

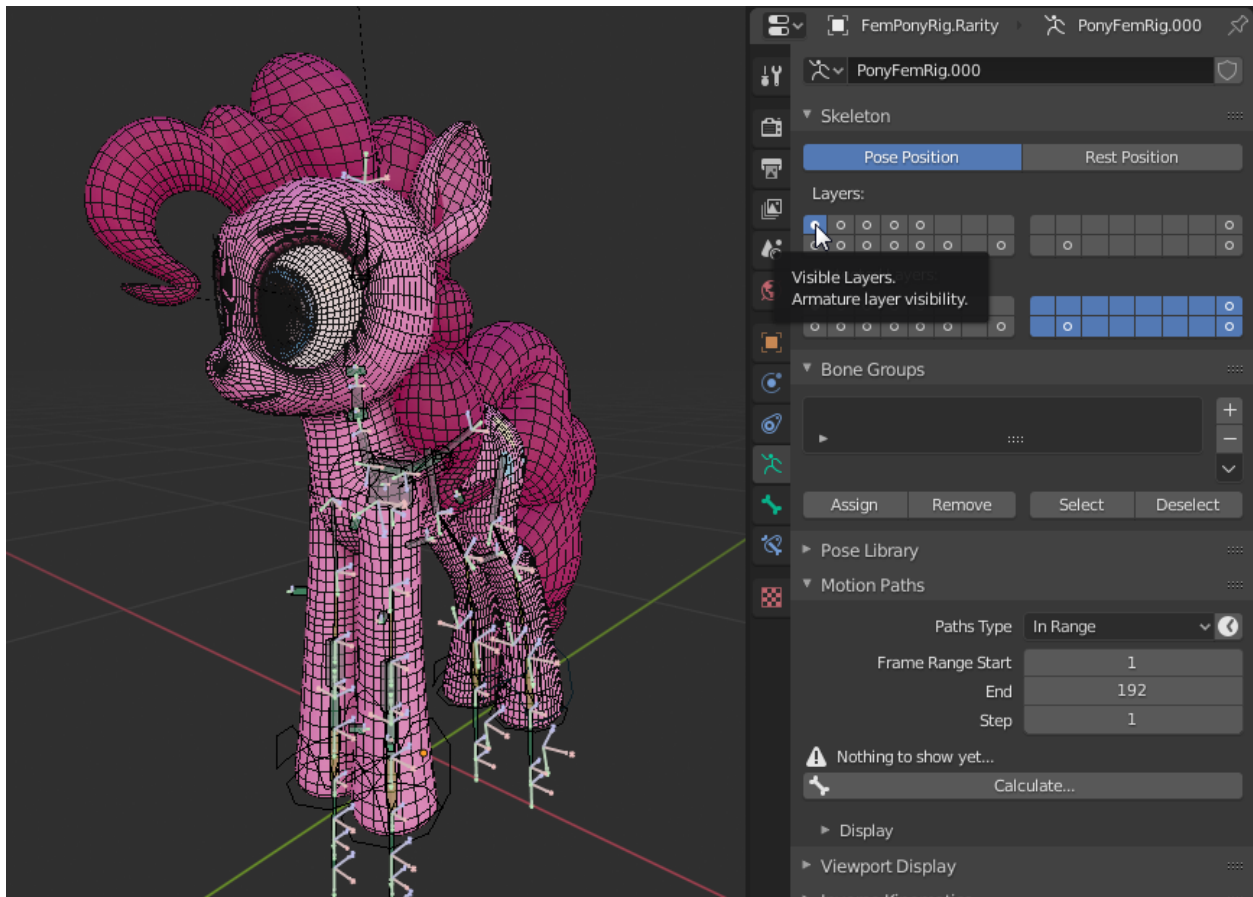
# THIS DOCUMENTATION IS UNDER CONSTRUCTION

So bare with me! (Updated for Public Beta Version 1.0 and for use in Blender 2.81)



While this is still under construction, I'll just be showing some of the basic essential knowledge on how the rig works. I will assume that you have basic Blender knowledge, so I will not be providing basic hotkeys, or showing how to get to common places within Blender and what-not. There are lots of resources out there that can help you get started with Blender (Look for *Blender Guru* on YouTube if you want to get started).

# Bone Layers



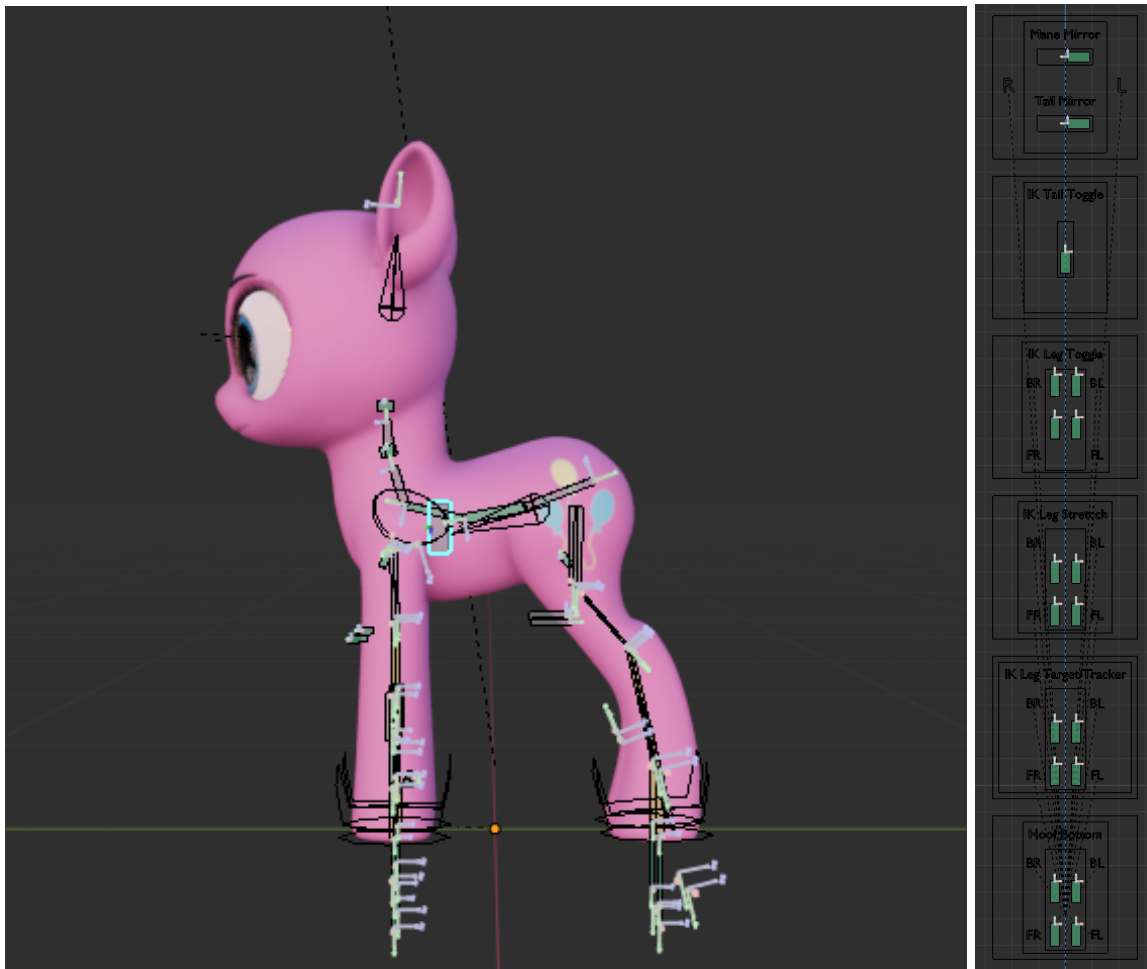
The main thing I want to focus on first before anything else is Bone Layers within the armature of these models. Basically, the bone layers separate the vast amount of deform and controller bones present in the armature. You can toggle the visibility of the layers to show only certain things. The first layer, for example, only shows important main body controllers that controls the legs, spine, and head. And you can hold shift and select more than one bone layer to view more than one bone layer at a time.

