Chapter 2 - Particles

This page contains chapter 2 of the Houdini to 3ds Max Alembic Workflow tutorial, covering the export of Particles.

Overview

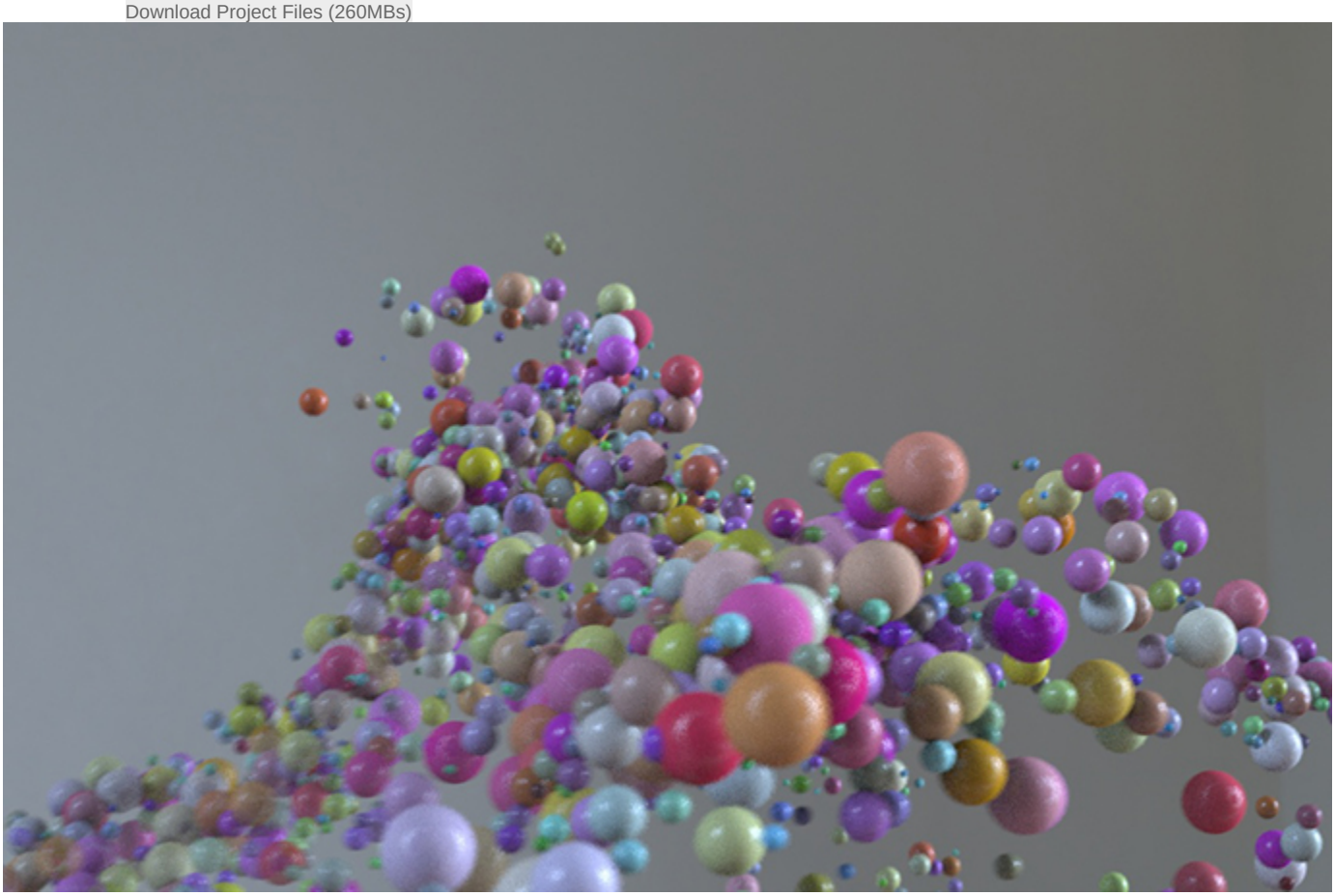
As in the previous chapter, the Houdini setup will not be discussed in detail. You can check the provided **chapter_04_particles.hip** file for reference, which you can download from the button below.

A few notes when exporting particles from Houdini as Alembics to be loaded through the **V-Ray Proxy Import**:

1. The equivalent of the **pscale** attribute is **width**. You can remap it in a Wrangle node with **f@width = f@pscale;**
2. The **v** attribute is automatically remapped and read by V-Ray when **motion blur** is enabled. You should be aware, however, that **v is not used when the topology/point count does not change** - in this case V-Ray calculates the velocity (and thus - the motion blur) based on the transformations /deformations of the points.
3. All **custom attributes** are preserved and can be accessed through the **Vertex Color Texture** [1]. You need to feed it to a material, apply that material to the V-Ray Proxy and press the **Update button** on the texture to refresh the **Channel Name** drop-down menu on the **Vertex Color** texture.

You can download the project files from here:

Download Project Files (260MBs)

