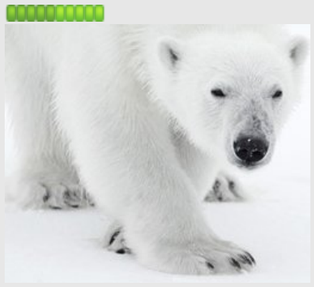
05-24-2011

Drunkie

Ursus maritimus

WIREMOD SUPER MOD



Join Date	Feb 2009
Location	Canada
Posts	6,625
Blog Entries	1

Graphics Library Documentation

Introduction:

This documentation outlines the complete list of functions for the C graphics library in the GPU. The library is used for rendering objects in 2d and 3d environments. Source code can be found in your garrysmod data folder
under: *GPUChip/lib/drivers/drv_gl.txt*

Example programs can be found in the SVN under the examples folder:

* 3d_cube.txt
* 3d_icosahedron.txt
* 3d_letter_a.txt
* 3d_tunnel.txt
* line_graph.txt
* table.txt
* texture.txt
* trig.txt

If you are looking for a deeper explanation of everything, please check here: HL-ZASM Documentation

How to Use:

You can use these functions simply by including the source code in your program like this:

Code:

```
main();

#include <drivers\drv_gl.txt>

void main()
{
    // Program logic here

    glExit();
}
```

You must always call glExit() or dexit.

Function Listing:

Clear functions:

Returns	Function	Description
void	glClear(float r, float g, float b)	Clears the screen with a color.
void	glClear4(float r, float g, float b, float a)	Clears the screen with a color and transparency value.
void	glClearTexture( )	Clears the screen using currently binded texture. Use glBindTexture(str) to set a texture.
void	glHWClear(float n)	Sets hardware clearing on the GPU. When set to 0 it will make the screen never clear and maintain state between frames.

Color functions:

Returns	Function	Description
void	glColor(float r, float g, float b)	Sets the current draw color.
void	glColor4(float r, float g, float b, float a)	Sets the current draw color with transparency value.
void	glBrightness(float r, float g, float b, float a)	Sets the brightness. A is intensity (0 to 1).
void	glContrast(float r, float g, float b, float a)	Changes the contrast. A is intensity (0 to 1).
void	glShade(float n)	Shades the current color.
void	glShadeNorm(float n)	Shades the current color by a normal value.

Texture functions:

Returns	Function	Description
void	glBindTexture(char* str)	Sets an external texture to use. glBindTexture('pathname') to set and glBindTexture(0) to discard texture.
void	glTexture(float id)	Sets the texture sampling index from the sprite buffer. If you had 4 sprites of 256x256, index 0 would sample from [0,0], index 1 would sample from [256,0], index 2 would sample from [0,256], index 3 would sample from [256,256]. The size of the sample is set by glTextureSize(float n).
void	glTextureSize(float n)	Sets the size of the texture to sample from the sprite buffer.
void	glTextureRotation(float n)	Sets the rotation of a texture.
void	glTextureScale(float n)	Sets the scale of a texture.
void	glTextureCenterUV(float u, float v)	Sets the center UV coordinate of a texture.
void	glTextureOffsetUV(float u, float v)	Sets the offset to the UV coordinate of a texture.
void	glTextureDataPtr(float n)	Sets the pointer to where texture data is stored.
void	glTextureDataSize(float n)	Sets the count of the texture data to read, as pointed to by glTextureDataPtr(float n)

Frame functions:

Returns	Function	Description

void	glSetRenderTarget(float n)	Sets the buffer to start drawing into. The following enums can be used with this function:
		Code:
		`0 - GL_BUFFER_FRONT`
		`1 - GL_BUFFER_BACK`
		`1 - GL_BUFFER_SPRITE`
		`2 - GL_BUFFER_VERTEX`

void	glSleep(float ms)	Forces the frame to sleep for a specified time in milliseconds. Only works when rendering in the front buffer. You can use this to reduce FPS lag.
void	glExit( )	Marks the end of a program. Must always be used.

