


Before We Begin:


- Tools/Options - Under Document Properties, set the scene units to Inches
- Create any surface

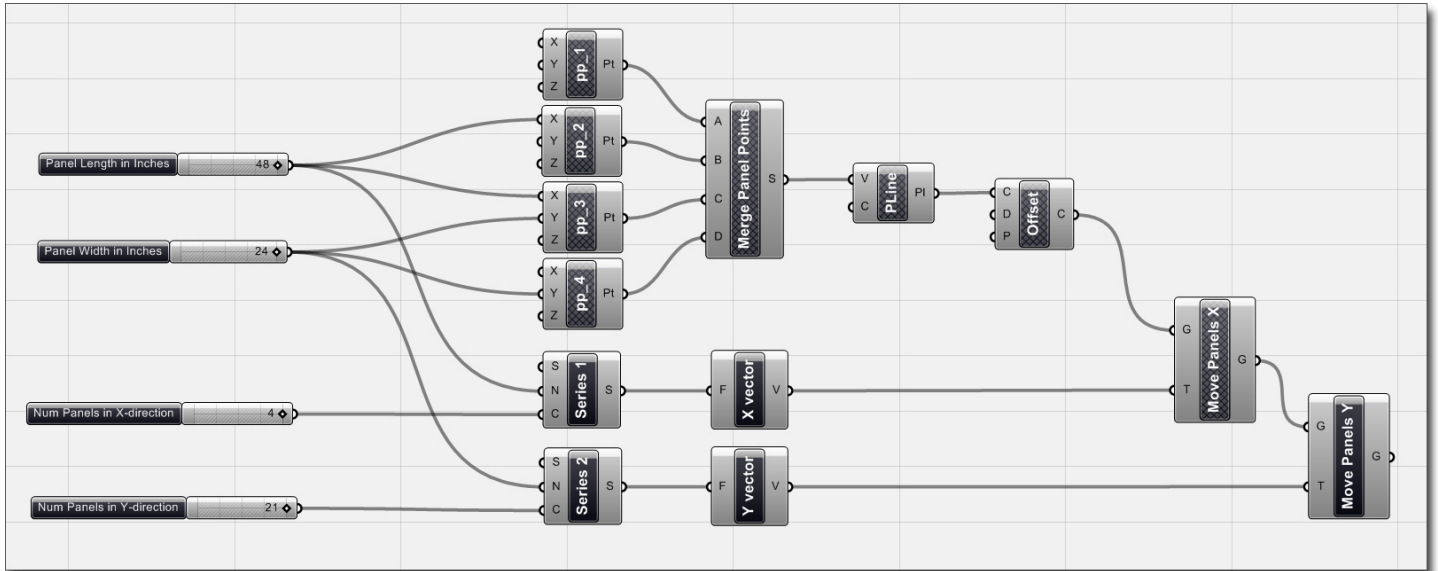
 *The definition needs a surface that is larger than the point grid area in order to function properly, so the first step is to generate a surface using any of Rhino's surface creation methods. Since this tutorial was written for a specific project in mind, we will use those dimensions to start our project. The suspended ceiling was 16'-0" wide x 42'-0" long, so our surface needs to be slightly bigger than these dimension. The Rhino scene file that accompanies this tutorial has two surfaces that are already built to these dimensions.*

