3D SOUND FOR 3D GAMES - USER GUIDE

INTRODUCTION

This 3D sound engine plugin for Unity will transform your 3D games and give you the edge in user engagement and experience. Developed after over 25 years research into bat and human echolocation; it taps into the brains capability of positioning sound in 3D space.

This software will give you the full three dimensions of sound; all you have to provide is the sound clip, listener and sound source game objects, and set up a few parameters. The sound engine does all the work by positioning the sound with respect to the listener and object's position in 3D space.

But the key feature of this plugin is its ability to play multiple sound sources in 3D without overloading the CPU. It can comfortably play around 50 3D sounds simultaneously on a Smartphone (many more on a PC), which would be useful in an interactive and highly charged part of a game.

THE PLUGIN INSTALLATION

The effect plugin will be available for download via the Unity Asset Store. It will appear as two DLLs; one for the editor interface, and the other for the sound engine itself. These will be integrated into two demos; a room with six independent sound sources (spheres) bobbing up and down, and a room with 30 flies darting about.

Once the files have been successfully imported into your game or application, you can add the '3D Sound For 3D Games' component via two menu options. First select the game object in the Hierarchy section of Unity you wish to attach the component to. Then either select from the top menu 'Component->Echolocation Ltd->3D Sound For 3D Games', as per Figure 1:

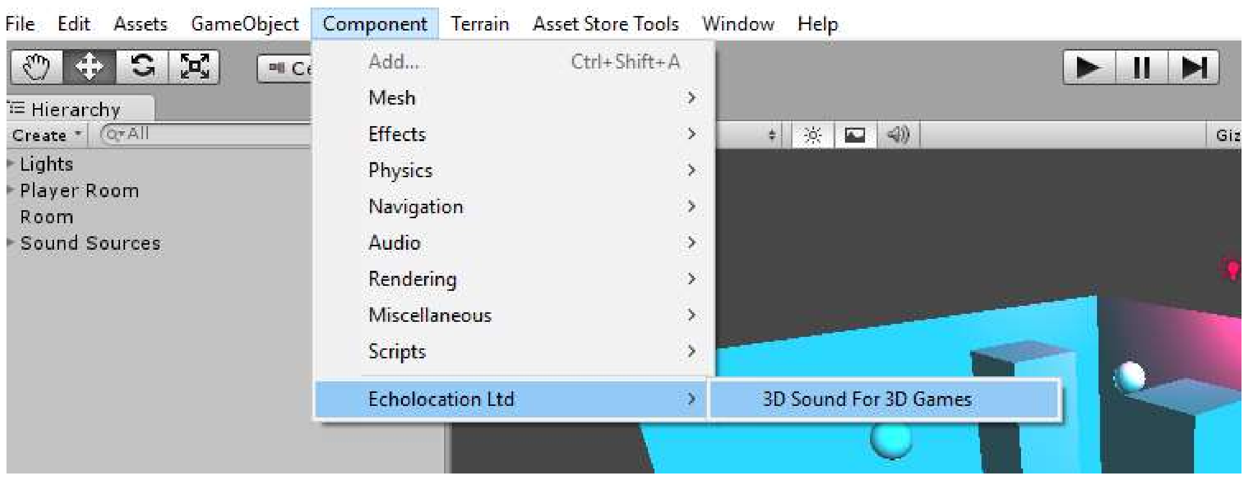


Figure 1

Or select from the 'Add Component' button in the Inspector and navigate to 'Echolocation Ltd->3D Sound For 3D Games', as per Figure 2.

