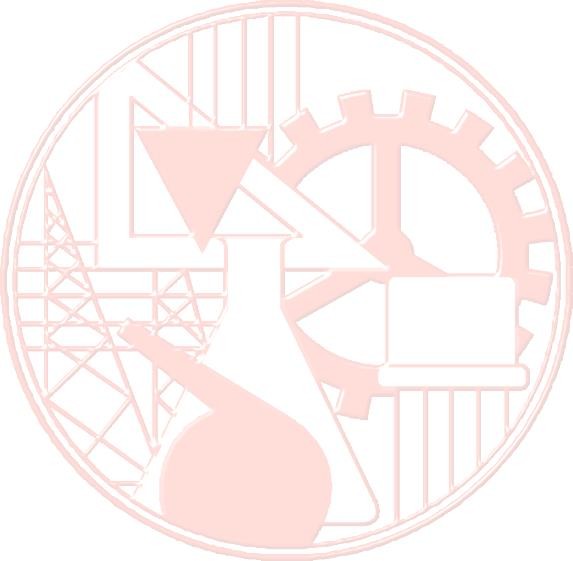
**I**nstituto **S**uperior de **E**ngenharia de

**L**isboa

ENGENHARIA INFORMÁTICA E DE

COMPUTADORES

***PROCESSAMENTO DE IMAGEM E BIOMETRIA***

***PRIMEIRO TRABALHO PRÁTICO***

*LI61N-MI2N*

**Autores – Grupo 3:**

40602, Sara Sobral

40686, Eduardo António

***Índice***

[Desenvolvimento de conclusões 3](#_Toc481793862)

[Exercício 1 3](#_Toc481793863)

[Função image\_details.m 3](#_Toc481793864)

[Função image\_details.m 4](#_Toc481793865)

[Exercício 2 5](#_Toc481793866)

[Função medical\_image\_enhancement.m 5](#_Toc481793867)

[Função fingerprint\_enhancement.m 6](#_Toc481793868)

[Função face\_detection.m 6](#_Toc481793869)

[Exercício 3 7](#_Toc481793870)

[Exercício 4 8](#_Toc481793871)

[Função codeCardGenerato 8](#_Toc481793872)

[Exercício 5 9](#_Toc481793873)

# Desenvolvimento de conclusões

## Exercício 1

### Função image\_details.m

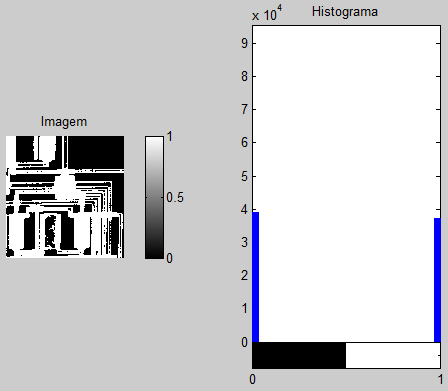
* **Obtenha informação sobre uma imagem. espacial; resolução em profundidade; valores mínimo, médio e máximo de intensidade; medida de contraste; entropia da imagem. Apresentar a imagem e o respetivo histograma.**

Utilização da função imfinfo para obter informação sobre a imagem, como a resolução espacial, a resolução em profundidade e o tipo de cor da imagem para saber se é necessário obter as componentes (R, G e B).

Utilização da função imread para ler a imagem a partir do ficheiro.

A função recebe como parâmetro a imagem que se pretende analisar. As imagens estão presentes em GenericImages.zip.

