P&D Information Systems and Signal Processing

Week 3 - February 24-28, 2020
DOA-informed Beamforming

Giuliano Bernardi, Randall Ali, Santiago Ruiz [1], Marc Moonen

Version 1.0

Part 1: Preliminaries

1. Modify the file `create_micsigs.m` by adding the following functionality:

   - Add spatially uncorrelated white microphone noise to each microphone signal with a power of 10% of the target audio source signal power in the **first** microphone. When using speech signals, compute the target audio source signal power only over time segments in which the speech source is active, identified by means of the MATLAB command

     `VAD=abs(speech(:,1))>std(speech(:,1))*1e-3;` (1)

     and use the command `speech(VAD==1,1)` to select the active speech samples in the first column of the variable `speech`. Recall that the power of a signal can be obtained through the command `var`.
   - Make sure that the target audio source and the noise component in the microphone signals are separately available as two matrix variables, e.g. `speech` and `noise`, so that the actual microphone signals can then be computed as `mic=speech+noise`.
   - Compute the SNR in the first microphone signal, using the power of the signals in the first column of `speech` and `noise`, P_s and P_n, respectively. Recall that the SNR is evaluated by means of the operator $10\log_{10}(\cdot)$ when the argument is a power ratio.

     Important: For all experiments in Week 3, set the sampling frequency to $f_s = 44.1$kHz. Include a test at the beginning of `create_micsigs.m` to check if the sampling frequency of the RIRs is indeed 44.1kHz, to avoid any mistakes.

2. Test your implementation:

   - Use `mySA_GUI.m` to create an acoustic scenario with two speech sources as target audio sources and without any noise source, and run the modified m-file `create_micsigs.m`. You should obtain an SNR of (approximately) 10 dB (why?). Listen to the first microphone signal (use the command `soundsc`) and discern the white noise.
   - Add a babble noise source to the same acoustic scenario and re-run `create_micsigs.m` (use `Babble_noise1.wav`). The SNR should now be smaller than 10 dB.
   - Remove the babble noise source (use one of the 'Del' buttons in the GUI) and add it again, this time closer to the microphone array. Re-run `create_micsigs.m`. The SNR should now be even smaller.

[1] For questions and remarks: `santiago.ruiz@esat.kuleuven.be`.

Part 2: DOA-informed delay-and-sum beamformer

1. Beamforming (or spatial filtering) is a signal processing technique used in sensor arrays for directional signal reception and transmission. Here, beamforming will be used to create a focused 'beam-like' sensitivity pattern for the microphone array. This is achieved by combining the microphone signals in such a way that some sources with specific DOA's experience constructive interference, while sources with other DOA's experience destructive interference. The simplest beamforming procedure is the so-called 'delay-and-sum beamformer' (DAS BF), and is explained briefly in http://www.labbookpages.co.uk/audio/beamforming/delaySum.html (check this out, skip the C functions).

2. Create a new m-file `DAS_BF.m` performing the following tasks:

   - Run `create_micsigs.m`.
   - Estimate the DOA of the target audio source based on the microphone signals in `mic` (re-use the code in `MUSIC_wideband.m`). In a multi-source scenario, it will be assumed that there is only one target audio source, i.e. the source closest to the 90° (broadside) direction, and that the total number of spatially localized sources Q is known (both target and noise sources). Identify Q based on the variables in `Computed_RIRs.mat` .
   - Design a DAS BF that steers towards the target audio source.
   - Apply this DAS BF to the signals in `speech` and `noise`, yielding two single-channel signals, which are stored in the variables `speech_DAS` and `noise_DAS`, respectively. The DAS BF output signal `DAS_out` is then the sum of `speech_DAS` and `noise_DAS`.
   - Plot the first microphone signal and the DAS BF output signal `DAS_out`. Apply a scaling factor in the DAS BF output signal such that the target audio source component has a similar power in both plots.
   - Compute the DAS BF output SNR using the signals in `speech_DAS` and `noise_DAS`, and store it in the variable `SNR_out_DAS`.

