

Disciplina: Computação Gráfica

Professor: Marcio Sarroglia Pinho

Trabalho 01 – Diagrama de Voronoi

Felipe Freitas e Lucas Wolschick

Sumário

1. O propósito	3
2. Explicações Gerais	3
2.1 Início do programa	3
2.2 Movimento Interno	3
2.3 Movimento Externo	4
2.3.1 Inclusão de pontos em polígonos côncavos	4
2.3.2 Inclusão de pontos em polígonos convexos	4
2.3.3 Inclusão de pontos em polígonos convexos utilizando a informação de vizinhança disponível no diagrama de Voronoi	4
3. Cenários de teste	5
3.1 Cenário 1 - 20 Polígonos	5
3.1.1 Método inicial	6
3.1.2 Movimento interno	7
3.1.3 Inclusão de pontos em polígonos côncavos	7
3.1.4 Inclusão de pontos em polígonos convexos	9
3.1.5 Inclusão de pontos em polígonos convexos com informação de vizinhança	10
3.2 Cenário 2 - 100 Polígonos	11
3.2.1 Método inicial	12
3.2.2 Movimento interno	13
3.2.3 Inclusão de pontos em polígonos côncavos	13
3.2.4 Inclusão de pontos em polígonos convexos	14
3.2.5 Inclusão de pontos em polígonos convexos com informação de vizinhança	16
3.3 Cenário 3 - 500 Polígonos	17
3.3.1 Método inicial	18
3.3.2 Movimento interno	19
3.3.3 Inclusão de pontos em polígonos côncavos	20
3.3.4 Inclusão de pontos em polígonos convexos	21
3.3.5 Inclusão de pontos em polígonos convexos com informação de vizinhança	22
4. Conclusões	23

1. O propósito

O propósito deste relatório é implementar e analisar três métodos relacionados à Computação Gráfica: Inclusão de pontos em polígonos côncavos, inclusão de pontos em polígonos convexos e inclusão de pontos em polígonos convexos utilizando a informação de vizinhança disponível no Diagrama de Voronoi. Os métodos serão avaliados em cenários com 20, 100 e 500 polígonos.

2. Explicações Gerais

2.1 Início do programa

Ao iniciar o programa, após carregar todos os polígonos e desenhar o ponto, deve-se descobrir quais são os vizinhos do polígono e, posteriormente em qual posição e polígono o ponto está localizado inicialmente.

Para o primeiro, temos de passar por todos os polígonos e armazenar as informações de quais outros polígonos compartilham uma aresta com este. Para isso, utilizamos dos envelopes e, caso haja colisão entre o envelope do polígono em questão e o do próximo, considera-se que eles são vizinhos, visto que, como todos os polígonos são convexos, sempre que houver colisão há também um compartilhamento de aresta.

Depois disso, passamos novamente por todos os polígonos, em ordem numérica, e realizamos o produto vetorial para saber se o polígono em questão contém o ponto. Quando alcançarmos o polígono, paramos o algoritmo e salvamos em uma variável em qual polígono estamos. Este processo funciona bem se, por acaso, o primeiro polígono for o central, mas caso o polígono inicial seja o último da lista, teríamos que realizar $\text{quantidadeDePoligonos} \times \text{quantidadeDeVerticesDosPoligonos}$ operações, o que é muito mais custoso.

2.2 Movimento Interno

Ao movimentarmos o ponto, executamos o algoritmo de produto vetorial apenas no polígono em que estávamos anteriormente, que foi salvo no passo anterior. Como só temos de testar N vezes, onde N é o número de vértices do polígono anterior, este é um meio eficiente de definir se ainda estamos dentro do mesmo polígono ou se saímos dele, mas não é o melhor para definir para qual polígono fomos, como vimos anteriormente.

2.3 Movimento Externo

Ao movimentarmos o ponto, caso o algoritmo anterior aponte que não estamos mais no mesmo polígono, vamos realizar 3 testes: inclusão de pontos em polígonos côncavos, inclusão em polígonos convexos e inclusão em polígonos convexos com a informação de quem são os vizinhos do polígono em questão.

2.3.1 Inclusão de pontos em polígonos côncavos

Para melhorar a eficiência deste algoritmo, vamos utilizar um algoritmo de faixas para limitar a validação apenas aos polígonos que estiverem dentro da mesma faixa que a linha horizontal usada para teste. Todas as vezes que um polígono estiver dentro dos limites desta faixa, vamos passar por todas as suas arestas e contar quantas vezes existe uma intersecção. Se este número for par, entendemos que o ponto não está neste polígono. Repetimos o algoritmo até encontrar algum polígono no qual a quantidade de intersecções seja ímpar e que, portanto, contém o ponto.

2.3.2 Inclusão de pontos em polígonos convexos

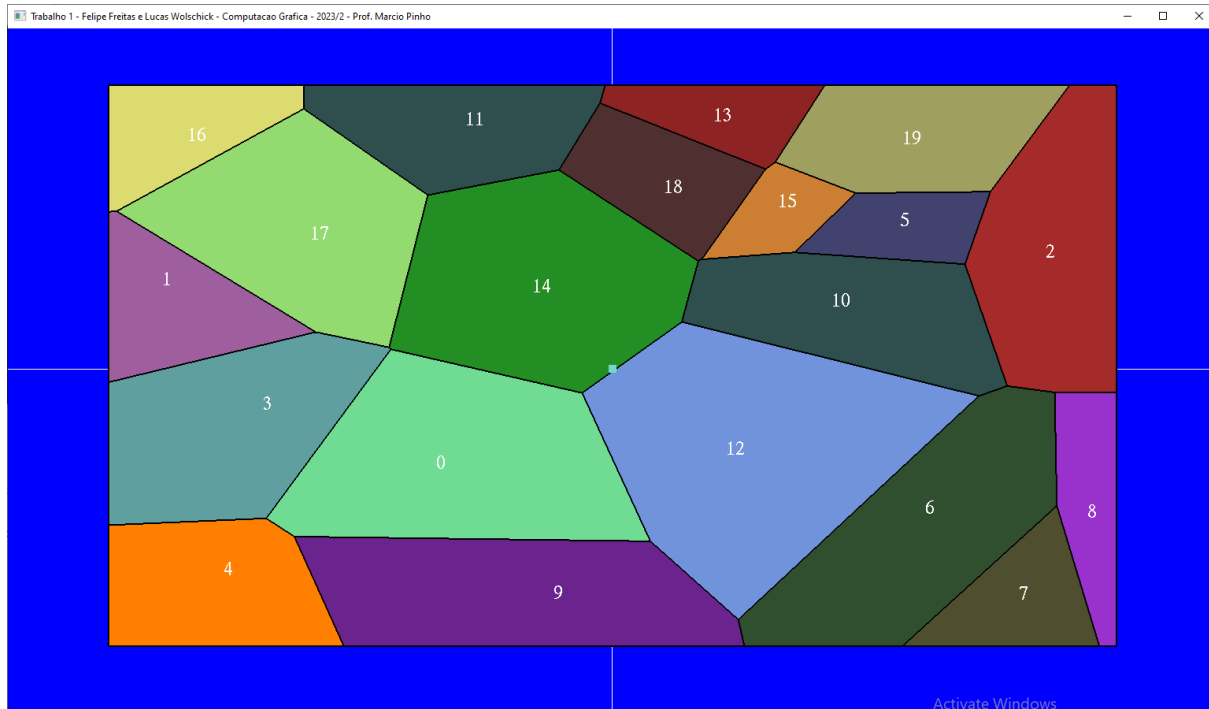
Desta vez, optou-se por utilizar um algoritmo de inclusão em envelopes para reduzir a quantidades de polígonos que devem ser testados, ou seja, vamos passar por todos os polígonos do diagrama e, caso o ponto esteja dentro do seu envelope, vamos executar novamente a função de produto vetorial para determinar se o ponto está dentro do polígono em questão. Repetimos o processo até que algum dos polígonos contenha o ponto, isto é, o resultado do produto vetorial seja consistente para todas as arestas.

2.3.3 Inclusão de pontos em polígonos convexos utilizando a informação de vizinhança disponível no diagrama de Voronoi

Aqui, ao invés de reduzir as possibilidades por uma linha que cruza todo o diagrama ou então ter de passar por todo o diagrama para determinar quais envelopes envolveriam o ponto, vamos percorrer um limite de N polígonos - onde N é a quantidade de vizinhos do polígono em que o ponto estava antes - que foi obtida no passo 2.1. Para cada um dos vizinhos, vamos executar o algoritmo de produto vetorial, buscando descobrir para qual vizinho e, principalmente, por qual aresta, o ponto passou.