* **Apresentação de resultados**
* Exemplo de imagem binária: circ\_bw.tif



resolução espacial = 76160

resolução em profundidade = 1 bit/pixel

valores mínimo de intensidade = 0

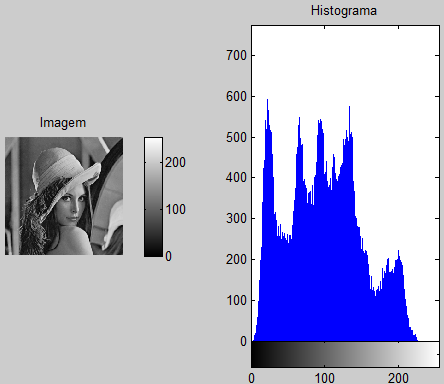
valores médio de intensidade = 0.48871

valores máximo de intensidade = 1

medida de contraste = 6.0206

entropia da imagem = 0.99963

* Exemplo de imagem monocromática: lena.gif



resolução espacial = 65535

resolução em profundidade =

8 bit/pixel

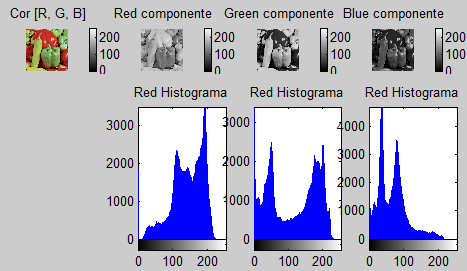
valores mínimo de intensidade = 3

valores médio de intensidade = 99

valores máximo de intensidade = 238

medida de contraste = 35.5268

entropia da imagem = 7.5683

* Exemplo de imagem a cores: peppers.png

resolução espacial = 262144

resolução em profundidade = 24

bit/pixel

|  |  |  |  |
| --- | --- | --- | --- |
| componente | red: | green: | blue: |
| valores mínimo de intensidade = | 0 | 0 | 0 |
| valores médio de intensidade = | 144.1151 | 112.8049 | 64.1288 |
| valores máximo de intensidade = | 253 | 255 | 255 |
| medida de contraste = | 48.0967 | 48.1308 | 48.1308 |
| entropia da imagem = | 7.3316 | 7.5605 | 7.0196 |

### Função image\_details.m

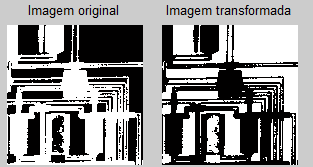
* **Obtenha informação sobre uma imagem e a uma versão transformada por T, sendo T uma transformação de intensidade genérica.**

A função recebe como parâmetro a imagem que se pretende analisar. As imagens estão presentes em GenericImages.zip.

Função T utilizada foi a inversa, ou seja, o índice 0 da *lookup table* corresponde à intensidade mais elevada que a imagem pode ter.

Utilização da função intlut para gerar a imagem transformada.

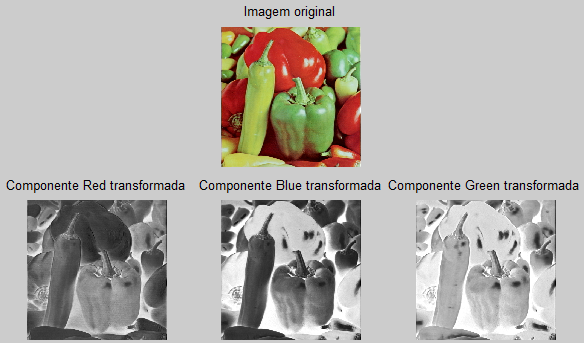
* **Apresentação de resultados**
* Exemplo de imagem binária: circ\_bw.tif



* Exemplo de imagem monocromática: lena.gif



* Exemplo de imagem a cores: peppers.png



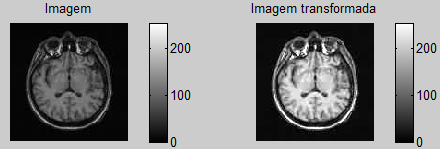
## Exercício 2

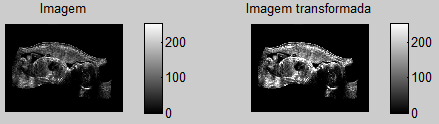
### Função medical\_image\_enhancement.m

