PYROADACOUSTICS: A ROAD ACOUSTICS SIMULATOR BASED ON VARIABLE LENGTH DELAY LINES

Stefano Damiano and Toon van Waterschoot

KU Leuven, Dept. of Electrical Engineering (ESAT), STADIUS Center for Dynamical Systems, Signal Processing and Data Analytics,
Leuven, Belgium
stefano.damiano@esat.kuleuven.be | toon.vanwaterschoot@esat.kuleuven.be

ABSTRACT

In the development of algorithms for sound source detection, identification and localization, having the possibility to generate datasets in a flexible and fast way is of utmost importance. However, most of the available acoustic simulators used for this purpose target indoor applications, and their usefulness is limited when it comes to outdoor environments such as that of a road, involving fast moving sources and long distances travelled by the sound waves. In this paper we present an acoustic propagation simulator specifically designed for road scenarios. In particular, the proposed Python software package enables to simulate the observed sound resulting from a source moving on an arbitrary trajectory relative to the observer, exploiting variable length delay lines to implement sound propagation and Doppler effect. An acoustic model of the road reflection and air absorption properties has been designed and implemented using digital FIR filters. The architecture of the proposed software is flexible and open to extensions, allowing the package to kick-start the implementation of further outdoor acoustic simulation scenarios.

1. INTRODUCTION

With the recent advent of (semi-)autonomous vehicles, companies and researchers are devoting an ever-growing effort to designing sensors and algorithms that target the perception and analysis of the environment. In this context, acoustic signals have been exploited in two main areas: first, microphone arrays have been placed on Unmanned Aerial Vehicles (UAV) to be used for sound source localization, identification and tracking [1, 2]. In particular, acoustic signals have proven useful in search and rescue applications [3, 4], where hidden sound sources can not be detected by sensors relying on visual clues. Second, microphone arrays are being used on (semi-)autonomous cars to improve their safety or enhance the ability of the driver to perceive the environment (e.g. acoustic sensing could help hearing-impaired people to detect approaching vehicles) [5]. Audio signals are also useful for the early detection of danger (e.g. car horns) and of emergency vehicles [6].

The ongoing and expanding development of algorithms for outdoor sound source detection and localization motivates the need of datasets to build and test new models. In particular, the algorithms exploiting deep learning techniques require a large amount of training data, very specific to the application they target. Some datasets for UAV applications have been proposed in the literature [4, 7], and some non-specific datasets [8] have been used to compensate for the hurdles of data collection. It is difficult, instead, to find suitable audio data of a road scenario, collected with microphones mounted on cars moving in the traffic. Moreover, since the research on microphone array architectures for automotive applications is still ongoing, a flexible approach to data generation, that can easily adapt to new array geometries and sound scenes, is of utmost importance to facilitate algorithm development. For this purpose, simulators constitute a powerful tool for the generation of large amounts of data during the design of new algorithms. Many indoor acoustic simulators have been proposed in the literature [9], but few are available that target outdoor applications. Moreover, most of the existing tools are built for noise analysis more than for the computation of simulated microphone signals [10, 11], and therefore are rarely useful in our context.

For this reason, the motivation behind *pyroadacoustics* is to provide researchers with a framework to simulate the propagation of sound signals emitted by sources moving on a road. The software we present in this paper is available at [12] and is built on top of the Object-Oriented Programming (OOP) features of Python, that make its architecture open for future extensions to different scenarios. In its first version, the proposed package targets the simulation of a highway-driving scene, defined by:

- One omnidirectional sound source, moving on an arbitrary trajectory with an arbitrary (variable) speed, and emitting a sound signal specified by the user.
- A static array of omnidirectional microphones, with arbitrary geometry, used to record the signal produced by the source.
- A stationary background noise of diffuse type, with an arbitrary SNR specified by the user.
- A set of environmental parameters that define the atmospheric conditions (i.e. the temperature T, pressure p_s and relative humidity h_{rel}).
- The acoustic properties of the road surface, depending on its construction material. The material can be chosen from a database of asphalt types, available in the software package, or defined from scratch by the user, and is characterized by a set of acoustic absorption coefficients at specific frequencies.

A detailed description of all the available options and parameters is provided in the package documentation [12]. In the current version of the simulator, the only reflecting surface considered is the ground, thus the presence of further nearby objects or buildings is neglected. This is a reasonable assumption for the modeling of a non-urban road such as a highway. Furthermore, the effect of wind is not taken into account. Note that, although the microphone array is assumed to be static, the *relative* motion between the source and the array is simulated, thus the presence of moving microphones can be faced by properly defining the trajectory of the source.

To design the proposed simulator, we exploited the delay line element [13], widely used in the literature for the implementation of both musical audio effects and Doppler simulators [14, 6]. Differently from [6], in *pyroadacoustics* both the direct sound and the sound reflected from the road surface are modeled and simulated. This enables to include in the simulations not only the Doppler effect, but also the interference of the direct and reflected sound waves. In order to achieve a realistic simulation of the reflection, a model of the asphalt surface has been included. The effect of the atmospheric absorption has also been implemented.

This paper aims to describe the design of the proposed acoustic simulator, and is structured as follows. In Sec. 2 we introduce the physics of the Doppler effect, the atmospheric acoustic absorption and the reflection properties of the road surface. In Sec. 3 we review the implementation of the delay line element and describe its use to model acoustic wave propagation. We thoroughly discuss the design and implementation of the *pyroadacoustics* simulator in Sec. 4, and show its capabilities in Sec. 5. Finally, in Sec. 6 we draw conclusions and envision future extensions of the simulator.

2. MODELING ROAD ACOUSTICS

The propagation of sound waves outdoors has been widely studied in the literature and shows peculiar characteristics, determined by both the intrinsic non-stationarity and inhomogeneity of the atmosphere, and the complexity of the acoustic scene, involving background noise, dynamic sound sources and the presence of scattering objects. In this scenario, the main acoustic phenomena discussed in the literature include the spreading of sound waves, the atmospheric absorption, the scattering of acoustic waves at the surface of the ground and objects located in the scene, the effect of wind on sound propagation, the temperature and pressure variations, causing variations in the speed of sound and refraction [15, 16]. Moreover, the Doppler effect, often negligible in indoor scenarios, must be considered in the presence of moving sources. In this section we will discuss the acoustic phenomena implemented in *pyroadacoustics*, namely:

- The Doppler effect, relevant in road scenarios including sound sources that move at relatively high speed.
- The air absorption, significant when sound waves travel large distances.
- The road reflection, being the main component of the scattered sound field in a highway scenario.

