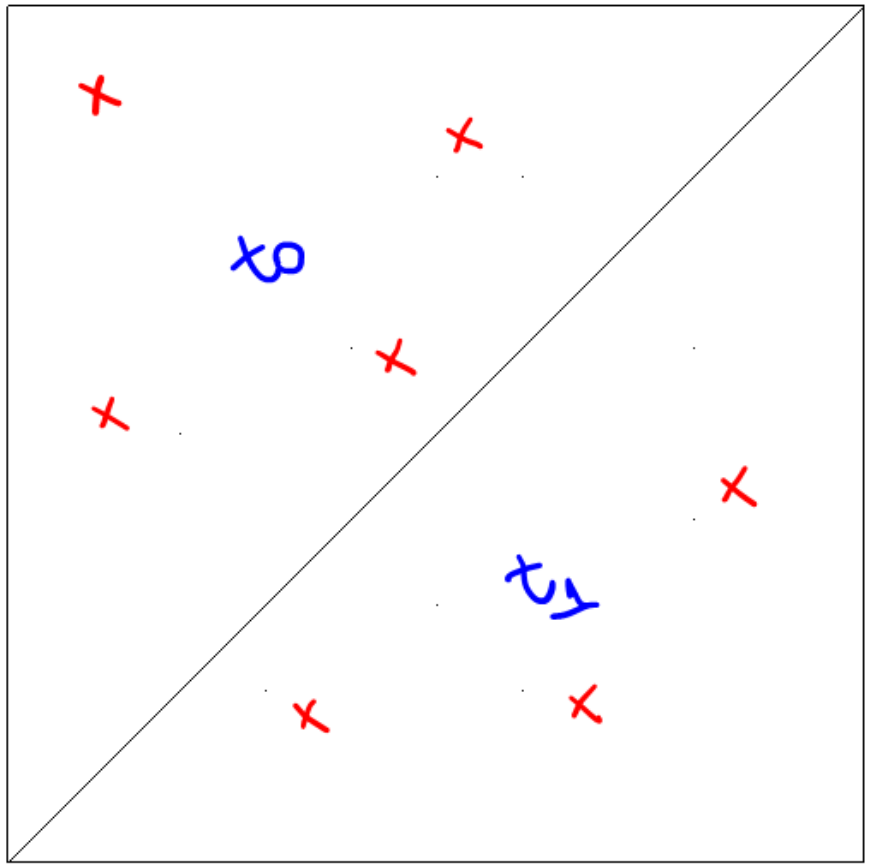
Triangulación de Delaunay

Benjamín Mellado
Profesora María Cecilia Rivara
CC5501-Mallas Geométricas

Creación de la malla

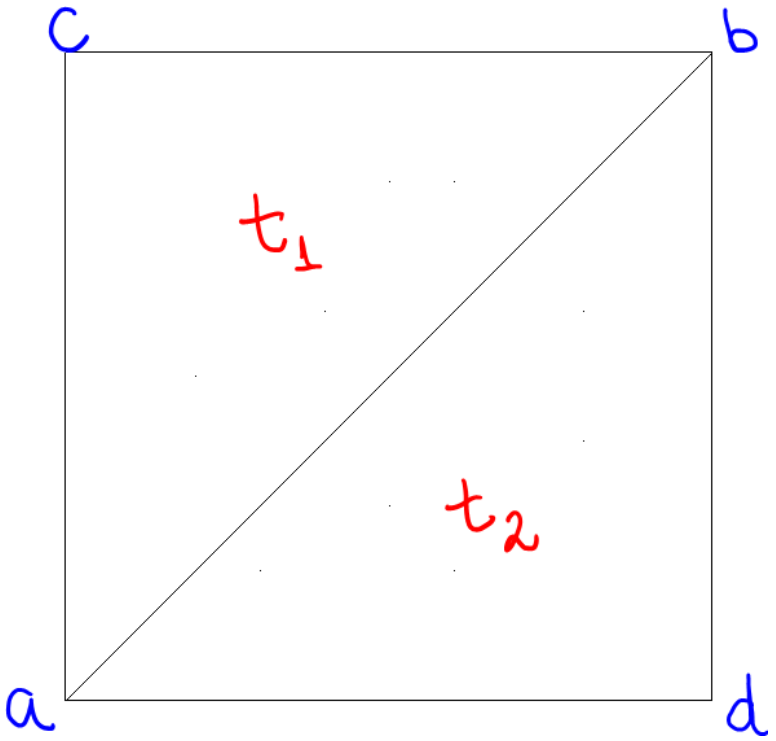


- Se crea un cuadrilátero alrededor de los puntos con un ancho de margen.
- Se inicia la triangulación trazando una diagonal en el cuadrilátero.

