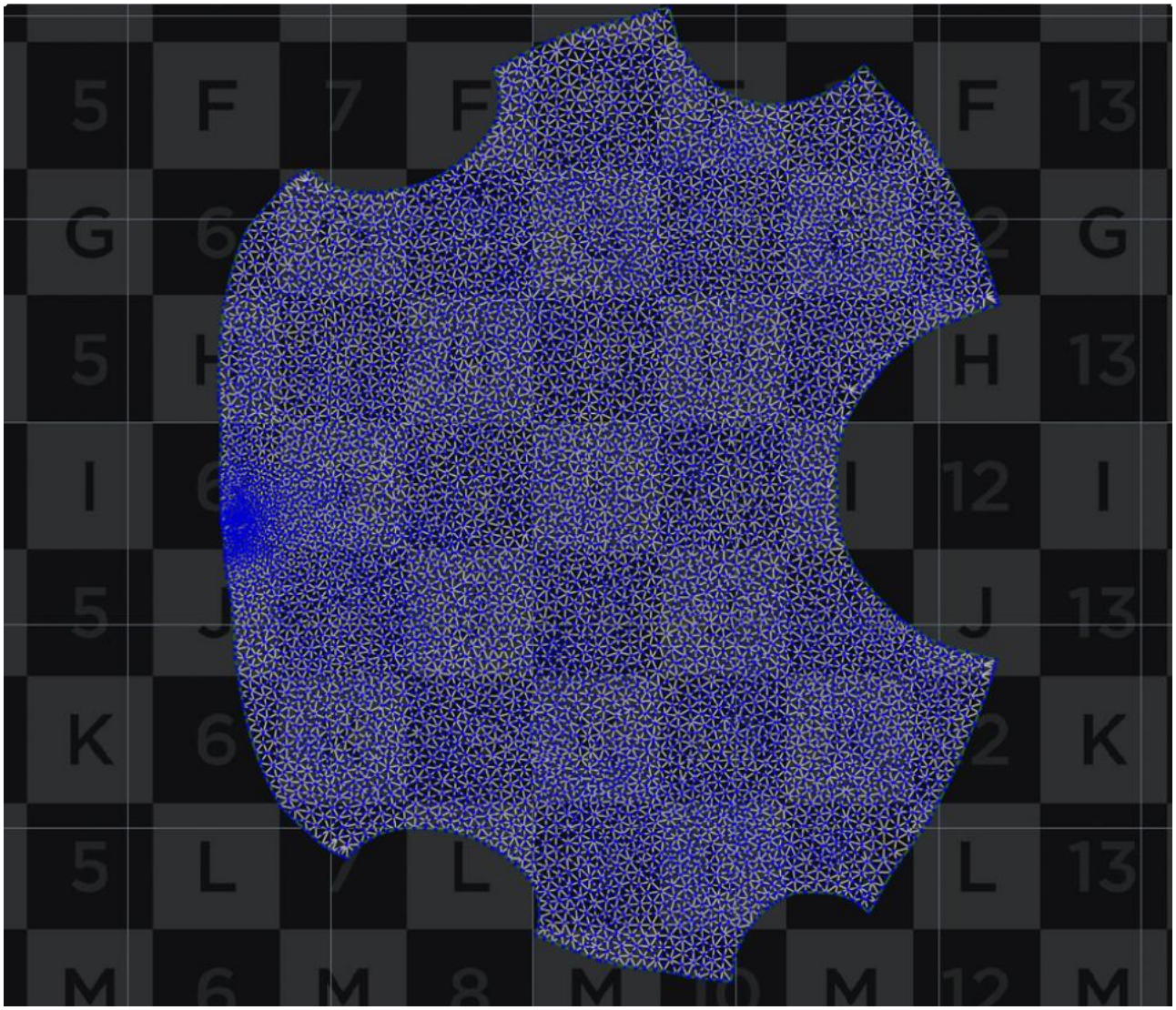
Basic UV Lookup Data Structure

Brief introduction

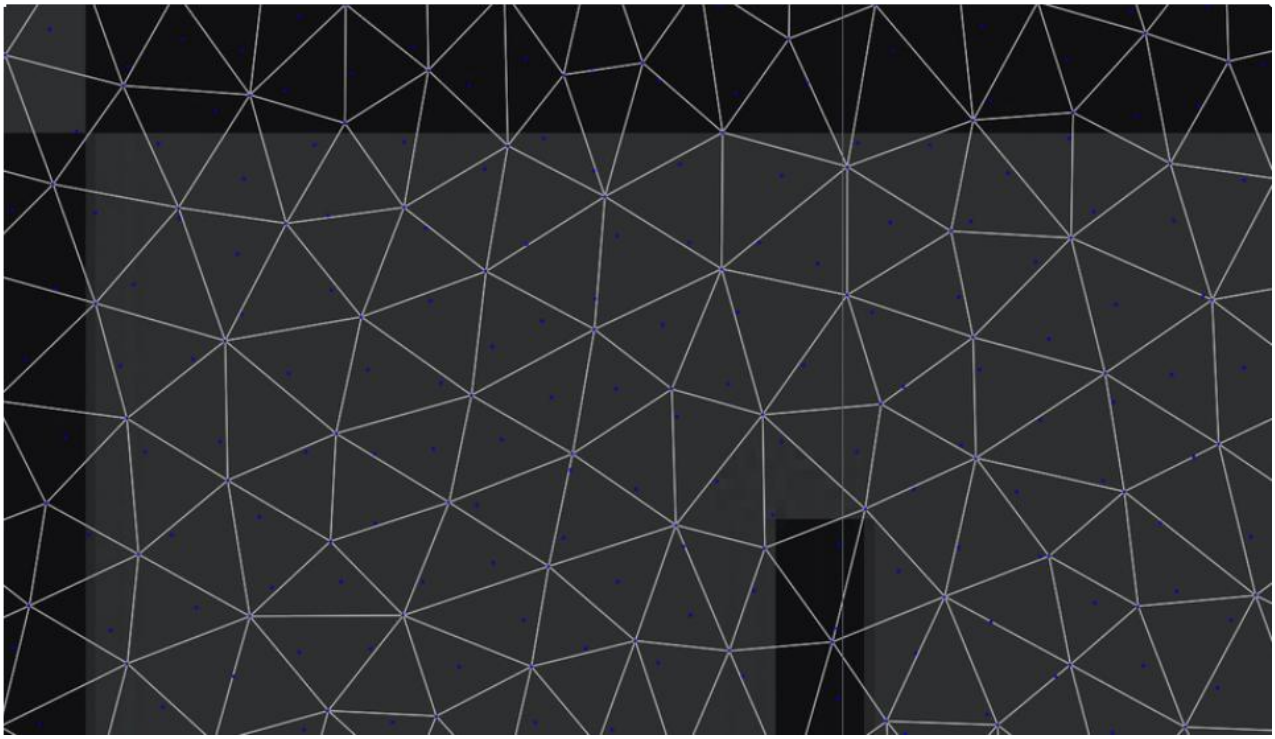
This project is to create a node that can find the closest triangle face for a point on the mesh. It returns the primitive number of the point'sclosest triangle and the barycentric coordinates of the point in the triangle.

The existing houdini function xyzdist() can achieve this goal, but it's for both 2D and 3D object. In our case, we only operate on the 2D uv coordinate, so the existing one may not be efficient enough. This project is about to create a node only for 2D object and be more efficient.

Here is the uv coordinate:

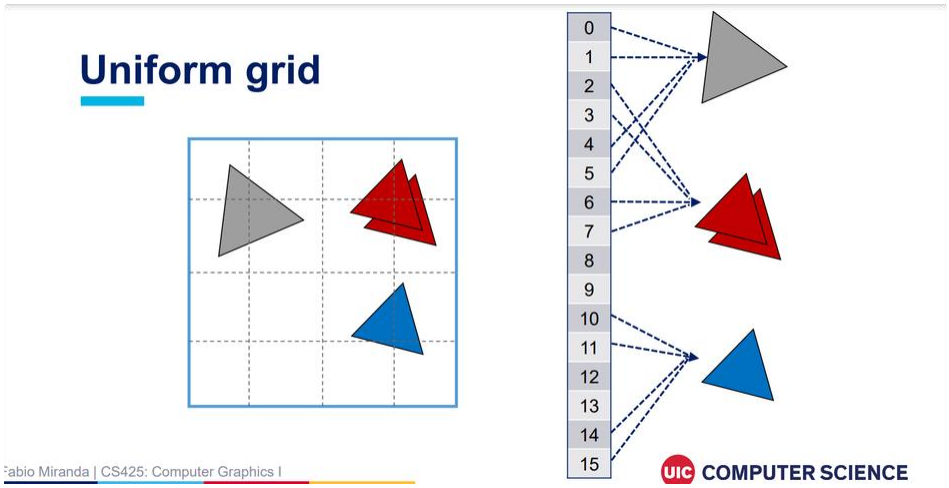


After zooming in, we can see the scatter point



Algorithm

Store the triangles in a grid so that when it comes to find which scatter point is in a triangle, just loop all the triangles in the cell that thescatter point locates in.



Uniform grid: construction and query

- Array of 3D voxels
  - Each voxel: list of pointers to colliding objects.
- Indexing function:
  - 3D point → cell index (constant time!)
- Construction:
  - Initialize cells for grid with size $w * h$
  - For each object $\mathrm{p}(x, y)$:
    - Compute grid cell using (x, y).
    - Store p in cell.
- Query:
  - For query rectangle $(x_1, y_1) \times (x_2, y_2)$:
    - Compute subgrid for (x_1, y_1) and (x_2, y_2).
    - For all cells inside subgrid, report all objects.
    - For all cells on the border of the subgrid, test objects against rectangle.

- Get the uv coordinate of scatter points and triangles from Houdini
- Create a structure to store triangles' primitive number in the mesh
- Loop on the cell to find the closest triangle to the scatter point