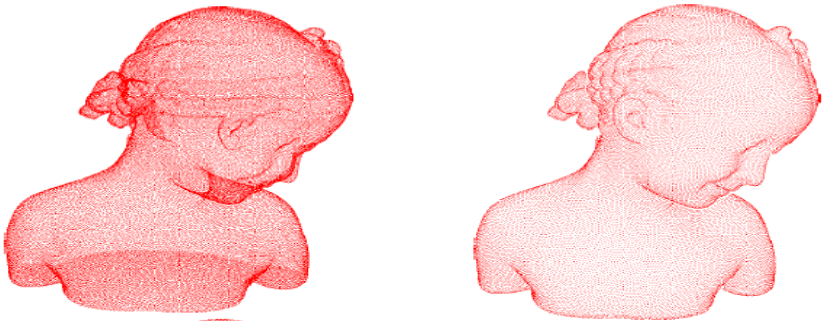
Relatório Projeto Final de Programação

Aluno: Luis Fernando Marin Sepulveda

19 de junho de 2021

1 Introdução e Motivação

A construção de figuras geométricas a partir de pontos é uma das metodologias utilizadas para a representação de objetos por meio de ferramentas computacionais. Existem várias técnicas que permitem transformar os conjuntos de pontos, a eficácia dessas técnicas depende do objetivo a ser alcançado, por exemplo a Figura 1 mostra duas representações diferentes de um mesmo conjunto de pontos, porém a Figura (b) mostra uma técnica de representação que mostra apenas os pontos visíveis de um ponto de observação, o que permite uma interpretação mais compreensível.



(a) *Conjunto original.* (b) *Interpretação de um ponto de vista.*

Figura 1: Exemplo de exibição de ponto [6].

Esses conjuntos de pontos são geralmente conectados por arestas, o que permite a criação de malhas poligonais, sua construção determina em grande parte a qualidade de interpretação do objeto. Em geometria computacional existem várias técnicas que são utilizadas como base para a construção da malha poligonal, este projeto de programação visa apresentar diversas técnicas para a construção de malhas triangulares e quadriláteras, que permitem ao usuário ver as diferenças entre as estruturas criadas a partir conjuntos de pontos.

1.1 Organização do Trabalho

A organização do trabalho é realizada da seguinte forma: A seção de objetivos apresenta o objetivo geral e os objetivos específicos do projeto de programação.

A seção Fundamentos apresentará a definição dos elementos básicos que compõem o desenvolvimento dos módulos, bem como a descrição das estruturas geométricas resultantes.

Na seção de Planejamento e Desenvolvimento, é apresentada uma descrição dos requisitos funcionais e não funcionais, casos de uso necessários, bem como dos diagramas de modelos, da mesma forma que são apresentadas as ferramentas de desenvolvimento utilizadas.

Finalmente, A seção Testes descreve como os testes foram realizados e seus resultados.

2 Objetivos

2.1 objetivo geral

Desenvolver uma ferramenta computacional que permita a construção de uma malha de triângulos e quadriláteros.

2.2 objetivos específicos

1. Crie uma função que permite criar conjuntos de pontos aleatórios dentro de um espaço bidimensional, com limites definidos pelo usuário.
2. Desenvolva uma função que permita a construção de uma malha triangular usando uma abordagem incremental.
3. Desenvolver uma função que permita a construção de uma malha triangular de Delaunay.
4. Construir uma função que permita a construção de uma malha de quadriláteros.

3 Fundamentos

Esta seção descreve alguns elementos básicos da geometria sobre os quais os módulos são construídos, esses elementos quando combinados formam estruturas, que são representadas por diferentes conjuntos de dados e cuja representação pode ser feita por meio de um gráfico.

3.1 Entidades Geométricas

3.1.1 Vetor

Segmento direcionado de uma linha reta no espaço euclidiano, uma das extremidades (ponto A) é considerada a origem, enquanto a outra (ponto B) é considerada a extremidade do vetor. Tal vector puede ser denotado por a, $\bar{a}$, $\vec{a}$ ou $\bar{AB}$. Um vetor cuja origem e fim coincidem é considerado um vetor zero e geralmente é denotado por 0. Um vetor é caracterizado por seu módulo (ou comprimento), que é igual ao comprimento do segmento AB e é denotado por $|\mathbf{a}|$, e vai de A para B [4].

3.1.2 Orientação

Em um espaço bidimensional, a orientação entre os pontos depende de como a rota é feita para ir de um ponto de origem a um ponto de destino: se a rota percorrida descreve um movimento no sentido anti-horário, a orientação é positiva, caso contrário é definida como negativo [7].

3.1.3 Polígono

Uma linha tracejada fechada, a saber: Se $A, ..., A_n$ são pontos diferentes, dos quais não há três em uma linha, então a coleção de segmentos $[A_1A_2]$, $[A_2A_3]$, ..., $[A_n, A_1]$ é chamada de polígono ou curva poligonal [3].

3.1.4 Triângulo

Três pontos (os vértices) e segmentos de linha reta (os lados) com pontos finais nesses pontos. Às vezes, a definição de um triângulo se refere à parte convexa do plano que é delimitada pelos lados do triângulo (o triângulo sólido) [8].

3.1.5 Quadrilátero

Dados quatro vértices, A, B, C e D, onde não existem três colineares e de forma que para qualquer par de segmentos (arestas) AB, BC, CD e DA não tenham um ponto comum ou simplesmente tenham um ponto final comum. É definido o quadrilátero $ABCD$, que consiste nos quatro segmentos mencionados. [5].

3.1.6 Delaunay triangulation

No contexto de um conjunto de pontos finitos S, um triângulo é Delaunay se seus vértices estão em S e seu círculo aberto está vazio, isto é. não contém nenhum ponto em S. Observe que qualquer número de pontos em S pode estar na circunferência de um triângulo de Delaunay. Uma aresta é Delaunay se seus vértices estão em S e ela tem pelo menos um circundisk aberto vazio. Uma triangulação de Delaunay de S, é uma triangulação de S em que cada triângulo é Delaunay. A triangulação Delaunay em duas dimensões tem uma vantagem surpreendente: entre todas as triangulações possíveis de um conjunto fixo de pontos, a triangulação Delaunay maximiza o ângulo mínimo. Ele também otimiza vários outros critérios geométricos relacionados à precisão da interpolação [2].

3.1.7 Catmull-Clark-based quadrilaterals

Quando aplicado a uma malha poligonal, produz uma malha quadrilateral. A aplicação do algoritmo pode ser apresentada em três etapas: Primeiro, determine o ponto central de cada aresta do polígono. Segundo, calcular o ponto central do polígono. Finalmente, junte os novos pontos criados através de arestas, o resultado é uma malha de quadrados [1]

3.1.8 Quadrilateral Meshing Using 4 Clustering

Consiste em uma metodologia apresentada por Velho [9], na qual a subdivisão de catmull-clark é aplicada a uma malha de triângulos que foi inicialmente agrupados, tomando como critério de união, o comprimento das arestas adjacentes a dois triângulos.

4 Planejamento e Desenvolvimentó

Esta seção apresenta os requisitos funcionais e não funcionais, os modelos de design usados no processo de desenvolvimento de recursos e as ferramentas usadas.

4.1 Requisitos Funcionais e Não Funcionais

Os requisitos serão classificados em dois critérios, em relação à prioridade e ao esforço. Essa classificação de requisitos é importante para estimar o tempo de modelagem e desenvolvimento. O critério de prioridade pode ser avaliado como: Essencial, Importante ou Desejável. O critério de esforço pode ser avaliado em: Baixo, Médio ou Alto. A Tabela 1 mostra os requisitos funcionais e a Tabela 2 mostra os requisitos não funcionais.

Requisitos Funcionais	**Prioridade**	**Esforço**
O usuário poderá criar um conjunto aleatório de pontos.	Essencial	Baixo
As informações de ponto, aresta e triângulo devem ser armazenadas usando estruturas de dados topológicos.	Importante	Médio
Para cada conjunto de pontos, o sistema criará uma triangulação incremental.	Essencial	Alto
Para cada conjunto de pontos, o sistema criará uma triangulação Delaunay.	Essencial	Alto
Para cada triangulação, o sistema criará um agrupamento, produzindo uma malha de quadriláteros e triângulos.	Essencial	Alto
Para cada agrupamento, o sistema criará uma malha de quadriláteros.	Essencial	Alto

Tabela 1: Especificação Requisitos Funcionais.