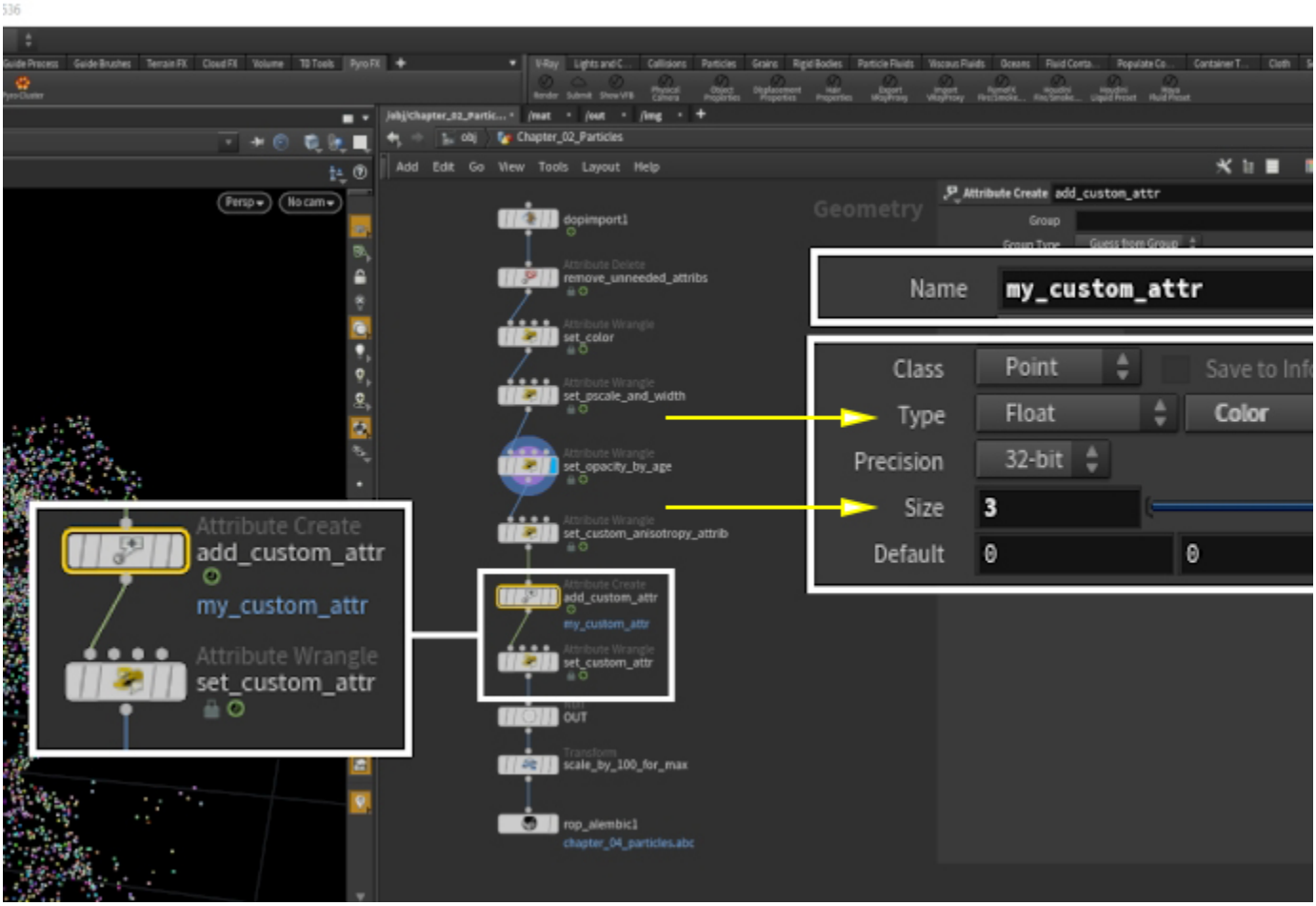


Create and Export Custom Attributes

Assuming you already have a particle simulation in your scene and you have brought it back to the SOP level using a DOP Import or a similar method, let's see how to create custom attributes and successfully carry them over to 3ds Max for use in a V-Ray Material.

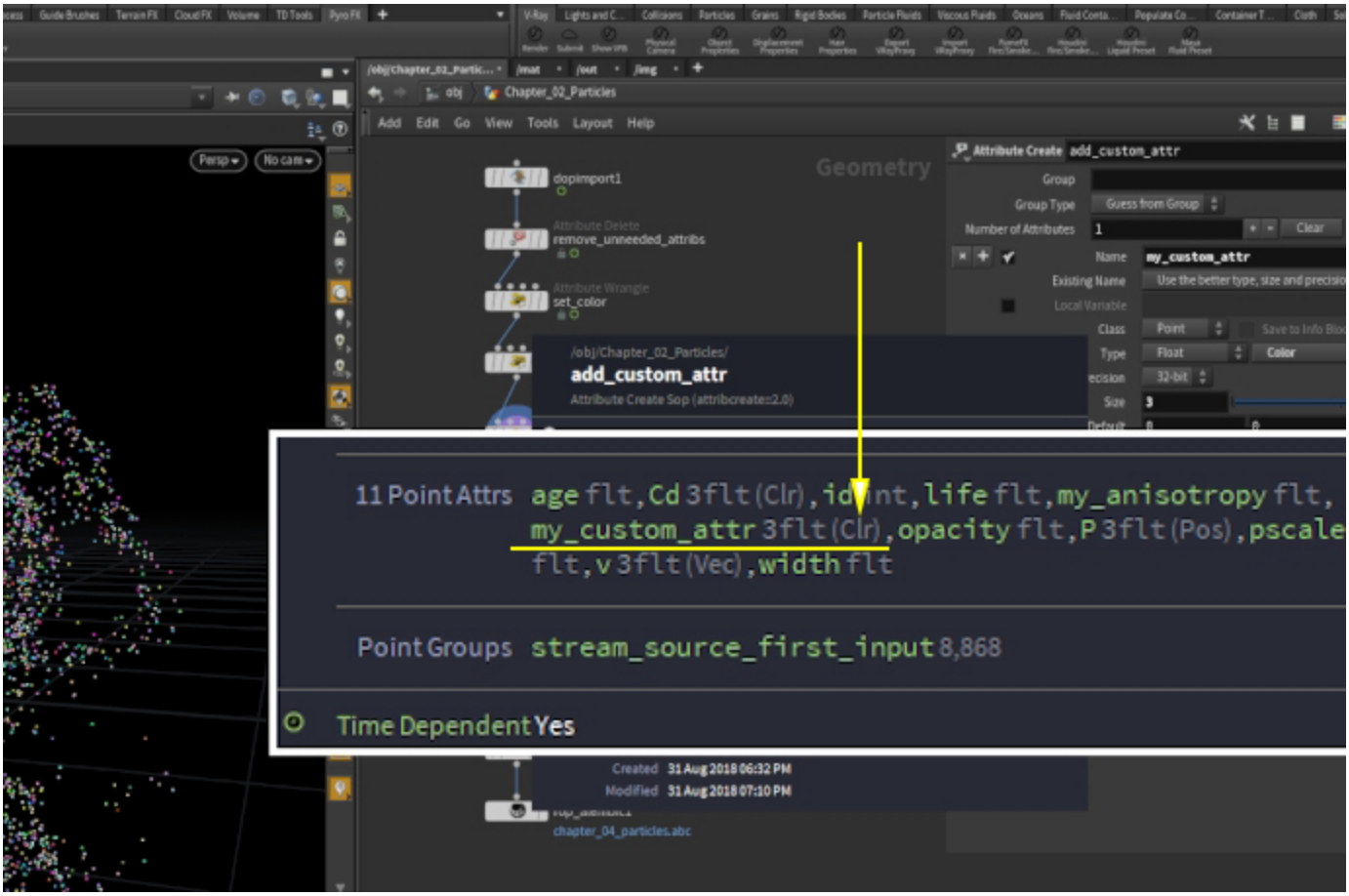
Drop an **Attribute Create** node at the end of your chain and create a **my_custom_attr** attribute.

