Applied Computer Science, vol. 14, no. 3, pp. 5–14
doi:10.23743/acs-2018-17

Submitted: 2018-08-16
Revised: 2018-09-05
Accepted: 2018-09-10

audio fade, fade-out shape, real-time computing, HTML5, JavaScript

Lucian LUPŞA-TĂTARU[*]

NOVEL TECHNIQUE OF CUSTOMIZING THE AUDIO FADE-OUT SHAPE

Abstract

The fade-out sound effect designates a strict decreasing of the volume of an audio signal, starting from the detected initial level down to silence. Audio fade-outs are usually processed in off-line mode that is within audio editors, by employing different transcendental functions to set the time-related evolution of the audio level. However, the proliferation of mobile devices as well as the increasing popularity of web applications, implemented in HTML5/JavaScript, press for computing techniques suitable for real-time processing. In this context, having in view that the computation of the outputs of transcendental functions is very time-consuming, in order to construct the audio fade-out profile, a technique suitable for real-time computing is developed in the present investigation. The suggested technique is validated by means of an implementation in pure JavaScript, put forward with the purpose of immediate testing.

1. INTRODUCTION

The HTML5 version of the web core language has been designed to control multimedia content (*W3C Recommendation for HTML5*, 2017). Thus, complex web applications, implemented in HTML and JavaScript, have increased in popularity, especially in the activity areas of entertainment, education, and multimedia development (Devlin, 2012; Jacobs, Jaffe & Le Hegaret, 2012; Powers, 2011). Particularly, the HTML DOM Audio Object, brought in along with HTML5, combines advanced properties and methods that allow a straightforward real-time processing of audio content.

[*] Department of Electrical Engineering and Applied Physics, Faculty of Electrical Engineering and Computer Science, "Transilvania" University of Braşov, Bd. Eroilor No. 29, Braşov, RO-500036, Romania, https://iesc.unitbv.ro, lupsa@programmer.net, lucian.lupsa@unitbv.ro

In the field of audio engineering, the fade-out effect is related to volume processing and reflects a strictly monotonic decreasing of the audio level of the signal from a detected initial level down to silence. The fade-out sound effect is implemented in order to bring off a smooth ending of a certain audio content that is without perceivable "glitches" (clicks) (Case, 2007; Corey, 2017; Langford, 2014; Reiss & McPherson, 2015).

The fade-out length, i.e. the time interval over which the audio level is gradually reduced to zero, should be correlated with the detected music genre (Benetos & Kotropoulos, 2010; Panagakis, Kotropoulos & Arce, 2014). It has to be emphasized that long fade-outs, greater than 10 s, carried out at low absolute values of rates of change of the audio volume, are to be applied upon droning (sustained) sounds in order to enforce an obvious but extremely smooth ending (Langford, 2014; Potter, 2002).

The fade-out shapes are usually customized in off-line mode by employing various digital audio editors. The well-known preset shapes of the fade-out sound effect are of linear, exponential, logarithmic and S-curve type. More precisely, the preset shapes of the fade-out effect are determined by linear and transcendental functions (exponential, logarithm, sine) of time variable. Since a linear fade-out is performed at a constant rate of change of the audio level, smoothing the ending region of the signal is equivalent to increasing the fade-out length. This is the main reason why the fade-out shapes are also customized by means of transcendental functions of time variable. Moreover, various well-established psychophysical investigations highlight the benefits of implementing fade-outs of a variable rate of change of audio level (Case, 2007; Corey, 2017; Fastl & Zwicker, 2007; Hartmann, 1998; Roederer, 2008). In this context, it has to be emphasized that the employment of the exponential function to shape the fade-out profile determines a quick decay of the audio volume from the detected initial level down to the level at the halfway point (fade-out midpoint), followed by a soft decline of audio volume towards the end of the fade-out. Thus, by adopting the exponential shape for a fade-out effect, one perceives a smooth ending of the audio content i.e. a smooth transition to silence, without notable "clicks". Having in view that it is closely related to the manner in which sounds do naturally decay, such kind of time-related evolution of the audio level comes to be advantageous for fading-out synthesized soundscapes (Krause, 2008; Langford, 2014). The opposite of the exponential fade-out shape is the logarithmic curve shape, which determines a smooth decay of the audio volume in the beginning of the fading-out, followed by an abrupt transition to silence.

