Large Format Renders

When rendering large images, it's important to understand how your scene will perform. This document covers performance and not quality. Remember that larger renders will reveal more detail and your textures and modeling should hold up at larger sizes.

Since many things in RenderMan are resolution dependent, like dicing and mipmapping, you will find your scene will take more memory to render at larger resolutions. You may also find the memory needed to store the image framebuffer will be large as well. But there are ways to ameliorate some of this memory grab and still comfortably render your image.

To begin, your image format should be OpenEXR as it allows for better management through the EXR driver in RenderMan.

Ways to reduce the workload

- Reduce the number of AOVs you're outputting if at all possible
- OpenEXR
    - Scanline, set to horizontal bucket order, typically most efficient
    - Tiled, do you care about random order buckets?
        - No: disable the rewrite option, **`"string rewrite" "false"`**
        - Yes: leave rewrite on but you should expect a memory spike at the end
- Turn off incremental rendering, this prevents many things from being in-flight at once. Note some Integrators generating photons may enforce incremental
- Avoid using *custom* display or sample filters as we cannot deallocate the framebuffer
- Avoid PxrEdgeDetect as it invokes a read region callback and prevents deallocation
- Avoid Cryptomatte, but if it must be used, you may reduce the accuracy of the merged samples. This causes it to merge more close-by samples as the image is rendered
- Render to crop regions and stitch later (note that anything generating photons like PxrVCM and PxrUnfiied may have a different result using photons as we only generate them for the crop region shown and not the whole scene.)
- Trick: Turn the image to the side so it can be narrow, this allows each scanline horizontally to complete sooner and release more often