Sumit Basu, SteveSchwartz, and Alex Pentland. "Wearable Phased Arrays for Sound Localization and Enhancement."
In Proceedings of the IEEE Int'l Symposium on Wearable Computing 2000 (ISWC '00). Atlanta, GA. October, 2000.

Wearable Phased Arrays for Sound Localization and Enhancement

Sumit Basu, Steve Schwartz, and Alex Pentland
MIT Media Laboratory
E-mail: {sbasu,schwartz,sandy}@media.mit.edu

Abstract

We present the idea of a flexible phased array of microphones for wearable computers. We show how such an array can be used for both source localization and signal enhancement. As a result, it can help to solve two fundamental problems for audio input to wearables: determining who is speaking when (user commands vs. nearby speech, who is speaking in a conversation, etc.) and obtaining high-quality audio without the use of a headset microphone. We describe methods for learning the mapping between phase delays and world coordinates without specifying the array geometry and requiring minimal effort from the user. Last, we describe an implementation we have built of such an array using low-cost microphones and show some preliminary results for source localization and speaker-change detection.

1. Introduction

Audio is a critical input modality for wearable computers – in its most obvious form, it allows us to give commands to our systems without using our hands. In addition, we can use it to record conversations, identify people by their voices, and react to cues from the auditory environment. However, to accomplish these tasks well, it is necessary to know who is speaking when, or more generally, where the sound is coming from (i.e., was that a command from the user or an utterance from someone standing nearby?). Next, in order to do speaker identification reliably, we need to know where speaker changes occur, since identification is only possible when information is integrated over contiguous blocks of speech (at least a half-second or so long) from a single speaker. Furthermore, we would prefer not to encumber the user with a headset microphone and at the same time get high-quality audio input from sources other than the user.

As a possible solution to these problems, we propose the notion of a flexible, wearable phased array of microphones. On the hardware side, this simply means having a number of omnidirectional microphones mounted around the body (three microphones in our implementation). On the software side, we can use computationally inexpensive signal processing techniques to estimate the direction of arrival of sounds and also enhance the signal to noise power ratio by N times (where N is the number of microphones). Knowing where the sound is coming from allows us to detect speaker changes and distinguish speech from the users. Signal enhancement allows us to get high-quality input without a close-talking microphone. The computations are simple enough that we can estimate the source direction and do the signal enhancement in real time without taking up a significant fraction of the CPU. While placing an array on the body does provide some challenges not encountered in the traditional arrays with fixed, known geometries, we will show how we can deal with or not worry about most of them.

We have implemented a test version of our concept using three microphones attached to the upper body, with which we have seen that we can reliably determine the direction of incoming sound, detect speaker changes, differentiate the user's speech from other sounds, and enhance the signal quality. In the sections that follow, we will describe how we achieved this state. We begin with the theoretical foundations of direction estimation and signal enhancement, describe our setup and experiments, and finally show some preliminary results from conversations analyzed using the array. While more work is necessary to perfect the mechanisms presented, we believe the basic experiments already show how useful a phased microphone array can be for a wearable device.

2. Background and Methods

The concept of phased arrays is quite old – they have been heavily used and developed since the early days of radar. A guidebook such as [4] gives a complete description of various geometries and methods. However, there are three assumptions typically made in this work that are not satisfied here.

First, the signals dealt with in the radar/antenna community are either of a known form (a pulse that is sent out and modified in parametric ways upon reflection) or within a small frequency range. Second, the signals are coming from a great distance, allowing for a "plane wave" assumption (i.e., the wavefronts appear as straight lines instead of circles because the source is so far away). Third, the array geometry is assumed to be both known and fixed. In our case, we have to deal with an unknown signal (usually speech) that has a very wide spectral range and is often coming from nearby (the user's mouth!). To make the problem even worse, our array geometry is unknown and constantly in motion.