2.1. The Doppler Effect

In the analysis of moving sound sources having a velocity that is not negligible relative to the speed of sound in the air, the Doppler effect impacts how the radiated signal is received at some measurement position. For the sake of simplicity, without loss of generality, let us consider a static microphone located at position $\mathbf{r}$ and a source moving with constant velocity $\mathbf{v}$, located at position $\mathbf{r'}$ at a certain time τ. Here, the position and velocity vectors are defined in a three-dimensional Cartesian coordinate system. If the source emits a sinusoidal signal of frequency f, the receiver will observe a signal at a shifted frequency given by [13]

$$f_r = f \frac{c}{c - \langle \mathbf{v}, \frac{\mathbf{r}-\mathbf{r'}}{\|\mathbf{r}-\mathbf{r'}\|_2} \rangle} , \qquad (1)$$

where c denotes the speed of sound in air, $\langle \cdot, \cdot \rangle$ is the inner product (here chosen to be the dot product in $\mathbb{R}^3$) and $\|\cdot\|_2$ is the vector ℓ_2-norm. Thus, the Doppler effect will cause the received frequency to be higher than the emitted one when the source is moving towards the receiver, and lower when it is moving away from it.

2.2. The Sound Absorption Properties of Air

In this section we give an overview of the absorption properties of the atmosphere. The formulae presented here have been widely discussed in the literature and can be found (with minor variations) in multiple references, e.g. [17, 18]. Nonetheless, in order to keep this paper self-contained, in the following we present the expressions needed for the implementation discussed in Sec. 4.2.

As sound propagates through the air, part of its energy is converted to heat and thus dissipated. The most significant energy losses are caused by the viscosity of air, the heat conduction and the molecular relaxation of nitrogen and oxygen [19, 13]. This effect is known in the literature as air absorption, and is significant at high frequencies and long distances between the sound source and the receiver [15]. If we consider a plane wave having pressure p_0 at the position $\mathbf{r'}$ of a sound source, the pressure received by a microphone at position $\mathbf{r}$ is given by [15]

$$p(\mathbf{r}) = p_0 e^{-\frac{\alpha}{2}\|\mathbf{r}-\mathbf{r'}\|_2} , \qquad (3)$$

where [17]

$$\alpha = \frac{\alpha_0}{10 \log_{10} e} , \qquad (4)$$

and α_0 is the absorption coefficient, measured in $\mathrm{dB/m}$, depending on frequency, humidity, temperature and pressure. From (3) and (4) we can compute the dB-scale attenuation of the sound pressure caused by air absorption as

$$L_a = 20 \log_{10} \tfrac{p(\mathbf{r})}{p_0} = -10\alpha \|\mathbf{r}-\mathbf{r'}\|_2 \log_{10} e = -\alpha_0 \|\mathbf{r}-\mathbf{r'}\|_2 \qquad (5)$$

In the ISO 9613-1 standard [17] a closed-form expression for the absorption coefficient α_0 is given, see Eq. (2) at the bottom of the page. Here f is the frequency of the source signal, T is the air temperature, $T_0 = 293.15\,\mathrm{K}$, p_s is the atmospheric pressure, $p_{s0} = 1\,\mathrm{atm}$ is the reference atmospheric pressure, $f_{r,\mathrm{O}}$ is the oxygen relaxation frequency

$$f_{r,\mathrm{O}} = \frac{p_s}{p_{s0}} \left(24.0 + 4.04 \cdot 10^4 h \frac{0.02 + h}{0.391 + h} \right) , \qquad (6)$$

and $f_{r,\mathrm{N}}$ is the nitrogen relaxation frequency

$$f_{r,\mathrm{N}} = \frac{p_s}{p_{s0}} \left(\frac{T}{T_0} \right)^{-\frac{1}{2}} \left(9 + 280 h e^{-4.17\left[(T/T_0)^{-1/3} - 1\right]} \right) . \qquad (7)$$

$$\alpha_0 = 8.686 f^2 \left[1.84 \cdot 10^{-11} \left(\frac{p_s}{p_{s0}} \right)^{-1} \left(\frac{T}{T_0} \right)^{1/2} + \left(\frac{T}{T_0} \right)^{-5/2} \left(0.01275 \frac{e^{-2239.1/T}}{f_{r,\mathrm{O}} + f^2/f_{r,\mathrm{O}}} + 0.1068 \frac{e^{-3352/T}}{f_{r,\mathrm{N}} + f^2/f_{r,\mathrm{N}}} \right) \right] \mathrm{dB/m} . \qquad (2)$$

In (6) and (7) h denotes the molar concentration of water vapor,

$$h = h_{\text{rel}} \frac{p_{\text{sat}}}{p_s} , \tag{8}$$

where h_{rel} is the relative humidity expressed as a percentage [18],

$$p_{\text{sat}} = p_{s0} \cdot 10^{\left[-6.8346\left(\frac{T_{01}}{T}\right)^{1.261}+4.6151\right]} , \tag{9}$$

and $T_{01} = 273.16\,\text{K}$.

Thus, using equations (2)-(9) we are able to compute the absorption coefficient at any frequency f, for arbitrary atmospheric conditions characterized by a set of parameters $\{T, p_s, h_{\text{rel}}\}$.