3. Cenários de teste

3.1 Cenário 1 - 20 Polígonos



3.1.1 Método inicial

Após contarmos e salvarmos os vizinhos de todos os polígonos, foram necessárias 81 chamadas ao método ProdVetorial para determinar a posição inicial do ponto, conforme imagens abaixo.

```
D:\Programming\GitHub\EngenhariaSoftwarePUCRS\Computacao_Grafica\Trabalho01\Voronoi\OpenGL.exe 20
Programa OpenGL
[SYS] Inicializando programa...
[SYS] Inicializando Diagrama de Voronoi...
[SYS] Lendo poligonos do arquivo casosTeste/20.txt...
[SYS] Calculando limites do diagrama Voronoi...
    Largura: 999.44
    Minimo:(-115.164, -82.57)
    Maximo:(1084.16, 908.27)
    Meio:(484.5, 412.85)
[SYS] Limites do diagrama Voronoi calculados!
[SYS] Analizando informacoes de vizinhanca...
    Poligono 0 tem 6 vizinhos.
    Poligono 1 tem 4 vizinhos.
    Poligono 2 tem 6 vizinhos.
    Poligono 3 tem 5 vizinhos.
    Poligono 4 tem 3 vizinhos.
    Poligono 5 tem 4 vizinhos.
    Poligono 6 tem 6 vizinhos.
    Poligono 7 tem 3 vizinhos.
    Poligono 8 tem 3 vizinhos.
    Poligono 9 tem 4 vizinhos.
    Poligono 10 tem 7 vizinhos.
    Poligono 11 tem 5 vizinhos.
    Poligono 12 tem 7 vizinhos.
    Poligono 13 tem 4 vizinhos.
    Poligono 14 tem 7 vizinhos.
    Poligono 15 tem 5 vizinhos.
    Poligono 16 tem 3 vizinhos.
    Poligono 17 tem 5 vizinhos.
    Poligono 18 tem 5 vizinhos.
    Poligono 19 tem 4 vizinhos.
[SYS] Informacoes de vizinhanca analisadas!
[SYS] Gerando cores dos poligonos...
[SYS] Cores dos poligonos geradas!
[SYS] Criando ponto que sera movido...
[SYS] Ponto criado!
[SYS] Computando posicao inicial do ponto...
    Analisando se estou dentro do poligono 0
[SYS] Fazendo passo inicial...
[SYS] Verificando se o ponto ainda esta dentro do poligono 0...
    Ponto esta a direita da aresta 0
    Ponto esta a esquerda da aresta 1
    Ponto esta a esquerda da aresta 2
    Ponto esta a direita da aresta 3
    Ponto esta a direita da aresta 4
    Sinal do produto vetorial inconsistente, entao o ponto nao esta dentro do poligono 0
    Testando se o ponto esta dentro do poligono 1...
    Analisando se estou dentro do poligono 1

Analizando poligonos de 1-13 ...

[SYS] Verificando se o ponto ainda esta dentro do poligono 14...
    Ponto esta a direita da aresta 0
    Ponto esta a direita da aresta 1
    Ponto esta a direita da aresta 2
    Ponto esta a direita da aresta 3
    Ponto esta a direita da aresta 4
    Ponto esta a direita da aresta 5
    Ponto esta a direita da aresta 6
    Sinal do produto vetorial consistente, entao o ponto esta dentro do poligono 14
    Quantidade de chamadas para ProdVetorial para descobrir o poligono inicial: 81
[SYS] Posicao inicial do ponto computada! (484.5, 412.85)
```

3.1.2 Movimento interno

São necessárias apenas 7 chamadas, 1 para cada aresta do polígono, para determinar se o ponto ainda está no polígono que estava antes do último movimento, conforme imagem abaixo.

```
D:\Programming\GitHub\EngenhariaSoftwarePUCRS\Computacao_Grafica\Trabalho01\Voronoi\OpenGL.exe 20
[SYS] Movendo ponto para a diagonal superior esquerda...
[SYS] Movendo ponto horizontalmente -10.8416 unidades...
[SYS] Movendo ponto verticalmente 10.8416 unidades...

=====
[SYS] Checando se o ponto esta dentro dos limites da janela e do diagrama...
[SYS] Verificando se o ponto esta dentro dos limites do diagrama...
[SYS] O ponto esta dentro da area do diagrama!
[SYS] Continuando...

[SYS] Checando se o ponto esta no mesmo poligono que estava antes do ultimo movimento...
[SYS] Fazendo passo inicial...
[SYS] Verificando se o ponto ainda esta dentro do poligono 14...
    Ponto esta a direita da aresta 0
    Ponto esta a direita da aresta 1
    Ponto esta a direita da aresta 2
    Ponto esta a direita da aresta 3
    Ponto esta a direita da aresta 4
    Ponto esta a direita da aresta 5
    Ponto esta a direita da aresta 6
    Sinal do produto vetorial consistente, entao o ponto esta dentro do poligono 14
    Chamadas Produto Vetorial (Passo Inicial): 7
    Sigo no mesmo poligono (14)
    Nao ha mais nada a ser feito.

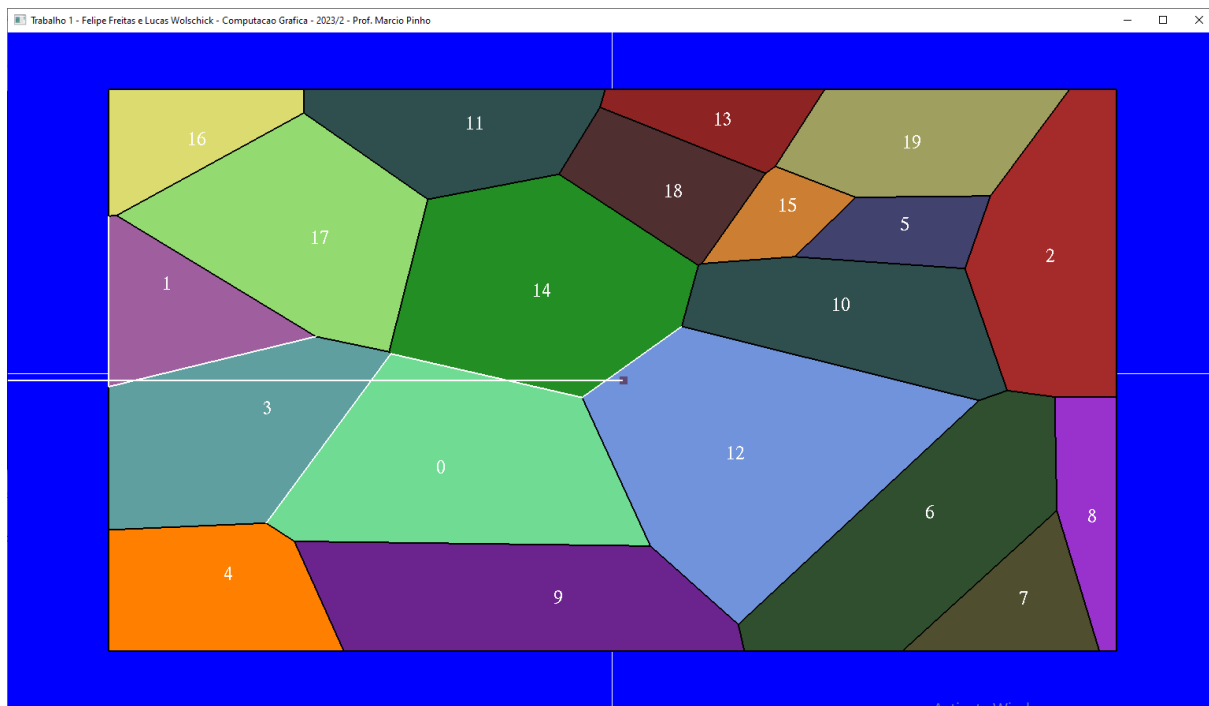
[SYS] Redesenhando a tela...
[SYS] Redesenhando os poligonos...
[SYS] Gerando cor aleatoria para o ponto...
```

3.1.3 Inclusão de pontos em polígonos côncavos

Após mover o ponto para fora do polígono anterior (14) e desenhar uma linha entre o início da tela e o ponto, vamos passar por todos os polígonos até N, onde N é o polígono para o qual me movi, conforme podemos confirmar na imagem a seguir. Como os polígonos 0, 1, 2, 3, 10 e 12 estão dentro da mesma faixa (horizontal) que a linha, vamos testar todas suas arestas procurando por intersecções, já para os polígonos de 4-9 e 11, como eles estão fora da faixa, não precisamos realizar nenhuma computação. Ao chegar no polígono 12, finalmente obtemos um número ímpar de intersecções com a linha traçada, o que nos permite concluir que estamos dentro deste polígono e nos permite passar para o próximo algoritmo. No caso deste diagrama, o total de vezes que chamamos o método `HaInterseccao` foi 34.

```
D:\Programming\GitHub\EngenhariaSoftwarePUCRS\Computacao_Grafica\Trabalho01\Voronoi\OpenGL.exe 20

[SYS] Fazendo teste de inclusao de pontos em poligonos concavos...
Aresta 0 do poligono 0 cruza a linha horizontal!
Aresta 1 do poligono 0 cruza a linha horizontal!
Aresta 2 do poligono 0 nao cruza a linha horizontal!
Aresta 3 do poligono 0 nao cruza a linha horizontal!
Aresta 4 do poligono 0 nao cruza a linha horizontal!
Aresta 0 do poligono 1 nao cruza a linha horizontal!
Aresta 1 do poligono 1 cruza a linha horizontal!
Aresta 2 do poligono 1 cruza a linha horizontal!
Aresta 3 do poligono 1 nao cruza a linha horizontal!
Aresta 0 do poligono 2 nao cruza a linha horizontal!
Aresta 1 do poligono 2 nao cruza a linha horizontal!
Aresta 2 do poligono 2 nao cruza a linha horizontal!
Aresta 3 do poligono 2 nao cruza a linha horizontal!
Aresta 4 do poligono 2 nao cruza a linha horizontal!
Aresta 5 do poligono 2 nao cruza a linha horizontal!
Aresta 6 do poligono 2 nao cruza a linha horizontal!
Aresta 0 do poligono 3 nao cruza a linha horizontal!
Aresta 1 do poligono 3 nao cruza a linha horizontal!
Aresta 2 do poligono 3 cruza a linha horizontal!
Aresta 3 do poligono 3 nao cruza a linha horizontal!
Aresta 4 do poligono 3 nao cruza a linha horizontal!
Aresta 5 do poligono 3 cruza a linha horizontal!
Ponto esta fora das faixas de interseccao do poligono 4, entao nao preciso testar o poligono 4
Ponto esta fora das faixas de interseccao do poligono 5, entao nao preciso testar o poligono 5
Ponto esta fora das faixas de interseccao do poligono 6, entao nao preciso testar o poligono 6
Ponto esta fora das faixas de interseccao do poligono 7, entao nao preciso testar o poligono 7
Ponto esta fora das faixas de interseccao do poligono 8, entao nao preciso testar o poligono 8
Ponto esta fora das faixas de interseccao do poligono 9, entao nao preciso testar o poligono 9
Aresta 0 do poligono 10 nao cruza a linha horizontal!
Aresta 1 do poligono 10 nao cruza a linha horizontal!
Aresta 2 do poligono 10 nao cruza a linha horizontal!
Aresta 3 do poligono 10 nao cruza a linha horizontal!
Aresta 4 do poligono 10 nao cruza a linha horizontal!
Aresta 5 do poligono 10 nao cruza a linha horizontal!
Aresta 6 do poligono 10 nao cruza a linha horizontal!
Ponto esta fora das faixas de interseccao do poligono 11, entao nao preciso testar o poligono 11
Aresta 0 do poligono 12 cruza a linha horizontal!
Aresta 1 do poligono 12 nao cruza a linha horizontal!
Aresta 2 do poligono 12 nao cruza a linha horizontal!
Aresta 3 do poligono 12 nao cruza a linha horizontal!
Aresta 4 do poligono 12 nao cruza a linha horizontal!
Numero de interseccoes e impar, entao o ponto esta dentro do poligono 12
Chamadas HaInterseccao (InclusaoPontosPoligonosConcavos): 34
[SYS] Continuando...
```