Because speech does not satisfy the usual assumptions, phased arrays have not been widely used for speech processing. There are a number of works that do beamforming (signal enhancement) with speech [2, 3, 5, 1], including [5], who develop an array built into a pair of eyeglasses that do fixed beamforming for a hearing aid. The use of arrays for speech localization has been even more limited – there has been some work by PictureTel and a handful of others such as [3]. Last, as far as we know, there is no other work before ours on building arrays on the body for source localization.

In the remainder of this section, we go through a brief description of the theory behind the two problems of source localization and beamforming and describe the methods and approximations we have used to apply these mechanisms in our work.

2.1. Sound Localization

The basic principle behind source localization with an array is that the sound arrives at the different microphones at different times. With enough microphones in a non-pathological geometry, the mapping between source locations becomes one-to-one, and we can thus estimate the position of the source exactly from only knowing the delays. Typically, we will not have the number of mics nor the geometry to reliably pinpoint the source in 3D, but we will be able to reasonably estimate its orientation in both azimuth and elevation.

The basic tool for determining the delay between two microphones is normalized cross correlation, $n[k]$. It can be expressed as:

$$n[k] = \frac{\sum_{i=1}^{N} s_1[i]s_2[i+k]}{(\sum_{i=1}^{N} s_1^2[i])^2(\sum_{i=1}^{N} s_2^2[i+k])^2} \qquad (1)$$

where $s_1[i]$ and $s_2[i]$ are the raw samples from the microphones. This measure is guaranteed to be inside $[-1, 1]$, and can be interpreted as the cosine of the angle between the two vectors $s_1[i]$ and $s_2[i+k]$. This comes from the definition of the inner product for two vectors a and b:

$$< a, b >= \sum_i a[i]b[i] = ||a||||b||cos\theta \qquad (2)$$

Solving for θ, we achieve the expression above. There are several interesting properties of this quantity. Since $n[k]$ reaches its maximum when s_2 is precisely k elements ahead of s_1, we can use this k as our estimate of the delay:

$$d_{1,2} = \arg\max_k n[k] \qquad (3)$$

Next, since audio has a good amount of low-frequency energy (i.e., slowly varying waves), small shifts will not drastically change the correlation, and thus we expect the signal $n[k]$ to be smooth in k. We can see that it is in figure 1 below.

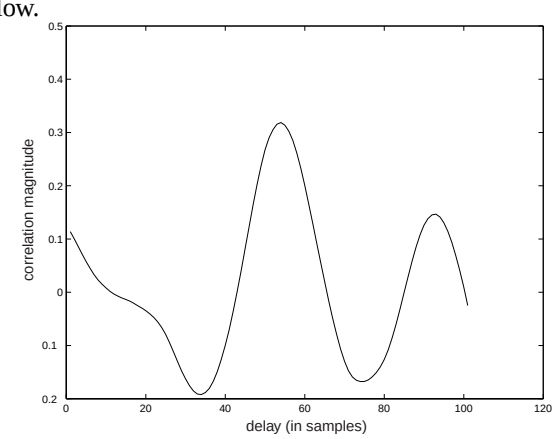

Figure 1. The normalized correlation between two microphone signals near the optimal lag. Note the smoothness due to the typically strong low-frequency content of speech.

The last important property is that when the signals are different at the different microphones, the correlation value will be close to zero. Thus when there is no coherent audio source, even the maximum correlation will be quite low. This acts as an important cue in our algorithm – when the correlation magnitude across all lags falls below a certain threshold, we assume there is no coherent source (or it is overwhelmed by noise) and do not attempt to localize it. (Note that in our more recent work, we have moved beyond a simple threshold and are now using dynamic programming to simultaneously decode source location and whether a coherent source is present.)