2.3. The Road Reflection

In an outdoor environment where only scattering from the ground surface is considered, the sound field captured by a receiver can be described as the sum of two components: the direct sound, propagating along a straight line between the source and the microphone, and the sound reflected by the ground surface. The direct and the reflected waves combine to produce areas where constructive or destructive interference can be observed, depending on the relative positions of the source and the receiver and on the frequency components of the signal [15]. When a sound wave hits the ground, and only specular reflection is assumed, the angle of reflection is equal to the angle of incidence ϑ on the ground surface itself. The reflection properties of the surface are characterized by the reflection coefficient [20]

$$R(\vartheta) = \frac{Z(\vartheta)\cos\vartheta - Z_0}{Z(\vartheta)\cos\vartheta + Z_0} , \tag{10}$$

where $Z(\vartheta)$ is the impedance of the ground surface, while Z_0 is the characteristic impedance of air. If the interaction between the air and the ground is independent of the angle of incidence of the impinging sound waves, the ground surface is said to be *locally reactive*, and its impedance Z does not depend on ϑ. In this case, the angular dependency of the reflection coefficient is fully expressed by the cosine terms in (10). This assumption, despite being an approximation, is widely adopted in environmental noise analysis and outdoor acoustics [15, 21].

The acoustic properties of a construction material are usually described by its absorption coefficient $\alpha_{\text{m}}(\vartheta)$, that denotes the fraction of the incident sound energy that is not reflected from a surface made of that material [20]. This parameter depends mainly on the porosity and the thickness of the surface layer, and is generally measured for normal incident waves (i.e. when $\vartheta = 0$) using the Kundt's tube [22]. Its value is related to the reflection coefficient (10) by the formula [20]

$$\alpha_{\text{m}}(0) = 1 - |R(0)|^2 . \tag{11}$$

For locally reacting surfaces, Z can be computed from the values of $\alpha_{\text{m}}(0)$ and $R(0)$ using equations (10), with $\vartheta = 0$, and (11). Its value can then be used in (10) to compute the reflection coefficient $R(\vartheta)$ for any incidence angle ϑ of the sound waves [23]. Note that, for ease of notation, the frequency dependency of the road reflection properties has not been explicitly indicated throughout this section, yet this dependency will be taken into account in the implementation in Sec. 4.3.

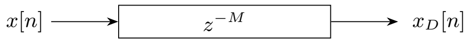


Figure 1: M-sample delay line.

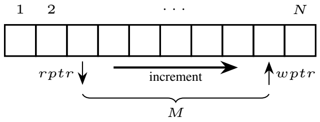


Figure 2: M-sample delay line as circular buffer, with read-pointer $rptr$ and write-pointer $wptr$. The pointers move in the direction indicated by the increment arrow.

3. THE DELAY LINE

The delay line is an elementary processing unit whose function is to introduce a time delay between its input and its output (Fig. 1). Given a discrete-time input signal $x[n]$, the signal at the output of the delay line can be written as

$$x_D[n] = x[n - M] . \tag{12}$$

If the parameter M changes over time, the resulting unit is called a *variable length* delay line, and can be used to model and simulate moving acoustic sources [13]. A software implementation of the delay line, proposed in [13], is based on the use of a circular buffer of length N (see Fig. 2). The input signal is written in the buffer sample by sample, at the position of a *write-pointer*. This position is increased by one at every time step. The output signal is read sample by sample at the position of a *read-pointer*, delayed by M samples with respect to the write-pointer. The delay M (i.e. the distance between the read and the write-pointer) is allowed to vary over time, as long as the condition $M \leq N$ is met.

3.1. Simulation of Acoustic Propagation Using Delay Lines

Delay lines produce pure time delays, that can be used to model acoustic waves propagating in a certain direction, with a fixed wave shape. In this scenario, $x[n]$ represents the signal generated by a sound source located at position $\mathbf{r}'$, and $y[n]$ corresponds to the signal received at position $\mathbf{r}$. The delay M is computed as

$$M = \frac{\|\mathbf{r} - \mathbf{r}'\|_2}{c} f_s , \tag{13}$$

where f_s denotes the sampling frequency, used to convert the propagation delay in seconds to the digital delay in number of samples. Since in practice M can take non-integer values, interpolated delay line reads are needed to deal with fractional delays (see Sec. 3.2). Note that when the positions of the read and the write-pointers are controlled by a physical model of sound propagation, the two are always separated by a minimum and maximum delay interval, that can be predicted if the trajectory of the source is known. Thus, the read-pointer never surpasses the write-pointer.

In a realistic scenario, the acoustic wave attenuates as it propagates, and some dispersion due to its interaction with the environment can be present (e.g. the air absorption). These phenomena can be implemented using digital filters cascaded to the delay line. In the most general case, when both sound attenuation and frequency-dependent air absorption are considered, the sound propagation can be modeled using the block scheme in Fig. 3. The delay line discussed in Sec. 3 enables to simulate multiple receivers by using multiple read-pointers. The read-pointers can

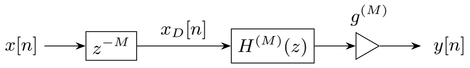


Figure 3: Acoustic traveling wave simulator: delay line models propagation, followed by air absorption filter and attenuation.

move independently, and the output signals can follow separate filtering paths to simulate different propagation losses [14]. This can be exploited to model reflections (or acoustic echoes) from nearby objects such as the ground surface.

3.2. Fractional Delays and Interpolated Delay Line Reads

In a road scenario, moving sound sources play a fundamental role. To simulate their behavior, the delay M must vary smoothly to adapt to the varying position of the sound source. Since M can take non-integer values, interpolated delay line reads are necessary. For this purpose, various fractional delay interpolators have been proposed in the literature [24], and their effect on variable delay lines has been discussed in [13]. We now present three different interpolators that we implemented in *pyroadacoustics*, namely the linear, all-pass and windowed-sinc interpolator.

Let us consider a delay line with fractional delay $M = M_{\text{int}} + \Delta$, with $M_{\text{int}} = \lfloor M \rfloor$ being its integer part and Δ, s.t. $0 \leq \Delta < 1$, its fractional part. The output signal of the M-samples delay line can be computed using a M_{int}-samples delay element followed by an interpolator block, as shown in Fig. 4.

A linear interpolator computes the delayed output sample as a weighted sum of the two neighboring samples available on the delay line, i.e. [13]

$$x_D[n] = x_M[n] + \Delta \cdot (x_M[n-1] - x_M[n]) \ , \tag{14}$$

where $x_M[n] = x[n - M_{\text{int}}]$. This procedure involves one multiplication and two additions. Notwithstanding the low computational complexity, this method produces amplitude distortion and aliasing for high frequency signals [25], as shown in Fig. 5.