IMPORTANT: set the **Type** to **Float Color** or **Float Vector** , and set the **Size** to **3**.



You can verify that the attribute type is correctly set up as **3flt (Clr)** by middle-mouse-button click on the Attribute Create node.

The attribute should now be listed as **my_custom_attr 3flt (Clr)**.

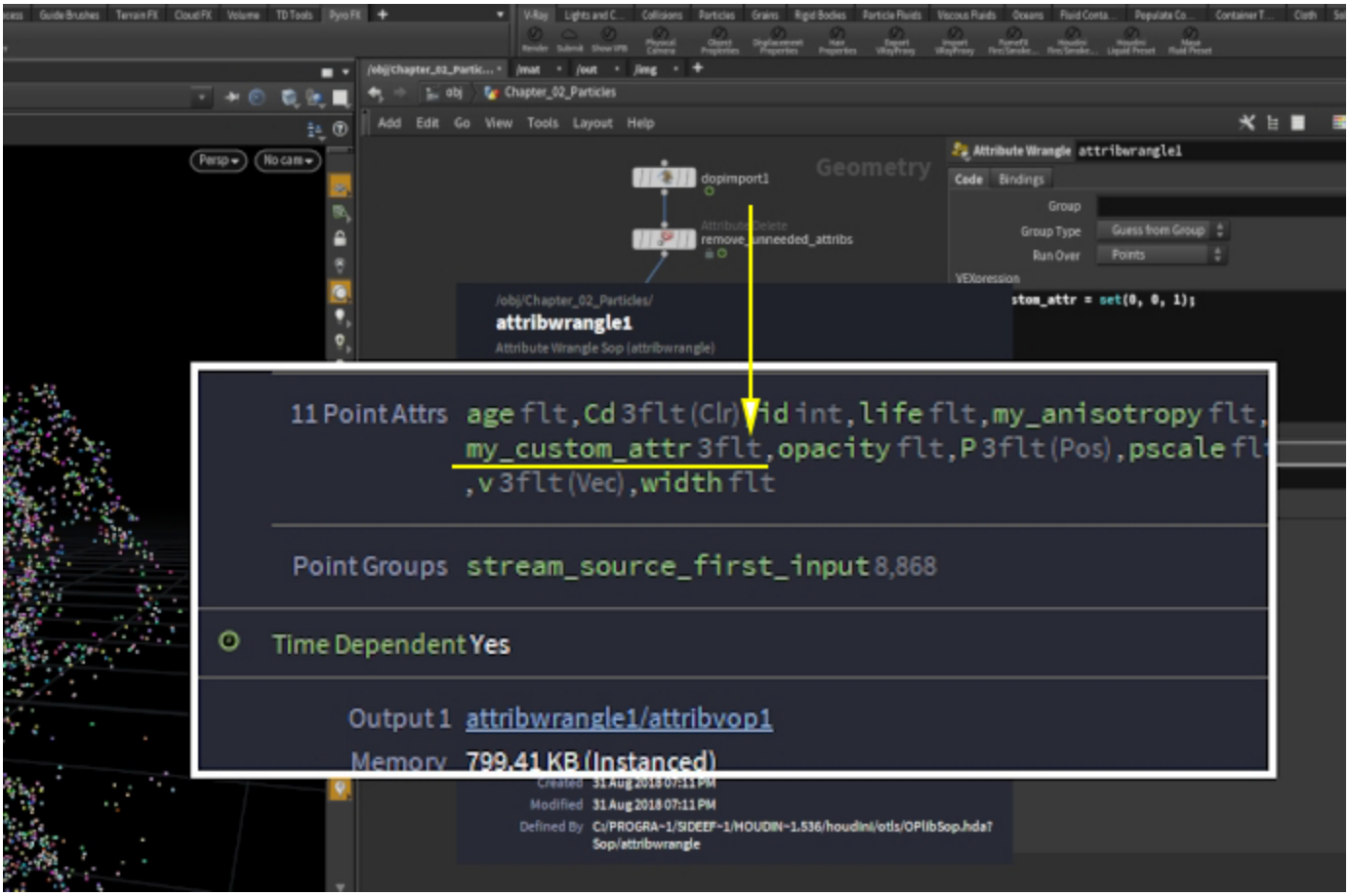


The image to the right shows the result of **v@my_custom_attr = set(0, 0, 1)** inside an **Attribute Wrangle** node.

You can see that **my_custom_attr 3flt** is displayed instead.

