iMX6 Specific Best Practices

Description

CGI Studio supports 2D and 3D content creation, composition, and rendering. For iMX6 the Candera graphics engine renders both 2D and 3D content via OpenGL ES API.

This chapter includes design and programming guidelines. It is strongly recommended to follow those hints to utilize *iMX6* in a performance-optimized manner.

Candera Graphics Engine supports both OpenGL ES 2.0 and OpenGL ES 3.0. Selected performance features of OpenGL ES 3.0 are available in OpenGL ES 2.0 as extensions.

Those OpenGL ES 2.0 extensions that are supported by iMX6 and Candera are listed below:

- Binary shader programs: Reduces start up times significantly.
- Direct texture: Vivante extension to access external image and video data.
- Texture compression: Latest ETC2 compression minimizes required memory bandwidth and storage for image data.
- Anisotropic filtering: Increased texture quality for steep angles of view.
- Broad range of buffer formats: Depth 24/32, Stencil 1/4/8, etc.
- 24-bit vertex element index: Support for higher polygon count in 3D models.
- Blending equations minimum and maximum: Enables color clamping.
- Synchronization: Fences support CPU / GPU synchronization.
- And many more.

> Many of those extensions have been introduced in OpenGL ES 3.0 API.

Multi Frequency Rendering

Multi frequency rendering requires either multiple buffered hardware layers (display planes) or software layers. While one layer is rendered and swapped with a higher frequency (e.g. swapping each frame results in 60 FPS), another layer is drawn partially and not swapped unless rendering is complete (e.g. swapping each second frame results in 30 FPS).

When software-layers are used, different update rates of the software-layers can be achieved through multi frequency rendering. It should be considered that the software layers still need to be drawn and blended in every frame and will consume memory bandwidth even when rendering to

the software-layer is done at a lower rate.

Special Alpha Blending Techniques

MSAA Blending

The iMX6 platform supports multi-sampling anti-aliasing (MSAA) to avoid jagged edges. MSAA renders multiple samples per pixel and combines them to a final color. This circumstance can be used to render transparent objects in a very performant manner, too. Instead of alpha blending via blend-mode, MSAA can be used to render only a selected percentage of samples per pixel. With that approach blend operations and read/write operations, which are typically required for blending, can be omitted.

Example:
MSAA is defined to accommodate 4 samples per pixel. The alpha value can define the coverage of samples to be written into frame buffer. Coverage of 50% means that half the samples are written, resulting in 2 of 4 samples being rendered. When the samples are finally combined to one pixel, the rendered object contributes only with 2 of 4 samples, which results in 50 % transparency.

Layer Blending

When blending multiple layers a correct alpha value must be written to the layer. By default objects overwrite the current alpha value of a pixel in the layer. In case of layer blending the alpha value of a pixel must be correctly combined with the existing alpha value.

Default alpha blend setting (optimized for performance):

- Source Blend Factor: One
- Destination Blend Factor: Zero

Alpha blend setting for blending multiple layers:

- Source Blend Factor: Inverse Destination Alpha
- Destination Blend Factor: One

Shader

Shader Precision

It is *strongly* recommended to use precision "*high*" in iMX6 shaders.

As medium and low precisions are converted to 32 bit internally, work load for conversion is

required, while only saving little amount of memory compared to textures.

Utilization

One strength of iMX6 is the fast shader computation unit. Utilize it best, by computing color values instead of texture fetches. Take care of pixel throughput and bandwidth utilization. Reduction of pixel coverage in frame-buffer is of utter importance.

Example:
Do not use full-screen backgrounds or big textures. In 2D reduce coverage by tessellated, smaller shaped images or in 3D even better by right-shaped geometry, with least pixel coverage.

Binary Shaders

Candera supports both binary shaders (pre-compiled) and binary shader programs (pre-compiled and linked) on the iMX6 platform. Use them to reduce start-up time significantly.

Textures

Procedural Textures

For iMX6 it is advised to use procedural textures whenever possible in order to both, overcome limited bandwidth and to utilize fast shader units.

Example:
Instead of storing a color gradient in a texture, calculate color in shader related to the texture coordinate given.

Texture Compression

Use **ETC2** texture compression on iMX6. This measure saves bandwidth and memory consumed.

> Textures with color gradients shall not be compressed, due to noticeable decrease in visual appearance.

Consider Tile Size

The iMX6 platform internally uses a tile-based rendering approach. Make sure to align render-area with the optimal tile size of 16x8 pixels. Small intersections of adjacent tiles shall be avoided. Partial coverage of tiles results in wasted performance.

Scene Partitioning

Recommended Partitioning of a 2D Scene

Attempt to reduce dimension of rectangles by omitting areas that are invisible (e.g. same color as background) or occluded (another opaque image is rendered on top of it). This spares pixel coverage.
Furthermore, only those regions need to be redrawn that are not covered by moving needle.
Example: A classical, round tachometer with a needle shall be rendered. The tachometer basically consists of a tachometer image and a needle image. Except of the tachometer's dial most pixels within tachometer's image have same color as background. Tip: Scissor the tachometer into multiple smaller rectangles to avoid rendering of a huge rectangular image, which renders big portion of same color on top of background. This measure spares pixel coverage. Furthermore, only those clipped tachometer image rectangles must be redrawn that are not covered by moving needle.
Advantages:

- Less pixel coverage.
- Reduced rendering updates.

Recommended Partitioning of a 3D Scene

Use materials instead of textures, no image data required.
Material can be defined directly in shader to reduce number of shader parameters.
Geometry allows occluding exactly those areas visible; no wasted pixels are rendered into framebuffer.
To mitigate aliasing effects (jagged edges), fade out silhouette with transparency.
Advantages:

- Utilize computation strength of iMX6 (shader units), avoid bandwidth utilization (read/write of image data), and spare memory allocation (image data).
- Ease of transitions.
- Advanced visual effects are possible.

Layer Configuration

The iMX6 hardware and layer capabilities are reflected in SceneComposer configuration.

Hardware Layers via Planes:
One IPU (Image Processing Unit) blends two planes efficiently with Display Processor (DP).

Restrictions of the planes:

- Background plane must be full-screen
- Foreground (Overlay) plane can have any size and position

Split of content in two layers allows rendering in multiple frequencies. See chapter Multi Frequency Rendering for more details.

Software Layers:
Software Layers are supported via off-screen surfaces. Up to eight software layers can be blended in one single pass. In successive blend-passes even more software layers are possible. The use of software layers is fully integrated in SceneComposer.

> Off-screen surfaces do not support anti-aliasing in OpenGL ES 2.0.

Arbitrary Warping:
Full support for arbitrary warping as required for Head-up display.
CGI Studio supports arbitrary warping for Head-up displays on iMX6. Warping is internally realized via morphed warping mesh.

Revision #3
Created 2023-03-02 05:02:26 UTC by Tsuyoshi.Kato
Updated 2025-01-24 10:37:28 UTC by Elisabeth Zauner