Before we continue, there are two important parameters that have been left out of the discussion: the correlation length, N, and the range of the lags k over which we estimate n. The latter is simpler in that a given microphone spacing and sampling rate will determine the minimum and

maximum lag possible. Since the speed of sound in air is approximately 331 meters/second, at a 16 kHz sampling rate sound moves approximately 2 cm per sample. For a spacing of r cm, then, the maximum lag on one side is $-r/2$, and $r/2$ coming from the other direction, yielding a total lag range of $r+1$ samples. Choosing N is more tricky. If we choose too long a window, we will smooth across speaker changes; with too small a window, our estimates will be noisy and unreliable. Based on our experiments, we have found overlapping windows of 1024 samples with a stepsize of 512 samples at 16 kHz are a good compromise, since the window is about 1/16th of a second (it is basically impossible to have utterances shorter than this) and the stepsize gives us a frame rate of about 32 per second. As a result, even a short "yeah" or "uh-huh" gives us at least 10 frames to work with.

The reader may wonder at this point why the difficulty of estimating this delay is even necessary for a wearable, when it seems the voice of the speaker could be easily differentiated based on magnitude (or even differential magnitude with respect to a second microphone). This is not as powerful for a number of reasons. First of all, in order to have the magnitude be greater only for the user, the microphone must basically be placed just in front of the mouth and thus leads to an unwieldy headset-mic solution. Otherwise, a speaker who is standing right in front of the user will appear as loud, or perhaps louder, than the user, as we have found empirically. Next, this method is basically incapable of estimating source direction with any precision, since the energy difference is not significant unless the source is right by one of the microphones.

Returning to our development, we now consider what information we actually have when we detect a phase lag d between two signals. In the 2D version shown in figure 2 below, we see that a constant difference between two fixed points yields one side of a hyperbola (the other side is not relevant since sound only propagates in a positive direction). In 3D, this is a hyperboloid (the hyperbola is rotated around the axis connecting the two mics. If we have three microphones arranged in a plane (as is the case in our implementation) , the surface of constraint from estimating the delay between two pairs (the third pair is redundant) is the intersection of two hyperboloids, which can be quite complex. Fortunately, the hyperboloids asymptotically becomes cones, though the intersection of cones can still be quite complicated to determine analytically (a line, parabola, two ellipses, etc.). Furthermore, given that the spacing of the microphones is on the order of 50 centimeters, we are often working in the region where the asymptotes are poor approximations. Lastly, we cannot even begin to solve for these surfaces if we do not know the geometry of our array, which will be the case with our wearable array.

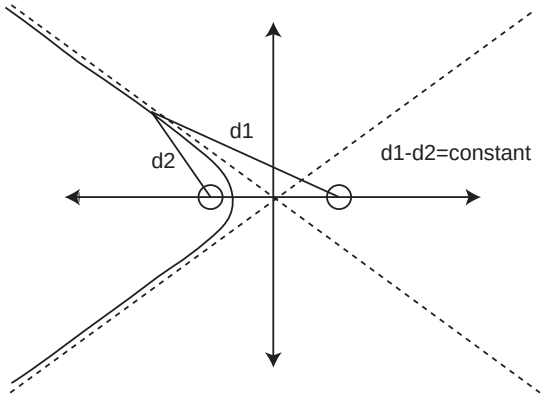

Figure 2. The hyperbola of constraint formed by a known delay between two microphones. The dotted lines represent asymptotes. Note that the constraint only takes one branch of the hyperbola since sound can only propagate in a positive direction.

As a result, we propose to learn the mapping between the two phase delays and the source direction. Note that in most cases, the intersection of the two hyperboloids will result in a parabola in space. Since our microphones will be in a plane in the front of the user's body, this parabola will be going through the user's body at its apex. Asymptotically, this parabola will become a "V." Because the user acts as an acoustic shield, the half of the "V" behind the user is blocked, and thus we can approximate the single asymptote with a 2D source direction (azimuth and elevation). Though this is an approximation, there is a significant advantage to learning the mapping as opposed to defining it explicitly: we can place the microphones anywhere we like. If our clothing for a particular day makes it impossible to use our usual configuration, we can simply move the mics and recalibrate.

