The Rendering Workflow

Kyle Climaco

Introduction

So in general what rendering is to have a scene description *(in our case a 3D scene in maya)* and create a 2D image. The computer has data such as the camera, objects, textures, shaders and lights and they all interact mathematically to answer the question, “What color is this pixel?”. I am here to talk about these parts and how they all fit with one another. I won’t go into too much detail with all of it just how to prepare these parts for Renderman in Maya, some essential fundamentals you need to understand, and a few helpful tips to speed up each part. By the end of this paper you should be able to understand the problem between time and quality and what that entails. As Well as how rendering fits into the pipeline.

This paper has three parts, Preparing a scene for rendering, understanding optimizations for noise and render time, and features from Renderman that we use in compositing.

Rendering Checklist

This was the checklist every lighter had to follow in some what chronological order to prepare a scene for rendering. I will be going into broad strokes and "translating what the documentation means" but really you should read the documentation on Pixar's website since it will be the most up to date. Eventually these specifics will be out of date but the general workflow will still be relevant in Renderman. I will also use this as sort of a table of context but shuffle some lines so I can group them were I saw fit.

- Scene
    - Geometry
        - Baked Animation
            - Characters
            - Props
        - Playblast and check Animation
        - Renderman Catmull-Clark subdivision scheme(On Applicable Models)
        - Imported References
        - Max Specular depth Overrides for Glass
        - Relative pixel Variance Optimization
    - Shaders
        - Named Texture nodes, material nodes, and shading groups
        - Baked Textures
        - <ws> relative reference for file paths
    - Lights
        - Light linking
        - Light Blockers/filters
        - Light Groups (limit to six)
            - Char_Key_LG (put eye dings here)
            - Char_Fill_LG
            - Char_Rim_LG
            - window_LG (for BA hallway)
            - HallwayScance_LG
            - HallwayFill_LG
                - or Hallway_LG
            - Fire_LG
            - Env_DayLight_LG (daytime)
            - SpotLight_LG
            - Env_LG (in night time)

        - light samples
    - Cameras
        - Name Space Deleted
    - Any Applicable Effect

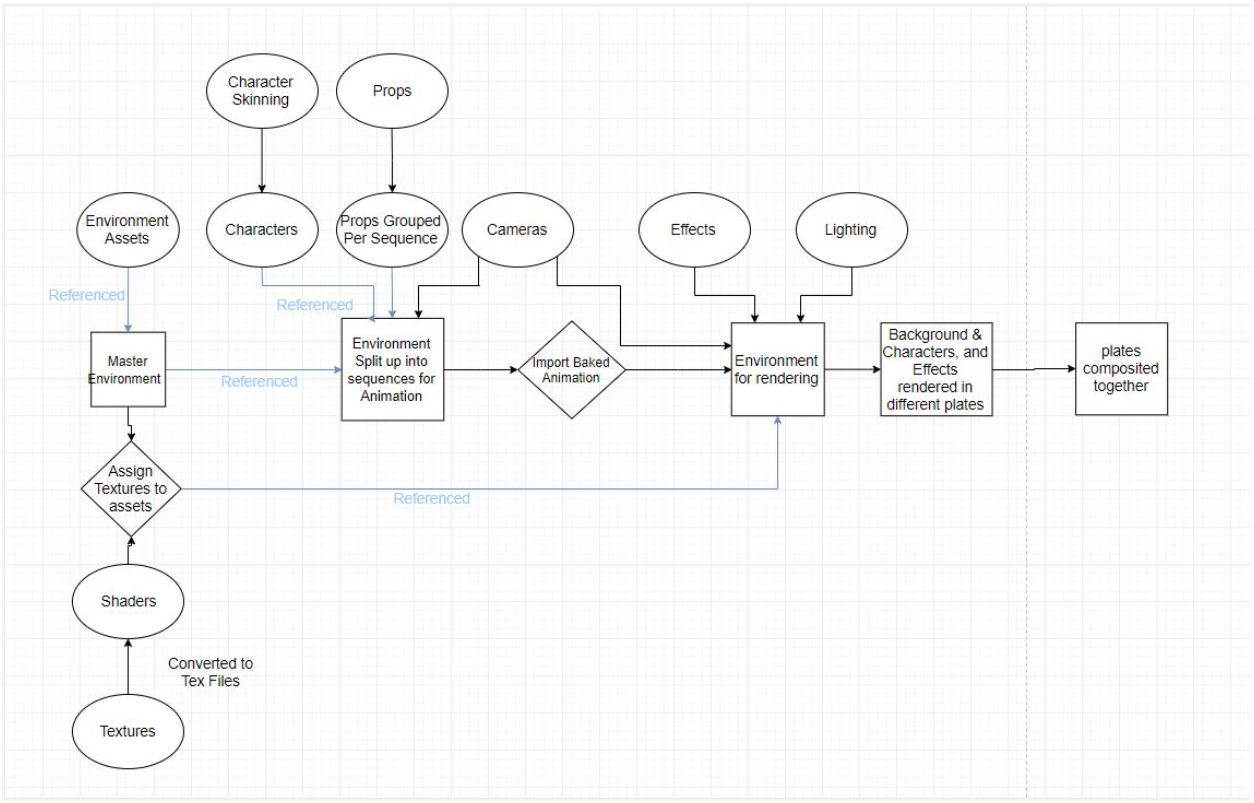
- Render Settings
    - Render using Renderman
    - Frame Range for scene
    - Render: Animation
    - Image file format *(For certain maya versions)*
    - Resolution Preset HD 1920x1080
- Renderman settings
    - Sampling: *Quality*
        - Min samples: 16
        - Max samples: 64
        - Pixel variance: .06
        - Dark Falloff: .025
        - Integrator Pxr Path Tracer
        - Max Depth 6 (can be reduced with no glass in shot)
        - Indirect Samples 1
        - Max specular Depth 1
        - Max Diffuse Depth 1

