Shout COOEE!
SP Cup Submission

Antoni Dimitriadis, Alvin Wong, Prasanth Parasu, Qingbei Cheng, Jennifer Jingyao Wu
School of Electrical Engineering and Telecommunications
The University of New South Wales
Sydney, Australia

***Abstract*—In this paper we present our team's solution to the problem outlined for the 2019 Signal Processing Cup, Drone-Embedded Sound Source Localization. We have analysed and identified the issues present in the baseline GCC-NONLIN system, and aim to build on this method by combating these weaknesses. This includes the implementation of a method of reducing noise at the input stage by exploiting our knowledge of it's characteristics, and the use of the Viterbi algorithm to assist with dynamic tracking. We outline our method in this paper, and demonstrate it's simulated performance on data that we created for this task. The results demonstrate the improvements made over the baseline system.**

I. Introduction

We are team 'Shout COOEE!', representing the University of New South Wales, Australia.

While sound source localization is a long-standing area of research in signal processing, the unique challenges introduced by UAV search and rescue represent the core of this year's SP Cup Challenge. The primary issue identified in the task syllabus (and evident in the recordings provided), is the dominant acoustic noise generated by interfering sources including that of the drone motors (highly harmonic and a function of motor speed), wind noise generated by the drone's movement and the propeller rotations, and other stationary structural noise.

For these reasons, the blind use of pre-existing methods will struggle in many reasonable use cases. [1] evaluates some of these methods and demonstrates the weaknesses in the case of speech sources at low SNR in the dynamic case. The MBSS_locate baseline system provided for the task contains implementations for multiple of these pre-existing methods, each of which employs a different technique to construct an angular spectrum, with peaks indicating most likely source locations. These methods are all introduced and described in [2], where it is also mentioned that a variant of the GCC-PHAT (generalised cross correlation with phase transform) shows the best performance for microphone arrays with spacing comparable to what is present in the drone for this task. This variant is referred to as GCC-NONLIN and provides the basis for our proposed method.

In order to improve upon this base method, we seek to exploit all additional information available to us. In this static task, this is effectively limited to the information we have on the nature of the noise generated by the drone itself, in the form of recordings of the individual rotors at various speeds. In the dynamic task, we exploit the fact that adjacent estimate frames are not independent, and the drone is unlikely to move large distances. This additional information allows us to improve upon the GCC-NONLIN baseline.

The rest of this paper is structured as follows. In Section II we review the GCC-NONLIN baseline method and introduce necessary notation. In Section III we introduce the proposed method. We discuss our evaluation strategies and data simulation in Section IV. In Section V we present our simulation results and draw relevant conclusions in Section VI.

II. The Baseline Method

As mentioned in the previous section, the baseline MBSS_locate system provided for the challenge contains multiple methods doing sound source localisation via computation of an angular spectrum. The best performing of these methods is GCC-NONLIN, and so it will be the focus of our analysis.

A. GCC-NONLIN and the Angular Spectrum

We will give a brief review of the GCC-NONLIN based construction of the angular spectrum, as described in [2].

Given a pair of microphones M_i, M_j with coordinates $\mathbf{m_i}, \mathbf{m_j}$ and input signals $\mathbf{x_i}, \mathbf{x_j}$ respectively, we define a pairwise angular spectrum $\psi_{\mathbf{m_i},\mathbf{m_j}} : \Theta \times \Phi \to [0, 1]$, where Θ & Φ represent the set of azimuthal angles and the set of elevations respectively. The function is evaluated as

$$\psi_{i,j}(\theta,\phi) = \max_t \sum_f \rho\left(\Re\left(\frac{\mathbf{X}_i(t,f)\overline{\mathbf{X}_j(t,f)}}{|\mathbf{X}_i(t,f)\overline{\mathbf{X}_j(t,f)}|}e^{-j2\pi f\tau_{i,j}(\theta,\phi)}\right)\right)$$

where $\mathbf{X}_i(t,f)$ is a time-frequency representation of $\mathbf{x_i}$, $\overline{z}$ is the complex conjugate of z, $\Re(z)$ is the real part of z, $\rho(u) = 1 - \tanh(\alpha\sqrt{1-u})$, and $\tau_{i,j}(\theta,\phi)$ is a function that relates the source position to the time delay of arrival at the two microphones for a wave propagating from that direction. This takes the following form:

$$P(\theta,\phi) = (\cos(\phi)\cos(\theta), \cos(\phi)\sin(\theta), \sin(\phi))$$
$$\tau_{i,j}(\theta,\phi) = \frac{(\mathbf{m_i} - \mathbf{m_j}) \cdot P(\theta,\phi)}{c \times |P(\theta,\phi)|}$$

