Synervoz Voice FX Extension for Agora

Created: August 24, 2021
Last Update: March 30, 2023

Product overview

Introduction

This extension provides four voice filters to users: echo, reverb, flanger and pitch shift. The user can customize values for each one of them to achieve different results.

Extension workflow

Users can enable disable the extension by using AgoraRTCEngine Kit method `enableExtensionWithVendor:@"Synervoz" extension:@"VoiceFilter" enabled:<true or false>`

Users also have to enter the apiKey and apiSecret:

```
[self.agoraKit setExtensionPropertyWithVendor:@"Synervoz"
extension:@"VoiceFilter" key:@"apiKey" value:@"EXAMPLE_API_KEY"];
[self.agoraKit setExtensionPropertyWithVendor:@"Synervoz"
extension:@"VoiceFilter" key:@"apiSecret" value:@"EXAMPLE_API_SECRET"];
```

Users also have a helper class available, called **SynervozVoiceFilter**, where they can control each filter.

Features

As noted above, this extension provides four filters, each of which has parameters that can be tweaked to alter what they sound like:

- **Echo**: Turning this filter on, users will have an echo filter processed when they speak;
- **Reverb**: Turning this filter on, users will be able to hear how their voices sound if they were in a room.
- **Flanger**: Flanging is a commonly used effect when making music. It could be used in the context of Agora for a singing app, among other purposes. It ultimately duplicates the signal and plays back the second with a short delay that varies in time.
- **Pitch Shift**: Turning this filter on will change the pitch of a user's voice. This might be used just for fun (e.g. sound like Darth Vader or a robot), to mask identity, or perhaps to

be used in conjunction with a time stretching effect (e.g. playing back audio at 1.5x, then lowering the pitch).

To see what each effect sounds like, you can download samples at https://www.synervoz.com/devtools/agora

Supported Platforms

The development environment has to meet the following requirements:

Android
- Android Studio 3.0 or later
- Android SDK API Level 21 or higher.

iOS
- Xcode 10.0 or later
- An iOS device running iOS 11.0 or later.

Pricing

This extension is provided for free.

Integration guide

Overview

This extension only requires three basic steps:
1. Import the VoiceFilters extension to your project
2. Create an AgoraRtcEngine kit instance, initialize it and enable the extension
3. Create a SynervozVoiceFilter instance to control the voice filters.

Prerequisites

The only prerequisite needed is knowledge of AgoraRtcEngineKit's basic functions.

How to implement

Android

1. Add the `VoiceFiltersAgoraExtension.aar` to your project
2. Import the following

```
import com.synervoz.voicefilters.SynervozVoiceFilter
import com.synervoz.voicefilters.AgoraExtensionPropertyInterface
import com.synervoz.voicefilters.SynervozVoiceFilterExtensionManager
import com.synervoz.voicefilters.VoiceEffectsComponentConfiguration
```

3. Create an instance of `SynervozVoiceFilter` and call `setAssetManager`

```
var synervozVoiceFilter = SynervozVoiceFilter()
synervozVoiceFilter.setAssetManager(applicationContext.assets)
```

4. Create an instance of `RtcEngine` by initializing it with RtcEngineConfig() and adding the extension calling `addExtension`

```
val config = RtcEngineConfig()
config.addExtension(SynervozVoiceFilterExtensionManager.EXTENSION_NAME)
mRtcEngine = RtcEngine.create(config)
```

To enable the extension, call RtcEngine's function `enableExtension`

```
mRtcEngine.enableExtension(
   SynervozVoiceFilterExtensionManager.EXTENSION_VENDOR_NAME,
   SynervozVoiceFilterExtensionManager.EXTENSION_AUDIO_FILTER_NAME,
   true
)
```

5. Create an instance of the SynervozVoiceFilter class.
6. Implement the AgoraExtensionPropertyInterface as following

```
synervozVoiceFilter.agoraExtensionPropertyInterface =
   AgoraExtensionPropertyInterface { vendorName, filterName, propertyName,
propertyValue ->
       mRtcEngine.setExtensionProperty(vendorName, filterName,
propertyName, propertyValue)
```

```
   }
```

7. To enable reverb (all filters have default values)

```
synervozVoiceFilter.reverbVoiceConfig =
   VoiceEffectsComponentConfiguration.
     ReverbVoiceEffectConfiguration().apply {
       dry = 1.0f
       wet = 1.0f
       mix = 1.0f
       width = 1.0f
       damp = 0.5f
       roomSize = 1.0f
       preDelay = 100.0f
       lowCut = 5.0f
   }
synervozVoiceFilter.enableReverb(reverbEnabled)
```

iOS

1. Add the `VoiceFiltersAgoraExtension.framework` to your project

2. Create an instance of `AgoraRtcEngineKit` and initialize it.

3. To enable the extension, call AgoraRtcEngineKit's function `enableExtensionWithVendor`
    a. For vendor, set **Synervoz**
    b. For extension, set **VoiceFilter**