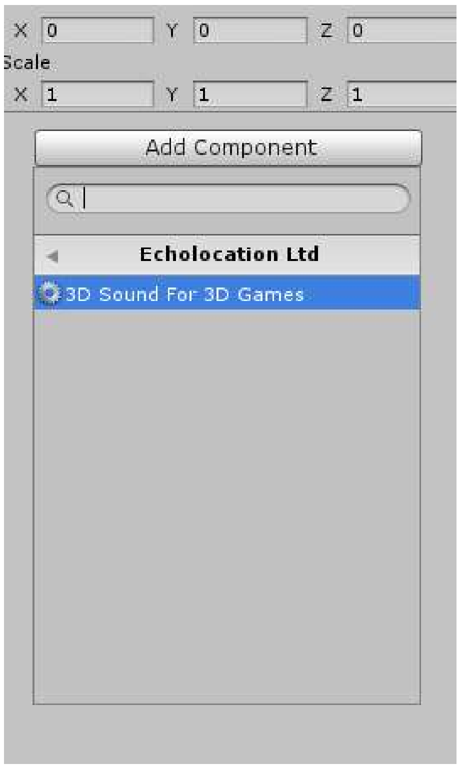
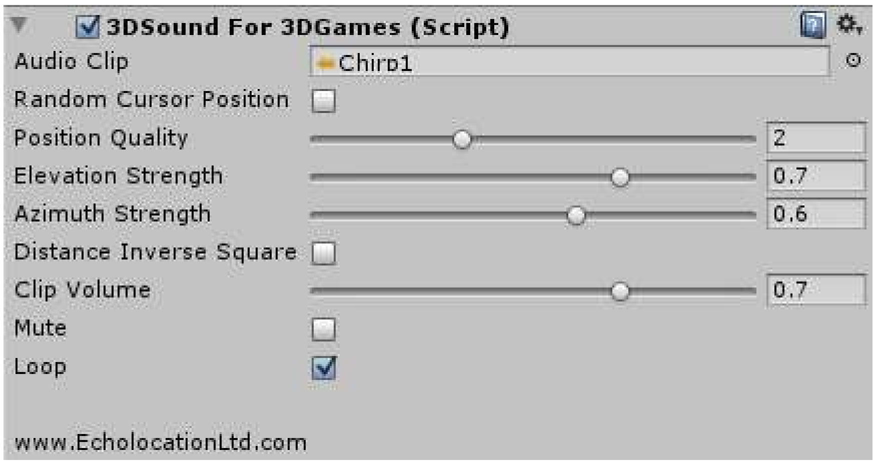


Figure 2

THE PLUGIN INSPECTOR EXTENSION

Figure 3 shows a screenshot of the plugin after it's been installed and attached to a game object.



3DSound For 3DGames (Script)

Audio Clip	Chirp1
Random Cursor Position	☐
Position Quality	2
Elevation Strength	0.7
Azimuth Strength	0.6
Distance Inverse Square	☐
Clip Volume	0.7
Mute	☐
Loop	✓

www.EcholocationLtd.com

Figure 3

The first option available is to provide a link to your chosen sound clip. Note that this clip must be stereo, and sampled at 44.1 kHz or above. The latter is necessary because the sound engine requires a wide bandwidth in order for it to function well, so the richer the sound is in harmonics, the better will be the positioning.

The Random Cursor Position tick box simply randomises the start position of the playback of a clip. This is essential if you have lots of cloned game objects that all start playing from the same clip at once. They will all be synchronised which will destroy the 3D effect, so you need to randomise their start positions to make them sound more natural. It is normally off by default, but it's use is demonstrated well in the Fly Demo scene.

The next option is the Position Quality, which is selectable between 1 through 4 in integer steps. The higher the value, the smoother will be the transition in sound positioning, but at the expense of CPU time. The optimal quality level is 2, which is the default; but if you have a rich sound passing close by, you might hear step changes in the position.

The Elevation Strength is provided so you can fine-tune the sounds positioning strength in the vertical plane. A setting of ‘0’ will provide a very flat sound, but increasing it will add the height component. You won’t need this setting to be very strong over headphones, but may need to increase it if you are listening over loudspeakers.

The Azimuth Strength can be used to alter the width of the sound. A setting of ‘0’ gives zero panning, but increasing it increases the panning up to a maximum at ‘1’. This is useful if you want to fine-tune the horizontal width of the sound engine, or if you want the panning effect to work effectively of loudspeakers (i.e. setting it to ‘1’). The optimum is ‘0.6’ for headphones.

The Distance Inverse Square tick box allows the sound to attenuate at a natural rate of the inverse square of the distance. Leaving it unticked leaves the attenuation at a linear proportion to the distance.

