SopaJS 1.1.0

JavaScript library package

♡

https://staff.aist.go.jp/ashihara-k/SopaJS.html

AIST

September 23, 2016

1 Introduction

SopaJS is a JavaScript library package for reproducing panoramic sounds on the Web browser. By using SopaJS, you can make the Web site with immersive panoramic sounds and the viewer of the site can play these sounds on the Web browser and listen to them through headphones. As long as JavaScript is supported, no other plugins are needed. SOPA that stands for 'Stream Of Panoramic Sound' is a custom data format to transmit panoramic sounds. See

https://staff.aist.go.jp/ashihara-k/pan_top.html

for further information. Some examples of panoramic sound are provided in the URL below

https://staff.aist.go.jp/ashihara-k/Sopa.html

If you are interested in how to make a SOPA file, the URL below may be helpfull.

https://staff.aist.go.jp/ashihara-k/EncEval.html

SopaJS package consists of 2 JavaScript (.js) files, 2 binary (.bin) files and a test project.

Sopa.js contains source code of the library and **Sopa.min.js** is its minified version.

Two binary files are **hrtf3d512.bin** and **phase3d512.bin**. They contain the HRTF (Head Related Transfer Function) data that are necessary for rendering binaural signals.

A sample project for evaluation is also included. It contains a panoramic sound file (**cygne22k.sopa**), a JavaScript file (**sopaTest.js**) and an HTML file (**test_page.html**). The source code of these files will be mentioned later in Section 4.

2 Getting started

To use SopaJS in your Web site, put '**Sopa.min.js**' and 2 binary files (**hrtf3d512.bin** and **phase3d512.bin**) in your Web server. Link '**Sopa.min.js**' to your HTML file by using the <script> tag. If '**Sopa.min.js**' is in the upper directory of the HTML file, the HTML source code is supposed to include the following line.

```
<script src="../Sopa.min.js"></script>
```

You have to specify the locations of '**hrtf3d512.bin**' and '**phase3d512.bin**' in the HTML source code or in your JavaScript code. In the sample project, the HTML file (**test_page.html**) contains the lines below.

```
<div id="hrtfUrl" style="display:none" >../hrtf3d512.bin</div>
<div id="phaseUrl" style="display:none" >../phase3d512.bin</div>
```

In the JavaScript file (**sopaTest.js**), the locations of these files are called in lines 30 and 32 by the following code.

```
document.getElementById("hrtfUrl")
document.getElementById("phaseUrl")
```

In the sample project, the location of the SOPA file (**cygne22k.sopa**) is also specified in '**test_page.html**' line 51.

3 Usage

3.1 Constructor

In your JavaScript code, you have to construct the Sopa instance. The Sopa constructor is defined as follows.

```
Sopa = function(url){};
```

The argument 'url' is the location of a panoramic sound (.sopa) file.

A simple way to construct an instance is like the following.

```
var sopa = new Sopa("https://staff.aist.go.jp/ashihara-k/mini/railway.sopa");
```

In the example above, `https://staff.aist.go.jp/ashihara-k/mini/railway.sopa` is the location of a SOPA file ('railway.sopa' in this case).

In the sample project, the location of the SOPA file is specified in '**test_page.html**' by the 51st line.

```
<div id="sopaUrl" style="display:none">cygne22k.sopa</div>
```

The Sopa instance is constructed in '**sopaTest.js**' by lines 22 and 23.

```
var sopaElem = document.getElementById("sopaUrl");
sopa = new Sopa(sopaElem.innerHTML);
```

3.2 Methods

To reproduce a SOPA file, at least the following methods have to be used.

```
Sopa.loadDatabase = function(hrtf, phase)
Sopa.loadSopaData = function()
Sopa.setup = function()
Sopa.Play = function()
```

Sopa.loadDatabase() starts loading the HRTF data from the binary (.bin) files whose locations are specified by the arguments 'hrtf' and 'phase.'

Sopa.loadSopaData() starts loading the SOPA data from the SOPA file whose location was used in the construction of the Sopa instance.

Sopa.setup() has to be called after loading the HRTF data and SOPA data has been completed. It returns 'false,' either when 'AudioContext' is not supported or when the SOPA data are not available. If it returned 'true,' you are ready to start reproducing panoramic sound.