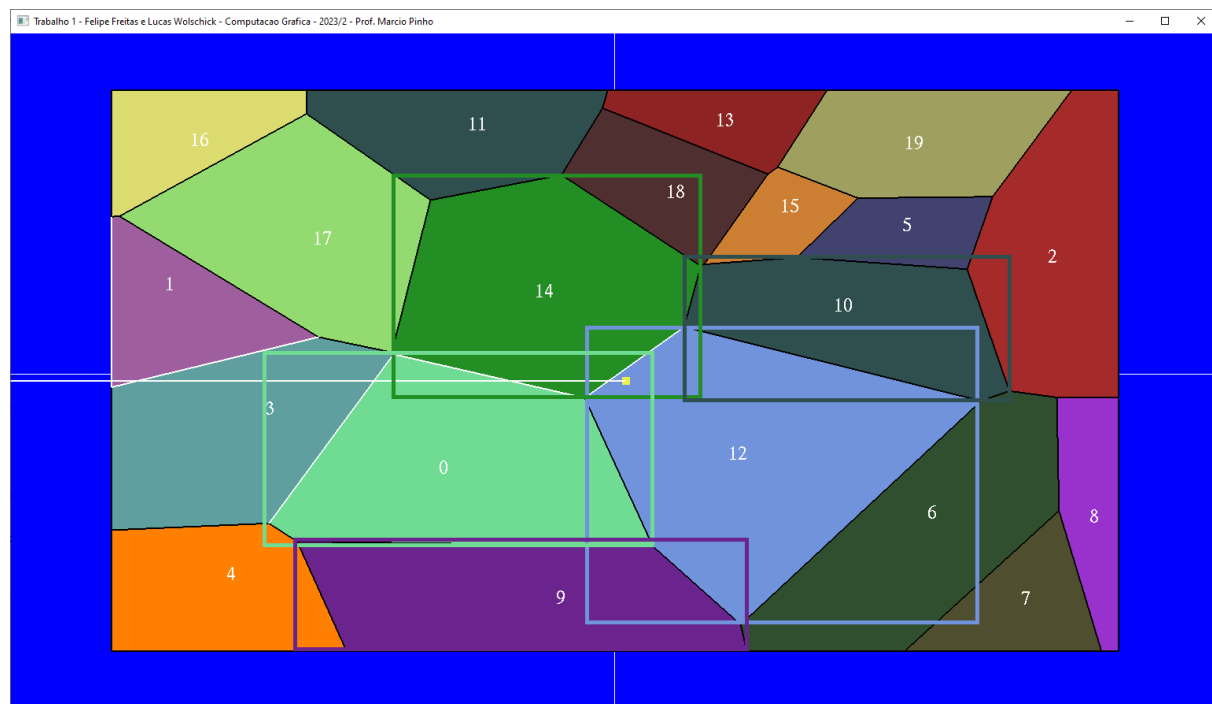
3.1.4 Inclusão de pontos em polígonos convexos

Após mover o ponto para fora do polígono anterior (14), vamos passar por todos os polígonos até N, onde N é o polígono para o qual me movi, conforme podemos confirmar na imagem a seguir. Como os envelopes dos polígonos de 1-11 não contém o ponto, não precisamos realizar nenhuma computação com estes, apenas com os polígonos 0 e 12. Aqui novamente vamos calcular o produto vetorial, mas apenas para as 5 arestas do polígono 0 e para as 5 arestas do polígono 12. Ao chegar no polígono 12 conseguimos concluir, através do resultado consistente do produto vetorial, em qual polígono estamos, tendo sido necessárias apenas 10 chamadas para a função ProdVetorial.

```
D:\Programming\GitHub\EngenhariaSoftwarePUCRS\Computacao_Grafica\Trabalho01\Voronoi\OpenGL.exe 20
Chamadas HaIntersecao (InclusaoPontosPoligonosConcavos): 34
[SYS] Continuando...

[SYS] Fazendo teste de inclusao de pontos em poligonos convexos...
Ponto esta a direita da aresta 0 do poligono 0
Ponto esta a esquerda da aresta 1 do poligono 0
Ponto esta a esquerda da aresta 2 do poligono 0
Ponto esta a direita da aresta 3 do poligono 0
Ponto esta a direita da aresta 4 do poligono 0
Sinal do produto vetorial inconsistente, entao o ponto nao esta dentro do poligono 0
Ponto nao esta dentro do envelope do poligono 1, entao nao preciso testar o poligono 1
Ponto nao esta dentro do envelope do poligono 2, entao nao preciso testar o poligono 2
Ponto nao esta dentro do envelope do poligono 3, entao nao preciso testar o poligono 3
Ponto nao esta dentro do envelope do poligono 4, entao nao preciso testar o poligono 4
Ponto nao esta dentro do envelope do poligono 5, entao nao preciso testar o poligono 5
Ponto nao esta dentro do envelope do poligono 6, entao nao preciso testar o poligono 6
Ponto nao esta dentro do envelope do poligono 7, entao nao preciso testar o poligono 7
Ponto nao esta dentro do envelope do poligono 8, entao nao preciso testar o poligono 8
Ponto nao esta dentro do envelope do poligono 9, entao nao preciso testar o poligono 9
Ponto nao esta dentro do envelope do poligono 10, entao nao preciso testar o poligono 10
Ponto nao esta dentro do envelope do poligono 11, entao nao preciso testar o poligono 11
Ponto esta a direita da aresta 0 do poligono 12
Ponto esta a direita da aresta 1 do poligono 12
Ponto esta a direita da aresta 2 do poligono 12
Ponto esta a direita da aresta 3 do poligono 12
Ponto esta a direita da aresta 4 do poligono 12
Sinal do produto vetorial consistente, entao o ponto esta dentro do poligono 12
Chamadas Produto Vetorial (InclusaoPontosPoligonosConvexos): 10
[SYS] Continuando...
```

Imagem simulada dos envelopes de alguns dos polígonos, visto que o método de desenhar envelopes não estava funcionando:



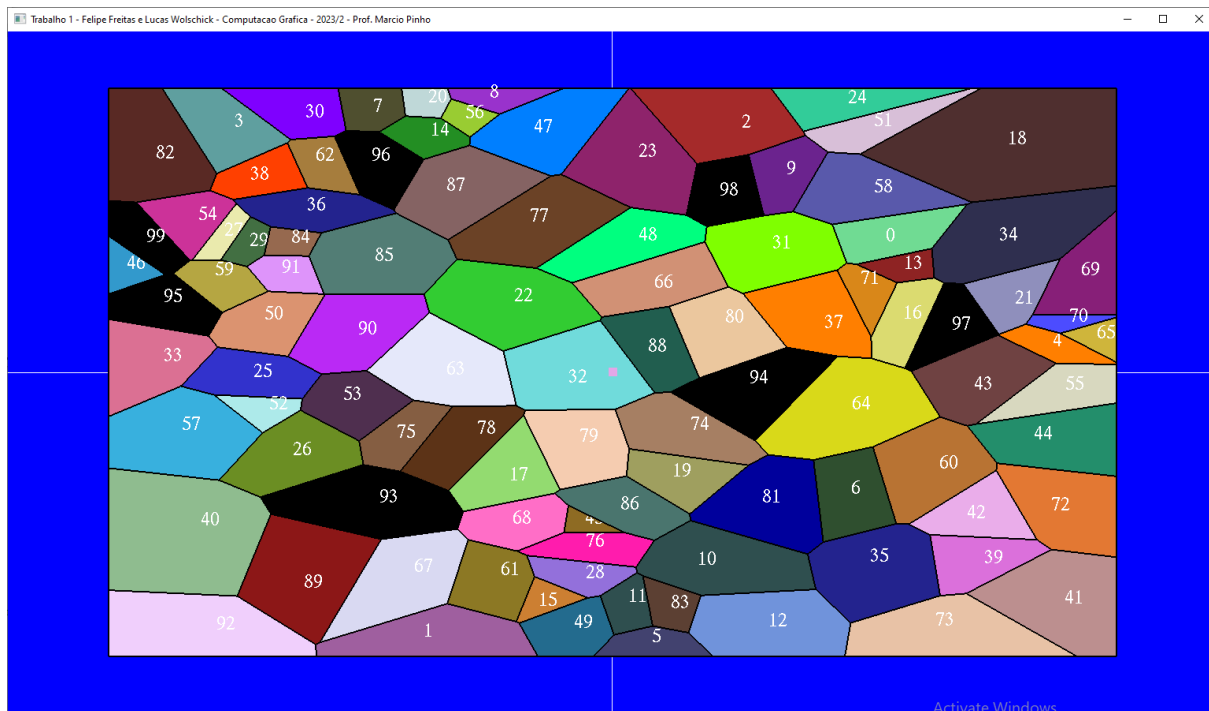
3.1.5 Inclusão de pontos em polígonos convexos com informação de vizinhança