A first efficient alternative is the first-order all-pass interpolation, defined by the difference equation [13]

$$x_D[n] = a \cdot (x_M[n] - x_D[n-1]) + x_M[n-1] \, , \tag{15}$$

where the factor a is computed as

$$a = \frac{1-\Delta}{1+\Delta} \, . \tag{16}$$

This interpolator can be implemented using the same number of operations as the linear interpolator, but is not affected by its artifacts, as discussed in [25] and shown in Fig. 5.

To further improve the accuracy of the interpolation, at the cost of a higher computational complexity, a windowed-sinc interpolator can be used. Its design, described in [24], is based on the least squares filter design method, and its impulse response is given by

$$h_{\text{int}}[n] = \begin{cases} W(n-\Delta)\,\text{sinc}\,(n-\Delta) & -\frac{N_w}{2} \leq n \leq \frac{N_w}{2} \\ 0 & \text{otherwise} \end{cases} , \tag{17}$$

where W is a window function of length $N_w + 1$ (with N_w even).

To show the effect of different interpolators on a variable delay line we set up a simple simulation: a sound source emitting, in the free-field, a sinusoid with $f = 3000\,\text{Hz}$ and fixed amplitude, is moving at speed $v = 5\,\text{m/s}$ towards a static microphone.

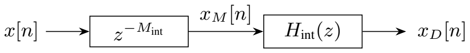


Figure 4: Interpolated delay line: the fractional delay $M = M_{\text{int}} + \Delta$, with $M_{\text{int}} = \lfloor M \rfloor$, $0 \leq \Delta < 1$, is implemented with a delay line of M_{int} samples followed by an interpolator.

We choose the sampling frequency $f_s = 8000\,\text{Hz}$ and simulate the propagation delay with a variable delay line as described in Sec. 3.1. No losses are considered in this simulation. Fig. 5 shows the DFT magnitude spectrum of the received signal obtained with the three different interpolators. While in Fig. 5a the aliasing produced by the linear interpolation is clearly visible, its impact is strongly reduced when we use the all-pass interpolator (Fig. 5b) and disappears when a windowed-sinc interpolator, implemented with a Hanning window of 31 samples, is adopted (Fig. 5c). An analysis of the impact of the three different interpolators on the simulator processing time is reported in Tab. 1.

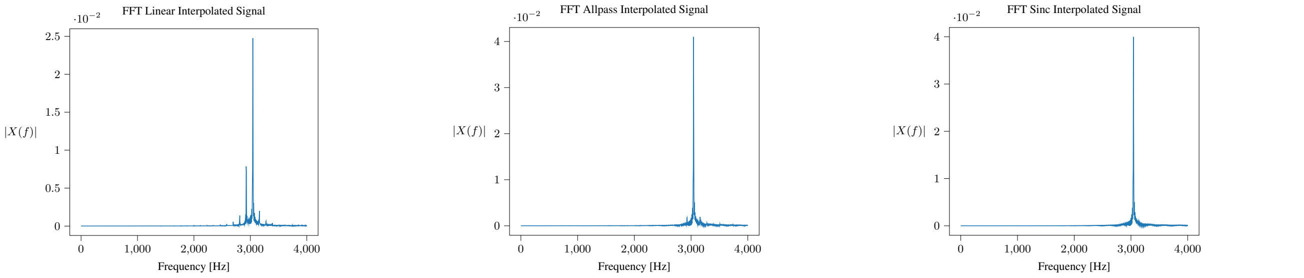
4. ROAD ACOUSTICS SIMULATION

The delay line described in Sec. 3 is the fundamental building block of *pyroadacoustics* and is used to simulate the acoustic propagation and the Doppler effect. In addition to these features, we included an accurate model of the acoustic reflection properties of the road surface, and of the sound absorption properties of the atmosphere. In this section we discuss the overall architecture of *pyroadacoustics*, and the implementation details of the air absorption and asphalt reflection properties introduced in Sec. 2.

4.1. Simulator Architecture

As stated in Sec. 1, *pyroadacoustics* targets the simulation of sound propagation on a non-urban road. We model this scenario as a single reflecting surface (i.e. the road surface), above which a sound source is moving with a certain velocity along an arbitrary trajectory, as sketched in Fig. 7. The sound produced by a (moving) source S reaches a receiver M by travelling along two different paths. The direct sound travels the distance d_1 from S to M, and suffers losses due to air absorption depending on d_1 as modeled by Eq. (3). The reflected sound travels from S to the reflection point on the road surface, Q, being affected by air absorption depending on the distance d_2. The position of Q is found by the law of specular reflection, as discussed in Sec. 2.3. The reflected sound wave, then, travels the distance d_3 to finally reach the microphone M, and is again affected by air absorption losses depending on d_3.

In order to simulate this environment, we defined the simulator architecture represented in Fig. 6. The propagation delays are implemented using two delay line blocks. The first delay line has two read-pointers: the first one is used to model the delay corresponding to the path from S to M, and the second one the delay occurring in the propagation from S to Q. Note that the delay line has only one write-pointer, that feeds in the signal emitted by the sound source. The second delay line has one read-pointer, corresponding to the delay related to the propagation from Q to M. Each read operation on the two delay lines is performed using an interpolator (see Sec. 3.2), whose type can be set by the user. Along each path depicted in Fig. 7 a FIR filter implements the air absorption (its design will be discussed in Sec. 4.2), and a gain element implements the attenuation due to spreading, proportional to the inverse of the travelled distance [20]. The road reflection is implemented using the FIR filter H_{refl}, discussed in Sec. 4.3. Note that since the



(a)

(b)

(c)

Figure 5: Comparison between fractional delay interpolation methods applied to a moving source emitting a sinusoidal signal. (a) linear interpolation; (b) first order all-pass interpolation; (c) windowed-sinc interpolation, $N_w = 30$.