3. Create a non-reverberant acoustic scenario with a 5-microphone array with inter-microphone distance of 5cm, and with one target audio source at a random position (use `speech1.wav`). Do not add any noise sources, such that only the microphone noise is added in `create_micsigs.m`. Run the file `DAS.m`. Listen to the input and output signals, and check whether the noise level has been reduced. A difference of 1-2dB may be observed compared to the predicted improvement in SNR (why?).

4. Place the target audio source in the broadside direction at, approximately, a distance of 2 m from the microphone array, and add a noise source at a similar distance from the microphone array, but with a 45° DOA. Run the file `DAS.m`, for three different noise signals:

   - White noise (use `White_noise1.wav`)
   - Babble noise (use `Babble_noise1.wav`)
   - An interfering speech source (use `speech2.wav`)

   Listen to the input and output signals, and observe the SNR improvement for all three cases. Is there a difference in performance for different noise types? Why (not)?

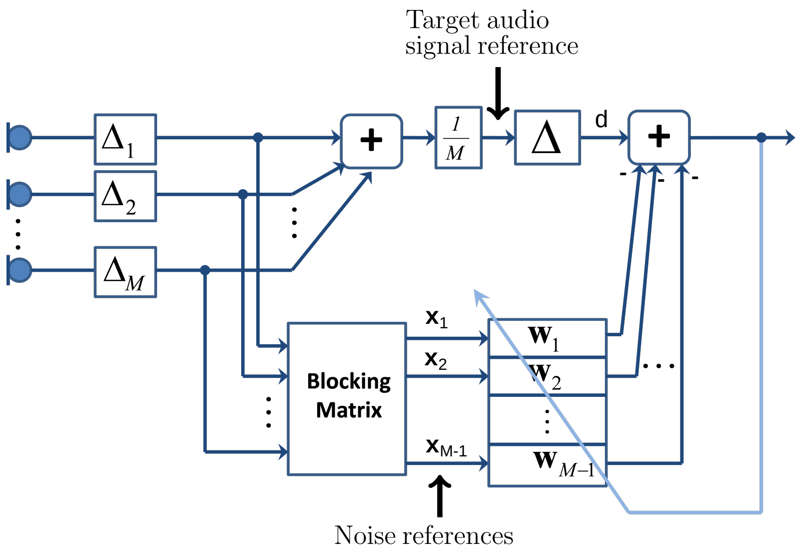

Figure 1: DAS-based GSC.

Part 3: DOA-informed generalized sidelobe canceler

1. The DAS BF can straightforwardly be extended to the (simplified) so-called 'generalized sidelobe canceler' (GSC) as depicted in Fig. 1. The top part is the DAS BF steering towards the target audio source. The DAS BF output signal (indicated as 'target audio signal reference') is expected to have a large target audio source component, but may still have a significant noise component. To reduce this noise component, an $(M-1) \times M$ so-called 'blocking matrix' is added together with a (multi-channel) adaptive filter. The blocking matrix forms $(M-1)$ so-called 'noise references', i.e. forms specific linear combinations of the delayed microphone signals in which the target audio source signal will (ideally) disappear. The so-called 'Griffiths-Jim blocking matrix', for instance, is given as

$$B = \begin{bmatrix} 1 & -1 & 0 & 0 & 0 & \cdots & 0 \\ 1 & 0 & -1 & 0 & 0 & \cdots & 0 \\ 1 & 0 & 0 & -1 & 0 & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & 0 & 0 & 0 & 0 & \cdots & -1 \end{bmatrix}. \tag{2}$$

   The multi-channel adaptive filter, finally, optimally filters the noise references, to subtract as much noise from the target audio signal reference as possible. The filters $\mathbf{W}_i$ are mostly FIR filters (L taps each), and can be adapted by using any adaptive filtering algorithm (LMS, RLS, etc.). Note that the (optional) additional delay Δ in the DAS BF signal path allows for a causal implementation of (otherwise) anti-causal filters $\mathbf{W}_i$ (explain).