Step 1: Create Parametric Panel Outlines

- Launch the Grasshopper Editor
- Params/Util/Number Slider - Drag out Number Slider component
 1. Rename Slider "Panel Length in Inches"
 - i. Set slider type: Integer
 - ii. Set lower limit: 12
 - iii. Set upper limit: 48
 - iv. Set value: 48
- Copy and Paste Panel Length in Inches slider
- Rename copied slider to "Panel Width in Inches"
- Set upper limit of Panel Width in Inches to 24
- Set value of Panel Width in Inches to 24
- Vector/Point/Point - Drag out Point component
- Rename Point component "pp_1"
- Check to make sure pp_1 X,Y, and Z values are set to zero
- Copy and Paste pp_1 component 3 times to create 4 total point components
- Rename the three new copied point components pp_2, pp_3, and pp_4 respectively
- Connect Panel Length in Inches slider to pp_2-X value and to pp_3-X value
- Connect Panel Width in Inches slider to pp_3-Y value and to pp_4-Y value
- Logic/Streams/Merge 4 - Drag out Merge 4 Streams component
- Rename M4 component "Merge Panel Points"
- Connect pp_1 to Merge Panel Points-A
- Connect pp_2 to Merge Panel Points-B
- Connect pp_3 to Merge Panel Points-C
- Connect pp_4 to Merge Panel Points-D
- Curve/Spline/Polyline - Drag out Polyline component
- Connect Merge Panel Points-S to Pline-V
- Set Pline-C Boolean to True (hint: this will create a closed polyline)
- Curve/Util/Offset - Drag out Offset Curve component
- Connect Pline-PI to Offset-C
- Set Offset-D (distance) to 0.125 (hint: we need to allow room for our t-bar structure, so this will offset the panel 1/8" on all sides, giving us a total 1/4" clearance for the structure)
- Turn Preview off for pp_1, pp_2, pp_3, pp_4, Merge Panel Points, and Pline
- Params/Util/Number Slider - Drag out Number Slider component
 1. Rename Slider "Num Panels in X-direction"




- i. Set slider type: Integer
 - ii. Set lower limit: 1
 - iii. Set upper limit: 4
 - iv. Set value: 4
- Copy and Paste Num Panels in X-direction
 - Rename copied slider to "Num Panels in Y-direction"
 - Set upper limit of Num Panels in Y-direction to 21
 - Set value of Num Panels in Y-direction to 21
 - Logic/Sets/Series - Drag out Series component
 - Rename Series component "Series 1"
 - Copy and Paste Series 1 component
 - Rename copied Series component "Series 2"
 - Connect Panel Length in Inches to Series 1-N
 - Connect Num Panels in X-direction to Series 1-C
 - Connect Panel Width in Inches to Series 2-N
 - Connect Num Panels in Y-direction to Series 2-C
 - Vector/Constants/Unit-X - Drag out Unit-X component
 - Rename Unit-X component "X vector"
 - Vector/Constants/Unit-Y - Drag out Unit-Y component
 - Rename Unit-Y component "Y vector"
 - Connect Series 1-S to X vector-F
 - Connect Series 2-S to Y vector-F
 - X-Form/Euclidian/Move - Drag out Move component
 - Rename Move component "Move Panels X"
 - Connect X vector-V to Move Panels X-T
 - Connect Offset-C to Move Panels X-G
 - Right click on the Move Panels X component and set the algorithm to "Cross Reference"
 - Turn Preview off for Offset component
 - You should now see four panels copied in the X-direction
 - X-Form/Euclidian/Move - Drag out Move component
 - Rename Move component "Move Panels Y"
 - Connect Y vector-V to Move Panels Y-T
 - Connect Move Panels X-G to Move Panels Y-G
 - Right click on the Move Panels Y component and set the algorithm to "Cross Reference"
 - Turn Preview off for Move Panels X
 - You should now see all twenty one copies of the row of four panels just created, totaling 81 panels in the entire system


 *Test the system by changing the values of the Panel Length in Inches, Panel Width in Inches, Num Panels in X-direction, and the Num Panels in Y-direction sliders. You could extend the definition by extruding the panel outlines to create the thickness of the panel material. What components would you use to solve this?*