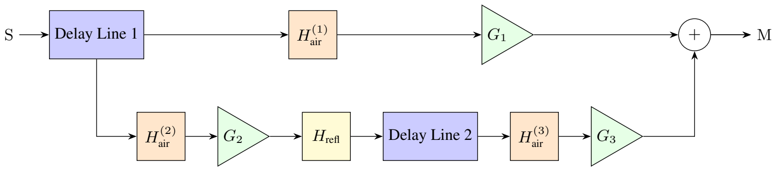


Figure 6: *Pyroadacoustics* block scheme.

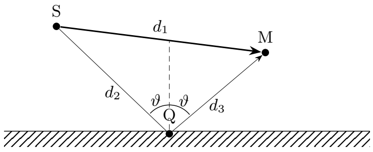


Figure 7: Geometry of implemented multipath propagation: the microphone M receives the direct signal emitted by the source S, propagating via d_1, and the reflected sound, via d_2 and d_3.

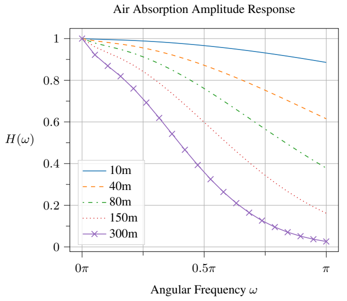


Figure 8: Amplitude response of air absorption FIR filters computed at different distances. Atmospheric parameters: $T = 20\,°\text{C}$, $p_s = 1\,\text{atm}$, $h_{\text{rel}} = 50\%$, $f_s = 16\,\text{kHz}$. $H(\omega)$ is computed at a set of $K = 20$ equally spaced frequencies in $[0, f_s/2]$.

air absorption and asphalt reflection filters are time-varying, they cannot be commuted and merged to reduce the number of blocks. To make the simulator more flexible, the user can choose which filters to be included or excluded from the simulation.

4.2. Air Absorption Implementation

The air absorption filter design is performed in two steps. First, after setting the sampling frequency f_s to be used in the simulations, let us consider a set of K angular frequencies $\omega_k = 2\pi f_k / f_s$ such that $\omega_k \in [0, \pi]$. Given the atmospheric parameters $\{T, p_s, h_{\text{rel}}\}$ of the simulation scene, we can compute the air absorption coefficient $\alpha_{0,k}$ at each ω_k using equations (2) to (9). Then, at every simulation instant, we can compute the relative attenuation due to atmospheric absorption at the instantaneous distances d_1, d_2, d_3, for each ω_k, by means of Eq. (5). By converting the result to linear scale, we obtain a set of values that represent the amplitude response $|H(\omega_k)|$ of a FIR filter, with a low-pass behavior as shown in Fig. 8.

The second step is the FIR filter design from its amplitude response. For this purpose, an effective technique is the weighted least squares method [26], that we implemented following [27]. Given the values of the amplitude response $|H(\omega_k)|$ at the angular frequencies ω_k, we want to retrieve the impulse response of the corresponding $L + 1$ taps filter. To this end, we first design the filter impulse response to be centered around the origin (i.e. zero-phase, $|H(\omega_k)| \equiv H(\omega_k)$), enforcing even symmetry in the impulse response, and then shift it to the right by $L/2$ samples to make it causal and linear-phase, obtaining the expression

$$H(\omega_k) = h_0 + 2\sum_{n=1}^{L/2} h_n \cos(\omega_k n) \,, \tag{18}$$

with $k = 0, 1, \ldots, K-1$. Eq. (18) can be written in matrix form as

$$\underbrace{\begin{bmatrix} H(\omega_0) \\ H(\omega_1) \\ \vdots \\ H(\omega_{K-1}) \end{bmatrix}}_{\underline{\mathbf{d}}} = \underbrace{\begin{bmatrix} 1 & 2\cos(\omega_0) & \cdots & 2\cos[\omega_0(L/2)] \\ 1 & 2\cos(\omega_1) & \cdots & 2\cos[\omega_1(L/2)] \\ \vdots & \vdots & & \vdots \\ 1 & 2\cos(\omega_{K-1}) & \cdots & 2\cos[\omega_{K-1}(L/2)] \end{bmatrix}}_{\mathbf{A}} \underbrace{\begin{bmatrix} h_0 \\ h_1 \\ \vdots \\ h_{L/2} \end{bmatrix}}_{\underline{\mathbf{h}}} . \tag{19}$$

The filter coefficients, stored in the vector $\underline{\mathbf{h}}$, can be retrieved as

$$\hat{\underline{\mathbf{h}}} = \arg\min_{\underline{\mathbf{h}}} \|\mathbf{A}\underline{\mathbf{h}} - \underline{\mathbf{d}}\|_2^2 \,. \tag{20}$$

This optimization problem has the well known solution

$$\hat{\underline{\mathbf{h}}} = \left[\left(\mathbf{A}^T\mathbf{A}\right)^{-1}\mathbf{A}^T\right]\underline{\mathbf{d}} \,, \tag{21}$$

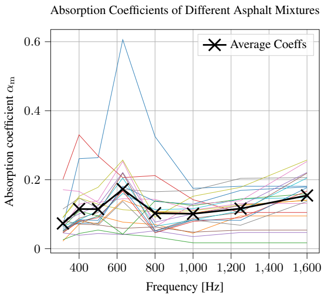


Figure 9: Normal incidence absorption coefficients of different asphalt mixtures [22], with corresponding average values.

where $\mathbf{A}^\dagger = \left(\mathbf{A}^T\mathbf{A}\right)^{-1}\mathbf{A}^T$ is the Moore-Penrose pseudo-inverse of matrix $\mathbf{A}$. Since from Eq. (19) we can see that $\mathbf{A}$ does not depend on distance, $\mathbf{A}^\dagger$ can be pre-computed, so that only the product of $\mathbf{A}^\dagger$ and $\underline{\mathbf{d}}$ (that, instead, does depend on distance) needs to be computed at every simulation instant. This makes the implementation of the least squares method more computationally efficient compared to standard implementations, in which the matrix $\mathbf{A}$ is recomputed at every time step. Since three different air absorption filters must be computed at each simulation instant (see Fig. 6), this approach significantly reduces the total execution time.

