# NOTE WHILE THE CONTENTS OF THIS TUTORIAL IS HOUDINI, STRUCTURE OF THE CONTENT AND ANNOTATIONS, ETC. APPLY TO ANY SUBJECT

# Title: How to Procedurally Create a Bridge

## “The Problems” (Challenge)

In this tutorial, you model a bridge using procedural techniques. You will be able to change any of the following parameters of the bridge procedurally: length, width, separation between the support legs, and sag. The intent is to enable an artist to very quickly make different versions of a bridge by changing parameters within our tool.Along the way you will learn how to apply procedural techniques and use Houidini’s Surface Operators (SOPS).

## Level:

Beginner

## Prerequisites:

Prior to doing this tutorial, you should:

* Be able to navigate the Houdini interface
* Know how to lay down and connect nodes
* Have a general understanding of the Surface Operators Context (SOPS)
* A basic understanding of high-school algebra. Specifically you should know what a parabola [ y = x2 ] is and have an understanding of its graph

## Topics Covered:

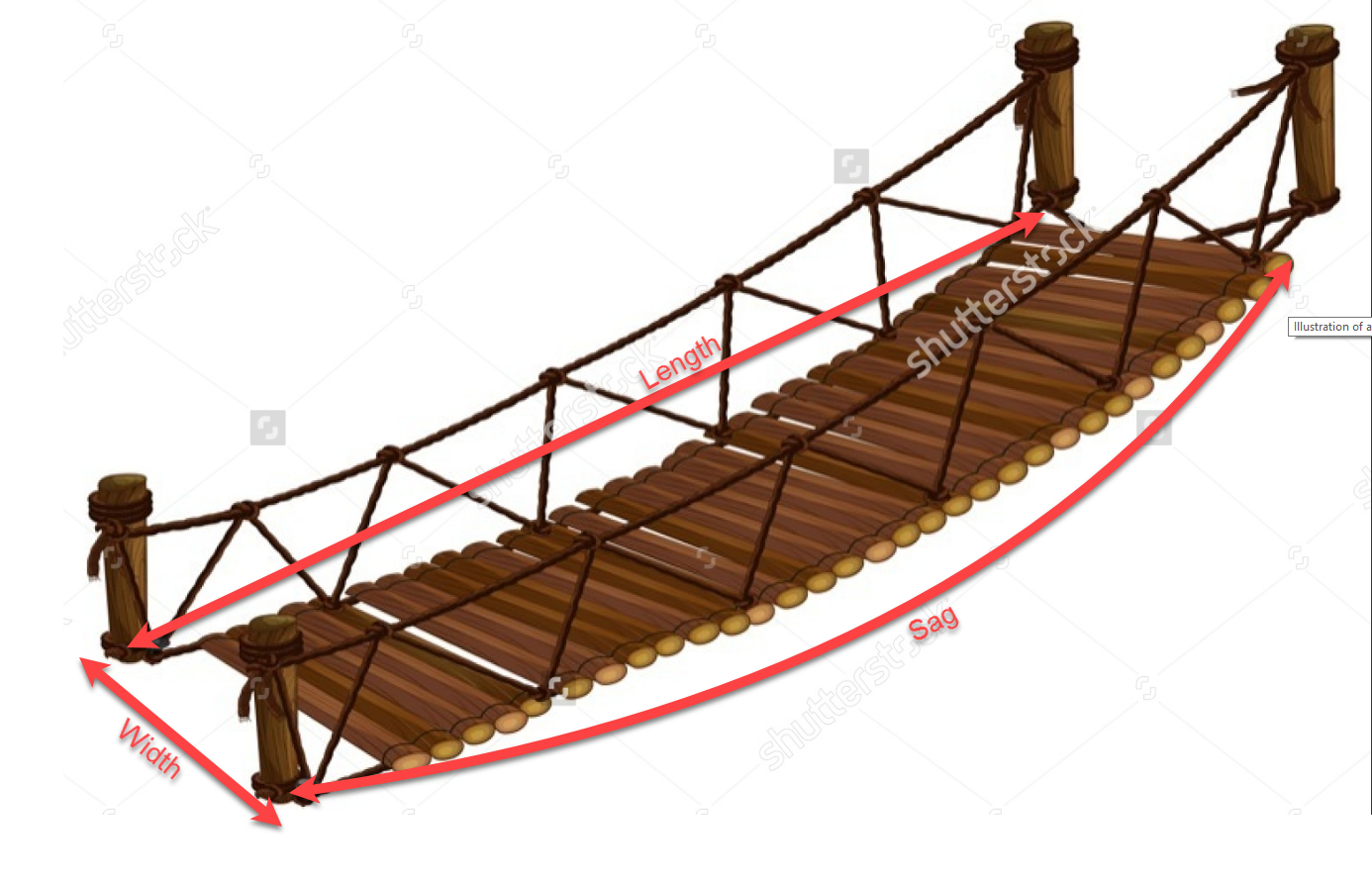
This tutorial will cover:

