Graphics driver and Hardware related Best-Practices

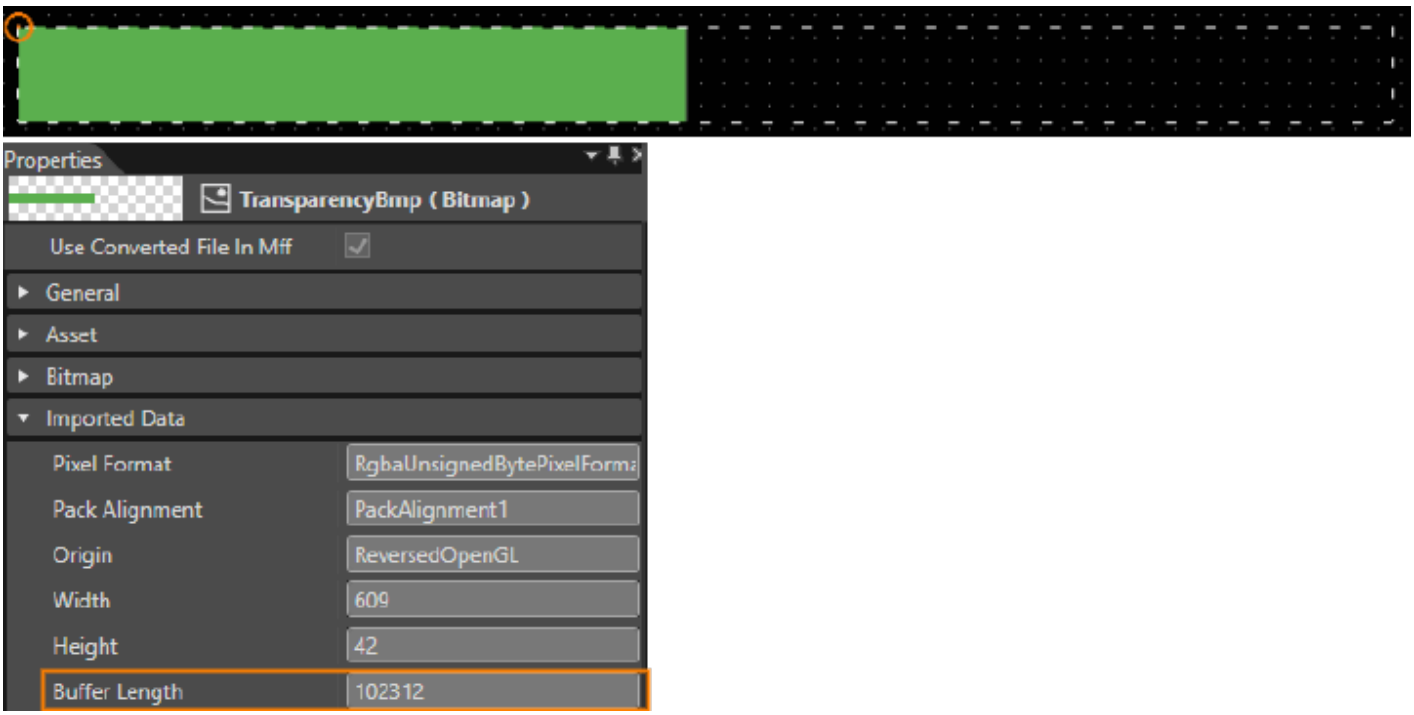
Images and Color formats

Best practice for image drawing:

- Always use as small images as possible to save memory consumption and memory bandwidth.
- Cut off transparent area in the bitmap before usage.