4.3. Asphalt Reflection Implementation

The implementation of the asphalt reflection is based on a similar filter design procedure. In the literature, the values of the normal incidence absorption coefficients for several asphalt types are available in tables [22, 28]. However, these coefficients are usually given for frequencies ranging from 0 to 2 kHz: therefore, an extrapolation procedure is needed in order to simulate wideband signals. In *pyroadacoustics* we included a database of coefficients of the different asphalt mixtures studied in [22], whose absorption properties are plotted in Fig. 9. In order to extrapolate the coefficients to higher frequencies, we perform a 2nd order polynomial curve fitting in the logarithmic domain. Let us consider the coefficients $\alpha_{\mathrm{m},k}$, computed in [22] for a set of K frequencies $f_{\mathrm{m},k}$, and another set of I frequencies $f_{\mathrm{m},i} \in [0, f_s/2]$ at which we want to compute the extrapolated coefficients $\hat{\alpha}_{\mathrm{m},i}$. Then, the extrapolation is performed in three steps:

1. Compute $\tilde{\alpha}_{\mathrm{m},k} = \log \alpha_{\mathrm{m},k}$ and $\tilde{f}_{\mathrm{m},k} = \log f_{\mathrm{m},k}$.
2. Retrieve the log scale extrapolated coefficients $\tilde{\alpha}_{\mathrm{m},i}$ at log frequencies $\tilde{f}_{\mathrm{m},i} = \log f_{\mathrm{m},i}$, by performing a 2nd order polynomial curve fitting on $\tilde{\alpha}_{\mathrm{m},k}$ and $\tilde{f}_{\mathrm{m},k}$.
3. Compute the linear scale coefficients $\hat{\alpha}_{\mathrm{m},i} = e^{\tilde{\alpha}_{\mathrm{m},i}}$.

The advantage of this approach over linear interpolation (adopted, e.g., in [9]) is that in the log domain the lower frequencies have a higher resolution, and thus the estimated coefficients in the known range have a better fit to the available data. Moreover, the logarithmic transformation smooths out steep variations in the coefficients: this improves the results when the extrapolation is performed over averaged data. Note that the extrapolation can be performed on any set of coefficients, either chosen from the database provided in the package or defined by the user. In Fig. 10 we plot the result of the extrapolation of the averaged coefficients shown in Fig. 9. It can be seen that by fitting a 2nd order polynomial and carrying out the extrapolation in the logarithmic domain, the peak between 0.5 kHz and 1 kHz is preserved. Once the absorption coefficients in the desired frequency range have been retrieved, and under the assumption of a locally reactive asphalt surface, we can exploit equations (10)-(11) to compute the values of the asphalt reflection coefficients, as described in Sec. 2.3.

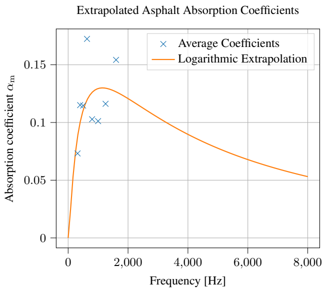


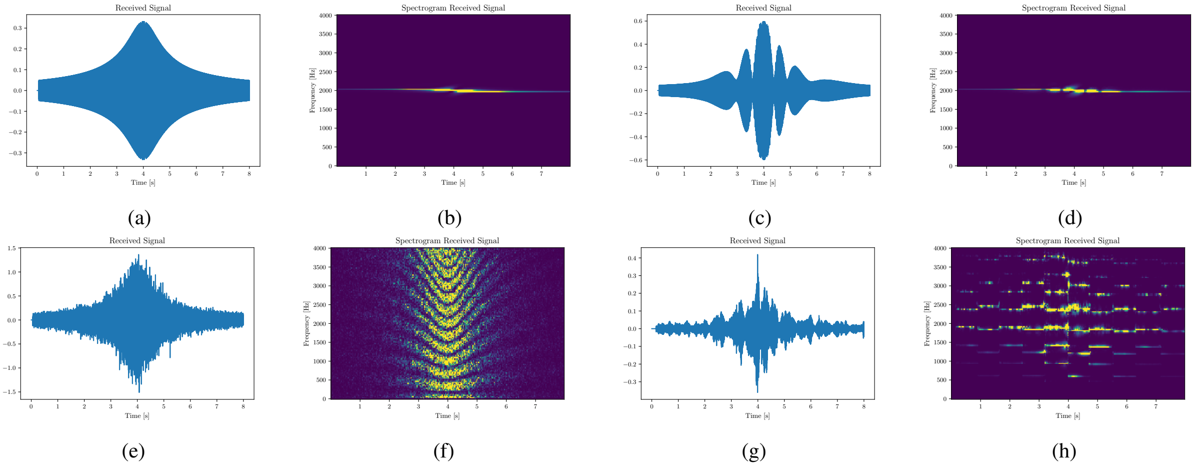
Figure 10: Extrapolated asphalt absorption coefficients. A second order polynomial extrapolation is performed after a logarithmic transformation. The known coefficients are the average absorption coefficients of Fig. 9, then extrapolated to $f \in [0, 8\,\mathrm{kHz}]$.

With the described approach, for each ϑ we are able to obtain a set of reflection coefficients $R(\vartheta)$ computed at some specific frequencies ω_i. These coefficients can be interpreted as the amplitude response of a FIR filter, that can be designed once again using the weighted least squares method. Note that from (11) we can see that with this procedure only the magnitude of the reflection coefficient is preserved. Therefore, the described filter design only considers the magnitude response, while discarding the phase. Since the angle of incidence of the sound wave on the road surface ϑ depends on the relative position of the source and the microphone array, the procedure has to be performed at each simulation instant. In order to speed up the simulation, however, we can avoid this by pre-computing the impulse response of the filters corresponding to a particular grid of incident angles, e.g. all the integer values of ϑ in the range $[-89, 89]^\circ$. Then, at each time step, the angle of incidence of the sound wave on the asphalt surface is computed and rounded to the nearest integer, and the corresponding filter is selected from the table.

The detailed models of asphalt reflection and air absorption we implemented in *pyroadacoustics* guarantee an improved simulation accuracy over similar simulators. The proposed package, in particular, allows the user to observe and analyze how the atmospheric conditions and the properties of the road surface affect the sound propagation. In the next section we will present some simulation results to show the possibilities offered by this software package.