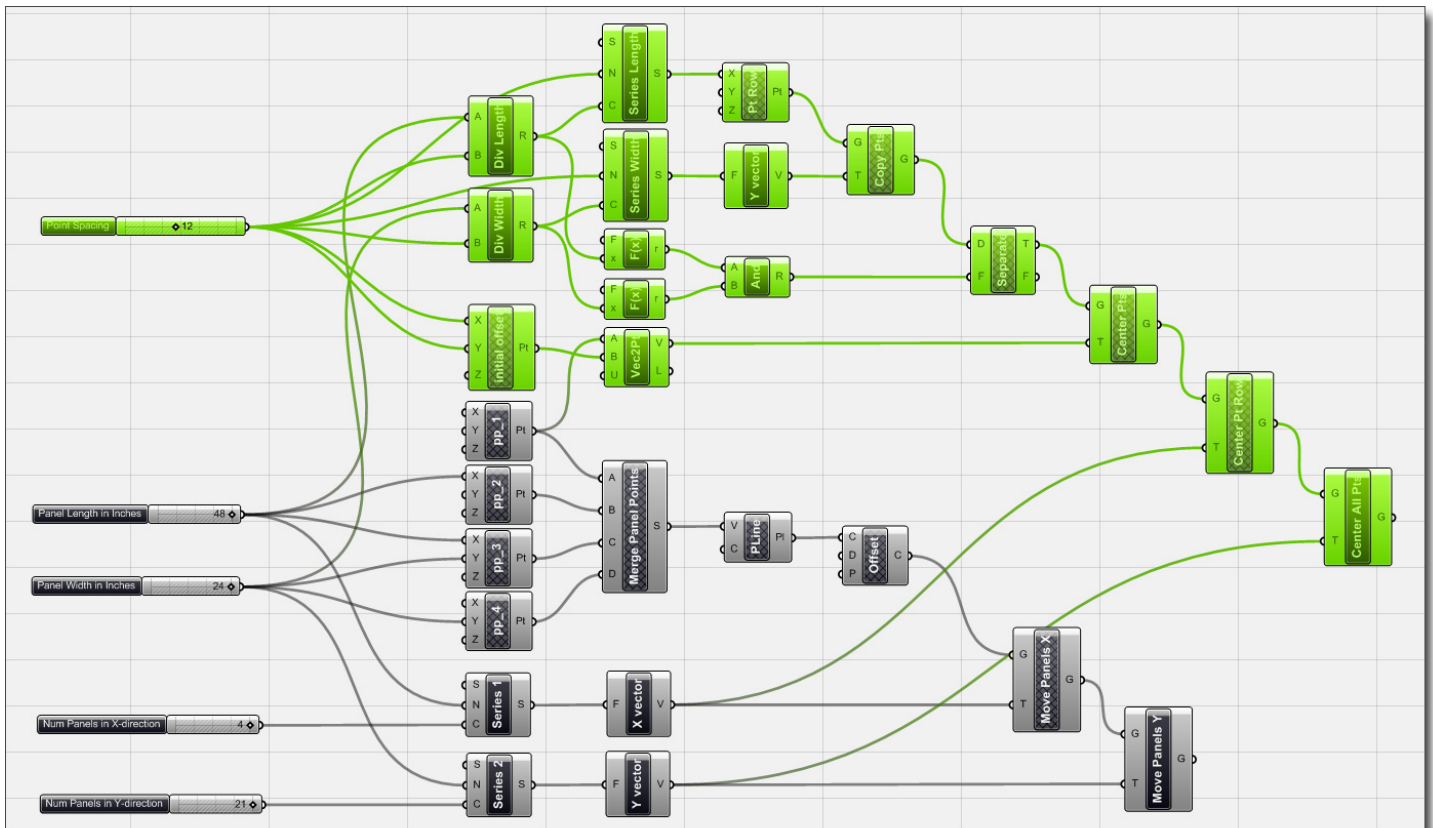
Step 2: Create self-centering point grid array

- Params/Util/Number Slider - Drag out Number Slider component
 1. Rename Slider "Point Spacing"
 - i. Set slider type: Integer
 - ii. Set lower limit: 2
 - iii. Set upper limit: 24
 - iv. Set value: 12
- Scalar/Operators/Divide - Drag out Divide component
- Rename Divide component "Div Length"
- Copy and Paste Div Length
- Rename copied Divide component "Div Width"
- Connect Point Spacing to Div Length-B
- Connect Point Spacing to Div Width-B
- Connect Panel Length in Inches to Div Length-A
- Connect Panel Width in Inches to Div Width-A
- Logic/Sets/Series - Drag out Series Component
- Rename Series component "Series Length"
- Copy and Paste Series Length component
- Rename copied Series component "Series Width"
- Connect Point Spacing to Series Length-N
- Connect Div Length-R to Series Length-C
- Connect Point Spacing to Series Width-N
- Connect Div Width-R to Series Width-C
- Vector/Point/Point - Drag out Point component
- Rename Point component "Pt Row"
- Connect Series Length-S to Pt Row-X (hint: you should now see a row of points that are being driven by the Point Spacing slider)
- Vector/Constants/Unit Y - Drag out Unit Y component
- Rename Unit Y component "Y vector"
- Connect Series Width-S to Y vector-F
- X Form/Euclidian/Move - Drag out Move component
- Rename Move component "Copy Pts"
- Connect Y vector-V to Copy Pts-T