* Geometry Context
* Copy node
* Copy Stamping
* Driving copy process with VEX

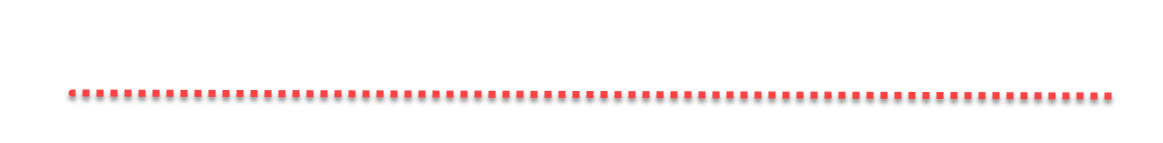
## Overview of our process:

We want a bridge that can be modified parametrically. Specifically we would like to be able to change:

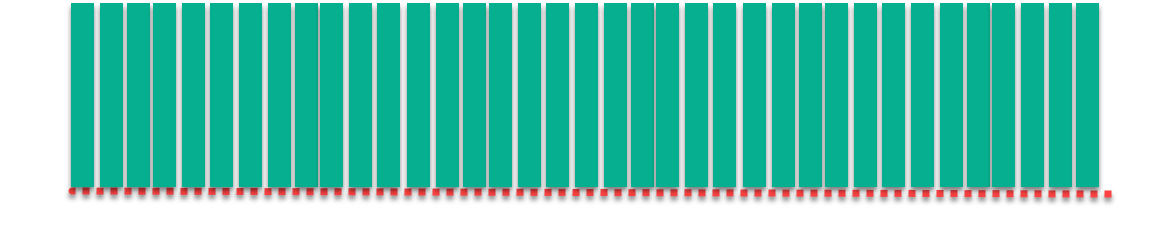
1. Length of the bridge
2. Width of the bridge
3. Sag of the bridge
4. Spacing between the legs



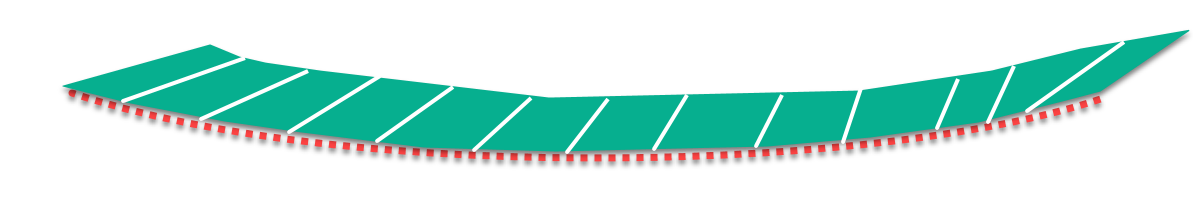
Our approach will be to first define create a straight curve, which will establish the length of our bridge.



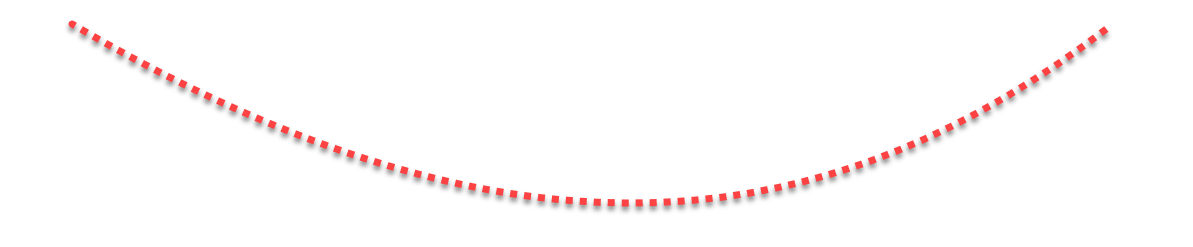
We will then copy flattened boxes across the bridge. These will serve as the boards across the bridge. The length of the boxes will determine the width of our bridge. Here we are looking at the bridge from the top.