Requisitos Não Funcionais	**Prioridade**	**Esforço**
A implementação deve registrar o custo do tempo computacional.	Essencial	Baixo
Cada conjunto de pontos receberá uma identificação baseada no nome do arquivo.	Importante	Baixo
O sistema deve criar uma imagem para cada iteração, mostrando o andamento de cada função.	Desejável	Baixo

Tabela 2: Especificação Requisitos Não Funcionais.

4.2 Casos de Uso

Esta seção apresenta os casos de uso, separados em subsistemas Figuras 2 - 7.

4.2.1 Ator

User: Refere-se a qualquer pessoa com acesso ao sistema.

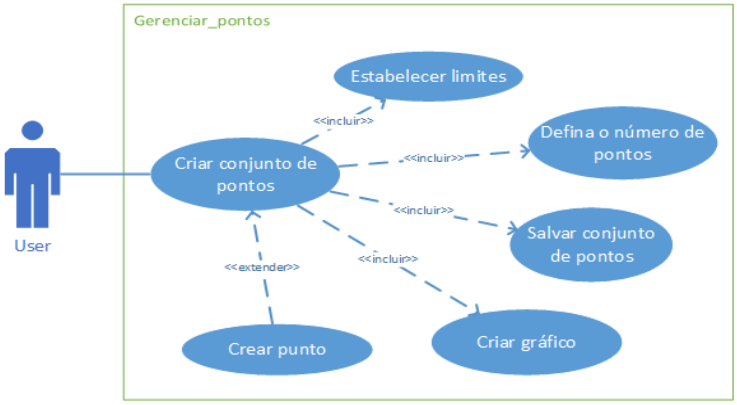


Figura 2: Caso de uso: Criar conjunto de pontos.

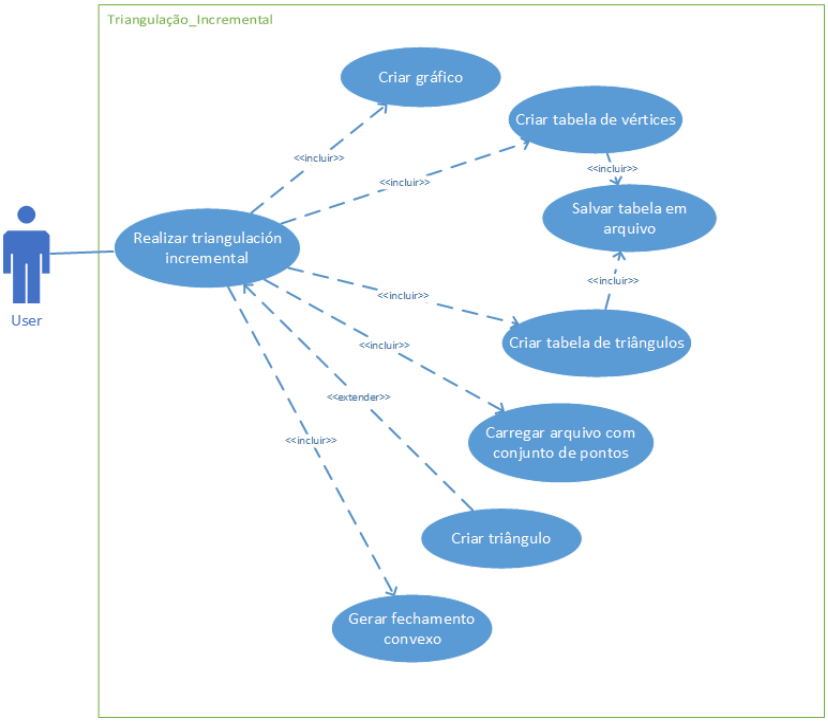


Figura 3: Caso de uso: Realizar triangulación incremental.

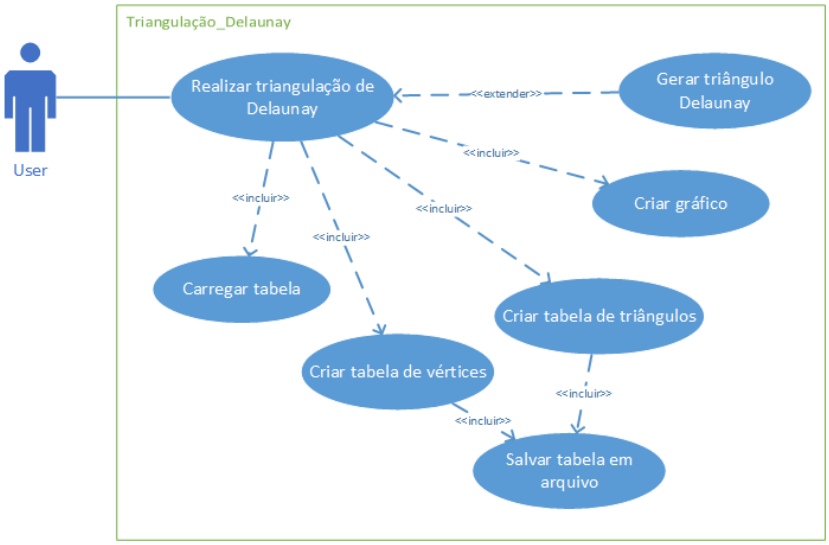


Figura 4: Caso de uso: Realizar triangulación Delaunay.

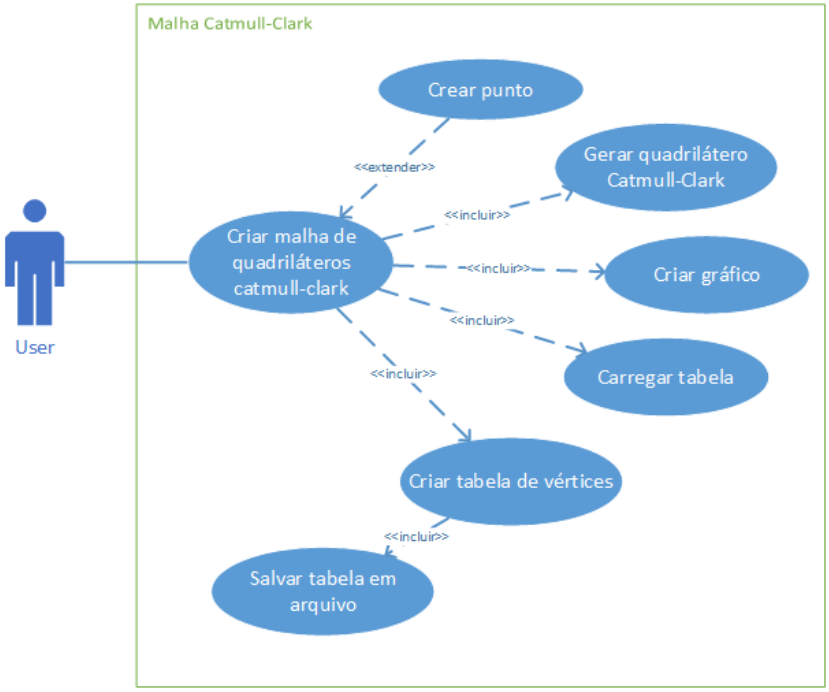


Figura 5: Caso de uso: Criar malha de quadriláteros Catmull-Clark.

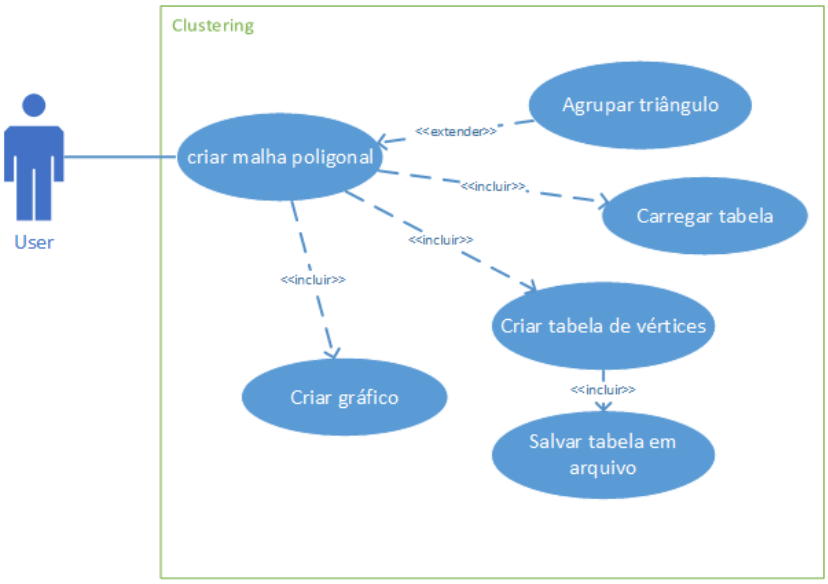


Figura 6: Caso de uso: Criar malha poligonal.