- Connect Pt Row-pt to Copy Pts-G
- Right Click on Copy Pts and set algorithm to Cross Reference (you should now see both rows and columns of points that are being driven by the Point Spacing slider)
- Scalar/Expressions/F1 - Drag out Single Variable Function component
- Right click on F of the F(x) component and select the Expression Editor
- Click on Display Function List to show available expression commands
- Under this list, you will see a definition of the Ceiling(x) expression
- Close Function list and set expression to "Ceiling(x) = x"
 -  *The ceiling expression will return the smallest integer greater than or equal to the specified number (x). Basically it's a round up expression.*
- Copy and Paste F(x) component
- Connect Div Length-R to the first F(x)-x
- Connect Div Width-R to the second F(x)-x
- Logic/Boolean/Gate And - Drag out Gate And component
- Connect the first F(x)-r to Gate And-A
- Connect the second F(x)-r to Gate And-B
 -  *Now, test the Point Spacing slider. Essentially what the two single variable functions are doing is checking the Point Spacing slider and dividing that number into the Panel Width and Length. If the Point Spacing slider value divided evenly into the Panel Width and Length, then the expression result will show a True boolean value. If it doesn't divide evenly into the Panel Width and Length, then the expression result will show a False boolean value. Try setting the Point Spacing Slider to 8 and the expression result will show True. Now set the Point Spacing Slider to 9 and the expression result will show False. The Gate And component combines both expression results into one boolean value. The Gate And component requires two True values for it to show an end result of True.*
- Logic/Streams/Dispatch - Drag out Dispatch component
- Rename Dispatch component "Separate"
- Connect Gate And-R to Separate-P
- Connect Copy Pts-G to Separate-L
- X Form/Euclidean/Move - Drag out Move component
- Rename Move component "Center Pts"
- Connect Separate-A to Center Pts-G
 -  *We need to create a translation vector that will center the points on the panel. To do this, we will need to create a diagonal vector from the origin point (0,0,0) to a point that has been offset from the origin by half the distance of the point spacing (pt spacing/2, pt spacing/2, 0). Now, when we change the point spacing, the offset vector will change as well.*
- Vector/Point/Point - Drag out Point component
- Rename Point component "Initial Offset"
- Connect Point Spacing to Initial Offset X & Y
- Right click on Initial Offset X and set Expression to "X/2"
- Right click on Initial Offset Y and set Expression to "Y/2"
- Vector/Vector/Vector 2Pt - Drag out Vector 2Pt component
- Connect pp_1 to Vec2Pt-A
- Connect Vec2Pt-V to Center Pts-T
- Connect Initial Offset-Pt to Vec2Pt-B
- Turn Preview off for Initial Offset, Pt Row, Copy Pts, and Separate components

 Test the system by changing the Point Spacing slider. You should get 1 panel of points that self-center themselves inside the bounding box of the panel. You should notice that the points only show up on the intervals that divide evenly into the Panel Width and Length. For instance, if our Panel Width is 24 and our Panel Length is 48, then the self-centered points will only show up when the Point Spacing slider is set to 2,3,4,6,8,12, and 24. This system will be useful when we copy this point array to the other panels in the system because the point spacing between panels will now be the same as the Point Spacing within each panel.


- X Form/Euclidean/Move - Drag out Move component
- Rename Move component "Center Pt Row"
- Connect Center Pts-G to Center Pt Row-G
- Connect X vector (step 1) to Center Pt Row-T
- Right click on Center Pt Row and set algorithm to Cross Reference
- Turn Preview off for Center Pts
- X Form/Euclidean/Move - Drag out Move component
- Rename Move component "Center All Pts"
- Connect Center Pt Row-G to Center All Pts-G
- Connect Y vector (step 1) to Center All Pts-T
- Right click on Center All Pts and set algorithm to Cross Reference
- Turn Preview off for Center Pt Row