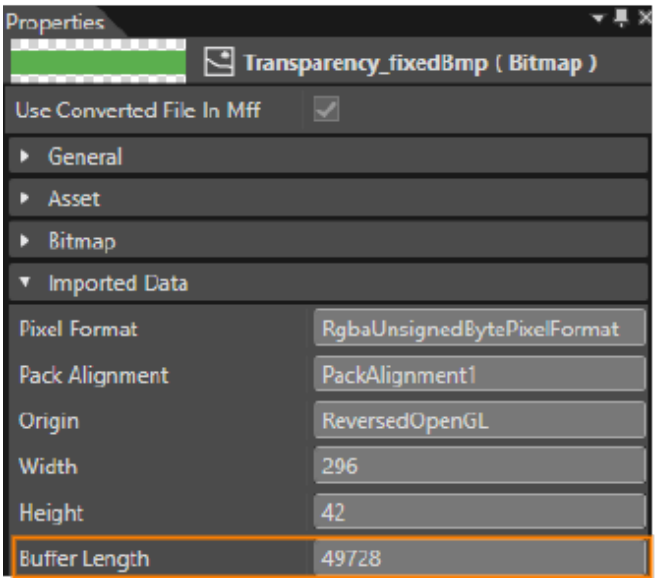
Bitmaps that contain transparent areas

<Origin>



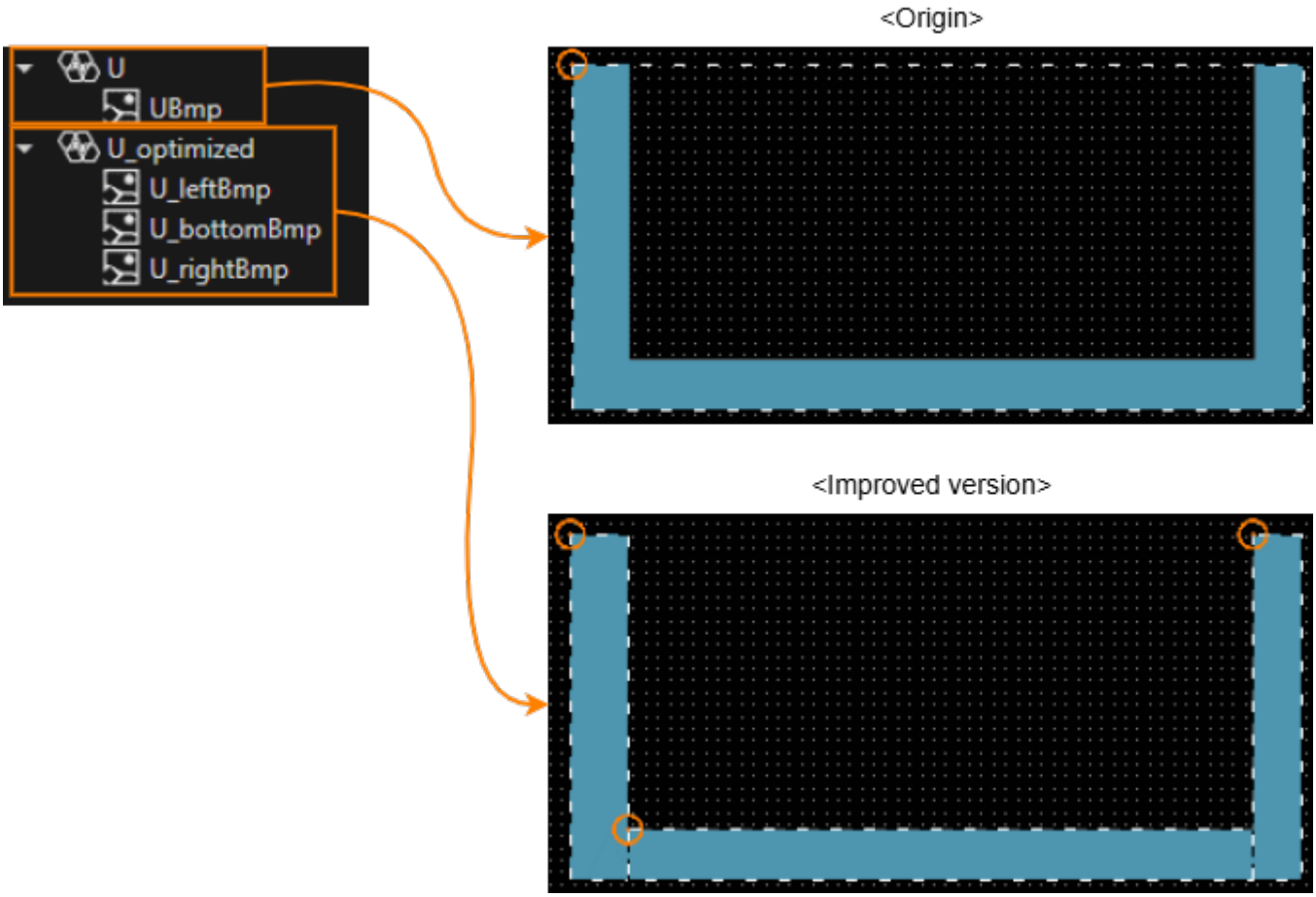
<Improved version>



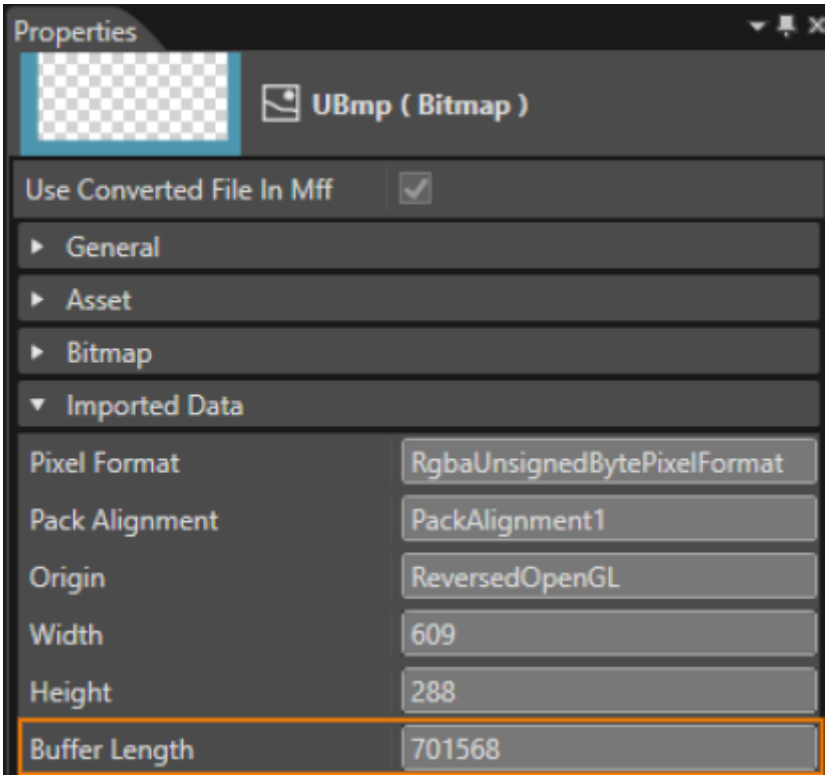


> Bitmap size difference: 102312 ⇔ 49728!

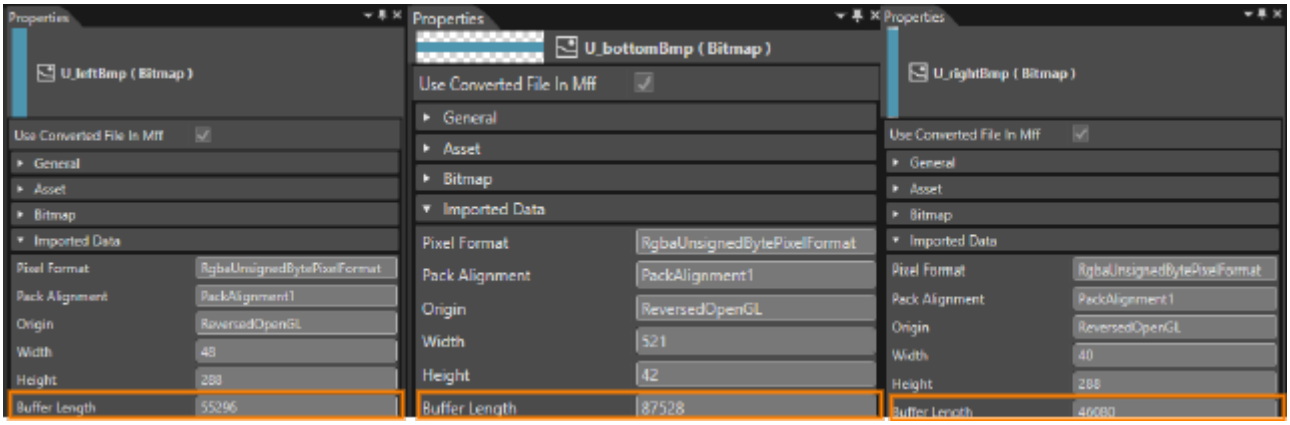
Modify bitmaps with large transparent areas ("U" shape)