Also note that we are only focusing on the bone layers on the left side. The other right side half are for bones that are not really meant to be used when posing. I toss most leftovers, or unposable bones in there. It's not to say they are all useless, as a lot of them are needed for the rig to work. But they don't really need to be thought about when posing the model, only when trying to understand the rig under the hood.

## Bone Layer 1



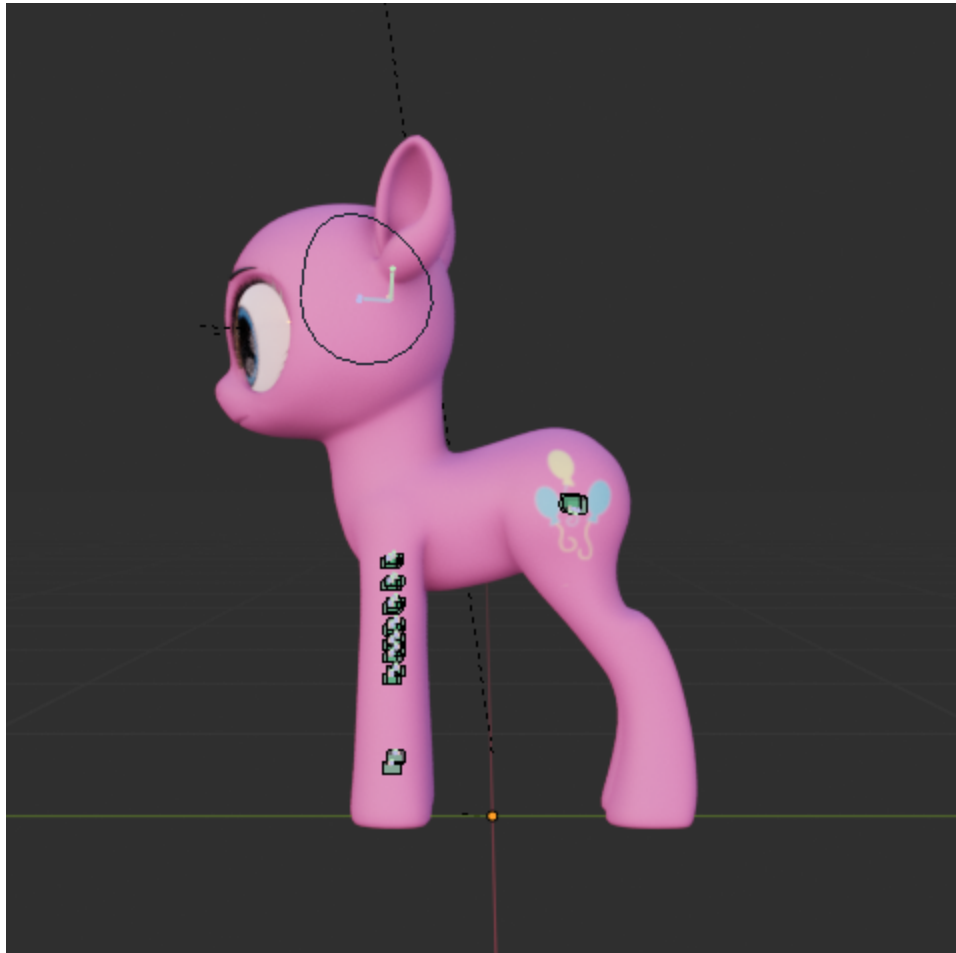
This bone layer contains all relevant bones that handle body posing, and also contains the bone controllers for the levers that hover above the model.



## Bone Layer 2



This bone layer generally shouldn't be used much. It contains all of the twist bones present in the rig along the entire front leg as well as the thighs. The primary deform bone for the head is present here as well, but it's not poseable here. You should use the head controller present in the first Bone Layer. The twist bones can be adjusted if their automated posing is giving you issues, however.



## Bone Layer 3



This bone layer contains primary, mostly shape key driven, bone controllers for the mouth. It contains lip corner posing, lip folding, cheek puffs, jaw and nose, and tongue controllers, among a few other things.



## Bone Layer 4



This bone layer contains primary deform bones for the lips. These bones have no limits on transformation, so you are as free as possible to make the mouth shape you want. Just be cautious about going too far and breaking the model. This layer also contains the same lip corner bones as Layer 3 for the sake of convenience.



## Bone Layer 5



This layer contains the rest of the miscellaneous mouth based controllers. There are deform bones for the tongue (unlike the shape key driven controllers from Layer 3), teeth deform bones, nostril deform bones, etc. Not a commonly used layer, but it's there if you need it.



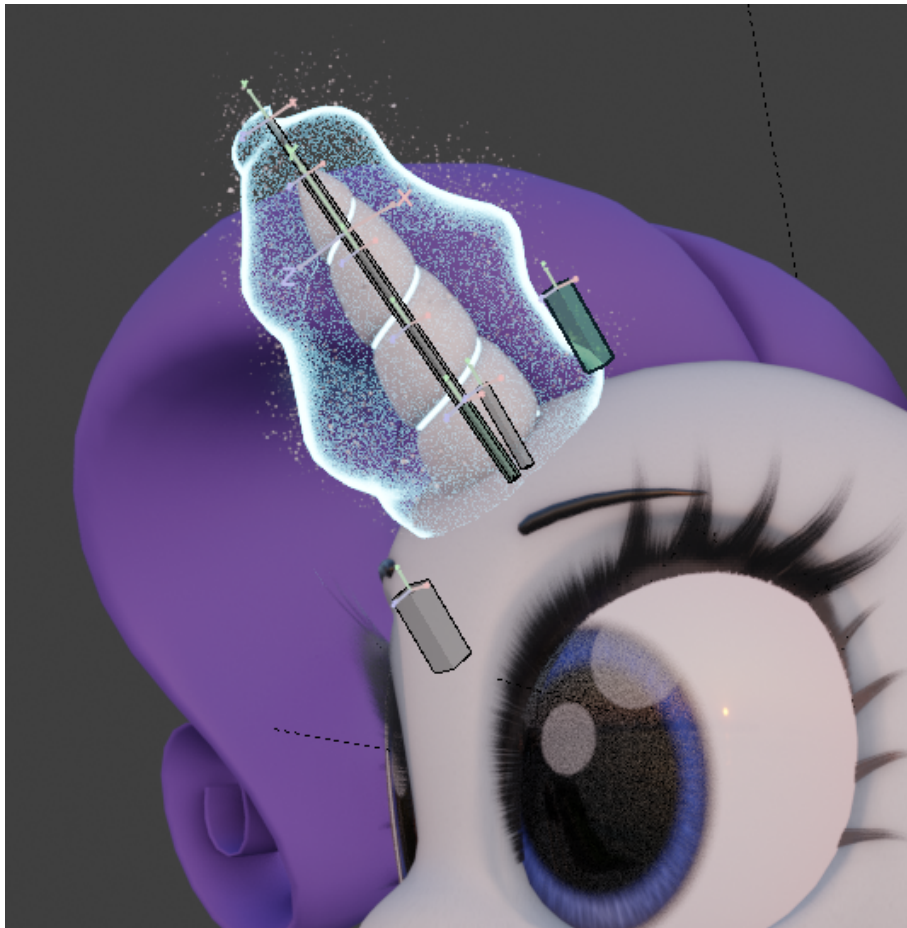
## Bone Layer 6 (Unicorn Only)



This bone layer will only appear in this configuration when you have a Unicorn armature selected. These deform and controller bones control the horn as well as the magic aura rotation, deformation, and brightness.

