Step1: 文本

中度可信度描述已自动生成

1. Technical debt: generator using the structure that provide us, not a ADT type, have limitation and it is immutable. For visualizer, not able to parsing other type like polygon
2. This immutable structure is not open for extension, it not good for use to create some feature. Such draw line need found the start and end id of vertex and found the it in the vertex list, it is not efficient

Step2:

The starter code has some flaws as it is just a simple “recording” structure used to store a given mesh into a file. As a result, we fix it by several steps below.

Firstly, the starter code is not structed properly since everything is in one class. So we divide the code into separate class by various objects like vertex,segment,polygon, which makes the code more modular and organized. It also makes the code more readable.

Secondly, the starter code only has a function of recording. We add some method for adding vertex, segment, polygon. We also add a transform method to make mesh we designed into the mesh that io library provided.

At last, we added new features like adding usedby to segments, which allows segments to remember which polygons use these segments. This feature could help us find neighbours of polygons easily.

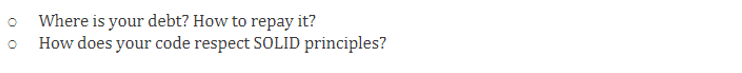
To ensure the invariants support the requests from the user, we added some checks. For vertex, the code checks whether a vertex with the same coordinates already exists in the mesh. If so, it returns the existing vertex instead of creating a new one. Similarly, before adding a new segment, the code checks whether a segment with the same two vertices already exists in the mesh. If so, it returns the existing segment instead of creating a new one. When adding a segment to a polygon, the code updates the "usedBy" attribute of the segment to include the polygon's ID, thus ensuring that the segment remembers which polygon it belongs to.

Testing helped me identify and fix bugs and logical errors in the code, such as incorrect indexing and null references, which would have otherwise caused the code to fail or produce incorrect results.

Step3:

手机屏幕截图

描述已自动生成



o Did you have to modify your visualizer for this part?

we have to modify our visualizer for this part because debug mode just is the other way to visualize the data we store in the io, so we have to modify our visualizer to display the several modes.

o How did you encapsulate the complexity of geometrical computations in your  
design?

During this project, we created ADT for mesh, polygon, segment and vertex separately, which encapsulates the complexity of geometrical computation, so users do not have to deal with those processes. For example, users can only input the number of vertexes they want or, and other properties, such as thickness and the number of relaxations for the graph to easily control this program.

o If one has to generate another kind of mesh (e.g., triangular tessellation instead of  
square based), how would your design support it?

Use can use the “-h” or "-help" to see the way that how to generate the mesh they want to generate( For the triangular tessellation, "-k bonus")

o Where is your debt? How to repay it?

Currently, our debt is the encapsulation problem of the random mesh generator, we are considering designing a more appropriate structure for future use. for paying this debt, we will learn and review the knowledge connect to encapsulation, and do more practice to accumulate experiences.

o How does your code respect SOLID principles?

Considering the product, we come up with, we are trying our best to follow the SOLID principles. For example, most of the classes we encapsulated have and only have the single-use, and most of our classes do not depend on anything, additionally, we only proved to users the interface that they need.