<Origin>



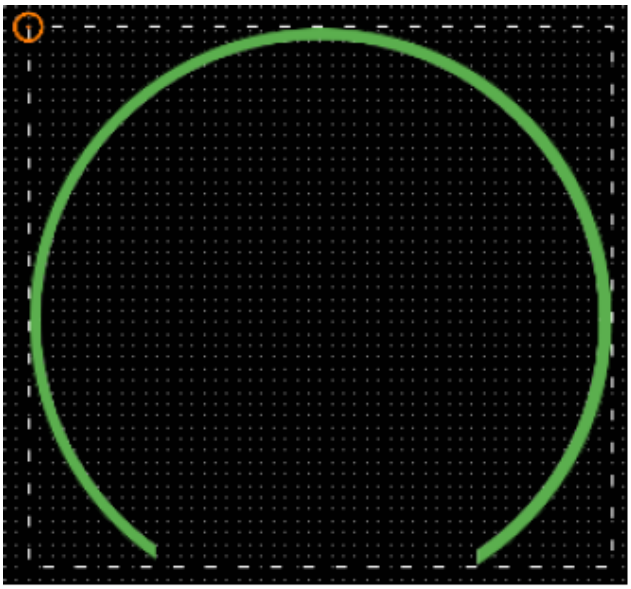
<Improved version>

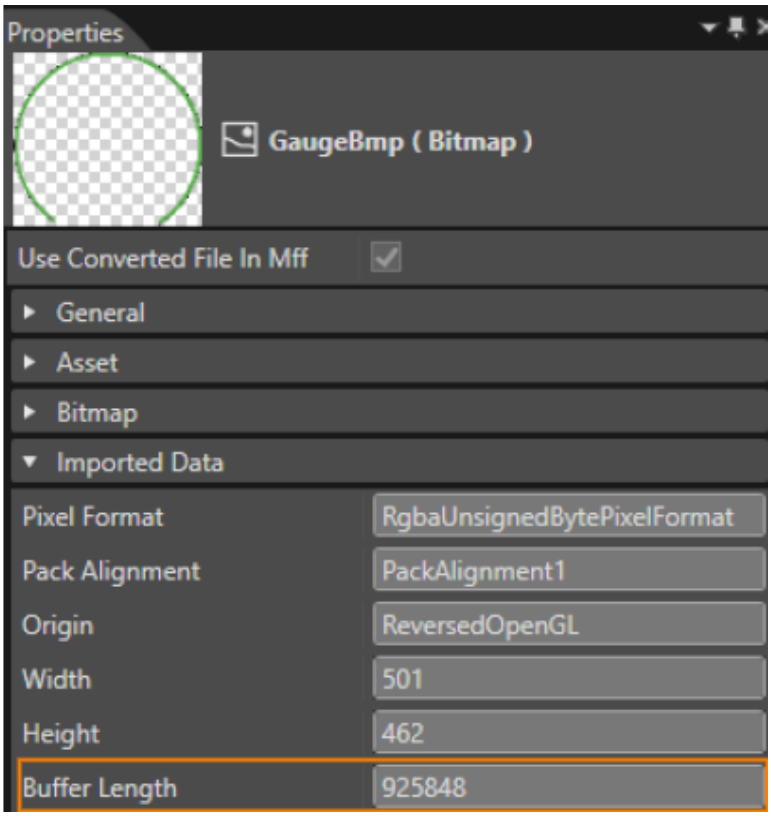


> Bitmap size difference: 701568 ⇔ 188904 (55296 + 87528 + 46080)!

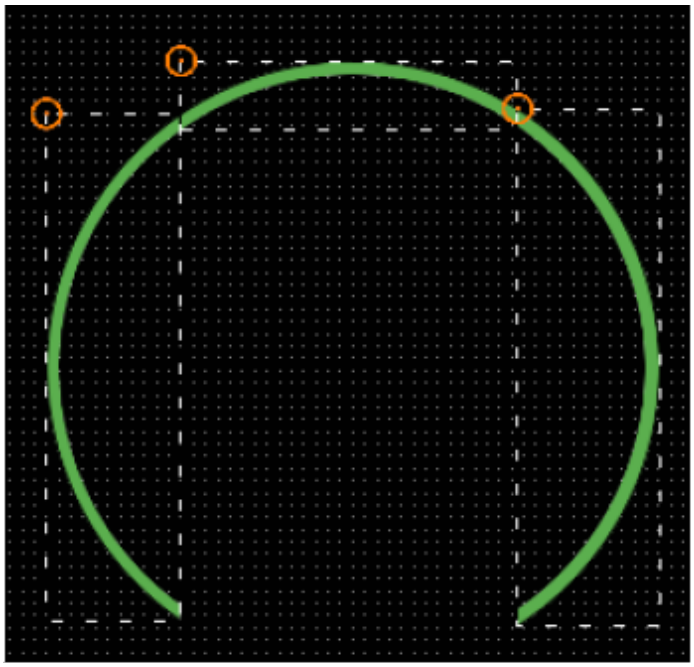
Modify bitmaps with large transparent areas (Gauge)

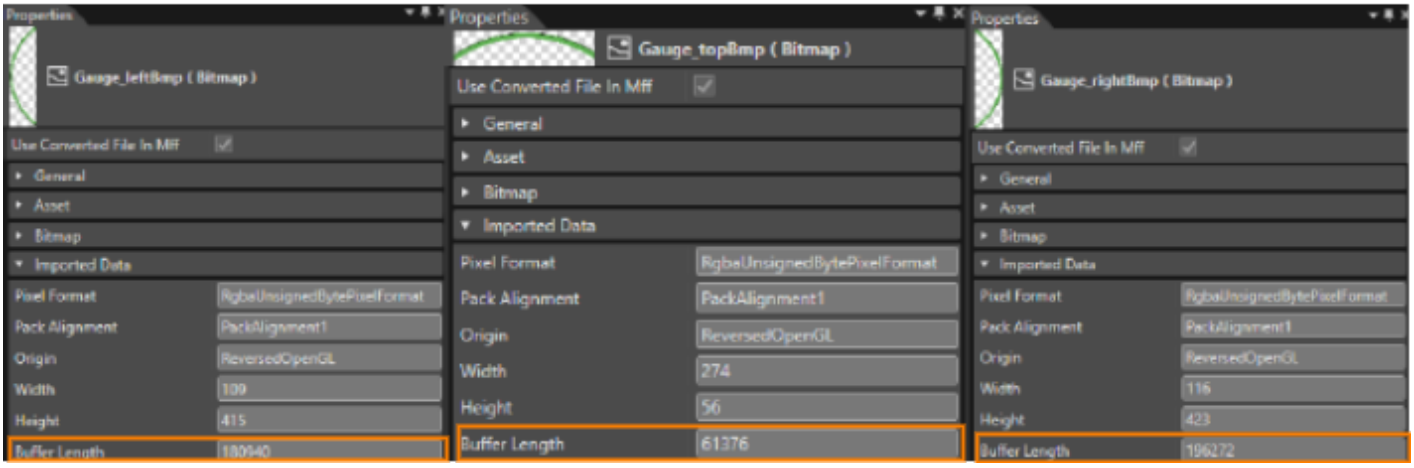
<Origin>





<Improved version>





> Size difference: 925848 ⇔ 438588 (180940 + 61376 + 196272)!

