Sample Code

Generating spatial audio from a multichannel audio stream

Convert 8-channel audio to 2-channel spatial audio by using a spatial mixer audio unit.

Download

iOS 16.0+ | iPadOS 16.0+ | macOS 13.0+ | Xcode 15.4+

Overview

Spatial audio makes sound seem like it's coming from all around you. To generate spatialized audio from a multichannel audio stream, you use a spatial mixer audio unit (AUSM). The audio streams can come from the output of a game engine, video player, or audio file.

The sample app shows you how to create an AUSM, and how to configure its channel layout and stream format. It streams multichannel input from an audio file, and down-mixes it to 2-channel spatial audio.

Create an audio unit spatial mixer

The sample uses the pull model to get the input from an in-memory file streamer. The `OutputAU` class pulls input from the `AudioKernel`, which manages the AUSM. The AUSM pulls audio from `AudioFileReader` to get input. On macOS, the output unit uses `kAudioUnitSubType_HALOutput` to interface with the audio device. On iOS, the interface is configured as `kAudioUnitSubType_RemoteIO`.

To begin setup of an AUSM, the sample initializes a mixer with a subtype of `kAudioUnitSubType_SpatialMixer`.

```
AudioComponentDescription auDescription = {kAudioUnitType_Mixer,
                                           kAudioUnitSubType_SpatialMixer,
                                           kAudioUnitManufacturer_Apple,
                                           0,
                                           0};
AudioComponent comp = AudioComponentFindNext(NULL, &auDescription);
assert(comp);

OSStatus err = AudioComponentInstanceNew(comp, &mAU);
assert(err == noErr);
```

After initializing the spatial mixer, the sample calls a convenience function to configure the output stream format and channel layout for stereo audio.

```
err = setStreamFormatAndACL(inOutputSampleRate, kAudioChannelLayoutTag_Stereo, kAudi
assert(err == noErr);
```

The sample then configures the input stream format and sets the channel layout to kAudioChannelLayoutTag_MPEG_7_1_A for 8-channel audio.

```
err = setStreamFormatAndACL(inInputSampleRate,
                            kAudioChannelLayoutTag_Atmos_7_1_4,
                            kAudioUnitScope_Input,
                            elem);
assert(err == noErr);
```

Configure for spatial audio

The sample sets the spatialization algorithm to AUSpatializationAlgorithm.spatializationAlgorithm_UseOutputType for the highest-quality spatial rendering across different hardware.

```
UInt32 renderingAlgorithm = kSpatializationAlgorithm_UseOutputType;
err = AudioUnitSetProperty(mAU,
                           kAudioUnitProperty_SpatializationAlgorithm,
                           kAudioUnitScope_Input,
                           elem,
                           &renderingAlgorithm,
                           sizeof(renderingAlgorithm));
assert(err == noErr);
```

The input channels are spatialized around the listener as far-field sources. The channel layout specifies the relative directions of the individual channels. The azimuth and elevation parameters control the audio rotation. The sample configures the source mode to `AUSpatialMixerSourceMode.spatialMixerSourceMode_AmbienceBed`. Use `AUSpatialMixerSourceMode.spatialMixerSourceMode_PointSource` for an object to render the input signal as a single source, except if rendering in-head with `AUSpatialMixerPointSourceInHeadMode.spatialMixerPointSourceInHeadMode_Bypass`.

```
UInt32 sourceMode = kSpatialMixerSourceMode_AmbienceBed;
err = AudioUnitSetProperty(mAU, kAudioUnitProperty_SpatialMixerSourceMode, kAudioUni
assert(err == noErr);
```

Spatial audio includes dynamic head tracking for apps that enable it. To configure head tracking, set `kAudioUnitProperty_SpatialMixerEnableHeadTracking`. For more information about supported devices, see Listen with spatial audio for AirPods and Beats.

```
UInt32 ht = 1;
err = AudioUnitSetProperty(mAU,
                           kAudioUnitProperty_SpatialMixerEnableHeadTracking,
                           kAudioUnitScope_Global,
                           0,
                           &ht,
                           sizeof(UInt32));
```

See Also

Audio Units

- Audio Unit v3 Plug-Ins

Deliver custom audio effects, instruments, and other audio behaviors using an Audio Unit v3 app extension.

Audio Components

Find, load, and configure audio components, such as Audio Units and audio codecs.

Audio Unit v2 (C) API

Configure an Audio Unit and prepare it to render audio.

Audio Unit Properties

Obtain information about the built-in mixers, equalizers, filters, effects, and other Audio Unit app extensions.

Audio Unit Voice I/O

Configure system voice processing and respond to speech events.