Neste método, após sair do polígono anterior (14), vamos analisar apenas os N vizinhos do polígono – no caso, 7 - conforme a imagem anterior. Como os envelopes dos vizinhos 2-5 não contém o ponto, não precisamos realizar nenhuma computação com estes, apenas com o primeiro e último vizinho. Mais uma vez, precisamos fazer o produto vetorial nos polígonos cujos envelopes contém o ponto e, como temos poucos polígonos, a única diferença entre esta solução para o problema e a solução anterior é que esta analisa menos polígonos inicialmente, mas ambas realizam operações apenas sobre os 2 polígonos que contém o ponto. Em ambos os casos chamamos a função ProdVetorial um total de 10 vezes, porém, com a última, sabemos que saímos pela aresta de número 4, conforme imagem abaixo:

```
D:\Programming\GitHub\EngenhariaSoftwarePUCRS\Computacao_Grafica\Trabalho01\Voronoi\OpenGL.exe 20
[SYS] Fazendo teste de inclusao de pontos em poligonos convexos utilizando a informacao de vizinhanca...
Qtd de vizinhos encontrados: 7
Comparando com 1o vizinho
  Testando aresta 0 do vizinho 1
  Ponto esta a direita da aresta 0 do vizinho 1
  Testando aresta 1 do vizinho 1
  Ponto esta a esquerda da aresta 1 do vizinho 1
  Testando aresta 2 do vizinho 1
  Ponto esta a esquerda da aresta 2 do vizinho 1
  Testando aresta 3 do vizinho 1
  Ponto esta a direita da aresta 3 do vizinho 1
  Testando aresta 4 do vizinho 1
  Ponto esta a direita da aresta 4 do vizinho 1
Sinal do produto vetorial inconsistente, entao o ponto nao esta dentro do vizinho 1
Comparando com 2o vizinho
Ponto nao esta dentro do envelope do vizinho 2, entao nao preciso testar
Comparando com 3o vizinho
Ponto nao esta dentro do envelope do vizinho 3, entao nao preciso testar
Comparando com 4o vizinho
Ponto nao esta dentro do envelope do vizinho 4, entao nao preciso testar
Comparando com 5o vizinho
  Testando aresta 0 do vizinho 5
  Ponto esta a direita da aresta 0 do vizinho 5
  Testando aresta 1 do vizinho 5
  Ponto esta a direita da aresta 1 do vizinho 5
  Testando aresta 2 do vizinho 5
  Ponto esta a direita da aresta 2 do vizinho 5
  Testando aresta 3 do vizinho 5
  Ponto esta a direita da aresta 3 do vizinho 5
  Testando aresta 4 do vizinho 5
  Ponto esta a direita da aresta 4 do vizinho 5
Sinal do produto vetorial consistente, entao o ponto esta dentro do vizinho 5
O ponto foi para o poligono 12
Chamadas Produto Vetorial (InclusaoPontosPoligonosConvexos): 10
Aresta cruzada: 4
[SYS] Continuando...
=====
```

3.2 Cenário 2 - 100 Polígonos

Neste cenário existem polígonos “sem” cor (pretos), mas o resultado independe desta falta.



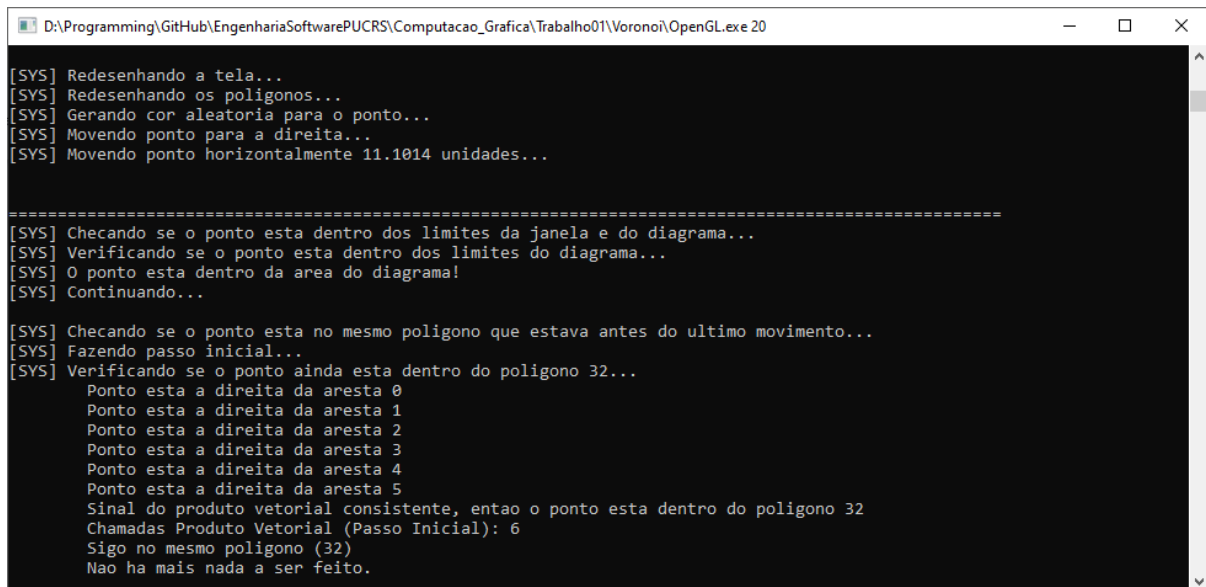
3.2.1 Método inicial

Após contarmos e salvarmos os vizinhos de todos os polígonos, foram necessárias 180 chamadas ao método ProdVetorial para determinar a posição inicial do ponto, conforme imagem abaixo.

```
D:\Programming\GitHub\EngenhariaSoftwarePUCRS\Computacao_Grafica\Trabalho01\Voronoi\OpenGL.exe 20
Programa OpenGL
[SYS] Inicializando programa...
[SYS] Inicializando Diagrama de Voronoi...
[SYS] Lendo poligonos do arquivo casosTeste/100.txt...
[SYS] Calculando limites do diagrama Voronoi...
    Largura: 1024.4
    Minimo:(-119.14, -104.272)
    Maximo:(1110.14, 1116.27)
    Meio:(495.5, 506)
[SYS] Limites do diagrama Voronoi calculados!
[SYS] Analizando informacoes de vizinhanca...
    Poligono 0 tem 6 vizinhos.
    Poligono 1 tem 6 vizinhos.
    Poligono 2 tem 6 vizinhos.
    Poligono 3 tem 4 vizinhos.
    Poligono 4 tem 6 vizinhos.
Analizando poligonos de 5-95 ...
    Poligono 96 tem 6 vizinhos.
    Poligono 97 tem 7 vizinhos.
    Poligono 98 tem 5 vizinhos.
    Poligono 99 tem 5 vizinhos.
[SYS] Informacoes de vizinhanca analisadas!
[SYS] Gerando cores dos poligonos...
[SYS] Cores dos poligonos geradas!
[SYS] Criando ponto que sera movido...
[SYS] Ponto criado!
[SYS] Computando posicao inicial do ponto...
    Analisando se estou dentro do poligono 0
[SYS] Fazendo passo inicial...
[SYS] Verificando se o ponto ainda esta dentro do poligono 0...
    Ponto esta a esquerda da aresta 0
    Ponto esta a esquerda da aresta 1
    Ponto esta a direita da aresta 2
    Ponto esta a direita da aresta 3
    Ponto esta a esquerda da aresta 4
    Sinal do produto vetorial inconsistente, entao o ponto nao esta dentro do poligono 0
    Testando se o ponto esta dentro do poligono 1...
    Analisando se estou dentro do poligono 1
[SYS] Fazendo passo inicial...
[SYS] Verificando se o ponto ainda esta dentro do poligono 1...
    Ponto esta a esquerda da aresta 0
    Ponto esta a esquerda da aresta 1
    Ponto esta a esquerda da aresta 2
    Ponto esta a esquerda da aresta 3
    Ponto esta a esquerda da aresta 4
    Ponto esta a direita da aresta 5
    Ponto esta a direita da aresta 6
Testando poligonos de 2-31 ...
    Testando se o ponto esta dentro do poligono 32...
    Analisando se estou dentro do poligono 32
[SYS] Fazendo passo inicial...
[SYS] Verificando se o ponto ainda esta dentro do poligono 32...
    Ponto esta a direita da aresta 0
    Ponto esta a direita da aresta 1
    Ponto esta a direita da aresta 2
    Ponto esta a direita da aresta 3
    Ponto esta a direita da aresta 4
    Ponto esta a direita da aresta 5
    Sinal do produto vetorial consistente, entao o ponto esta dentro do poligono 32
    Quantidade de chamadas para ProdVetorial para descobrir o poligono inicial: 180
[SYS] Posicao inicial do ponto computada! (495.5, 506)
```

3.2.2 Movimento interno

Ao contrário do cenário anterior, apesar de aqui haver mais polígonos, como o inicial possui apenas 6 arestas é necessário fazer somente 6 chamadas para determinar se o ponto ainda está no polígono inicial (32), conforme imagem abaixo.



```
D:\Programming\GitHub\EngenhariaSoftwarePUCRS\Computacao_Grafica\Trabalho01\Voronoi\OpenGL.exe 20

[SYS] Redesenhando a tela...
[SYS] Redesenhando os poligonos...
[SYS] Gerando cor aleatoria para o ponto...
[SYS] Movendo ponto para a direita...
[SYS] Movendo ponto horizontalmente 11.1014 unidades...

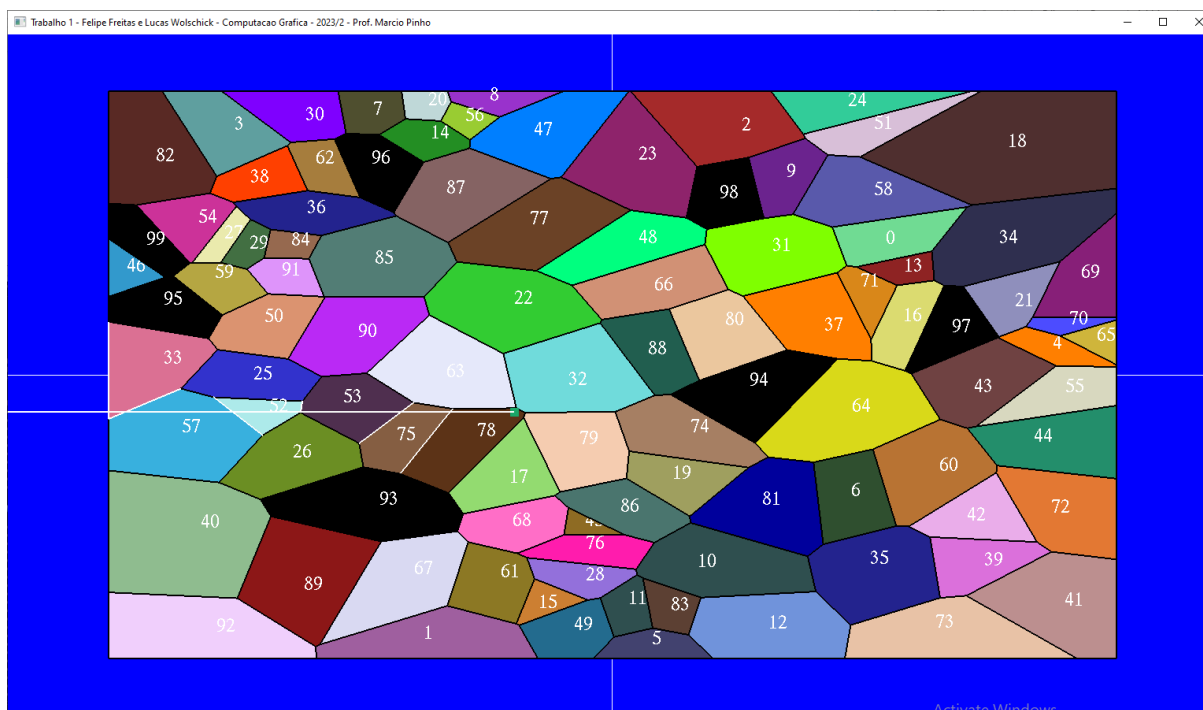
=====
[SYS] Checando se o ponto esta dentro dos limites da janela e do diagrama...
[SYS] Verificando se o ponto esta dentro dos limites do diagrama...
[SYS] O ponto esta dentro da area do diagrama!
[SYS] Continuando...

[SYS] Checando se o ponto esta no mesmo poligono que estava antes do ultimo movimento...
[SYS] Fazendo passo inicial...
[SYS] Verificando se o ponto ainda esta dentro do poligono 32...
    Ponto esta a direita da aresta 0
    Ponto esta a direita da aresta 1
    Ponto esta a direita da aresta 2
    Ponto esta a direita da aresta 3
    Ponto esta a direita da aresta 4
    Ponto esta a direita da aresta 5
    Sinal do produto vetorial consistente, entao o ponto esta dentro do poligono 32
    Chamadas Produto Vetorial (Passo Inicial): 6
    Sigo no mesmo poligono (32)
    Nao ha mais nada a ser feito.
```