    - Sampling: *High Quality*
        - Min samples: 16
        - Max samples: 128
        - Pixel variance: .03
        - Dark Falloff: .01
        - Integrator Pxr Path Tracer
        - Max Depth 10 (can be reduced with no glass in shot)
        - Indirect Samples 3
        - Max specular Depth 6
        - Max Diffuse Depth 2
    - Features
        - Motion Blur: 2D Motion Vectors
        - Camera Blur: Checked
        - Motion Samples 0
        - Cryptomattes
            - path
                - <imagedir>/cryptomattePath.<f4>.exr
            - Material
                - <imagedir>/cryptomattemattr.<f4>.exr
    - AOV
        - Ci
        - a
        - NgN
        - p
        - DP Camera Time
        - dPdtime

        - Shadow (write below manually, not copy and paste)
            - lpe:shadows;C[DS]+<L.>
        - Albedo
        - emissive
        - MatteID0(eyes)
        - MatteID1(Theresa)
        - Z
        - light Group Dependant Passes
            - Direct Diffuse
            - Indirect Diffuse
            - Direct Specular
            - Indirect Specular
            - Subsurface
            - Single Scatter
            - Transmissive Glass Lobe
            - Emissive(fire)
            - (These needs to be merged *minus* out of the respective specular passes to be tuned individually)
                - *Direct specular Clearcoat lobe*
                - *Indirect Specular Clearcoat lobe*
                - *Direct specular iridescence lobe*
                - *Indirect specular iridescence lobe*
    - Advanced
        - Reetreant procedural: checked
        - Output Holdout Matte: As separate AOV (only if we do holdouts)
        - Water Tight Dicing: Checked (selective to displacements)

- Batch render one frame
    - optimize light samples
    - optimize indirect diffuse

Our Production Workflow



(sorry the text is so small zoom in if you are viewing in drive)

THE MOST IMPORTANT PART OF THIS PIPELINE IS DON'T ANIMATE WITH THE SHADERS IN THE SCENE.

DON'T LIGHT WITH THE RIG IN THE SCENE. *(unless you are prelighting the scene and exporting the lights because your turn over for that scene is literally in one day).*

Project Managers! All right so to put this into context before I dive into our specific pipeline certain things I say will be irrelevant depending on how you do your pipeline, so use some critical thinking. Each piece of this tree is a maya scene so get into a good habit now because these scenes get heavy right before rendering them and all the gunk nodes will pile up in Maya. Spheres are typically in the assets folder *(or in a specific folder like textures are in sourceimages folder)* and squares are in the scene folder.

In context for rendering we had to get the scene file with all the textures assigned and baked in a file, the animation file with everything referenced, And the base lighting such as the lighting from the environment and in our case the lights from the scances.

Preparing a Scene for Renderman 22.3 and Maya 2018

These specifics will be outdated as Pixar and Autodesk love to update everything and making it more "streamline" for the user. My goal here is to show you these concepts and how it was done for this version so you can take these concepts and apply it to your current version.

rendering optimization
Compositing workflow
(I will explain these steps later in the paper)

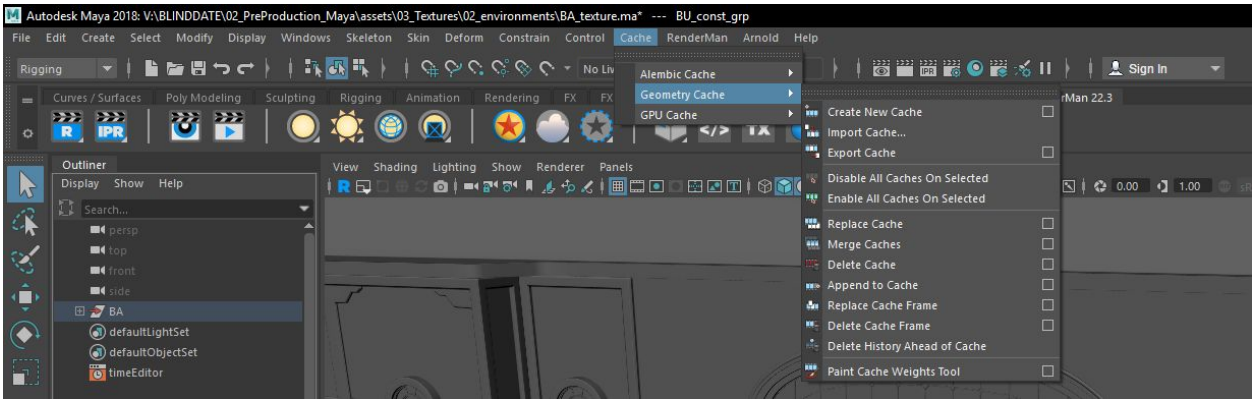
- Scene
  - Geometry
    - Baked Animation
      - Characters
      - Props
    - Playblast and check Animation
    - Renderman Catmull-Clark subdivision scheme(On Applicable Models)
    - Imported References
    - Max Specular depth Overrides for Glass
    - Relative pixel Variance Optimization
  - Cameras
    - Name Space Deleted
  - Lights
    - Light linking
    - Light Blockers/filters
    - Light Groups (limit to six)
      - Char_Key_LG (put eye dings here)
      - Char_Fill_LG
      - Char_Rim_LG
      - window_LG (for BA hallway)
      - HallwayScance_LG
      - HallwayFill_LG
        - or Hallway_LG
      - Fire_LG
      - Env_DayLight_LG (daytime)
      - SpotLight_LG
      - Env_LG (in night time)
    - light samples

Baking Animation(ACTION!)

I will assume that you already have a clean environment file with all the textures assigned and a character geometry file with textures assigned with no rig correctly scaled to the scene rig. As this will be covered in Kevin Barwicks Paper.

So in film it is “lights, camera, action!” but in digital 3D Animation we flip it and say “action, camera, lights!” Our first step is to start with open the animation file and bake all of the animation out of the rig. Renderman does not like how things are rigged in renderman specifically when it comes to locked objects in a parented hierarchy and anything that is constrained. We have three methods for baking animations and you will need to troubleshoot what method will work best for the what rig.