2. Create a new m-file `GSC.m` performing the following tasks:

- Run DAS_BF.m.
- Compute the output signal of the GSC depicted in Fig. 1. Use the Griffiths-Jim blocking matrix, and use the normalized LMS (NLMS) algorithm for the multi-channel adaptive filter with a stepsize $\mu = 0.1$, filter length $L = 1024$ and $\Delta = \frac{L}{2}$. In later steps these parameters may be changed (optimized) and/or other blocking matrices may be experimented with.
- Add the GSC output signal to the figure created in DAS_BF.m (Compensate for the delay Δ that was applied in the GSC signal path). In the same figure, also add the first column of speech_DAS.
- Make an estimate of the SNR in the GSC output signal and store it in the variable SNR_out_GSC. Print this value in the Matlab workspace. Assume that the noise is stationary, i.e., that the noise has an equal power during 'noise-only' segments and 'speech+noise' segments, and hence that the noise power can be estimated during 'noise-only' segments by means of the VAD.

3. For the non-reverberant scenario from Part 2 (and for the three different noise types):
    - Listen to the output signals of the blocking matrix. Is there any target audio source signal component in the noise references (i.e. so-called 'speech leakage')?
    - Listen to the output signals and check the plots. Does the GSC perform better than the DAS BF?

      Remark: The estimated output SNR can sometimes be inaccurate, in particular in the case where the noise is a speech signal. Care should be taken with assessing the performance of the GSC based on the output SNR only.

4. Experiment with an NLMS stepsize $\mu > 1$. Listen to the GSC output signal and explain the observed effect. What causes this phenomenon? ***Hint***: The effect is most pronounced in the case where the noise is a speech signal.

Part 4: DOA-informed GSC with VAD

1. Simulate the same acoustic environment, but this time set the reverberation time to T60=0.5s. Place the microphone array in the middle of the room and make sure that the sources are close enough to the array (e.g., at 1m distance) such that the MUSIC algorithm can still correctly identify the DOA. Run GSC.m (with a white noise signal as the localized noise source) and analyze the following:
    - Listen to one of the output signals of the blocking matrix and compare the speech leakage to the non-reverberant case.
    - Compare the amplitude of the target audio source signal component in the output of the GSC and the DAS-BF, respectively (Explain).

2. Modify GSC.m such that it includes a VAD to switch off the filter adaptation whenever the target audio source is active. Note that (in a first version) the VAD can use the clean speech signal (which in practice, however, is not available).

3. Test your implementation on the earlier defined (reverberant) scenario, and check whether the speech leakage problem has indeed been solved.

Part 5: Extension of the DOA-informed GSC with VAD to the head-mounted microphones case

Consider the set of microphone signals generated from the same loudspeaker position on the left-hand side and one on the right-hand side as used in Week 2/ Part 5.

1. Apply the DOA-informed GSC with VAD to the two microphones pertaining to the right ear, $\mathbf{y_{R1}}, \mathbf{y_{R2}}$.
2. Apply the DOA-informed GSC with VAD to the two microphones pertaining to the left ear, $\mathbf{y_{L1}}, \mathbf{y_{L2}}$
3. Apply the DOA-informed GSC with VAD to the two frontal microphones, $\mathbf{y_{L1}}, \mathbf{y_{R1}}$
4. How do the results differ based on the chosen pair of microphones? Why?
5. Modify the DOA-informed GSC with VAD such that it can use all the four microphones, $\mathbf{y_{L1}}$, $\mathbf{y_{L2}}$, $\mathbf{y_{R1}}$, and $\mathbf{y_{R2}}$.
6. Compare the performance of the algorithm when 2-microphone and 4-microphone arrays are used.