If you select the Magic Aura object itself (not the armature), and go to the particle settings, you'll see that there's an inactive particle system. You can use this as a template to the Magic Dust that's used to complete the visual effect of the Magic Aura in 3D space. Generally what you want to keep in mind when changing what frames the particle system spawns particles and such, is that you'll need to adjust the amount of total particles based on how long the particles will keep spawning. Generally 300-350 per frame at 24 fps (7800 particles per second) is a decent number to go with.

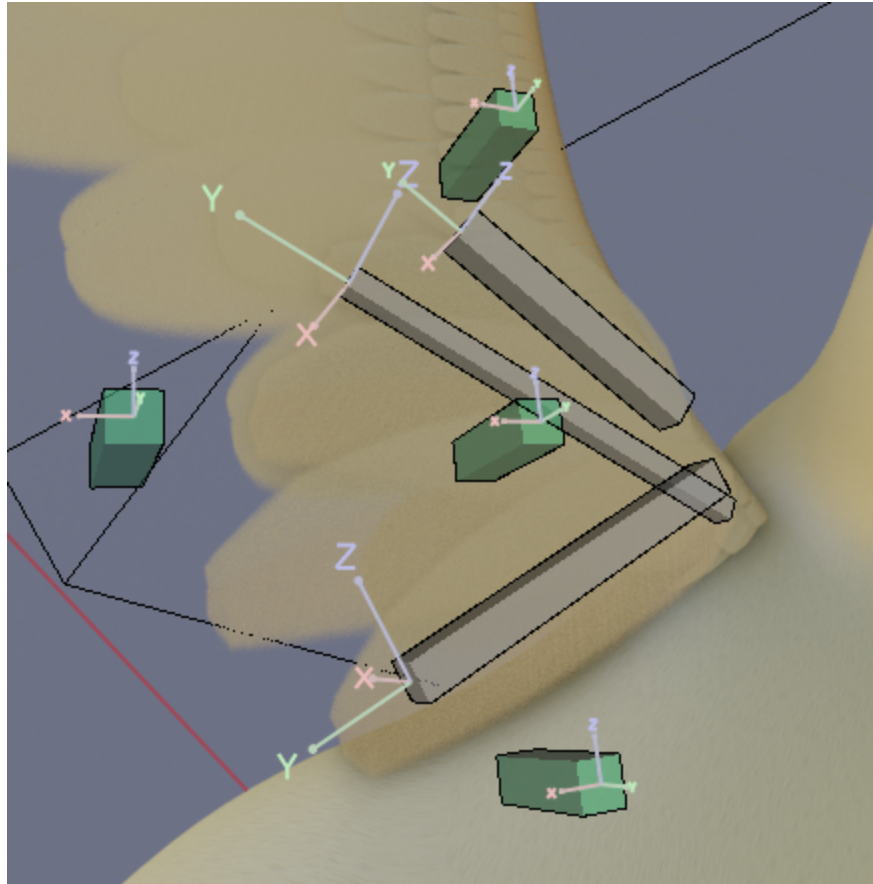
To complete the whole effect, you'll need to make sure to composite in glare streaks to make sure the particles actually flare the lens, and makes the effect feel more real.



## Bone Layer 6 (Pegasus Only)



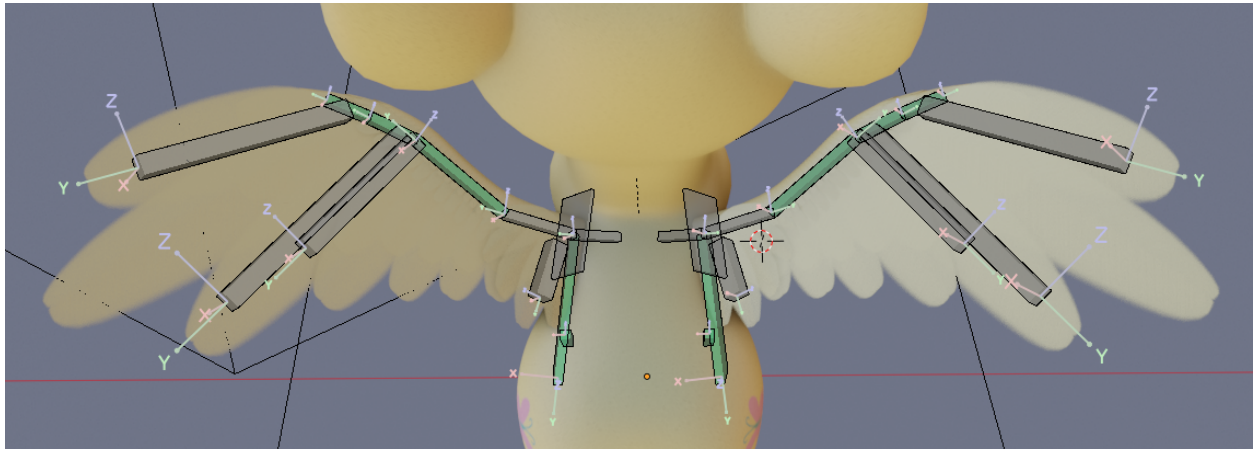
For Pegasus only, there are controllers for configuring the shoulder feathers for the wing.



## Bone Layer 7 (Pegasus Only)



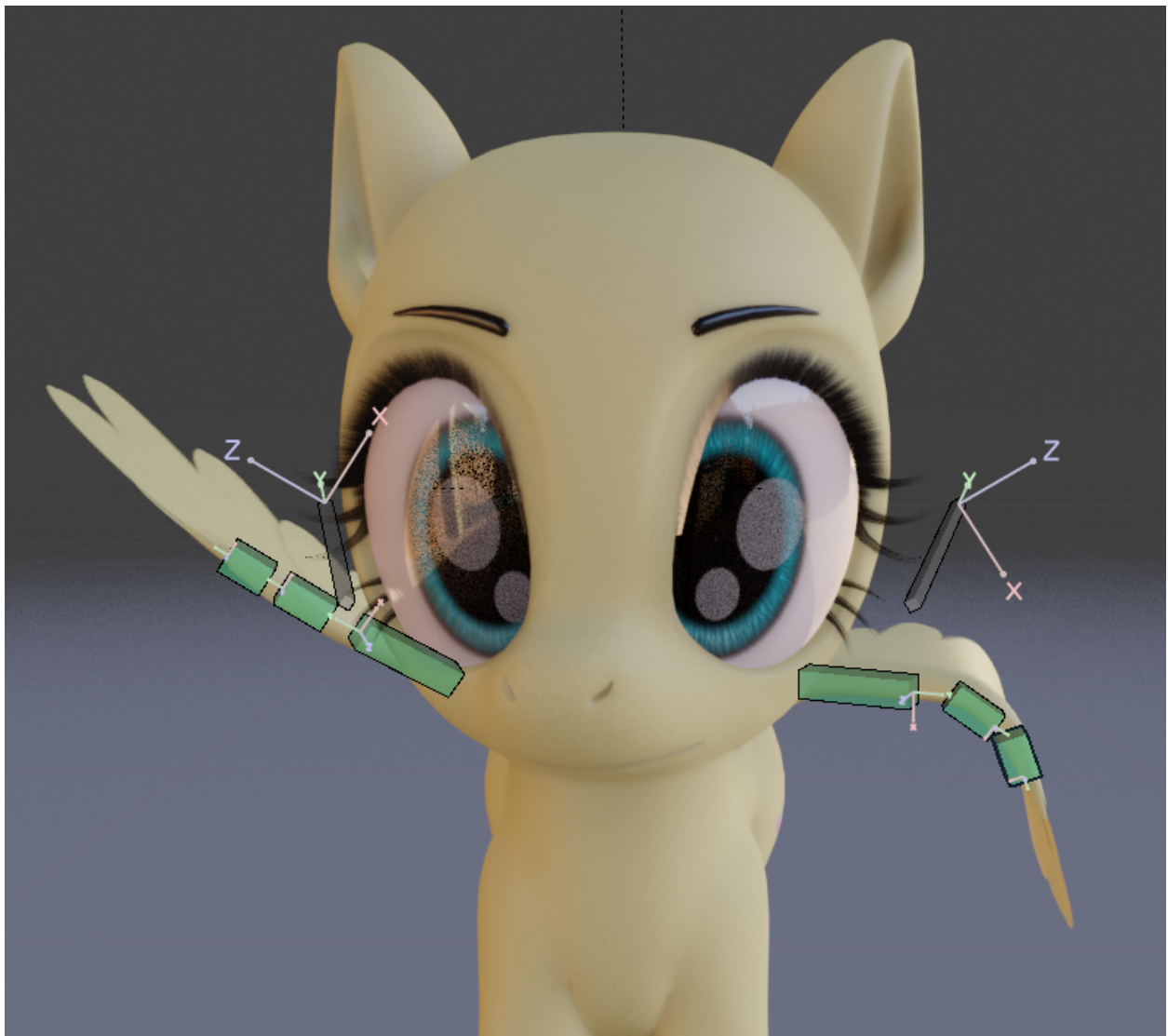
These Pegasus wing controllers control most of the wing posing. Not everything, but most of it. You'll notice some tracking bones at the base that helps keep the wings aligned with the flank, which especially helps when the wings are folded up. The feather bones you see in this layer influence all of the other feather bones in some way that are hidden away in a different Bone Layer. And of course, the wing arms themselves are a bit limited in rotation. Due to probably my own ineptness on how to configure constraints and drivers for bones that need to rotate on more than one axis, I've delegated the local X-axis rotation to the wing arm bones in this layer, while everything else with the wing arms are handled in the next layer, Bone Layer 8.



