Audio processing: lab sessions

Session 1: Introduction to the acoustic simulation environment

Alexander Bertrand, Giuliano Bernardi and Randall Ali[1]

October 2017

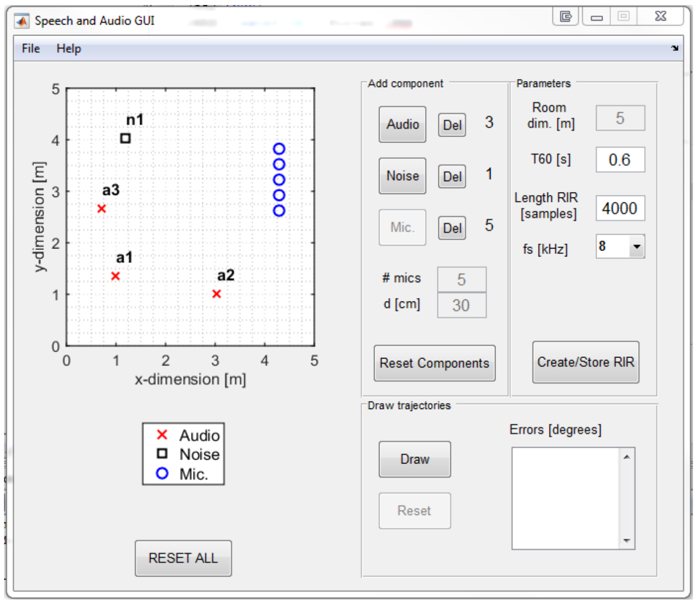
- **Goals:**
  I) Using a simulation environment to model room acoustics and multi-source scenarios.
  II) Taking a first step towards direction-of-arrival (DOA) estimation based on microphone signal correlation analysis.
- **Required files from course website:** `sim_environment.zip`, `speech1.wav`, `speech2.wav`, `Babble_noise1.wav` (download from `http://homes.esat.kuleuven.be/~dspuser/dasp`)
- **Required files from previous sessions:** none
- **Outcome:** 4 m-files: `create_micsigs.m`, `TDOA_corr.m`, `DOA_corr.m`, `DOA_corr_multi.m`
- **Deliverables:** See Session 2
- **Prelimenary remarks:**
  - If you are not familiar with MATLAB, you should first go through the MATLAB tutorial, which can be found on the course website. It is recommended to first make all exercises in this tutorial. Ask your TA for help if you run into problems. If you are already familiar with MATLAB, you can skip this tutorial.
  - Make sure you set the MATLAB path to your home-folder (do **not** save on the local hard-drive, since this will be erased each time you log off or when the computer crashes).
  - Work in m-files, and save your work regularly, in order to be able to repeat your experiments later on.
  - When plotting a result when the graphical user interface (GUI) is open, make sure to first open a new figure using the command `figure`. If not, the plot will be drawn in the GUI window.
  - In this session, as well as in all following sessions, it is assumed that the target source(s) for which the direction of arrival (DOA) is estimated are on the left-hand side of the microphone array.

1 Exercise 1-1: Simulate room impulse responses

In this exercise, you will use a simulation environment to generate room impulse responses (RIRs) from a user-defined acoustic scenario.

1. Download the zip-file `sim_environment.zip` from the course website.

[1]For questions and remarks: `giuliano.bernardi@esat.kuleuven.be` and `randall.ali@esat.kuleuven.be`



Audio 3
Noise 1
Mic. 5
# mics 5
d [cm] 30
Room dim. [m] 5
T60 [s] 0.6
Length RIR [samples] 4000
fs [kHz] 8

Figure 1: The room acoustics GUI.

2. Extract the zip-file in a new folder in your home directory. Open MATLAB, and set your MATLAB path to this new folder.
3. Compile the included mex-file by typing the command `mex simroommex.c` in MATLAB.
4. Open the graphical user interface by typing `mySA_GUI`. The window that pops up allows you to define a set of parameters used by the simulation environment:
   - *Room dimension*: the size of the room in meters (the room has a square shape).
   - *T60*: the T60 reverberation time (in seconds) of the room. Set the T60 to zero to create a non-reverberant room.
   - *Length RIR*: the length of the room impulse responses that are simulated (in number of samples). Using shorter lengths result in a faster simulation time, but then you should make sure that significant parts of the reverberant tail of the RIR are not cut off.
   - *Sampling frequency*: the frequency at which the RIRs are sampled. Large sampling rates yield more realistic results, but require more time to simulate.

   Set the room size to 10, the T60 to 1.5 s, and keep the other parameters to their default values,
5. The simulation environment allows you to place a microphone array of n microphones in a linear configuration (with a vertical orientation). Choose the number of microphones equal to 3 in the field denoted by '# mics'. Choose an inter-microphone distance of d =5 cm in the field denoted by 'd [cm]'. Then click on the Mic-button, and click somewhere in the grid to define the position of the lower microphone in the array.

