Universidad del Rosario | Escuela de Ingeniería, Ciencia y Tecnología | MACC

Computational and Differential Geometry
Homework 2

Professor: Nicolás Avilán Vargas, Ph.D.

Indicaciones

1. Fecha de entrega: 8 de octubre de 2023 hasta las 11:55 pm.
2. Único medio de entrega e-aulas.
3. Formato de entrega: Archivo **.ipynb** con códigos en python, descripciones de códigos y procesos, y respuestas a las preguntas. **Ni .py, ni .zip**.
4. Solo es permitido el uso de librerías "básicas" (numpy, matplotlib, seaborn, pandas, etc). En ningún caso será válida la solución lograda, total o parcialmente, por el uso de una librería especializada para resolver problemas de geometría computacional.
5. La tarea **debe** realizarse **individualmente**.
6. Cualquier tipo de fraude o plagio es causa de anulación directa de la evaluación y correspondiente proceso disciplinario.
7. Las entregas están sujetas a herramientas automatizadas de detección de plagio en códigos.
8. Las tareas no entregadas antes de la hora indicada tendrán calificación de 0.0.

Support each piece of code with a thorough explanation of its methods, techniques, functions, and tricks. Reference your search source (papers, books, tutorials, websites, etc.). Add any necessary bibliographical references or links.
There is an attached file with a set of ordered points in order to develop the suggested activities (datos.txt). As seen in Figure 1, these points are the vertices of the simple polygon that will be triangulated.

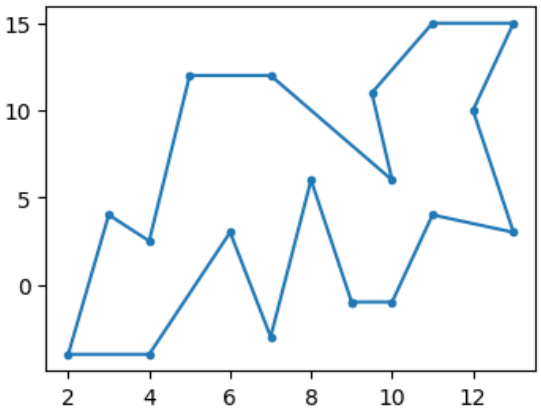

Figure 1: Simple polygon to be triangulated.

Any sweep line algorithm is an option. Please cite your source if you don't use your own code.

1. Write down a code able to identify if the points are ordered in a counterclockwise order. The code should inverse the order of points so that they are counterclockwise ordered if they are in the opposite order. **Explain and implement your procedure**.

2. Write down a code able to read the ordered points and create a doubly-connected edge list for the simple polygon. **Print the doubly-connected edge list related to the simple polygon.**

3. Implement an algorithm able to split the given polygon into y-monotone polygons. Give your answer in terms of doubly-connected edge lists. **Plot the polygon split into y-monotone polygons**.

4. Implement the triangulation procedure and apply it to each y-monotone polygon. Give your answer in terms of a doubly-connected edge list. **Plot the triangulated polygon**.

5. Find the vertices from which a minimum number of cameras could be used to guard the entire polygon. **Identify and plot in the polygon the places where the cameras should be localized**.

Submit:
Upload to the platform an **.ipynb** file with answers, codes, descriptions and plots.