To start reproducing panoramic sound, what you have to do is just to call **Sopa.Play()**. Please note that **Sopa.Play()** has to be called after **Sopa.setup()** returned 'true.' **Sopa.Play()** can be used also to stop the reproduction.

The following additional (public) methods are also available.

```
Sopa.beingPlayed = function()
Sopa.currentOffset = function()
Sopa.databaseReady = function()
Sopa.fftWinSize = function()
```

Sopa.getSampleRate = function()
Sopa.setCardioid = function(card, focusHor, focusVer)
Sopa.setLastLoop = function(bool)
Sopa.setPan = function(deg)
Sopa.setUrl = function(url); Sopa.setTilt = function(deg)
Sopa.totalOffest = function()

You can check if the SOPA data are being played or not by **beingPlayed()** since it returns 'true' only while the SOPA data are being played.

CurrentOffset() returns the current sample position of the SOPA data.

DatabaseReady() returns 'true' if the database (.bin) files are loaded successfully.

FftWinSize() returns FFT window size if the SOPA file is loaded successfully. When it returns '0,' it means either the SOPA file is not yet loaded or a network error is caught.

You can get the sampling rate of the SOPA data by calling **getSampleRate()**.

In the reproduction of panoramc sounds, there are 3 directionality modes. They are 'Omnidirectional,' 'Cardioid,' and 'Cardioid^2.' You can switch directionality modes by **setCardioid(card, focusHor, focusVer)**. The first argument has to be either 0, 1, or 2. If it is 0, 'Omnidirectional' mode is selected. 1 is for 'Cardioid' and 2 is for 'Cardioid^2.' The second and third arguments are the horizontal angle and vertical angle of the target direction in either 'Cardioid' and 'Cardioid^2' directionality modes. They have to be given in radians. If both of them are 0, the frontal (facing) direction is the target of the cardioid directionality. You can try these directionality modes in the URL below.

`https://staff.aist.go.jp/ashihara-k/vcm.html`

A SOPA file will be played in an endless 'loop playback' mode. If you want to play it in the 'single play' mode, you can do this by calling **setLastLoop(true)**. If **setLastLoop(true)** has been called, the reproduction automatically stops at the end of the SOPA file.

By using **setPan(deg)**, you can manipulate pan of panoramic sound. The augument 'deg' is an integer between -180 and 179.

By using **setTilt(deg)**, you can manipulate tilt of panoramic sound. The augument 'deg' is an integer between -90 and 89.

Although the location of the SOPA file has to be specified when the instance is constructed, it can be changed afterward by calling **setUrl(url)**. The argument 'url' is the location of the SOPA file.

TotalOffset() returns the current sample position in the reproduction. In the loop playback mode, the return value of this method can be greater than the total samples of the SOPA file.

4 Sample project

4.1 Outline

The sample project that consists of **cygne22k.sopa**, **sopaTest.js** and **test_page.html** is a simple project but it supports panning control so that the user can change pan of the panoramic sound while listening to panoramic sound.

Since necessary code to reproduce panoramic sound and to control panning are included in the project, it may be the starting point of your project.

The URL below provides the working demo of the project.

```
https://unit.aist.go.jp/hiri/hi-infodesign/as_js/SopaJS/
samples/test_page.html
```

Needless to say, JavaScript has to be enabled in the Web browser.

Please visit the URL below, too. You will see how panoramic sound can be combined with the panoramic image (WebGL is used).

```
https://unit.aist.go.jp/hiri/hi-infodesign/as_still/cygne22k.
html
```

In this sample project, it is assumed that the file tree looks like what is shown in the figure below.

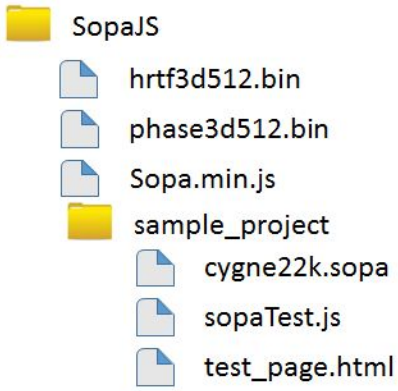

Fig. 1: File tree of the sample project

The SOPA file (**cygne22k.sopa**) and the JavaScript file (**sopaTest.js**) are in the same directory as the HTML file (**test_page.html**). Two binary files and **Sopa.min.js** is in the upper directory.