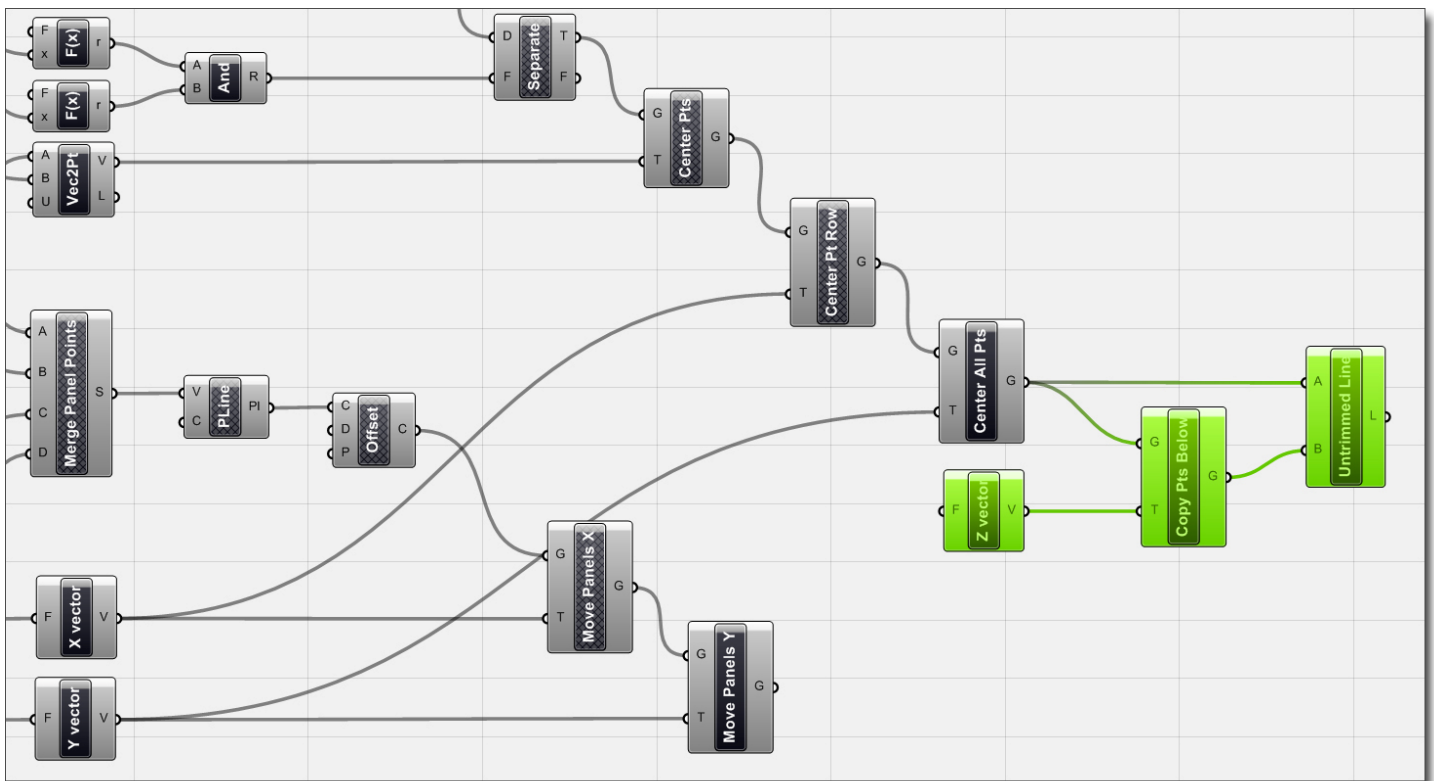
Step 3: Copy point array in Z-direction and draw line between points (below the surface)

- Vector/Constants/Unit Z - Drag out Unit Z component
- Rename Unit Z component "Z vector"

- Right click on Z vector-F and set value to -120.0


 Our units are set to inches, so our translation vector is $-10'-0"$. You could also set this up to a slider if you wanted the points to be parametric, however it is unnecessary in this example.

