overview

• The purpose of this tutorial is to create a folding origami skin based on parametric principles.

• The first step is to make a square grided surface based on one convex and concave curves. Their z-axis planars are aligned and then arrayed.

• The second step is to use data structures in order to get access to every point on the surface.

• The third step is to create the vectors and the line segments defined by start point, tangent and length which will follow the direction of the folding surfaces.

• The last step is to connect all the end points of the vectors to create the surfaces for an uniform folding + the option to create an attracting point which would dictate a non-uniform one.
+ Draw two curves in Rhinorecors - one should be the the 'base curve' and the other one the 'profile curve'.
  • It does not matter what kind of curves you are drawing - as long as they are perpendicular to each other.
+ Connect grasshopper and rhino objects.

+ From the Curve toolbar > Analysis > select Planar - to test the profile curve for planarity.
  • at its end, you will see a plane which is parallel to the curve
From the Vector toolbar > Plane > select Align plane to perform minimal rotation in order to align a plane with a guide vector
- the plane before was parallel to the curve, but it was not parallel to a vector in the Z-axis.
- Select from the Vector toolbar > Constants > Unit Z [vector in a Z direction]

From the Curve toolbar > Utilities > choose Flip
- If more than 50% of the base curve orientation is anti-parallel to the profile curve, the base curve is flipped and the ‘flip action’ is set to True. If no guide is provided, the curve is always flipped.
- Again from the Curve toolbar > Division > select Pframes (perpendicular frames)
- Assign a slider to [N - number of segments], which will be the amount of the frames on the base curve.
  - PFrames - generate a number of equally spaced, perpendicular frames along a curve.
Select the XForm toolbar > Euclidian > Orient
- Orient - Orientation is sometimes called a ‘ChangeBasis transformation’. It allows for remapping of geometry from one axis-system to another.

From the Params toolbar > Geometry > Point
- Point - Represents a list of 3D Point coordinates.

Select Curve toolbar > Spline > PLine [poly line]
- PLine - Create a polyline connecting a number of points.
Choose Logic toolbar > Sets > Cull Pattern : right click on Cull.P (boolean parameter) > manage boolean collection

- Cull (remove) elements in a list using a repeating bit mask [like the chess board]. The bit mask is defined as a list of Boolean values. The bit mask is repeated until all elements in the data list have been evaluated.
- Select the Cull parameter and you will see several green and red pairs of points - due to the fact that in the boolean manager we have [false,false,true,true].

Delete the command in the boolean manager and type [false, true]
- Now we have every other point to be green and red.
Select another Cull Pattern parameter, but this time in the boolean manager type [True, False]. This will allow us to separate every point from its neighbors and to be in a different group.

Choose from the Vector toolbar > Point > Distance.
- Distance: Compute the distance between two point coordinates.

From the Params toolbar > Special > Panel - the numbers in the panel represent the distance between the points on the base curve.
- Panels can also receive their information from elsewhere. If you plug an output parameter into a Panel, you can see the contents of that parameter in real-time. All data in Grasshopper can be viewed in this way. You are able to see all the data ‘behind’ every component.
Since our profile curves have been assigned to the perpendicular frames on the base curve - the number of the frames will dictate the density of the origami surface.

Select from the Curve toolbar > Division > DivLenght (Divide Length)

- DivLenght - Divide a curve into segments with a preset length

You can see how the profile curves are divided by the same distance length numbers in the yellow panel.
The more we increase slider’s number, more points are dividing the profile curves.

The numbers in the panel are decreasing which is telling us that the vertical curves are divided by points with smaller distance.

- It is good, when we are modeling in Grasshopper, not to keep the density of the surface too big in order not to slow down our modeling process.

- Keeping in mind all the information so far we are going to enter into the second part of this tutorial where we will start working with data structures.
From the Logic toolbar > List > Shift List [Shift]
- Shift List - Items in the list are moved towards the start and the end of a list if the shift offset is positive or negative.