We sum this over all pairs of microphones to construct a global angular spectrum in a time frame t as

$$\mathbf{\Psi}_t(\theta,\phi) = \sum_{i<j} \psi_{i,j}(\theta,\phi) \tag{1}$$

III. Proposed Method

We see little value in attempting to create another method of computing an angular spectrum, and so instead we focus on the baselines shortcomings. These are primarily the need for input noise filtering, and the fact that it is a single-snapshot system (it does not use the dependencies between estimate frames). We provide two additions to the algorithm in order to combat these issues. More specifically, we use Wiener filtering on our inputs to reduce the impact of the ego-noise, and a modified implementation of the Viterbi algorithm [3], [4] for tracking in the dynamic task.

A. Noise Reduction & Wiener Filtering

The first adaption to the baseline GCC-NONLIN method required is the use of an input noise reduction technique to help combat the dominant acoustic noise previously mentioned. It is mentioned in [1] that when present, the wind noise is by far the loudest interfering source and is almost entirely present below 1kHz. Without much data to analyse the wind noise, we found that simply applying a 1kHz cut-off high pass filter made a large difference to the broadband source examples. This is primarily because doing so does not make a large difference to amount of source information provided, given that the source occupies frequencies reaching the ultrasonic range.

This method however is not at all viable in the cases where the source is speech as too much speech energy is lost in the process. In this scenario, the best we can do is use a Wiener filter with an estimate of the noise covariance matrix based on the real-time rotor speed measurements and the ego-noise recordings provided for the challenge. We provided an ad hoc method of constructing the noise covariance matrix that is given as follows:

1) For each $i = 1, \ldots, 4$
   a) Check current speed v_i for motor i.
   b) Compute nearest upper and lower speeds as $v_{i,l} = 10\lfloor v_i/10 \rfloor$, $v_{i,u} = v_{i,l} + 10$.
   c) Compute upper and upper weights as $w_u = (v_i - v_{i,l})/10$, $w_l = 1 - w_u$.
   d) Extract $\mathbf{n}_{i,l}$ as 10 seconds of audio from recording of motor i at speed $v_{i,l}$, and similarly $\mathbf{n}_{i,u}$.
   e) Combine to give composite noise sample as $\mathbf{n}_i = w_u \mathbf{n}_{i,u} + w_l \mathbf{n}_{i,l}$
2) Combine noise samples as $\mathbf{n} = \sum_i \mathbf{n}_i$
3) Construct a time frequency representation N of $\mathbf{n}$, and compute the 8 by 8 empirical covariance matrix for each frequency bin. This is done using MBSS_covariance.

This covariance matrix is then used as the input to MBSS_wiener to attenuate the ego-noise from the microphone signals.

B. Viterbi Algorithm for multiple-frame Tracking

For the dynamic case, we implement a slightly modified version of the Viterbi algorithm, originally proposed in [3] and generalised in [4], in order to compute a Viterbi Path (with respect to certain cost functions). We now introduce our slightly modified version of the algorithm.

Let T be the total number of observation frames, and K be the number of possible states per frame. Then we say $s_{k,t}$ is the kth state of the tth frame, where $1 \leq t \leq T, 1 \leq k \leq K$. Let S_t be the set of the K possible states in frame t. We define an emission function $\varepsilon_t : S_t \to [0, 1]$, and a transition function $\tau_t : (S_{t-1} \times S_t) \to [0, 1]$. The algorithm aims to select the sequence of states $(x_t)_{t=1}^T$, with each $x_t \in \{s_{k,t}\}_{k=1}^K$, that maximises the product

$$\prod_{t=1}^{T} \varepsilon_t(x_t)\tau_t(x_{t-1}, x_t)$$

To this end we define two $K \times T$ matrices, C (the Cost matrix) and P (the path matrix). $C_{i,j}$ stores the the maximal product

$$\prod_{t=1}^{j} \varepsilon_t(x_t)\tau_t(x_{t-1}, x_t)$$

with the added condition that $x_j = s_{i,j}$. $P_{i,j}$ stores the state x_{j-1} that generated the maximal sum in $C_{i,j}$. These matrices are populated column-by-column, according to the following two update equations:

$$C_{i,j} = \max_k \left[C_{k,j-1}\varepsilon_j(s_{i,j})\tau_j(s_{k,j-1}, s_{i,j})\right] \tag{2}$$

$$P_{i,j} = \operatorname*{argmax}_k \left[C_{k,j-1}\varepsilon_j(s_{i,j})\tau_j(s_{k,j-1}, s_{i,j})\right] \tag{3}$$

