy Dasgupta, Mayank Kabra, and Nakul Verma, "Learning the structure of manifolds using random projections," in *Neural Information Processing Systems*, 2007.
- [7] Sanjoy Dasgupta and Yoav Freund, "Random projection trees and low dimensional manifolds," *UCSD Technical Report CS2007-0890*, 2007.
- [8] F.M.J. Willems, T.J. Tjalkens, and T. Ignatenko, "Context-tree weighting and maximizing: Processing betas," in *Proc. of Inaugural Workshop ITA (Information Theory and its Applications)*, 2006.
- [9] D.H. Johnson and D.E. Dudgeon, *Array Signal Processing: Concepts and Techniques*, New Jersey: Prentice Hall, 1993.