

Re: DirectShow DirectSound microphone low-latency

Source:

<http://www.tech-archive.net/Archive/Development/microsoft.public.win32.programmer.directx.audio/2007-09/msg00>

- *From:* "Chris P." <msdn@xxxxxxxxxxxxx>
 - *Date:* Mon, 17 Sep 2007 16:05:30 -0400
-

On Sun, 16 Sep 2007 16:12:02 -0000, insidemethods@xxxxxxxxxx wrote:

I have it working better now by checking at the end of FillBuffer for the next network packet and checking to see if more than 1 packet is buffered. If so, then I assume we are already behind. We are in fact behind, but what to do about it?

Considering that we want to have the lowest latency possible — if we are behind, then we have to somehow "catch up".

This sounds gross but what I decided is to just throw away the oldest packet(s). It is a little gross, maybe a better scheme would be to speed up the rendering to faster than real-time (by adjusting the timestamps) which might work OK for video and not be too noticeable, but audio is might not fare so well.

For video you can adjust the timestamps and it will speed up or slow down the presentation (but not for what is already buffered in samples). You can throw frames away as well but this creates problems for subsequent frames at the decoder.

Previously I was doing something like this, but only the first time Fillbuffer was called, and that just sort of worked. Now it is working much more smoothly.

The main problem is due to dropping video packets. The video will look bad until the next key frame. I have tried continuing to drop packets till the next key frame — that's not too bad. It might be better not to do this, especially if the key frame rate is high.

If frames are lost you have the choice of not presenting frames until the next keyframe or continuing to pass delta frames for decoding that may cause unusual artifacts. Depending on the decoder and the format it may

Re: DirectShow DirectSound microphone low-latency

not present anything until the next keyframe anyway.

There has to be a better way, but how practical is it? The problem still seems to be mostly at the beginning (as though there is extra CPU overhead as the video decoder is processing the first batch of packets?). But it does still happen later, occasionally. Without a scheme like above, or doctoring the timestamps or something, then we will gradually get more and more latency (which doesn't meet the spec), until we run out of extra buffers.

You can always create a live clock source in your filter (complicated), see "Live Sources" in the DirectShow documentation. You can also just doctor the time stamps, which amounts to the same end result.

<http://www.chrisnet.net/code.htm>

[MS MVP for DirectShow / MediaFoundation]

.