## Bone Layer 8 (Pegasus Only)



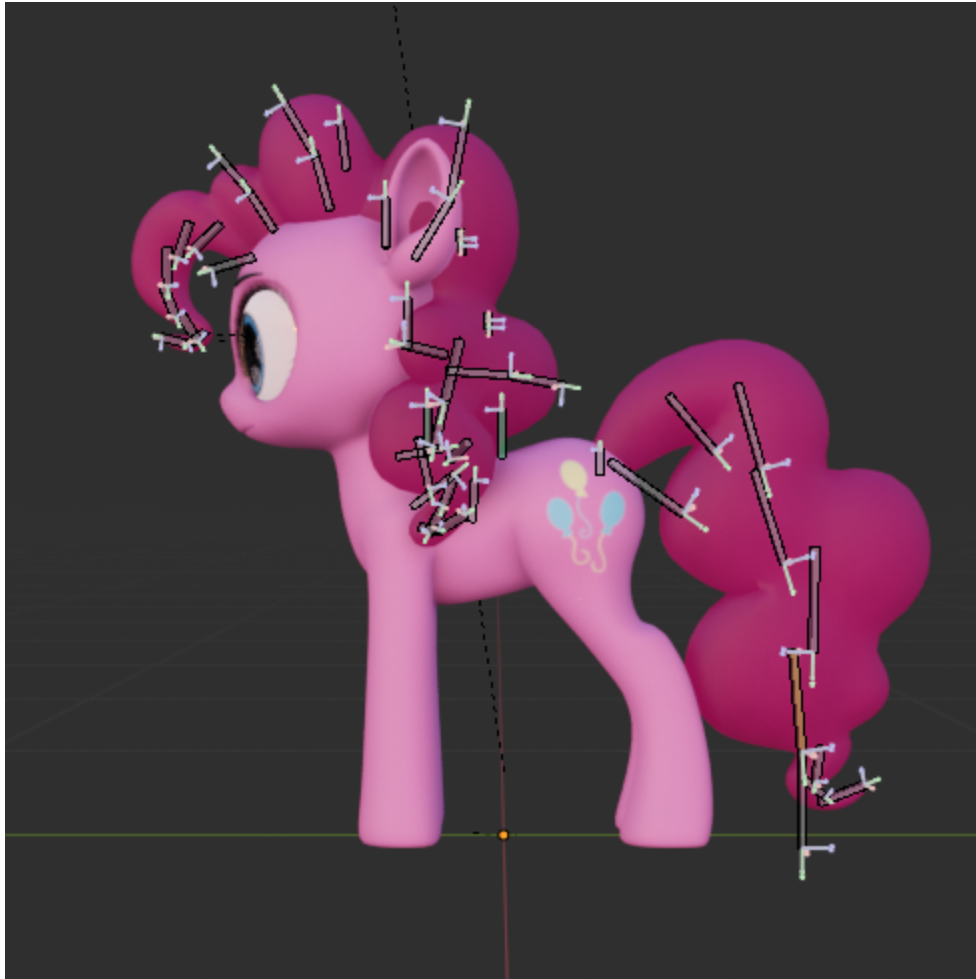
These bones handle all other axis of rotation for the wing arms. They handle up and down motion (local Z-Axis rotation), as well as twist motion (local Y-Axis rotation). There's also a lattice deform bone for the feathers that allow the feathers to rotate up and down. This may need to be developed further, as it's a bit limiting. The wings as a whole is without a doubt the most complicated, and most taxing part of this entire rig. But it isn't perfect. If anyone has any feedback regarding these wings, which wouldn't surprise me, please use the feedback google form (found at the end of this documentation)!



## Bone Layer 9



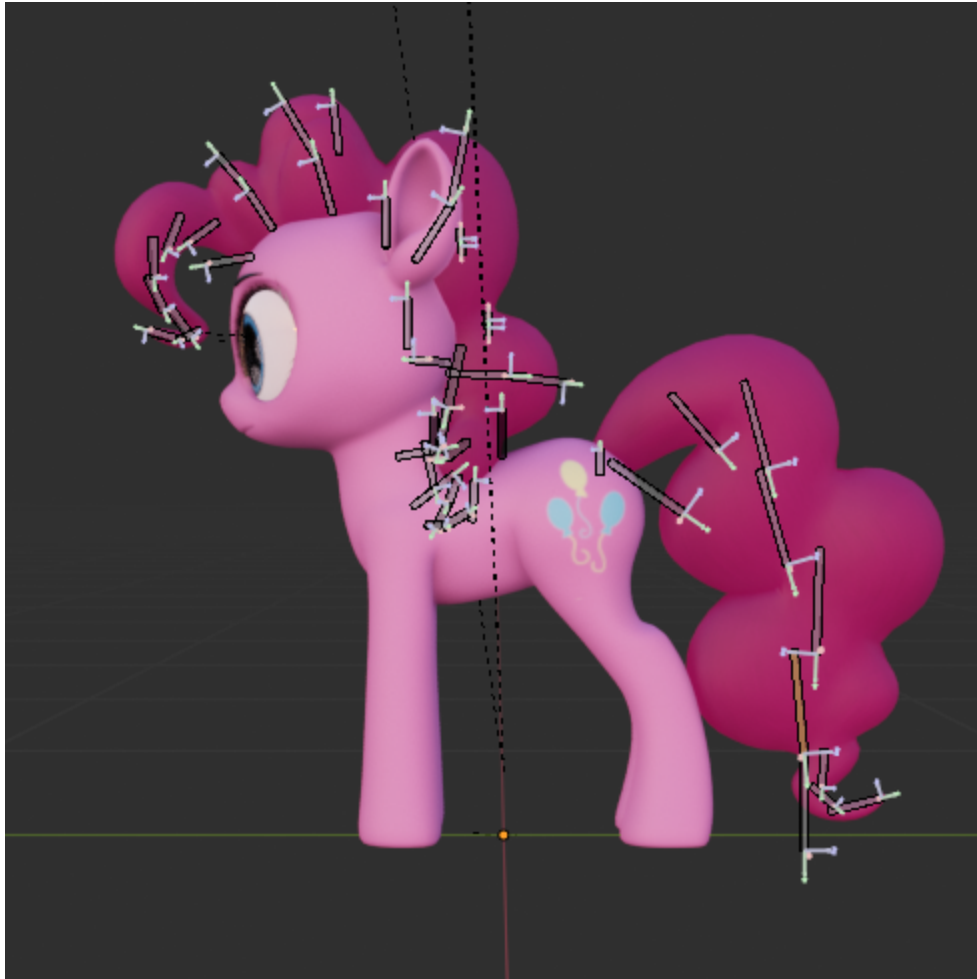
This layer contains the hair deform and controller bones for the mane and tail of any of the characters.