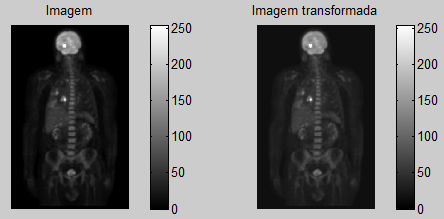
* **Realiza transformações de intensidade adequadas para melhorar a legibilidade das mesmas. Apresente os resultados obtidos para cada imagem deste conjunto.**

A função recebe como parâmetro a imagem que se pretende analisar. As imagens estão presentes em MedicalImages.zip. Se a imagem recebida for binária ou coloridas é convertida para níveis de cinzento para que se possa aplicar a função imadjust. Esta função ajusta os valores de intensidade da imagem, ou seja, mapeia os valores de intensidade da imagem para novos valores de forma a que 1% dos dados sejam saturados em baixas e altas intensidades. O que faz aumentar o contraste da imagem de saída..

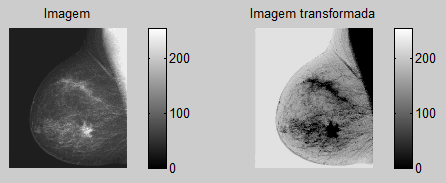
* **Apresentação de resultados**







No caso de PET1.tif obtemos os tumores depois o contorno do corpo, a apresentamos a sua soma.

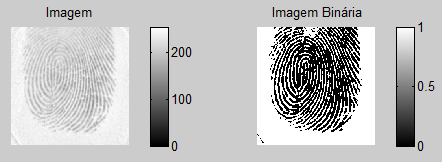
No caso XRay1.tif obtemos a inversa à qual subtraímos a imagem original.

### Função fingerprint\_enhancement.m

* **Para uma imagem de impressão digital produz uma versão binária da mesma, tentando separar as riscas do fundo.**

A função recebe como parâmetro a imagem que se pretende analisar. As imagens estão presentes em FingerprintImages.zip. Nesta função calcula-se o limiar ótimo para transformar a imagem na sua versão binária (método de Otsu) através da função im2bw..

* **Apresentação de resultados**



### Função face\_detection.m

* **Para uma imagem de face, procura localizar os extremos da face e afixar um retângulo a delimitar a face.**

A função recebe como parâmetro a imagem que se pretende analisar. As imagens estão presentes em FaceImages.zip. Foi utilizado um detetor de objetos (<https://www.mathworks.com/help/vision/ref/vision.cascadeobjectdetector-class.html>) que utiliza o algoritmo Viola-Jones. É criado um detetor de objetos com classificação modelo ‘FrontalFaceCART’, utliza-se a função step que processa os dados de entrada de acordo com o algoritmo do objeto e retorna uma matriz [x y largura altura], que especifica em pixels, o canto superior esquerdo e o tamanho de uma caixa delimitadora.

* **Apresentação de resultados**



## Exercício 3

* **Identifique o(s) problema(s) na imagem e proponha uma técnica (ou mais) para a sua correção;**
* circles.bmp

**Problema(s)**: As imagens apresentam ruido padronizado com diferentes níveis de intensidade.

|  |  |
| --- | --- |
| Imagens | Proposta(s) de correção |
| circles\_1.bmp | Aplicar uma transformação em frequência passa baixo para remover o ruido que não pertence ao espetro da imagem e aplicar um filtro de *sharpening* para salientar as transições entre os círculos da aplicação anterior. |
| circles\_2.bmp |
| circles\_3.bmp |
| circles\_4.bmp |
| circles\_5.bmp |

* face1.bmp

**Problema(s)**: As imagens apresentam diferentes intensidades de ruido impulsivo salt and peppe.

|  |  |
| --- | --- |
| Imagens | Proposta(s) de correção |
| face1\_1.bmp | Aplicar um filtro espacial não linear mediana para eliminar pontos de ruido impulsivo com valores extremos. A dimensão da mascara mediana pode variar para corrigir as diferentes imagens. Obriga à ordenação dos valores, colocando os valores 0 (pepper) à esquerda, os valores 255 (salt) à direita e os valores dos pixéis originais no centro, obtendo assim um valor do contexto da imagem original. |
| face1\_2.bmp |
| face1\_3.bmp |
| face1\_4.bmp |
| face1\_5.bmp |