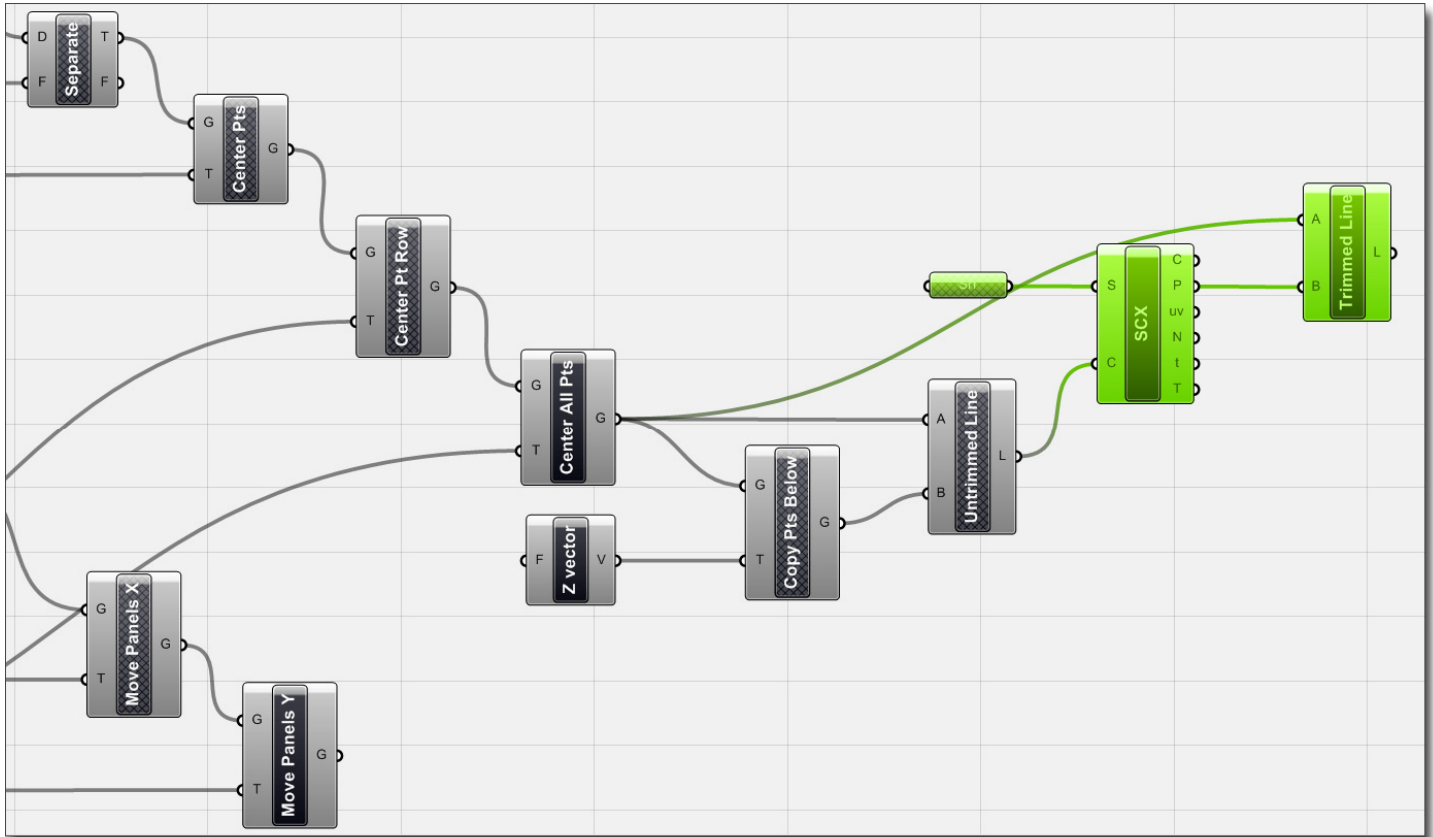
- X Form/Euclidean/Move - Drag out Move component
- Rename Move component "Copy Pts Below"
- Connect Z vector-V to Copy Pts Below-T
- Connect Center All Pts-G to Copy Pts Below-G
- Curve/Primitive/Line - Drag out Line component
- Rename Line component "Untrimmed Line"
- Connect Center All Pts-G to Untrimmed Line-A
- Connect Copy Pts Below-G to Untrimmed Line-B



Step 4: Solve surface/curve intersection and draw new line between points

- Params/Geometry/Surface - Drag out Surface component
- Right click on Srf component and Set One Surface
- When prompted, pick pre-generated surface already created in the scene (see steps in the "Before we Begin" section)
- Intersect/Physical/SC Intersect - Drag out Surface/Curve Intersection solver
- Connect Untrimmed Line-L to SCX-C
- Connect Srf to SCX-S
- Turn Preview off for Copy Pts Below, Untrimmed Line, and Srf
- Curve/Primitive/Line - Drag out Line component
- Rename Line component "Trimmed Line"


- Connect SCX-P to Trimmed Line-B
 - Connect Center All Pts-G to Trimmed Line-A
-  Try assigning different surfaces to the Srf component. The trimmed lines will automatically update to any new surface and desired Point Spacing.



