**STEP 00 • IDEA**

- Setting up a framework
- Surface from lines

- Generate triangular polylines out of the surface
  - Note triangle ordering sequence

- Defining a component
  - Get a centroid of a triangle surface

- Get a line following the normal direction
- Move and scale the base triangle surface
+ Get vertices out of two triangular surfaces
+ Draw 3 curves out of 3 sets of 4 points

+ Offset 3 curves

+ Move and scale the base triangular surface
+ Get vertices out of surfaces

+ Draw 3 curves out of 3 sets of 4 points
+ Get vertices from two surfaces
+ Draw 3 curves out of 3 sets of 3 points

+ Draw 3 lines

+ Get points

+ Draw interpolation curves out of points
+ Sweep 2 rails to get 6 bottom surfaces

+ Loft 3 transitional surfaces

+ Sweep 2 rails to get 3 side surfaces

+ Component application onto framework
Duplicate and flip triangular curves to make an interlocking structure.
+ **Draw rhino curves**
+ Connect the curves with a GH curve object and then loft

+ **A number slider for surface division**
+ 2 one-variable function objects for both U and V
  - For U, set 4\(x\)
  - For V, set 2\(x\)

+ **Divide the lofted surface**

+ **VB script object with 3 input and 4 output tabs**
  - 'pts' as list of on3dpoint
  - 'divU' & 'divV' as integers
Map 1 dimensional linear point list onto 2 dimensional list

- ptsList = list of list of on3dpoint (2 dimensional list, U & V)
- ptsRow = list of on3dpoint (1 dimensional list, V direction)

1 dimensional list: (n)
2 dimensional list: (U)(V)

Make point arrays
Make polylines with the point arrays
Set the polylines as output A

Repeat for output B
+ Repeat for output C

+ Repeat for output D

+ Output: 4 sets of triangular polylines
To avoid data matching confliction, start with 2 triangular polylines instead of one
- Connect rhino curves with GH curve object
- Connect Graft Tree with GH curve object
  - Note the different data structures
  - By Grafting, every item is now in the same hierarchy instead of being a member of 1 dimensional list, which will eventually give more control on them

Get centroids

Draw lines using a Line SDL object

To get normal vectors at each centroid, attach SrfCP and EvalCrv object
  - SrfCP is to get a surface uv point which is the closest one from a certain point. Usually, the point does not necessarily have to be on the surface. However in our case, since the centroid is on the surface itself, SrfCP object simply converts the coordinate system of the centroid from world into UV.
+ Explode surfaces to get edge lengths
+ Convert edge curves into lines to use open nurbs length method
+ Attach VB script object to get the longest edge lengths

+ Line lengths (Height of the components) are to be 75% of the lengths of the longest edges

+ Evaluate(L=0.2) the lines to get reference points for move

+ Move the base surfaces to the reference points we’ve just got
+ Scale the surfaces by 0.2

+ Get vertices out of the scaled surfaces: explode and then retrieve vertices using cull patterns (T-F-F/F-T-F/F-F-T)

+ Repeat for the lower surfaces

+ Make 4 point curves with degree 3, not periodical
+ Connect point objects (parameter) to Crv object following the order
+ Connect points to the Crv object
  - Note the point connecting order

+ Fails to draw curves
  - Note the different data structures

+ To make them have the identical data structure, attach Graft tree objects as shown

+ Repeat for the other curves
+ Attach offset objects to the curves

+ Offset distances are 10% of the longest edge lengths

+ Set offset base planes

+ Repeat for the rest of curves
  + Bottom curves are done
To get side curves, move the base surfaces along with center lines to the points at $L=1$, $L=0.6$, $L=0.4$ and $L=0.2$

Set scale factors: $0.2/0.2/0.3/1.0$

Explode the surfaces and retrieve vertices

Repeat for the rest of surfaces
1. Get curves through points

2. Repeat for the other curves
   - Side curves are done

3. To get the middle curves, move the base surfaces to the points at $L=0.4$ and $L=0.2$
   - Set the scale factors: $0.3 / 1.0$

4. Explode the lower surfaces and retrieve vertices
+ Explode the upper surfaces and evaluate edges at L=0.5 to get midpoints

+ Retrieve points

+ Make curves through 3 points with degree 2

+ Repeat for the other curves
+ Done with the middle curves
+ Get top curves

+ Move the base surfaces at L=1.0 and scale down by 0.2

+ Retrieve edges
+ Top curves are done

+ Now, we need some more curves to build bottom surfaces
+ First, we need curves shown in the figure. To avoid confusion, extract those curves using receiver objects
Second, we need center points shown in the figure, so get it from eval object from the previous step.

Also, we need the base surfaces to get vertices out of them.

Align them to make it clear.

Get mid points out of curves (L=0.5).
+ Get vertices out of the base surfaces

+ Make interpolate curves through 3 points
  + Not working

+ Compare data structures

+ To make them have identical data structure, attach some Graft tree objects to the corresponding points
Complete interpolate curves

Split curves using shatter object
• Note how to get t values

Repeat shattering

Shatter the interpolate curves
+ Get t values using CrvCP object
  - CrvCP is basically same with SrfCp, but it works on curves

+ Repeat shattering

+ Done with curve works

+ Now we are ready to make surfaces
Get bottom surfaces with sweep 2 rails
- Note the curve input order

Nothing happened
- Input data structure looks fine

To check out the direction of rail curves, bake them and check their directions

The rail curves are running in the wrong directions
+ Flip rail curves

+ Repeat sweep 2 rails

+ Get middle surfaces by lofting

+ Repeat lofting
- Get side surfaces using sweep 2 rails
- Profile curves are not running in the same direction
- Bake profile curves to check out their directions
- Flip top curves
Done with the components
STEP03 • COMPONENT APPLICATION

+ Connect the components and the framework (A and B only)

+ Offset the loft surface (base surface for the framework)

+ Flip triangular polylines (C and D only) and connect them to the component to make an interlocking structure system
+ Draw some curves you want

+ Set those curves as multiple input