Taking into account that the computation of the outputs of transcendental functions is very time-consuming, the present investigation is directed towards customizing the shape of fade-out sound effect with a view to real-time volume processing. Such kind of approach is essential for the development of web applications requiring audio processing. By adopting a rational function that encompasses three coefficients in order to set the rate of change of the audio level,

it will be shown that the resulted fade-outs can act either as the fade-out effect of exponential type or as the fade-out effect of logarithmic type, depending on the values of the encompassed coefficients. Notice that, with the purpose of customizing the audio fade-in profile, a technique suitable for real-time computing has recently been introduced and implemented by considering that the audio volume is the output of a rational function that incorporates two coefficients (Lupsa-Tataru, 2017).

The employment of JavaScript programming language is justified by the fact that it supplements the enhanced functionality of HTML5 with a view to developing pervasive multimedia applications (Devlin, 2012; Jacobs, Jaffe & Le Hegaret, 2012; Powers, 2011). Particularly, the audio volume processing in HTML5/JavaScript is straightforward to perform by means of the "currentTime" and "volume" properties of HTML DOM Audio Object. While "currentTime" property is used to return the current playback time within the audio content, the "volume" property is used to set the audio level, considering that the level of 0 indicates silence whilst the level of 1 (the default value) corresponds to the highest volume. More precisely, the returned value of the "currentTime" property of the Audio Object is to be used as the input of the function of time variable that has been employed to customize the fade-out shape whilst the output of this function is to be used to set the value of the Audio Object "volume" property. On the other hand, the discrete-time processing can straightforwardly be accomplished by associating the "timeupdate" HTML DOM media event with the audio element.

2. THE FADE-OUT SHAPING

Since the valuation of the outputs of transcendental functions (exponential, logarithm, sine), intensively used to customize the fade-out profiles in off-line mode i.e. within digital audio editors, is very time-consuming and unsuitable for real-time computing, we consider here that the time-related evolution of the audio level during fading-out is determined by the following rational function:

$$\begin{cases} v(\tau) = \dfrac{\tau - \alpha}{\beta\tau - \gamma} \\ \tau \in [0, \tau_f] \end{cases} \tag{1}$$

wherein α, β, γ are the coefficients used to adjust the fade-out profile whilst τ_f represents the fade-out length.

Designating by t_{ref} the instant of time at which the fading-out effect is initiated, we have:

$$\tau = t - t_{ref} \tag{2}$$

One perceives that in contrast to implementing transcendental functions to shape the fade-out profile, the implementation of (1) comes to be straightforward since it does not call for additional methods to carry out complex mathematical tasks.

In order to depict the shape of a fade-out effect, function (1) has to be strictly decreasing. Hence, the following conditions have to be met:

$$v(0) = v_0 \tag{3}$$

$$v(\tau_h) = v_h \tag{4}$$

$$v(\tau_f) = 0 \tag{5}$$

$$0 < v_h < v_0 \tag{6}$$

where v_0 is the detected initial volume while τ_h is the halfway point i.e. the fade-out midpoint that is

$$\tau_h = \tau_f/2 \tag{7}$$

Taking into account (1), (3)–(5) and (7), one receives a system of three equations in the three variables (unknowns) α, β and γ, i.e.

$$\alpha/\gamma = v_0 \tag{8}$$

$$\frac{\tau_h - \alpha}{\beta\tau_h - \gamma} \equiv \frac{\tau_f/2 - \alpha}{\beta\,\tau_f/2 - \gamma} = v_h \tag{9}$$

$$\frac{\tau_f - \alpha}{\beta\tau_f - \gamma} = 0 \tag{10}$$

The solution to the system (8)–(10) provides the coefficients encompassed by (1) in terms of fade-out length τ_f, the detected initial volume v_0, and the ratio between the volume v_h at the midpoint τ_h and the initial volume, designated by variable ρ_h. More precisely,

$$\alpha = \tau_f \tag{11}$$

$$\beta = \frac{2\rho_h - 1}{\rho_h v_0} \tag{12}$$

$$\gamma = \tau_f/v_0 \tag{13}$$

where

$$\rho_h = v_h/v_0 < 1 \tag{14}$$

The derivative of function (1) yields the instantaneous rate of change of audio volume during fading-out i.e.

$$v'(\tau) \equiv \frac{dv}{d\tau} = \frac{\alpha\beta - \gamma}{(\beta\tau - \gamma)^2} \tag{15}$$

Having in view (11)–(13) and (14), respectively, it follows that

$$\alpha\beta - \gamma = \frac{\tau_f}{v_0}\left(1 - \frac{1}{\rho_h}\right) < 0 \tag{16}$$

and, implicitly,

$$v'(\tau) \equiv \frac{dv}{d\tau} < 0 \tag{17}$$

Hence, with condition (14), initially expressed by means of (6), the rational function (1), imposing the fade-out profile, is strictly decreasing, acting in the direction of decaying the audio level.