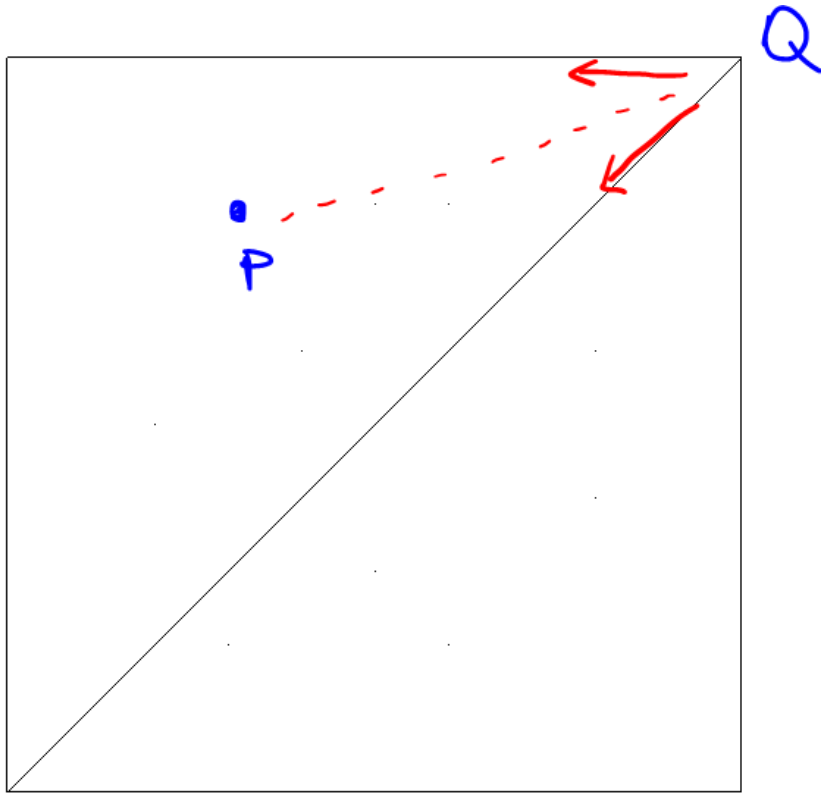
Creación de estructura

- La estructura es una matriz que en la primera columna contiene los triángulos y en la segunda columna sus referencias.
- Las referencias contienen la posición del triángulo opuesto al vértice.

Triangulos (p1,p2,p3)	Lista de referencias
T1 (a,b,c)	[None,None,1]
T2 (a,d,b)	[None,0,None]

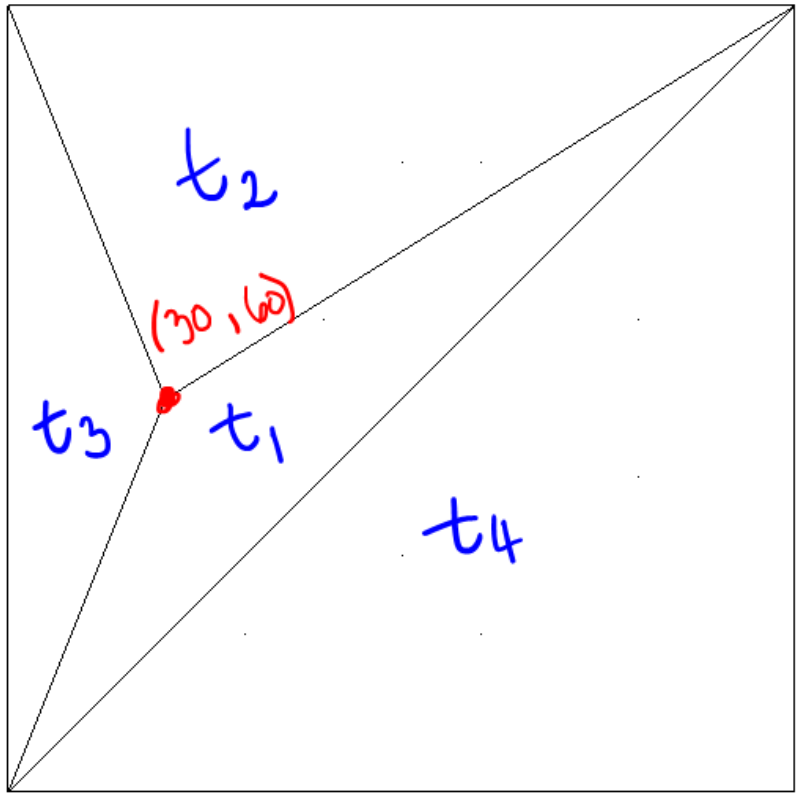


Inserción de punto



- Se fija un punto Q, y se selecciona el triángulo que “envuelve al punto Q”