Also note that $\tau_1(s_{k,0}, s_{i,1}) = 1$ for all i since the states are undefined for $t = 0$. The algorithm runs until it finishes populating the two matrices, at which point there will be an element $C_{\eta,T}$ in the Tth column of C that contains the complete maximal product. From here we assign $x_T = s_{\eta,T}$ and construct our Viterbi path as follows:

Initialise $k = \eta$, $t = T$. Loop through the following steps until $t = 1$

1) Set $k_{\text{prev}} = P_{k,t}$
2) Set $x_{t-1} = s_{k_{\text{prev}},t-1}$
3) Update $t = t - 1$, $k = k_{\text{prev}}$

This process backtracks through the columns of P and assign the Viterbi path to the $(x_t)_{t=1}^T$ in reverse order.

The motivation for implementing this algorithm for the in-flight dynamic task was to stop the estimates from erratically jumping due to spurious peaks in $\mathbf{\Psi}_t$. We know that large angular displacements between frames are highly unlikely or impossible potentially impossible, as the drone cannot travel this far in between frames. The by penalising large transitions, we should reduce the possibility of this occurring. This gives rise to some assignments.

We let $T = 15$, since there are 15 frames per file and we wish to produce 1 estimate per frame. We tuned K over all the data we had available and found $K = 8$ to be best. Our states are angular coordinates representing an estimated location; that is $s_{k,t} = (\theta, \phi)_{k,t}$. Our emission function $\epsilon_t = \mathbf{\Psi}_t$, the normalised, pairwise GCC-NONLIN global angular spectrum for frame t, from Equation 1. The K possible states

for a frame t are chosen as the coordinates that generate the K most dominant peaks of $\mathbf{\Psi}_t$. Finally, our transition function $\tau_t = \tau$ and is defined as follows:

$$\tau(\theta_1, \phi_1, \theta_2, \phi_2) = \begin{cases} e^{-\frac{d^2}{2\sigma^2}} & d < d_{\max} \\ \epsilon & \text{otherwise} \end{cases}$$

where ϵ is a very small constant, and $d = d(\theta_1, \phi_1, \theta_2, \phi_2)$ is the great-circle distance between the spherical coordinates $(\theta_1, \phi_1, 1)$ & $(\theta_2, \phi_2, 1)$. $d_{\max}$ and σ are adjustable parameters, but some rudimentary tuning lead to the assignments $d_{\max} = d(0, 0, 40, 0)$ and $\sigma = 0.2$ respectively. The transition function output is depicted in Figure 1.

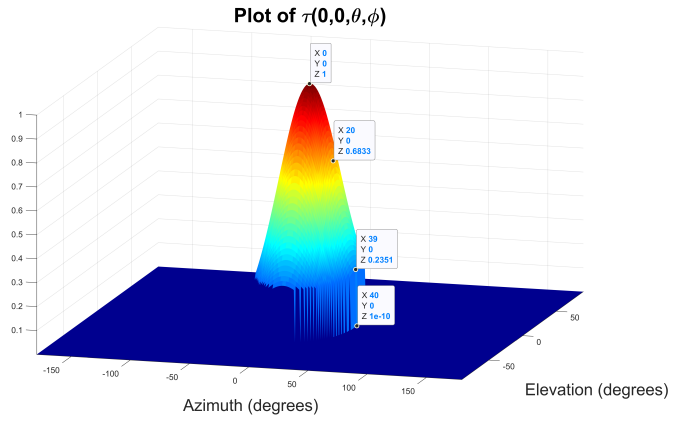

Fig. 1. Surf plot of the transition map τ with $\sigma = 0.02$ and $d_{\max} = d(0, 0, 40, 0)$.

C. COOEE (Proposed)

We will refer to the proposed method as COOEE from here on. The full method is displayed in a block diagram in fig. 2. It is comprised of the Wiener filtering input stage followed by the GCC-NONLIN baseline spectrum computation for each frame, and finished off with multi-frame tracking via the Viterbi algorithm.