Single-colored bitmaps

For single-colored bitmaps use BitmapBrushColorBlend and use an alpha only color format e.g. A8 or A4. If further size reduction is required there are also compressed formats but those prohibit scaling and rotation on all types of LBO render targets.

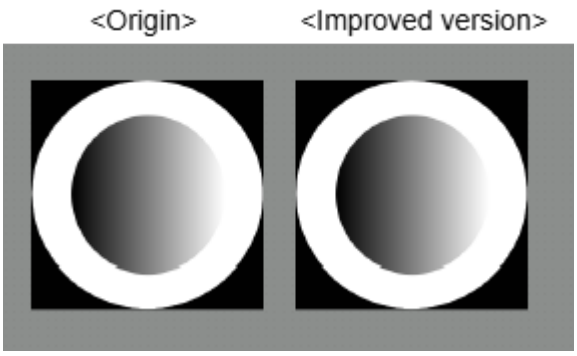
<Origin>



<Improved version>



Visual result:



> size reduction: RGBA: 222028 ⇔ 74092!

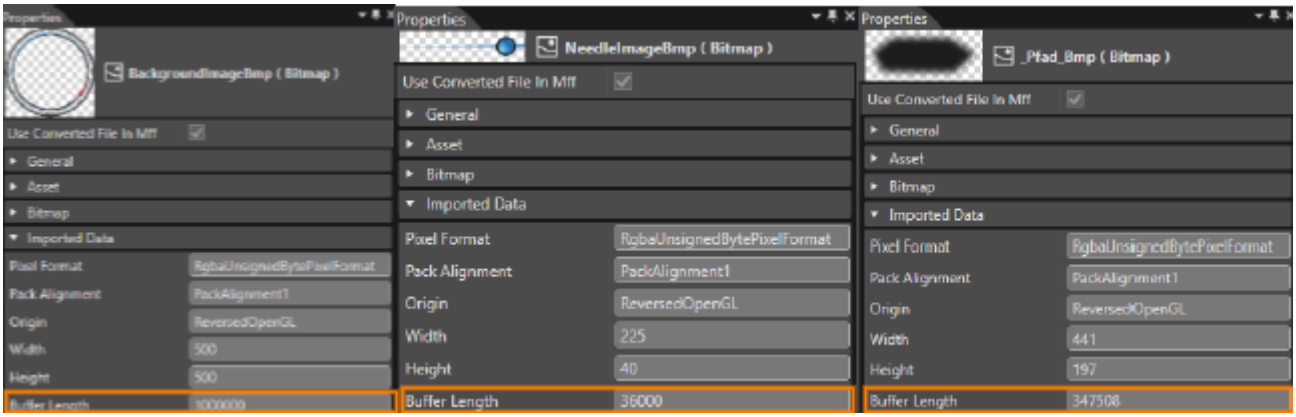
Combine Bitmaps

Bitmaps can be combined to one bitmap at design time. E.g., tube, scale, and numbers should already be combined if possible.

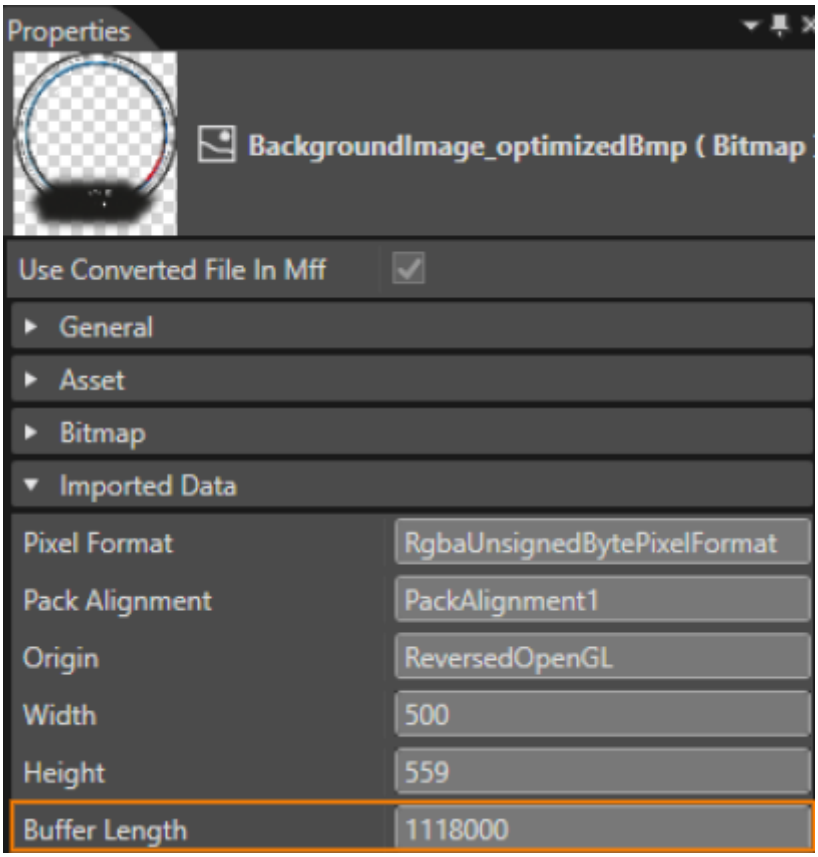


Reduce number of objects from 5 to 2. Avoid rendering of non-translatable static text. Only one moving object left and maybe the gauge itself can be put into a background image.

<Origin>



<Improved version>



> Bitmap size reduction: 1383508 (1000000 + 347508 + 36000) ⇔ 1154000 (1118000 + 36000)!