We have:



Geometry Cache method

We found this method worked the best on character

Steps:

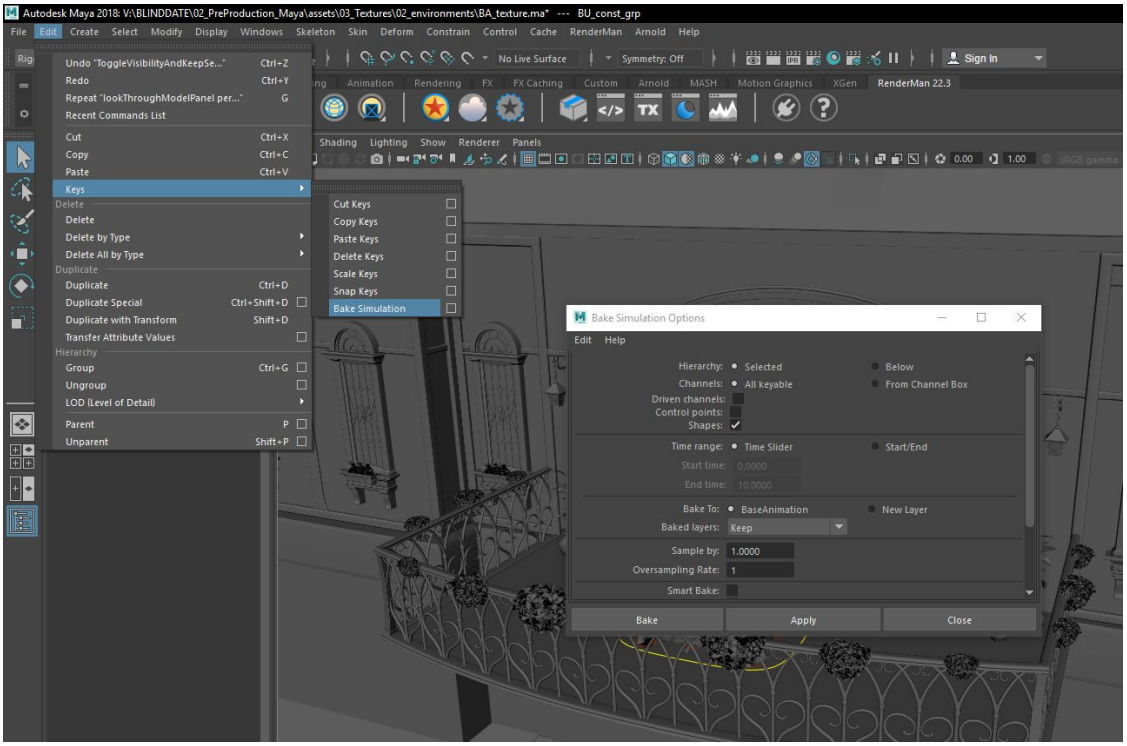
1. import any references under file>reference editor
2. select all geometry of the rig (**NO CONTROlLERS)**
3. cache > Geometry Cache > create new Cache
4. change cache directory to .../ncache/s000_00
5. cache name to CH_geo *ex: GO_geo for Gordon and TH_geo for Theresa*
6. cache time range: Start/End
7. input correct frame range
8. **Close animation file and open environment scene with and shaders attached**
9. Import Character geometry file with shaders attached
10. select all the Character geometry and import Cache
11. locate XML in the ncache folder and import Cache
12. check and playblast animation
13. delete namespace under windows > general editors > namespace editor
14. Save this file s000_00_Lighting

Pros:

- scene runs faster than the bake simulation method
- The rig is not present in the file sent to render
- renderman can accurately calculate vertex motion vectors

Cons:

- When distributing renders to multiple machines every file needs the ncache file
- when selecting geometry for caching the amount of objects need to match
    - Ex: The rig we had eye covers hidden and the model didn't so importing cache gave errors so we



Bake Simulation Method:

We found this works best with props

Steps:

1. import any references under file>reference editor
2. select either the geometry or the control curve of the props(Troubleshoot which works for the specific prop for renderman)
3. Go to Edit > keys > Bake Simulation select the square for the option box
4. checkmark Control points.
5. Time Ranged to Start/End
6. input correct frame range.
7. this will create a key every frame for all the controllers or all the vertices selected
8. select to top group node and File > export selection and name appropriately.
9. import the object to your lighting scene and delete its namespaces.

Importing cameras (Camera!)

Now that we have that finished it's time to import your cameras and the rest of your base lighting.

Ideally you want to have your camera's all in separate files but what ends up happening is the animator will tweak the camera specifically for the the shot and it is your job to get the most up to date camera. This is all workflow dependant so make sure you communicate with your Layout lead and the animator for that shot to see if you have the most up to date camera.

DELETE THE NAMESPACE FOR THE CAMERA we have had issues rendering with camera namespaces this might not be relevant in the future but with this version it was giving us errors and it took hours to solve.

Lighting for Animation (Lights!)