## Bone Layer 10



This layer contains the *flipped* variant of the hair deform and controller bones. You will need to make sure you flip the hair mirror lever to make the mirrored version of the hair mesh visible. (also all mane and tail related levers above the model are present in Layer 9 and 10)



## Bone Layer 11



This layer contains all things that heavily affect the eyelids. The eyelid blink controller is here, as well as a few others that shape the eyelid region, and others like cheek pushes.



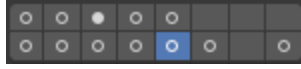
## Bone Layer 12



These bones control all things to do with the eyeball. So that includes Iris and Pupil scale; shape key driven by the way. As well as other controllers that control eyeshine brightness, whether or not the eyeshine is mirrored, as well as the positioning of the eyeshine. Eyelash deform bones also rest in this layer.



## Bone Layer 13



These are eyebrow controller bones. This is probably the most complicated part of the rig under the hood, but the complication under the hood was intended to make posing simple, hence the low controller bone count. There's also a controller bone that restores the original eyebrow length my models use to have.



## Bone Layer 14

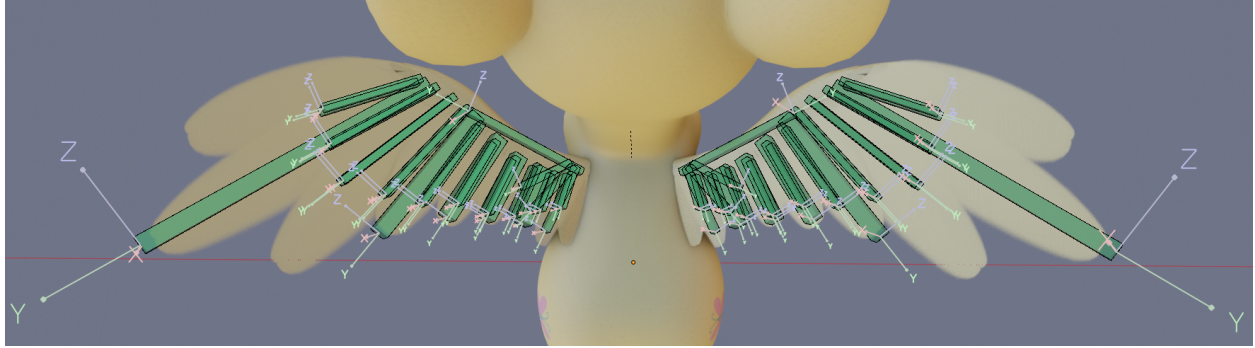


The ear deform and controller bones are here. Definitely don't want to neglect these when you pose the model. Recently I've added support for twist bones at the base of the ear so that the ear can once again have a wide base while still having good deformation when the ears rotate around the head.



## Bone Layer 15 (Pegasus Only)

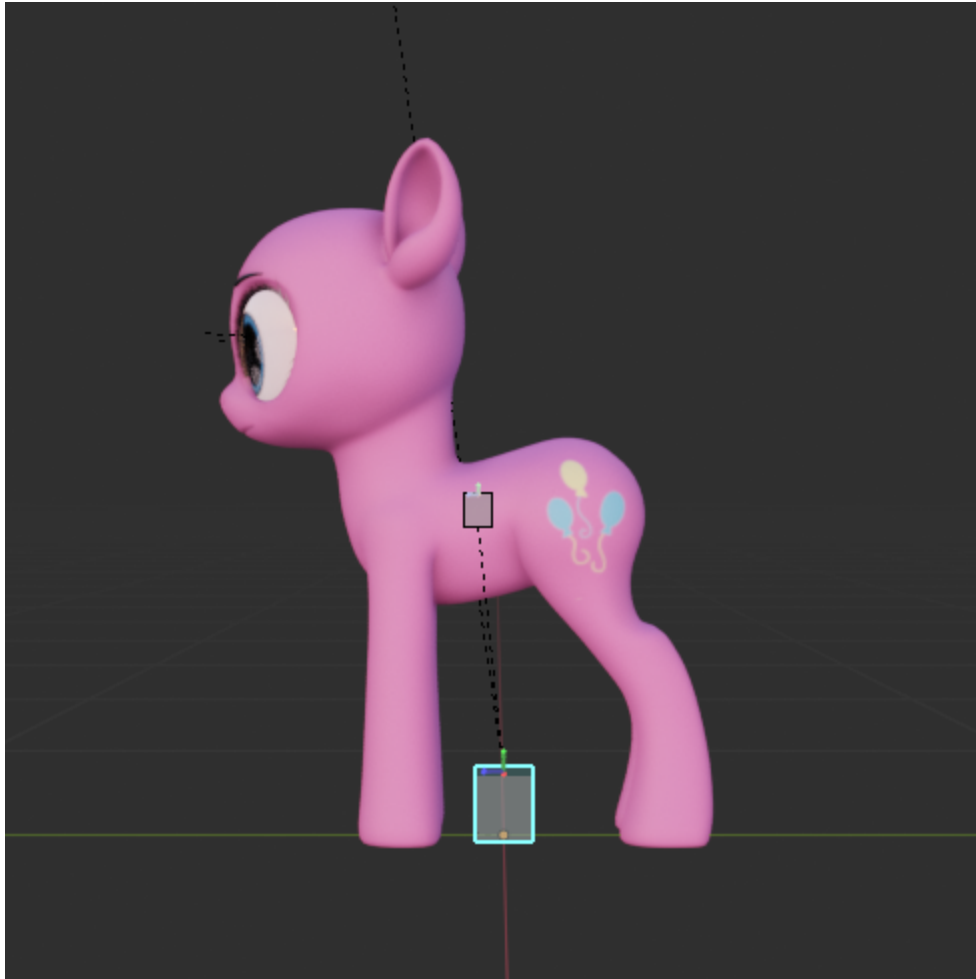
These pegasus feather bones are bones you normally wouldn't need to adjust, as they are automated based on the primary feather controller bones from Bone Layer 7. But if some feathers are causing issues, like clipping, this is where you'd adjust individual feathers.



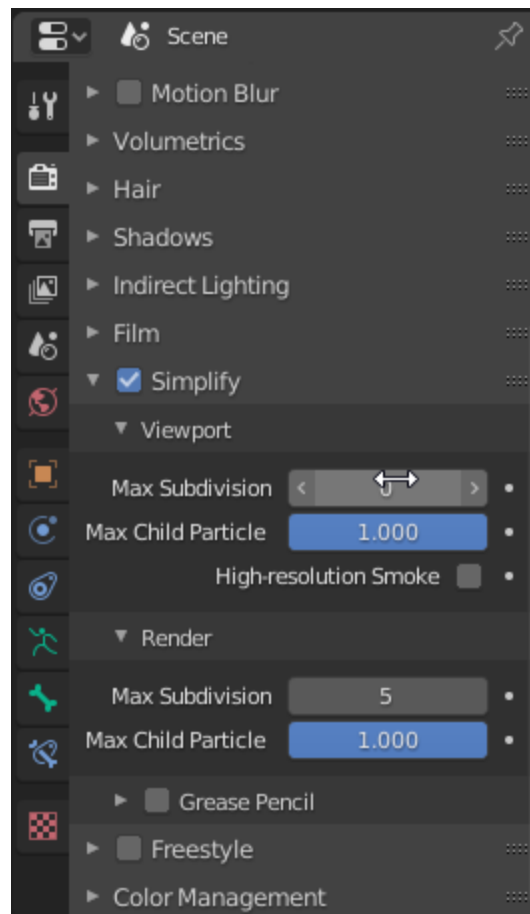
## Bone Layer 16



Nothing too special here. Just the origin bone, as well as a main bone around the center of mass. These bones are the highest on the bone hierarchy chain overall.