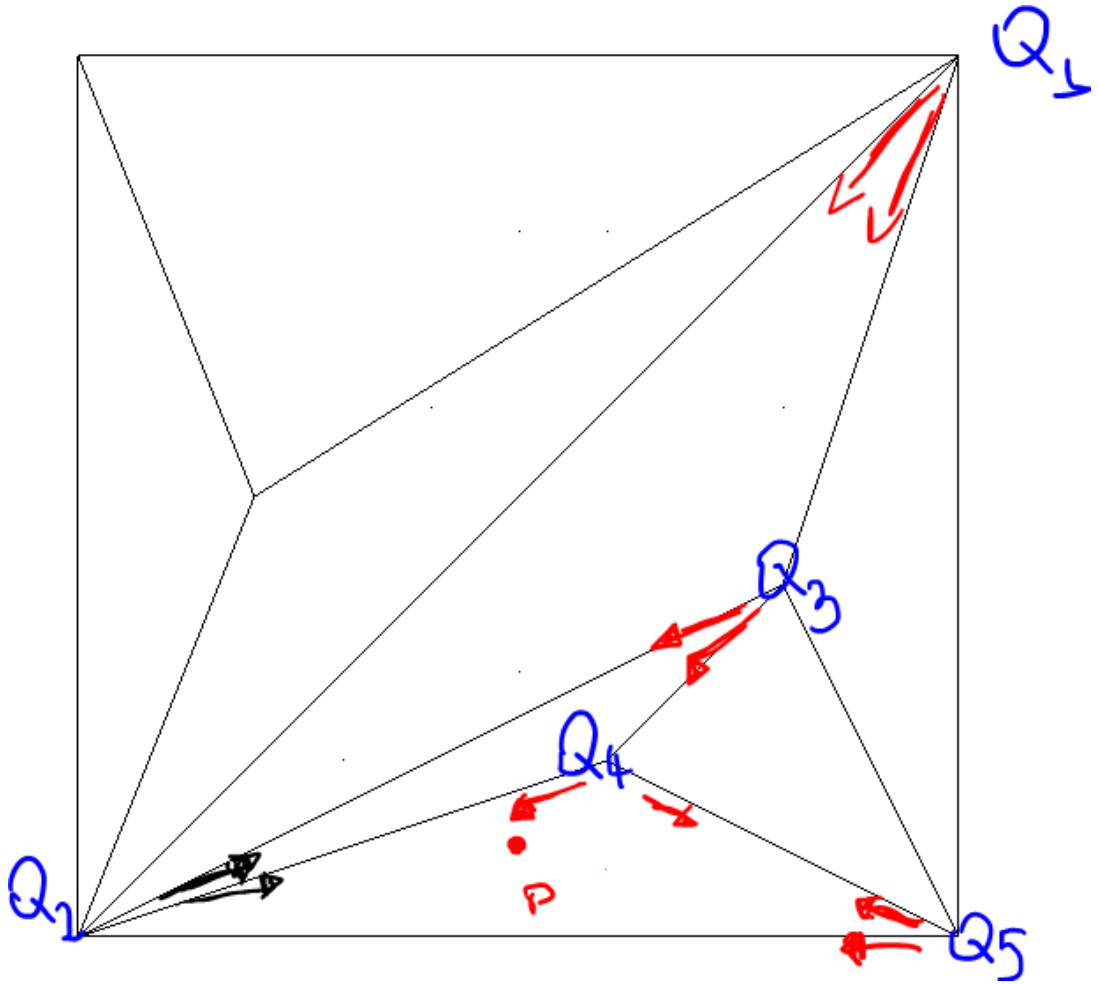
Inserción de punto



- Se insertan tres nuevos triángulos y se elimina al que contiene a los tres.
- Los nuevos triángulos se referencian entre sí y también con la referencia del triángulo antiguo.

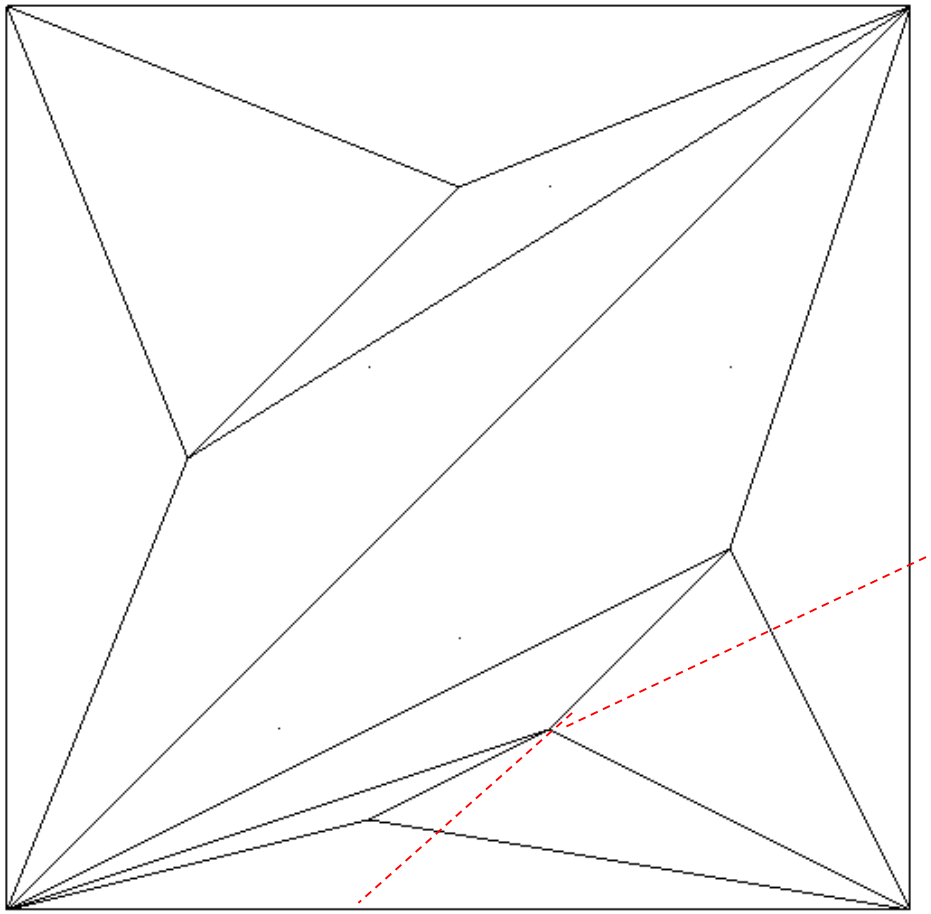
Triángulos (p1,p2,p3)	Lista de referencias
T1 nuevo	[1,2,3]
T2	[None,0,None]
T2 nuevo	[None,2,0]
T3 nuevo	[1,None,0]