4.2 SOPA file

Cygne22k.sopa is a sample of panoramic sound in which a piano, a cello and a harp are at the directions of approximately 0°, 135°, and 240°, respectively in the horizontal plane (counter-clockwise).

4.3 HTML code

In the HTML file (**test_page.html**), the location of a SOPA file (**cygne22k.sopa**) is specified in the 51st line.

```
<div id="sopaUrl" style="display:none">cygne22k.sopa</div>
```

It is assumed that the SOPA file ('**cygne22k.sopa**' in this case) is in the same directory as the HTML file. If it is not, please modify the code adequately.

The locations of the binary (.bin) files are also given in lines 54 and 55.

```
<div id="hrtfUrl" style="display:none">../hrtf3d512.bin</div>
<div id="phaseUrl" style="display:none">../phase3d512.bin</div>
```

It is assumed that these binary files are in the upper directory.

Sopa.min.js and **sopaTest.js** are linked by lines 57 and 58.

```
<script src="../Sopa.min.js"></script>
<script src="sopaTest.js"></script>
```

Sopa.min.js is assumed to be in the upper directory.

In **sample_page.html**, four <button> elements are defined. They will be used in **sopaTest.js**.

4.4 JavaScript code

In the sample project, '**sopaTest.js**' deals with a Sopa instance whose name is 'sopa' (line 23) to play panoramic sounds and control panning.

In constructing the Sopa instance ('sopa' in this case), the location of '**cygne22k.sopa**' is used as the argument.

```
sopa = new Sopa(sopaElem.innerHTML);
```

In line 37, **sopa.loadDatabase()** is called with arguments 'hrtfStr' and 'phaseStr.' These arguments are the locations of the binary files that contain the HRTF data.

Immediately after **sopa.loadDatabase()** is called, **sopa.loadSopaData()** is called in line 38.

```
sopa.loadSopaData();
```

This method starts loading the SOPA file ('**cygne22k.sopa**' in this case).

In lines 113 and 114, 'PLAY' button is enabled after confirming that loading data was completed successfully by checking **sopa.fftWinSize()** and **sopa.databaseReady()** in line 111.

Clicking on 'PLAY' button is followed by **sopa.setup()** (line 71) and if it returns 'true,' **sopa.Play()** is called (line 75) to start the reproduction.

Sopa.Play() is called again in line 87. This time, it is to stop the reproduction. In this sample project, the sound is played in a 'loop payback' mode. The listener, therefore, has to click the 'Stop' button to stop the reproduction.

If you want to play the sound in the 'single play' mode rather than the 'loop playback' mode, call **sopa.setLastLoop(true)** in your JavaScript code either before or during the first loop.

Panning of sound can be controlled easily by the method '**Sopa.setPan(deg)**.' See lines 46 and 59. The argument ('anglHor' in this case) is the horizontal angle. It should be between -180 and 179. In the sample project, **sopa.setPan(anglHor)** is triggered by click on the buttons.

Although in this project, only the horizontal angle (pan) can be changed, you can also change the vertical angle (tilt) by using the method '**Sopa.setTilt(deg)**.' In that case, 'deg' has to be between -90 and 89.

Sopa.setCardioid(toggle,focusHor,focusVer) is called in lines 36 and 97 to change the directionality mode. The first argument has to be either 0, 1, or 2. 0 is for 'Omnidirectional.' 1 and 2 are for 'Cardioid' and 'Cardioid^2,' respectively. It is toggled by click on the button in the sample project.

The second and third arguments are the horizontal and vertical angles of the target direction in radians. In the sample project, variables 'focusHor' and 'focusVer' are always 0. That means the longitude and latitude of the target direction are 0 (the facing direction). In 'Cardioid' and 'Cardioid^2' modes, therefore, sounds from the listener's rear are suppressed.

License

ashihara-k@aist.go.jp