Smooth

A área de entretenimento tem apresentado tecnologias cada vez mais avançadas. Na última CES, em 2013, novos modelos de TVs com resolução de 8K (ou *Super Hi-Vision*) foram demonstrados. Isso significa que esses equipamentos têm 16 vezes "melhor" resolução que o famoso *FullHD*.

Além disso, a partir da incorporação de microprocessadores e microcontroladores nesses aparelhos, será muito comum a execução de algoritmos de filtros e estênceis nas imagens que eles projetam. Todos estes algoritmos já são bem conhecidos da área de processamento de imagem.

Um destes estênceis é denominado *smooth*. Seu principal objetivo é eliminar um pixel não representativo na imagem em relação aos pixels vizinhos, ou seja, eliminar ruídos da imagem.

O exemplo mais conhecido dessa técnica *smooth* é o algoritmo que utiliza um grupo de pixels de tamanho 3x3 e remove o ruído por meio de uma média aritmética, como pode ser visto na Figura 1.

x_0	x_1	x_2
x_3	x_4	x_5
x_6	x_7	x_8

$$x_4' = \frac{x_0 + x_1 + x_2 + x_3 + x_4 + x_5 + x_6 + x_7 + x_8}{9}$$

Figura 1. x_4' é o novo valor para o pixel x_4 em um grupo de 3x3 pixels.

Porém, esse algoritmo só funcionaria diretamente no pixel de uma imagem caso ela estivesse em escala de cinza, limitando seu uso em ambientes reais.

Um pixel de uma imagem colorida pode ser decomposta em diversos modelos de cores: RGBA, CMYK, etc. Assim, cada pixel é decomposto em cores primárias, conforme o modelo de cores. Por exemplo, um pixel da cor branco pode ser representado, no modelo RGBA (*Red-Green-Blue-Alpha*), como uma quadrupla (R, G, B, A), sendo seus valores (255, 255, 255, 0).

Para aplicar o *smooth* nessa imagem colorida, separa-se cada valor da cor primária do pixel e aplica-se o algoritmo. Ou seja, aplica-se o algoritmo de *smooth* para o valor R do

pixel, depois para o valor G do pixel, depois para o valor B e, por fim, para o A. Dessa forma, o novo pixel colorido calculado tem novos valores para (R, G, B, A). A Figura 2 mostra esse cálculo apenas para o valor de R de um pixel.

r_0	r_1	r_2
r_3	r_4	r_5
r_6	r_7	r_8

$$r_4' = \frac{r_0 + r_1 + r_2 + r_3 + r_4 + r_5 + r_6 + r_7 + r_8}{9}$$

Figura 2. r_4' é o novo valor de R para o pixel colorido x_4' em um grupo de 3x3 pixels.

Além disso, é necessário resolver o cálculo do *smooth* para os pixels de borda. Pelo exemplo da Figura 2, percebe-se que alguns pixels não existiriam na imagem real quando r_4 for um pixel de borda. Nesse caso, várias soluções podem ser adotadas, mas a forma mais simples é adotar algum valor fixo para a quadrupla (R, G, B, A) como, por exemplo, (0, 0, 0, 0).

Seu Desafio é escrever uma versão paralela e/ou distribuída do algoritmo de *smooth* que utiliza média aritmética em um grupo de 5x5 pixels e em uma imagem colorida com resolução de até 8K (ou *Super Hi-Vision*).

Entrada

O arquivo de entrada está em formato binário (*little-endian*) com apenas uma imagem. Os dois primeiros valores do arquivo, de tamanho de 16 bits, são a largura ($1 \leq X \leq 7680$) e a altura da imagem ($1 \leq Y \leq 4320$).

Os pixels são coloridos, de 32 bits cada, e são os próximos valores do arquivo. Cada pixel está no formato RGBA, conforme mostra a Figura 3 ($0 \leq R,G,B,A \leq 255$).

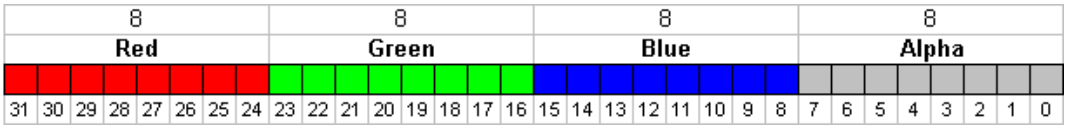

Figura 3. Formato de um pixel no modelo de cores RGBA.

Os X ($1 \leq X \leq 7680$) primeiros pixels formam a primeira linha da imagem, seguidos de X pixels para a segunda linha da imagem, num total de Y linhas ($1 \leq Y \leq 4320$).

Além disso, assuma valores para (R, G, B, A) = (0, 0, 0, 0) para o cálculo dos pixels de borda.

Os dados devem ser lidos de um arquivo denominado image.in.

Saída

A saída deve ser feita em um arquivo binário, mantendo a mesma estrutura citada para o arquivo de entrada: apenas uma imagem, os dois primeiros valores de 16 bits são a largura e a altura da imagem, seguidos dos pixels da imagem com 32 bits cada, no formato RGBA, com *X* pixels por linha (total de *Y* linhas).

Os resultados do programa devem ser escritos em um arquivo denominado image.out.