To learn a mapping, we first must choose a model. At this stage, we are using the simplest possible – a linear mapping between delay space and 2D orientation, i.e.,

$$[\ \theta \quad \phi\ ]^T = A[\ d_1 \quad d_2\ ]^T \tag{4}$$

Because there is no affine component to this model, we need to make sure the point (0,0) is aligned with the user's body. Since the natural choice for the origin is the point directly in front of the user, we can easily get the data necessary for this by having the user repeatedly snap his fingers in front of his chest. We then use the cross-correlation mechanism to estimate the delay and realign all three signals such that the delay is zero for this orientation. Note that for this entire calibration phase, we use a longer time window (20000 samples) to make sure all of the snaps are captured. Because we

are using multiple sound devices, these delays can be quite different from run to run, so this initial front-snapping must be done every time the system is started up.

At this point, the user steps through a sequence where he snaps to his right, to his left, below his waist, and then speaks a short utterance (all upon cue from the program). We now have four labeled 2D $[d_1 d_2]$ pairs, $[d_1^1 d_2^1]$ through $[d_1^4 d_2^4]$. Writing the i, j element of A as a_{ij}, we can rewrite equation 4 as

$$\begin{bmatrix} d_1^1 & d_2^1 & 0 & 0 \\ 0 & 0 & d_1^1 & d_2^1 \\ d_1^2 & d_2^2 & 0 & 0 \\ 0 & 0 & d_1^2 & d_2^2 \\ \vdots & \vdots & \vdots & \vdots \end{bmatrix} \begin{bmatrix} a_{11} \\ a_{12} \\ a_{21} \\ a_{22} \end{bmatrix} = \begin{bmatrix} \theta^1 \\ \phi^1 \\ \theta^2 \\ \phi^2 \\ \vdots \end{bmatrix} \tag{5}$$

If we write this equation as $Da = b$, we can easily find the least-squares solution for a as

$$\hat{a} = (D^T D)^{-1} D^T b \tag{6}$$

Empirically, this mechanism has worked quite well for mapping the delays to an actual source direction. We are currently conducting experiments to quantitatively test the accuracy of this mapping under various degrees of deformation of the array. Note that since the microphones will stay in the same place on the user's wearable, we do not need to relearn A from run to run – we only need to recalibrate the overall phase between the microphones with the front-snapping.

An important issue which we haven't discussed thus far is the actual placement of the microphones. As we mentioned before, a pair spacing of r cm results in $r + 1$ samples of resolution along that pair (at 16 kHz). There is also a constraint on the maximum separation – the further apart the microphones get, the more periods of a signal can exist between the microphones. If the correlation range is $r + 1$ samples, the period of a harmonic sound must be greater than this. Otherwise, it will have more than one peak within this space and will cause aliasing in the phase delay (there will now be more than one correlation peak of the same height within the delay range). This lower bound on period length translates to an upper bound on frequency. We can express this as:

$$f_{max} = \frac{f_s}{c_r} \tag{7}$$

where f_s is the sampling frequency and c_r is the number of samples in the correlation range $(r + 1)$. Furthermore, if we examine where the c_r comes from,

$$c_r = \frac{2l}{v_s} f_s + 1 \tag{8}$$

where l is the separation of the microphones (in meters) and v_s is the velocity of sound in air. The factor of two and the one appear because the lag can go from a minimum of $-l/v_s$ to a maximum of l/v_s samples: although the propagation delay along the line between the microphones is only l/v_s samples, the sound could be coming from either direction. For periodic sounds, these effects go away, since the signal is repeated in both directions. As a result, we can simply write c_r as $\frac{l}{v_s}$. When we combine this with equation 7, we see that f_{max} is independent of the sampling frequency:

$$f_{max} = \frac{v_s}{l} \tag{9}$$

For our maximum spacing of 50cm and 16kHz sampling frequency, this corresponds to a maximum frequency of 331 Hz. To prevent aliasing, we thus have to first lowpass filter the signals at this frequency. This does not typically hurt our performance for speech localization, since the adult human voice typically falls within the 80 Hz to 400 Hz range, with few speakers over 300Hz. Localizing higher pitched sounds (children, singing, etc.), however, would not be possible with this spacing. If we wish to increase this frequency range, we need to decrease the microphone spacing, which will reduce our correlation range c_r (equation 8). However, increasing the sampling frequency will increase the correlation range again *without* affecting f_{max}. As a result, we can have both an arbitarily high cutoff frequency and an arbitrarily large correlation range if we can bring the mics arbitrarily close together (which we wish to do anyway) and sample at an arbitrarily high rate. The only downside of the latter is the increased computational load.