3.2.3 Inclusão de pontos em polígonos côncavos

Após mover o ponto para fora do polígono anterior (32) e desenhar uma linha entre o início da tela e o ponto, vamos passar por todos os polígonos até N, onde N é o polígono para o qual me movi (78), conforme podemos confirmar na imagem a seguir. Os polígonos de 0-31, 34-42, 45-51, 54, 56, 58-63, 65-73, 76 e 77, que não estão dentro da mesma faixa (horizontal) que a linha, e o ponto 32 (que é minha origem), não precisam ser testados quanto a intersecção com a reta, nos deixando com os outros 11 polígonos (33, 43-44, 52-53, 55, 57, 64, 74-75 e 78) por testar. Ao comparar com o polígono onde estou (78), finalmente obtemos um número ímpar de intersecções com a linha traçada, o que nos leva ao próximo algoritmo. No caso deste diagrama, o total de vezes que chamamos o método `HaInterseccao` foi 63, conforme imagem abaixo.

```
D:\Programming\GitHub\EngenhariaSoftwarePUCRS\Computacao_Grafica\Trabalho01\Voronoi\OpenGL.exe 20
[SYS] Checando se o ponto esta no mesmo poligono que estava antes do ultimo movimento...
[SYS] Fazendo passo inicial...
[SYS] Verificando se o ponto ainda esta dentro do poligono 32...
    Ponto esta a direita da aresta 0
    Ponto esta a esquerda da aresta 1
    Ponto esta a esquerda da aresta 2
    Ponto esta a direita da aresta 3
    Ponto esta a direita da aresta 4
    Ponto esta a direita da aresta 5
    Sinal do produto vetorial inconsistente, entao o ponto nao esta dentro do poligono 32
    Chamadas Produto Vetorial (Passo Inicial): 6
    O ponto nao esta no mesmo poligono que estava antes do ultimo movimento!
[SYS] Continuando...
[SYS] Fazendo teste de inclusao de pontos em poligonos concavos...
    Ponto esta fora das faixas de interseccao do poligono 0, entao nao preciso testar o poligono 0
    Ponto esta fora das faixas de interseccao do poligono 1, entao nao preciso testar o poligono 1
    Analisando poligonos de 2-75 ...
    Ponto esta fora das faixas de interseccao do poligono 76, entao nao preciso testar o poligono 76
    Ponto esta fora das faixas de interseccao do poligono 77, entao nao preciso testar o poligono 77
    Aresta 0 do poligono 78 nao cruza a linha horizontal!
    Aresta 1 do poligono 78 nao cruza a linha horizontal!
    Aresta 2 do poligono 78 nao cruza a linha horizontal!
    Aresta 3 do poligono 78 cruza a linha horizontal!
    Aresta 4 do poligono 78 nao cruza a linha horizontal!
    Aresta 5 do poligono 78 nao cruza a linha horizontal!
    Numero de intersecoes e impar, entao o ponto esta dentro do poligono 78
    Chamadas HaInterseccao (InclusaoPontosPoligonosConcavos): 63
[SYS] Continuando...
```



3.2.4 Inclusão de pontos em polígonos convexos

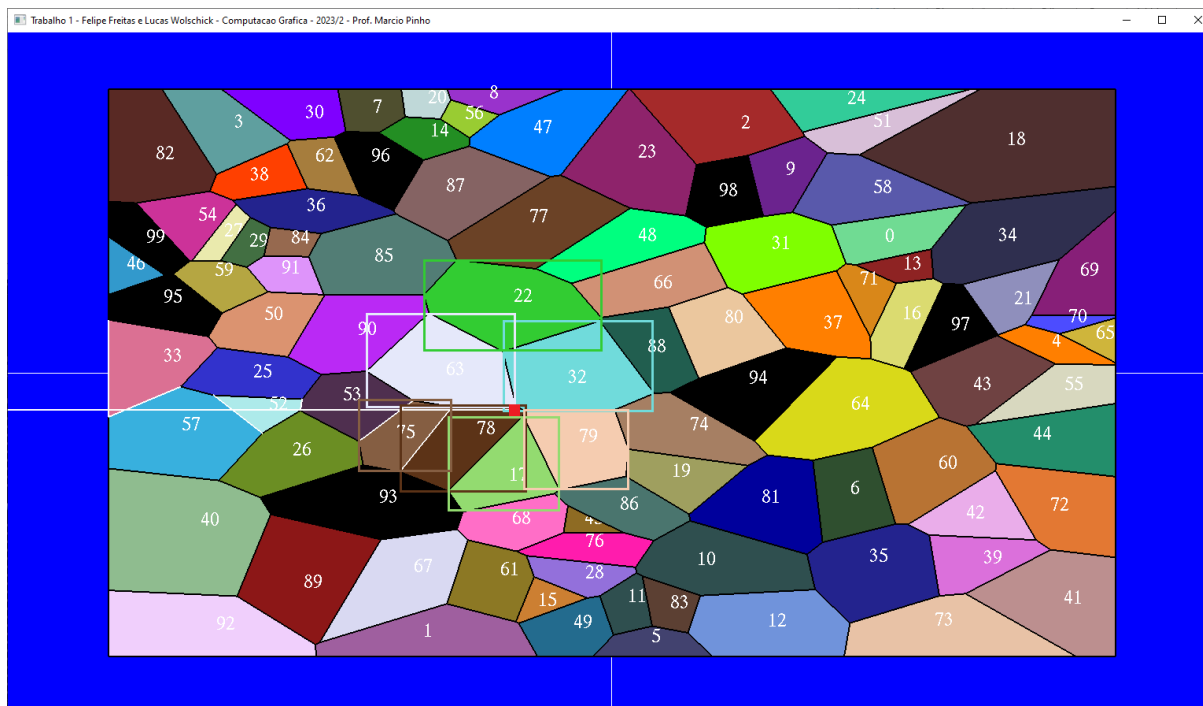
Após mover o ponto para fora do polígono anterior (32), vamos passar por todos os polígonos até N, onde N é o polígono para o qual me movi, conforme podemos confirmar na imagem a seguir. Como apenas o envelope do polígono 78 envolve o ponto, só realizamos a computação do produto vetorial com 1 polígono (embora fosse possível criar uma condição para, caso houvesse apenas 1 envelope, assumir o 'dono' deste como o polígono que contém o ponto). Certamente, tal polígono contém o ponto e, como este tem 6 lados, realizamos 6 vezes o produto vetorial e passamos para o próximo algoritmo, que deve chegar mais rápido nesta conclusão, visto que este apesar de fazer 1 computação mais 'pesada' apenas ainda precisou recusar todos os envelopes do diagrama até encontrar um que contivesse o ponto.

```
D:\Programming\GitHub\EngenhariaSoftwarePUCRS\Computacao_Grafica\Trabalho01\Voronoi\OpenGL.exe 20
[SYS] Fazendo teste de inclusao de pontos em poligonos convexos...
Ponto nao esta dentro do envelope do poligono 0, entao nao preciso testar o poligono 0
Ponto nao esta dentro do envelope do poligono 1, entao nao preciso testar o poligono 1
Ponto nao esta dentro do envelope do poligono 2, entao nao preciso testar o poligono 2
Ponto nao esta dentro do envelope do poligono 3, entao nao preciso testar o poligono 3
Ponto nao esta dentro do envelope do poligono 4, entao nao preciso testar o poligono 4
Ponto nao esta dentro do envelope do poligono 5, entao nao preciso testar o poligono 5
Ponto nao esta dentro do envelope do poligono 6, entao nao preciso testar o poligono 6
Ponto nao esta dentro do envelope do poligono 7, entao nao preciso testar o poligono 7
Ponto nao esta dentro do envelope do poligono 8, entao nao preciso testar o poligono 8
Ponto nao esta dentro do envelope do poligono 9, entao nao preciso testar o poligono 9

Analizando polígonos de 10-30...
Ponto nao esta dentro do envelope do poligono 31, entao nao preciso testar o poligono 31
Ja sei que nao estou no mesmo poligono que estava antes do ultimo movimento, entao nao preciso testar o poligono 32
Ponto nao esta dentro do envelope do poligono 33, entao nao preciso testar o poligono 33
Ponto nao esta dentro do envelope do poligono 34, entao nao preciso testar o poligono 34
Ponto nao esta dentro do envelope do poligono 35, entao nao preciso testar o poligono 35
Ponto nao esta dentro do envelope do poligono 36, entao nao preciso testar o poligono 36
Ponto nao esta dentro do envelope do poligono 37, entao nao preciso testar o poligono 37
Ponto nao esta dentro do envelope do poligono 38, entao nao preciso testar o poligono 38
Ponto nao esta dentro do envelope do poligono 39, entao nao preciso testar o poligono 39
Ponto nao esta dentro do envelope do poligono 40, entao nao preciso testar o poligono 40
Ponto nao esta dentro do envelope do poligono 41, entao nao preciso testar o poligono 41
Ponto nao esta dentro do envelope do poligono 42, entao nao preciso testar o poligono 42

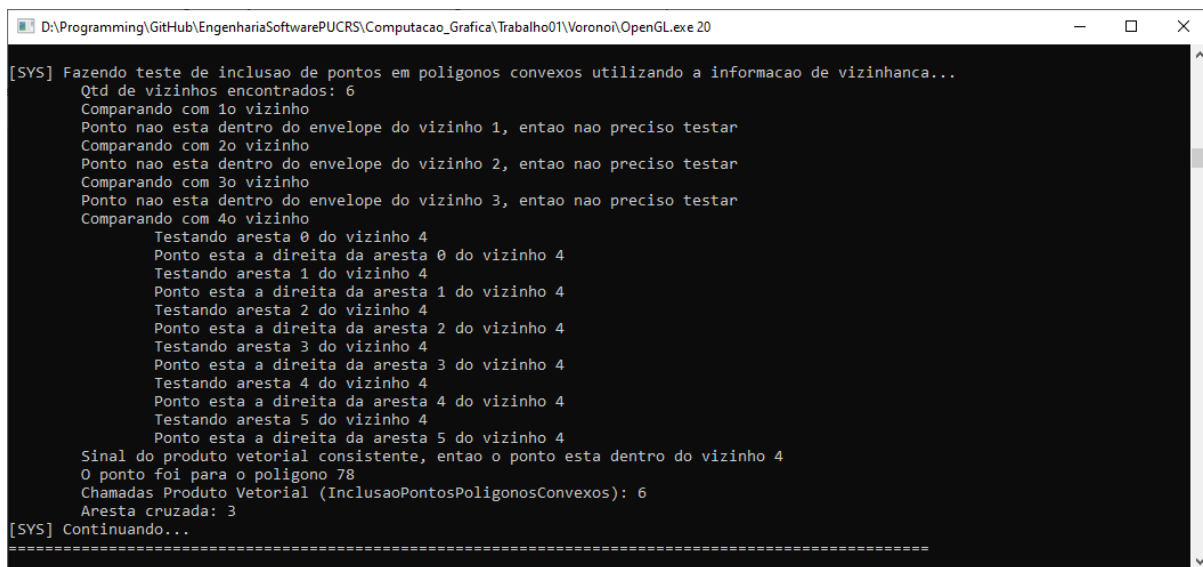
Analizando polígonos de 43-74 ...
Ponto nao esta dentro do envelope do poligono 75, entao nao preciso testar o poligono 75
Ponto nao esta dentro do envelope do poligono 76, entao nao preciso testar o poligono 76
Ponto nao esta dentro do envelope do poligono 77, entao nao preciso testar o poligono 77
Ponto esta a direita da aresta 0 do poligono 78
Ponto esta a direita da aresta 1 do poligono 78
Ponto esta a direita da aresta 2 do poligono 78
Ponto esta a direita da aresta 3 do poligono 78
Ponto esta a direita da aresta 4 do poligono 78
Ponto esta a direita da aresta 5 do poligono 78
Sinal do produto vetorial consistente, entao o ponto esta dentro do poligono 78
Chamadas Produto Vetorial (InclusaoPontosPoligonosConvexos): 6
[SYS] Continuando...
```