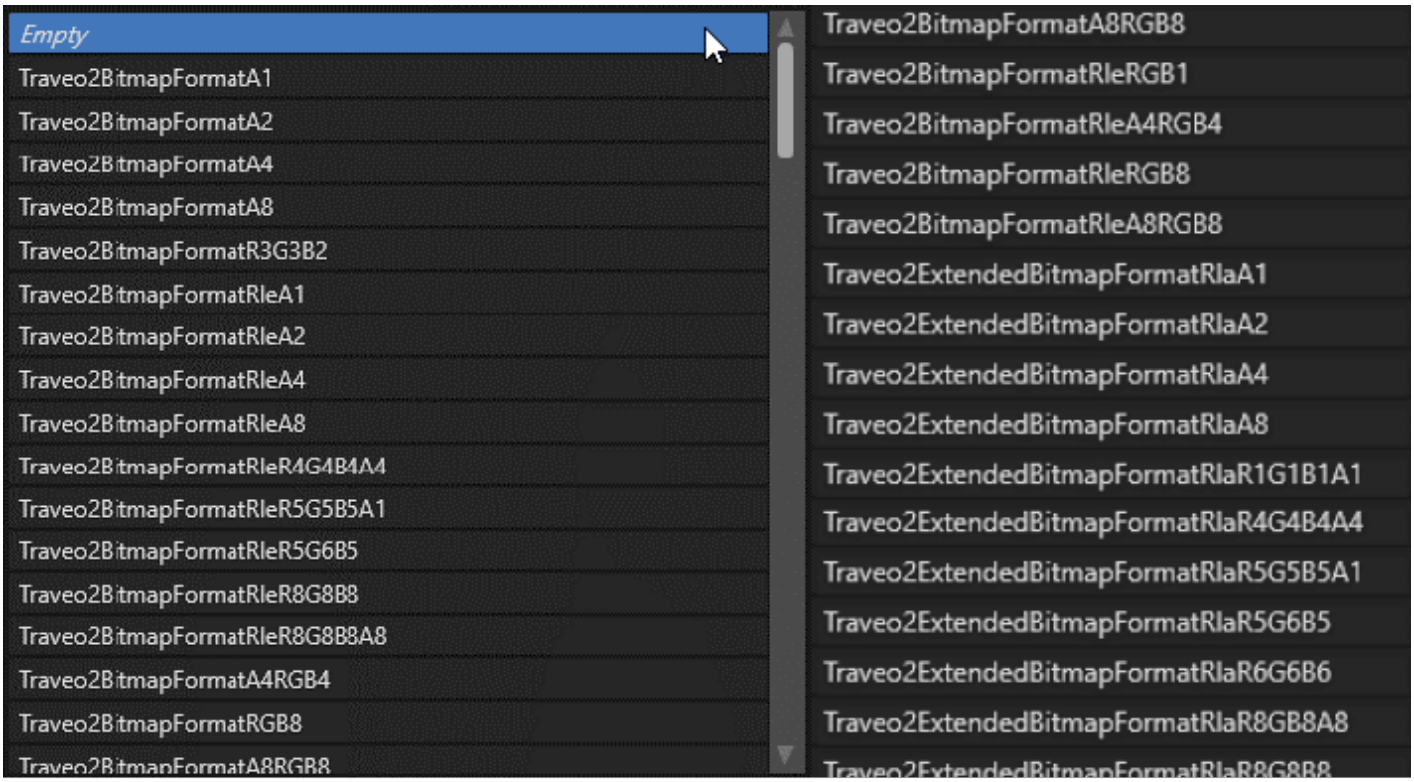
Compressed image formats

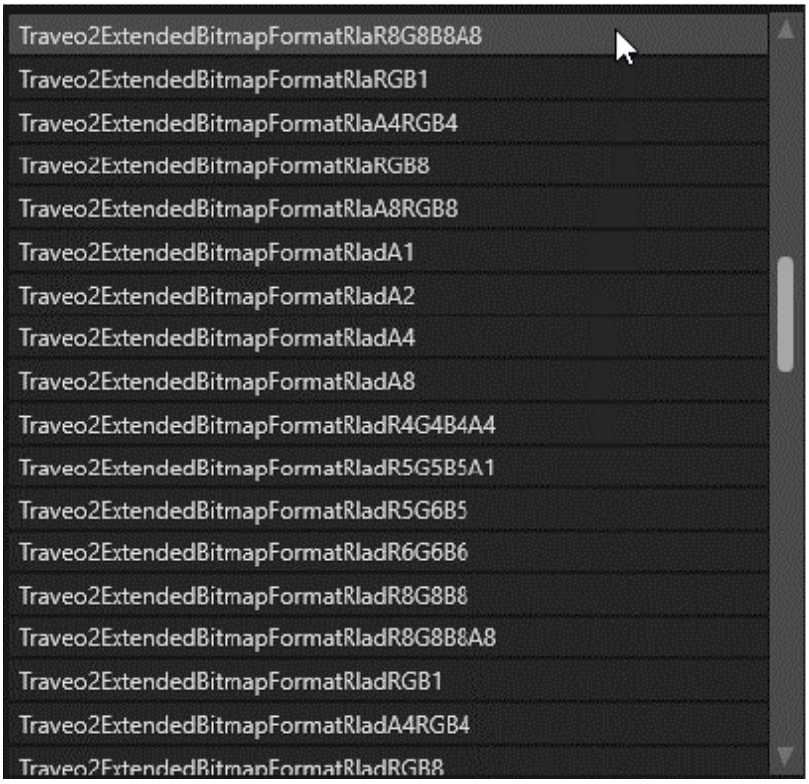
Best practice:

- Use RLE/RLA and CLUT compressions as much as possible wherever possible.
- Prepare the images that they will fit to an image-with alignment multiple of 8.
  If you have a bitmap which has transparencies at the left or right side, you can remove some pixels in x direction to get a multiple of 8. If there are no transparencies, you must add some transparent pixels at the right side of the bitmap to have a multiple of 8.

Usage of Bitmap profiles

There are many different bitmap formats available for Traveo2. The goal is always to use a format which uses least memory as possible. The drawback is that some formats have restrictions regarding transformation (rotation, scaling, ...).
Best practice is to use RLE formats to reduce memory size. AX (A1, A2, ...) formats are used to reduce memory further if just recoloring is needed (white or grayish bitmaps can be recolored in CGI Studio).
If a channel (RGBA) has less than 8 bits you will have a lossy compression.
For special use cases there are formats which can dither bitmaps (Nq, Mc, Nq_FsDithered, Mc_FsDithered. Based on the look and feel on the target, usually different formats of such type must be tested to decide what to use.