To introduce sag into the bridge we will modify the original curve to follow the shape of a parabola with a very slight curvature.



For a lot of sag, we use a parabolic curve with a big coefficient (y=10x)



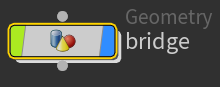
And for a small sag we use a curve with a log coeffiecient (y=.1x)



In reality our coefficient will always be very low (less than .06) or we will have a very saggy bridge.

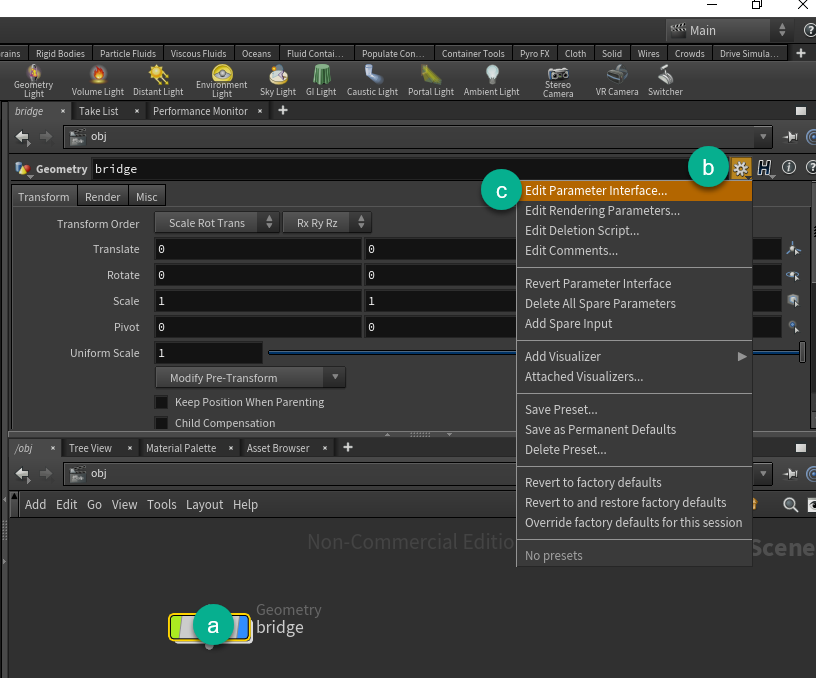
## Implementation

1. Put down a geo node and rename it to bridge



1. We are now going to add parameters our bridge geo object.  
   Do the following:
   1. Select the bridge geo node
   2. Click on the gear to modify the node
   3. Click on Edit Parameter Interface