Imagem simulada com o ponto ampliado e vermelho para melhor visualização, com os envelopes de alguns dos polígonos:



3.2.5 Inclusão de pontos em polígonos convexos com informação de vizinhança

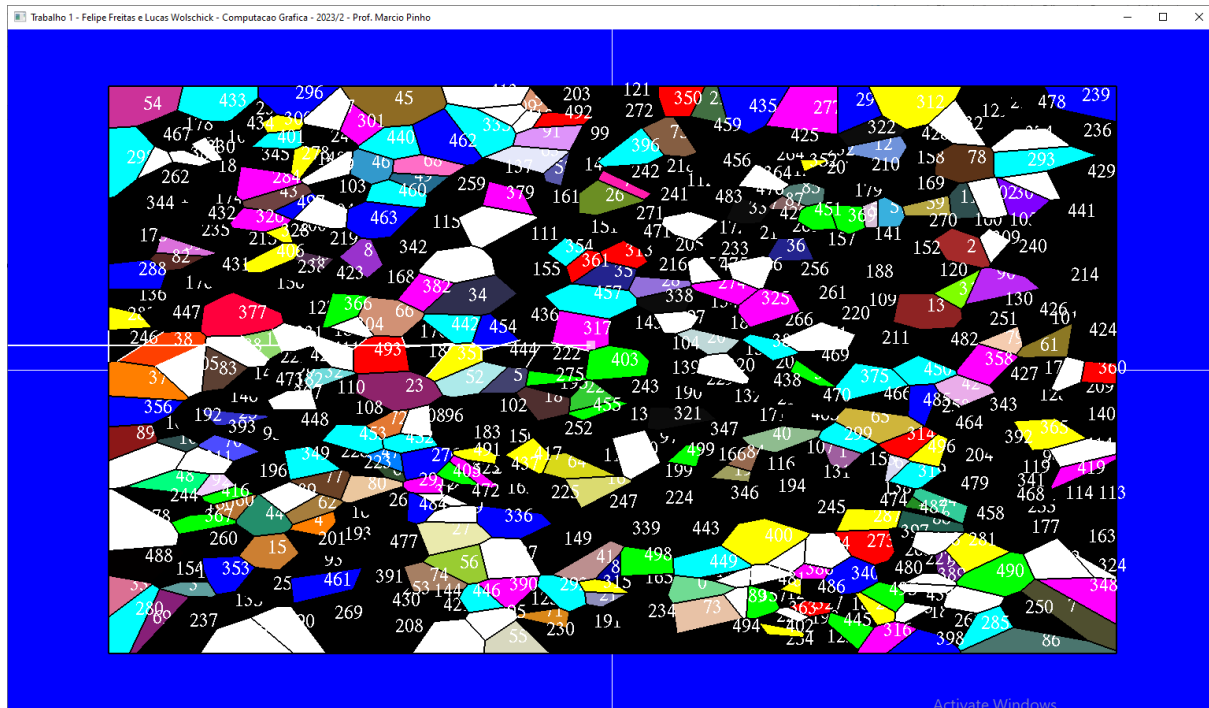
Conforme citado anteriormente, após sair do polígono anterior (32) vamos analisar os N vizinhos do polígono – no caso, 6 - conforme a imagem anterior. Novamente, os envelopes dos 3 primeiros vizinhos não contêm o ponto, então realizamos o teste no quarto vizinho. Após realizar o produto vetorial com os 6 vértices deste, concluímos que é esta aresta (a 4ª, denominada 'aresta 3' na imagem abaixo) que foi cruzada no último movimento. É nítida a diferença, agora que temos 5x mais polígonos do que no cenário 1, de realizar o teste apenas nos vizinhos e realizá-lo contra todos os polígonos do diagrama, e esta diferença apenas cresce conforme aumenta o tamanho dos diagramas.



```
D:\Programming\GitHub\EngenhariaSoftwarePUCRS\Computacao_Grafica\Trabalho01\Voronoi\OpenGL.exe 20
[SYS] Fazendo teste de inclusao de pontos em poligonos convexos utilizando a informacao de vizinhanca...
      Qtd de vizinhos encontrados: 6
      Comparando com 1o vizinho
      Ponto nao esta dentro do envelope do vizinho 1, entao nao preciso testar
      Comparando com 2o vizinho
      Ponto nao esta dentro do envelope do vizinho 2, entao nao preciso testar
      Comparando com 3o vizinho
      Ponto nao esta dentro do envelope do vizinho 3, entao nao preciso testar
      Comparando com 4o vizinho
          Testando aresta 0 do vizinho 4
          Ponto esta a direita da aresta 0 do vizinho 4
          Testando aresta 1 do vizinho 4
          Ponto esta a direita da aresta 1 do vizinho 4
          Testando aresta 2 do vizinho 4
          Ponto esta a direita da aresta 2 do vizinho 4
          Testando aresta 3 do vizinho 4
          Ponto esta a direita da aresta 3 do vizinho 4
          Testando aresta 4 do vizinho 4
          Ponto esta a direita da aresta 4 do vizinho 4
          Testando aresta 5 do vizinho 4
          Ponto esta a direita da aresta 5 do vizinho 4
      Sinal do produto vetorial consistente, entao o ponto esta dentro do vizinho 4
      O ponto foi para o poligono 78
      Chamadas Produto Vetorial (InclusaoPontosPoligonosConvexos): 6
      Aresta cruzada: 3
[SYS] Continuando...
=====
```