Pipeline functions:

Returns	Function	Description

void	glCoordPipe(float c)	Sets the coordinate system. The following enums can be used with this function:
		Code:
		`0 - GL_CPIPE_DIRECT`
		`1 - GL_CPIPE_RESOLUTION`
		`2 - GL_CPIPE_0_1`
		`3 - GL_CPIPE_N1_1`
		`4 - GL_CPIPE_N256_256`

void	glVertexPipe(float v)	Sets the vertex pipe. The following enums can be used with this function:
		Code:
		`0 - GL_VPIPE_XY`
		`1 - GL_VPIPE_YZ`
		`2 - GL_VPIPE_XZ`
		`3 - GL_VPIPE_XYZPROJ`
		`4 - GL_VPIPE_XYTRANSFORM`
		`5 - GL_VPIPE_XYZTRANSFORM`

Hardware functions:

Returns	Function	Description
void	glVertexMode(float n)	Sets vertex mode on the GPU. This enables drawing to the world. You must enable this mode to use textures.
void	glReset(float n)	Resets the GPU
void	glHalt(float n)	Halts the execution of the GPU.
void	glRAMReset(float n)	Resets the RAM on the GPU
void	glHScale(float n)	Sets horizontal scale
void	glVScale(float n)	Sets vertical scale
void	glHWScale(float n)	Sets hardware scale. Can be used to create many copies of the image on the screen.
void	glHWRotate(float n)	Sets hardware rotate (rotates the entire screen).

Offset functions:

Returns	Function	Description
void	glOffset(float x, float y)	Sets the drawing offset on the screen. Can be used to offset objects without changing draw positions.
float	glOffsetX( )	Returns the current offset on the X axis.

float	glOffsetY( )	Returns the current offset on the Y axis.
void	glCenter(float x, float y)	Sets the center point of the screen.

Async functions:

Returns	Function	Description
void	glAsyncReset(float n)	Resets async rendering.
void	glAsyncClk(float n)	Enable or disable async rendering.
void	glAsyncFreq(float n)	Sets the async frequency (speed of rendering).
void	glEntryPoint(float idx, float ptr)	Sets the entry point for the program.
void	glBegin( )	Begin rendering in async mode.
void	glEnd( )	End rendering in async mode.
void	glSwap( )	Copies the sprite buffer to the front buffer.
void	glSync( )	Waits until the next frame to draw in async mode.

Cursor functions:

Returns	Function	Description
void	glCursor(float n)	Enables or disables the cursor.
float	glCursorX( )	Returns the cursor's position on the X axis (0 to 1).
foat	glCursorY( )	Returns the cursor's position on the Y axis (0 to 1).

Circle functions:

Returns	Function	Description
void	glCircleQuality(float n)	Sets the quality of circles (number of sides).
void	glCircleStart(float n)	Sets the start point of a circle in radians
void	glCircleEnd(float n)	Sets the end point of a circle in radians.

Screen scaling functions:

Returns	Function	Description
void	glScreenScale(float n)	Sets the scale of the entire screen.
void	glScreenScaleX(float n)	Sets the X scale of the entire screen.
void	glScreenScaleY(float n)	Sets the Y scale of the entire screen.

2D Graphics functions:

Returns	Function	Description
void	glCircle(float x, float y, float radius)	Renders a circle with specified radius.
void	glRect(float x, float y, float dx, float dy)	Renders a solid rectangle from top-left corner to bottom right-corner.
void	glRectWH(float x, float y, float w, float h)	Renders a solid rectangle from top-left corner to specified width and height.
void	glORect(float x, float y, float dx, float dy)	Renders an outlined rectangle from top-left corner to bottom right-corner.
void	glORectWH(float x, float y, float w, float h)	Renders a outlined rectangle from top-left corner to specified width and height.
void	glPixel(float x, float y)	Renders a single pixel (1x1 rectangle).
void	glLine(float x, float y, float dx, float dy)	Renders a line from one point to another.
void	glLineWidth(float w)	Sets the width of a line. Affects 2D lines and 3D wireframe.
void	glPoly2D(float* buffer, float count)	Renders a 2D polygon. First parameter is the pointer to where the vertex data is stored, the second is the number of vertices.

Text functions:

Returns	Function	Description

void	glFont(float id)	Sets the type of font to use. The following enums can be used with this function:
		Code:
		`0 - GL_FONT_LUCIDA_CONSOLE`
		`1 - GL_FONT_COURIER_NEW`
		`2 - GL_FONT_TREBUCHET`
		`3 - GL_FONT_ARIAL`
		`4 - GL_FONT_TIMES_NEW_ROMAN`
		`5 - GL_FONT_COOLVETICA`
		`6 - GL_FONT_AKBAR`
		`7 - GL_FONT_CSD`

void	glFontAlign(float n)	Sets the horizontal font align. The following enums can be used with this function:
		Code:
		`0 - GL_ALIGN_LEFT`
		`1 - GL_ALIGN_CENTER`
		`2 - GL_ALIGN_RIGHT`

void	glFontVAlign(float n)	Sets the vertical font align. The following enums can be used with this function:
		Code:
		`0 - GL_VALIGN_TOP`
		`1 - GL_VALIGN_MIDDLE`
		`2 - GL_VALIGN_BOTTOM`

void	glFontSize(float n)	Sets the size of the font.

