Thread Control in Katana

Setting Traversal Thread Count

As RfK converts the Katana scene graph into the Rix scene graph for rendering it will do so in parallel, spawning new threads at location branches. The default is "all on" meaning it will fork on every child location in the scene which is unlikely to be ideal for most scenes but allows for deeper control by the author for control of parallelism. There are are varying levels of control which will take some adjustment to come up with a configuration that works with your pipeline and subsequently for individual scenes.

1. **prmanGlobalStatements.plugin.traversalMethod:** The global traversal setting. When set to 'Serial' it will disable parallel scene traversal entirely. Not generally recommended but can be useful for timing comparisons and rentrant diagnostics (i.e. am I seeing this problem in both serial and parallel traversal?).
2. **prmanGlobalStatements.plugin.maxTraversalThreads:** Number of threads requested for traversal: this is the first place to start tuning. By default the max is set to 4 but depending on your machine configuration and your scene layout you may get double or half the performance at this level. For the technically inclined, this is the value used to initialize RfK's TBB task scheduler. Setting this to the TBB default of '0' is **not** recommended. Empirically we have seen significant slowdowns when setting this to '0', likely due to conflict with other TBB sessions in play.
3. **prmanStatements.traversal.forceSerial:** Per-location setting for scene or asset fine-tuning. It is highly recommended to set this true at component or group levels such that the traversal does not try to parallelize down to the gprim level. Please see PrmanObjectStatements for more detail on location-level thread tuning.

> **Performance Note**
>
> Understand that some branches processed serially may improve time to first pixel (the time it takes for the render to begin sending pixels) and is very much scene dependent, but important to understand that parallel traversal on everything may be costly.

Setting Render Thread Count

Currently there are three ways to set the number of render threads in RfK. The first two will trigger a "`-t:#`" addition to the command line arguments for prman which, by RenderMan definition, will override any thread specification in the RIB (the third option).

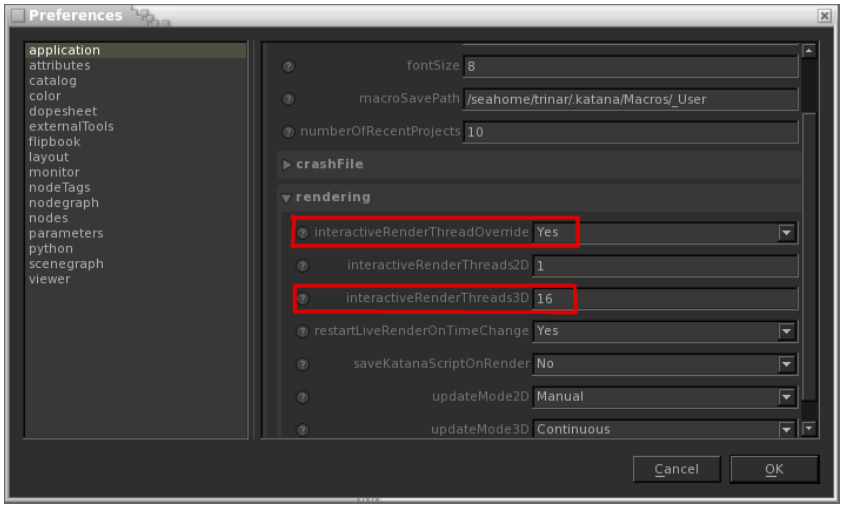
Ways to set thread count, **in order of precedence**:

1. Katana preferences: **interactiveRenderThreadsOverride** and **interactiveRenderThreads3D** (equivalent to the use of the "`--threads3d`" command line parameter in batch mode).
2. Katana attribute: **renderSettings.renderThreads.**
3. PRMan option: **prmanGlobalStatements.options.limits.threads**.

How and where to set these values are detailed below.

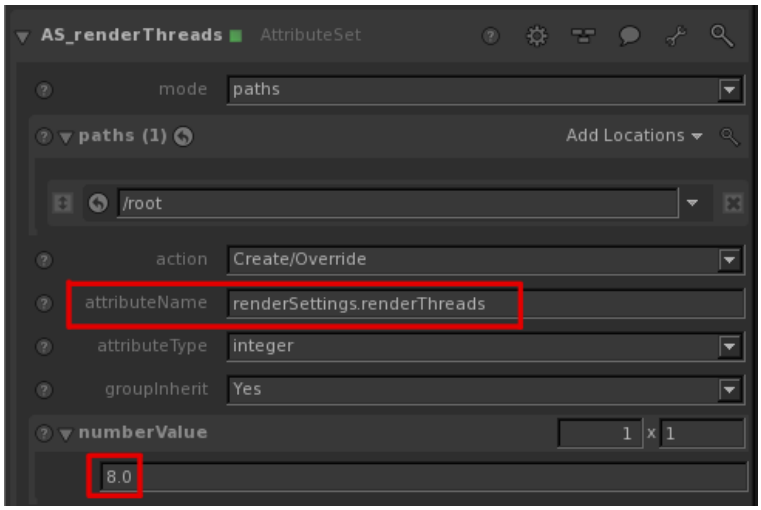
Option 1: interactiveRenderThread Preferences

Below is a screen capture of the preferences **interactiveRenderThreadsOverride** and **interactiveRenderThreads3D** (equivalent to the use of the "`--threads3d`" command line parameter in batch mode). If you want to definitively set the number of render threads this will be the place to do it. Set **interactiveRenderThreadsOverride** = *Yes* and set the value of **interactiveRenderThreads3D** to your desired threads:



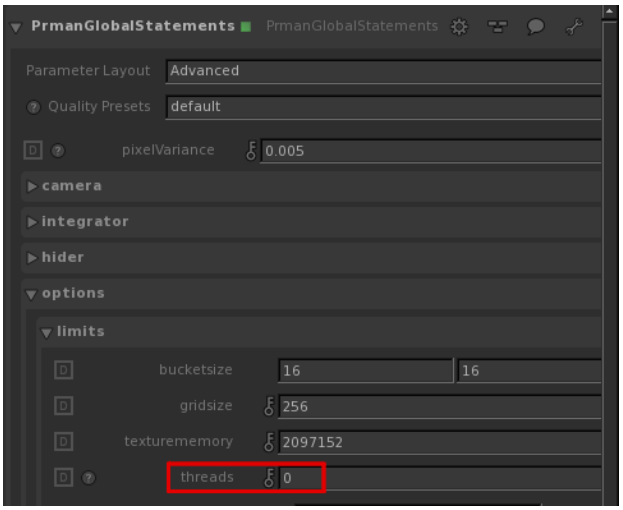
Option 2: renderThreads Attribute

The attribute **renderSettings.renderThreads** must be set via AttributeSet or OpScript as it is not exposed in RenderSettings. Below is an example of setting the attribute with an AttributSet node:



Option 3: limits.threads prman Option

The prman option for **limits.threads** is set via the **Options** section of the PrmanGlobalStatements node:



This attribute translates literally to RIB:

```
Option "limits" "int threads" [N]
```

As it is lowest in the precedence list, this thread option is overridden if either (1) preferences or (2) render settings have a thread value enabled.

Special Values

In all cases, if a value of 0 (zero) is given to prman then all CPUs will be utilized for rendering. If a negative value (-N) is specified then prman will use all but N CPUs for rendering.