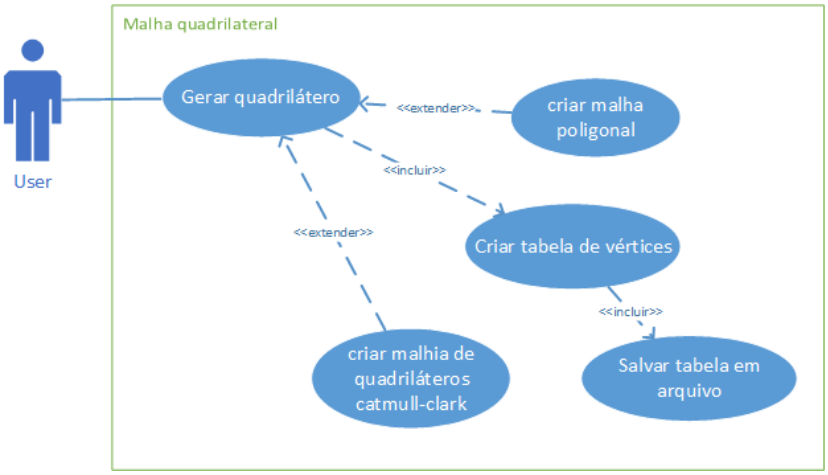


Figura 7: Caso de uso: Gerar quadrilátero.

4.3 Diagrama de Classes

A Figura 8 mostra o diagrama de classes desenvolvido para a funcionalidades.

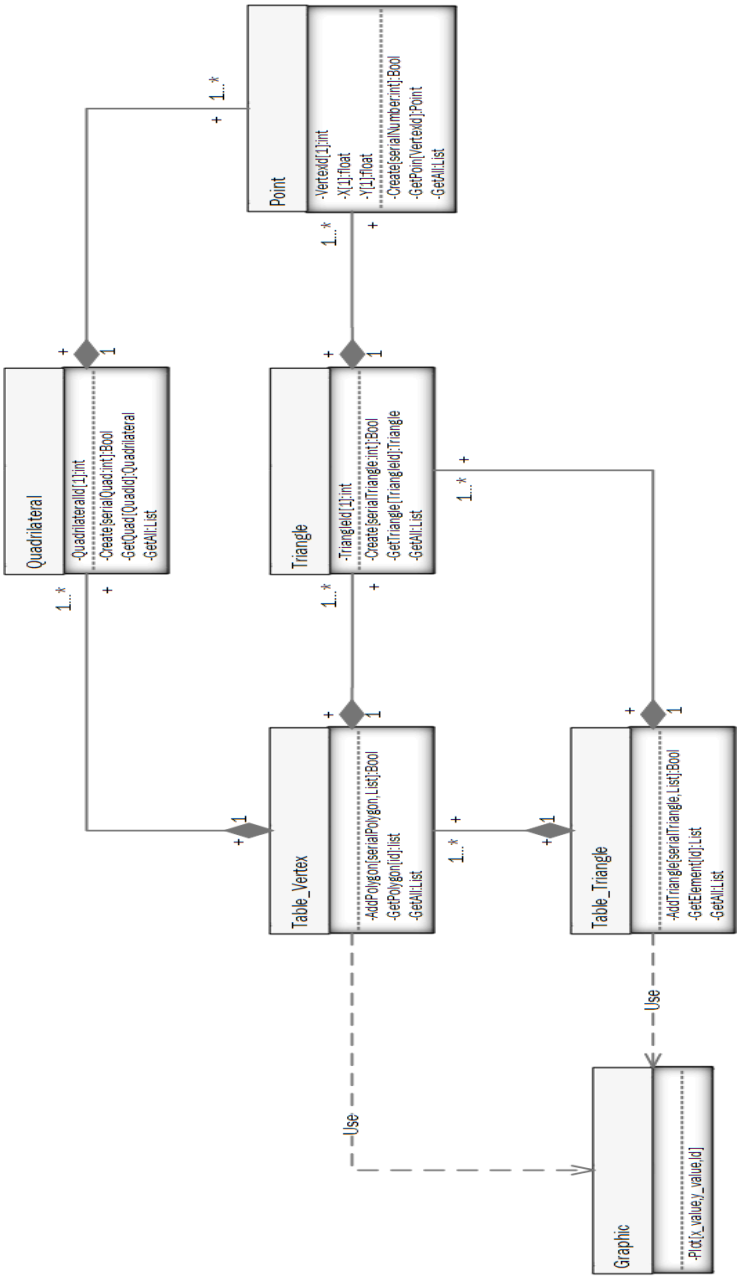


Figura 8: Diagrama de classes.

4.4 Diagramas de Sequência

Esta seção apresenta os diagramas de sequência para os diferentes casos de uso, bem como suas descrições.

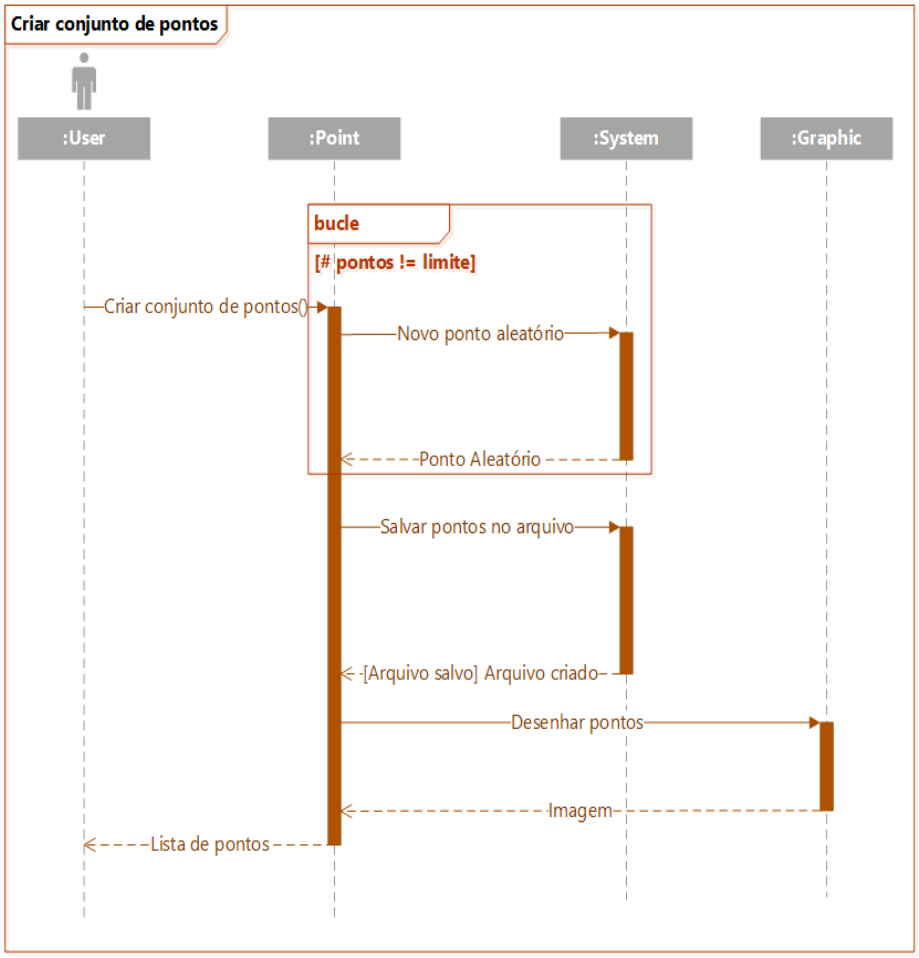


Figura 9: Diagrama de Sequência (Criar conjunto de pontos).