Create two Shift components, connect them with the Divide Length, and set their integers to [1] and [-1].

Select from the Param toolbar > Special > Param Viewer [data tree structure]
- Param Viewer can store in hierarchical structures similar to a branching tree. Data is still stored in lists, but each list now has a 'path', which is a series of indices that describe the position of the data branch inside the tree.

The top Param Viewer shows the data structure graphically, the bottom one as a list. To switch between the two modes - double click on the component.

ivo.a.semerdjiev • isemerdj@iit.edu • digitalearchfab.com/portal/
Choose the Logic toolbar > List > Shift List [Shift]
Create four Shift components, connect them with the Param Viewers, and set their integers to [1] and [-1] as two pairs.
- The Shift components will either move the lists up or down a number of increments dependent on the value of the shift offset.

Select from the Logic toolbar > Tree > Branch [Tree Branch]
- The Branch gives us a specific branch from the data tree. Elements in a list are identified by their index number. This component allows you to have an access to a single item or a group of items from a list by index.
- Connect the two branches with the first Param Viewer group components and renamed them Branch_1 and Branch 2 for an easier access later on in the tutorial.
Do the same procedure for the bottom group of the Param Viewer.

+ Rename the two new branches: Branch_3 and Branch_4
  
  - The reason why our “branches” are connected to the Shift List at the begging is because those list’s are our original data structure of our points.
  - The Param Viewer is giving us a complete control over all item and last but not least it is has a better graphic representation of the stored information.

+ From the Curve toolbar > Primitive > Line
+ Connect it with the first two Shift components which represent the begging of the data structures.
  
  - The line parameter will substitute the curves and will give us the straight profile polylines, which will serve us as the basic frames of the origami surfaces.
Select from the Curve toolbar > Analisys > Evaluate Curve

- Evaluate Curve - will evaluate each curve at a specific parameter. If the length of a single curve is 1 and the parameter is 0.5, it will mark [make a point] at the middle of the curve. If it is 0.25, the point will be at 1/4 of the curve, and so on.

Set the parameter at 0.5, right click on [C] and select [Reparametarize] - Reparameterize changes the curves to have a domain from zero to one.

To get a full access to all middle points of the vertical profile curves - choose the Param toolbar > Special > Param Viewer

Add two more Shift List components with their [1] and [-1] integers
Choose from the Logic toolbar > Tree > Tree Branch

Connect them to the Evaluate Curve component, where are stored our middle points of the vertical profile curves and rename them to Branch_5 and Branch_6.

We have to create horizontal polylines through the original first points of the surface.

The last Param Viewer and the pair of Shift List components we must connect to the old Divide Length parameter in order to gain control over those points.
Assign the last two Branch Tree components to the Shift Lists and the Divide Length parameter - where are all the original points on the surface. Rename them - Branch_7 and Branch_8 for an easier navigation in the following steps.

Select from the Curve toolbar > Primitive > Line. Connect it to Branch_7 and Branch_8 to create the horizontal straight polylines of the origamy surfaces.
From the Curve toolbar > Analysis > Evaluate Curve

- Will give us the mid-points of the newly made horizontal curves.
- Set the parameter number at \([t]\) to 0.5, right click on \([C]\) and select [Reparametrize] to changes the domain of the curve from zero to one.

The next step will be to get the vertical and horizontal polylines through all the mid-points of the curves.
- Choose from the Logic toolbar > List > Shift List - assign their positive and negative number \([1]\) and \([-1]\) and connect them to the Evaluate Curve component.
+ Select the Curve toolbar > Primitive > Line
+ Assigned it to the both Shift List components and we will get the vertical curves through the middle points of the horizontal polylines.