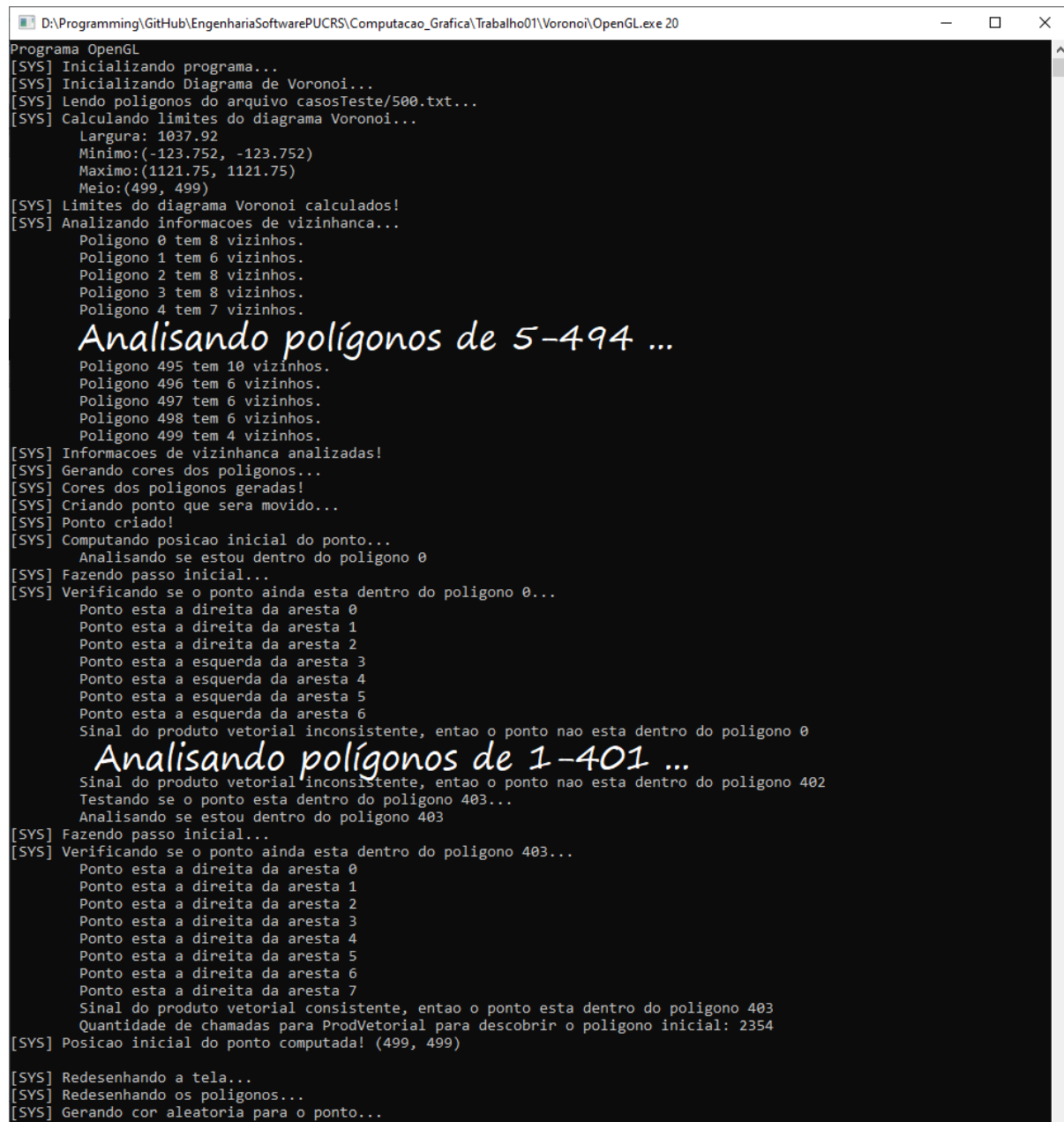

3.3 Cenário 3 - 500 Polígonos

Neste cenário existem diversos polígonos sem cor (pretos), mas novamente isto não impacta no resultado ou na análise.



3.3.1 Método inicial

Após contarmos e salvarmos os vizinhos de todos os polígonos, foram necessárias 2354 chamadas ao método ProdVetorial para determinar a posição inicial do ponto, conforme imagem abaixo.



```
Programa OpenGL
[SYS] Inicializando programa...
[SYS] Inicializando Diagrama de Voronoi...
[SYS] Lendo poligonos do arquivo casosTeste/500.txt...
[SYS] Calculando limites do diagrama Voronoi...
    Largura: 1037.92
    Minimo: (-123.752, -123.752)
    Maximo: (1121.75, 1121.75)
    Meio: (499, 499)
[SYS] Limites do diagrama Voronoi calculados!
[SYS] Analizando informacoes de vizinhanca...
    Poligono 0 tem 8 vizinhos.
    Poligono 1 tem 6 vizinhos.
    Poligono 2 tem 8 vizinhos.
    Poligono 3 tem 8 vizinhos.
    Poligono 4 tem 7 vizinhos.
    Poligono 495 tem 10 vizinhos.
    Poligono 496 tem 6 vizinhos.
    Poligono 497 tem 6 vizinhos.
    Poligono 498 tem 6 vizinhos.
    Poligono 499 tem 4 vizinhos.
[SYS] Informacoes de vizinhanca analisadas!
[SYS] Gerando cores dos poligonos...
[SYS] Cores dos poligonos geradas!
[SYS] Criando ponto que sera movido...
[SYS] Ponto criado!
[SYS] Computando posicao inicial do ponto...
    Analisando se estou dentro do poligono 0
[SYS] Fazendo passo inicial...
[SYS] Verificando se o ponto ainda esta dentro do poligono 0...
    Ponto esta a direita da aresta 0
    Ponto esta a direita da aresta 1
    Ponto esta a direita da aresta 2
    Ponto esta a esquerda da aresta 3
    Ponto esta a esquerda da aresta 4
    Ponto esta a esquerda da aresta 5
    Ponto esta a esquerda da aresta 6
    Sinal do produto vetorial inconsistente, entao o ponto nao esta dentro do poligono 0
    Poligono 402
    Sinal do produto vetorial inconsistente, entao o ponto nao esta dentro do poligono 402
    Testando se o ponto esta dentro do poligono 403...
    Analisando se estou dentro do poligono 403
[SYS] Fazendo passo inicial...
[SYS] Verificando se o ponto ainda esta dentro do poligono 403...
    Ponto esta a direita da aresta 0
    Ponto esta a direita da aresta 1
    Ponto esta a direita da aresta 2
    Ponto esta a direita da aresta 3
    Ponto esta a direita da aresta 4
    Ponto esta a direita da aresta 5
    Ponto esta a direita da aresta 6
    Ponto esta a direita da aresta 7
    Sinal do produto vetorial consistente, entao o ponto esta dentro do poligono 403
    Quantidade de chamadas para ProdVetorial para descobrir o poligono inicial: 2354
[SYS] Posicao inicial do ponto computada! (499, 499)

[SYS] Redesenhando a tela...
[SYS] Redesenhando os poligonos...
[SYS] Gerando cor aleatoria para o ponto...
```

3.3.2 Movimento interno

Independente do cenário, é certo que o único fator importante para o método inicial é a quantidade de lados do polígono inicial e, como este possui 8 arestas, são necessárias 8 comparações para garantir que o ponto se mantém no polígono 403.

```
D:\Programming\GitHub\EngenhariaSoftwarePUCRS\Computacao_Grafica\Trabalho01\Voronoi\OpenGL.exe 20
[SYS] Redesenhando os poligonos...
[SYS] Gerando cor aleatoria para o ponto...
[SYS] Movendo ponto para a diagonal superior esquerda...
[SYS] Movendo ponto horizontalmente -11.2175 unidades...
[SYS] Movendo ponto verticalmente 11.2175 unidades...

=====
[SYS] Checando se o ponto esta dentro dos limites da janela e do diagrama...
[SYS] Verificando se o ponto esta dentro dos limites do diagrama...
[SYS] O ponto esta dentro da area do diagrama!
[SYS] Continuando...

[SYS] Checando se o ponto esta no mesmo poligono que estava antes do ultimo movimento...
[SYS] Fazendo passo inicial...
[SYS] Verificando se o ponto ainda esta dentro do poligono 403...
    Ponto esta a direita da aresta 0
    Ponto esta a direita da aresta 1
    Ponto esta a direita da aresta 2
    Ponto esta a direita da aresta 3
    Ponto esta a direita da aresta 4
    Ponto esta a direita da aresta 5
    Ponto esta a direita da aresta 6
    Ponto esta a direita da aresta 7
    Sinal do produto vetorial consistente, entao o ponto esta dentro do poligono 403
    Chamadas Produto Vetorial (Passo Inicial): 8
    Sigo no mesmo poligono (403)
    Nao ha mais nada a ser feito.
```

3.3.3 Inclusão de pontos em polígonos côncavos

Após mover o ponto para fora do polígono anterior (403) e desenhar uma linha entre o início da tela e o ponto, passamos por até N pontos, sendo N o número do ponto em que estou, conforme podemos confirmar na imagem a seguir. Desta vez, a linha cruzou pelos envelopes dos polígonos 17, 20, 38, 61, 79, 101, 104, 143, 159, 170, 185-186, 211, 222, 246, 305 e 317 (um total de 17), enquanto todos os outros 300 foram poupados de qualquer computação mais extensa. Neste diagrama, o total de vezes que chamamos o método `HaInterseccao` foi apenas 94, o que é um tanto quanto surpreendente, visto que são 500 polígonos ao todo dos quais 318 foram comparados, conforme imagem abaixo.

```
D:\Programming\GitHub\EngenhariaSoftwarePUCRS\Computacao_Grafica\Trabalho01\Voronoi\OpenGL.exe 20
=====
[SYS] Checando se o ponto esta dentro dos limites da janela e do diagrama...
[SYS] Verificando se o ponto esta dentro dos limites do diagrama...
[SYS] O ponto esta dentro da area do diagrama!
[SYS] Continuando...

[SYS] Checando se o ponto esta no mesmo poligono que estava antes do ultimo movimento...
[SYS] Fazendo passo inicial...
[SYS] Verificando se o ponto ainda esta dentro do poligono 403...
    Ponto esta a direita da aresta 0
    Ponto esta a direita da aresta 1
    Ponto esta a direita da aresta 2
    Ponto esta a esquerda da aresta 3
    Ponto esta a direita da aresta 4
    Ponto esta a direita da aresta 5
    Ponto esta a direita da aresta 6
    Ponto esta a direita da aresta 7
    Sinal do produto vetorial inconsistente, entao o ponto nao esta dentro do poligono 403
    Chamadas Produto Vetorial (Passo Inicial): 8
    O ponto nao esta no mesmo poligono que estava antes do ultimo movimento!
[SYS] Continuando...

[SYS] Fazendo teste de inclusao de pontos em poligonos concavos...
    Ponto esta fora das faixas de interseccao do poligono 0, entao nao preciso testar o poligono 0
    Ponto esta fora das faixas de interseccao do poligono 1, entao nao preciso testar o poligono 1
    Ponto esta fora das faixas de interseccao do poligono 2, entao nao preciso testar o poligono 2
    Ponto esta fora das faixas de interseccao do poligono 3, entao nao preciso testar o poligono 3
    Ponto esta fora das faixas de interseccao do poligono 4, entao nao preciso testar o poligono 4
    Ponto esta fora das faixas de interseccao do poligono 5, entao nao preciso testar o poligono 5
    Ponto esta fora das faixas de interseccao do poligono 6, entao nao preciso testar o poligono 6
    Ponto esta fora das faixas de interseccao do poligono 7, entao nao preciso testar o poligono 7
    Ponto esta fora das faixas de interseccao do poligono 8, entao nao preciso testar o poligono 8
    Ponto esta fora das faixas de interseccao do poligono 9, entao nao preciso testar o poligono 9
    Ponto esta fora das faixas de interseccao do poligono 10, entao nao preciso testar o poligono 10
    Ponto esta fora das faixas de interseccao do poligono 304, entao nao preciso testar o poligono 304
    Aresta 0 do poligono 305 nao cruza a linha horizontal!
    Aresta 1 do poligono 305 nao cruza a linha horizontal!
    Aresta 2 do poligono 305 cruza a linha horizontal!
    Aresta 3 do poligono 305 cruza a linha horizontal!
    Aresta 4 do poligono 305 nao cruza a linha horizontal!
    Ponto esta fora das faixas de interseccao do poligono 306, entao nao preciso testar o poligono 306
    Ponto esta fora das faixas de interseccao do poligono 307, entao nao preciso testar o poligono 307
    Ponto esta fora das faixas de interseccao do poligono 308, entao nao preciso testar o poligono 308
    Ponto esta fora das faixas de interseccao do poligono 309, entao nao preciso testar o poligono 309
    Ponto esta fora das faixas de interseccao do poligono 310, entao nao preciso testar o poligono 310
    Ponto esta fora das faixas de interseccao do poligono 311, entao nao preciso testar o poligono 311
    Ponto esta fora das faixas de interseccao do poligono 312, entao nao preciso testar o poligono 312
    Ponto esta fora das faixas de interseccao do poligono 313, entao nao preciso testar o poligono 313
    Ponto esta fora das faixas de interseccao do poligono 314, entao nao preciso testar o poligono 314
    Ponto esta fora das faixas de interseccao do poligono 315, entao nao preciso testar o poligono 315
    Ponto esta fora das faixas de interseccao do poligono 316, entao nao preciso testar o poligono 316
    Aresta 0 do poligono 317 cruza a linha horizontal!
    Aresta 1 do poligono 317 nao cruza a linha horizontal!
    Aresta 2 do poligono 317 nao cruza a linha horizontal!
    Aresta 3 do poligono 317 nao cruza a linha horizontal!
    Aresta 4 do poligono 317 nao cruza a linha horizontal!
    Numero de interseccoos e impar, entao o ponto esta dentro do poligono 317
    Chamadas HaInterseccao (InclusaoPontosPoligonosConcavos): 94
[SYS] Continuando...
```