Descrição - Criar conjunto de pontos
A sequência Criar conjunto de pontos Figura 9, começa com o usuário solicitando a criação de pontos, na qual ele deve informar os limites e a quantidade de pontos desejados. Sempre que o número de pontos requerido não for atingido, Point pedirá ao sistema para criar pontos aleatórios, ao atingir o número de pontos requerido, Point pede ao sistema para salvar os pontos em um arquivo e então pede ao Graphic para desenhar o pontos. Finalmente, a lista de pontos criada é devolvida ao usuário.

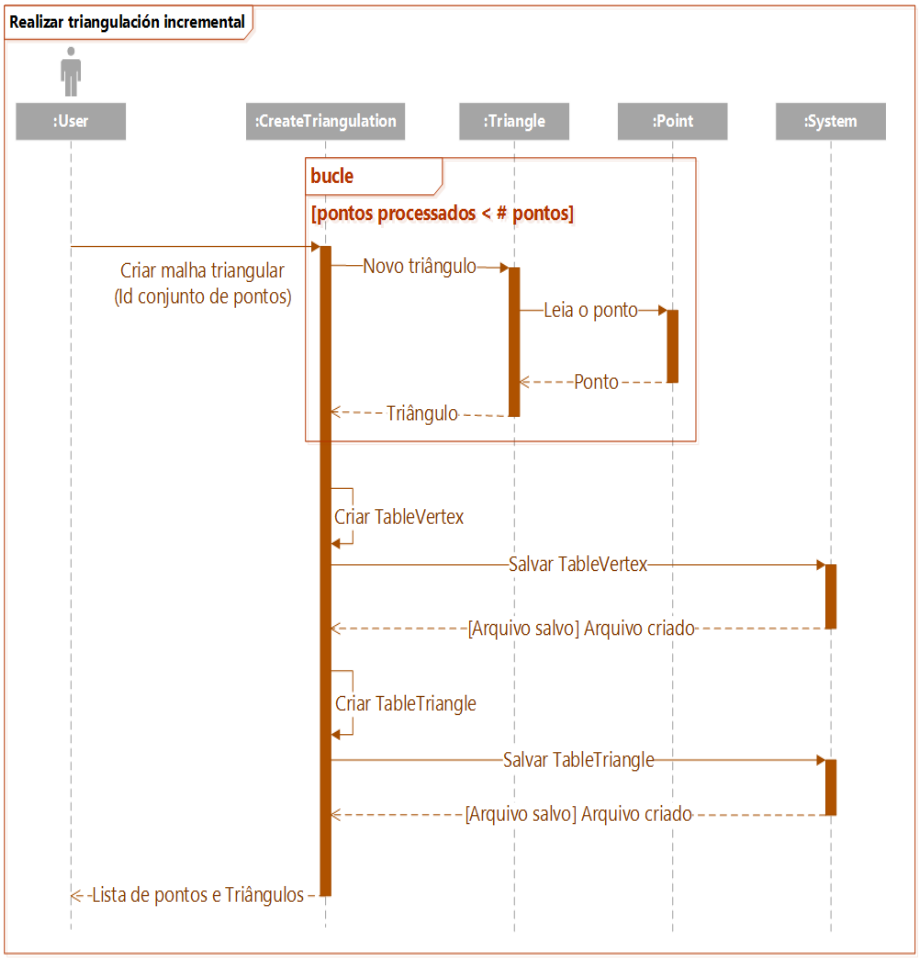


Figura 10: Diagrama de Sequência (Realizar triangulación incremental).

Descrição - Realizar triangulación incremental
A sequência Realizar triangulación incremental Figura 10, começa com o usuário solicitando a criação de malha triangular informando el id del conjunto de puntos. Até que todos os pontos sejam processados, a função CreateTriangulation pede a Triangle para criar um novo triângulo, para isso pede a Point que forneça os pontos de forma ordenada. Em seguida, a função CreateTriangulation constrói as tabelas Vertex e Triangles, pedindo ao System para salvá-las em arquivos. Finalmente, pontos e triângulos são enviados ao usuário.

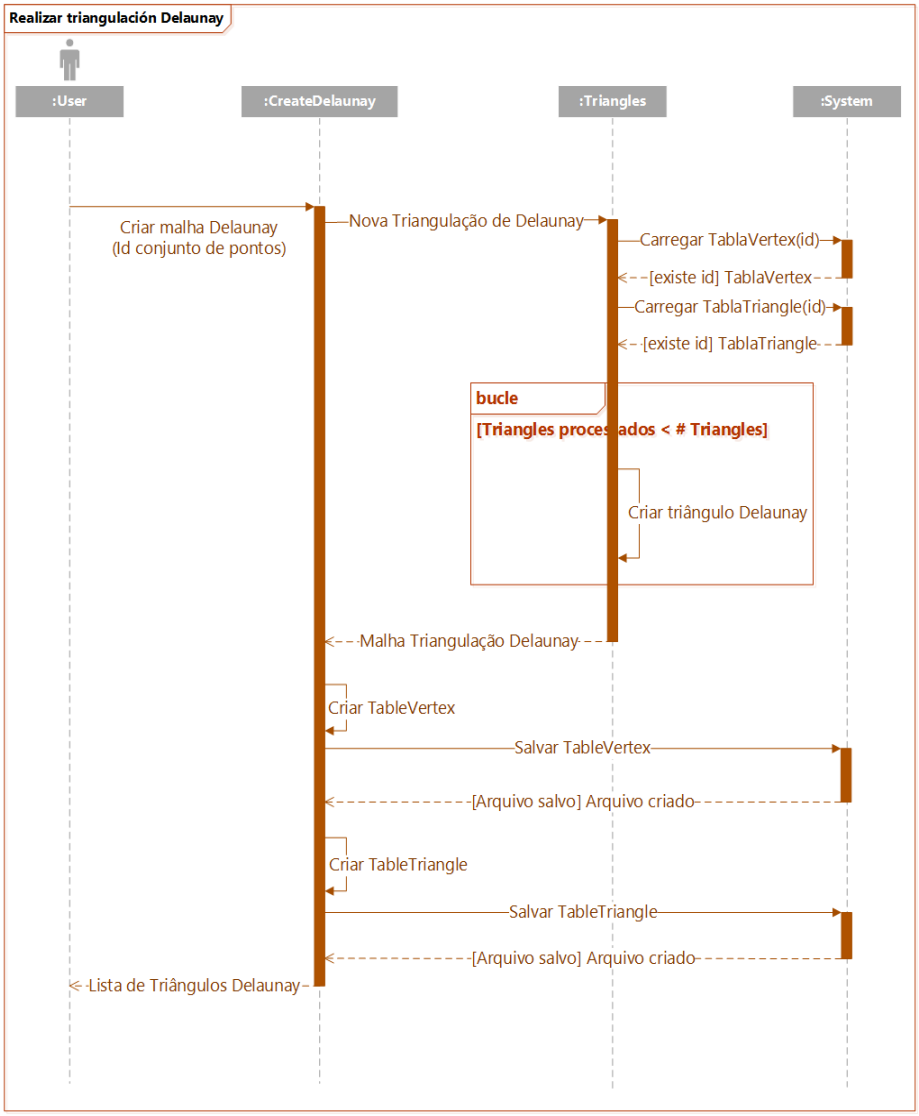


Figura 11: Diagrama de Sequência (Realizar triangulación Delaunay).

Descrição - Realizar triangulación Delaunay
A sequência Realizar triangulación Delaunay Figura 11, começa com o usuário solicitando a criação de malha triangular, informando el id del conjunto de pontos. CreateDelaunay envia a solicitação para criar uma nova triangulação de Delaunay para Triangles. Triangles pede ao System para carregar as tabelas Vertex e Triangles, se o Id existir, o System retorna as tabelas após a leitura dos arquivos. Até que todos os triângulos contidos nas tabelas sejam processados, Triangles criará um triângulo de Delaunay, no final ele retornará uma malha de Delaunay. CrateDelaunay cria novas tabelas Vertex e Triangle e pede ao sistema para criar novos arquivos. Finalmente, a malha triangular Delaunay é devolvida ao usuário.

