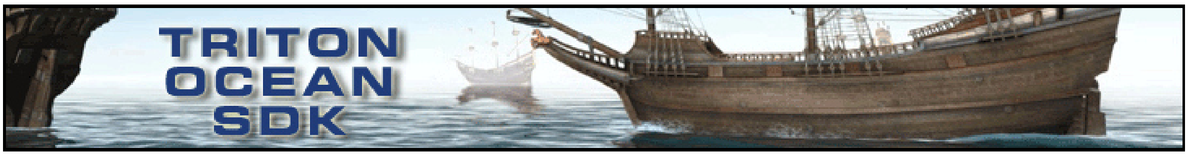

Thread: Framebuffer texture: glTexImage2D() vs. glTexStorage2D()

Thread Tools Display

06-10-2016, 04:51 AM #1

john_connor
Junior Member
Regular Contributor
Join Date: May 2016
Posts: 170

Framebuffer texture: glTexImage2D() vs. glTexStorage2D()

hi,

i've recently read that it is much better for performance to use immutable texture storages, allocated by glTexStorage2D(), rather than glTexImage2D()
for the framebuffer i need a texture that can be resized if the user resizes the window, but apparently that doesnt happen all the time

is it a good idea to use glTexStorage2D() for my framebuffer textures ?

----

another question:

i've problems using the depth texture, here's how i've se it up

Code :
```
glBindTexture(GL_TEXTURE_2D, m_framebuffer.depth.ID());
glTexImage2D(GL_TEXTURE_2D, 0, GL_DEPTH_COMPONENT24, m_framebuffer.
glTexParameteri(GL_TEXTURE_2D, GL_TEXTURE_MAG_FILTER, GL_NEAREST);
glTexParameteri(GL_TEXTURE_2D, GL_TEXTURE_MIN_FILTER, GL_NEAREST);
glTexParameteri(GL_TEXTURE_2D, GL_TEXTURE_WRAP_S, GL_CLAMP_TO_EDGE)
glTexParameteri(GL_TEXTURE_2D, GL_TEXTURE_WRAP_T, GL_CLAMP_TO_EDGE)

// ...

glFramebufferTexture(GL_FRAMEBUFFER, GL_DEPTH_ATTACHMENT, m_framebu
```

here's my fragment shader:
// used for postprocessing, both textures are the framebuffer attachments

Code :
```
#version 450


uniform vec2 DisplaySize = vec2(1200, 900);

uniform sampler2D scenecolortexture;
uniform sampler2D scenedepthtexture;
```

```
layout (location = 0) out vec4 FragmentColor0;


void main(void)
{
        vec2 texcoords = gl_FragCoord.xy / DisplaySize;

        // inversion
        //FragmentColor0 = vec4(vec3(1.0 - texture(scenecolortextur

        // grayscale
        //vec4 color = texture(scenecolortexture, texcoords);
```

the screen appears completely gray, meaning that i'm not getting 0 as depth, but something between 0 and 1
how is it done right ?

Reply With Quote

06-10-2016, 07:01 AM #2

GClements

Senior Member
OpenGL Guru

Join Date: Jun 2013
Posts: 2,025

> Originally Posted by **john_connor**
>
> *i've recently read that it is much better for performance to use immutable texture storages, allocated by glTexStorage2D(), rather than glTexImage2D()*
> *for the framebuffer i need a texture that can be resized if the user resizes the window, but apparently that doesnt happen all the time*
>
> *is it a good idea to use glTexStorage2D() for my framebuffer textures ?*

Immutable storage reduces creation and validation overhead, avoids various pitfalls (e.g. neglecting to create some levels, layers or faces, or creating them with incompatible dimensions or formats), and allows the creation of texture views. But I'm not sure that access is any faster. The wiki page doesn't mention performance as a reason for using immutable storage.

For resizing, using immutable storage means that you have to delete and re-create the texture rather than just its storage.

> Originally Posted by **john_connor**
>
> *the screen appears completely gray, meaning that i'm not getting 0 as depth, but something between 0 and 1*
>
> *how is it done right ?*

There's nothing obviously wrong with the posted code. But you're not showing anything related to the rendering. If the rendered scene fills the viewport (i.e. there's no "background"; every pixel is drawn at least once), it's entirely possible that the resulting depth values are all in a fairly narrow range. Try increasing the near distance and reducing the far distance to see if you get more variation in depth values.

Reply With Quote

06-10-2016, 07:17 AM #3

Alfonse Reinheart

Senior Member
OpenGL Lord

Join Date: May 2009
Posts: 5,872

> Originally Posted by **GClements**
>
> *For resizing, using immutable storage means that you have to delete and re-create the texture rather than just its storage.*

It should be noted that using mutable storage doesn't mean that resizing a texture is a trivial operation for the driver. The only difference is that you specifically are not forced to destroy one texture object and create a new one. The driver still has to do memory reallocation work.

It's generally much better to allocate a texture that is as big as the window will reasonably be, then only use a subregion of it if the window is smaller. Perhaps present the user with a setting for the max window size.

The Unofficial OpenGL SDK
OpenGL Loader Generator, for making OpenGL loaders
Learning Modern 3D Graphics Programming Through OpenGL

Reply With Quote

06-10-2016, 08:33 AM #4

mhagain

Senior Member
OpenGL Pro

Join Date:	Jan 2007
Posts:	1,503

> Originally Posted by **Alfonse Reinheart**
>
> *It should be noted that using mutable storage doesn't mean that resizing a texture is a trivial operation for the driver. The only difference is that you specifically are not forced to destroy one texture object and create a new one. The driver still has to do memory reallocation work.*

The probability is that destroying and recreating the texture is what the driver actually does behind-the-scenes if one uses mutable storage. The OP should realise that the OpenGL spec says nothing about how drivers should implement mutable storage, nor does it make any promises about performance.

Reply With Quote

Quick Navigation **OpenGL coding: beginners** **Top**

« Previous Thread | Next Thread »

Posting Permissions

You may not post new threads	**BB code** is On
You may not post replies	**Smilies** are On
You may not post attachments	**[IMG]** code is On
You may not edit your posts	**[VIDEO]** code is Off
	HTML code is Off
	Forum Rules

-- OpenGL Style

Contact Us OpenGL.org Archive Top

All times are GMT -7. The time now is 05:44 AM.

Powered by vBulletin® Version 4.2.3