6. Click on the Audio-button to also place a target audio source in the room (at the left hand side of the microphone array). Repeat this to place a second source.

7. Click on the Noise-button to place a noise source in the room.

8. Click on 'Create/Store RIRs', and wait until a confirmation appears.

9. If you have done this correctly, a new mat-file `Computed_RIRs.mat` should appear in the current MATLAB directory. Open this mat-file and check which variables it contains. Try to find out what each variable represents (based on the user-defined values in the simulation environment). The RIRs are stored in the variables `RIR_source` and `RIR_noise`. Try to figure out what each dimension of these variables represents. How many RIRs have been generated, and why?

10. Create a new figure and plot the RIR from source 2 to microphone 3. Can you recognize the direct path component, the early reflections, and the reverberant tail?

2 Exercise 1-2: Generate microphone signals

In this exercise, you will use the simulated RIRs to emulate the audio signals that are recorded by the microphone array. Check the help file of the following commands, which may be useful in the following task: `audioread`, `fftfilt`, `resample`, `load`, `save`.

1. Create an m-file with the name '`create_micsigs.m`'. This file will be also used for generating microphone signals in all other exercise sessions, hence it is essential that you pay particular attention to the parametrization of the code, i.e. the code should be able to function for changes in the number of microphones, target and noise sources, different sampling frequencies, etc. There are a few initialization steps to be performed within this m-file as follows:

   (a) In the beginning of the m- file, create a cell array that will be able to set the filenames that we will use for the target (speech signal), e.g. speechfilename{1} = speech1.wav, speeechfilename{2} = speech2.wav, etc.

   (b) Repeat the same procedure as in (a) to store the file names of the noise sources.

   (c) Set the desired length of the recorded microphone signals in seconds.

   (d) For each target source and noise source, re-sample them accordingly to match the sampling frequency of the RIR (not the other way around!).

   The file should then perform the following tasks:

   - Read out the file `Computed_RIRs.mat`.
   - Generate the n microphone signals that are recorded by the microphone array as defined in the simulation environment, i.e., containing the target+noise mixtures based on the RIRs stored in `Computed_RIRs.mat`.
   - Put the n microphone signals in the columns of a matrix variable `mic`. Save this variable in a mat-file with the same name, together with the value of the RIR sampling rate.
   - Plot the first two microphone signals in a single plot in different colors (using `hold on`).

2. Test the file '`create_micsigs.m`' in a non-reverberant scenario ($T60 = 0$ s) with a single speech and noise source. Use the file `speech1.wav` for the target speech source, and use the file `Babble_noise1.wav` for the noise source (these files can be downloaded from the course website). Listen to the signal recorded by the first microphone using the command `soundsc`.

   Remark: Note that the GUI contains several reset buttons to reset or redefine the values of certain parameters or the source/microphone locations. Also note that there is a save and load option to save a scenario for later use.

3. Repeat this procedure for the same source-microphone configuration, but where the room dimension is set to 10m and the reverberation time is set to 2 s (make sure your impulse response is sufficiently long). Listen to the microphone signal and compare with the previous scenario. What do you conclude in terms of the speech intelligibility in highly reverberant scenarios?

4. Create a new scenario containing a 2-microphone array with d =1 cm, and a single target source impinging in a 45° angle on the microphone array (note that the microphone array has a vertical configuration). There is no noise source. Set the sampling frequency of the RIRs to 44.1 kHz, and set the reverberation time to 0 s (no reverberation). Try to predict what the RIRs for both microphones will look like, and in what sense the two microphone signals will differ from each other. Confirm by simulation.

5. Simulate the same scenario, but now with a sampling frequency of 8000 Hz. Make a new plot of the two microphone signals in a single figure (using `create_micsigs.m`). What do you see? Explain...

6. With the previous scenario, you have discovered a problem that may hamper the assessment of multi-microphone audio processing algorithms when using simulated audio signals based on the simulated RIRs. Besides increasing the sampling frequency of the RIRs, what else can you do to reduce this effect?

3 Exercise 1-3: Cross-correlation-based time-difference-of-arrival (TDOA) estimation

1. Create a new scenario with 2 microphones with d =10 cm, and a single target source (no noise source) in a 10 m room with a reverberation time of 0 s (no reverberation). Use a sampling frequency of 16 kHz to generate the RIRs. Put the source on the left-hand side of the microphone array, close to a 45° angle with respect to the vertical axis of the microphone array. Make sure there is a few meters distance between the source and the microphone array.