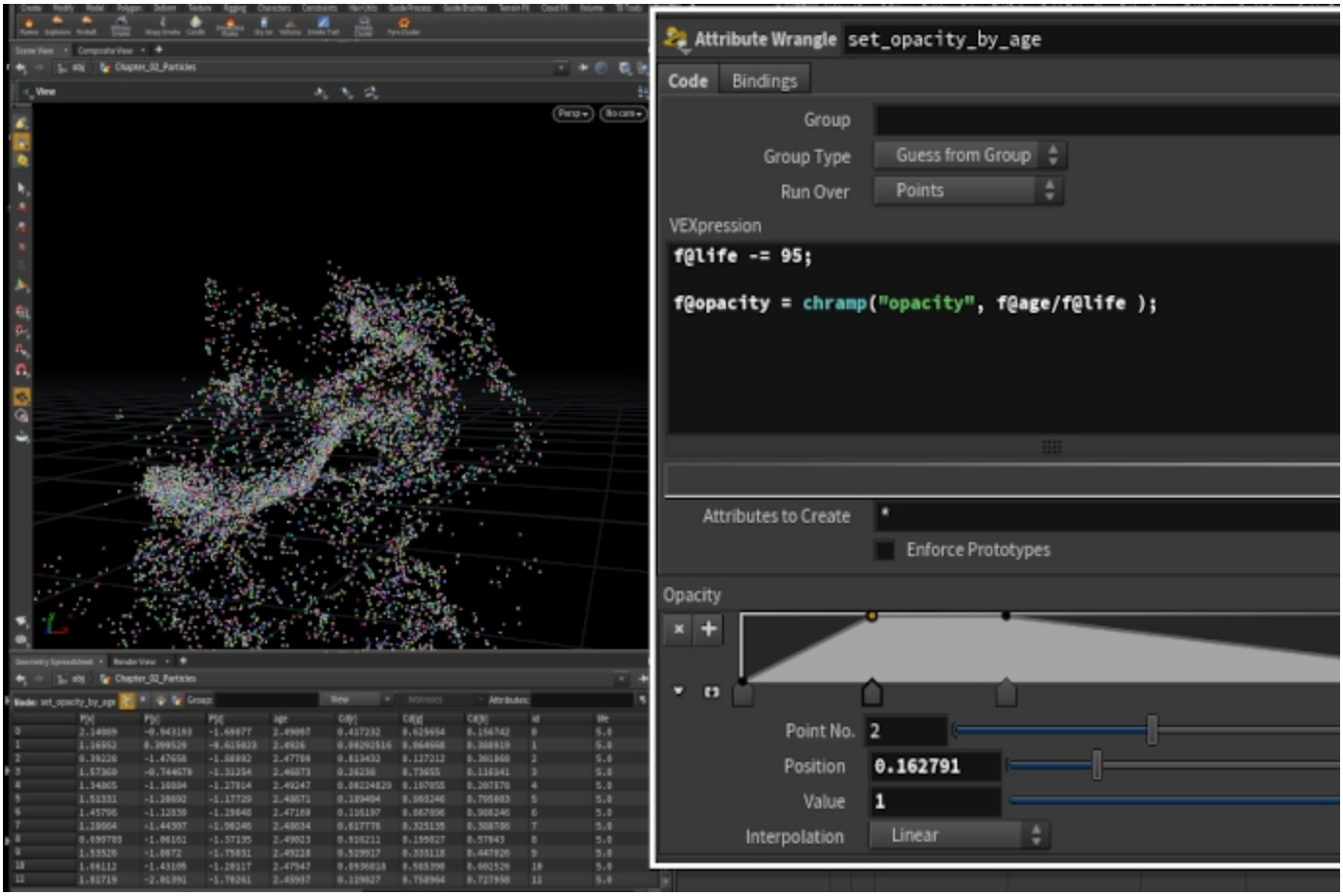
To recap: creating the attribute directly inside the Wrangle node will not work! This is successfully written inside the Alembic, the V-Ray Proxy in 3ds Max also successfully imports it, even the Vertex Color texture displays it as an option for you. If you plug it and try to render, however, the output is black.

Generate your attributes with an Attribute Create SOP and only use the Wrangles to tweak them.



The rest of the setup is **Attribute Wrangle** nodes used to set up **custom float attributes**. You don't need to go to the Attribute Create node for floats.

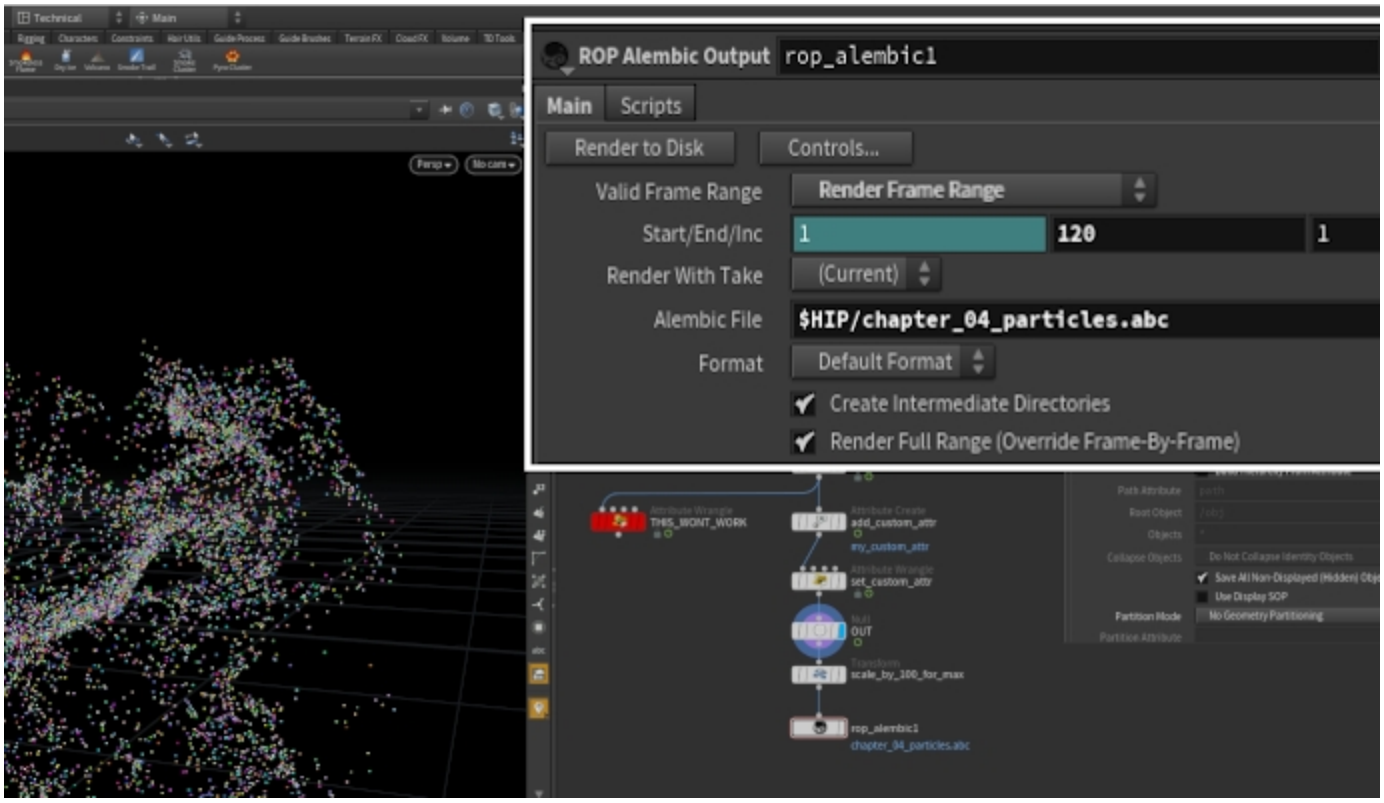
In the picture, a custom **opacity** attribute is mapped to a **ramp** whose **input** is the **normalized age** of the particle.



The particles are **exported** to **Alembic** using the **ROP Alembic Output** node.

The **Range** is set to frames **1 to 120**.