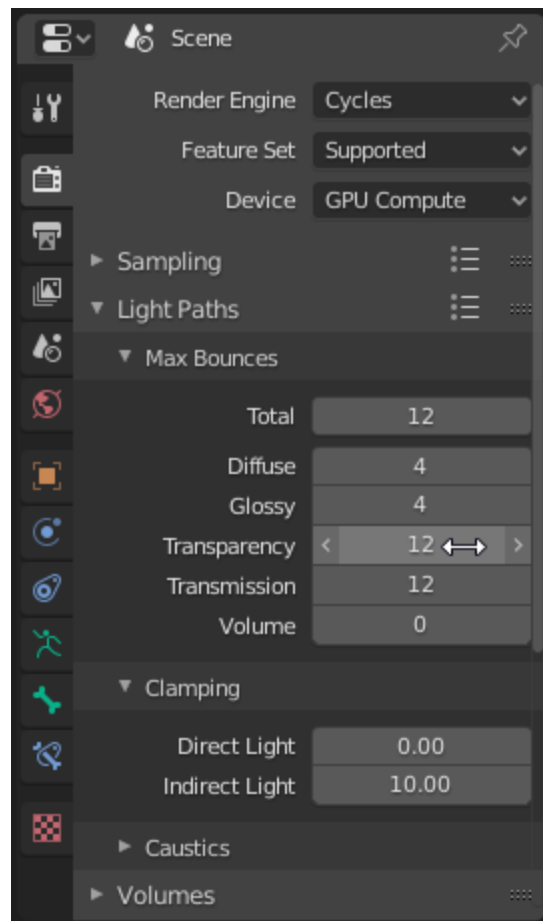


# Simplify Configuration



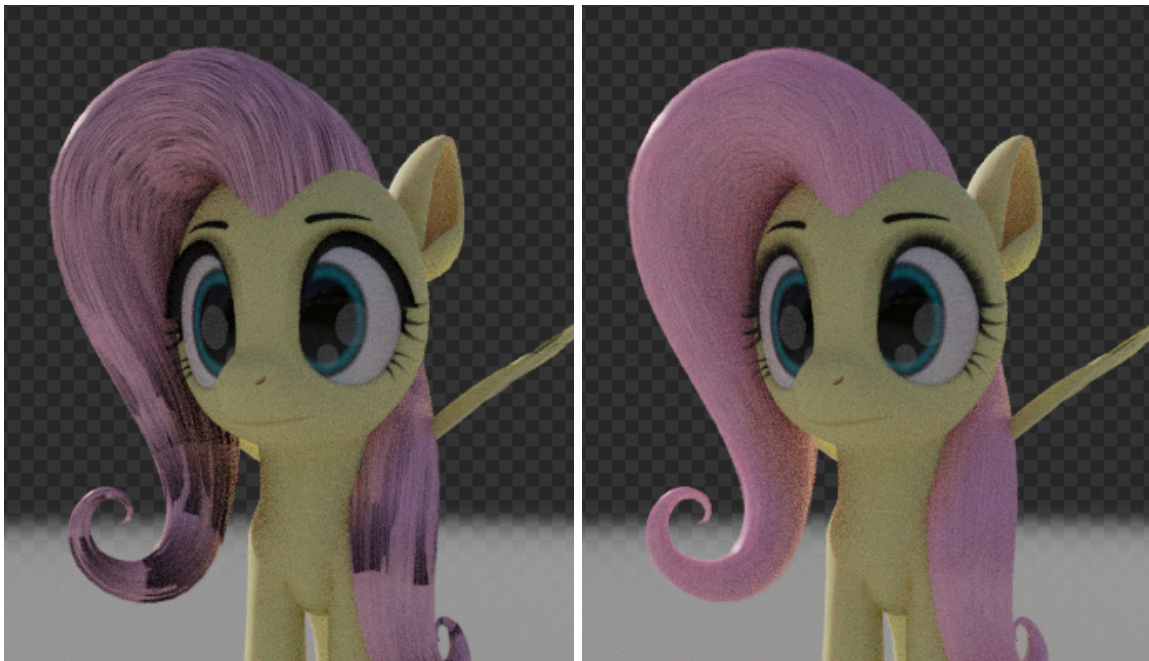
I highly advise using this Blender feature when using my models. My models are incredibly taxing, mostly due to the layered hair meshes used to create the nice looking hair effect, but other factors like subdivision and posing while using Eevee rendering can make posing the model a bit of a chore as the performance of Blender slugs along. If you want to get things done quickly, use the simplify settings above that disable subdivision in the viewport. It will make the models look incredibly jagged, but it will immensely help you if you need better performance out of the viewport.

# Light Paths Configuration

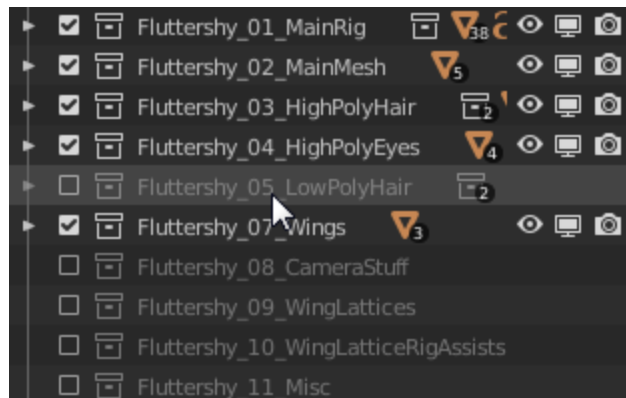


Whenever you make a new scene, and the models start to render in weird ways, the first thing you should look into is the light path configuration. It is absolutely required that you have the Transparency setting set to at least **12 max bounces**. Otherwise, the hair on the model will start giving very visibly unappealing artifacts.

As you can see below, the Fluttershy mane on the left is not ideal. That's because the scene is currently set to only have a maximum of **2** transparency bounces. The one on the right is set to **12**.

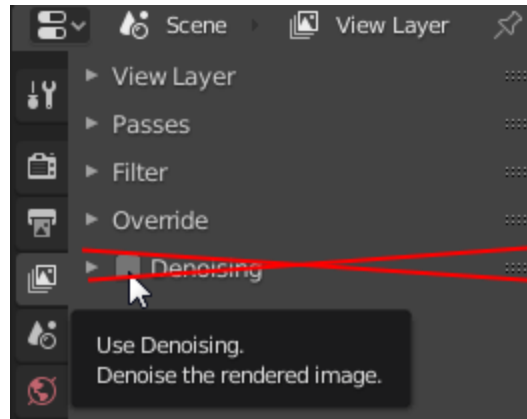


## High Poly Hair and Low Poly Hair



Generally, whenever you want to make a final render, you always want to have the High Poly Hair collection enabled and set to render, while having Low Poly Hair disabled for render. But for the sake of viewport performance, you may want to disable the High Poly Hair collection outright and use the Low Poly Hair collection instead. The Low Poly hair doesn't have hair layers, it's just one single mesh with no transparency. They are not meant to be used for final renders of any kind, as material configuration and colors may be off. But they should be suitable for viewport purposes. (Wings can also be very taxing on viewport performance. Disable that collection temporarily if you need to as well).