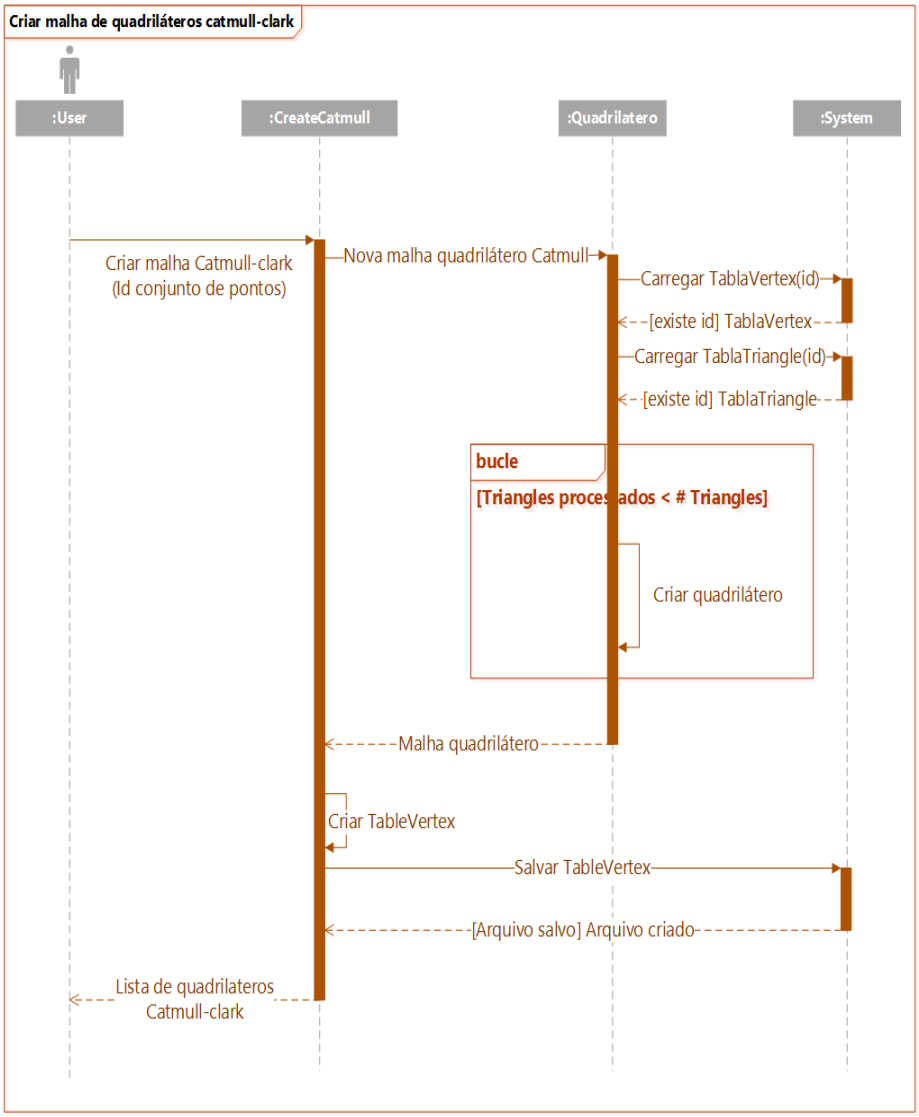


Figura 12: Diagrama de Sequência (Criar malha de quadriláteros catmull-clark).

Descrição - Criar malha de quadriláteros catmull-clark
A sequência Criar malha de quadriláteros catmull-clark Figura 12, começa com o usuário solicitando a criação de malha Catmull-clark, informando el id del conjunto de pontos. CreateCatmull envia a solicitação para criar um novo quadrilátero Catmull para Quadrilatero. Quadrilatero pede ao System para carregar as tabelas Vertex e Triangles, se o Id existir, o System retorna as tabelas após a leitura dos arquivos. Até que todos os triângulos contidos nas tabelas sejam processados, Quadrilatero criará um quadrilatero, no final ele retornará uma malha. CrateCatmull cria uma nova tabela Vertex e pede ao sistema para criar um novo arquivo. Finalmente, uma lista de quadrilateros Catmull-clark é devolvida ao usuário.

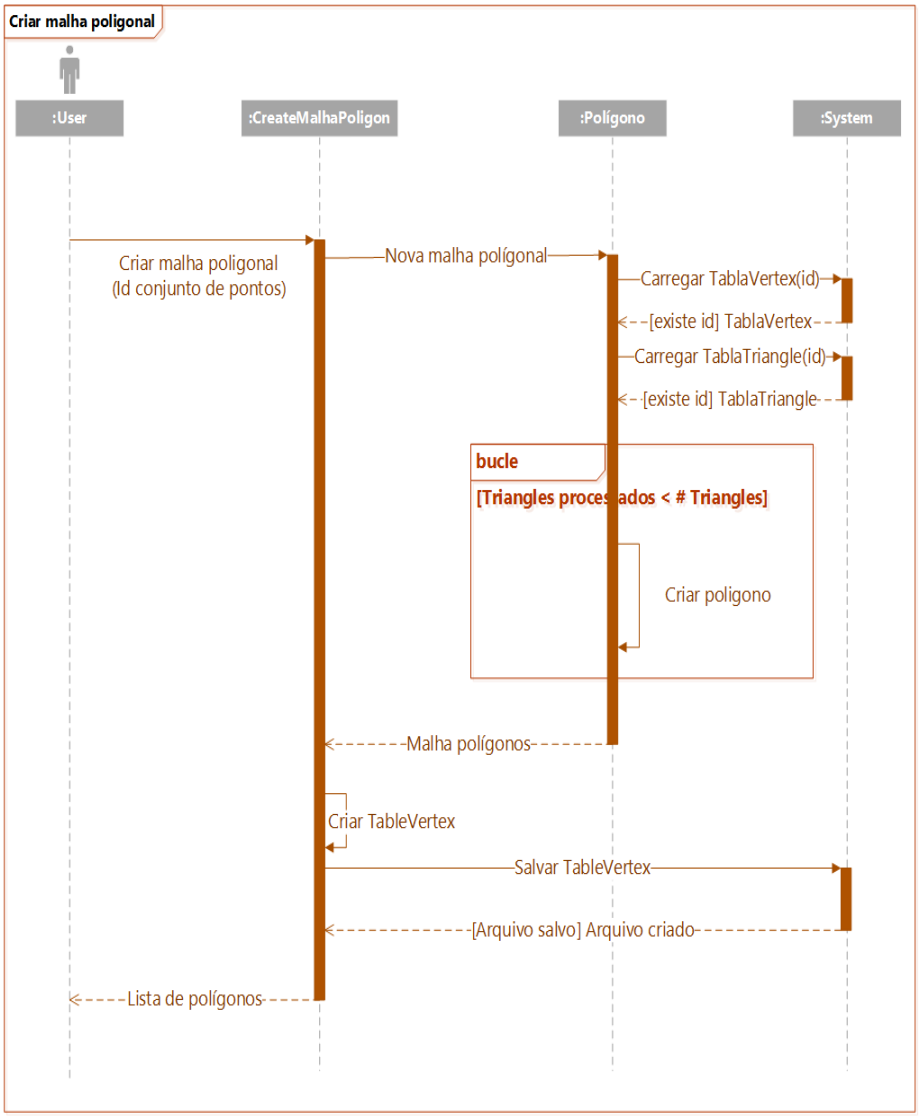


Figura 13: Diagrama de Sequência (Criar malha poligonal).

Descrição - Criar malha poligonal
A sequência Criar malha poligonal Figura 13, começa com o usuário solicitando a criação de malha poligonal, informando el id del conjunto de pontos. CreatePoligon envia a solicitação para criar uma nova malha poligonal para Poligono. Poligono pede ao System para carregar as tabelas Vertex e Triangles, se o Id existir, o System retorna as tabelas após a leitura dos arquivos. Até que todos os triângulos contidos nas tabelas sejam processados, Poligono criará um poligono, no final ele retornará uma malha. CreatePoligon cria uma nova tabela Vertex e pede ao sistema para criar um novo arquivo. Finalmente, uma lista de poligonos é devolvida ao usuário.