A **Transform SOP** is placed in the chain right before the Alembic ROP with the **Uniform Scale** set to **100**. This is done to compensate for the difference between the default units of Houdini and 3ds Max.

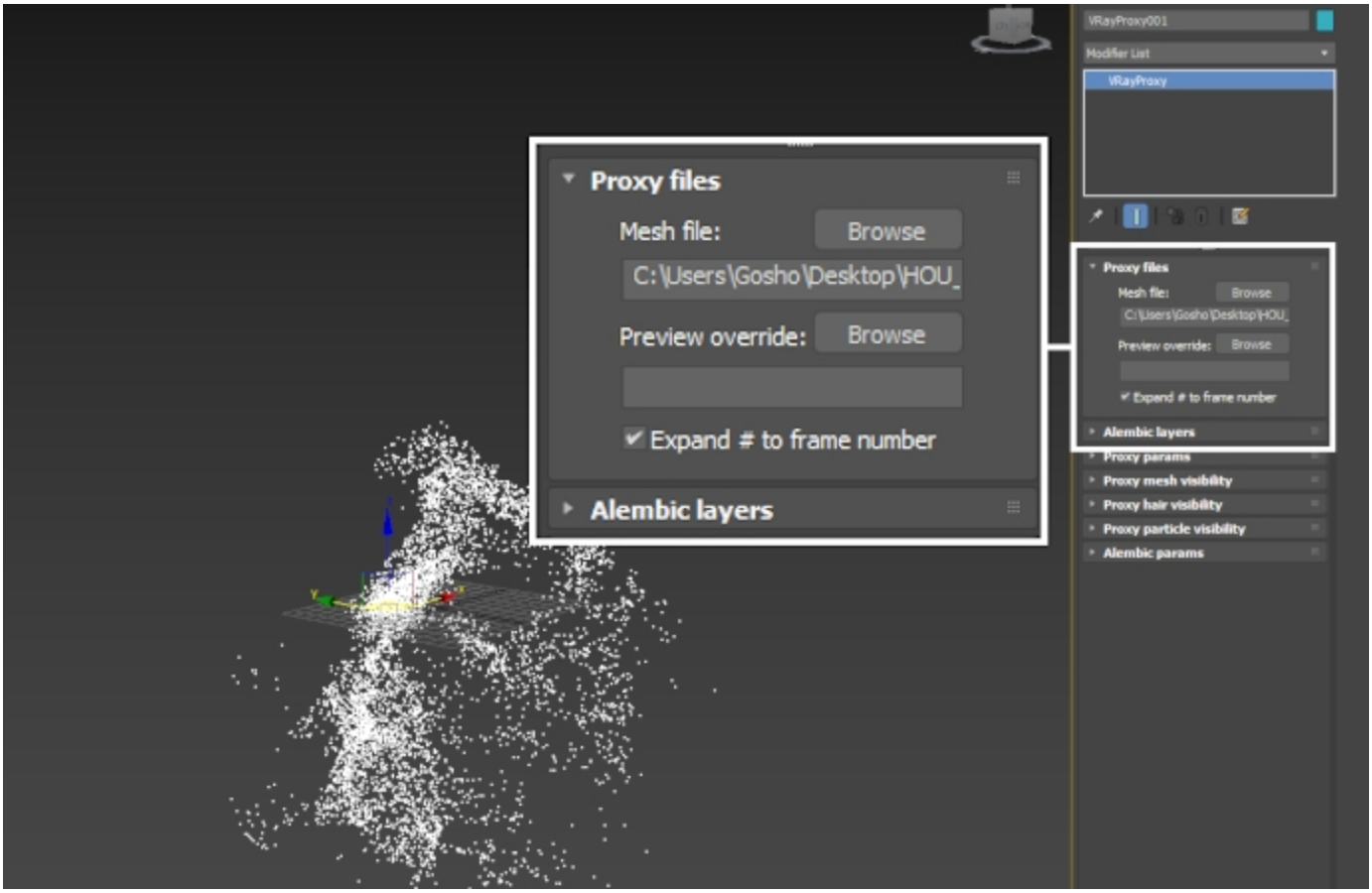


Import into 3Ds Max

Back to 3ds Max, go to **Create VRay VRayProxy**.

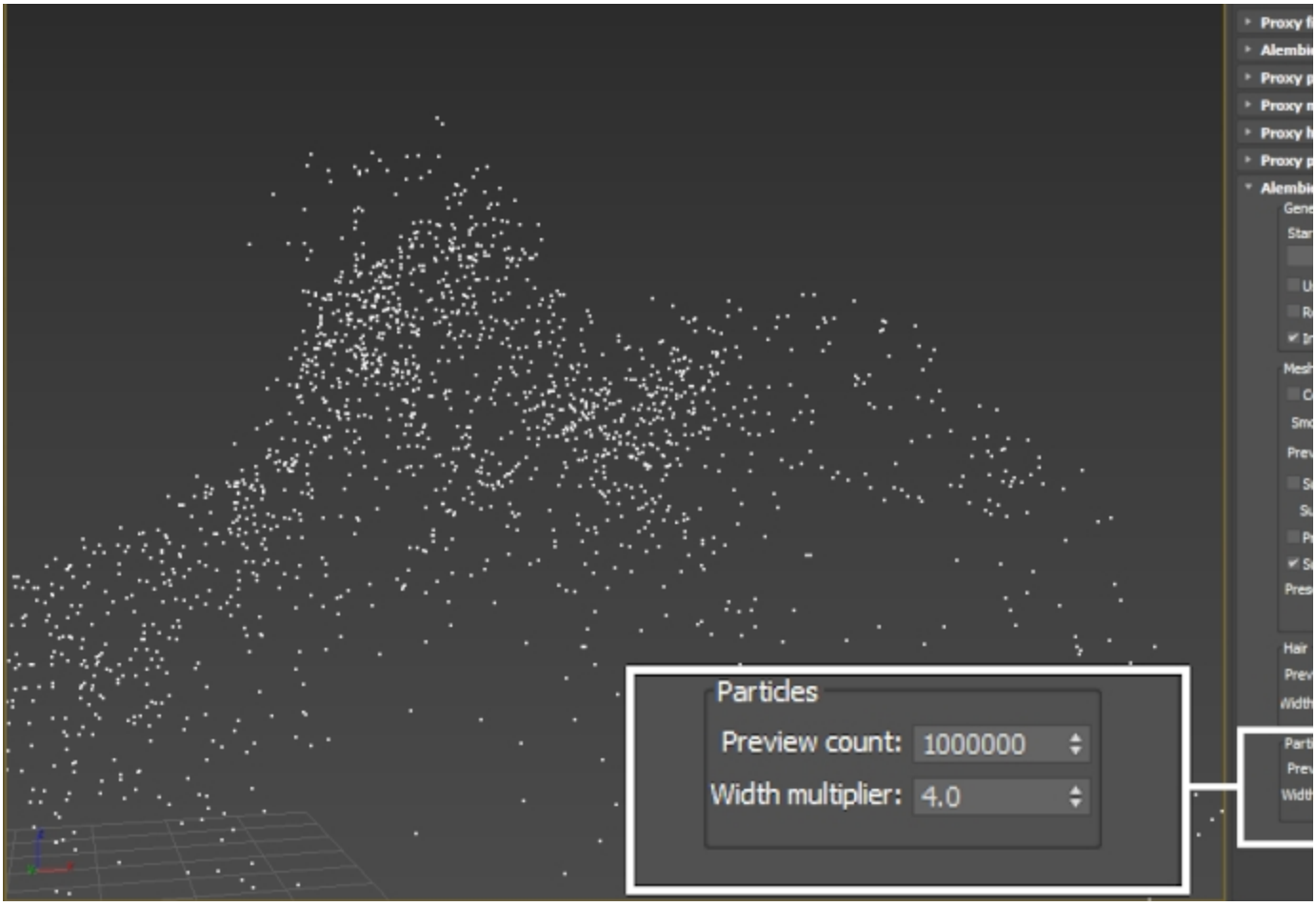
Load the exported **Alembic** file from the **Proxy Files** rollout **Mesh File Browse** button...