float	glTextWidth(char* str)	Returns the width of a string.
float	glTextHeight(char* str)	Returns the height of a string.
void	glWriteString(float x, float y, char* str)	Writes a string on the screen.
void	glWriteFloat(float x, float y, float n)	Writes a float on the screen.
void	glWriteInt(float x, float y, float n)	Writes a integer on the screen.
void	glWriteFormat(float x, float y, char* str)	Writes a formatted string on the screen.
float	glParamList( )	Returns parameter list address (for formatted strings)

3D Graphics functions:

Returns	Function	Description
void	glPoly3D(float* buffer, float count)	Renders a 3D polygon. First parameter is the pointer to where the vertex data is stored, second is the number of faces on the polygon. You can set the fill mode of the polygon by using glFillMode(float n)
void	glFlush( )	Flush all of the vertices to the screen to be rendered.

void	glEnable(float n)	Enables special features. The following enums can be used with this function:
		Code:
		`0 - GL_VERTEX_ZSORT`
		`1 - GL_VERTEX_LIGHTING`
		`2 - GL_VERTEX_BUFFER`
		`3 - GL_VERTEX_CULLING`
		`4 - GL_VERTEX_DCULLING`
		`5 - GL_VERTEX_TEXTURING`

void	glDisable(float n)	Disables special features. The following enums can be used with this function:
		Code:
		`0 - GL_VERTEX_ZSORT`
		`1 - GL_VERTEX_LIGHTING`
		`2 - GL_VERTEX_BUFFER`
		`3 - GL_VERTEX_CULLING`
		`4 - GL_VERTEX_DCULLING`
		`5 - GL_VERTEX_TEXTURING`

void	glLightPos(float x, float y, float z)	Sets the position of the light source in 3D space.
void	glLightColor(float r, float g, float b, float a)	Sets the color of the light source.

void	glFillMode(float n)	Sets the fillmode for 3D polygons. The following enums can be used with this function:
		Code:
		`0 - GL_FILL_SOLID`
		`1 - GL_FILL_WIREFRAME`
		`2 - GL_FILL_TEXTURE`

void	glLookAt(float x, float y, float z, float tx, float ty, float tz, float ux, float uy, float uz)	Sets the camera's position, target and up vectors for 3D projections. Position is where the camera is located in 3D space, target is the vector the camera should aim at, and up defines the up direction of the camera which should be (0,1,0) most of the time.
void	glPerspective(float fov, float asp, float znear, float zfar)	Sets the perspective for 3D projections. Field of view defines how much of an angle you see, aspect ratio is the ratio of width to height (usually 1), zNear and zFar defines the range that objects should be visible in and not be clipped from the Z plane.
void	glRotate(float x, float y, float z, float w)	Sets the rotation in 3D projections for axis X, Y and Z. W is the angle of rotation. All values are in radians.
void	glTranslate(float x, float y, float z)	Sets the translation for 3D projections. This will move the position of an object in 3D space.
void	glScale(float x, float y, float z)	Sets the scale for 3D projections. The scale will change the size of 3D objects on axis X, Y and Z.
void	glZOffset(float n)	Adds an offset to the Z-axis in 3D projections.
void	glCullDistance(float n)	Sets the culling distance in 3D projections. You must enable GL_VERTEX_DCULLING to use this.

void	glCullMode(float n)	Sets the culling mode in 3D projections. The following enums can be used with this function:
		Code:
		`0 - GL_CULL_FRONT`
		`1 - GL_CULL_BACK`

void	glLightMode(float n)	Sets the lighting mode in 3D projections. The following enums can be used with this function:
		Code:
		`1 - GL_LIGHT_FRONT`
		`-1 - GL_LIGHT_BACK`

void	glVertexArray(float n)	Sets the memory address of where to read vertex data from. When this is used, all polygon draw functions will take an index array instead. The index array contains pointers to what vertices to connect in the vertex array.

Other functions:

Returns	Function	Description
float	glIndex( )	Returns the ID of who the GPU is running on.

*Last edited by Drunkie; 02-04-2012 at 04:26 PM. **Reason:** updated documenation and added newer, cleaner doc format*