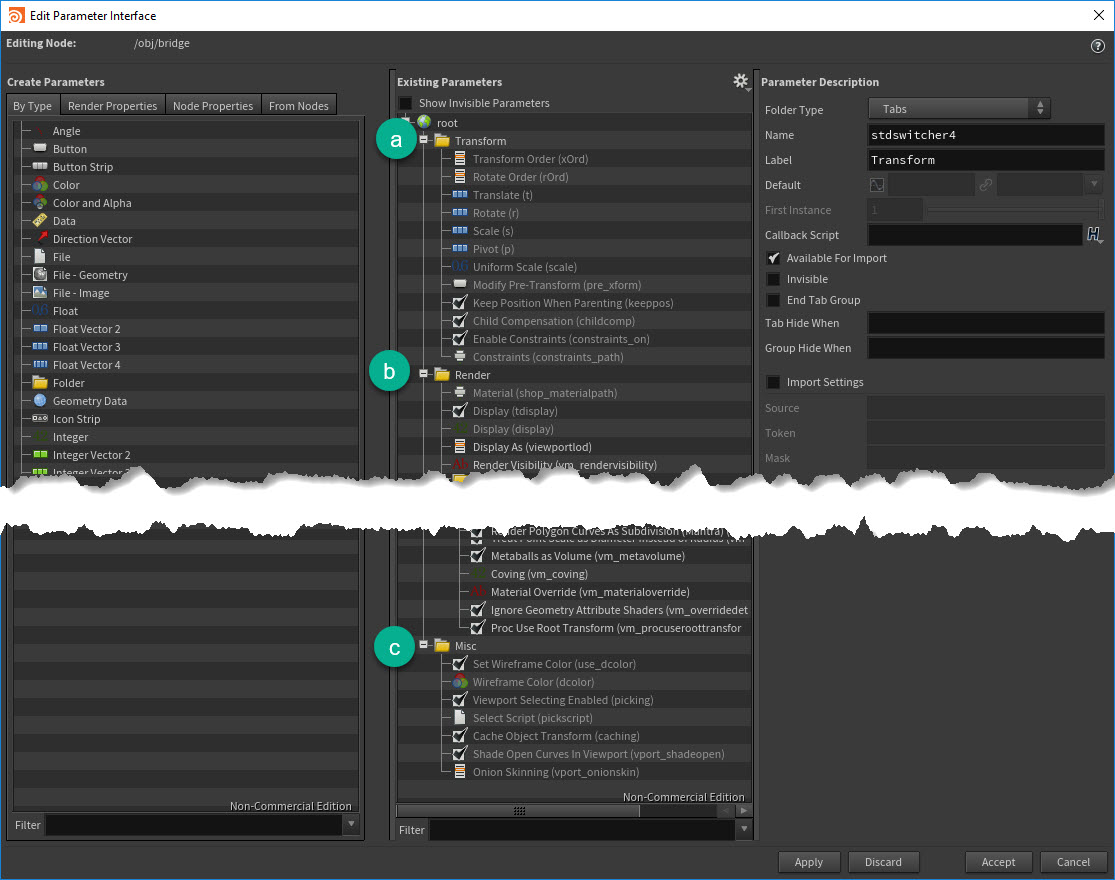
Why:

We would like control some aspects about the bridge from a global level. Here we can define data-entry fields on the interface to allow the user to enter numbers that we will use to shape the bridge. Specifically, we will create entry boxes for: length, width, sag, and spacing between the legs.  
  


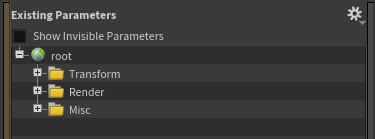
1. We are now going to collapse the Transform, Render, and Mic categories in the Existing Parameters box  
   Do the following:
   1. Click on the  next to the folder next to Transform to collapse it
   2. Click on the  next to the folder next to Render to collapse it
   3. Click on the  next to the folder next to Misc to collapse it

Why:

We do not want the new parameters that we create to end up under these categories.



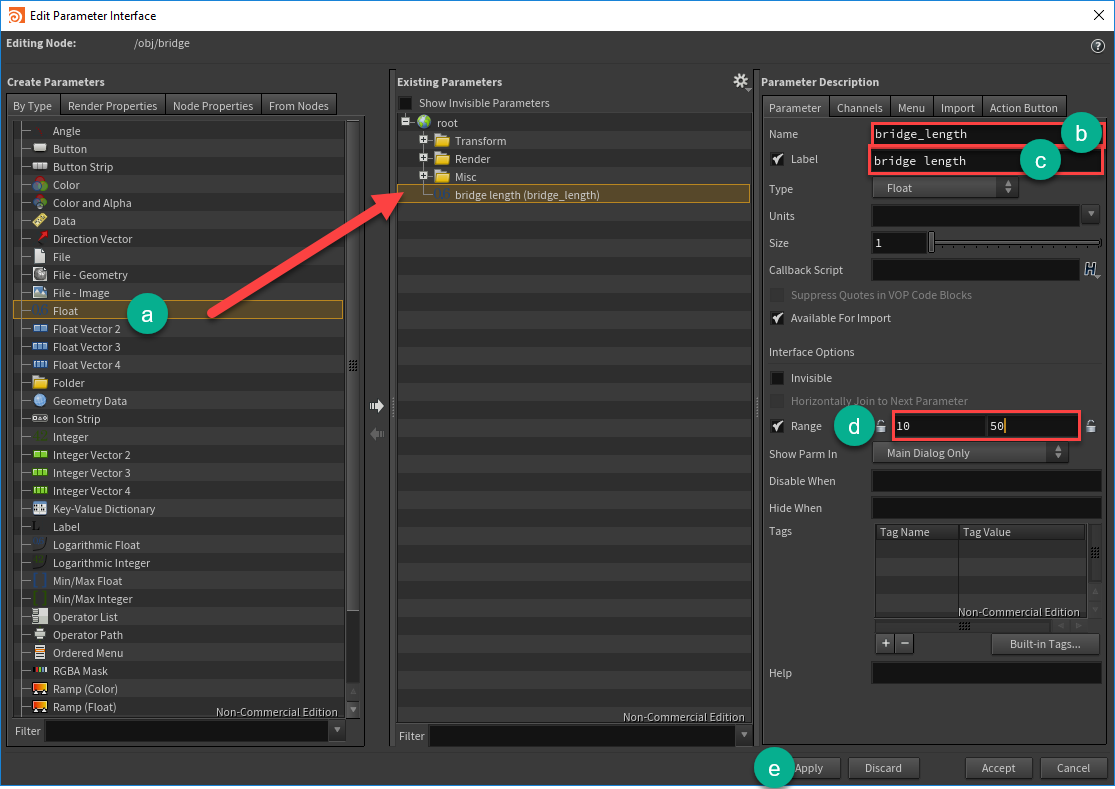
After you do this, the Existing Parameters box will look like:



1. We will now create the actual parameter entry boxes. I will show you how to do one specific one and then ask you to repeat the same steps for the rest.   
   Do the following:
   1. Select Float from the Create Parameters box and drag it over to Existing Parameters box. Be sure to NOT drop it inside of the existing folders (Transform, Render, Misc). It will highlight to show you where it will drop
   2. Change the name to bridge\_length. This must have not spaces in it.
   3. Change the label bridge length. You can name it anything you want. This will be the name that is shown on the interface when the user modifies the bridge.
   4. Change the lower end of the range to 10 and the upper range to 50. This will enable our bridge to be between 10 meters and 50 meters long.
   5. Click on Apply. This will take the values that entered without closing the Edit Parameter Interface box. If you click Accept, Houdini will also accept the value but it will also close the edit Parameter Interface box.

Why:

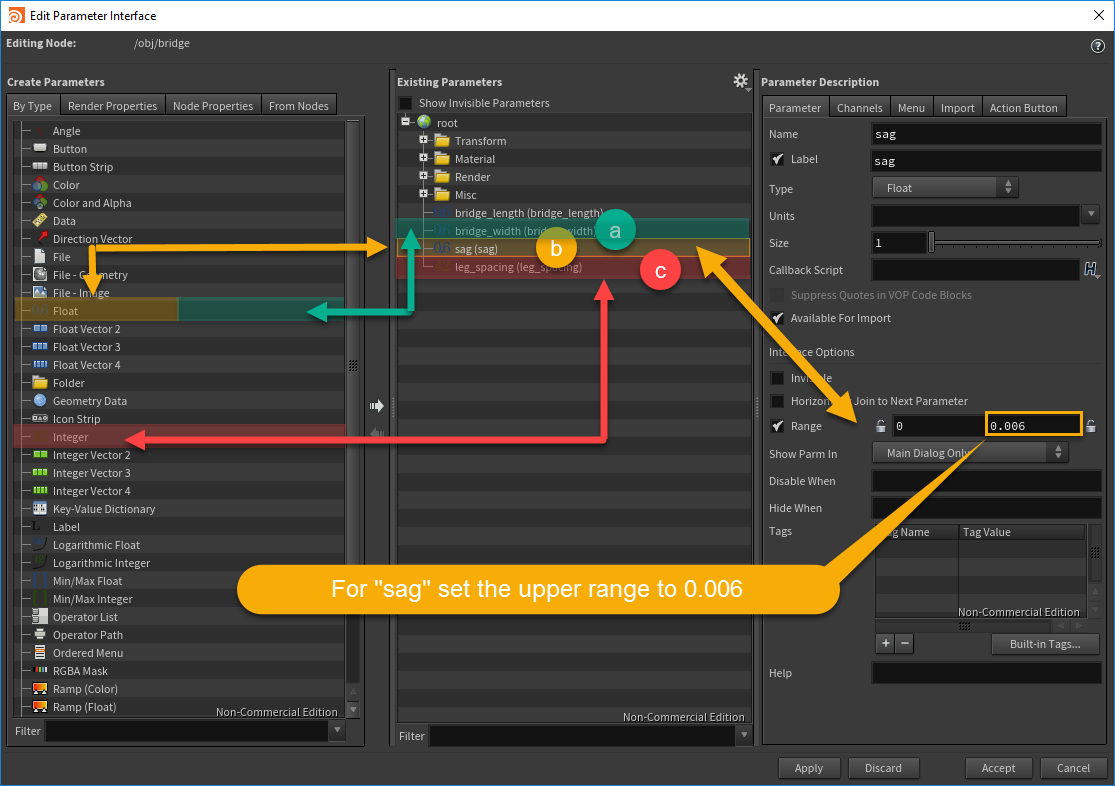
This step actually crates the UI element that will enable the user to enter values to drive the shape of the bridge.