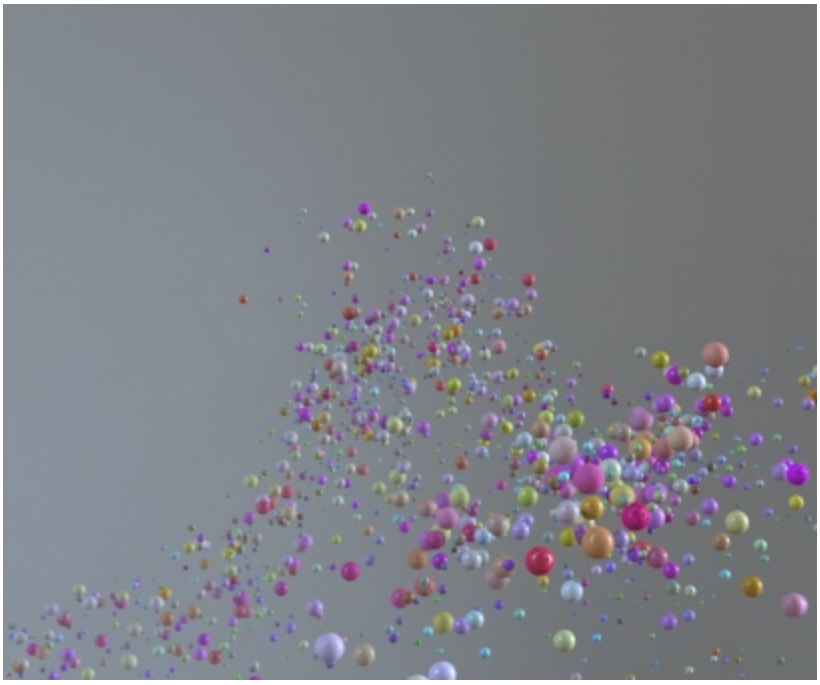
You may need to go to the **Proxy params** rollout and **enable Flip axis**.

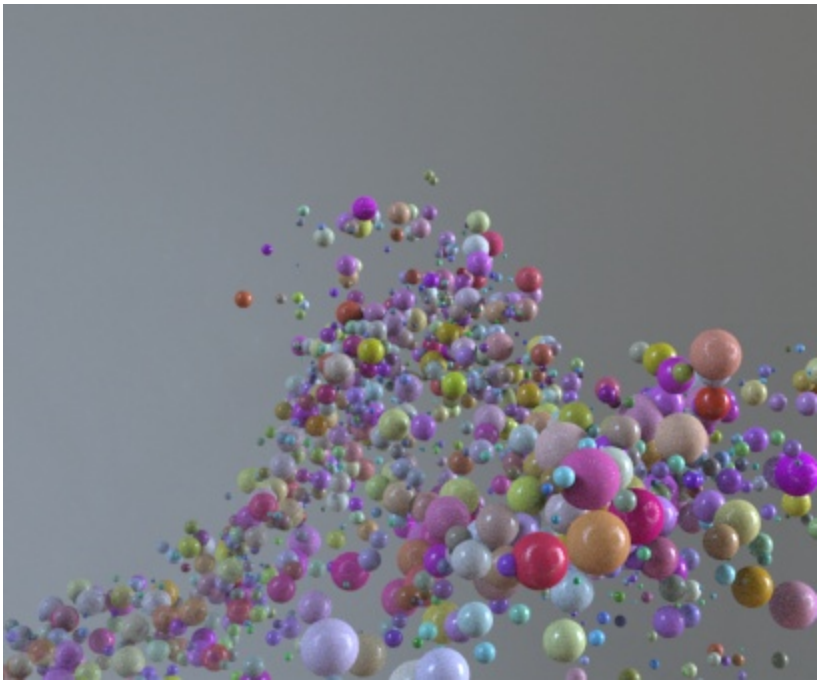


The VRay Proxy node comes with an overall control for the **size of the particles** which can be found under the **Alembic params Particles Width multiplier**.

The **Width multiplier** will NOT overwrite the **@width** attribute coming from Houdini but simply scale it based on the specified value.