3.3.4 Inclusão de pontos em polígonos convexos

Após mover o ponto para fora do polígono anterior (403), vamos passar por todos os polígonos até N, onde N é o polígono para o qual me movi (no caso, 317), conforme podemos confirmar na imagem a seguir. Como no cenário anterior, apenas 1 envelope contém o ponto, porém, foi necessário verificar a colisão com todos os 316 envelopes anteriores para chegar a esta conclusão. Como o polígono contém 5 lados, realizamos apenas 5 vezes a chamadas da função ProdVetorial e seguimos adiante para o último algoritmo testado, conforme imagem abaixo.

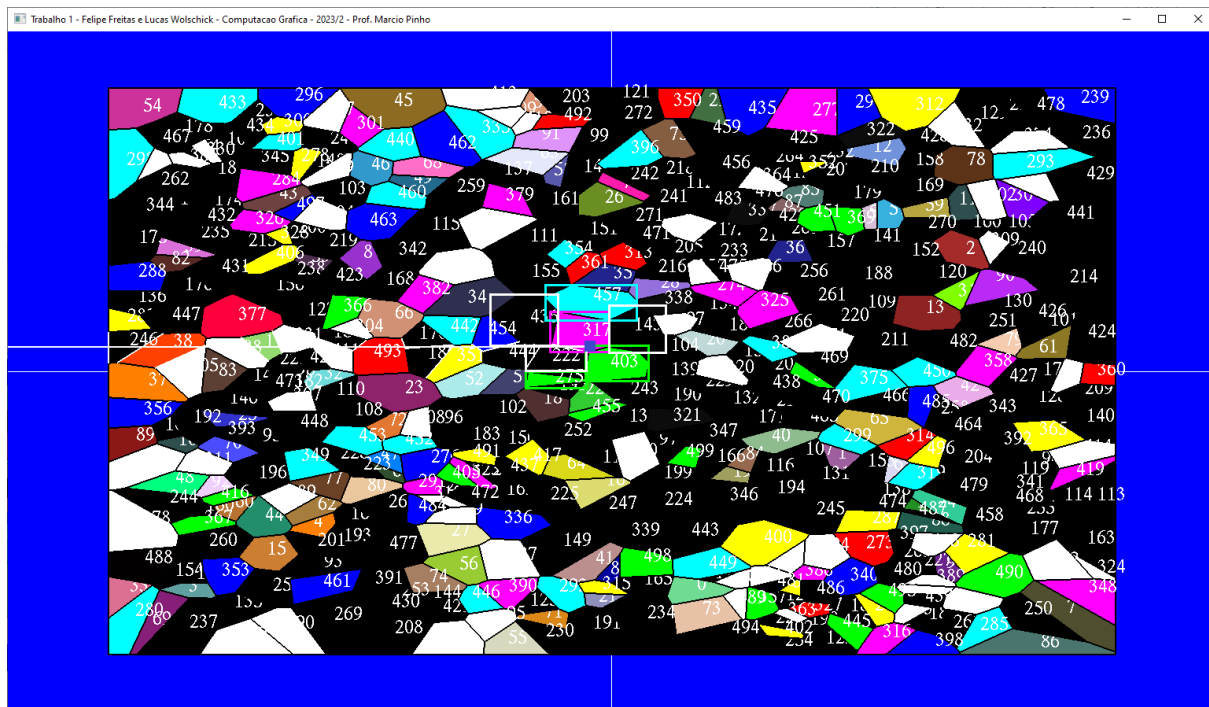
```

D:\Programming\GitHub\EngenhariaSoftwarePUCRS\Computacao_Grafica\Trabalho01\Voronoi\OpenGL.exe 20
[SYS] Continuando...

[SYS] Fazendo teste de inclusao de pontos em poligonos convexos...
Ponto nao esta dentro do envelope do poligono 0, entao nao preciso testar o poligono 0
Ponto nao esta dentro do envelope do poligono 1, entao nao preciso testar o poligono 1
Ponto nao esta dentro do envelope do poligono 2, entao nao preciso testar o poligono 2
Ponto nao esta dentro do envelope do poligono 3, entao nao preciso testar o poligono 3
Ponto nao esta dentro do envelope do poligono 4, entao nao preciso testar o poligono 4

Analizando polígonos de 5-312 ...
Ponto nao esta dentro do envelope do poligono 313, entao nao preciso testar o poligono 313
Ponto nao esta dentro do envelope do poligono 314, entao nao preciso testar o poligono 314
Ponto nao esta dentro do envelope do poligono 315, entao nao preciso testar o poligono 315
Ponto nao esta dentro do envelope do poligono 316, entao nao preciso testar o poligono 316
Ponto esta a direita da aresta 0 do poligono 317
Ponto esta a direita da aresta 1 do poligono 317
Ponto esta a direita da aresta 2 do poligono 317
Ponto esta a direita da aresta 3 do poligono 317
Ponto esta a direita da aresta 4 do poligono 317
Sinal do produto vetorial consistente, entao o ponto esta dentro do poligono 317
Chamadas Produto Vetorial (InclusaoPontosPoligonosConvexos): 5
[SYS] Continuando...
  
```

Imagem simulada com o ponto ampliado e azul escuro para melhor visualização, com os envelopes de alguns dos polígonos:



3.3.5 Inclusão de pontos em polígonos convexos com informação de vizinhança

Finalmente, após sair do polígono anterior (403) vamos percorrer seus N (8) vizinhos, como mostrado na imagem a seguir. Desta vez, tivemos o azar de ter cruzado a aresta do último vizinho, mas como os envelopes dos outros não continham o ponto, tivemos de realizar o teste do produto vetorial apenas 5 vezes, uma para cada aresta deste último. Conforme imagem abaixo, cruzamos a 7ª (última) aresta do polígono e terminamos os testes.

```

D:\Programming\GitHub\EngenhariaSoftwarePUCRS\Computacao_Grafica\Trabalho01\Voronoi\OpenGL.exe 20
[SYS] Fazendo teste de inclusao de pontos em poligonos convexos utilizando a informacao de vizinhanca...
  Qtd de vizinhos encontrados: 8
  Comparando com 1o vizinho
  Ponto nao esta dentro do envelope do vizinho 1, entao nao preciso testar
  Comparando com 2o vizinho
  Ponto nao esta dentro do envelope do vizinho 2, entao nao preciso testar
  Comparando com 3o vizinho
  Ponto nao esta dentro do envelope do vizinho 3, entao nao preciso testar
  Comparando com 4o vizinho
  Ponto nao esta dentro do envelope do vizinho 4, entao nao preciso testar
  Comparando com 5o vizinho
  Ponto nao esta dentro do envelope do vizinho 5, entao nao preciso testar
  Comparando com 6o vizinho
  Ponto nao esta dentro do envelope do vizinho 6, entao nao preciso testar
  Comparando com 7o vizinho
  Ponto nao esta dentro do envelope do vizinho 7, entao nao preciso testar
  Comparando com 8o vizinho
    Testando aresta 0 do vizinho 8
    Ponto esta a direita da aresta 0 do vizinho 8
    Testando aresta 1 do vizinho 8
    Ponto esta a direita da aresta 1 do vizinho 8
    Testando aresta 2 do vizinho 8
    Ponto esta a direita da aresta 2 do vizinho 8
    Testando aresta 3 do vizinho 8
    Ponto esta a direita da aresta 3 do vizinho 8
    Testando aresta 4 do vizinho 8
    Ponto esta a direita da aresta 4 do vizinho 8
  Sinal do produto vetorial consistente, entao o ponto esta dentro do vizinho 8
  O ponto foi para o poligono 317
  Chamadas Produto Vetorial (InclusaoPontosPoligonosConvexos): 5
  Aresta cruzada: 7
[SYS] Continuando...
=====

```

4. Conclusões

Após realizar todas estas análises, podemos concluir que a complexidade dos algoritmos e a quantidade de cálculos mais custosos que devem ser realizados não crescem linearmente de acordo com a quantidade de polígonos, mas claramente existe uma diferença grande de complexidade entre eles e é muito vantajoso utilizar o 'algoritmo de Voronoi', isto é, levar em consideração apenas a vizinhança. Para que o algoritmo não fizesse sentido seria necessário ou que o ponto pudesse 'pular' por cima de um vizinho, isto é, atravessá-lo em um único movimento, ou então o polígono o qual se compara os vizinhos ter mais lados do que o diagrama tem de polígonos, o que não deve acontecer com frequência.