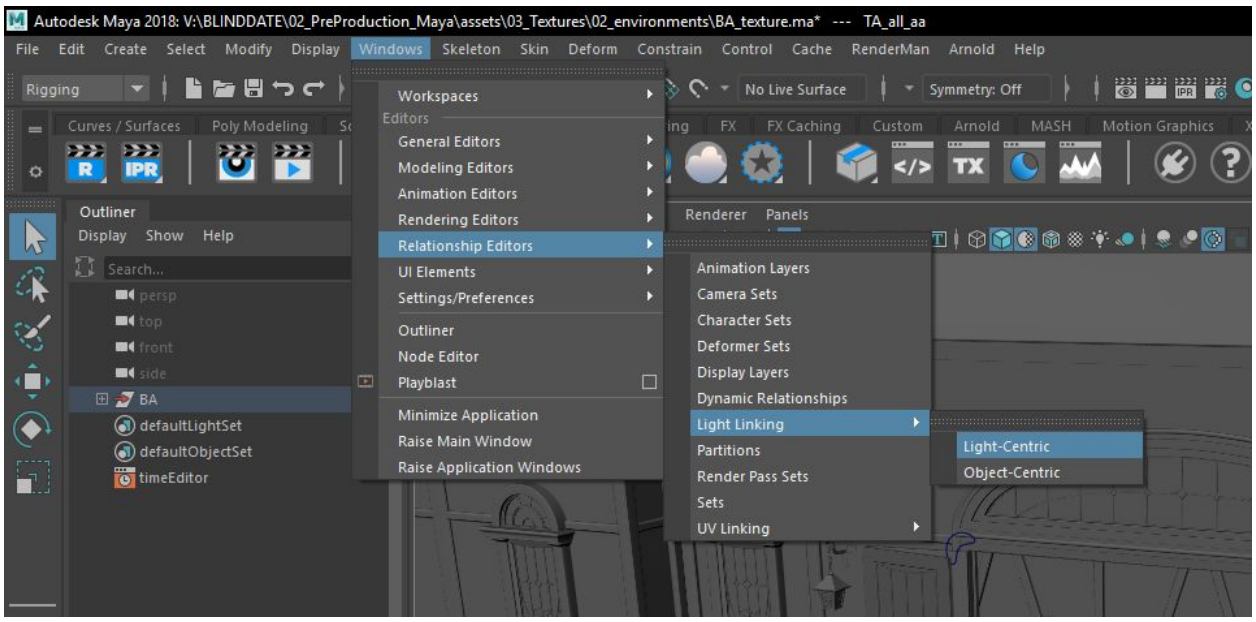
As for the lighting we closely followed suit with how Pixar does their lighting which is Color script > Master lighting > Shot lighting. Master lighting is to keep scene to scene consistent and Shot lighting is to make sure characters are correctly lit and any tweaks your lighting lead might want. **We did not have light rigs for characters in the rig!** one of the fundamentals of lighting is motivated and unmotivated lights. I wanted the shot lighter to be mindful of what their doing and add them by hand. Having a key light follow a character when there is no logical light in the scene will make the character feel like they are not in the environment at all; like in a bad way. There is a time and place for everything but make sure it is with reason. The lights you add after the base lighting should be a rough extension to these lights. *(for the most part of course. It's art you know how these "rules" work. I mean look at the show lighting like where does that light even come from?)*
Starting from Top left Clockwise: Day Time, Night Time, Show Time, and Hallway lighting.




I won't go into to much detail in how to light a scene artistically but **Lighting for Animation: The Art of Visual Storytelling** by Michael Tanzillo should be your bible when it comes to lighting. it will cover everything you need to know.

Light linking in Renderman 22.3 for Maya 2018

You want to minimize light linking, done poorly and the scene is not physically accurate. It's actually relatively simple how to light link. Character Lights need to be light linked to their specific character this will pay off when we get in to light groups in the AOV section.

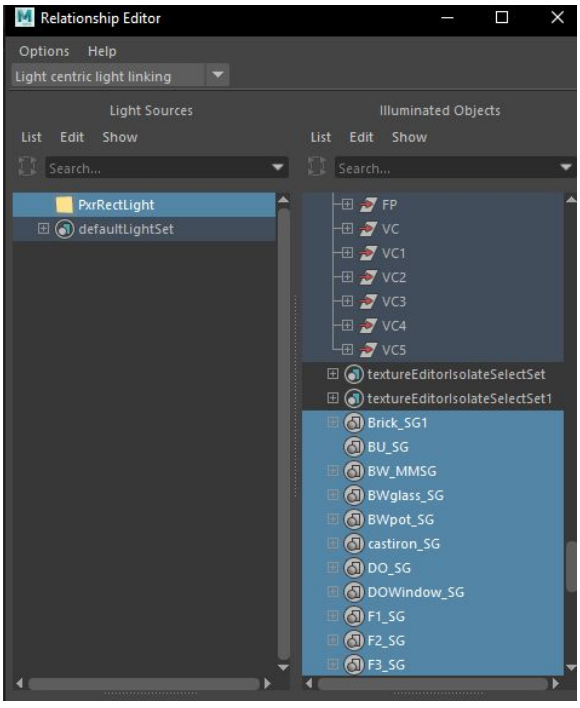


Windows > Relationship Editors > Light Linking> Light-Centric

select your light on the left and whatever is highlighted on the right is what it is affecting.

Notice there are shading groups in here and the the group node for objects. This is why it needs to be organized and shading groups named properly.

Light linking implementation for Renderman is also kinda clunky so sometimes it matters to have both the object node and the shading group for the respective objects to be unlinked. If the light still won't unlink ,create a new light, match transforms and settings and delete the old light.



Light Filters

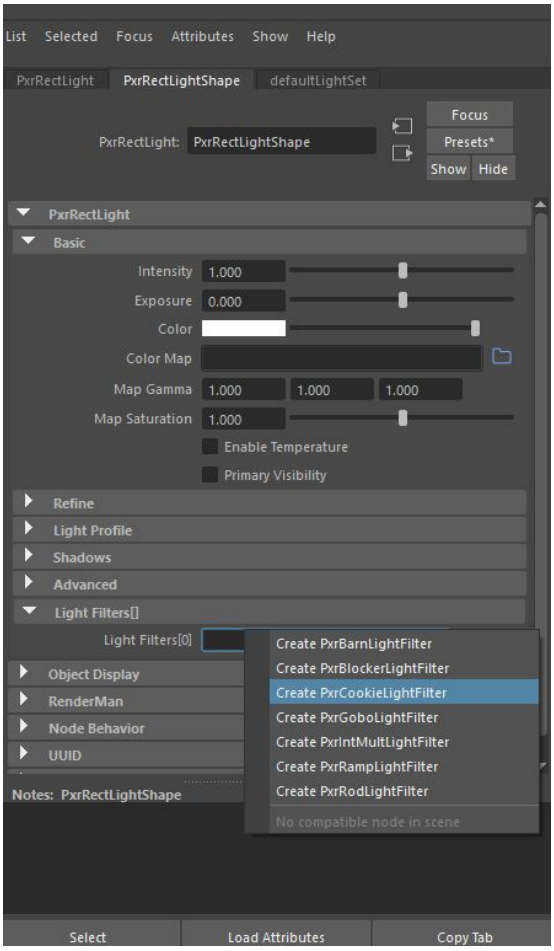
Light filters are pretty self explanatory to understand what they are. In these scene we used a cookie light filter to give a variation in value and make it feel like it was a bright day out. This was the image we used for the cookie light filter and this is the final shot that made it into the film.