Particles
Preview count: 1000000
Width multiplier: 4.0



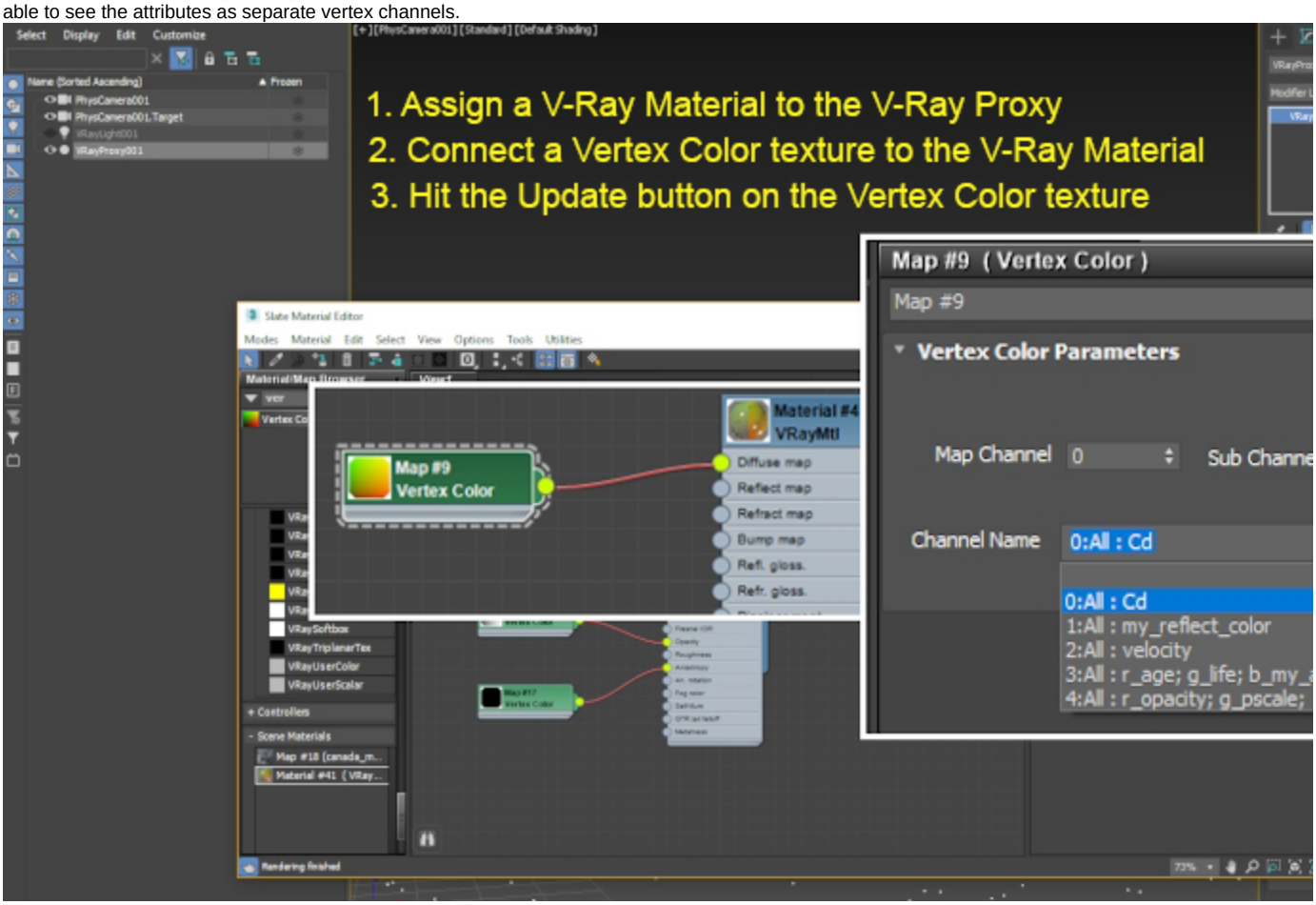


Width Multiplier is set to 2/4

Assign a V-Ray Material to the V-Ray Proxy.

Create a Vertex Color [1.1] texture. Note that the Channel Name drop-down is empty at the moment. You need to hook up the Vertex Color texture to the V-Ray Material and press the Update button for the attributes in the Alembic file to show up.

Assign the Vertex Color texture to the Diffuse map parameter of the V-Ray Material and press Update. You should now be able to see the attributes as separate vertex channels.



