Implementation of the paper
“Enrichment Textures for Detailed Cutting of Shells (SIGGRAPH 2009)”

Introduction
This project simulates a physical process of tearing a paper using the “Enrichment Texture” introduced in this paper. The values in this texture are continuous in its domain except the location of the crack. Enrichment texture is bound to the polynomial surface to determine how vertices move in 3D space by accessing to this texture and it is also used to simulate cutting details on the crack just using a coarse mesh.

Enrichment Texture
The left figure shows the color representation of enrichment texture in a certain time step (we have hundreds of different enrichment textures for each time step to simulate the complete tearing process). The range of the value in the texture is from -1 to 1 and is mapped to the color space from pure red to pure blue. The enrichment texture is obtained by solving a Laplace equation with Neumann boundary condition (at the boundary of the quad domain) and Dirichlet boundary condition (at the crack). The result is pre-solved in MATLAB and is imported to Direct3D to generate textures at the initialization stage.

I also tried to implement the Jacobi iteration method to solve the equation in real time using Compute Shader of DX11 to let each thread deal with each texel in parallel.
However, the convergence of the function is so slow that a vast number of loops is needed to obtain a decent result (more than 5000 loops for solving on a 100×100 texture), which makes the rendering speed far from real-time performance (less than 2 FPS for solving on the 100×100 texture). The low performance makes me give up using this method.

**Rendering the surface and crack**

Since the paper is represented as a polynomial surface defined by several basis functions and control points, DX11 Tessellation Engine is used to build the surface. The set of control points used is a combination of two sets of control points, one of which is multiplied by the enrichment function, whose values are stored in the enrichment texture, to generate surface discontinuity along the crack.

Vertices are evaluated in the Domain Shader. However, simple evaluation cannot cut off the divided crack because triangles are connected to each other. The figure below shows those stretched triangles on the crack.

![Stretched triangles on the crack](image1)

The solution is to use Geometry Shader to duplicate triangles on the crack and re-evaluate some of their vertices to make the two set of triangles fit the two parts of surface divided by the crack. Then, for rendering the detailed cutting edges, in Pixel Shader, we need to access to the enrichment texture to discard pixels in the cut-away part within the two set of triangles. The following two figures well explain the process.

![Geometry Shader solution](image2)

The figure below shows the final result. You can see that the triangles on the crack are not complete due to the throwing of pixels according to the enrichment texture.
Please note that the resolution of the enrichment texture must be greater than resolution of the mesh, otherwise we cannot demonstrate more details on the crack compared with the relatively coarse mesh (in my implementation, a 200×200 enrichment texture and a 50×50 quad mesh is used).