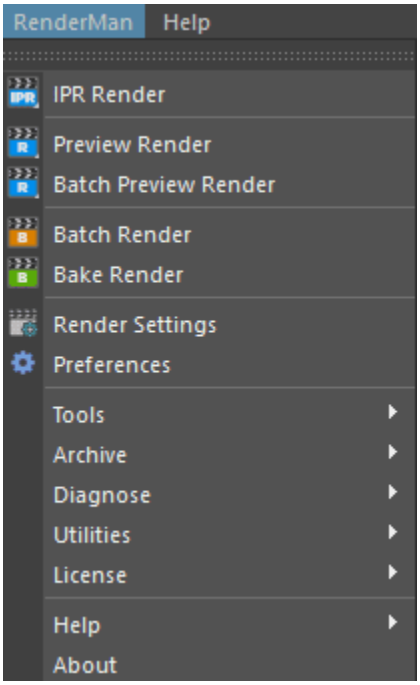
Batch Rendering in Maya

Starting a Batch Render

Batch renders can be started from the RenderMan Menu. Maya's menu entry for Batch Render also works.

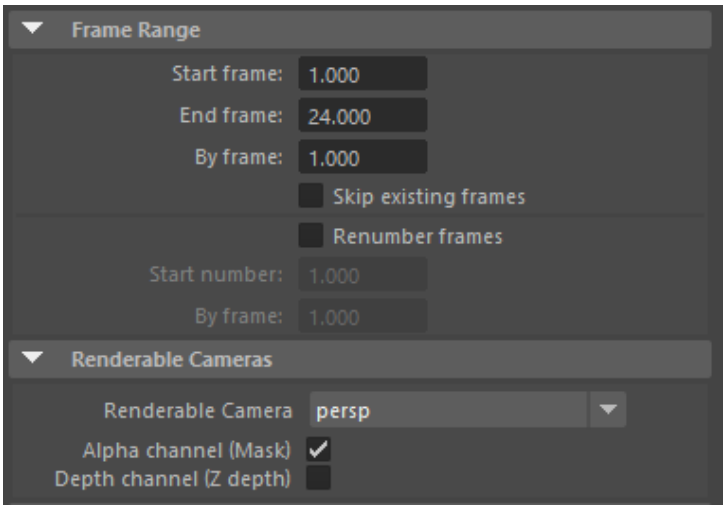


Stopping a Batch Render

Batch renders can be deleted or paused in Local Queue or Tractor. Right-click on the job in the list for a menu.