A bigger problem is that we would like to keep the array in a reasonably fixed geometry with respect to the body so that we can hope to apply the mapping learned above. Unfortunately, the size of the body is limited, and the size of the part of the body that is fixed (the torso) is smaller yet. We maximized the use of this space by placing the microphones on the shoulders and near the beltline as shown in figure 3 below. Because the shoulder mics are only about 25 centimeters or so apart, we do not get very good resolution in the horizontal direction. However, as our experiments show, the resolution is sufficient to separate two other speakers in a conversation who are within 60 degrees of each other. Because the user's mouth is at maximal phase with respect to one of the pairs, it is quite easy to pick out the speech from the user.

Another issue which we have already touched on is the fact that the array is always in motion in two senses. First

of all, the microphones are moving relative to each due to the user shifting his shoulders, his clothing moving around, etc. Secondly, the array is moving as a whole when the user turns his body to face a different speaker, when he is walking, etc. The first issue is not too much of a problem – the relative motion of the microphones rarely seemed to outweigh the noise in the delay estimation, which is at least one or two samples. This is due to the fact that the sound is moving at 2 cm per sample, so even if the microphone moves within a radius of 4 cm, the delay will only shift by a maximum of 2 samples. However, the motion of the array is a whole is more vexing – it becomes quite difficult to identify different speakers by their location if their location signal keeps changing.

As a result, we cannot depend on the phase to identify a speaker and must depend on speaker identification techniques. As we mentioned earlier, such techniques require long (a half-second or more) chunks of contiguous speech in order to perform well, since they can only provide reliable information when integrating log-likelihoods over a good number of samples. Otherwise, the overlap in features from speaker to speaker makes classification impossible. In other words, for speaker ID to work, it needs a signal that tells it when a speaker change occurs. We can provide precisely this cue from our system by looking for "jumps" in the source direction. In order to find these jumps, we look for a change in the mean over a sliding window of 80 frames (40 on each side) – if the mean on one side of the window differs from the mean on the other side by a threshold, we mark a potential speaker change. Figures illustrating this process are shown in the experiments section below. We are currently developing a more accurate means of tracking changes which we will give a preliminary overview and demonstration of in our results section.

Note that we should be able to use speaker ID methods even if we do not know the speakers *a priori*. When we find a contiguous chunk of a sufficient length, we could build a new model for it. When the next chunk appears, we could then compare it to all the models we have from before. If it matches one of the previous models, we can incorporate it into the model; otherwise, we could begin a new model.

There are some remaining difficulties in the source localization problem. The most significant among these is that of reflections. When the user is standing close to a hard surface such as a whiteboard, each signal hits the microphones twice – once from the source, and once from the reflection. As a result, two distinct (and often equal) phase peaks appear. There are several simple solutions to this problem. Since the distance to the source is always shorter than the reflected distance, the first occurence (i.e., the peak closer to the zero lag) is always the actual source. As long as the dual nature of the peak can be detected, this issue can be resolved. We hope to incorporate this capability in a later version of our system. Thus far, we have only encountered one situation when this was a problem (the user was right next to a whiteboard) and expect it will not occur that often.

2.2. Beamforming

The concept behind beamforming (signal enhancement with an array) is quite simple. The basic idea is to align the signals and add them up. Since we have already found the delays between the signals for the source localization methods above, this is straightforward for us to do. The remaining question is how much it actually buys us.