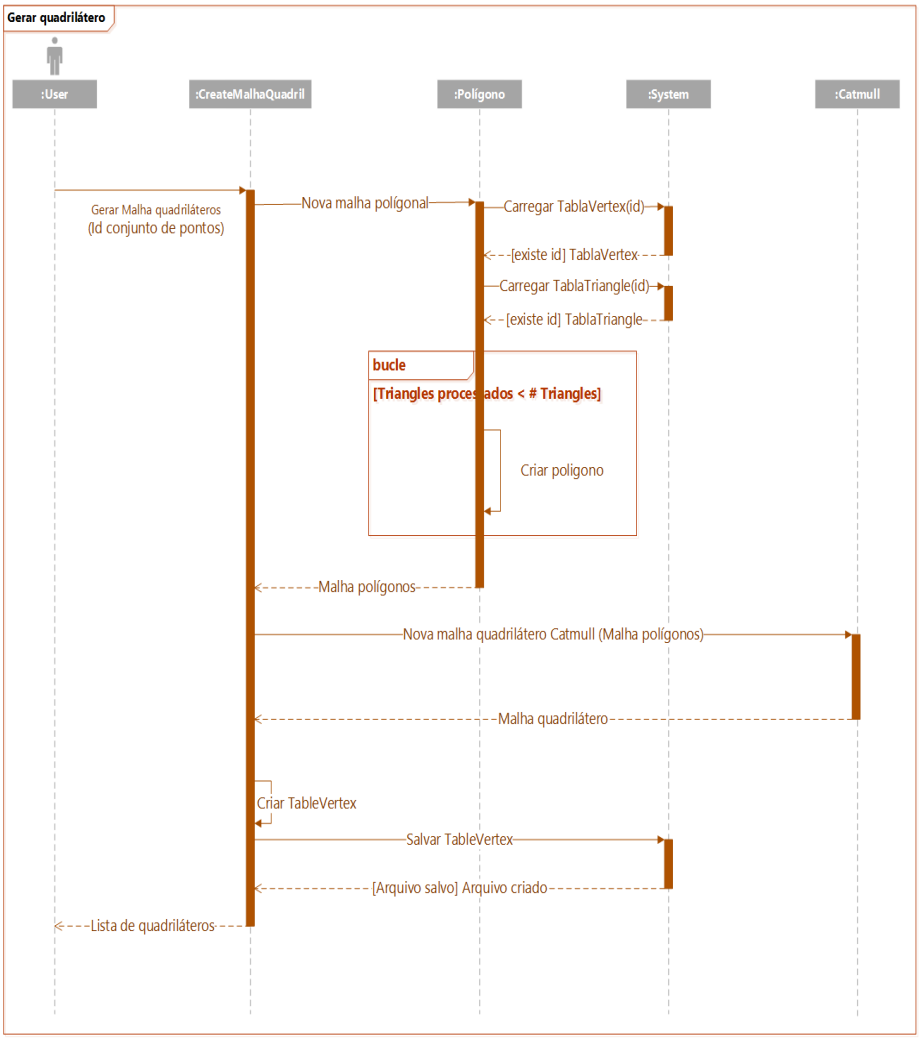


Figura 14: Diagrama de Sequência (Gerar quadrilátero).

Descrição - Gerar quadrilátero
A sequência Gerar quadrilátero Figura 14, começa com o usuário solicitando gerar malha de quiadriláteros, informando el id del conjunto de pontos. CreateMalhaQuadril envia a solicitação para criar uma nova malha poligonal para Poligono. Poligono pede ao System para carregar as tabelas Vertex e Triangles, se o Id existir, o System retorna as tabelas após a leitura dos arquivos. Até que todos os triângulos contidos nas tabelas sejam processados, Poligono criará um poligono, no final ele retornará uma malha. CreateMalhaQuadril envia a solicitação para criar um novo quadrilátero para Catmull, Catmull transforma a malha poligonal em uma malha quadrilateral Catmull-Clark. Então CreateMalhaQuadril cria uma nova tabela Vertex e pede ao sistema para criar um novo arquivo. Finalmente, uma lista de quadriláteros é devolvida ao usuário.

4.5 Plataforma de Desenvolvimento e Ferramentas

Esta seção apresenta as bibliotecas e ferramentas de desenvolvimento que foram utilizadas para a construção dos módulos.

Como IDE, Spyder 4.1.4 foi usado com Python 3.8.3, A Tabela 3 apresenta as bibliotecas utilizadas e suas versões.

Biblioteca	**Versão**
matplotlib	3.2.2
glob2	0.7
numpy	1.18.5
pandas	1.0.5
sympy	1.6.1

Tabela 3: Lista de bibliotecas.

5 Testes

Para garantir a funcionalidade de cada módulo, os testes foram compostos por testes unitários.

5.1 Testes dos Requisitos

Os testes foram concebidos para avaliar o comportamento das funcionalidades implementadas. Foi elaborado um roteiro simples para testar cada um dos requisitos funcionais estabelecidos.

1. O módulo implementado permite ao usuário criar diferentes conjuntos de pontos, definindo a quantidade e os limites. A Figura 15 mostra os resultados dos conjuntos criados.

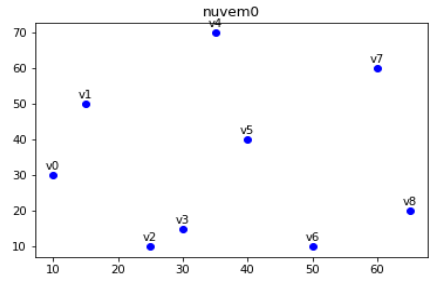
2. O módulo desenvolvido permite a criação de uma malha triangular utilizando uma abordagem incremental. A Figura 16 mostra os resultados da triangulação.

3. O módulo implementado permite a criação de uma triangulação baseada em Delaunay. A Figura 17 mostra os resultados da triangulação.

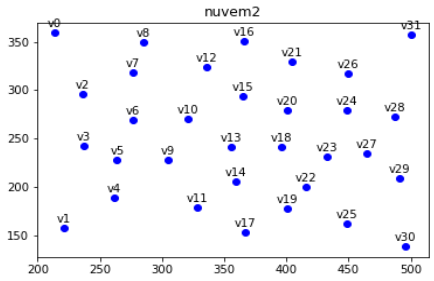
4. O módulo desenvolvido permite a criação de um cluster de triângulos, gerando uma malha de polígonos. A Figura 18 mostra os resultados do agrupamento.

5. O módulo implementado permite a criação de quadriláteros baseados em Catmull-Clark. A Figura 19 mostra os resultados do agrupamento.

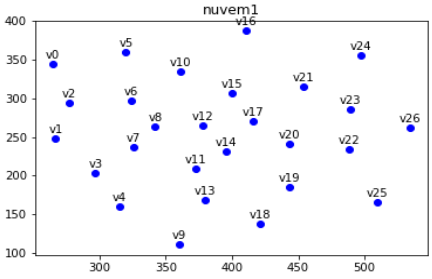
6. O módulo desenvolvido permite a criação de quadriláteros baseados em Catmull-Clark, aplicados a um cluster de triângulos. A Figura 20 mostra os resultados dos quadriláteros.



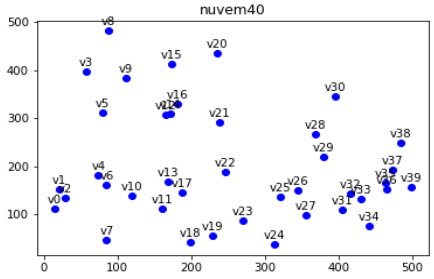
(a) *Conjunto de 9 pontos.*



(b) *Conjunto de 32 pontos.*

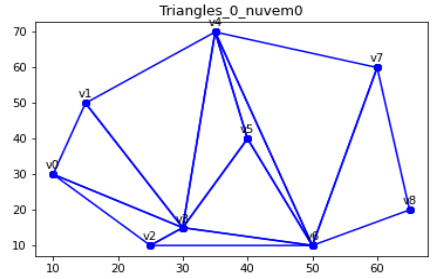


(c) *Conjunto de 27 pontos.*

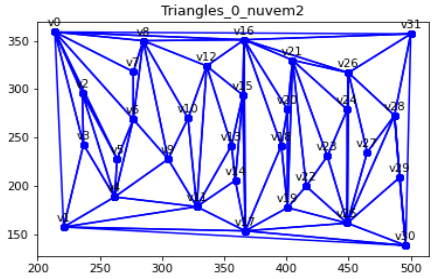


(d) *Conjunto de 40 pontos.*

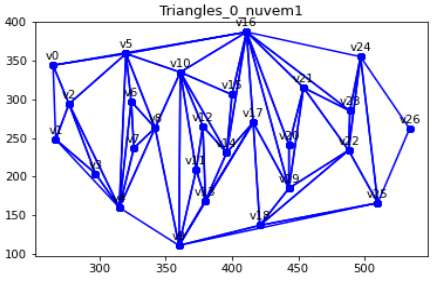
Figura 15: Pontos gerados.



