METU **Department of Computer Engineering** **Spring 2026**

CENG 589—Digital Geometry Processing

ASSIGNMENT II March 23, 2026
Instructor: Y. Sahillioğlu 2 Weeks

Mesh Kernel Computation and Clustering (start early!)

Your task is to extract 3D guarding points from watertight meshes and group them into clusters based on visibility.

Bounding Box and Sampling {*20 points*} Create a bounding box around the given mesh and uniformly sample this bounding box to form a rectangular grid with a chosen resolution. Mark the sample points falling inside the mesh. Make sure your bounding box grid resolution is high enough so that the samples inside the mesh are not sparse. There exists several inside/outside tests for polygons. For this assignment, it is sufficient to use the following simple approach:

Algorithm 1 Inside/Outside Test

```
Sample bbox
for each grid point g do
    if (g − a) · n > 0 then
        remove g                                  ▷ is outside
    end if
end for
```

where n is the *unit* normal of the *closest triangle* to grid sample g, and a is the point connecting g to the face (e.g. midpoint of the triangle). *Be careful with the vector direction*, it should be from the face to the sample g not vice versa. The upcoming sections will work with the interior samples; so you may trim the samples falling outside of the mesh. You may also discard the samples that are too close to surface to avoid errors due to numerical instability. Visualize this grid sampling with inside/outside test and include it in your report.

Kernel Computation {*30 points*} A polygon is called star-shaped if there exists at least one point that can see every other point within the polygon (recall the art gallery problem from lecture slides 02). Kernel is the region encapsulating the guarding points of the star-shaped polygon. In 3D, kernel is computed by intersecting the half-planes defined by faces. Figure 1a illustrates this intersection for 2D polygons. In this assignment, we will compute guarding points with a different approach. In Figure 1b, observe that the face normal and the vector connecting guarding point to the face always form a wide angle. Count the number of faces where a grid sample makes a wide angle based on its dot product:

Algorithm 2 Computing Visibility Parity of Grid Samples

```
for each grid point g do
    if (g − a) · n < 0 then
        g.parity ← g.parity + 1
    end if
end for
```

Note that if the given mesh is star-shaped, the samples with the highest parity scores will lie inside the kernel of this mesh. These samples should all share the same parity score as they are able to see all the faces. Add a button to visualize only the highest parity samples, and include a screen recording in your deliverables. If the mesh is star-shaped, the displayed samples should correspond to the mesh kernel.

Internal Clustering {*50 points*} Since most of the meshes are not star-shaped, we won't have a kernel directly, but instead we will group the samples based on their parity to decompose the mesh.

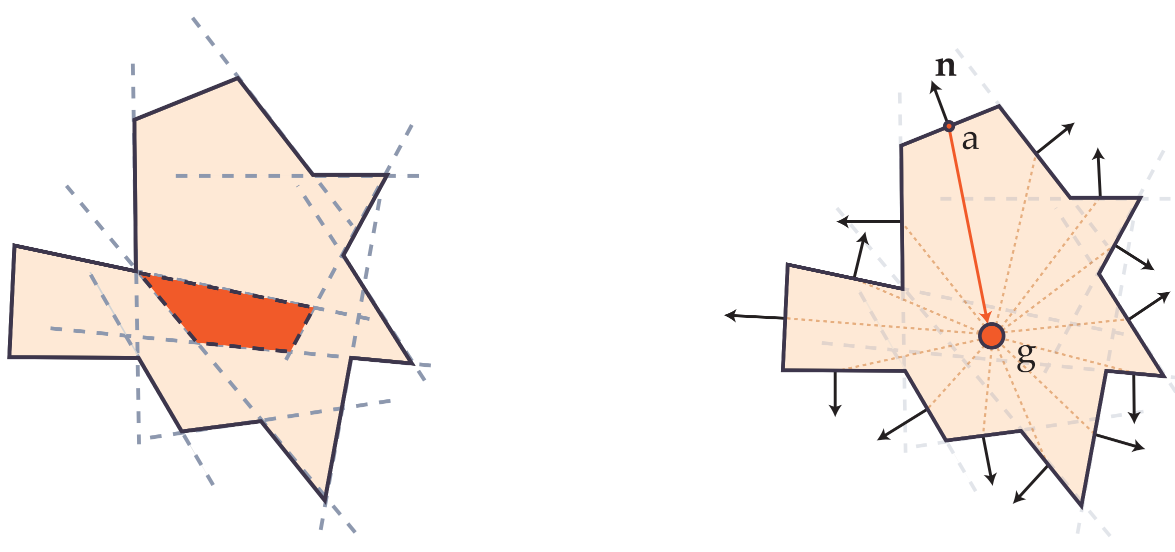
Use Breadth-First-Search (BFS) to traverse the grid samples and cluster them with respect to their parity scores computed in the previous section. Clusters should contain the samples sharing the same parity. Start from a seed point g, use BFS until you obtain the cluster and continue this loop until you obtain a set of clusters in which each sample belongs one of those clusters:

Algorithm 3 Clustering Samples Based on Parity

```
loop                                            ▷ If convex, loop just once
    Start BFS from g with parity p
    for each neighbor n with same parity p do
        put n to same cluster with g
    end for
end loop                                        ▷ Until all grid samples are visited
```

For BFS traversal, use a rectangular 26-neighborhood where in each lattice, there are $3 \times 3 \times 3$ vertices, and excluding the sample g itself leaves 26 neighbors (see Figure 5 in this link). Color the mesh faces based on closest grid point's cluster. Provide user interaction that allows selecting a particular cluster, which highlights the corresponding faces and makes the other clusters' points invisible. Include screen recordings of these tasks in your deliverables with appropriate file names.

Experiment with different heuristics for choosing the initial seed point of the BFS traversal. Observe and report their effects on the outcome. A simple heuristic is to start with a point with the highest parity, and once its BFS traversal is finished, continue with the remaining points in the same way.



(a) (b)

Figure 1: Star-shaped polygon with highlighted kernel area (a) and a guarding point within kernel (b). Notice all the vectors connecting the normals to the guarding point make a wide angle, where the dot product of these vectors returns a negative scalar.

Submission This assignment constitutes 20% of your final grade. Good luck. Test with the meshes in ~ys/meshes2.zip and send to ys@ceng.metu.edu.tr your code, executable, and a report containing screenshots as well as your interesting observations.