Let us assume we have already aligned the signals, so the three inputs to be summed are $s_1 = s + n_1$, $s_2 = s + n_2$, and $s_3 = s + n_3$. In general, the noise signals n will be uncorrelated with each other – basically, it is the sum of the air moving around the microphones and all the other sources in the room, which are all coming in at different phases from the source, and are thus uncorrelated with each other when shifted by delay for the source direction. Of course, if the noise sources are directional and of low enough frequency, they will not be completely uncorrelated from each other despite the phase shift, but will be less correlated than the source itself. To simplify the calculation, though, let us consider the typical case of completely uncorrelated noise. In that case, the sum of the signals is

$$3s + n_1 + n_2 + n_3 \tag{10}$$

When a signal is multiplied by a constant, its variance (power) goes up as the square of the constant. However, when uncorrelated variables are summed, their variances simply add. Thus, if the noise signals are all of the same power, the power of the signal is going up by a factor of nine while the power of the noise is only going up by a factor of three. In general, with uncorrelated noise signals, we see this n-fold increase in signal-to-noise power when we combine n signals.

3. Experimental Setup

For the experiments in the next section, we put together our first version of a wearable phased array shown in figure 3. The microphones are low-power omnidirectional Gentek elements (approximately $12 apiece) mounted in rubber capsules and attached to the vest with velcro. The circuit boards they are attached to come from Telex M-560 USB microphones (approximately $60 apiece) – we simply removed their original microphone elements and replaced them with

the Gentek elements. We did this because of the omnidirectional magnitude response and extremely flat phase response as opposed to the directional element shipped with the Telex. We originally chose the USB microphone path because of its low cost and its practicality for wearables. Also, they are low-enough power that they can be used with tiny USB hubs that do not require external power. Furthermore, three microphones at 16 kHz only occupy 0.77 Mbits of bandwidth, less than a tenth of the 7-12 Mbit range of USB buses.

Unfortunately, there are a number of aspects to the current performance of USB microphones that have driven us away from this solution in our recent work. The first issue is with the Telex USB converters themselves, which we were aware of at the time of submission. These devices do not produce data at precisely 16 kHz, but instead at rates "close" to 16 kHz – most likely the result of using cheap oscillators. Since this is different for each microphone, it causes a slow drift between microphones which becomes noticeable after a few seconds of data. However, it is not difficult to compensate for these rates (as we have done in the paper) if they are constant. However, since the time of the submission, we have discovered that these rates are temperature sensitive as well, which makes it necessary to calibrate these rates on every run. Worst of all, very recently we have found in longer term experiments that the microphones will drop samples – often 20 samples at a time. While this is not a significant effect for single microphone applications, it is fatal for delay estimation, since the phase delay between the microphones is now changing arbitrarily. As a result, we have moved away from USB audio and are putting together a new system using a small laptop's internal sound card (stereo line-in) and an additional PC sound card, allowing for a total of four input channels.

For the experiments in this paper, the rig was worn by a user, but the microphones were plugged into a desktop machine (a dual-processor 600 Mhz Pentium III running Windows 2000). The phase-estimation code only consumed 8% of the total CPU on this machine, and on separate tests consumed about 25% of the CPU on the Sony Picturebook (300 MHz Pentium I) which we are using as the wearable platform for this application. The relatively low computational load is due to our heavy usage of the Intel MMX/SIMD operations through the Intel Performance Libraries, which optimizes operations such as normalized correlation.

4. Experiments and Results

For the experiments, a user wore the vest and participated in three conversations, each involving three people (including the user). The conversations varied in length from two minutes to twelve minutes, and the estimation and

Figure 3. Initial implementation of wearable phased array.

sampling rate compensation mechanisms worked reliably throughout these intervals. This was an unrestricted conversation and thus was filled with interruptions and small interjections (such as "uh-huh" and "yeah"). The system catches about 70% of all speaker changes (if we include the interjections as speaker changes) and 85% of long-segment changes (segments longer than 1 second). Only 5% of the reported changes were false alarms. We can see some examples of this in the two minutes of speech shown in the sequence below. Figure 4 shows the phase estimates for frames where the peak correlation energy was greater than 0.5.

Figure 5 shows the noisy lag signals plotted against each other. The three distinct clusters for the three speakers are clear. Finally, Figure 6 shows the system's estimates of speaker change locations (vertical lines) superimposed on the raw lag values.