4. Import SynervozVoiceFilter class

```
     //Objective-C
     #import <VoiceFiltersAgoraExtension/SynervozVoiceFilter.h>

     //Swift
     import VoiceFiltersAgoraExtension
```

5. Create an instance of `SynervozVoiceFilter` class.

6. To enable the echo voice filter and change its configuration

```
      //Objective-C
      EchoVoiceEffectConfiguration echoConfig;
      echoConfig.mix = 1.0f;
      [voiceFilter setEchoConfiguration:echoConfig inEngine:self.agoraKit];
      [voiceFilter enableEcho:YES inEngine:self.agoraKit];

      //Swift
      EchoVoiceEffectConfiguration echoConfig =
EchoVoiceEffectConfiguration(mix: 1.0)
      voiceFilter.setEcho(echoConfig, inEngine: agoraKit)
      voiceFilter.enableEcho(true, inEngine: agoraKit)
```

7. To enable the pitch shift filter and change its configuration

```
      //Objective-C
      PitchShiftVoiceEffectConfiguration pitchShiftConfig;
      pitchShiftConfig.shift = 500;
      [voiceFilter setPitchShiftConfiguration:pitchShiftConfig
inEngine:self.agoraKit];
      [voiceFilter enablePitchShift:YES inEngine:self.agoraKit];

      //Swift
      PitchShiftVoiceEffectConfiguration pitchShiftConfig =
PitchShiftVoiceEffectConfiguration(shift: 500)
      voiceFilter.setPitchShiftConfiguration(pitchShiftConfig,
inEngine:agoraKit)
      voiceFilter.enablePitchShift(true, inEngine:agoraKit)
```

8. To enable the reverb filter and change its configuration

```
      //Objective-C
      ReverbVoiceEffectConfiguration reverbConfig;
      reverbConfig.damp = 1.0f;
      reverbConfig.dry = 1.0f;
      reverbConfig.lowCut = 5.0f;
reverbConfig.mix = 0.7f;
      reverbConfig.preDelay = 100.0f;
      reverbConfig.roomSize = 1.0f;
      reverbConfig.wet = 1.0f;
      reverbConfig.width = 1.0f;
      [voiceFilter setReverbConfiguration:reverbConfig
inEngine:self.agoraKit];
      [voiceFilter enableReverb:YES inEngine:self.agoraKit];
      //Swift
      ReverbVoiceEffectConfiguration reverbConfig = 
ReverbVoiceEffectConfiguration(dry: 1.0, wet: 1.0, mix: 1.0, width: 1.0, 
damp: 1.0, roomSize: 1.0, preDelay: 100.0, lowCut: 200.0)
      voiceFilter.setReverbConfiguration(reverbConfig, inEngine: agoraKit)
      voiceFilter.enableReverb(true, inEngine: agoraKit)
```

9. To enable the flanger filter and change its configuration

```
      //Objective-C
      FlangerVoiceEffectConfiguration flangerConfig;
      flangerConfig.bpm = 80.0f;
      flangerConfig.depth = 1.0f;
      flangerConfig.lfoBeats = 30.0f;
      flangerConfig.stereo = true;
      flangerConfig.wet = 1.0f;
      [voiceFilter setFlangerConfiguration:flangerConfig 
inEngine:self.agoraKit];
      [voiceFilter enableFlanger:sender.isSelected inEngine:self.agoraKit];

      //Swift
      FlangerVoiceEffectConfiguration flangerConfig = 
FlangerVoiceEffectConfiguration(wet: 1.0, depth: 1.0, lfoBeats: 30.0, bpm: 
100.0, stereo: true)
        voiceFilter.setFlangerConfiguration(flangerConfig, inEngine: 
agoraKit)
        voiceFilter.enableFlanger(true, inEngine: agoraKit)
```

API Reference

iOS/Android

- Classes:
  - **SynervozVoiceFilter**: The helper class that allows you to control the voice filters and their configuration. With this class you can enable or disable each filter, change their configuration using the configuration structs and reset the values to default.
- Structs:
  - **EchoVoiceEffectConfiguration:** Holds the echo filter configuration values.
  - **ReverbVoiceEffectConfiguration**: Holds the reverb filter configuration values.
  - **FlangerVoiceEffectConfiguration**: Holds the flanger filter configuration values.
  - **PitchShiftVoiceEffectConfiguration**: Holds the pitch shift configuration values.