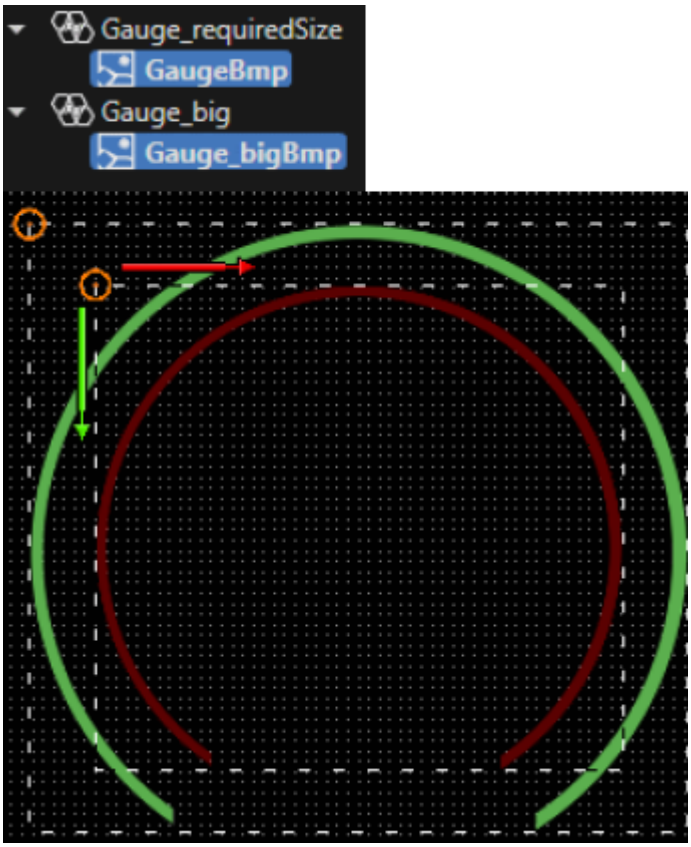




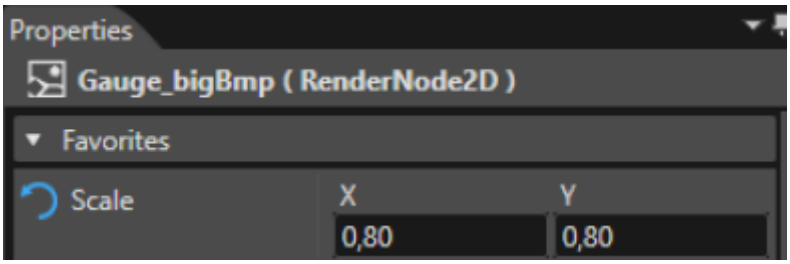
Transformations

Best practice Scaling and Rotation:

- These operations require reverse lookup and therefore need more bandwidth and compute power than a simple BLIT. If not necessary, avoid these.
- It is better to use images with already corrected dimensions instead of scaling down.

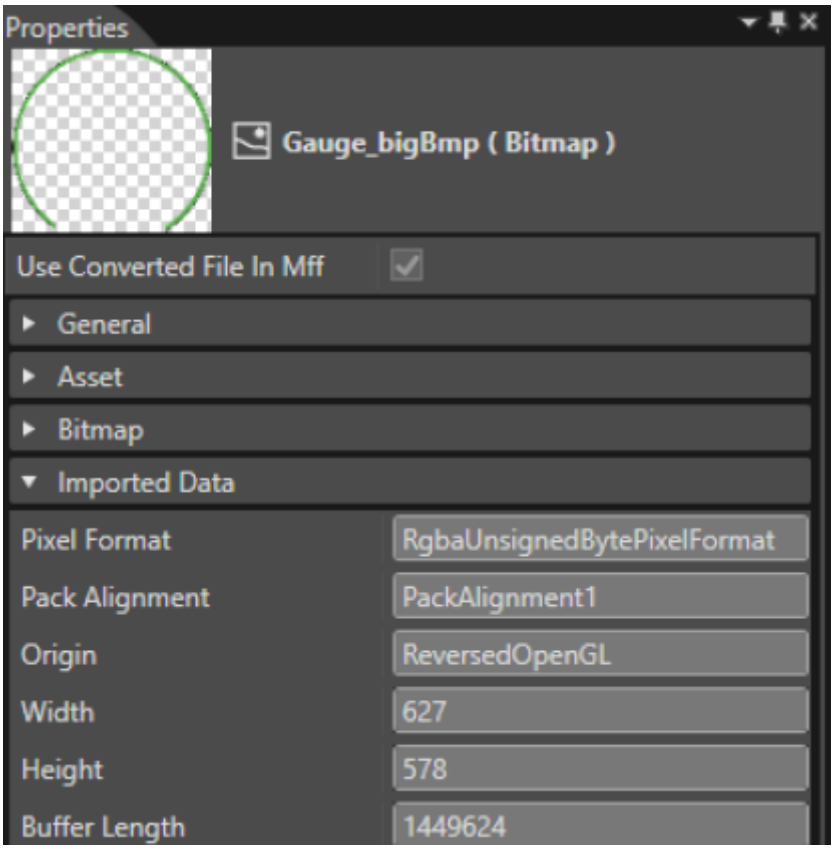


The red gauge should be the required size. The green scale is too big and must be scaled down by to 80%.

