MAC 5747 GEOMETRIA COMPUTACIONAL

LISTA DE EXERCÍCIOS 3

1. Instruções

Entregue os exercícios 5, 6 e 8 na aula do dia ?? de abril, quarta-feira.

2. Exercícios

(1) (Área de um polígono) Seja P um polígono (convexo ou não) com vértices $v_0, v_1, \ldots, v_{n-1}$ listados em ordem anti-horária ao percorrermos a fronteira de P. Mostre que se $v_i = (x_i, y_i)$ $(i = 0, \ldots, n-1)$ então

$$2\mathcal{A}(P) = \sum_{i=0}^{n-1}(x_i y_{i+1} - y_i x_{i+1}),$$

onde $\mathcal{A}(P)$ denota a área do polígono P.

(2) (Área de um polígono) Seja P um polígono (convexo ou não) com vértices $v_0, v_1, \ldots, v_{n-1}$ listados em ordem anti-horária ao percorrermos a fronteira de P. Escreva em `C` uma função que calcula a área de P. Qual a complexidade de tempo da sua função?

(3) (Exercício 1.6.4.1 de [1] — *Integer overflow*) Nos computadores onde variáveis do tipo inteiro podem assumir valores entre -2^{31} e $+2^{31}$, qual é o maior valor que podemos atribuir às coordenadas dos pontos a, b e c para que não ocorra *integer overflow* na função *Area2* (veja Primitivas)?

(4) Descreva em `C` a implementação de um polígono através de listas circulares duplamente ligadas. Reescreva as funções do "Capítulo" Primitivas usando a sua implementação.

(5) Qual a redundância existente no código da função `IntersectProp`? Como remover esta redundância?

(6) (Exercício 1.6.4.2 de [1]) Descreva o que predicado `IntersectProp` faz se o comando **if** é removido. Note que depois desta remoção o predicado `Intersect` ainda funciona.

(7) (Exercício 1.6.4.3 de [1] — ineficiência de `Intersect`) Simule `Intersect` e determine o número máximo de chamadas à `Area2` que podem ser feitas. Apresente uma nova implementação que evita chamadas duplicadas.

(8) (Exercício 1.6.4.5 de [1] — `InCone`) Prove que ab está no cone no vértice a se e somente se no máximo uma das seguintes chamadas de `Left` devolve FALSE: `Left`(a, a_+, b), `Left`(a, b, a_-), `Left`(a, a_-, a_+), onde a_+ e a_- são os vértices sucessor e predecessor de a na fronteira de P, respectivamente.

(9) (Exercício 1.6.4.5 de [1] — `Diagonal`) Mostre que uma das chamadas de `InCone` e `Diagonal` pode sem removida ser alterar o resultado.

Referências

[1] J. O'Rourke, *Computational geometry in C*, Cambridge University Press, Cambridge, 1993, Second Edition, 1998.