This is a fairly typical example from our data – the three speakers were separated reliably, and regions of speech vs. silence (via the correlation magnitude) were also accurately identified. There is one error at the 1150 mark, where there is another speaker change (which lasted for 1 second), but the mean-window approach we are currently using often smooths over such very short speaker segments.

While the mean-window approach is very simple to compute, this smoothing effect is not acceptable if we want to capture short-duration speaker changes. As a result, we have been developing a more sophisticated algorithm based on dynamic programming that uses the history of raw correlation values at each possible phase delay to determine

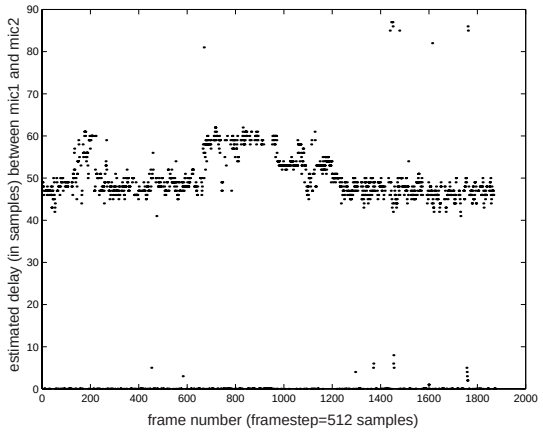

Figure 4. Raw lag values between two microphones.

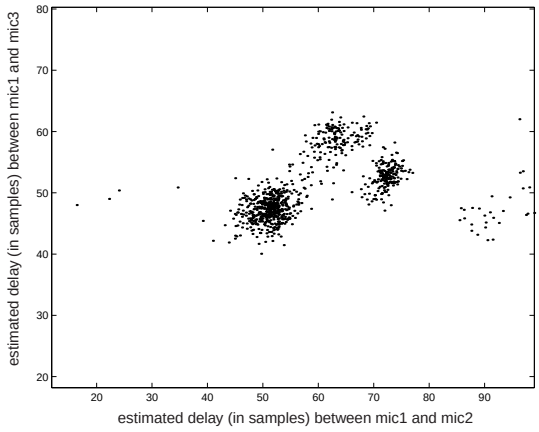

Figure 5. The two lag values plotted against each other. Note the clear emergence of the three clusters for the three speakers.

the most likely path between sound positions and silences. Though we have only tested this on a small amount of data, it seems to be far more effective, capturing more than 95% of speaker changes (including interjections), and is capable of reliably separating speakers that are much closer together (only 30 degrees apart). An example of this algorithm's performance is shown in figure 7. The dotted line shows the decoded path, and the jumps in the line signify speaker changes. This segment corresponds to 10 seconds of speech with 7 speaker changes, all of which are completely caught by the algorithm. The short utterances include laughter and a 1/3 second interjection. We will present the details of this algorithm in a later paper.

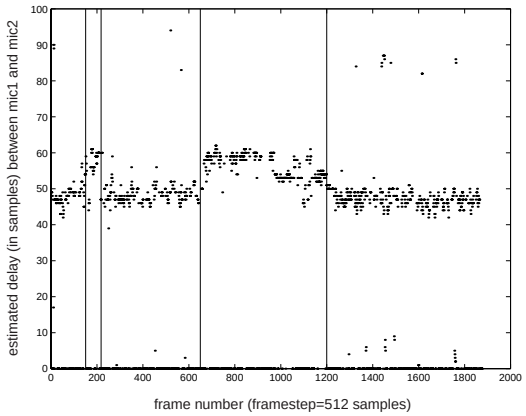

Figure 6. Speaker change detections using the original mean-window algorithm overlaid on the raw lag values. All are correct; however, the system missed a speaker change at 1150 due to the brevity of the utterance (1 second).