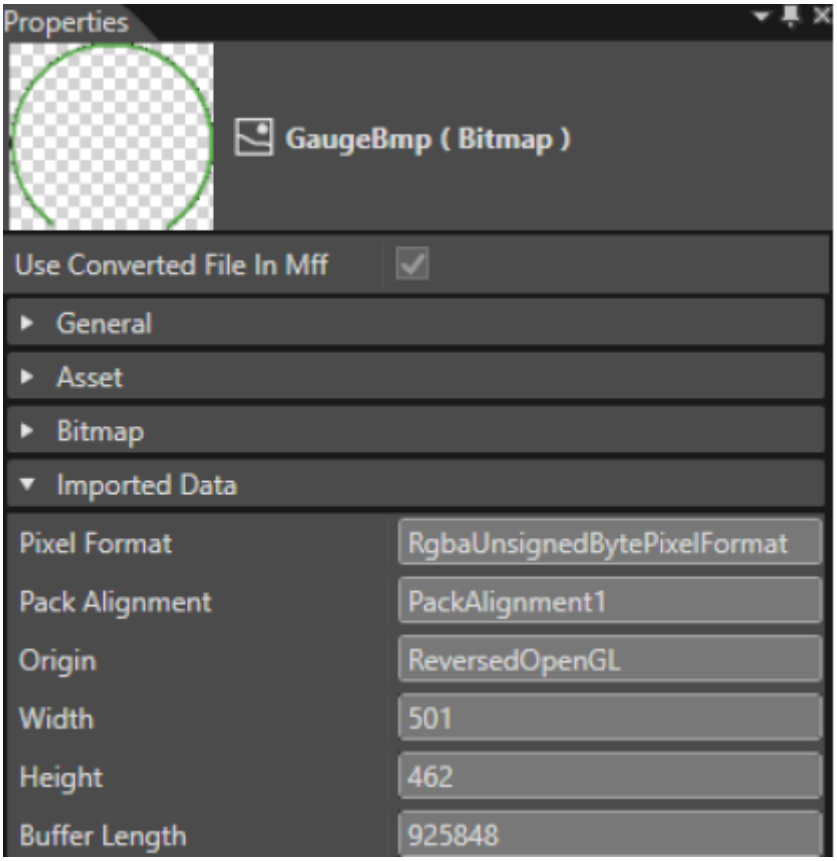


Needed size of gauge will be 925848

<Origin>



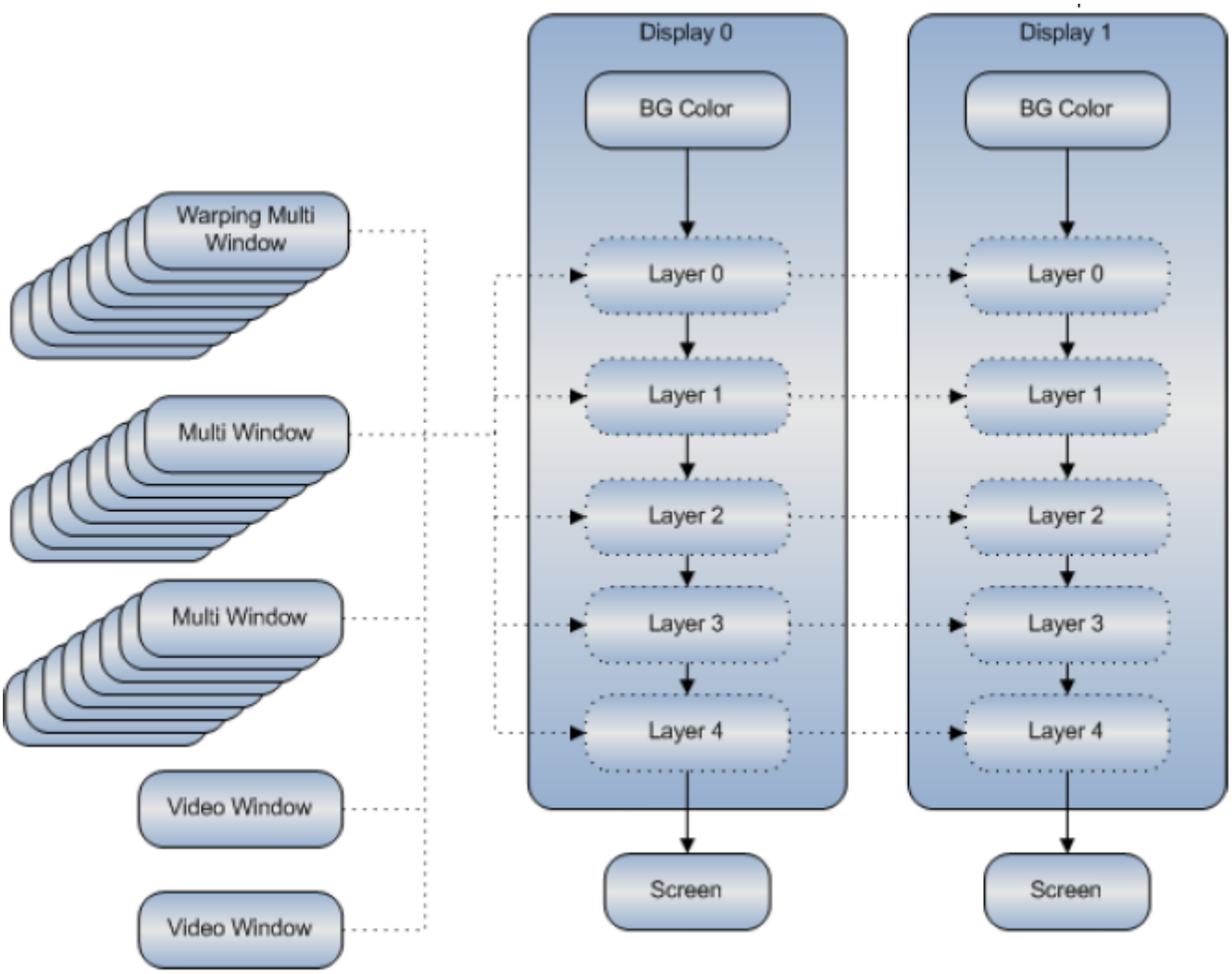
<Improved version>



> Bitmap size comparison: 1449642 ⇔ 925848!

Using Layers

In general, there is always a tradeoff between performance (incl. memory band-with) and memory consumption (e.g., VRAM). Traveo II allows, with the new introduced “Layer-modes” IBO, LBO and OTF, to steer in one of these directions.
Traveo II supports (like Cypress Amber) a huge amount of “separated” Layers, up to 5 “Main-Windows”, 3 of the windows can be split up to 8 sub-windows.
Large area screens shall always be split up to several sub screens to save VRAM consumption.



Layer types

- **IBO Mode:** regular “dual render target / buffer” approach.
  - **Best practices:**
    - Use IBO layers for dynamic text drawing with Text engine.
    - Use for content which will not change every frame.
    - Use only small areas/content with IBO (For more details, please refer to "Notes for the different modes (IBO/LBO/OTF)" on this page)
- **LBO Mode:** Line based rendering operation
  - Increased BLIT Engine performance
  - **Note the restrictions, following features are NOT supported in LBO Mode:**
    - ROP2 or ROP3 ➔ Mask operations
    - Scaling of compressed images
    - Color Lookup Table (CLUT)

    - **Note the limited functionality:**
        - Rotation: only in steps of 90°
        - Scaling: limited quality for scale factors <0.5
        - Rendering images with compression in LBO mode requires "COPSes" (For more details, please refer to Layer usage)
- **OTF Mode:** On-The-Fly mode
    - Display to a window with a small buffer (a few lines only) in memory
    - All BLIT operations use LBO mode