This filter was on the environment light filter and we used the analytic mode and added some blur and changed the intensity of the filter and the position on screen. This filter is very powerful and I would suggest reading the Pixar documentation but I hesitate on having a link as this will be broken in the future.

Image provided by: Angelica Duvic

Steps:

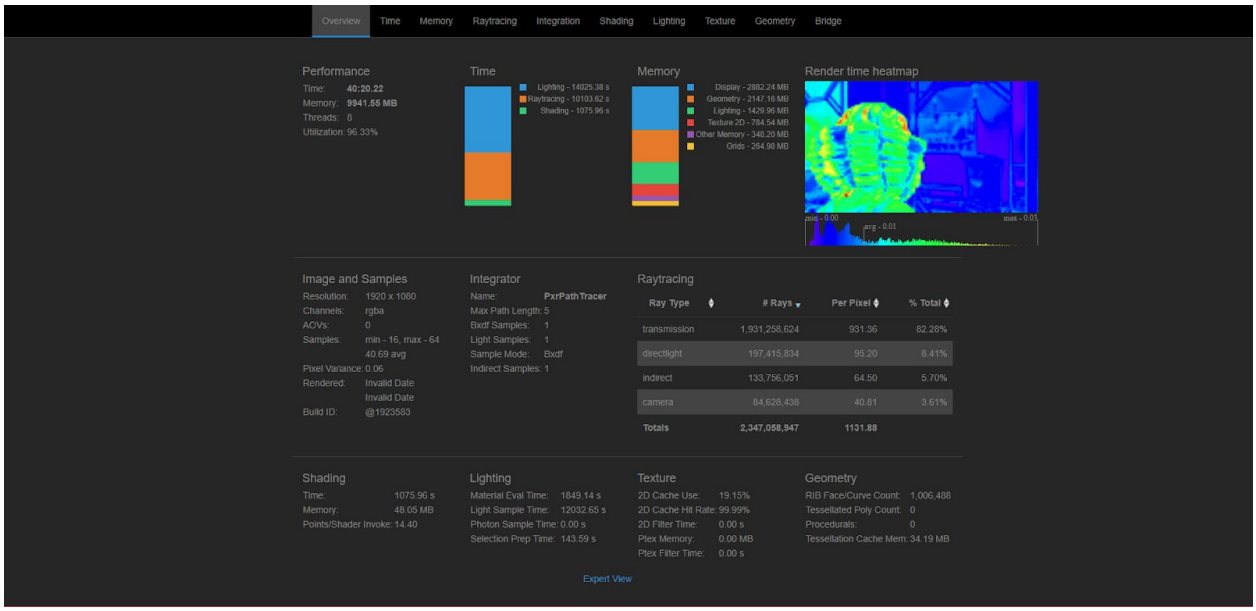
1. Create or select the light you want to apply the filter to.
2. In the shape tab for the light go to Light Filters[]
3. right click on the textbox and select what filter you want to use.
4. Maya will create a new node for the filter
5. to clean this up we parent the light filter to the respective light but really you can put it anywhere.
6. name the light filter appropriately.



now there are seven filters you can play around with each giving a different result. Depending on Pixar's developers moods they might add, change or combine these in later iterations. They might even put this tab in a different place entirely. If it's not here then the documentation will tell you where they moved it. I can't predict what they do in the future but this is the best I can do.

But the filter I actually use the most is the IntMult Light filter oftentimes you want to "brighten up the characters" but you still want to make it feel believable. We would light link Theresa or Gordon to the int mult light filter and increase the Intensity or the opposite. For example our bright white table is almost always overexposed and you want it to be physically accurate before you tweak it nuke. I would light link the int mult light filter tweak the intensity to say .75 and it still feels natural and is throwing indirect rays accurately.

Understanding and Optimizing Renders



RENDER STATS CAN BE FOUND IN: <projectfolder>/renderman/scene/rib/scene.xml

Open this on Chrome browser. This might change in your iteration of Renderman so make sure you check the documentation

This is about to be a technical deep dive of understanding render times, noise, and all the different parameters we used to get the film finished by premiere. Really one or two people need to understand this part of the process because it is highly involved and tweaks are as easy as inputting different values in boxes to make a significant outcome of the final picture.

To start out simply we could have easily had 30 second render times and with 9074 *(OVER 9000!)* rendered frames our short would have been done in 75.62~ hours on one machine. But this would mean our max samples would be at 1 and our resolution would have been at like 960x540. The frame would horrible and even the common folk watching our film would notice.

Rendering overview

Now before we get into this I will go over some fundamentals you need to understand before reading on. Pixar path tracer uses an iterative Monte-Carlo Algorithm that picks a value for a given pixel in a certain probability. Every iteration is a guess that gets closer and closer to a value. In our case the RGBA values of that pixel and with each iteration it will **converge** to a certain value. Every Iteration it will get closer and closer to its true value. A set of pixels in this equation is called a **sample** shown as a blue box you see when you IPR.

