HOW VIRTUAL BECOMES REAL

Discretise a NURBS surface into a structural Diagrid (G_01)

Creative team
Alberto Pugnale and Stanislav Roudavski

Content developed by
Alberto Pugnale, Alessandro Liuti and Louis Gadd

Supported by the faculty
"Strategic Initiative Funding"

THE UNIVERSITY OF MELBOURNE

msd
Melbourne School of Design

FACULTY OF ARCHITECTURE, BUILDING AND PLANNING
www.msd.unimelb.edu.au

0:00 SLIDES

1 cover slide

- This tutorial explains how to discretise a generic NURBS surface into a Diagrid. The objective is to define a structural grid of beams to design a steel and glass gridshell. At the end of this tutorial, you should be able to model parametrically in Grasshopper the geometry of several well-known projects

0:18 4 slides with examples

For example:

- You could replicate the first gridshell ever built, that is Shukhov's building for the plate rolling workshop in Russia. More details on its geometry, structure and construction can be found in the paper listed in this slide
- You could redraw the gridshell roof of the DG Bank court in Berlin
- You could redesign parametrically the roof of King's Cross Station in London
- The Smithsonian courtyard by Foster and Partners
- Or even some of Nervi's RC structures, as the ribs are based on the principle of the Michell frames. The sport Hall in Rome and Australia Square in Sydney are two examples of this kind.

0:50 1 slide with form-found shape, Hippo House and British Museum

- Just a couple of clarifications before starting:
    - The code described in this tutorial will work on any NURBS surface you feed to the Grasshopper definition
    - However, to generate a form-resistant geometry first, follow the instructions provided in tutorial G_01.2
    - Remember that projects such as the Hippo house in Berlin are not discretised following this definition. They are translational surfaces meant to generate planar quads. They are explained in a separate video
    - Special patterns, such as the one of the British Museum gridshell, or of the National Maritime museum gridshell, cannot be generated using this tutorial. They'll be described in a separate tutorial

Now let's start!

1:25 GRASSHOPPER

- First, let's open the Rhino **Options** panel and make sure the **Units** are set to **Metres**. This is required in order to feed this code results to other plugins, such as Karamba
- Now, you can draw the surface you will work on directly in Rhino, four-point surface. Remember that you are always drawing in metres

- You can import the surface in Grasshopper by **Right Clicking** on the **Surface** component and choosing **Set one surface**. Otherwise, you can use a predefined parametric surface, such as the one resulting from video tutorial G01.2

- Now, to discretise the surface into quads, you have to use two components. First, **DivideDomain**, which divides the parametric domain of the surface (UV) into parts
- Then, you have to use **Isotrim** to geometrically trim the surface in a set, or list, of sub-surfaces

2:20 SLIDE

- Now, before generating the Diagrid, have a look at this slide
- In the bottom part, both diagonals are traced for each and every parametric quad. You can easily access the quad vertices with the **Deconstruct Brep** component and connect with lines the vertices with index 0-2 and 1-3. In this way, however, the two diagonals won't intersect for a doubly-curved surface, but only when the faces are planar
- The correct method is shown instead in the upper part of the slide. You need to select every second face of the grid and connect only the vertices 0-2. Then, by inverting the face selection, you generate the second order of diagonals, connecting the vertices 1-3
- In this way, all the gridshell beams are connected at the nodes
- Now let's go back to Grasshopper

3:04 GRASSHOPPER

You can create a chessboard in Grasshopper by using the **CullPattern** component on the list that **Isotrim** provides:

  - To filter this list, you need to apply a **True/False pattern** to it. You can feed it into the **CullPattern** command
  - It's important to remember to **Right Click** on the **Panel** and **Uncheck Multiline Data** – otherwise it won't work
  - The first thing you notice is that, to get the correct chessboard pattern, you need to change the **V-count slider type** to **Odd.** So let's add some **Sliders**

- Now, to obtain the reverse quads set, you just need to **Copy/Paste** this **CullPattern**, then **Right Click** on the **Pattern input label** and select **Inverse**

- You now need to connect the vertices of the faces in the correct order to get the Diagrid
- By using **DeconstructBrep** you can retrieve a set of nested lists containing the 4 vertices for each face (here in the V output)
  - "nested list" means "list made by other lists" and, in this case, it means that you have an sorted list of vertices with indexes 0-1-2-3 for each face
  - With **ListItem** you can access the point set with index 0 individually
  - And by **Zooming In**, and **Clicking** on the **+** icon three times, you will add some points with index 1, 2, 3 to the outputs list

- Now, to draw the first set of lines of the Diagrid, you will connect with a **Line** component point set 1 to point set 3
  - you will use a the **Join** component to get continuous polylines from these segments, which can then be unrolled for fabrication as shown in separate video tutorials, F01 and F02
- The same applies for the second set of lines of the Diagrid, which will connect points set 0 to point set 2

- Now you have to draw the edge beams and diagonal bracing
- For a first direction of diagonal bracing, you will draw lines between points set 0 and point set 3
  - Here you notice that, since you are working on a scattered chessboard pattern, you need to draw lines between sets 0 and 3 also for the second set of faces
  - By using the **Join** component on the two sets of segments, you'll get continuous polylines, and remember to **Flatten** the input for that
  - You might notice the last line of bracing is missing
    - this is not relevant since the edge beams will override all the perimetric lines
- if you want to draw the bracing in the other direction, just connect point sets 2 and 3

- To draw the edge beams, you need to feed the "Isotrimmed" surfaces into **BrepJoin.** Then, you need to use **BrepEdges** to retrieve the naked edges
    - Now, it might happen that the edges are 2nd or 3rd degree curves, and you need to "rebuild" these into 1st degree lines. So let's use the **Rebuild** component and:
        - Set the **Degree** to 1
        - And **Number of Control Points** to 2

Now you see there are separate outputs for:
- the main members in the first and second direction
- the diagonal members in the "u" and "v" directions
- the edge beams

From these outputs it is now possible to:
- either proceed to optimise the structure – refer to tutorial G01.2
- or fabricate the elements – refer to tutorials F01 or F02

6:58 SLIDE
- This is the end of the tutorial. Now, your Grasshopper definition should look like this:
    - In the first part, a parametric surface is divided into a list of sub-surfaces
    - In the middle part, every second sub-surface is selected, and four lists of vertices with index 0-1-2 and 3 are extracted
    - In the third part, the Diagrid beams are generated, including diagonal bracing. Remember that these elements are parametrised in this way to simplify fabrication and structural analysis
    - In the bottom part, the edge beams are finally added

7:29 END OF TUTORIAL G01