Búsqueda de punto



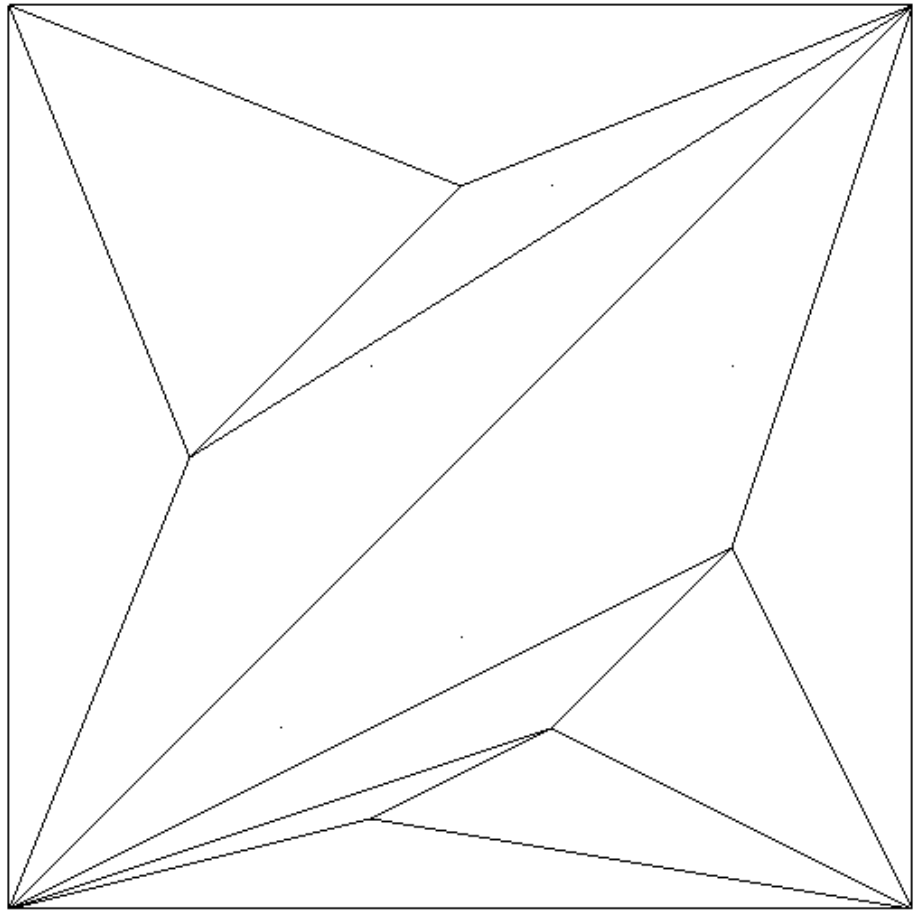
- Se construye un camino con los triángulos que contienen al punto, el Q va variando a medida que se avanza.

Caso crítico de búsqueda



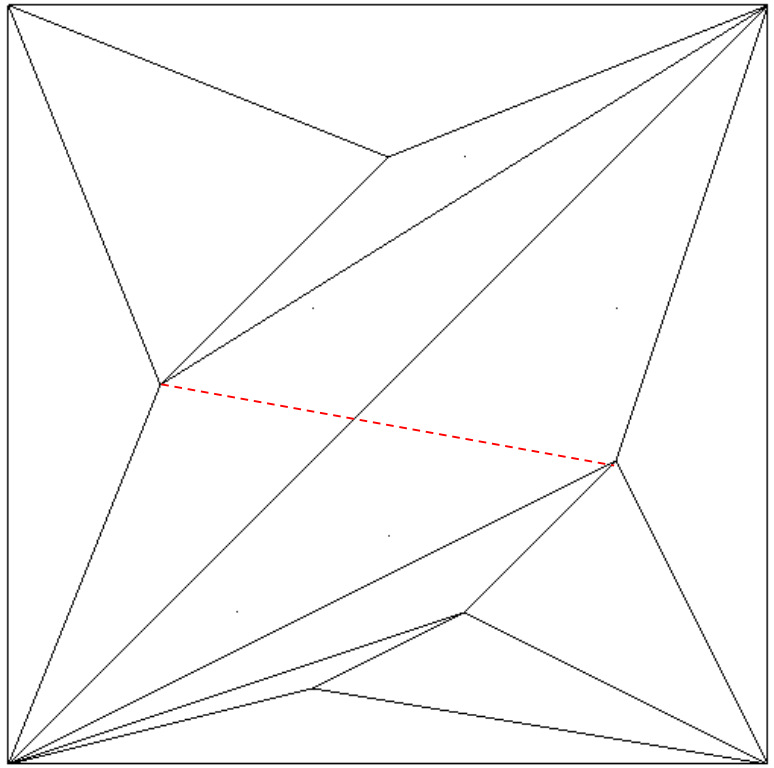
- Zona sin ser cubierta para continuar la búsqueda en caso de que el punto se encuentre allí.
- Se soluciona buscando en el triángulo adyacente a cualquiera de los dos puntos Q1 o Q2 y se procede recursivamente con la búsqueda.