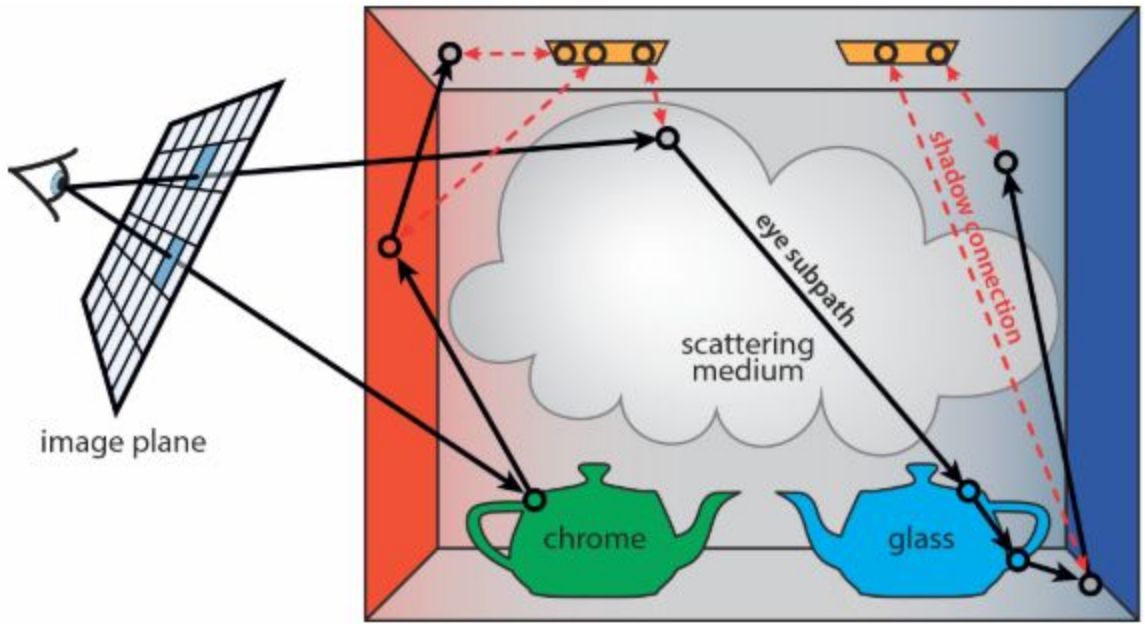
Now Renderman emulates the physical properties of light. Say we were to measure the wavelength of a certain area in space we would find that the exact frequency of light will get to a point where there are too many places after the decimal that are significant. In **adaptive sampling** there is a cut off point that we can specify its called **pixel variance**. The best example I can give that we use "everyday" is PI. We are told that 3.14 is enough for most application but of course the value of pi is a *lot* longer like 3.14159265359..... like an infinite amount of numbers. Now we don't have an infinite amount of time to wait per frame this is where settings come in.

All the settings will change how long or how noisy the render is. Materials, lighting, and the composition of the geometry based on the lighting will also affect how long it will take to render or how noisy it gets. An example of a heavy hitter to render times is subsurface scattering, it takes both longer to calculate each pixel and takes a long time for it to converge. I won't be going into details of how hard each feature in Renderman is expensive but to give some examples, volumes, refractive objects such as glass, and xgen curve hairs. But every scene will have indirect lighting and I will go into detail in how we dealt with it.

To put this all into context our typical scene using the quality settings provided in the render checklist averages to about 20 minutes per frame, with our entire scene and characters. These were on our computers that have a i7 6700k, 4 cores and 8 threads, the cinebench scores are about 888 according to https://www.cpu-monkey.com/en/cpu-intel_core_i7_6700k-518. Cinebench is a great benchmark because it's literally a rendering benchmark that is widely used for processors. 64 samples is the lowest I would advise going with the scope of our production. for the High quality we were typically looking at 40 minutes to an hour on a single machine. But this would typically render in 10-15 minutes on the "Renderfarm". We have 3 server racks with 2 Intel Xeon Gold 6138F processors and 2 server racks with 2 intel Xeon E5-2699V4! These cinebench at 3310 for the Xeon gold 6148F and 3240 for the Xeon E5-2699V4 and there are two of them in each server rack! Now there are some inefficiencies with having two of them on one computer but this averaged out to 5-6 computers per server rack if tuned correctly. I will menton how later on.

We don't have a multi million dollar Renderfarm like Pixar but we can try! I've dubbed ours the render garden because its small and feels very personal. Anyways now to put them into perspective, the production preset in maya is 64 min samples, 4098 max samples a a pixel variance of .005. As of 2019 of this program can not match those aggressive settings. Maybe one day when UCF wants to throw money at us. it would take 6 - 10 hours on the servers per frame with those settings but hey a TD can dream right?

Path Tracing

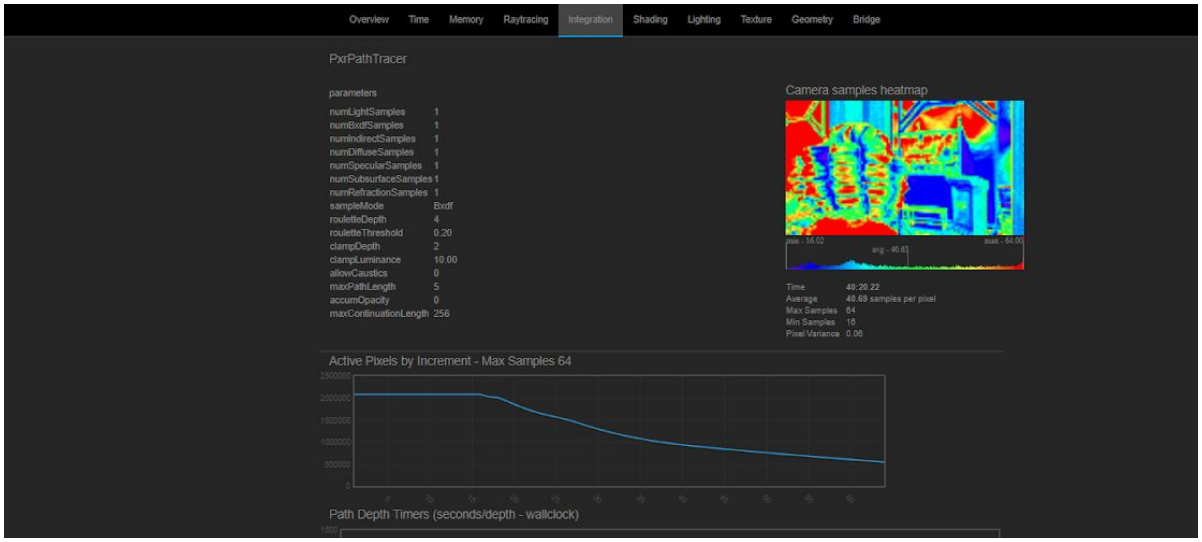


Path tracing can be its own tech paper entirely but here is what you need to know. In reality light bounces scatters, reflects, absorbed, etc. we mimic all of this in computers with complex math. First we generate a ray from the camera for each pixel on the image plane. if this ray intersects an object then it will run an emittance function based on the shader material and then a new ray with a new direction is created. We then compute the bidirectional reflectance distribution function for the material. We then recursively call this function that stops when we reached the maximum ray depth. with all these values we input these variables into to the rendering equation and get the color for that pixel, this repeats if that pixel gets resampled again.