1. We will now repeat the actions we took in the previous step to create fields for: bridge\_width, sag, and leg\_spacing.   
     
   Do the following:
2. Create a Float parameter with the name of bridge\_width. Set the lower range to 0 and upper range to 10.
3. Create a Float parameter with the name of sag. Set the lower range to 0 and the upper range to 0.006.
4. Create an Integer parameter with the name of leg\_spacing. Set the lower range to 0 and the upper range to 10.
5. Be sure to hit Apply after each time you create a paremeter. Then after you create the final parameter, click on Accept.

Why:

This step provided the user with a user interface to enter the values that we will use in our network to affect the shape of our bridge. For this to work, it is important that we specified the correct “Type” for our fields. In our case, all of our fields were Floats, except for the field leg\_spacing, which are going to require is an integer.   
  
Restricting the range to values that will produce predictable results is also important at this stage. In our case, the restriction of the sag to a maxiumum value of 0.006 is important because this is going to server as the coefficient in our parabolic shape.   
  
We probably should have restricted the lower bound to be slightly above 0, as entering zero will produce no bridge. But since the user has the option of manually overriding these range values that we have provided, we will enforce the bottom range at a later point using VEX.



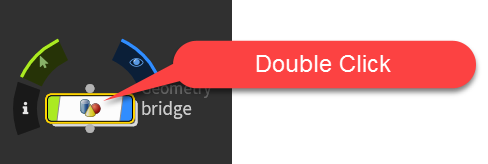
1. With our parameters now set up, we can start to build our bridge. We will do so in the geometry context.

Do the following:

1. Double click on the bridge geo node to go from the Scene context into the Geometry context

Why:

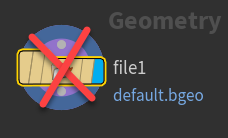
The actual geometry for the bridge needs to be building “inside” of a geometry object.



1. When you first open the geometry context, you will see a file1 node, which can be used to load a 3D object. We do not need this node, so we will delete it.

Do the following:

1. Select the file1 node and click delete.



1. We will now put down a box. We will manipulate this box later to form the slats across our bridge.

Do the following:

1. Put down a box node



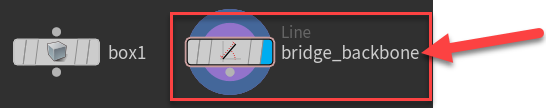
1. We will now create a line, which will serve as the backbone for our bridge. We will shape this line and then use the resulting shape to

Do the following:

1. Create a line node
2. Change the name to bridge\_backbone.

Why:

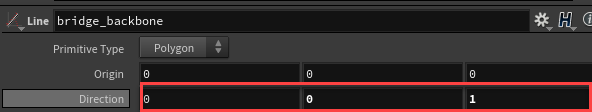
Blah blah blah blah



1. We will now change the direction of the line to go down the z-axis. The default is for it to go up the y-axis.

Do the following:

1. Change the direction from 0,1,0 to 0,0,1



Why:

