Computação Multimédia

Resolução de segundos testes by Pedro Feiteira

Teste 13/14

1. a) ? **b)** ?

2. a) Os sistemas de recuperação de informação multimédia tipicamente usam metadata e não diretamente a informação pois é o método mais rápido para obter as informações importantes deste tipo de conteúdo. Se não fosse usada metadata para "caracterizar" todo o conteúdo multimédia de um sistema, cada vez que fosse feita uma query, todo este conteúdo teria de ser verificado para o sistema saber se correspondia à query, o que iria provocar um decréscimo bastante acentuado na performance de um sistema deste tipo. Assim, com a ajuda da metadata, cada vez que é necessário recuperar a informação multimédica alusiva a uma fotografia ou vídeo, é processada a sua metadata. Quando é feita uma query ao sistema, basta comparar esta com a metadata de todo o conteúdo disponível no sistema, melhorando bastante a sua performance. **b)** Objetos detetados numa imagem/vídeo e cor predominante, por exemplo. Estas podem usadas para filtrar toda a informação existente num sistema com bastante conteúdo de multimédia.

3. a)

4.

5.

Teste 14/15

1. a) Para diminuir o contraste da imagem deve-se de realizar ...

2.

```
    void applyErosion(OfImage* img, bool intensity){
        if(intensity)
            cv::Mat structure = (cv::Mat_<double>(3, 3) << 1, 1, 1, 1, 1,
1, 1, 1, 1);
        else
            cv::Mat structure = (cv::Mat_<double>(3, 3) << 1, 0, 1, 0, 0,
0, 1, 0, 1);
        unsigned char* pixels = img.getPixels().getData();
        for(int i = 0; i < img->getHeight(); i++){
            for(int j = 0; j < img->getWidth(); i++){
                //DO THE THING
            }
        }
    }
```

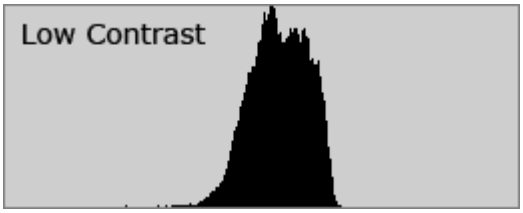
3. ??

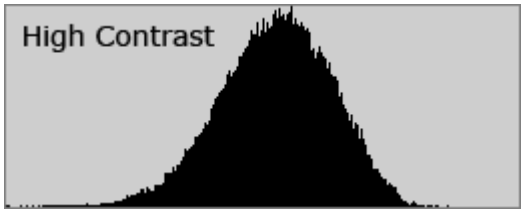
4. Na minha opinião deve de ser aplicado este filtro com diferentes orientações à mesma imagem. Depois deveria-se de sobrepor todas as resultantes, obtendo assim uma informação mais detalhada da textura de certo conteúdo. Como metada, guardaria-se esta imagem e também o indíce dos filtros que lhe deram origem. O espaço seria o tamanho da imagem + o espaço necessário para guardar os indices.

5. Para obter deve-se de aplicar um filtro de provoque blur na imagem em questão. Este blur não é nada mais que uma extensão/atenuação das edges da imagem. Depois bastaria subtrair a esta imagem resultante, a imagem original sobrando assim as diferenças entre elas, ou seja, os contornos. (Desenho)

Teste 15/16

1. a)





Através de uma transformação linear com m = ? e b = ? porque...

2. a) Que a altura da nota sobe, depois mantém-se dois tempos, volta a subir, desce e, por fim, volta a subir **b)** Não. Pois apesar de terem a mesma altura (O código de Parsons, denominado formalmente como código de Parsons para contornos melódicos, é uma notação simples usada para identificar um trecho musical por meio do movimento melódico — o movimento da altura das notas para cima e para baixo by Wikipédia) podem não representar a mesma música. **c)** No final, movimento da altura de B aumenta enquanto que em C mantém-se.

3.

```
void transform(OfImage* img, int threshHold){

    OfPixels pixels = img->getPixels();

    for(int i = 0; i < pixels.size(); i++){
        char p = pixels[i];
        if(p >= threshHold){
            pixels[i] = 255 - p;
        }
    }

}
```

4. a) A imagem passa baixo é a imagem mais à direita e a imagem passa alto é a do meio. Pois os filtros de low pass filtering (passa baixo) provocam a atenuação dos contornos dos elementos da imagem, deixamdo-a desfocada. Já os filtros passa alto provocam o efeito oposto (sharpener), isto é, salientam as frequências mais

altas, criando ruído na imagem. **b)** Para low pass filter pode-se obter através de um Gaussian Blur. Para high pass filters pode-se obter através do filtro Sobel, por exemplo. I(x,y) * Filter **c)**

```
OfImage* applyFilter(OfImage* img, cv::Mat filter, int filterSize){
    cv::mat image;
    std::vector<cv::mat> appliedFilter;
    OfPixels pixels = img.getPixels();
    cv::mat temp;
    for(int i = 0; i < pixels.size(); i++){
        image = getFromImage(pixels, filterSize);
        temp = image * filter;
        appliedFilter.push_back(temp);
    }
    return processVectorToImage(appliedFilter);
}
```

5. É realizado este procedimento para o algoritmo SIFT poder conhecer o keypoint de forma invariante, ou seja, mesmo que o keypoint representado na imagem que está a ser comparada apresente uma escala diferente, o algoritmo continua a reconhecê-lo. Por isso é que no nome do próprio algoritmo já consta esta condição (Scale-Invariant).

6. Duas imagens podem ser comparadas através da sua metadata. (Exemplos de técnicas concretas?)

Teste 17/18

1. a) Estes filtos pertencem à classe dos low pass filters, ou seja, estes atenuam os contornos das imagens. O filtro gaussiano é nada mais que a transformação de cada pixel de uma imagem realizada com a função gaussiana. Já o filtro de média é apenas uma matriz onde todos os seus elementes têm o valor 1/9. Esta última, por ser simples, faz com que o desfoque elimine alguns elementos da imagem, tornando-a menos nítida que o filtro gaussiano. **b)** À imagem original aplica-se um destes filtros para criar desfoque. Este desfoque não é mais do que o "prolongamento" dos contornos presentes na imagem. Quando subraí-mos esta imagem resultante pela imagem original, as suas diferenças são os contronos existentes na imagem.

2. | 0 1 1 | | -1 0 1 | | -1 -1 0 |

Porque iria criar a sombra do lado esquerdo como mostra a figura 2.

3. Sobreposição de várias imagem onde foram usados os filtros mais vantajosos para destacar as diferentes texturas destas imagens. Perderia-se alguma nitidez e formas da imagem original.

4. a) A imagem é dividida em segmentos e para cada segmento são aplicados os filtros de contornos com várias orientações. Se num dado segmento for verificado que existe um cortorno com determinada orientação este é inserido no histograma. **b)**

```
void createEdgeHistogram(OfImage* img){
    //dividir img aos bocados
    //para cada tile, aplicar os 4 filtros de edges
    //se o filtro dá match com o segmento então hist(numTile)
```

```
["orientação"]++
}
```

5. Contraste. Pois imagens/videos com o contraste desregulado tiraria qualidade estética a este conteúdo multimédia. Esta seria calculada com a seguinte fórmula: C = |Ip - In| / |In|. Assim, para cada tipo de imagem seria adaptado o seu contraste melhorando a sua estética. **(Enchimento de Tchouriço)**