With a view to HTML5/JavaScript implementation, one has to consider that the default audio level is just the highest one, and corresponds to the value of 1 (Devlin, 2012; Lupsa-Tataru, 2017; Powers, 2011; WebPlat WG). Thus, if one assumes that the detected initial audio level is precisely the default one i.e. the highest one, then, based on (11)–(14), we have:

$$v_0 = 1$$

$$\rho_h = v_h$$

$$\alpha = \gamma = \tau_f$$

$$\beta = 2 - \frac{1}{\rho_h}$$

For this specific case, Figure 1 and Figure 2 highlight the fade-out curves i.e. the outputs of (1) for fade length $\tau_f = 8$ s and $\tau_f = 10$ s, respectively. Within each figure, the fade-out shape is determined by the value of ratio (14), which, in the present case, equals the audio level at the fade-out midpoint. Both figures clearly emphasize that the resulted fade-out effects corresponding to $\rho_h = 0.2$

and $\rho_h = 0.3$ will act similar to the fade-out of exponential type while the fade-outs obtained for $\rho_h = 0.7$ and $\rho_h = 0.8$ will act similar to the fade-out effect of logarithmic type. More precisely, the fade-out curves corresponding to $\rho_h = 0.2$ and $\rho_h = 0.3$ indicate a quick decline of the audio level in the beginning of fading-out, followed by a smooth transition to silence. On the other hand, the fade-out curves received for $\rho_h = 0.7$ and $\rho_h = 0.8$ emphasize a soft decay of the audio level in the beginning of fading-out, followed by a quick transition to silence.

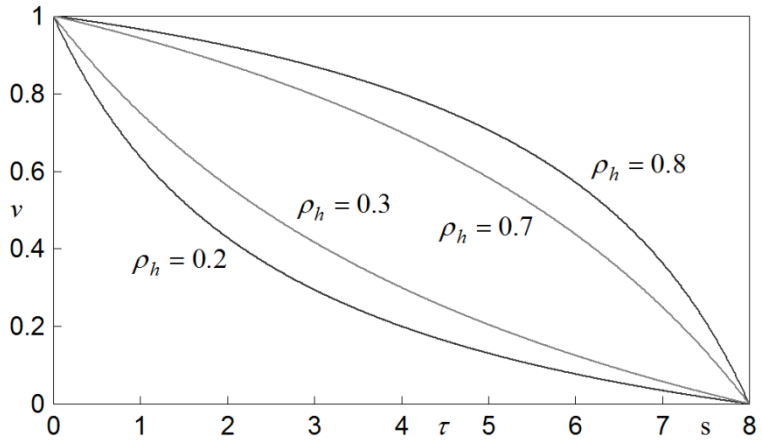

Fig. 1. Fade-out curves for fade length $\tau_f = 8$ s and the initial audio level $v_0 = 1$ (the default value in HTML5)

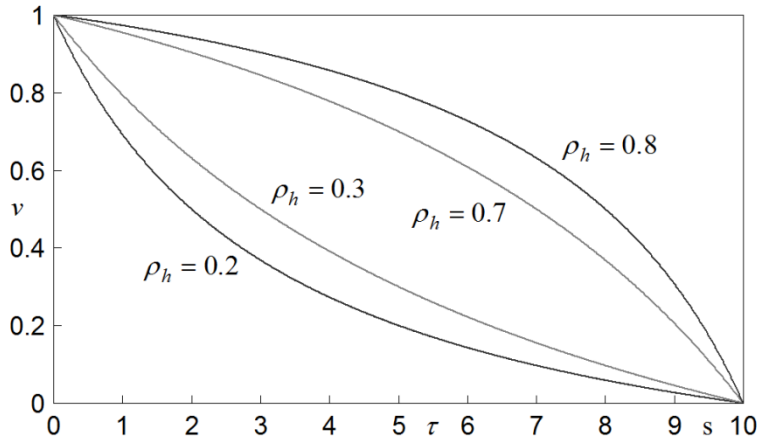

Fig. 2. Fade-out curves for fade length $\tau_f = 10$ s and the initial audio level $v_0 = 1$ (the default value in HTML5)

3. JAVASCRIPT IMPLEMENTATION

Both for the sake of simplicity and to plainly highlight the fade-out profiles that could be received by straightforwardly computing the output of (1), Figure 1 and Figure 2 have been plotted for the circumstance in which the initial audio level, occurring at the beginning of fade-out effect, is identical to the default audio level in HTML5 that, in this case, is the highest one.