Blah blah blah blah

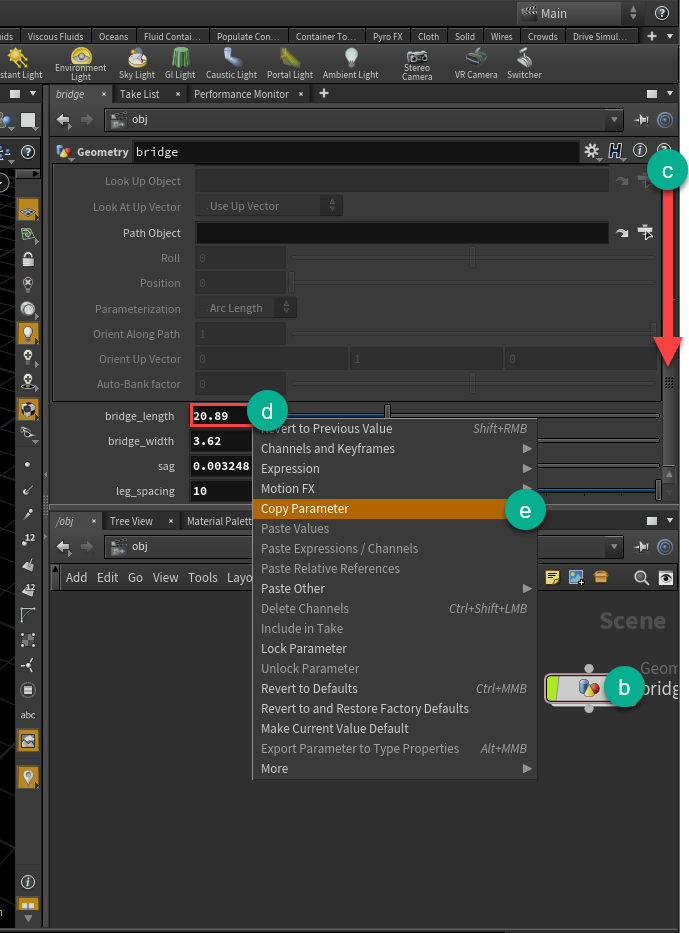
1. We will now tell Houdini that we want to use the bridge\_length parameter that we made earlier to drive the Length of our line. We also want to make the number of points on our line to be equal to double the amount we specify in bridge\_length.  
     
   Note:  
   This procedure will require us to between the geo and scene level. I will show you all of the steps involved to link the bridge\_length to the Length of the lines. Then for the number of points, I will ask you to repeat the procedure. From now on, whenever I say “link parameter A to B”, be sure to follow this procedure.

Do the following:

1. Go up to Scene level by hitting the U key while you mouse hovers over the graph editor.

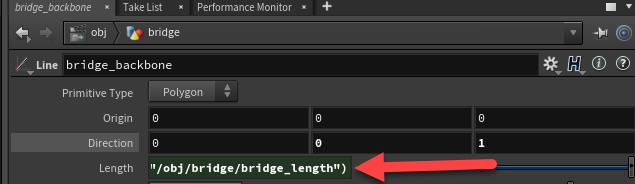


1. Select the bridge geo node
2. Scroll down to the bottom of parameter box to find the parameters that we created earlier.
3. Right Click inside of the box that says bridge\_length. This will make a drop down menu appear
4. Select Copy Parameter
5. Double Click on the Bridge geo object (not shown as a step below) to go back into the Geo level.



1. Select the bridge\_backbone line object that you made earlier.
2. Right Click inside of the Length box.
3. Select Paste Absolute References. Alternatively you can click on Paste Relative References (see yellow step i). Both will work. The only difference is the way that Houdini specifies the path to the objects.