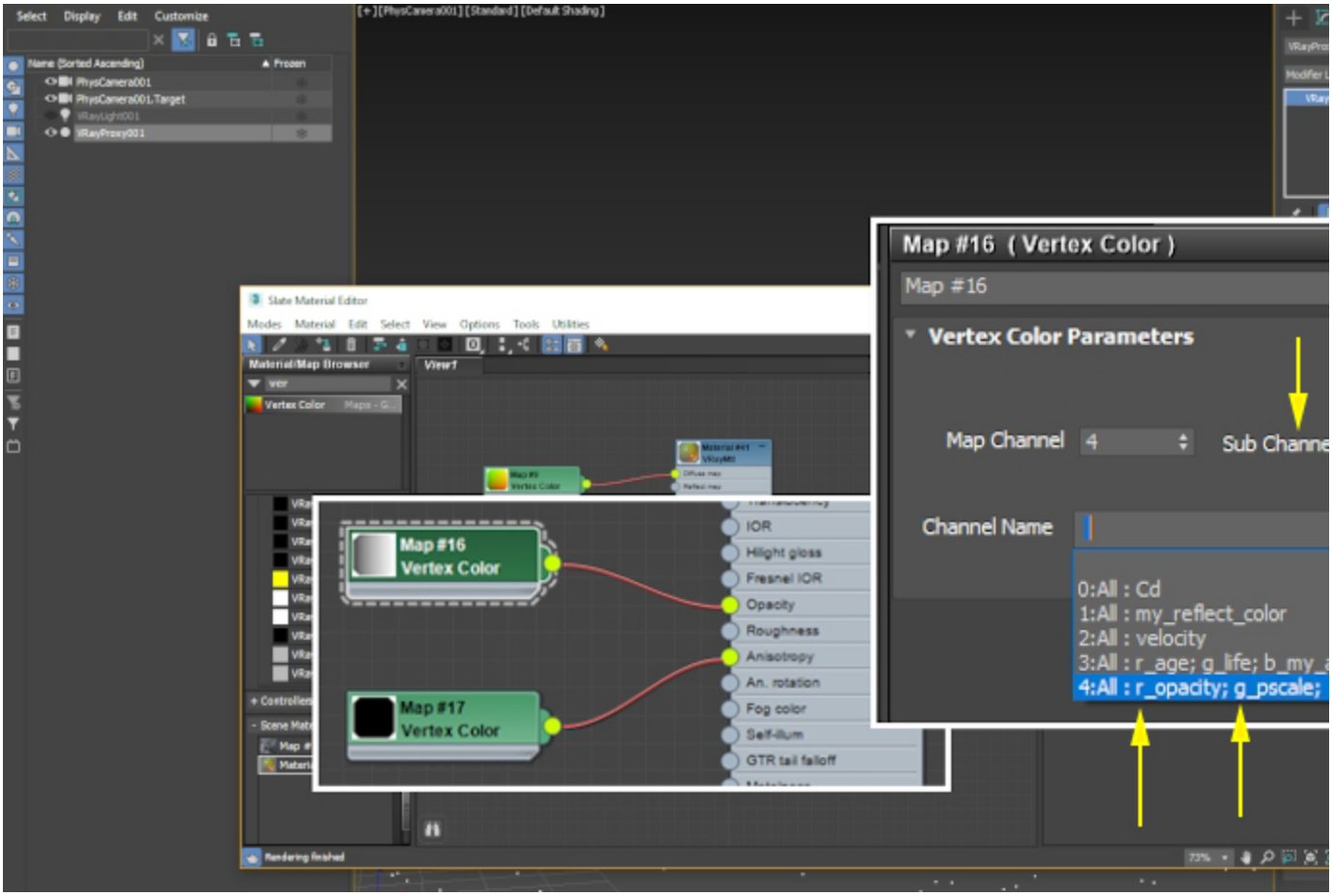
Setup for Float Attributes

The exact same workflow applies for the **Float** attributes but with one extra step – **you need to pay attention to the component of the Channel that the float attribute is assigned to**, and specify that same component in the **Sub Channel** drop-down.

In the picture to the left, you can see that the **opacity** attribute we created in Houdini is mapped to **Channel 4** under the **Red** (named **r_opacity** where **r_** means the **Red** component, **g_** means the **Green** component and **b_** means the **Blue** component) component.

To pull ONLY the **opacity** attribute, we have set the **Sub Channel** drop-down to **Red**.

If we want to map the opacity based on the life attribute, we would have to pick Channel 3 and Sub Channel Green.

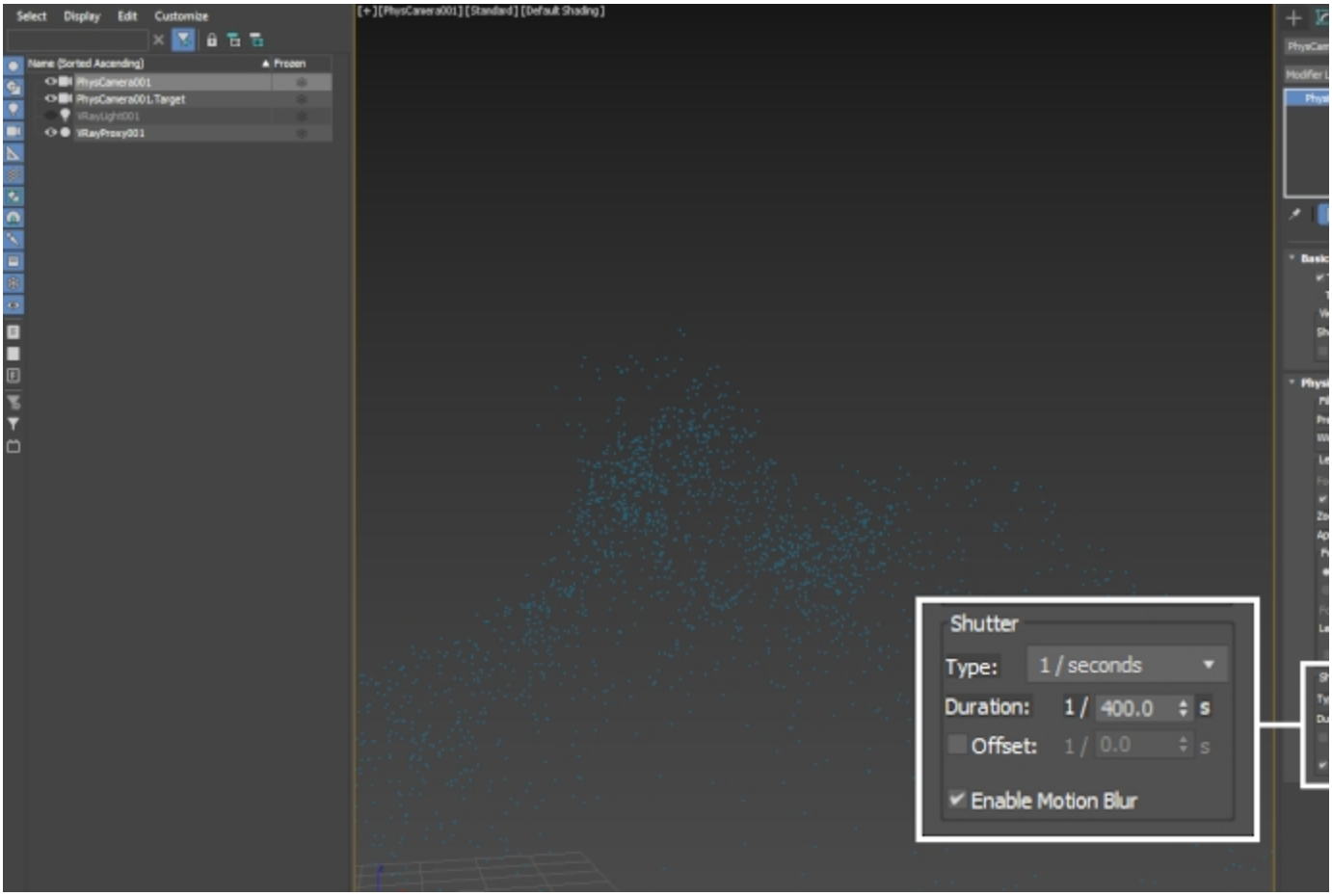


Motion Blur

Finally, to render the particles with **motion blur**, you can create a **V-Ray Physical Camera** and select **Enable motion blur** under the **Physical Camera Shutter** section of the parameters.

V-Ray automatically recognizes and uses the velocity attribute.The **magnitude** of the **motion blur effect** is determined by the **Shutter Duration** parameter.

To recap: the **@v attribute is used because the point count *is* changing**. This is a simulation - each frame contains more and more particles. **If the number of points was static, the @v attribute would be ignored**.



Footnotes

1 (1.1) – The use of Vertex Color texture can be changed with either VRayUserColor or VRayUserScalar textures when reading color sets and attributes of an alembic.