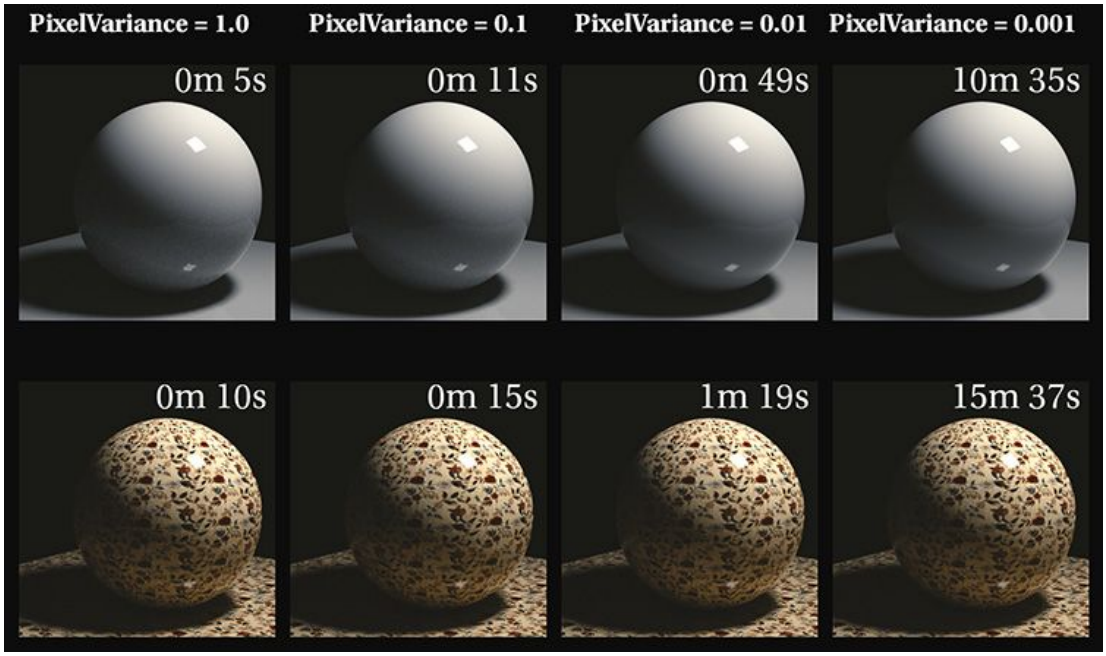
Renderman Settings

In Renderman many default values are either 0 and -1 and these usually mean something special. for example for min samples when set the zero the value of min samples is the square root of the max samples for example. if your max samples is 64 and your min samples is 0 then, min samples = $\sqrt{64}=8$. Alright so let's go over adaptive sampling.

Adaptive Sampling



Not all pixels are equally as noisy so it would be a waste of time to sample all the pixels all the time at some point Renderman will stop and say that pixel is good enough. Using the image above you can see the parts how Renderman stopped rendering at each pixel. Cooler areas means it had a few samples and the hot red areas means it went up to the max samples. The graph shows how many pixels were still being calculated denoted by the Y axis and how many samples were calculated in the X axis. Ideally you want a mix of both and no red areas. what determines how “picky” Renderman gets is a value called pixel variance. you minimum samples is when Adaptive sampling is not applied and your max samples is when it will stop rendering and output an image. **Pixel Variance is the most important parameter for optimizing quality and render time.** Lower pixel variance will be an exponential change in render times.



- Renderman settings
    - Sampling: *Quality*
        - Min samples: 16
        - Max samples: 64
        - Pixel variance: .06
        - Dark Falloff: .025
        - Integrator Pxr Path Tracer
        - Max Depth 6 (can be reduced with no glass in shot)
        - Indirect Samples 1
        - Max specular Depth 1
        - Max Diffuse Depth 1
- Renderman Catmull-Clark subdivision scheme(On Applicable Models)
- Max Specular depth Overrides for Glass
- Relative pixel Variance Optimization
    - Advanced
        - Reetreant procedural: checked
        - Output Holdout Matte: As separate AOV (only if we do holdouts)
        - Water Tight Dicing: Checked (selective to displacements)

- Batch render one frame
    - optimize light samples
    - optimize indirect diffuse

Render setting

so this part of the check list is straightforward that don't really require an explanation but something you have to redo almost every time you rerender.

- Render Using : RenderMan
    - This will usually be on by default unless you delete your pref folder so you will rarely change this setting.
- Render Animation check marked.
    - this will allow you to input the frame range
- Frame Range for scene
    - This will only matter if you batch render in Maya but really to save computer memory you should use command line to start your renders.
    - There is a caveat to using command line renders though. If you were to normally type in a command the frames will render slower and slower because windows will cull threads from the process. How we got around this is we used a python script under Z:\BLINDDATE\02_PreProduction_Maya\scripts on the server. what this script did was it ran a command for each frame in a loop tricking maya batch that we were queuing single frame renders.
- Resolution set to 1080p

- for the foreseeable future we will probably render at this resolution for a while
- while developing the lighting you want ipr in a low resolution and set this to 1080p before you render.

