Tips for working in complex scenes: by Deborah R. Fowler

(These are guidelines, and ultimately like many things in Houdini there may be several right answers and often "it all depends").

1. ***Display only what you need.***

   Working in a complex scene may require you not to see all the geometry that you are creating. You may even be using "instancing" to delay the geometry until render time to further optimize rendering. H13 has made this even easier!

   If you are working on a shader, don't work with all the objects displayed, turn on only the object you are working on, you can also easily copy that element into another file to work on it unconstrained.

   If you are working on layout or lighting, you can use prototypes using Houdini's switch node to easily have the first frame for quick previewing and subsequent frames for rendering.

   Take advantage of the render only and display flags combined with the null node as discussed in class.

   In addition, if you don't need to display it in shaded mode, use wireframe, your interactions with the software display will be faster.

2. ***Render small resolutions for tests.***

   Unless you are working on texturing/shadow/sampling parameters, render small tests for quicker feedback. You shouldn't be waiting long periods of time to test lighting/shading changes – there is no need to render the entire scene huge. Work smart == work fast.

3. ***Optimize your textures.***

   Houdini's mantra render works most efficiently with RAT files. It takes but a few seconds to bring your tif, jpeg etc. source file into **mplay** (bottom of the Render tab on the main menus) and save it as a **rat** file. Most importantly, make sure the size is reasonable and appropriate. Large files being accessed for displacement, color and so on are going to take time, so make sure these files are not huge, particularly when you have multiple items using these files. (Start with 512 or smaller, size up only if artifacts appear and it is necessary). Again, use your judgment to work smart.
   (mplay is analogous to Maya's fcheck but has different features).

4. ***Select the render that is optimal for your scene, split up your scene if necessary.***

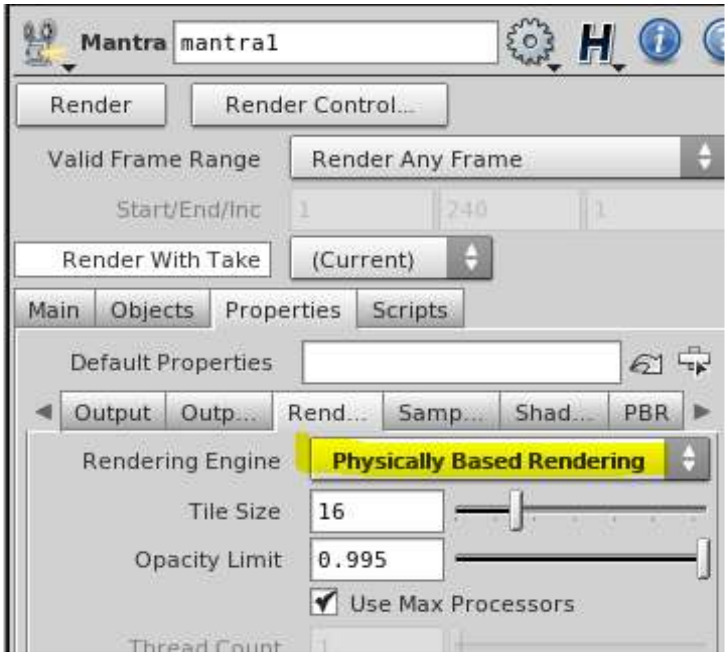
   Also it can be helpful to split up your scene into render layers, and if you so desire render passes. Different layers may be using different rendering/shading algorithms.

   For examples, a scene with a lot of reflections or refractions can be rendered in PBR, whereas an element of the scene that is using sprites to create smoke should be rendered using Micropolygon render. NOTE: Now for 99.9% of what you do **use PBR!**

5. ***Brief rundown of the rendering options in the Mantra node:***
For 99.9% of what you do, you should use Physically Based Rendering.

It may look as though there are many choices, however they are really a mix and match of rendering and shading algorithms.

PBR uses raytracing and physically based shaders.



Mantra mantra1

Valid Frame Range: Render Any Frame

Start/End/Inc: 1 240 1

Render With Take: (Current)

Default Properties:

Rendering Engine: Physically Based Rendering

Tile Size: 16

Opacity Limit: 0.995

☑ Use Max Processors

Thread Count: 1

History of Options:

- **Micropolygon Rendering: - equivalent to old renderman's reyes algorithm**
  -Scenes with huge amounts of complexity that cannot be raytraced efficiently
  -Scenes with lots of motion blur or depth of field – computationally expensive shading happens once and is sampled in space/time
  -Scenes with lots of semi-transparent elements (volumes, sprites, points) – this sort of layering can be slow to raytrace
  -Scenes with displacements
  (each primitive diced up into micropolygons which are shaded and sampled independently)

- **Ray Tracing – can think of the this as "Mental Ray"**
  -(PBR uses raytracing, but select the PBR option)
  -Scenes with reasonable levels of complexity – as complexity increases scenes become exponentially (give or take) slower
  -Scenes that require raytraced shading effects (such as reflections, refractions)

-Scenes that require very sharp details in shading – micropolygon rendering tends to introduce softness by nature of the underlying algorithm
(sampling by sending rays from the camera. Each surface hit by a ray will trigger a surface shader execution.)

- **Micropolygon Physically Based Rendering – DON'T USE**
  -designed for scenes with lots of motion blur
  (sampling by micropolygones, shading and illumination physically based.

- **Physically Based Rendering – *****USE THIS ONE*****
  (sampling using raytracing and shading using physically based.
- **Photon Map Generation – for backward compatibility – why bother when PBR does it all**