# Cycles Denoising



Given how complicated these models are to render in Cycles, there will be lots of noise even in simple scenes. It would take ages to render a noiseless image without any form of denoising. So this is absolutely essential for people who want to render in Cycles, especially if it's an animation.

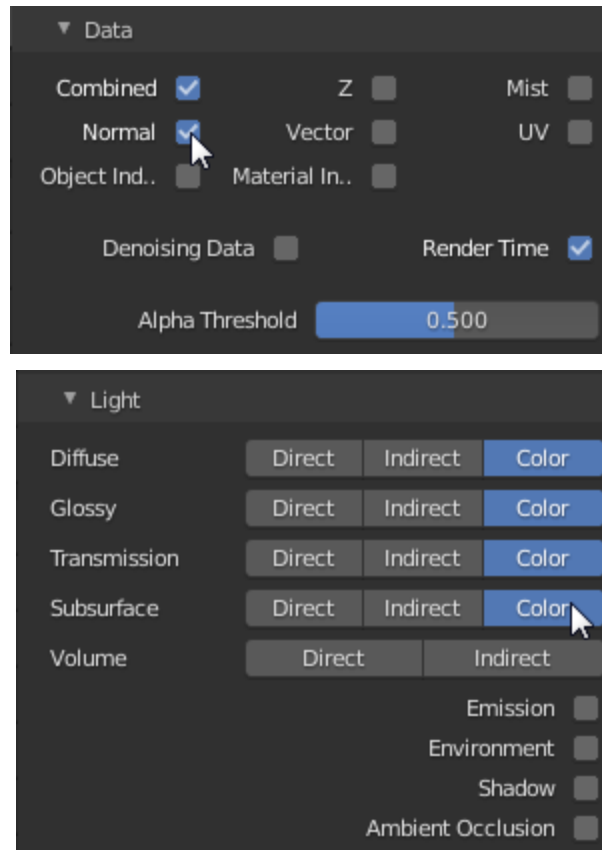
There's a specific method I've been using to denoise my models for not too long now. In Blender 2.81, they introduced a new way to denoise via the compositor, which is way better than the old denoise option seen in the View Layer tab in your properties window. So... **DO NOT USE THE OLD DENOISER**. It's inferior.

What you want to use instead is the Denoise node found in the compositor, which uses a completely different algorithm developed by Intel.

Now there are tutorials out there on how to use this thing, but how I use it is going to be a bit different, because the traditional setup for this node will not work very well due to the abundance of transparency found in the hair.

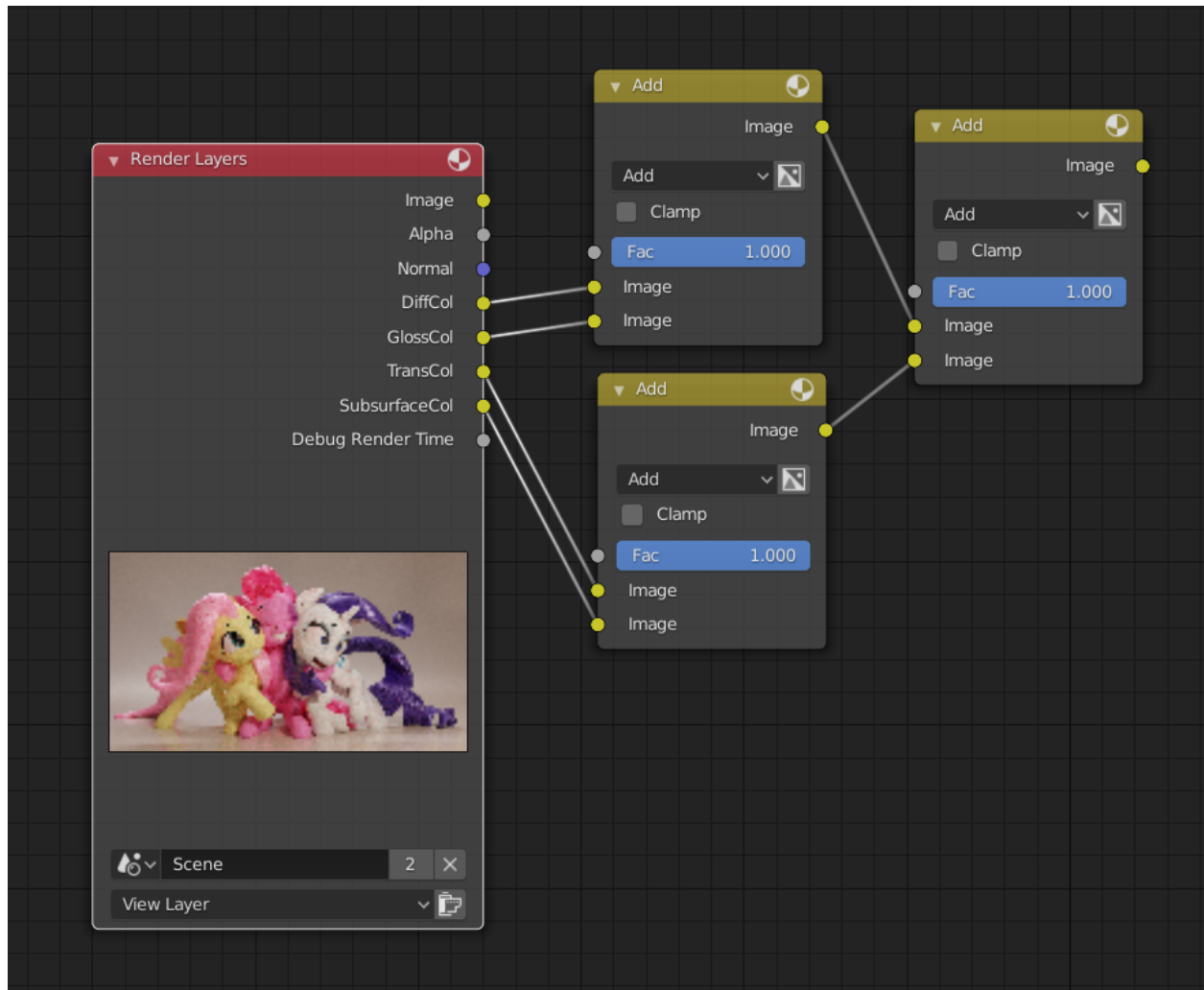
Before we go to the compositor, in your View Layer tab in the properties window, make sure you have these passes enabled:

- Normal
- Diffuse Color
- Glossy Color
- Transmission Color
- Subsurface Color



We'll be using all of these passes for the denoise node in the compositor. Traditionally, all we'd enable is *Denoising Data*, but all of the passes associated with that have proved to be problematic for my models specifically due to how they handle transparency. So we'll be using the default Normal pass instead, and we'll be constructing our own Albedo pass by adding together all of the plain color passes in the scene.

So, in the compositor, first we just add all of the color passes together to create our albedo pass that'll be used for denoising.