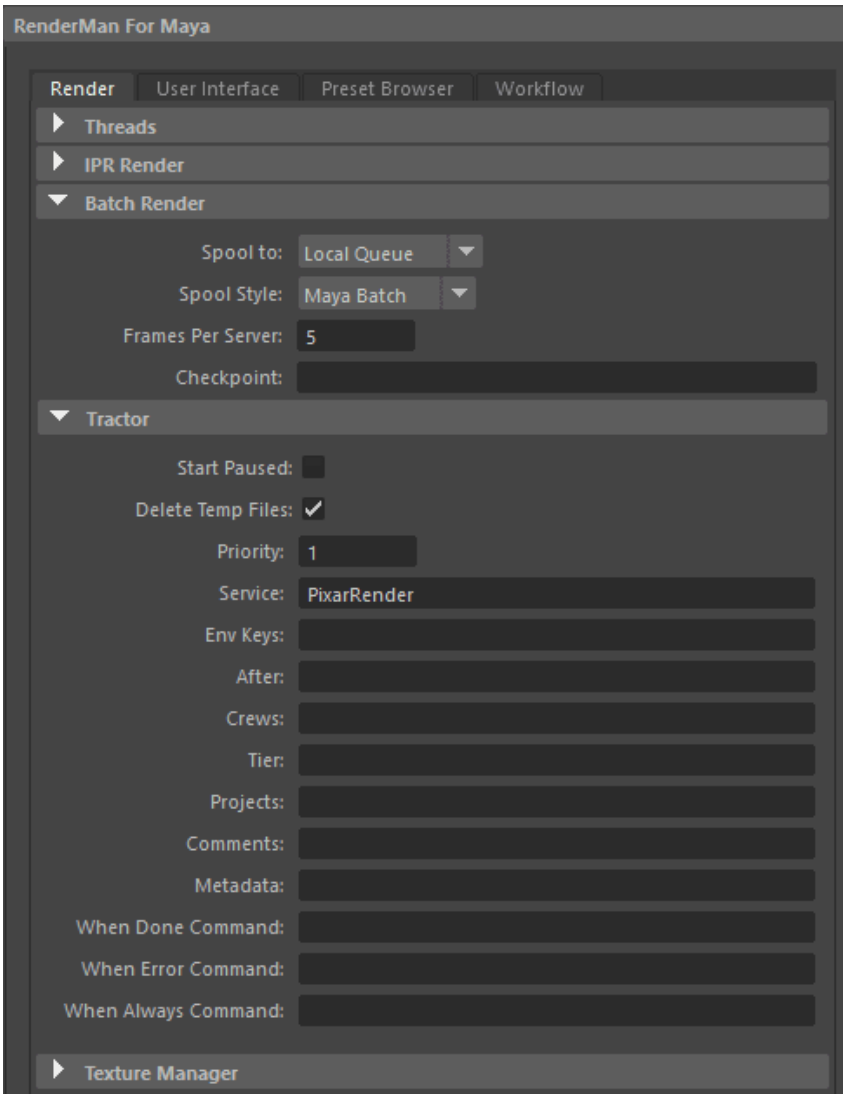
Render Settings

The settings from the Render Settings window are respected for batch renders. In particular, Frame Range and Renderable Cameras are specified under the Common tab. For preview renders the current frame or active camera are used instead.



Batch Queuing

Batch renders are always spooled to either LocalQueue (default) or Tractor. Batch queuing preferences are available in the Maya Preferences window, which opens when you choose the option box for the Batch Render menu item.



Output Locations

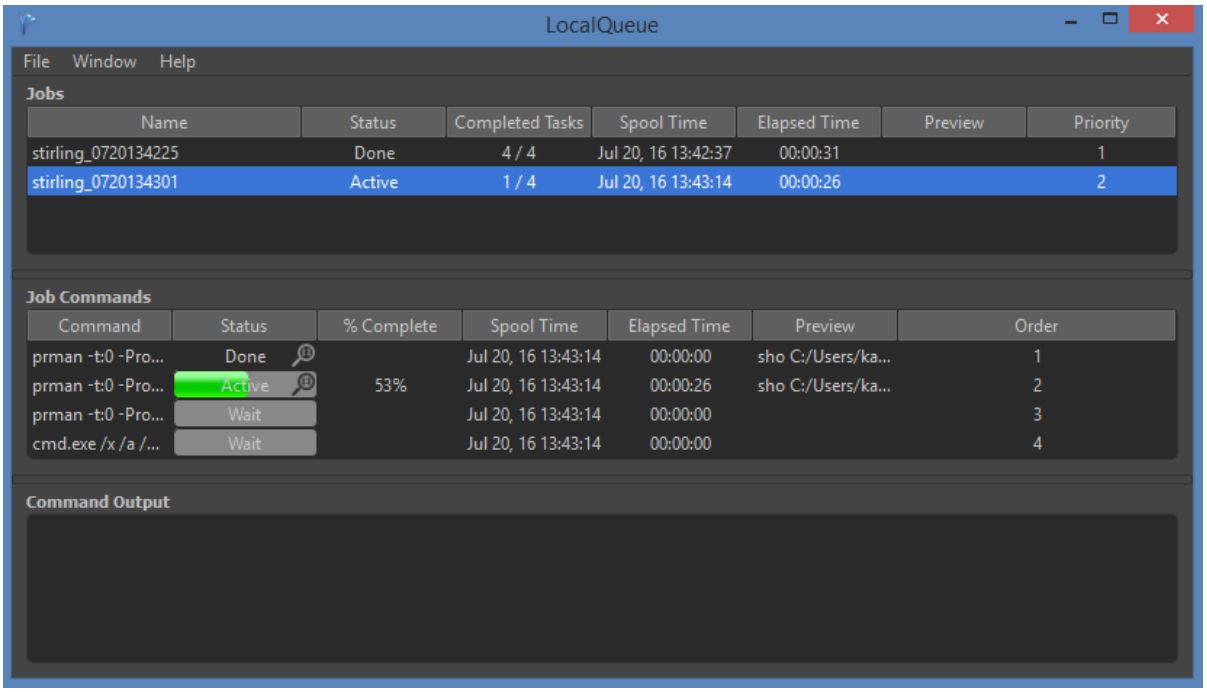
By default, we use Mayabatch to render the saved Maya scene file. Maya renders from memory similarly to a preview render from the UI, without the intermediate RIB export.