However, the implementation advanced here has been structured for the most general case that corresponds to the situation in which, before the fading-out, the user is allowed to alter the audio volume during playback. This implies the computing of coefficients (11)–(13), shaping the fade-out profile, just prior to the fade-out initiation.

Although the fade-out length and the ratio (14) between the volume at the fade-out midpoint and the detected initial volume can be easily changed within the code, we have set $\tau_f = 10$ s and $\rho_h = 0.2$ in order to ensure a smooth transition to silence, without perceivable "clicks" (glitches). In this context, in Figure 3 we have illustrated the fade-out curves with the initial volume selected as parameter. As aforementioned, with a view to JavaScript implementation, the audio volume has to be located within the interval [0, 1] where the value of 0 denotes silence whilst the value of 1 indicates the highest level (the default volume). The fade-out profiles in Figure 3 clearly emphasize that the resulted fade-out effect will act similar to the fade-out effect of exponential type.

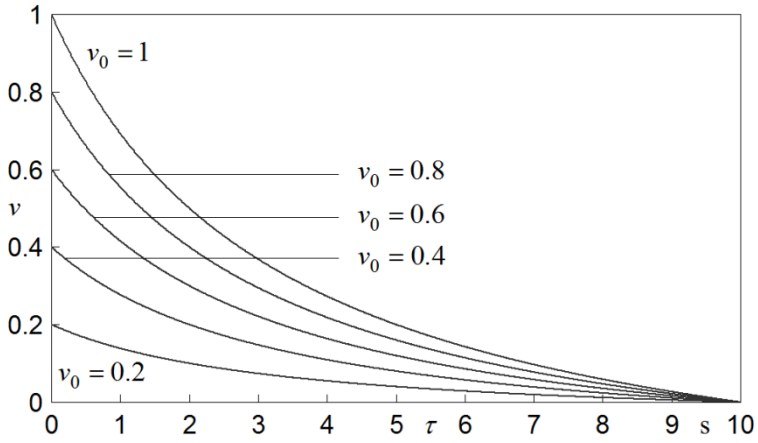

Fig. 3. Fade-out curves for fade length $\tau_f = 10$ s and ratio $\rho_h = 0.2$, with initial audio level v_0 as parameter

The code of the application developed to validate the suggested technique of audio fade-out shaping is given next, with the purpose of immediate employing.

```
<!DOCTYPE html>
<html>
<head>
   <title>Fade-out</title>
</head>
<body>
<script>
var ae = document.createElement( “AUDIO” );
                        // creates the audio element
ae.preload = “auto”;      // audio content is loaded with the page
ae.controls = true;       // displays the standard audio controls
ae.src = “sample.mp3”; // points to an audio/mpeg file
ae.addEventListener( “timeupdate”, setVol );
   // links the timeupdate event to the audio element
document.body.appendChild( ae );
   // appends the audio element to the document

var tauF = 10.0;          // fade-out length, in second
var rhoH = 0.2;
   // ratio between the volume at fade-out midpoint and the initial volume
var refTime = 20.0;
   // (expected) instant of fade-out initiation, in second
var alpha, beta, gamma;
   // the coefficients of function shaping the fade-out; global scope
var fadeOut = false;      // indicates the state of fading-out

function v( t ) {
   var newVol = ( t - alpha ) / ( beta * t - gamma );
   return newVol;
}
function setVol() {
   var tau = ae.currentTime - refTime;

   if ( fadeOut ) {
      var currentVol = v( tau );

      if ( currentVol > 0.0 ) {
         ae.volume = currentVol;
      }
      else {
         ae.volume = 0.0; ae.pause();
      }                   // end of fade-out
   }
   else if ( tau >= 0.0 ) {
      /* this block is executed only once */
      var initVol = ae.volume;
         // detects the initial volume

      alpha = tauF;
      beta = ( rhoH + rhoH - 1.0 ) / rhoH / initVol;
      gamma = tauF / initVol;
         // computes the coefficients of function shaping the fade-out
      fadeOut = true;
   }
}
</script>
</body>
</html>
```

One observes that the instant of fading-out initiation is here $t_{ref} = 20$ s. Moreover, having in view that the length of fade-out effect is 10 s, it follows that the duration of the audio content has to be greater than 30 s. Anyhow, it has to be pointed out that the initiation of the fade effect could be accomplished by means of an auxiliary event linked to the audio element, taking into account that the "timeupdate" media event is already associated with the audio element in order to achieve the discrete-time processing (Lupsa-Tataru, 2017).

