# Rendering ribbons in babylons js.

The problem: when doing the 3D rendering of a figure, we need to slightly deform and stretch some faces. The simplest way to achieve that is by splitting a face into triangles.

However in the fucking piece of shit that babylon js is, if we draw each shape as a separate mesh (and because it’s a fucking retarded library doing this is already complicated enough) the borders will be displayed as a line.

The problem with this is that that line looks too much like a crease that every origami figure is supposed to have, which makes it very confusing and unnatural looking.

I try to find a solution to render a mesh consisting of two or more faces with no visible borders between them.

The three d data structure looks like this:

blintzThreeD = JSON.parse('{"facesColorOne":[

{"perimeter":[{"x":1.0,"y":0.0,"z":0.0,"coords":[1.0,0.0,0.0]},{"x":0.0,"y":1.0,"z":0.0,"coords":[0.0,1.0,0.0]},{"x":0.003942649342761229,"y":0.003942649342761229,"z":-0.08862397936135699,"coords":[0.003942649342761229,0.003942649342761229,-0.08862397936135699]}],

"path1":[

{"x":1.0,"y":0.0,"z":0.0,"coords":[1.0,0.0,0.0]},

{"x":0.0,"y":1.0,"z":0.0,"coords":[0.0,1.0,0.0]}],

"path2":[

{"x":0.003942649342761229,"y":0.003942649342761229,"z":-0.08862397936135699,"coords":[0.003942649342761229,0.003942649342761229,-0.08862397936135699]},

{"x":0.003942649342761229,"y":0.003942649342761229,"z":-0.08862397936135699,"coords":[0.003942649342761229,0.003942649342761229,-0.08862397936135699]}]},

{"perimeter":…

And this is the call that creates the ribbon, where the ribbon geometry is is extracted from each structure as above starting with “perimeter”:

let ribbonGeometry = [[{"x":0.9, "y":0.3, "z":0.1}, {"x": 0, "y": 1, "z": 0}], [{"x": 1, "y": 0, "z": 0}, {"x": 1, "y": 0, "z": 0}]]

let ribbon = BABYLON.Mesh.CreateRibbon("ribbon", ribbonGeometry, false, false, 0, scene, false, BABYLON.Mesh.FRONTSIDE);

The version of babylon I’m using is the latest stable one: 2.2 at the time of writing.

One possibility would be what in babylon is called “edge rendering”. Which is a call like this

ribbon.enableEdgesRendering(0.95);

Unsurprisingly, this doesn’t work. Because babylon is a piece of shit or because it has another usage (ie for spheres or cubes or some other shape)? I still haven’t figured that out.

This is the forum where I could ask for help: <http://www.html5gamedevs.com/forum/28-questions-answers/>

A bunch of triangles can be simulated with paths. For example, rendering this three paths will result in two triangles:

**var** path2;  
**var** path3;  
**var** path4;  
path2 = [];  
path3 = [];  
path4 = [];  
  
path2.push( **new** BABYLON.Vector3(0, 1, 0) );  
path3.push( **new** BABYLON.Vector3(-1, -2, -1) );  
path4.push( **new** BABYLON.Vector3(2, -4, 1) );  
  
path2.push( **new** BABYLON.Vector3(0, 1, 0) );  
path3.push( **new** BABYLON.Vector3(0, -2, -1) );  
path4.push( **new** BABYLON.Vector3(2, -4, 1) );  
  
path2.push( **new** BABYLON.Vector3(0, 1, 0) );  
path3.push( **new** BABYLON.Vector3(1, -2, -1) );  
path4.push( **new** BABYLON.Vector3(2, -4, 1) );  
  
**return** [path2, path3, path4];

Then you can disableEdgeRendering.

Naturally the face doesn’t need to be broken in triangles. Using paths, quadrilaterals are also possible.

This solution, while feasible, requires a big amount of work, such as new type of face where the points don’t share the same plane, for deformed quadrilaterals, passing a different data structure for paths that render more than one face. Vertices and edges that are to be ignored in the front end when picking a vertex and edge as a fold end and so on.

The solution for the moment will be to use always straight faces (faces that remain undeformed and not broken into triangles). 3D rendering is still possible with such constraint and it makes life much easier.