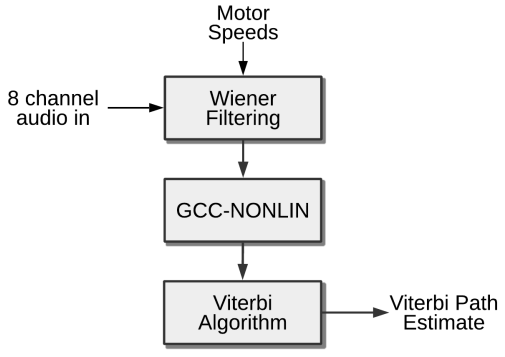

Fig. 2. A block diagram of COOEE (Proposed)

IV. Evaluation Strategies

One of the biggest limitations in the challenge was the lack of development data provided to us for the use of evaluating our modifications and tuning parameters. The provided development data consisted of the following: noiseless static broadband and speech, noiseless flight broadband, and individual motor recordings. This led to the need for us to develop a way of assessing performance in another way. As such we decided to simulate and record data to expand our development data set to achieve a reasonable performance estimate when comparing different methods. This involved simulating sources and mixing motor recordings at varying SNRs to source files for wider and robust testing. There were also attempts made to take recordings with a live drone to obtain robust testing data for the flight task.

A. Simulating Data

For each simulated file, a source's coordinates (θ, ϕ, d) is randomised within the bounds, $-180 \leq \theta \leq 179$, $-45 \leq \phi \leq 45$, and $100 \leq d \leq 1000$, where d is in centimetres. The randomised source coordinates are used to calculate the time delay in each channel as follows:

$$t_i = \frac{|\mathbf{p} - \mathbf{m_i}|}{c}$$

where $\mathbf{p}$ is the randomised source's Cartesian coordinates, c is the speed of sound, and $\mathbf{m_i}$ is the coordinates f the ith microphone as in section III.

Given the 8 delays and a mono input signal $\mathbf{x}$, we can simulate a directional source by appropriately delaying the mono signal on each channel. These delays will not typically be integers, and can either be implemented with a shifting filter (windowed sinc for best accuracy), or with the use of upsampling as follows:

1) Up-sample mono signal $\mathbf{x}$ by a factor of k, to get $\mathbf{x}_k$.
2) Set the ith output channel $\mathbf{y}_{k,i}$ as $\mathbf{x}_k$ delayed by $\text{round}(t_i k f_s)$ samples.
3) Down-sample $\mathbf{y}_{k,i}$ by a factor of k to get back $\mathbf{y}_i$, the ith channel of the simulated signal.
4) Trim front and back of each output channel to get rid of extra zeros and make them all the same length.

This process allows shifting with an approximate sub-sample accuracy of $1/k$. For our data creation, we used $k = 16$.

Given four individual motor recordings n_1, n_2, n_3, n_4 at some specified speed s, the beginning and end silences are removed, and a randomised section is cut from each recording. These sections are mixed into one motor noise recording $\mathbf{n}$. The motor noise is amplified or attenuated to achieve some desired signal-to-noise ratio (SNR) given some audio signal x using the following equation:

$$\mathbf{n}_{SNR} = \frac{\mathbf{n}}{n_{rms}} * \frac{x_{rms}}{10^{\frac{\text{SNR}}{20}}}$$

Using the above we were able to simulate the following test cases: noisy static broadband, noisy static speech, and noisy flight broadband. It is important to note that the flight broadband was mixed with stationary noise, and therefore does not represent one of the most challenging aspects of that test case.

B. Recording Data

As was mentioned in the bonus task report and demonstrated in the video, we made recordings from a live drone with ground truth measurements taken using a motion capture system. Although this was not required for the bonus task, we originally intended to use the data to assist in the evaluation and tuning of our method. Although the pure noise recordings taken on the drone were fine, we encountered a few issues with the recordings that were taken in the presence of a source. The main problem was that many of the recordings suffered from clipping, and due to time constraints we were unable to fix these issues and organise re-recordings with the equipment required. For this reason, the data was not used for evaluations.

V. Simulation Results

Using the simulated data mentioned in the previous section, we have evaluated the performance of COOEE against the GCC-NONLIN baseline. We will not display the results for the noisy static broadband tests since they are not in the test set and both methods performed near perfectly.