To improve the computational capabilities, we have introduced the "fadeOut" variable of global scope. This variable plainly indicates the state of fading-out, i.e. it holds the value of false before the fading-out initiation whilst during fading-out, when volume processing is performed, it stores the value of true. Hence, the computation of coefficients (11)–(13) is executed only once that is the first time the returned playback position i.e. the value of "currentTime" property is greater than or equal to the expected instant of fade-out initiation. More precisely, the computation of coefficients (11)–(13), which determine the audio fade-out shape, is carried out immediately after detecting the initial value of audio volume, explicitly interfering in expressions (12) and (13), respectively.

4. CONCLUSIONS

The audio fade-out shapes are usually customized in off-line mode i.e. within various digital audio editors by employing transcendental functions in order to enforce the time-related evolution of the audio volume. On the other hand, the growing popularity of multimedia applications put forward in the form of web applications, developed in HTML5/JavaScript, has led to an increasing demand for multimedia computing techniques suitable for real-time processing.

In this context, the present investigation advances a novel technique of shaping the audio fade-out profile with a view to real-time computing. Taking into account that the valuation of the outputs of transcendental functions is time-consuming, the fade-out shape is customized, in the present paper, by means of a rational function that is a function defined by an algebraic fraction encompassing a set of three coefficients. It is shown that the resulted fade-outs, constructed by computing the output of the employed rational function, can act either as the exponential fade-out or as the fade-out of logarithmic curve shape, in accordance with the set of values adopted for the three coefficients incorporated by the algebraic fraction defining the adopted rational function.

The appropriate implementation in pure JavaScript, where the discretization has been achieved by means of the "timeupdate" HTML DOM media event, completely validates the proposed technique of audio fade-out shaping.

REFERENCES

Benetos, E., & Kotropoulos, C. (2010). Non-negative tensor factorization applied to music genre classification. *IEEE Transactions on Audio, Speech, and Language Processing, 18*(8), 1955–1967. doi:10.1109/TASL.2010.2040784

Case, A. U. (2007). *Sound FX: Unlocking the Creative Potential of Recording Studio Effects.* Burlington, MA, USA: Focal Press.

Corey, J. (2017). *Audio Production and Critical Listening: Technical Ear Training.* New York, NY, USA: Routledge.

Devlin, I. (2012). *HTML5 Multimedia: Develop and Design.* Berkeley, CA, USA: Peachpit Press.

Fastl, H., & Zwicker, E. (2007). *Psychoacoustics – Facts and Models (Springer Series in Information Sciences).* Berlin, Germany: Springer-Verlag. doi:10.1007/978-3-540-68888-4

Hartmann, W. M. (1998). *Signals, Sound, and Sensation (AIP Series in Modern Acoustics and Signal Processing).* New York, NY, USA: Springer-Verlag.

Jacobs, I., Jaffe, J., & Le Hegaret, P. (2012). How the open web platform is transforming industry. *IEEE Internet Computing, 16*(6), 82–86. doi:10.1109/MIC.2012.134

Krause, B. (2008). Anatomy of the soundscape: Evolving perspectives. *Journal of the Audio Engineering Society (JAES), 56*(1/2), 73–80.

Langford, S. (2014). *Digital Audio Editing: Correcting and Enhancing Audio in Pro Tools, Logic Pro, Cubase, and Studio One.* Burlington, MA, USA: Focal Press.

Lupsa-Tataru, L. (2017). Shaping the fade-in audio effect with a view to JavaScript implementation. *Journal of Computations & Modelling, 7*(4), 111–126.

Panagakis, Y., Kotropoulos, C. L., & Arce, G. R. (2014). Music genre classification via joint sparse low-rank representation of audio features. *IEEE/ACM Transactions on Audio, Speech, and Language Processing, 22*(12), 1905–1917. doi:10.1109/TASLP.2014.2355774

Potter, K. (2002). *Four Musical Minimalists: La Monte Young, Terry Riley, Steve Reich, Philip Glass (Series: Music in the Twentieth Century).* Cambridge, UK: Cambridge University Press.

Powers, S. (2011). *HTML5 Media.* Sebastopol, CA, USA: O'Reilly Media.

Reiss, J. D., & McPherson, A. (2015). *Audio Effects: Theory, Implementation and Application.* Boca Raton, FL, USA: CRC Press.

Roederer, J. G. (2008). *The Physics and Psychophysics of Music: An Introduction.* New York, NY, USA: Springer Science + Business Media. doi:10.1007/978-0-387-09474-8

WebPlat WG (Web Platform Working Group). (2017). *W3C Recommendation for HTML5*. W3C technical reports index: https://www.w3.org/TR/html