5. PERFORMANCE AND RESULTS

To show the capabilities of *pyroadacoustics*, we run some simulations in the following scenario. First, we set the atmospheric parameters $T = 20\,^\circ\mathrm{C}, p_s = 1\,\mathrm{atm}, h_{\mathrm{rel}} = 50\%$ and the sampling frequency $f_s = 8\,\mathrm{kHz}$. Then, we consider a single microphone at position $[0, 0, 1]$ m and a sound source moving along a straight line, from position $[3, 20, 1]$ m to position $[3, -20, 1]$ m, with constant speed $v = 5\,\mathrm{m/s}$. In this scenario, we perform several simulations using different source signals. In order to improve the visualization of the differences between the results, no background noise is introduced. However, since the background noise supported by the simulator is stationary, its introduction would not affect the analyzed phenomena. The audio files related to the sim-



(a) (b) (c) (d)

(e) (f) (g) (h)

Figure 11: Waveform and spectrogram of simulated signals as received by the microphone: (a)-(b): sinusoidal signal, without asphalt reflection and air absorption; (c)-(d): sinusoidal signal, with asphalt reflection and air absorption; (e)-(f): white noise signal, with asphalt reflection and air absorption; (g)-(h): emergency vehicle siren signal, with asphalt reflection and air absorption.

ulations discussed in this section are available at [12].

To show the impact of the Doppler effect on the simulated sound, we first choose a sinusoidal source signal with $f = 2\,\text{kHz}$, and we simulate only the direct path. Fig. 11a-11b contain the waveform of the received signal and the corresponding spectrogram, computed using a 256 samples Tukey window with shape parameter 0.25, an overlap of 32 samples and Fourier transform length $N_{\text{FFT}} = 256$. Here, the Doppler shift is clearly visible.

We then repeat the simulation, this time simulating also the reflected path in the same scenario. The results are reported in Fig. 11c-11d. As the reflected sound is received by the microphone, it combines with the direct component and interference patterns emerge. The interference produces an effect known as *comb filtering* [13]: due to this phenomenon the spectrum of the received signal contains notches and peaks at different frequencies, that are related to the delay between the direct and the reflected sound. Since this delay depends on the relative distance between the source and the receiver, the position of peaks and notches varies throughout the simulation. This effect can be better visualized by assigning a white noise signal to the source, in the same previously defined scenario, as shown in Fig. 11e-11f. The presence of the variable asphalt reflection and air absorption filters affects the behavior of the comb filtering: the direct and reflected components, in fact, are not exact replicas. Instead, each of them is a differently filtered instance of the signal emitted by the source. Note that an analysis of the time-varying position of the notches of the comb filter can provide information about the relative position of the source and the receiver, and their relative velocity. Therefore, it is important to include in the simulation an accurate model of the road reflection and air absorption to enable the use of simulated data in the design of algorithms that exploit the spectral content of microphone signals in order to perform, e.g., sound source localization, identification and tracking.

The plots in Fig. 11g-11h show the output of the simulation when the source signal is a siren sound. A freely available audio sample of a *high-low* siren has been collected in an online repository for the purpose of running this demo.

Finally, in Tab. 1 we report the time required by the proposed software to simulate 1 sec of audio, when different signal paths and interpolators are used in the simulation. The simulations are performed using a 8 core Apple M1 CPU, and a 16 GB memory. The linear and allpass interpolators have a similar complexity, while using the sinc filter requires a longer execution time.

Table 1: Time in seconds required to simulate 1 sec of audio produced by a source moving at a constant speed, and captured by a single microphone, for different signal paths and interpolation methods. The sampling frequency is $f_s = 8\,\text{kHz}$.

Simulated Signal Path	Interpolation Method		
	Linear	Allpass	Sinc
Direct Path without Air Absorption	0.12	0.12	0.23
Direct Path with Air Absorption	0.19	0.19	0.29
Direct + Reflected Paths without Air Absorption	0.18	0.17	0.33
Direct + Reflected Paths with Air Absorption	0.35	0.34	0.50

6. CONCLUSIONS

In this paper we presented *pyroadacoustics*, a Python package for the simulation of sound propagation on a road, resembling a highway environment. The software enables to simulate the sound produced by a moving sound source and received by a stationary microphone array. Both the direct sound and the sound reflected by the road surface are considered. The proposed implementation allows to simulate the Doppler effect, and is provided with an extensive model of the sound interaction with the road surface and with the atmosphere. Demos of the simulation capabilities have been provided, that show the impact of the Doppler effect and of the interference between the direct and the reflected sound waves on the simulated microphone signal. The road environment can be defined in a flexible way, allowing the user to simulate the full sound field, or just the direct component, and to choose whether to include the air absorption in the simulation. This allows to balance the trade-off between physical accuracy and computational complexity depending on the specific requirements of each simulation.

The current version of *pyroadacoustics* does not support wind noise and further reflections from nearby objects. However, the open architecture based on the OOP-features of Python allows to easily extend the simulator to implement these features, which we foresee as future work. Also, multiple sound sources and moving background noise sources could be included in the simulator.

7. ACKNOWLEDGMENTS

The research leading to these results has received funding from the European Union's Horizon 2020 research and innovation programme under the Marie Skłodowska-Curie grant agreement No. 956962 and from the European Research Council under the European Union's Horizon 2020 research and innovation program / ERC Consolidator Grant: SONORA (no. 773268). This paper reflects only the authors' views and the Union is not liable for any use that may be made of the contained information.

8. REFERENCES

[1] M. Wakabayashi, H. G. Okuno, and M. Kumon, "Multiple Sound Source Position Estimation by Drone Audition Based on Data Association Between Sound Source Localization and Identification," *IEEE Robot. Autom. Lett.*, vol. 5, no. 2, pp. 8, 2020.

[2] L. Wang, R. Sanchez-Matilla, and A. Cavallaro, "Tracking a moving sound source from a multi-rotor drone," in *Proc. 2018 IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, Madrid, Oct. 2018, pp. 2511–2516, IEEE.