2. Create a new m-file `TDOA_corr.m` that performs the following tasks:

   - Calculate the sample delay between the direct path components of the two RIRs. This will serve as the TDOA ground truth.
   - Generate the two microphone signals from the above scenario where the target audio source is a white noise signal (re-use code from `create_micsigs.m`).

- Estimate the TDOA from the two microphone signals, based on their (time-domain) cross-correlation function. Do this by shifting a representative segment of one microphone signal over the other microphone signal and identify the maximum cross-correlation peak. What trade-off do we have here in terms of the length of this segment?
- Make a plot of the time-domain cross-correlation function. Indicate in a different color where the maximum peak is expected according to the ground truth (e.g., using `stem`).
- Print the value of the difference between the estimated TDOA (based on the microphone cross-correlations) and the ground truth TDOA in the MATLAB workspace.

3. Test the file in the above (non-reverberant) scenario.

   Only if you have a zero TDOA estimation error, you can continue with the next step.

4. Increase the reverberation time. How much reverberation is allowed before the TDOA estimation starts to show errors?

5. Re-simulate the non-reverberant scenario, and replace the white noise source by a speech source[2] in `TDOA_corr.m` (use the speech signal in `speech1.wav`). How does the shape of the cross-correlation function change compared to the white noise case? Why?

   Only if you have a zero TDOA estimation error, you can continue with the next step.

6. Again increase the reverberation time. Can you allow as much reverberation as with the white noise source? Why (not)?

4 Exercise 1-4: Cross-correlation-based direction-of-arrival (DOA) estimation

1. Create a new m-file `DOA_corr.m` that first estimates the TDOA as in the previous exercise, but then also transforms this estimate to a DOA estimate (using the far-field assumption). Note that the inter-microphone distance can be extracted from the variable `m_pos` in `Computed_RIRs.mat`. Use a value of $c = 340$ m/s for the speed of sound in air. The file should store the DOA estimate in a variable `DOA_est` and store this variable in a mat-file with the same name. The DOA estimate should be between 0-180°, where 90° corresponds to the broadside direction to the *left* of the microphone array, and where 180° corresponds to the end-fire direction where the source is *below* the microphone array.

2. Reconstruct the first (non-reverberant) scenario of Exercise 1-3, with a RIR sampling frequency of 44.1 kHz and a DOA of approximately 45 degrees. Run the `DOA_corr.m` file, using a speech signal for the audio source.

[2]Note that the speech signal is not continuously active, i.e., be careful when selecting a representative signal segment to compute the cross-correlation function.

3. When pressing the 'Draw' button in the GUI, the GUI reads out the `DOA_est.mat` file and plots the DOA on the screen to visualize how much the estimated DOA is off target.[3] It will also show the DOA estimation error in a separate window. Check how well your estimated DOA matches with the true DOA.

   Only if the DOA error is reasonably small (e.g. less than 10 degrees), you can continue with the next step.

4. Generate the same scenario, but now with multiple audio sources to investigate how the DOA estimation accuracy changes with the source position. Make a new m-file called `DOA_corr_multi.m` that estimates the DOAs for all the specified source positions. For each source, a separate DOA estimation experiment is performed where only this one source is active, i.e., the microphone array does *not* record a mixture of all the source signals. Save all the computed DOAs in a vector variable `DOA_est` and store this variable in a mat-file with the same name.

5. Use `DOA_corr_multi.m` to check the DOA estimation accuracy for different source positions in a non-reverberant scenario (sample the RIRs at 44.1kHz). Add 10 sources in the [0 20] degree interval, and 10 sources in the [70 90] degree interval. Which DOAs can be estimated more accurately; those closer to the end-fire direction or those closer to the broadside direction? Why?

   Hint: Make a plot of the `acos` function

6. Choose 20 random source positions and simulate the non-reverberant environment using RIRs sampled at 44.1 kHz and 16 kHz. Does the sampling frequency have a significant impact on the overall accuracy? Why (not)?

7. How does the inter-microphone distance affect the DOA estimation? Try to predict first, and confirm experimentally on the previous scenario with d =50 cm instead of 10 cm.

8. Select a sampling frequency of 16 kHz and gradually increase the amount of reverberation over different experiments. How does this affect the DOA estimation?

5 Exercise 1-5: Multi-source (T)DOA estimation

Create a non-reverberant scenario with 2 source signals where the RIRs are sampled at 44.1kHz and with d =10cm. Re-use the code in `TDOA_corr.m` to compute and plot the cross-correlation function between the two microphone signals in a scenario where both sources are *simultaneously active.* Try this with two uncorrelated white noise signals, as well as with two speech sources (use `speech1.wav` and `speech2.wav`). Make sure to use a time segment in which both sources are active. Based on the observation of the corresponding cross-correlation functions, would it be possible to estimate the DOAs of both sources?

[3]The number of estimated DOAs in the file `DOA_est.mat` must match the number of sources placed in the room.