Malla de triángulos sin Delaunay



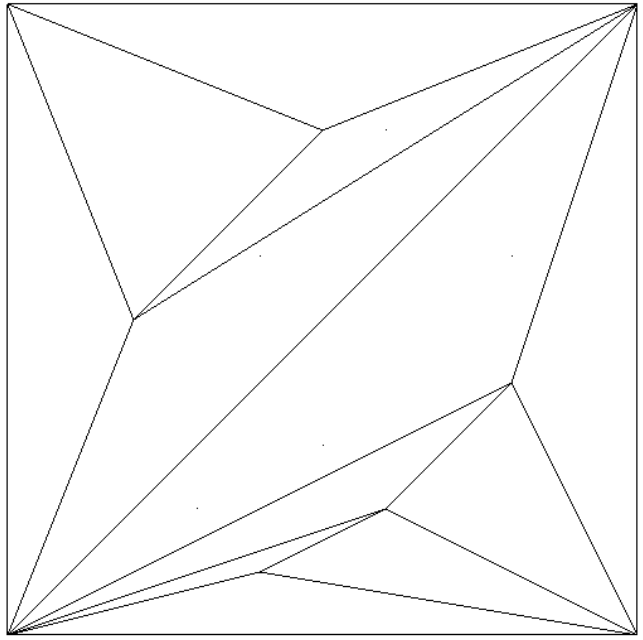
- Se genera una malla de triángulos con los puntos insertados.
- Los puntos se insertan sobre los triángulos existentes

Aplicación de algoritmo Delaunay

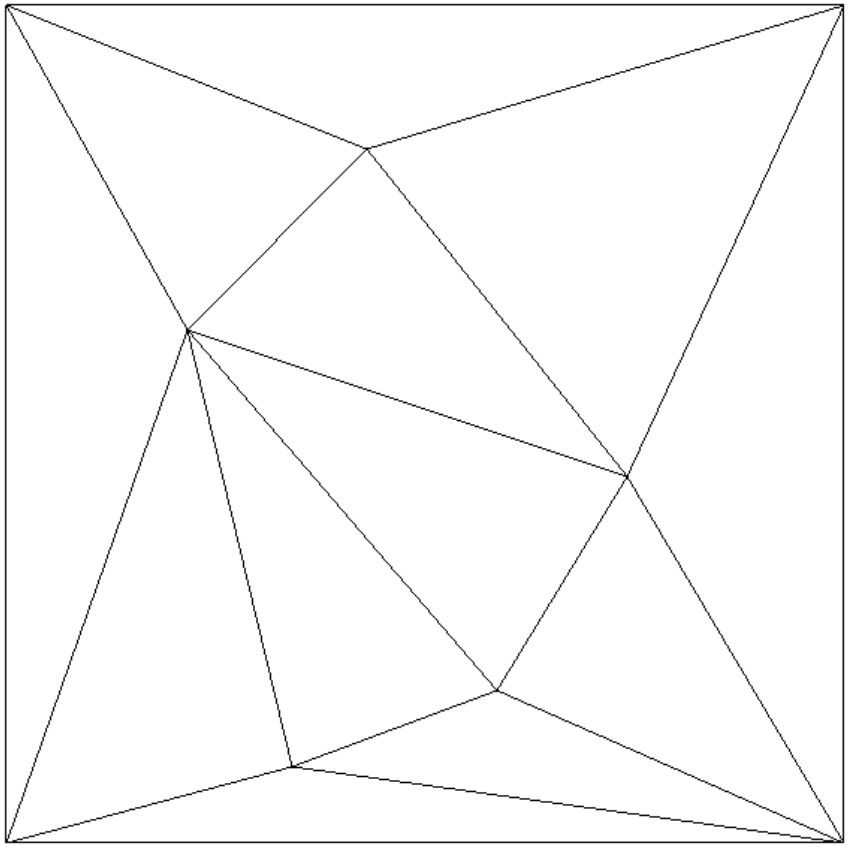


- Se realiza el test InCircle, que determina si el punto opuesto al triangulo se encuentra dentro o fuera de la circunferencia circunscrita al triángulo.
- Si se hace el cambio de arista, se referencian los dos nuevos triángulos con la referencia de los triángulos antiguos.
- Se aplica el algoritmo recursivamente sobre las aristas de los nuevos triángulos.
- Los dos nuevos triángulos reemplazan a los triángulos antiguos en la lista de triángulos.

Triangulación con Delaunay



Antes del LegalizeEdge



Después del LegalizeEdge

Dificultades encontradas

- Aproximadamente en el 1% de los casos el punto nunca se encuentra y la función de búsqueda entra en un loop infinito, es decir 1 de cada 100 puntos no podrá ser insertado. Esto se debe a que el punto entra en una zona "fantasma" de ángulo ciego donde no puede ser detectado por la función de búsqueda.

- Es más probable que este error suceda si la triangulación tiene puntos demasiado juntos generando triángulos demasiado finos.

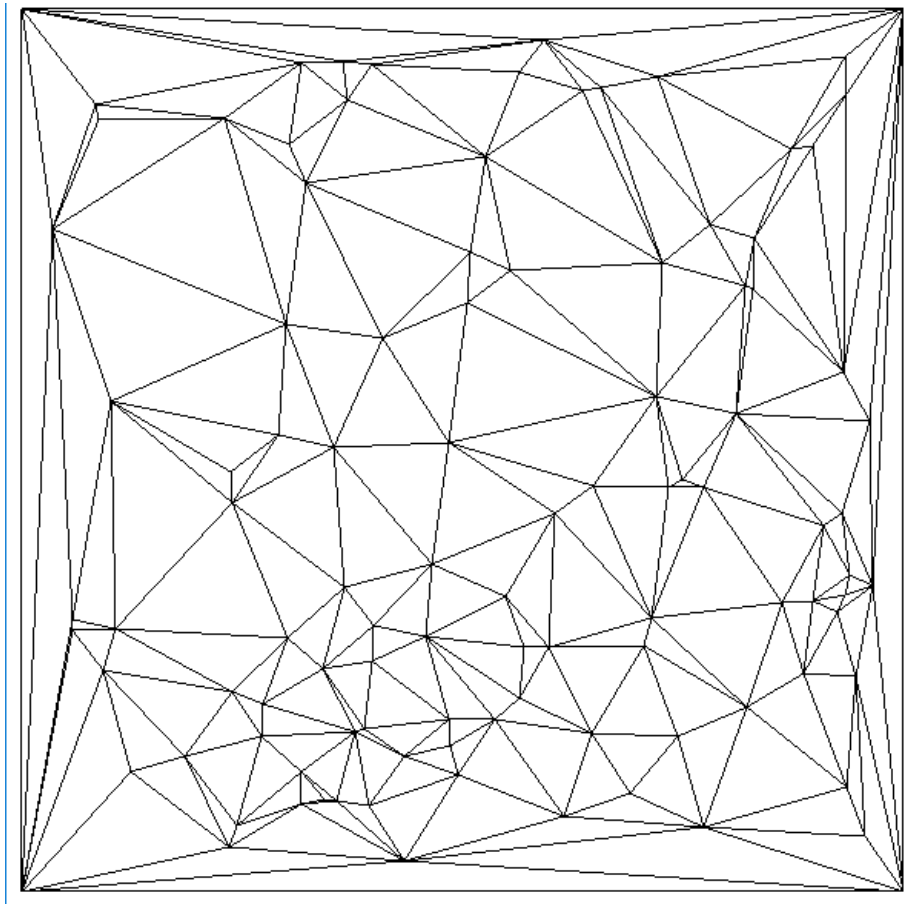
- Se soluciona cortando la función si las llamadas recursivas exceden el límite y se deja el punto sin insertar.