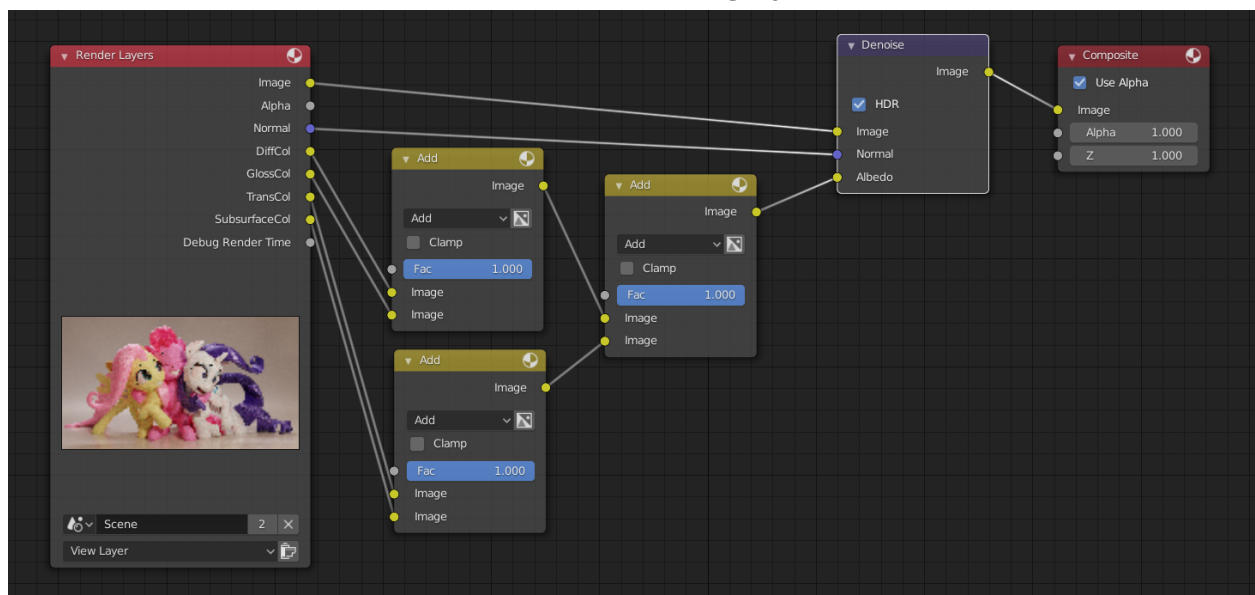


And our albedo pass for this scene will look like this when combined:



Don't worry about how washed up it looks. It's raw color data. But due to Filmic's color management affecting how we're seeing the albedo, and the fact that we're literally adding everything together which "overexposes" the image, it will still work perfectly fine as an albedo pass.

Now we add our **Denoise node** found in the **Filter** category.



And this is our end result!

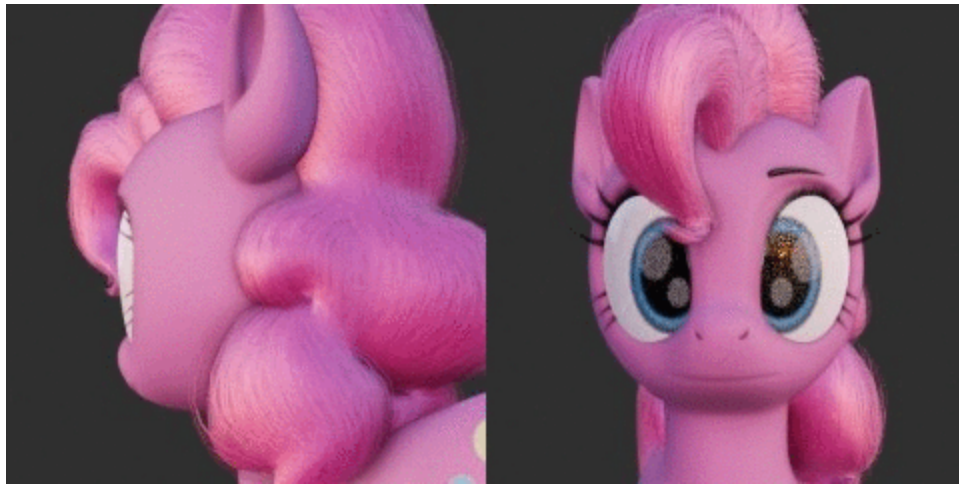
No Denoising:



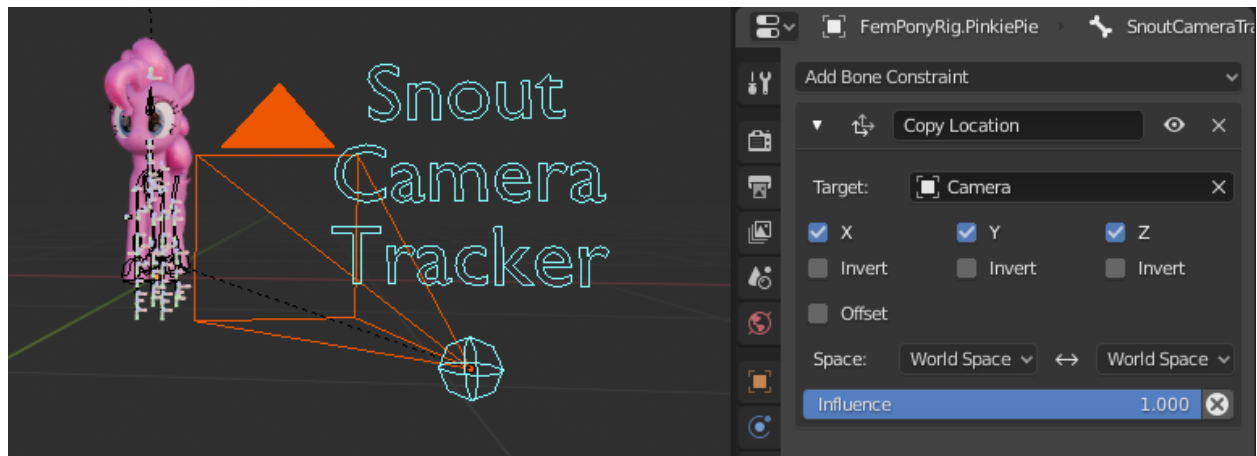
With Denoising:



# Snout Camera Tracker



This is a pretty important aspect of my models. There is a bone constraint setup that basically adjusts the shape of the snout depending on where the camera is. But in order for this to work, you need the Snout Camera Tracker Controller Bone, found in Bone Layer 1, to actually copy the location of whatever camera is currently active. Be very aware of what this tracker is doing. Your render can look really, *really* off if you don't keep track of this bone and make sure it's always copying the exact location the active camera is doing, especially when you have a scene when the camera changes during animation.



# Download

Google Drive Folder for my Models can be found here:

<https://drive.google.com/drive/folders/1285RzgZBvtUV43-ccJh0luJl7seBCq30>

# Feedback

If you ever encounter any issues with the model, like bad deformations or bones doing things they look like they shouldn't be, or issues with rendering or shading, or anything regarding the model itself and the rigging, please let me know! If you have any critiques or suggestions (for the model or even this documentation that I made), please let me know them as well! I know that this model might be a bit strange to use for people that aren't me for a while, and I'm relying on feedback to hopefully make a rig like this easier to use, and more flexible to use, in the coming months.

To give me feedback, you can use this Google Form. I'm going to be checking up on the responses to this form constantly, so please use this if you have any feedback to provide!:

<https://docs.google.com/forms/d/e/1FAIpQLScbPiFUOoKHdQhc-OG27FwB318B7Ekn4JRumlssBDdrN9kwHA/viewform>

Alternately, if you are an active \$5 or above Patron from my Patreon (linked below), you'll have access to my Discord Server, where I will also accept feedback there as well.

# Patreon

If you want to support me financially, you can help me by supporting my Patreon! On top of that, I might consider releasing more experimental versions of my models there, or small iterative releases before I make another whole new version release later for the general public to use:

<https://www.patreon.com/djthed>

# Changelog

- Public Beta v1.0
  - Initial Public Beta Release, which contains Pinkie Pie, Fluttershy, and Rarity.
- Public Beta v1.0.1
  - Fixed an issue with iris discoloration due to a missing texture that wasn't packed when it should've been.
  - Cleared world lighting data for Rarity and Fluttershy (forgot to do this before).