+ From the Curve toolbar > Analyss > Evaluate Curve
  • Set the parameter number at [t] to 0.5, right click on [C] and select [Reparametarize] to changes the domain of the curve from zero to one.
+ By evaluating the middle vertical lines, we would be able to reach their middle points which will serve us as a base list for creating the middle horizontal lines.
  • It is good to rename the component to [Eval / central_point], because these points are going to be very important for the rest of the grasshopper definition.
Choose the Curve toolbar > Primitive > Line
Connect it to the both Shift List components and we will get the horizontal curves through the middle points of the middle vertical lines.
+ By the end of the second part we should have a surface that will contain all of its main vertical and horizontal lines and those lines which are going through the mid point of all of them.

+ Our surface, should also be able to be broken down to its vertex points. They will serve us in the next step, to create those very essential vectors for the foldable surfaces.
+ Select from the Vector toolbar > Vector > Vector 2Pt
  - Vector 2Pt - Create a vector between two points.
+ Create two Vector 2Pt and connect them to the points of Branch_5, Branch_6 and the central_point.
  - They will make the horizontal vectors, which will be guides for the horizontal edges of the panels.

+ Choose the Vector toolbar > Vector > Display
  - Display - gives you the opportunity to see the vectors in the viewport.
+ Create two Display components and link them with the Branch_5, Branch_6 and the two Vector 2Pt parameters.
From the Curve toolbar > Primitive > Line SDL
- Line SDL - Create a line segment defined by start point, tangent and length.
- Create two Line SDL components and connect them to the points of Branch_5, Branch_6 and the two Vector 2Pt.

Select Vector toolbar > Point > Distance
- Assign the Distance parameter to the central point component and the Shift list component with [-1] integer from Branch_7 and Branch_8 part of our definition.
Choose Params toolbar > Special > Panel
+ Connect the Panel component with the Distance one in order to get the list of distance values between the points - vertices of the origami surfaces
+ From the Params toolbar > Special > Number Slider
  * We will rename the slider to [folding_value].

Create from the Scalar toolbar > Operators > Smaller
  * In order to get the maximum possible folding value of the surfaces, the slider number should be always smaller than all the distances between the points of the surfaces.
+ Assign another Panel to [true if A<B] - to see our visual confirmation of the differences between the numbers.
+ Connect the Number slider to the two Line SDL components

+ From the Curve toolbar > Analysis > End Points
  * Attaching End Points parameters to the both Line SDL components will give us an access to the both end points of each vector.
+ Select from the Vector toolbar > Vector > Vector 2Pt
+ Create two Vector 2Pt and connect them to the points of the two Shift Lists of Branch_7 and Branch_8 and the central_point.
  • They will make the vertical vectors, which will be guides for the vertical edges of the panels.

+ Choose the Vector toolbar > Vector > Display
+ Create two Display components and link them with the central_point and the two Vector 2Pt parameters.
→ Select from the Vector toolbar > Vector > Reverse
  • Reverse - Reverse a direction of a vector.
→ We will reverse both vectors, because we would like them to go in an opposite direction.

→ From the Curve toolbar > Primitive > Line SDL
  • Line SDL - Create a line segment defined by start point, tangent and length.
→ Create two Line SDL components and connect them to the both Shift Lists of Branch_7 and Branch_8 and the two Reverse components.
Connect the Number slider to the two Line SDL components.

From the Curve toolbar > Analysis > End Points
- Attaching End Points parameters to the both Line SDL components will give us an access to the both end points of each vector.
Choose the Vector toolbar > Plane > Plane 3Pt
- Plane 3Pt - Create a plane through three points, which will always be parallel to these points.
- Assign [A - Origin point] to the central_point

Assign [B - X direction point] to the Shift List with negative integer [-1] of Divide Lenght component.
Assign (C - Orientation point) to the Shift List with positive integer [1] of Branch_7 and Branch_8.
You will notice that through all central point now we created a plane that is parallel to every individual set of points of each future surface.

Select from the Vector toolbar > Vector > Unit
*Unit - stands out also for unitize vector - turns the selected vector into a unit vector which is always perpendicular to its plane of origin.
+ From the Curve toolbar > Primitive > Line SDL
+ Connect the Line SDL parameter to the central_point and the Unit vector.