Step 5: Create smart tag and assign to each vertical line

- Curve/Analysis/Length - Drag out Length component
- Rename Length component "Line Lengths"
- Connect Trimmed Line-L to Line Lengths-C
- Logic/List/List Length - Drag out List Length component
- Connect Center Pts-G to List Length-L
- Scalar/Operators/Multiply - Drag out Multiply component
- Rename Multiply component "Panel Num"
- Connect Num Panels in X-direction to Panel Num-A
- Connect Num Panels in Y-direction to Panel Num-B
- Logic/Sets/Series - Drag out Series component
- Rename Series component "Series 1"
- Set Series 1-S to 1.0
- Copy and Past Series 1 component
- Rename copied Series component "Series 2"
- Connect Panel Num-R to Series 1-C
- Connect List Length component to Series 2-C
- Logic/Sets/Duplicate Data - Drag out Duplicate Data component

- Rename Duplicate Data component "Dup 1"
- Copy and Paste Dup 1 component
- Rename copied Duplicate Data component "Dup 2"
- Connect Series 1-S to Dup 1-D
- Connect List Length-L to Dup 1-N
- Connect Series 2-S to Dup 2-D
- Connect Panel Num-R to Dup 2-N
- Right click on Dup 1-O and set boolean value to False
- Right click on Dup 2-O and set boolean value to True

 *The duplicate data components are prepping the data for each label. A simplified example would be to assume we have 4 panels in our system with 2 points per panel. Here is what the data would look like for each Duplicate Data component:*

- Dup 1 (for the panel #), keep list order intact? = False
1,1,2,2,3,3,4,4*
- Dup 2 (for the points), keep list order intact? = True
1,2,1,2,1,2,1,2*

We will use a three variable function to compile the Panel Number, the Line Number, and the length into one data tag. To do this, we will feed the Dup 1 Data to the first variable, the Dup 2 Data to the second variable, and the Line Lengths Data to the third variable.


- Scalar/Expressions/F3 - Drag out Three Variable Function
- Connect Dup 1-D to Function-x
- Connect Dup 2-D to Function-y
- Connect Line Lengths-L to Function-z
- Right click on Function-F and open up the Expression Editor
- Set the Expression to the following:
Format("Panel {0}_Line Num {1}_ {2:0.000}",x,y,z)

 *The expression above will create a data string that looks like this:
Panel 1_Line Num 1_72.000*


We have now created a data string that tells us that the Length of Line Number 1 on Panel Number 1 is 72.000 inches. We can change the accuracy of the Line measurement by changing the number of decimal zeros in the last part of the expression. For example, if we wanted our Line Length to read 72.0 instead of 72.000, our expression would look like this:

Format("Panel {0}_Line Num {1}_ {2:0.0}",x,y,z)


- Curve/Util/Divide Curve - Drag out Divide Curve component
- Connect Trimmed Line-L to Divide Curve-C
- Set Divide Curve-N to 2.0