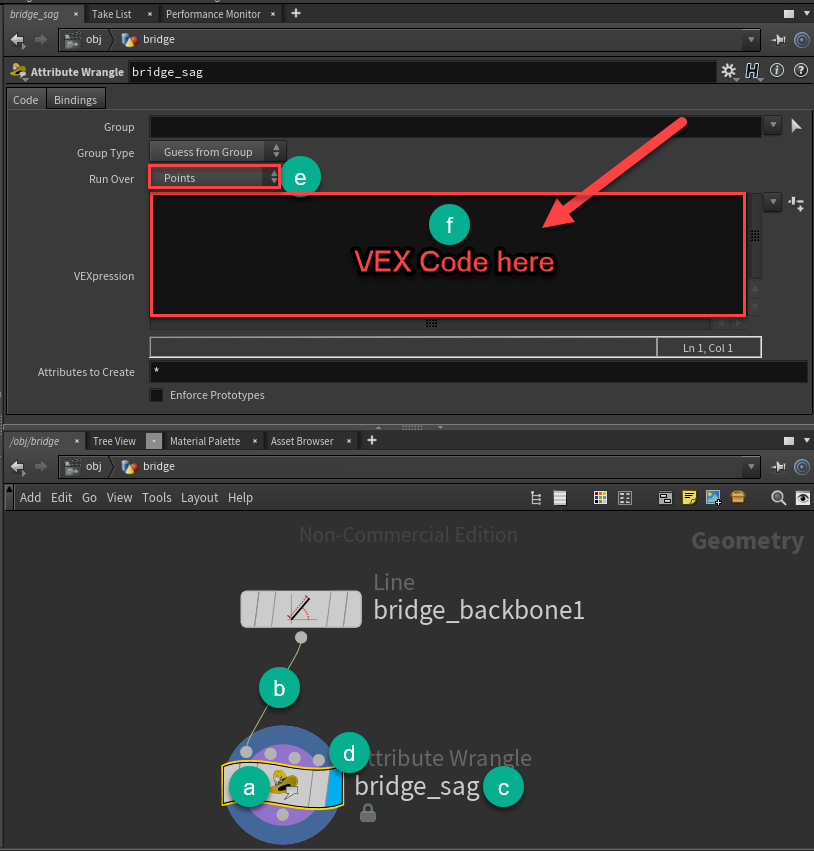
When you get done, Length box will change as shown below. Note the front end of the line clipped in the graphic below. It says: **ch("/obj/bridge/bridge\_length") \* 2**



1. We will now change the number of points in the line to be double the length of the line. Luckily, we still have the reference copied into our clipboard. So we can paste it using the same technique as in the previous step and multiple it by 2.  
     
   Do the following:
2. Paste the reference to the bridge\_length into the Length box in the line and add a \*2 at the end to multiply by two. It should say:

**ch("/obj/bridge/bridge\_length") \* 2**

1. We will now change the number of points in the line to be double the length of the line. Luckily, we still have the reference copied into our clipboard. So we can paste it using the same technique as in the previous step and multiple it by 2.  
     
   Do the following:
2. Create an Attribute Wrangle node
3. Connect the Line (bridge\_backbone) to the first input of the wrangle node
4. Change the name of your wrangle node to bridge\_sag
5. Put the visibility flag on the wrangle node.
6. Verify the Run Over parameter is set to Points. This will insure our VEX code runs over the points in our model.
7. We will be placing VEX code in this box for the next step.

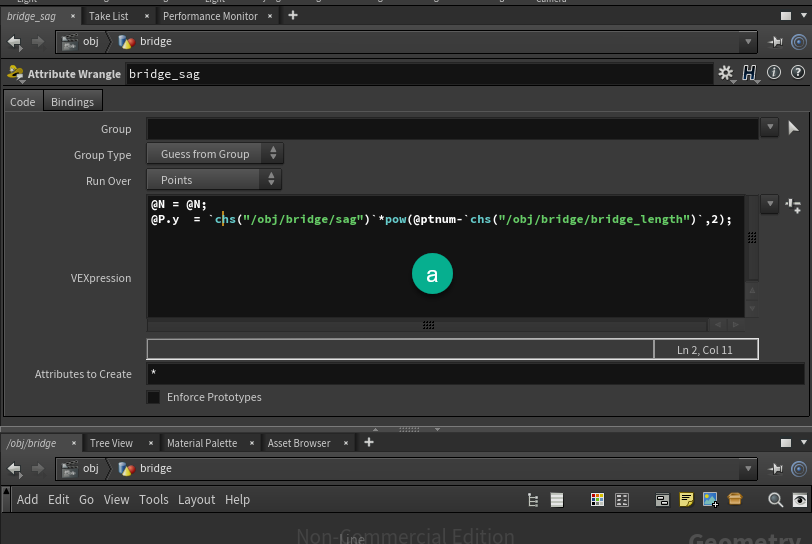


1. Add VEX code  
     
   Do the following:
2. Type the following in the VExpression box.

Note: Do not type the line numbers [1:, 2:, 3:, etc.]. The line numbers have been added to provide us with a rapid way to reference code.

1. @N = @N;
2. @P.y = `chs("/obj/bridge/sag")`\*pow(@ptnum-`chs("/obj/bridge/bridge\_length")`,2);

After you enter the code type Control + Enter to commit the code.



Why:

Blah blah blah blah

# Quiz

1. What does SOPS stand for?