[3] K. Nakadai, M. Kumon, H. G. Okuno, K. Hoshiba, M. Wakabayashi, K. Washizaki, T. Ishiki, D. Gabriel, Y. Bando, T. Morito, R. Kojima, and O. Sugiyama, "Development of Microphone-Array-Embedded UAV for Search and Rescue Task," in *Proc. 2017 IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, 2017, pp. 5985–5990.

[4] O. Ruiz-Espitia, J. Martinez-Carranza, and C. Rascon, "AIRA-UAS: An Evaluation Corpus for Audio Processing in Unmanned Aerial System," in *Proc. 2018 Int. Conf. Unmanned Aircraft Syst. (ICUAS)*, Dallas, TX, June 2018, pp. 836–845.

[5] O. Barak, N. Sallem, and M. Fischer, "Microphone Array Optimization for Autonomous-Vehicle Audio Localization Based on the Radon Transform," in *Proc. 5th Workshop Detection Classification Acoust. Scenes Events (DCASE 2020)*, Tokyo, Nov. 2020, pp. 1–5.

[6] M. Cantarini, L. Serafini, L. Gabrielli, E. Principi, and S. Squartini, "Emergency Siren Recognition in Urban Scenarios: Synthetic Dataset and Deep Learning Models," in *Intell. Comput. Theories Appl.*, vol. 12463, pp. 207–220. Springer International Publishing, Cham, 2020.

[7] M. Strauss, P. Mordel, V. Miguet, and A. Deleforge, "DREGON: Dataset and Methods for UAV-Embedded Sound Source Localization," in *Proc. 2018 IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, Madrid, Oct. 2018, pp. 1–8.

[8] J. F. Gemmeke, D. P. W. Ellis, D. Freedman, A. Jansen, W. Lawrence, R. C. Moore, M. Plakal, and M. Ritter, "Audio Set: An ontology and human-labeled dataset for audio events," in *Proc. 2017 IEEE Int. Conf. Acoust., Speech, Signal Process. (ICASSP '17)*, New Orleans, LA, Mar. 2017, pp. 776–780.

[9] R. Scheibler, E. Bezzam, and I. Dokmanic, "Pyroomacoustics: A Python Package for Audio Room Simulation and Array Processing Algorithms," in *Proc. 2018 IEEE Int. Conf. Acoust., Speech, Signal Process. (ICASSP '18)*, Calgary, AB, Apr. 2018, pp. 351–355.

[10] J. Picaut and N. Fortin, "I-Simpa, a graphical user interface devoted to host 3D sound propagation numerical codes," in *Proc. Acoust. 2012*, Nantes, France, Apr. 2012, pp. 9–14.

[11] A. Farina, "DISIA - Noise Simulation in Urban Areas," http://pcfarina.eng.unipr.it/Public/DISIA/, 1995.

[12] S. Damiano, "Pyroadacoustics - Python Road Acoustics Simulation," Available at: https://github.com/steDamiano/pyroadacoustics, 2022.

[13] J. O. Smith, *Physical Audio Signal Processing*, online book, 2010 edition, http://ccrma.stanford.edu/~jos/pasp/, 2010.

[14] J. O. Smith, S. Serafin, J. Abel, and D. Berners, "Doppler Simulation and the Leslie," in *Proc. 5th Int. Conf. Digital Audio Effects (DAFx-02)*, Hamburg, Germany, Sept. 2002, pp. 13–20.

[15] K. Attenborough, "Sound Propagation in the Atmosphere," in *Springer Handbook of Acoustics*, Thomas D. Rossing, Ed., pp. 117–155. Springer New York, New York, NY, 2014.

[16] M. Vorländer, *Auralization: Fundamentals of Acoustics, Modelling, Simulation, Algorithms and Acoustic Virtual Reality*, RWTHedition. Springer, Cham, 2020.

[17] ISO 9613-1, "Acoustics, Attenuation of sound during propagation outdoors — Part 1: Calculation of the absorption of sound by the atmosphere," Geneva, Switzerland, 1993, International Organization for Standardization.

[18] E. M. Salomons, *Computational Atmospheric Acoustics*, Springer, Dordrecht, first edition, 2001.

[19] K. U. Ingard and P. M. Morse, *Theoretical Acoustics*, Princeton University Press, Princeton, 1986.

[20] H. Kuttruff, *Acoustics: An Introduction*, Taylor & Francis, London, 2007.

[21] J.H. Rindel, "Modelling the angle-dependent pressure reflection factor," *Applied Acoustics*, vol. 38, no. 2-4, pp. 223–234, 1993.

[22] A. Ranzo, P. Di Mascio, and A. D'Andrea, "Absorption Coefficients of Asphalt Pavements," in *2nd Eurasphalt & Eurobitume Congress, 20-22 September, 2000, Barcelona, Spain: Proceedings of the Papers Submitted for Review.*, Breukelen, The Netherlands, Sept. 2020, vol. II, pp. 891–902.

[23] L. Nijs, G. Jansens, G. Vermeir, and M. van der Voorden, "Absorbing Surfaces in Ray-Tracing Programs for Coupled Spaces," *Applied Acoustics*, vol. 63, no. 6, pp. 611–626, June 2002.

[24] T. I. Laakso, V. Välimäki, M. Karjalainen, and U. K. Laine, "Splitting the Unit Delay - Tools for Fractional Delay Filter Design," *IEEE Signal Process. Mag.*, vol. 13, no. 1, pp. 30–60, Jan. 1996.

[25] J. Dattorro, "Effect Design, Part 2: Delay-Line Modulation and Chorus," *J. Audio Eng. Soc.*, vol. 45, no. 10, pp. 764–788, Oct. 1997.

[26] T. W. Parks and C. S. Burrus, *Digital Filter Design*, John Wiley and Sons, Inc, New York, June 1987.

[27] J. O. Smith, *Spectral Audio Signal Processing*, online book, 2011 edition, http://ccrma.stanford.edu/~jos/sasp/, 2011.

[28] M. Liu, X. Huang, and G. Xue, "Effects of double layer porous asphalt pavement of urban streets on noise reduction," *Int. J. Sustainable Built Environment*, vol. 5, no. 1, pp. 183–196, June 2016.