Subject: VideoPlayerCtrl using OpenGL + ffmpeg + portaudio
Posted by jjacksonRIAB on Thu, 15 Jul 2021 00:36:48 GMT
View Forum Message <> Reply to Message

It seems to play a good portion of what I throw at it but it also can develop Audio/Video sync issues and some videos flat out refuse to play. So what I will say is there is a lot of room for improvement, but this is my first attempt at anything like this and I learned a lot on the way but suggestions for improvements would be very welcome and if someone wants to go another direction, feel free to see if any of the code in there is useful for your own video control.

Notes:
This has not been tested on Windows at all and the dependencies are not satisfied.
It requires installation of: avcodec avformat avutil swscale swresample and portaudio

Location of packages
https://github.com/BornTactical/AudioPackage
https://github.com/BornTactical/VideoPackage
https://github.com/BornTactical/VideoCtrlTest
https://github.com/BornTactical/VideoCtrl

TO DO:
It needs volume control.
Sync issues need to be ironed out.
Some videos segfault.
CPU usage could be improved.
Windows support would be nice.
Support rendering surfaces other than just OpenGL.

I decided to release this early because I'm feeling a bit out of my league on it.

Subject: Re: VideoPlayerCtrl using OpenGL + ffmpeg + portaudio
Posted by Xemuth on Thu, 15 Jul 2021 10:55:08 GMT
View Forum Message <> Reply to Message

Hello jjacksonRIAB,

Thanks for sharing ! I wont really be able to talk about how you did it.

But I have a question concerning OpenGL. Correct me if I'm wrong but it seems you use OpenGL as a way to render your current frame on screen via a texture binding. However, AFAIK using OpenGL here seems to be overkilled for somes reasons :
- Since texture must be loaded in current VRAM, it mean at every frame you send data to VRAM (I think it's exepensive)
- Even if your frame / view don't move at all, OpenGL do a lot of calculation every frame to determine pixel position.   (again it expensive)
- OpenGL allow double buffering (even triple buffering with a bit of trickiness) but you seems to not exploit it.

- Even if you seems to use really basic openGL function, may older computer (old openGL version) will struggle to run (maybe wont) the player. (I think about
 Raspberry, if I remember well, the Pi3 support a really old openGL version on it. Maybe it wont work ? I will test it...).

as you say, "Support rendering surfaces other than just OpenGL" is a really good point to do. Have you planned to share this Ctrl on UppHub ?

Subject: Re: VideoPlayerCtrl using OpenGL + ffmpeg + portaudio
Posted by jjacksonRIAB on Thu, 15 Jul 2021 12:31:16 GMT
View Forum Message <> Reply to Message

Xemuth,
Yes, you're right on all points. What I was eventually hoping to use GL for was supporting accelerated color YUV->RGB conversion and hardware scaling but as you notice it uses a cached static SwScale context and the texture resizing that I thought was accelerated actually turns out to be CPU-based anyway. It also uses an ImageBuffer as the backing buffer so it could be easily adapted to work without OpenGL at all. One major thing that that I suspect would kill performance is that decoders ideally operate already in GPU-space and GPU memory so if you start out in that area you want to stay in GPU instead of bringing rendered frames back into CPU-side and then putting them back in the GPU again as GL textures.

Sure, I'd be happy to put it on UppHub.

Subject: Re: VideoPlayerCtrl using OpenGL + ffmpeg + portaudio
Posted by jjacksonRIAB on Thu, 15 Jul 2021 12:47:47 GMT
View Forum Message <> Reply to Message

So to clarify my understanding in an ideal scenario you want as much as possible in the GPU especially in the case of mobile devices. The majority of your budget is spent in decoding, not blitting, and mobile devices would not be able to decode certain types of videos at all without acceleration. If there is a way to provide for a pipeline that decodes and renders to the GPU without involving the CPU at all, that would be ideal. Barring that, maybe it's not useful to put it back in GL if I'm already using software scaling via SwScale and software color conversion.

If we wanted to go the software drawing route, I don't know what U++ provides for color conversion, I believe everything is RGB and YUV is not supported.

Subject: Re: VideoPlayerCtrl using OpenGL + ffmpeg + portaudio
Posted by jjacksonRIAB on Thu, 15 Jul 2021 14:42:40 GMT
View Forum Message <> Reply to Message

 https://stackoverflow.com/questions/57242800/how-to-convert-an-ffmpeg-texture-to-open-gl-texture-without-copying-to-cpu- memor

This gives a bit of a hint on how to stay within the GPU's memory space. Perhaps there should be some kind of reference renderer that stays CPU, using ImageBuffer and SwScaler for scaling/color conversion and then additional renderers that support the full GPU routes through GL/DirectX/Vulkan, supporting YUV->RGB through shader programs.

Subject: Re: VideoPlayerCtrl using OpenGL + ffmpeg + portaudio
Posted by Xemuth on Sat, 17 Jul 2021 21:26:33 GMT

View Forum Message <> Reply to Message

Hello JJacksonRIAB,

I just tried to compile VideoCtrlTest package on my Windows 10 and it tel me:

C:\Upp\jacksonRIAB\AudioPackage/AudioPackage.h (4): fatal error: 'portaudio.h' file not found

Since portaudio is an external header it can't compil on my computer until I install dependencies.

Subject: Re: VideoPlayerCtrl using OpenGL + ffmpeg + portaudio
Posted by jjacksonRIAB on Sat, 17 Jul 2021 22:00:42 GMT

View Forum Message <> Reply to Message

Yes, it needs portaudio installed to run. It doesn't need alsa though so I removed that unnecessary header.