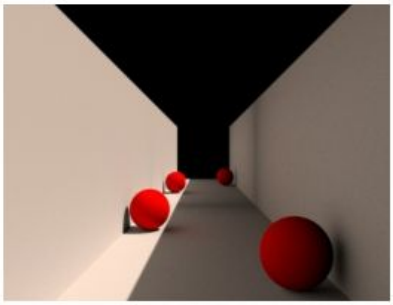
Renderman Settings

- Sampling: *Quality*
  - Min samples: 16
    - pretty straight forward if you want a fast render with low quality set this to 1
    - if you want to set this higher your reason should be that you want to keep your current pixel variance but there are noisy areas that are not being sampled enough and taken away by pixel variance
    - also if you are trying to render something smaller than a pixel like hair, if min samples are set too low then sometimes it will not appear on the frame and register sometimes and you will get flickering if the pixel variance is set too high.
  - Max samples: 64
    - so you will need to find out how many samples your image will need before you are satisfied with the quality and the noise
    - 64 is a good base line but 128 is good for our purposes
    - this will increase or decrease render times in a linear fashion assuming the pixel variance is roughly the same
    - If you can deduce where the noise is coming from and it is only that one part we have methods of cleaning that part up like indirect diffuse or one light that is still noisy. If there are simply too many sources where light is noisy increase this
  - Pixel variance: .06
    - as stated above this will be the first thing you want to optimize for your scene it will vary between environments but most likely not a shot to shot basis
    - with more max samples the lower the pixel variance should be or there will be too many pixels that will be culled before it reaches the max sample ceiling. Vice versa is also applicable. If set too low with low max samples every pixel will be sampled and you will not be using adaptive sampling optimally.

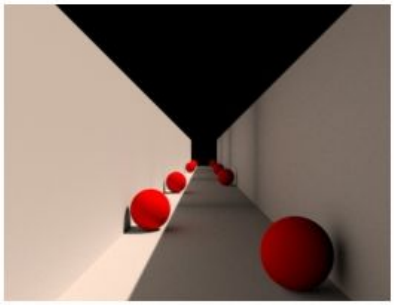
  - Dark Falloff: .025
    - This will de-prioritize lower luminance areas
    - noise is less noticeable in darker areas but if want to tweak those areas in post with color correct>gain the noise will become more apparent.
    - The default value is great but if you know the shot will be brightened up later in post lower this value

- Integrator Pxr Path Tracer
    - The different modes for Renderman, VCM is another alternative. Path Tracer only traces rays from the camera while VCM does it from the camera and the light. This is ofcourse slower but if the majority of your lighting is from a tough indirect source VCM will converge sooner.
    - There are many different Integrators in Renderman the ones you need to know about are VCM and Occlusion.
- Max Depth 6 (can be reduced with no glass in shot)
    - This needs to be 2 over than the highest Max n depth where n is specular or diffuse. If specular depth is 6 this value needs to be an 8.
    - This doesn't really affect render times unless your Specular Depth or Diffuse Depth is higher.
- bxdf Samples 1
    - very rarely we will change this value but if conclude that the noise is coming from the material itself such as subsurface or single scatter we will increase this value.
- Indirect Samples 1
    - At such a low max samples that we are limited by, the indirect passes will not converge at the same time the direct passes will. To solve this we increase this value. I only recommend increasing this to 2 or 3 but this will be incredibly expensive. It will increase the time it takes for our renders by 4 to 10 minutes a frame on the regular 6700k computers.
    - This will sample more rays after this first direct hit and this can really get out of hand exponentially so be careful.
- Max Specular Depth 1
    - When you turn this value to 1 all of your glass materials will most likely turn to black. We will go on a per object basis and turn them to 4-6 where ever it is required. at 4 the overall look will be at 90% of what renderman can achieve with specular experiment with higher values.
    - If you have multiple overlapping glass objects such as liquid in a wine bottle you will need to account every time a ray enters and leaves an object, for example a sheet of glass is 2, a glass cup will be 4 and stagnant liquid in a glass requires 6 at least.
- Max Diffuse Depth 1
    - This is how many times a ray can bounce to contribute to the sample.
    - The highest you should go for film is 2 but if you are rendering an interior for photo realism you should be going upwards to 10.
    - We stuck to 1 in most shots as there is only a slight differences as we increased this value

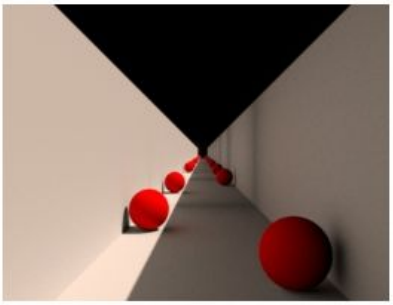
I've provided examples from Pixar's website below.



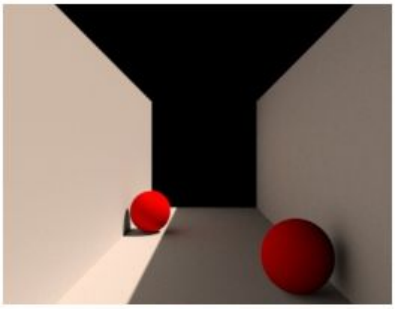
specular depth = 1



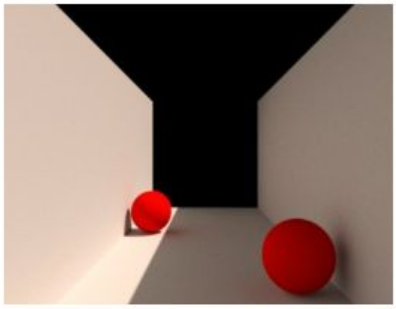
specular depth = 3



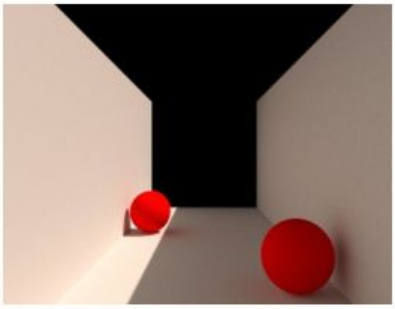
specular depth = 10



diffuse depth = 1



diffuse depth = 3



diffuse depth = 10

Per object optimization