When a batch render is started, RIB files and textures are stored on disk and the renderer reads those in. These are stored in a directory based on the scene name but with a time stamp appended. The Batch Context setting in the preferences is where the time stamp is specified as a variable called $JOBDATETIME. This avoids the possibility of multiple simultaneous jobs attempting to overwrite files.

Output locations are configured in RenderMan for Maya's Workspace tab.

Local Queue

LocalQueue is used to run a local queue of render jobs. It reads *job scripts* that are generated by RenderMan for Maya, and then runs the commands from the scripts to render your scene on the local computer.



Once a render job is loaded in LocalQueue, you should see it listed in the *Jobs* table. You can right-click a job to bring up a menu with various options for managing the job. Once you select a job, its commands will be displayed in the *Job Commands* table.

Checkpointing

By default checkpointing is enabled for batch renders. Images are updated on disk every five minutes. The checkpoint interval is configurable in the batch preferences. Incremental mode is enabled by default, under the Sampling tab in the Render Settings. This is necessary for checkpointing to work.

Maya Batch Rendering from the Command Line

You may also use additional flags: -rl (render layer), -crop, -preRender, -postRender, -preLayer, -postLayer, -preFrame, -postFrame, -jobid

From a command line use the following:

```
Render -r renderman sceneFile
```

It is also possible to only generate RIB without subsequently rendering.

```
Render -r renderman -rib sceneFile
```

A complete list of the options can also be seen by running:

```
Render -r renderman -h
```

If you get a warning like the following, you need to put rmanRenderer.xml and ribRenderer.xml from the RenderMan for Maya installation in a place where Maya can find it.

```
Cannot open renderer description file "rendermanRenderer.xml"
```

You can copy the files from the RenderMan for Maya installation,

```
eg. C:/Program Files/Pixar/RenderManForMaya-22.0/etc/rendermanRenderer.xml
```

Into the directory where Maya looks for these under the Maya installation,

```
eg. C:/Program Files/Autodesk/Maya2018/bin/rendererDesc/
```

Or you can set up the Maya environment variable called MAYA_RENDER_DESC_PATH so that rmanRenderer.xml and ribRenderer.xml will be found.

Prman Rendering from the Command Line (Advanced)

When doing a maya *batch* render, RenderMan for Maya generates RIB files and then prman (executable) is launched for those rib files. However, you may already have RIB files on disk and just want to run prman on them.

Job Structure

The RIB files generated by RenderMan for Maya live within the maya project. For example:

```
<maya_project>/renderman/myscene/rib/
```

RIB files are organized into subdirectories for each frame, like 0001, 0002, 0003. There is also one subdirectory called "job" which is for caches of static objects, and processing commands that do not need to happen every frame, such as converting textures and cleanup.

Here is an example of the commands that would typically happen for a three frame job. Note the use of the -cwd arg to specify the maya project as the current working directory that prman should run out of. The project relative path to the rib file is supplied as the last argument of the command. Rib files contain project relative paths by default.

```
prman -t:0 -cwd C:/Users/user/Documents/maya/projects/default/ renderman/test_0725120019/rib/job/job.rib

prman -t:0 -cwd C:/Users/user/Documents/maya/projects/default/ renderman/test_0725120019/rib/0001/0001.rib

prman -t:0 -cwd C:/Users/user/Documents/maya/projects/default/ renderman/test_0725120019/rib/0002/0002.rib

prman -t:0 -cwd C:/Users/user/Documents/maya/projects/default/ renderman/test_0725120019/rib/0003/0003.rib

prman -t:0 -cwd C:/Users/user/Documents/maya/projects/default/ renderman/test_0725120019/rib/job/post.rib
```

Each frame rib file, like 0001.rib references other rib files located in the same directory, for each camera or render layer that is active for the frame. Here is an example of typical contents of a frame directory:

```
renderman/test_0725120019/rib/0001:

0001.rib

perspShape_Final.0001.rib

perspShape_Final.0001.rlf

perspShape_Final.0001.xml
```

The file called 0001.rib is the "driver" RIB file for the frame. It will reference RIB files for each pass (for different cameras or render layers) that occur in the frame.

There are some files in the directory that aren't RIB files. What are those?

The user made RLF files (assigned for GPU archives) contain material and binding information. Materials are injected into the RIB file by a RIF at render time.

The XML file is generated during rendering. It contains diagnostic information about the render.