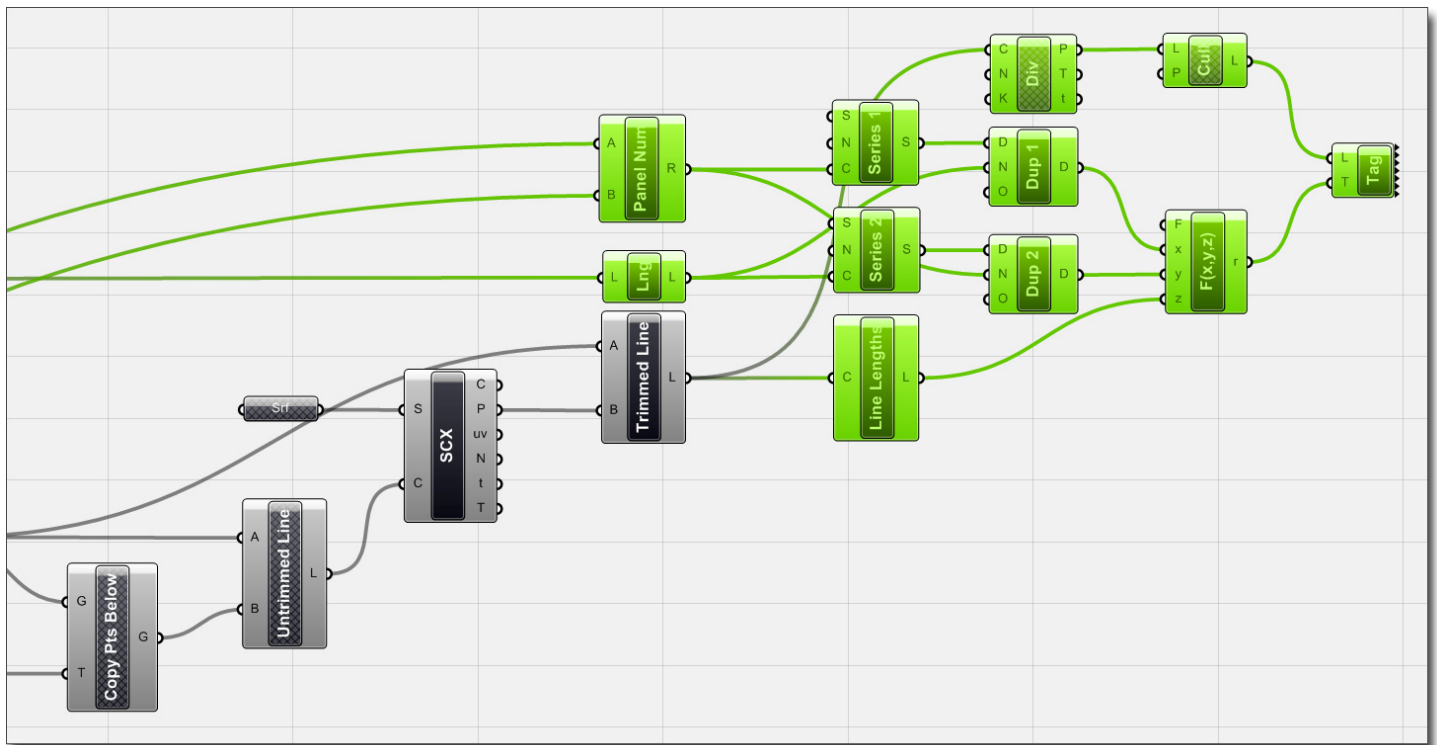
 *The Divide Curve component will divide each vertical line into 2 smaller lines of equal length, ultimately creating 3 points; The start point, the midpoint, and the end point.*

- Logic/Sets/Cull Pattern - Drag out Cull Pattern component
- Right click on Cull Pattern component and click Set Multiple Booleans
- When prompted, set the first boolean to False, second boolean to True, and the third boolean to False. Hit enter after you have set all three boolean values.

 *When we connect this Cull Pattern component to the Divide Curve component, it will cull the start point and end points because the first and third boolean values have been set to False. It will keep the midpoint because the second boolean value was set to True.*

- Connect Div-P to Cull-L
- Vector/Point/Text Tag - Drag out the Text Tag component
- Connect Cull Pattern-L to Text Tag-L
- Connect the three variable Function-r to Text Tag-T

 The text tag component will now place a text tag of our function expression that was defined as Panel #_Line #_Length at the midpoint of each vertical line. Because we first chose to generate our original point, and then we copied that point in the X-direction, and then copied that row in the Y-direction, our text tag will label our Line Number in the same manner. Similarly, since we copied the first full panel of points in the X-direction, and then copied those panels of points in the Y-direction, our text tag will label the Panel Number the same way. Select both the Trimmed Line component and the Text Tag component, and Bake the data into the scene. Use the distance command to double check that your Line Lengths and Text Tag data match up.



This tutorial has been written by Andrew Payne from LIFT architects. This software and its documents are in the public domain and are furnished “as is”. The author, makes no warranty, expressed or implied, as to the usefulness of the software and documentation for any purpose. This work is licensed under a Creative Commons Attribution-Share Alike 3.0 United States License. <http://creativecommons.org/licenses/by-sa/3.0/us/>

Should you have questions or inquiries, please contact him at the address below.

Andrew Payne
 LIFT architects
 2060 Vallejo Street, Apt. 4
 San Francisco, CA 94123
 (e) andyopayne@gmail.com
 (w) <http://www.liftarchitects.com>