The first test condition we will show is simulated noisy static speech. This may not be perfectly representative of the true test scenario, as there will be little wind noise present due to the nature of the motor noise files, and we also have recordings of the motors at the exact speed to initialise our wiener filter. It still provides a method of comparison against the GCC-NONLIN baseline however. The results are shown in fig. 3. The test consists of 20 simulated files at each SNR ranging from -20dB to 0dB.

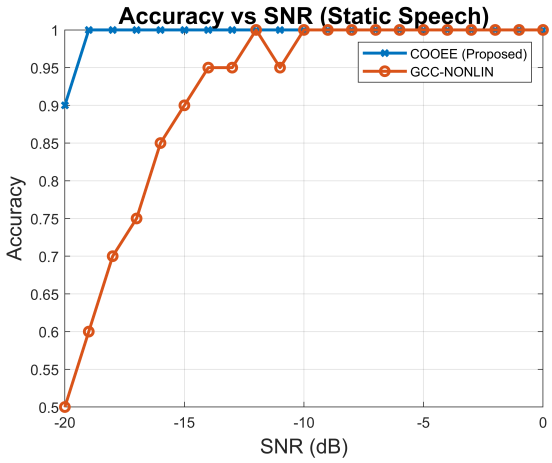

Fig. 3. Comparison of performance on simulated static speech with 20 files per SNR

Finally, we will show our results for simulated noisy dynamic broadband. Once again, this will not be the best representation of the final test scenario as we do not have the ability to simulate dynamic noise or wind noise. The results are shown in fig. 4. The test consists of 5 files per SNR, each 4 seconds long with 15 estimate frames, with SNRs ranging from -20dB to 0dB.

We do not expect these results to be an accurate representation of our algorithms performance on true test data for the reasons mentioned above. That being said, it definitely

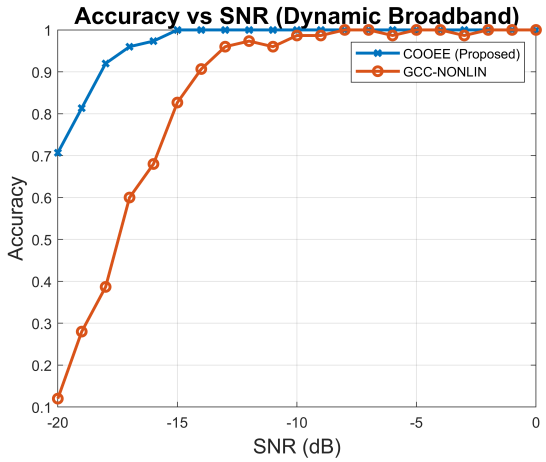

Fig. 4. Comparison of performance on simulated dynamic broadband with 5 files per SNR and 15 estimates per file

demonstrates a clear improvement over the baseline GCC-NONLIN method, especially in the low SNR scenario.

The main avenue for improvement that we could see from running these tests comes from observing the angular spectrum and visually attempting to judge the location of the source. The emission cost function in our Viterbi implementation only assigns cost based on the height of the peak, similar to the way the baseline itself assigns priority, when in reality this cost should likely be using more local spatial information. This could perhaps involve a measurement of the 'width' of the peak, or a measure of how much energy is in surrounding points. This is a weakness present in both the baseline and in COOEE and results in an inability to tell the difference between the shape of the peaks.

VI. Conclusions

The results demonstrate that the proposed method is superior to the baseline in all the simulations. Although the simulations lack the non-stationary noise qualities that can cause issues in real recordings, there is still a clear performance improvement. Given more time and resources, the main avenues for improvement would involve a new heuristic for estimating noise covariance at un-observed motor speeds, and a new local cost function for the Viterbi algorithm. We would also obtain new recordings with ground truths as was originally intended without the clipping errors and use these to further tune the algorithm.

References

[1] M. Strauss, P. Mordel, V. Miguet, and A. Deleforge, "DREGON: Dataset and Methods for UAV-Embedded Sound Source Localization." Institute of Electrical and Electronics Engineers Inc., 2018, pp. 5735–5742.

[2] C. Blandin, A. Ozerov, and E. Vincent, "Multi-source TDOA estimation in reverberant audio using angular spectra and clustering," *Signal Processing*, vol. 92, pp. 1950–1960, 2012.

[3] A. Viterbi, "Error bounds for convolutional codes and an asymptotically optimum decoding algorithm," *IEEE Transactions on Information Theory*, vol. 13, pp. 260–269, 1967.

[4] G. Forney, "The viterbi algorithm," *Proceedings of the IEEE*, vol. 61, pp. 260–269, 1973.