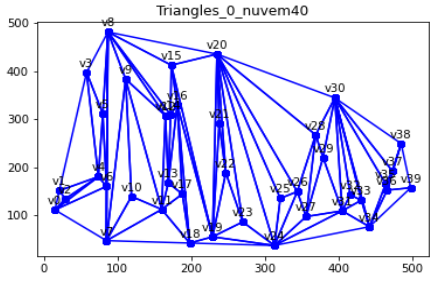
(a) *Triangulação incremental de 9 pontos.*



(b) *Triangulação incremental de 32 pontos.*

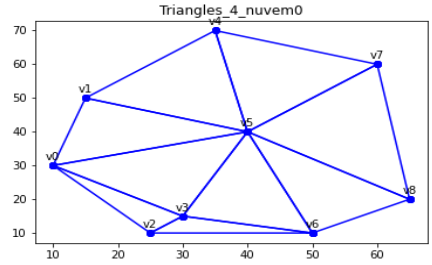


(c) *Triangulação incremental de 27 pontos.*

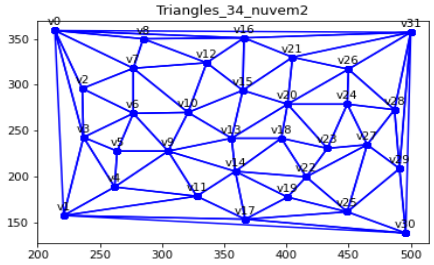


(d) *Triangulação incremental de 40 pontos.*

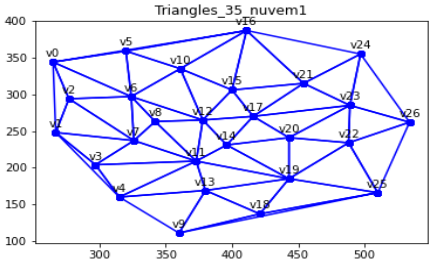
Figura 16: Triangulação incremental.



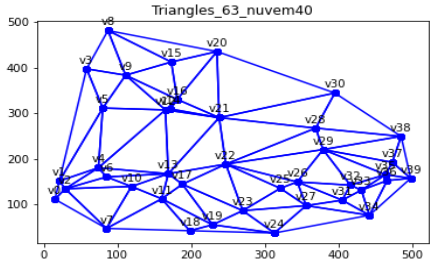
(a) *Triangulação Delaunay de 9 pontos.*



(b) *Triangulação Delaunay de 32 pontos.*

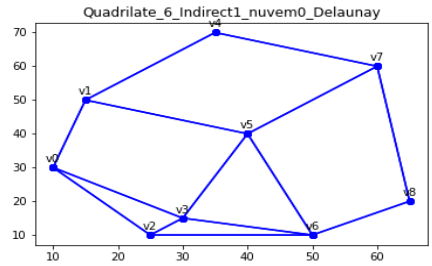


(c) *Triangulação Delaunay de 27 pontos.*

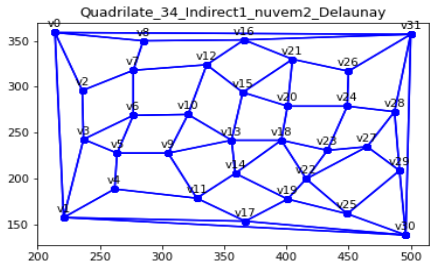


(d) *Triangulação Delaunay de 40 pontos.*

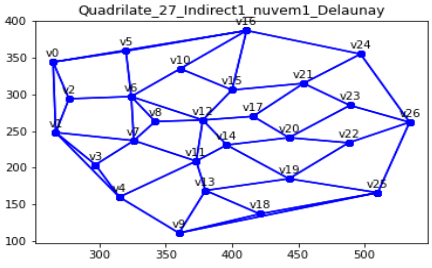
Figura 17: Triangulação Delaunay.



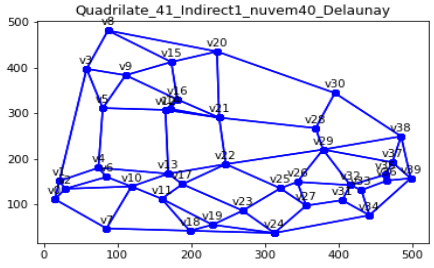
(a) *Agrupamento de 9 pontos.*



(b) *Agrupamento de 32 pontos.*

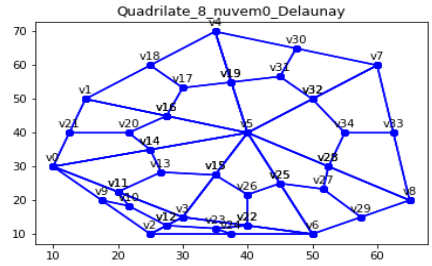


(c) *Agrupamento de 27 pontos.*

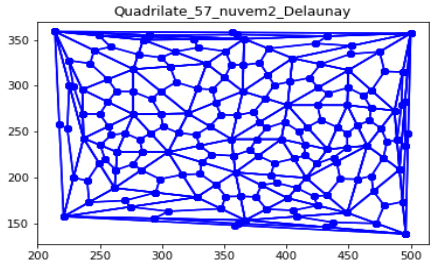


(d) *Agrupamento de 40 pontos.*

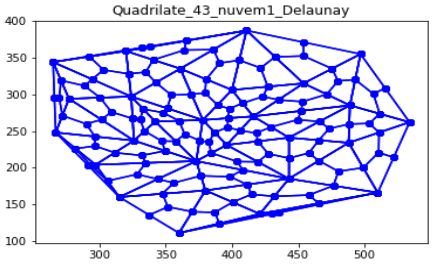
Figura 18: Agrupamento.



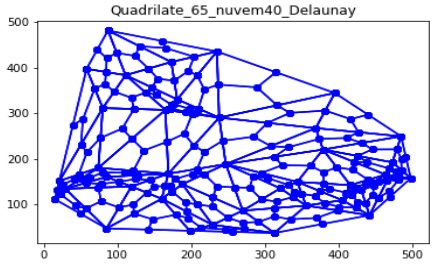
(a) *Catmull-Clark de 9 pontos de partida.*



(b) *Catmull- Clark de 32 pontos de partida.*

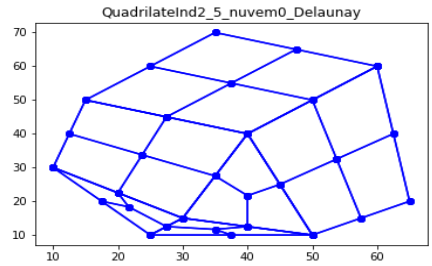


(c) *Catmull- Clark de 27 pontos de partida.*

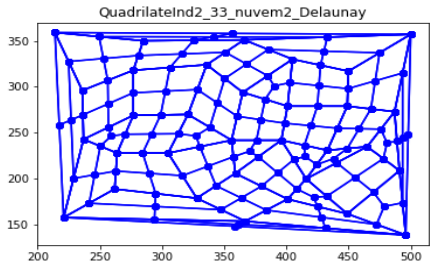


(d) *Catmull-Clark de 40 pontos de partida.*

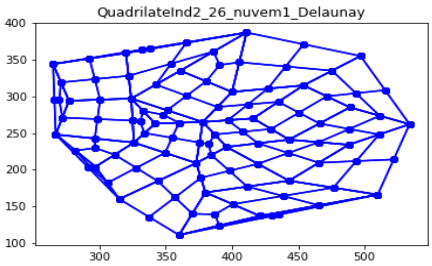
Figura 19: Quadriláteros baseados em Catmull-Clark.