5. Future Work

There are variety of directions in which we wish to extend this work. First and foremost, we wish to continue work on the new tracking algorithm and formalize the preliminary results by running more tests on labeled data. In addition, we have a number of ideas for further improvements and extensions to our methods. The first and simplest is to increase the sampling rate. As we showed earlier, this will allow us to bring the microphones closer together (and as a result increase the cutoff frequency) and increase the correlation range. If we sample all three microphones at 44.1kHz, we only consume 2.1Mbits of bandwidth. This would increase the processing load as well (since we would be using longer windows), but we could increase the stepsize between frames and sacrifice time resolution for phase accuracy at a constant computational load.

Other obvious extensions include implementing solutions to the multipath problem, trying nonlinear models to map between the phase delays and source directions, developing the speaker ID component, and building a user interface to browse the conversational data. A final project we are interested in pursuing is integrating a wide-angle wearable camera with this work. We could then learn the mapping between pixels in the camera and phase delays. When we detect speech from a particular source direction, we could then use flesh-finding techniques (as in [6]) to find the face associated with the sound. This could be a significant aid to the speaker recognition task. In addition, we could automatically take pictures of sound sources as we walked by

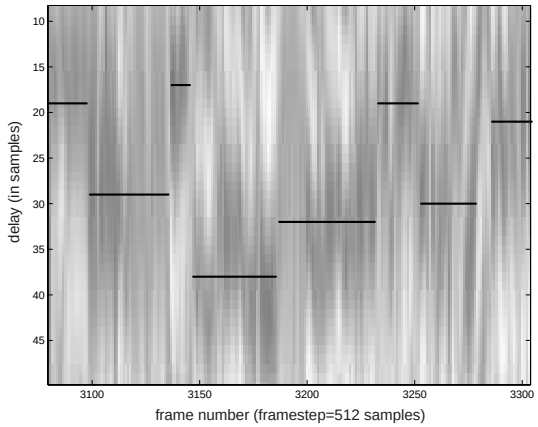

Figure 7. Speaker change detections using the new dynamic programming algorithm overlaid on the raw cross-correlogram. Darker values on the cross-correlogram signify higher correlation values. The contiguous line segments correspond to different speakers. Ten seconds of speech are shown; all seven speaker changes are caught by the algorithm, including a 1/3 second interjection.

them, thus using source direction as an interesting means of attentive foveation for our wearables.

6. Conclusions

We have presented the idea of a flexible phased array for wearable computers. We have shown how such an array can be used to estimate the direction of the sound source and enhance the signal-to-noise ratio of the sound – solutions to two problems that are fundamental for unencumbered audio input and auditory scene analysis for wearables. We have shown how this array can be placed arbitrarily on the body, how it can be calibrated with respect to the user's body with minimal user effort (a 15-second snapping session), and how it can be effective despite small changes in the array geometry due to user motion. We have built and described a test implementation of the wearable array with low-cost microphones and have successfully used it to detect speaker changes in a conversational setting. While there is a clear need for a more formal set of experiments to fully assess and optimize the performance of this system, we feel it is already clear that a phased array can be a very powerful mechanism for audio input and auditory analysis of the environment for wearable computers.

Acknowledgements: Many thanks to the reviewers for their detailed and insightful comments.

References

[1] S. Basu, M. Casey, W. Gardner, A. Azarbayejani, and A. Pentland. "Vision-Steered Audio for Interactive Environments". In *Proceedings of IMAGE'COM*, Bordeaux, France, May 1996.

[2] H. Cox. Robust adaptive beamforming. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 35(10):1365–1376, 1987.

[3] F. Khalil, J. Jullien, and A. Gilloire. Microphone array for sound pickup in teleconference systems. *Journal of the Audio Engineering Society*, 42(9):691–699, 1994.

[4] P. Mailloux. *Phased Array Antenna Handbook*. Artech House, Boston, 1994.

[5] R. Stadler and W. Rabinowitz. On the potential of fixed arrays for hearing aids. *Journal of the Acoustical Society of America*, 94(3):1332–1342, 1993.

[6] C. Wren, A. Azarbayejani, T. Darrell, and A. Pentland. Pfinder: Real-time tracking of the human body. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 19(7):780–785, July 1997.