+ Choose from the Logic toolbar > Script > F1 - a function of a single variable; {x}
  * Double click on the component to open the Expression Designer;
  * Variable #1 is {x}, so if we type x*1.5 the end result will be a number ‘x’ multiplied by 1.5
Assign the folding value to the function \( x \) and the function to the Line SDL component.

Since this vector is going to be perpendicular to the planes that we created, we have to make this vector larger than the edge's vectors of one foldable surface.

Choose the Vector toolbar > Vector > Display

Connect the Display component to the central_point the Line SDL parameter. On the left, we can see how the vectors in green colour are perpendicular to the planes and longer than the edge vectors.
+ From the Curve toolbar > Analysis > End Points
  • Attaching End Points parameter to the Line SDL component will allow us to 'reach' the points at the both ends of the vectors.
* Select from the Params toolbar > Geometry > Point
  * Assign four point components from Branch_1 to Branch_4 and renamed them with the same number in order for easier navigation.

* Select from the Params toolbar > Geometry > Point
  * Assign two point components to the both End points parameters from Branch_5 and Branch_6.
Select from the Params toolbar > Geometry > Point
• Assign two point components to the both End points parameters from Branch_7 and Branch_8.

Select from the Params toolbar > Geometry > Point
• Assign a point component to the Shift list with positive integer from the Divide length parameter and renamed it to points_9.
Select from the Params toolbar > Geometry > Point
- The last point component will be connected to the End points parameter which is part of the central_point and named it points_10.

Select from the Surface toolbar > Freeform > 4Point Surface
- 4Point Surface - Create a surface connecting three or four corner points.
- Connect the Surface parameter to points_10, points_6, and points_4 and rename the component - surface_A
Select from the Surface toolbar > Freeform > 4Point Surface
- Connect the Surface parameter to points_10, points_5, and points_1 and rename the component - surface_B

Select from the Surface toolbar > Freeform > 4Point Surface
- Connect the Surface parameter to points_10, points_5, and points_2 and rename the component - surface_C
+ Select from the Surface toolbar > Freeform > 4Point Surface
  - Connect the Surface parameter to points_10, points_8, and points_3 and rename the component - surface_D

+ Select from the Surface toolbar > Freeform > 4Point Surface
  - Connect the Surface parameter to points_7, points_10, and points_9 and rename the component - surface_E
• Select from the Surface toolbar > Freeform > 4Point Surface
  • Connect the Surface parameter to points_7, points_10, and points_1 and rename the component - surface_F

• Select from the Surface toolbar > Freeform > 4Point Surface
  • Connect the Surface parameter to points_10, points_8, and points_4 and rename the component - surface_G
  • Connect the Surface parameter to points_10, points_5, and points_3 and rename the component - surface_H
After connecting all 4Point Surfaces, we should have a model which looks like the above one.
If we increase the number value in the slider which is the [folding_value], our origami surfaces will start to fold all together.

If we increase the number value in the slider which is the [number of pframes - density of the surface], our surface is going to become denser with folding surfaces.
Make a point in Rhinoceros near the panelized surface and assign it to a point parameter in Grasshopper.
- Name the point component - attraction_point.

- Select Vector toolbar > Point > Distance
- Connect the Distance parameter to the attraction_point and the central_point.
From the Scalar toolbar > Operators > Division
- Division - Divides two numbers

Create a slider which renamed it [attraction_folding_value]
Choose a panel component and assign it to the Division parameter.
- It will gives us a list with number values which are the division between the distance of the point and the integer value of the slider.
+ Connect the Division component with ALL - Line SDL parameters and the function with one variable.

+ In order our model to work properly, the value of integers in the list to the left should be lower than that to the right.
  • The panel at the bottom should say for all values - True
+ An example of the folded surfaces depending on the attraction point.

+ If we move the point closer to the surfaces, they will respond to it closing themselves.