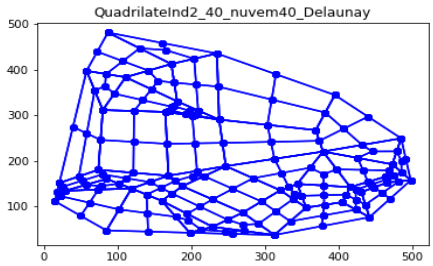
(a) *Quadriláteros de 9 pontos de partida.*



(b) *Quadriláteros de 32 pontos de partida.*



(c) *Quadriláteros de 27 pontos de partida.*



(d) *Quadriláteros de 40 pontos de partida.*

Figura 20: Quadriláteros após de cluster.

5.2 Análise de tempo computacional

Para medir o tempo computacional, nove conjuntos de pontos foram gerados aleatoriamente.

- A Figura 21 mostra o comportamento resultante para uma triangulação de Delaunay.

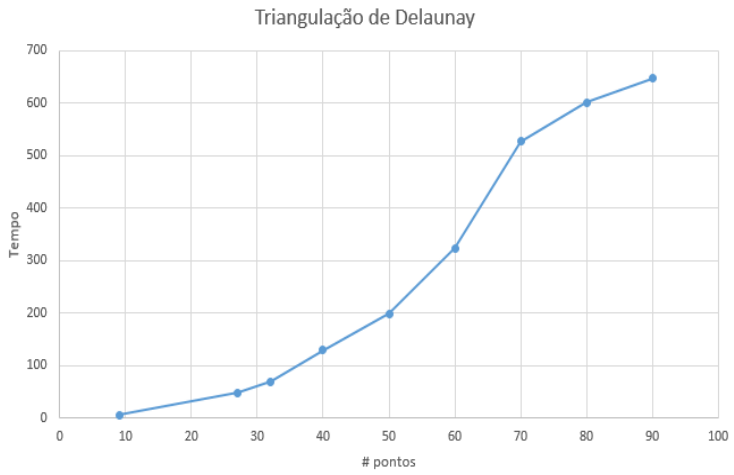


Figura 21: Tempo computacional para triangulação de Delaunay.

O tempo de cálculo resultante para o algoritmo de Delaunay apresenta um comportamento esperado de $O(n \log n)$.

- A Figura 22 mostra o tempo computacional para o modulo quadriláteros Catmull-Clark.

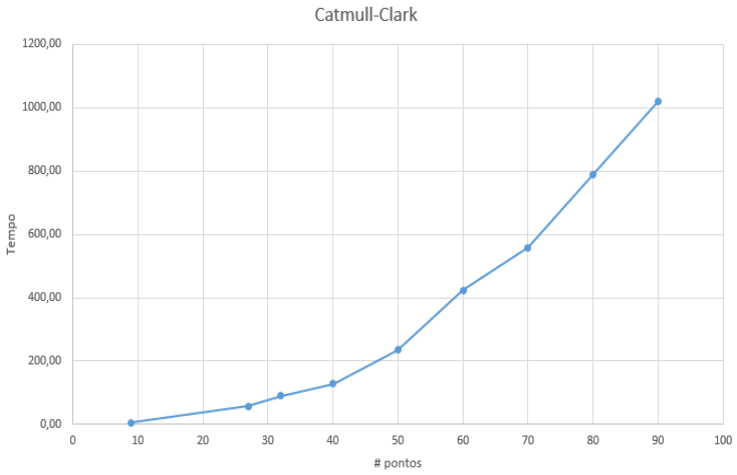


Figura 22: Tempo computacional para Catmull-Clark.

Ao aumentar o número de pontos, o modulo Catmull-Clark exibe um

comportamento $O(n \log n)$, o alto tempo computacional é explicado pelo grande número de triângulos.

- A Figura 23 mostra o tempo computacional para o modulo agrupamento de triângulos.

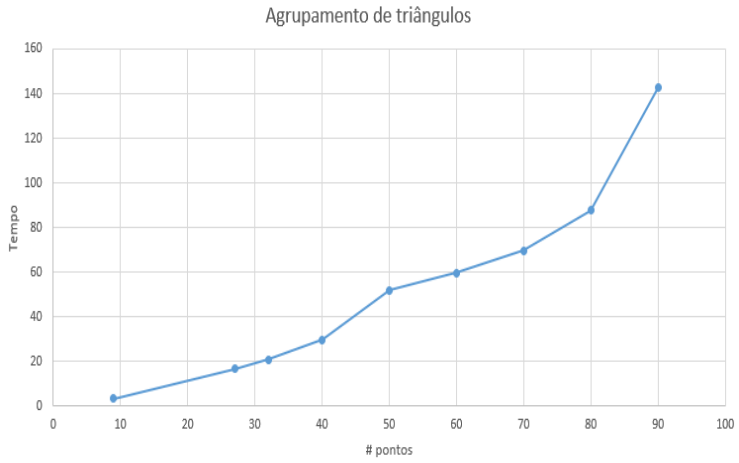


Figura 23: Tempo computacional para agrupamento de triângulos.

Ao aumentar o número de pontos, o modulo de agrupamento exibe um comportamento que pode estar entre $\Omega(n \log n)$ e $O(n^2)$.

- A Figura 24 mostra o tempo computacional para o modulo Quadriláteros após de cluster.

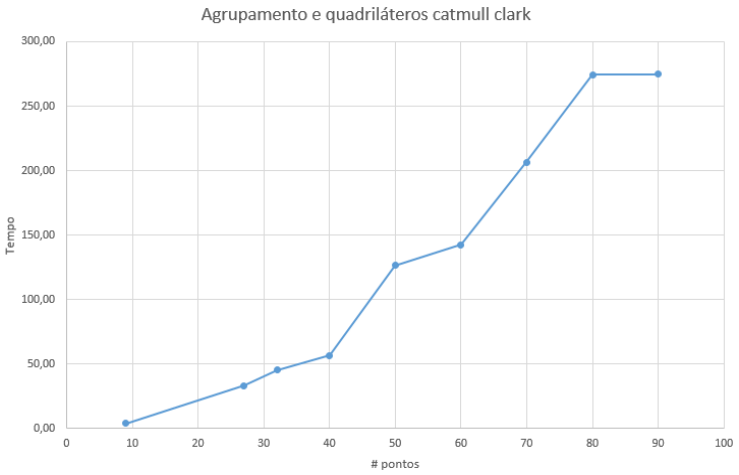


Figura 24: Tempo computacional para Quadriláteros após de cluster.

Ao aumentar o número de pontos, o modulo quadriláteros após de cluster exibe um comportamento que pode estar localizado entre $\Omega(n)$ e $O(n \log n)$.

Referências

[1] E. Catmull and J. Clark. Recursively generated B-spline surfaces on arbitrary topological meshes. *Computer-Aided Design*, 1978. ISSN 00104485. doi: 10.1016/0010-4485(78)90110-0.

[2] S. W. Cheng, T. K. Dey, and J. R. Shewchuk. *Delaunay mesh generation*. 2012. ISBN 9781584887317. doi: 10.1201/b12987.

[3] H. Coxeter and H. Coxeter. *Introduction to Geometry*. Wiley Classics Library. Wiley, 1989. ISBN 9780471504580. URL `https://books.google.com.br/books?id=c0ld-crynsIC`.

[4] M. Hazewinkel. *G*, pages 160–343. Springer Netherlands, Dordrecht, 1989. ISBN 978-94-009-5997-2. doi: 10.1007/978-94-009-5997-2_2. URL `https://doi.org/10.1007/978-94-009-5997-2_2`.

[5] A. Hirst and M. J. Greenberg. Euclidean and Non-Euclidean Geometries - Development and History. *The Mathematical Gazette*, 1995. ISSN 00255572. doi: 10.2307/3620116.

[6] S. Katz, A. Tal, and R. Basri. Direct visibility of point sets. *ACM Transactions on Graphics*, 2007. ISSN 07300301. doi: 10.1145/1276377.1276407.

[7] C. Lehmann. *Analytic Geometry*. J. Wiley & sons, Incorporated, 1942. URL `https://books.google.com.br/books?id=lfhQAAAAMAAJ`.

[8] E. A. Maxwell, H. S. M. Coxeter, and S. L. Greitzer. Geometry Revisited. *The Mathematical Gazette*, 1968. ISSN 00255572. doi: 10.2307/3614178.

[9] L. Velho. Quadrilateral Meshing Using 4-8 Clustering. *Proc. CILANCE'00*, 2000.