- Por lo tanto se alcanza en un 99% la cantidad de triángulos ideales, según la fórmula $tri\acute{a}ngulos = 2n - 2 - k$.

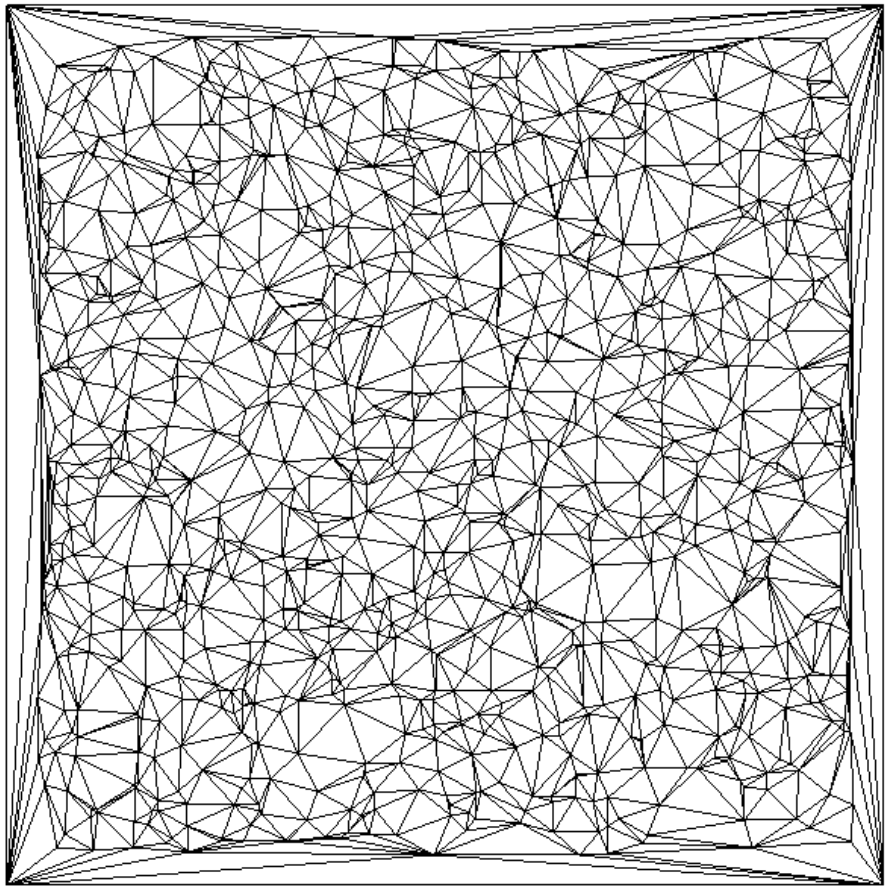
Análisis de errores

- En ciertos casos como se pudo ver anteriormente, la búsqueda no es capaz de hallar el punto, esto según el análisis de los errores arrojados por Python, se debe a que la función cae en un estado de recursividad infinita, denominada “loop”, es decir, la función busca por ciertos triángulos pero el camino retorna al mismo triángulo por donde comenzó la búsqueda, o simplemente retorna a un triángulo que se encontró en la búsqueda y que a partir de ese momento no pudo salir del ciclo.
- Esto error se genera por poca consistencia en una parte de la implementación de búsqueda, es decir, la búsqueda no fue correctamente implementada, dejando unos vacíos en la parte de los puntos críticos (ver diapositiva 7), ya que al traspasar la búsqueda a otro triángulo, no sabremos si ese triángulo podrá continuar correctamente con la búsqueda del punto y llevarla a su fin.
- Una opción para solucionar esto y debido a que este error se produce en casos muy particulares con probabilidad de 1%, es de comenzar la búsqueda de nuevo de ese punto, pero asignando un Q diferente que puede cumplir con algunas condiciones de idoneidad, como seleccionar un Q que se encuentre más cerca del punto P y de esta forma el camino para finalizar la búsqueda sea más corta.

