## Compressão de Imagens

A compressão de imagens, a arte e a ciência de reduzir o volume de dados necessários para representar uma imagem, é uma das tecnologias mais úteis e comercialmente bem-sucedidas na área do processamento digital de imagens. O número de imagens comprimidas e descomprimidas diariamente é impressionante e o processo de compressão e descompressão é praticamente invisível ao usuário. Qualquer pessoa que tenha uma câmera digital, que navegue pela Internet ou que assista aos mais recentes filmes de Hollywood em *Digital Video Disks* (DVDs) se beneficia dos algoritmos e padrões que analisamos neste capítulo.

Para entender melhor a necessidade de compactar representações de imagens, pense no volume de dados necessários para armazenar um vídeo digital de duas horas no padrão SD (*standard definition*) com resolução de 720 × 480 × 24 bits. Um filme digital (ou *vídeo*) é uma sequência de *quadros de vídeo* na qual cada quadro é uma imagem estática colorida. Como os reprodutores de vídeo devem exibir os quadros em sequência em velocidades de aproximadamente 30 fps (quadros por segundo, de *frames per second*), os dados de vídeos digitais SD devem ser acessados em

$$30\frac{\text{frames}}{\text{s}}\times(720\times480)\frac{\text{pixels}}{\text{frames}}\times3\frac{\text{bytes}}{\text{pixels}}=31.104.000\ \text{bytes/s}$$

e um filme de duas horas consiste em

$$31.104.000\frac{\text{bytes}}{\text{s}}\times(60^2)\frac{\text{s}}{\text{h}}\times2\text{h}\cong2,24\times10^{11}\text{bytes}$$

ou 224 GB (gigabytes) de dados. Vinte e sete DVDs de camada dupla de 8,5 GB (considerando discos convencionais de 12 cm) seriam necessários para armazená-lo. Para que um filme de duas horas caiba em um único DVD, cada quadro deve ser comprimido — em média — por um fator de 26,3. A compressão deve ser ainda maior no caso de vídeos para televisores de alta definição (HD, de high definition), nos quais as resoluções da imagem chegam a 1.920 ×
1.080 × 24 bits/imagem.

### 13.4. Compressão RLE(Run-Length Encoded)

Para comprimir a imagem vamos utilizar um dos métodos mais simples, que se baseia na ideia que, se selecionarmos um pixel da imagem ao acaso, existe uma grande probabilidade de que os seus vizinhos tenham a mesma cor. O compressor varre o *pixmap*, linha por linha, à procura de subsequências de pixeis da mesma cor e substitui essa sub-sequência por um par de valores **(número de pixeis, cor)**. Por exemplo, supondo que cada cor era representada apenas por um inteiro, a sequência:

12, 12, 12, 12, 12, 12, 12, 12, 12, 35, 76, 112, 67, 87, 87, 87, 5, 5, 5, 5, 5, 5, 1...

seria comprimida para:

**9**,12, 35, 76, 112, 67, **3**, 87, **6**, 5, 1

onde os números sublinhados indicam o número de repetições consecutivas.

Na implementação, é necessário distinguir os números que correspondem aos valores da cor e os que corresponde ao número de repetições. Uma das possibilidades é a seguinte:

Um grupo de *m* pixeis que são todos diferentes é precedido pelo valor - *m*.

Na sequência anterior seria codificada por:

**9**,12,-4, 35, 76, 112, 67, **3**, 87, **6**, 5,?, 1,...

onde o valor em ? só será determinado depois de ser conhecido o resto da sequência. Verifica que agora, podes identificar univocamente se um elemento é o valor de uma cor ou um contador...

Se no arquivo original aparecerem mais de 127 valores iguais consecutivos, divide-se em grupos de 127. Por exemplo, se 12 aparecesse 600 vezes seguidas, ficaria:

127, 12, 127, 12, 127, 12, 127, 12, 92, 12.

Mas como nas imagens a cores, cada cor é decomposta em 3 componentes (R,G,B), cada sequência das cores básicas devem ser codificadas separadamente. A sequência

(171, 85, 34),(172, 85, 35),(172, 85, 30),(173, 85, 33),...

deve ser separada em 3 sequências

(171, 172, 172, 173,...),(85, 85, 85, 85,...),(34, 35, 30, 33,...)

Cada sequência deve ser codificada separadamente.

Por outro lado, a compressão deve ser feita linha a linha, de acordo com o formato PPM.

**Implementação**

O programa deve ler um arquivo .PPM (cabeçalho e pixmap) e guardar a imagem numa variável multi-dimensional. Para comprimir deve-se ler a imagem linha a linha e para cada linha e cada cor básica (R,G,B) aplicar o algoritmo acima, escrevendo diretamente para a saída-padrão. Note que cada linha da ``imagem'', vai corresponder a 3 "linhas" e, pela ordem, correspondem à cor vermelha, verde e azul.

Mas antes de começar a comprimir, deve-se escrever (na saída-padrão) a largura *l* e a altura *h* da imagem original.

**Descompressão**

Dado um ficheiro comprimido no formato **RLE**, pretende-se obter a imagem original no formato PPM. Começa-se por ler do arquivo comprimido a largura *l* e a altura *h*, da imagem original e com essa informação pode-se escrever o cabeçalho do PPM (segundo as indicações acima).

Depois lê-se a restante informação, como signed char, supondo que o primeiro valor é um contador:

- se for positivo, o valor seguinte deve ser repetido tantas vezes quanto o valor do contador
- se for negativo, indica quantos valores seguintes vão ser de pixeis diferentes

Para saber quando a linha terminou deve-se ir somando o valor absoluto dos contadores. Quanto terminar a primeira linha para a primeira cor, deve-se considerar a segunda cor e depois a terceira. Assim termina a primeira linha da imagem. Seguir o processo para as restantes.

Melhoramentos

Este método de compressão funciona bem quando as áreas de cores iguais são horizontais. Mas, caso contrário, o arquivo final poderá ser maior. Como melhoria, poderíamos selecionar modo de varredura da imagem inicial: por linhas, por colunas ou pelas diagonais ((0,0), (0,1), (1,0), (1,1), (0,2), (0,3), (1,2), (2,1), (3,0)...), então, modificar o algoritmo de compressão.