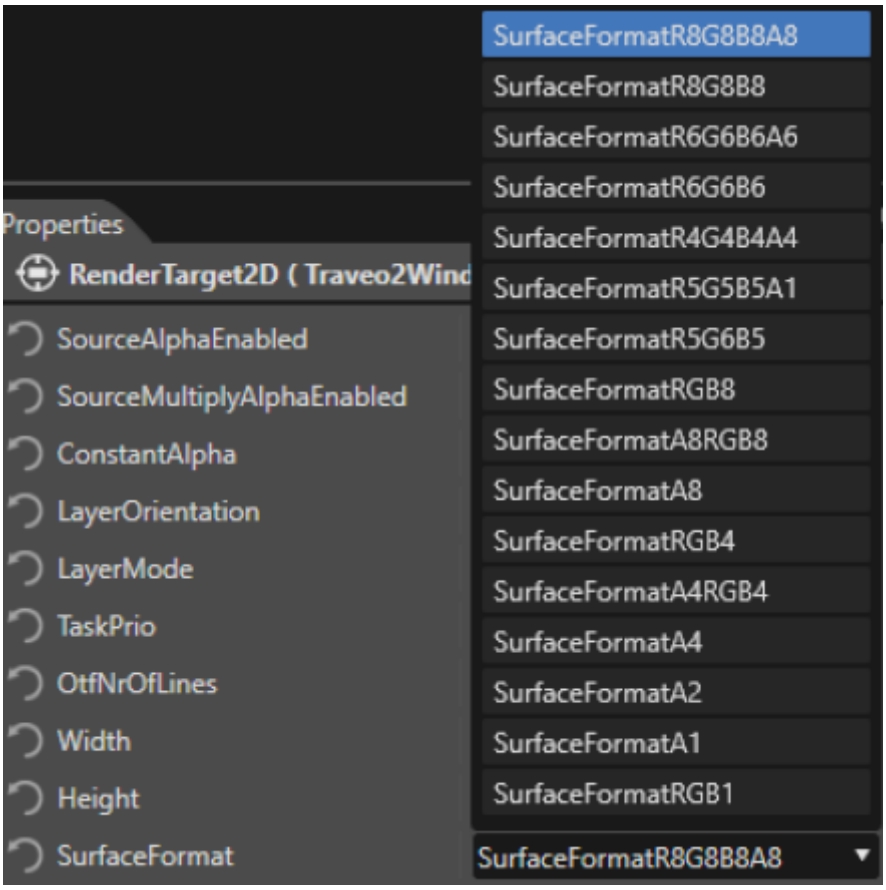
    - **Note the restrictions/limitation:**
        - All source images shall be copied to VRAM!
        - Maximum OTF width is limited to 1600!

    - **Best practice:**
        - Use as less as possible OTF layers to ensure the targeted performance.
        - Attention to the number of line buffers: There is only a setting of a power of 2 available for the number of line buffers. 16, 32, 64, 128, ... because this will increase the VRAM consumption significantly!
        - Use OTF layers whenever you must draw high frequently changed content (e.g., gauges, pointer, ...)

- **Notes for the different modes (IBO/LBO/OTF)**
    - If necessary, it is possible to mix the usage of different modes (IBO and OTF) by using several tasks in the command sequencer.
    - Keep in mind that an IBO mode task will not yield the command sequencer until it has completed.
    - A long running IBO task can therefore prevent an LBO or OTF task from completing in time which can lead to visible artifacts.
    - The same can happen if the complexity in one slice is too high to be completed in time.

SurfaceFormat optimization

Before starting the project, it must be decided which format could match the expectations of a customer. If less bits per pixel are sufficient for the look and feel it should be chosen to save VRAM. Keep in mind that for overlapping Layers/RenderTargets an Alpha-value is necessary, otherwise no blending of layers is possible.



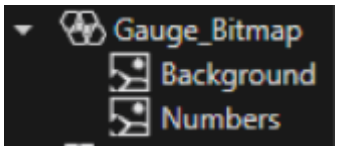
Text

Best practice for drawing text related content:

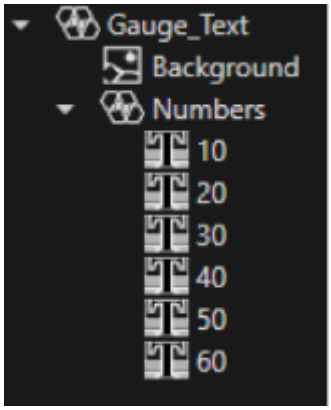
- Always pre-render / rasterize non-translatable text.
- Use static pre-rendered numbers instead of rendering numbers as text (e.g., for Speedometer, RPM, Odometer, ...) Especially for speed use a behavior to switch prerendered digits instead of rendering a text.



Switch prerendered images instead of text

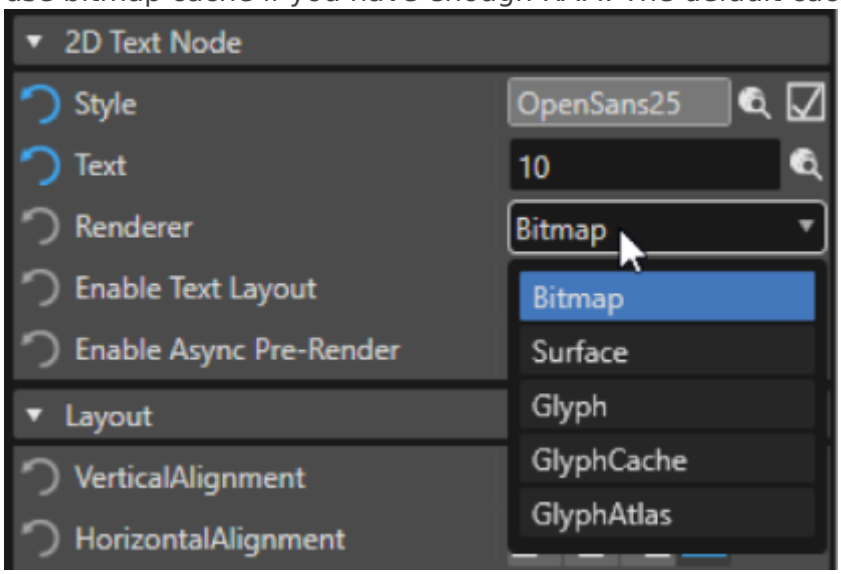


Numbers are combined in one image



Each number as TextNode

- Text cache type can balance RAM and CPU use. Use no cache if less RAM is available or use bitmap cache if you have enough RAM. The default cache type is set to Bitmap:



Revision #7
Created 2025-03-19 04:50:25 UTC by Kanai Tomoaki
Updated 2025-04-17 21:45:22 UTC by Kanai Tomoaki