The Volume control does what is says, but note that it is the overall volume controller for the clip, i.e. it controls the volume level for all the attached game objects. You can control the sound level for each individual game object, but this is only achieved via a script.

Mute simply mutes the sound and Loop allows the clip to be looped.

One final comment about the Editor Extension; when you hit the play button, unless a sound source object has not already been added, the plugin will immediately create a sound source below the extension.

CONTROLLING THE PLUGIN VIA C# SCRIPT

The plugin currently only works with C# scripting.

The only functions that you will need to use are the following:

Setup3D(int layer)
SetListenerPosition(GameObject goListener)
Play3D(GameObject go)
Play3D(GameObject go, float volume)

Before calling these methods, you need to find or get a link to the game object you want to position in 3D sound space. Once this has been collected, you need to get a copy of the sound engine component. From here you call the first method Setup3D (in Start()); passing it the layer[s] the environment is in. This is for sound occlusion purposes and allows the code to ignore any objects that don’t physically get in the way. Plus it will speed up the processing in sophisticated environments. Further information on layer masking can be found in the Unity documentation.

You can then start frequently calling the SetListenerPosition and Play3D functions, passing them the game object components, and optionally the volume of the sound for the game object for the latter (default is full volume).

Note that the sound engine will keep a record of these game objects, but frequent calls to both will be required to update the game object’s position information. Also, once the Play3D’s game object has been destroyed, the sound engine will forget about it, and will stop playing the sound linked with that object.

Also note that in addition to the plugin keeping track of the SetListenerPosition and Play3D game objects; it also keeps track of the playback position of the clip for the various game objects. So if one game object calls the method at time = 0 and the clip immediately starts playing; another game object can call the same method at time = 1 second, and the second version of the clip will start playing in a different 3D position 1 second later. The sound will thus be out of synch; improving the realness and richness of the sound.

THE 3D ROOM DEMO

The plugin is accompanied by two demos; with one that contains a 3D room with six independent spherical sound sources moving up and down; with each sphere emitting a different sound. You can navigate through the room using the arrow keys, and look using the mouse. You should be able to hear the sounds move up and down and left and right as you move and look around the room. You should also be able to appreciate a sound's distance and occlusion by the pillars in the room. The best effect can be heard over headphones, but it still works well over loudspeakers.

The sound engine plugin keeps track of the objects through constant calls to the SetListenerPosition and Play3D functions via the Update() method in the AudioControl script. When the play button is pressed, the plugin creates its own sound source, and starts to play the sound specified in the editor; positioning this sound with respect to the position of the game object and listener in 3D world space.

THE FLY DEMO

The second demo shows how the plugin can be used with cloned game objects. It contains a fly prefab, and a script that randomises the direction of the fly after a pre-set time interval. However, note that the 3D plugin is not attached to the fly game object, as it is in the six sound sources in the previous demo. This is because we need a separate game object for the sound source, and pass the cloned game objects to the plugin via a script; otherwise we will be creating endless instances of the plugin.

Attached to the Sound Source is an Audio Control script that looks after these cloned flies from the prefab, and you can increase or decrease the number of flies in the scene. Decreasing it to 1 is a good demonstration of the sound engine in action; allowing the fly to zoom around your head is quite creepy. Then increasing the number of flies starts to sound like a swarm, until the CPU cannot cope with the number of 3D sound positions anymore. On my i7 quad core PC with sound card, I've been able to reach 500+ flies using this demo, but that was only at 30% CPU capacity. There's therefore a capability of reaching 1500+ 3D sound sources, which would be needed for good room-modelled real-time reverberation. This will be available in the Pro version of the plugin.

USING THE PLUGIN IN YOUR GAME

All you require for this plugin to work in Unity 4.0 and beyond are two DLLs (in the Editor and Plugins folders). And since the plugins are free for commercial use (under the Unity EULA), you can now explore efficient and effective 3D sound in all your games for all platforms offered by Unity. Happy building!

If you have any comments or suggestions for further enhancements, please email them to echolocationltd@gmail.com.