Resultados finales

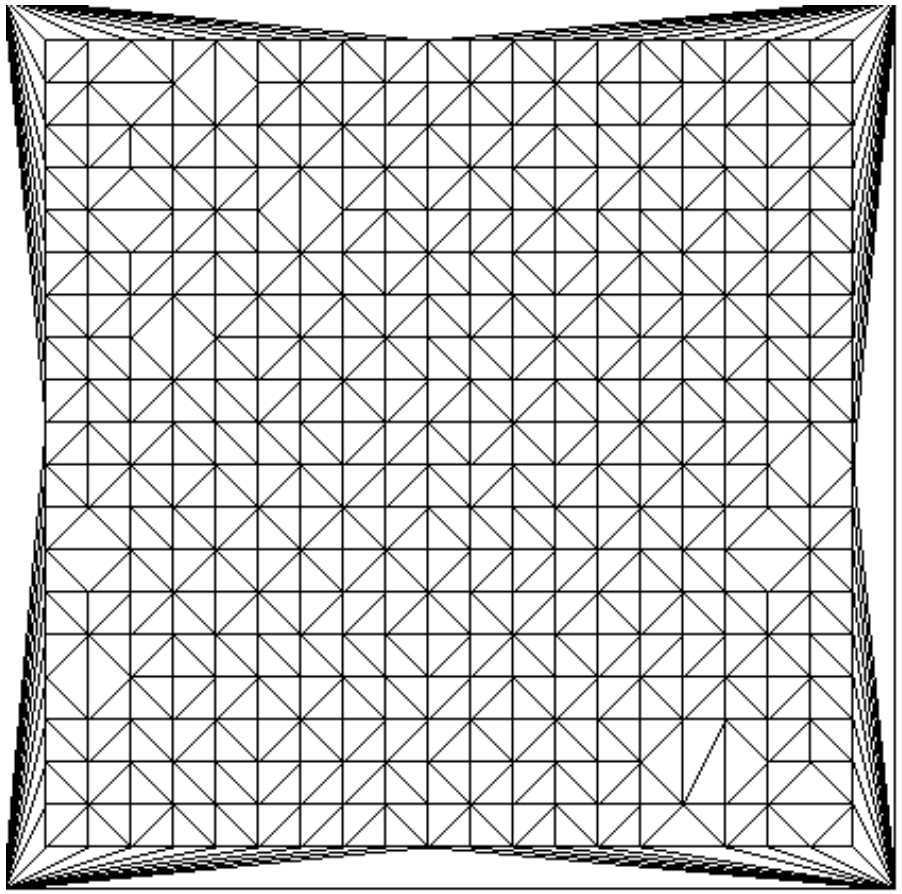


100 puntos aleatorios

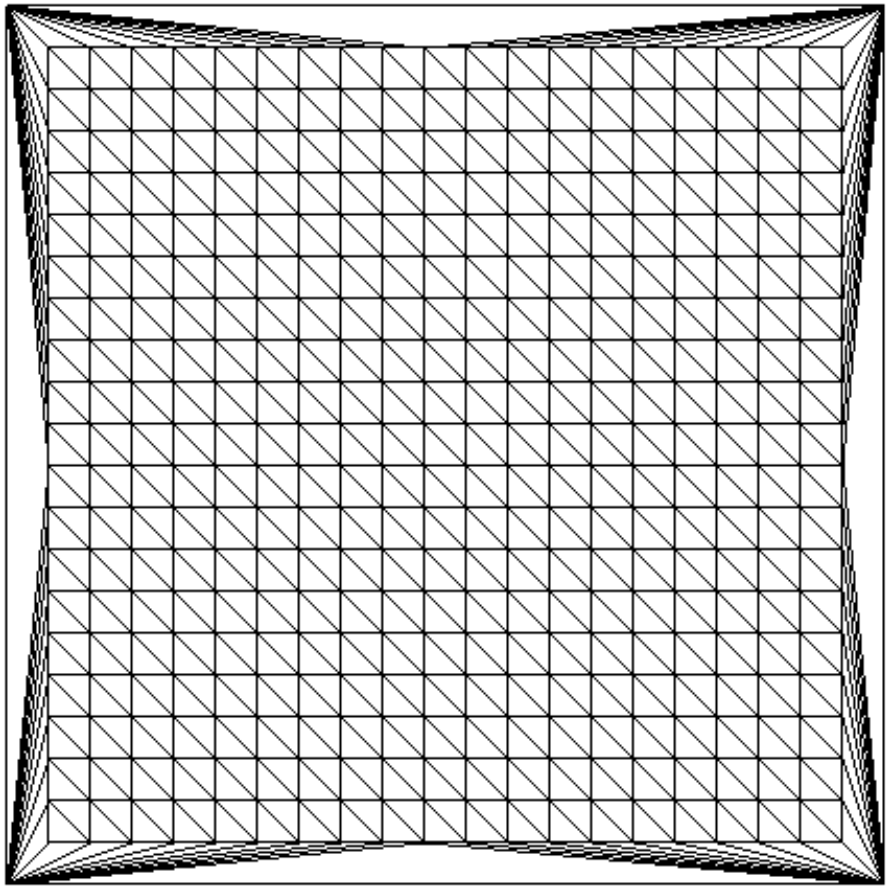


1000 puntos aleatorios

Resultados finales

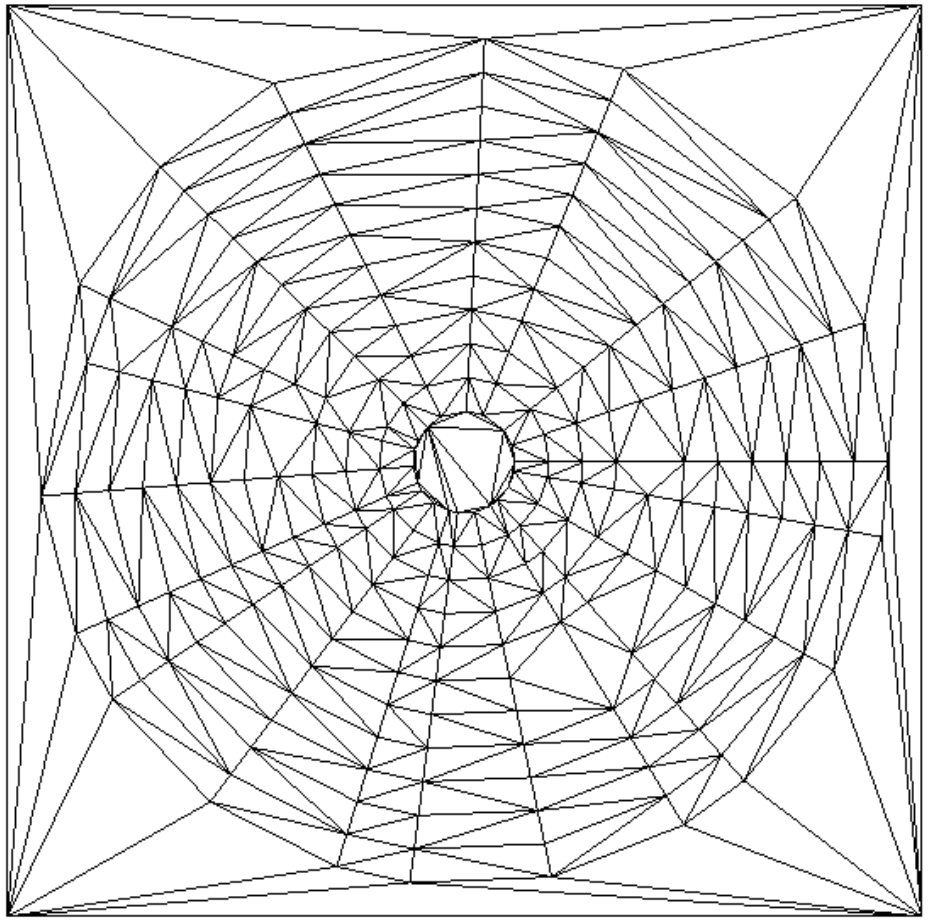


Grilla de 400 puntos aleatorios.



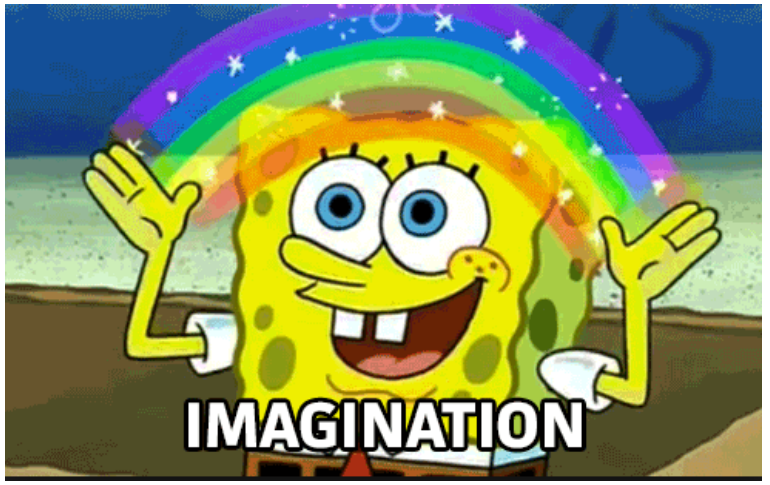
Grilla de 400 puntos ordenados.

Resultados finales

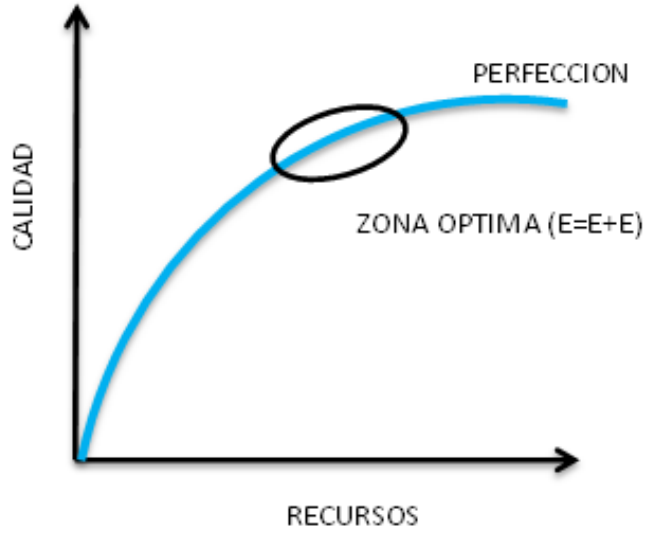


Malla circular de puntos

Aprendizaje y proceso



1. Si puedes imaginarlo, puedes programarlo…
2. “Lo perfecto es enemigo de lo bueno” Voltaire.
3. Hacer testing de todo lo que se programa.



Conclusiones

- La búsqueda de puntos recorriendo un camino guiado acelera la inserción de puntos de manera importante.
- Mientras menos uniforme sea la malla de puntos, menos uniforme quedará la triangulación
- En contra de lo pensado, la inserción de puntos de forma ordenada (para este programa), ayuda a obtener una mejor triangulación.