* finger1.bmp

**Problema(s)**: As imagens estão esborratadas, o nível de *blured* vai aumentando de imagem para imagem.

|  |  |
| --- | --- |
| Imagens | Proposta(s) de correção |
| finger1\_1.bmp | Aplica rum filtro espacial linear passa alto (como o laplaciano) para realizar *sharpening* sobre a imagem, ou seja, afiar e salientar as transições das zonas brancas para as pretas. |
| finger1\_2.bmp |
| finger1\_3.bmp |
| finger1\_4.bmp |
| finger1\_5.bmp |

* lena.gif

**Problema(s)**: Todas as imagens apresentam ruido.

|  |  |
| --- | --- |
| Imagens | Proposta(s) de correção |
| lena\_1.bmp | Aplicar um filtro espacial linear passa baixo para suavizar a imagem de forma a eliminar o ruido existente. |
| lena\_2. bmp |
| lena\_3. bmp |
| lena\_4.bBmp |
| lena\_5. bmp |

* squares.gif

**Problema(s)**: As imagens estão esborratadas, o nível de *blured* vai aumentando de imagem para imagem.

|  |  |
| --- | --- |
| Imagens | Proposta(s) de correção |
| squares\_1.gif | Aplicar um filtro espacial passa alto de deteção de contornos. |
| squares\_2.gif |
| squares\_3.gif |
| squares\_4.gif |

* **Compare a imagem restaurada com a imagem original.**

Um MAE baixo e um MSE alto indica ocorrência de outliers no conjunto de dados.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Imagens | Brilho | Contraste | Entropia | MSE | MAE |
| circles\_1.bmp | 4.4514 | 0 | -2.8446 | 0.65889 | 4.444 |
| circles\_2.bmp | 2.9665 | 0 | -2.5703 | 0.70064 | 2.9599 |
| circles\_3.bmp | 0.61072 | 0 | -1.4036 | 0.98042 | 0.60997 |
| circles\_4.bmp | 0.61281 | 0 | -1.4109 | 0.97893 | 0.61208 |
| circles\_5.bmp | 0.62746 | 0 | -1.4329 | 0.97481 | 0.62737 |
|  |  |  |  |  |  |
| face1\_1.bmp | 0.077698 | 0 | 0.066425 | 0.46234 | 1.6896 |
| face1\_2.bmp | 0.078674 | 0 | 0.060538 | 0.45258 | 1.7401 |
| face1\_3.bmp | 0.11615 | 0 | 0.06493 | 0.44232 | 1.8273 |
| face1\_4.bmp | 0.10431 | 0 | 0.06076 | 0.41498 | 1.8867 |
| face1\_5.bmp | 0.11865 | 0 | 0.062689 | 0.41351 | 2.1005 |
|  |  |  |  |  |  |
| finger1\_1.bmp | -3.3247 | 0 | 0.14186 | 0.4341 | 5.8699 |
| finger1\_2.bmp | -2.5914 | 0 | 0.17585 | 0.14763 | 8.3205 |
| finger1\_3.bmp | -1.8762 | 0 | 0.22112 | 0.069748 | 11.5573 |
| finger1\_4.bmp | -1.1661 | 0 | 0.252 | 0.04567 | 14.8908 |
| finger1\_5.bmp | -0.46469 | 0 | 0.25703 | 0.032974 | 0.032974 |
|  |  |  |  |  |  |
| lena\_1.gif | 0.31051 | 0 | 0.029094 | 1 | 14 |
| lena\_2.gif | 0.31051 | 0 | 0.029094 | 1 | 14 |
| lena\_3.gif | 0.31051 | 0 | 0.029094 | 1 | 14 |
| lena\_4.gif | 0.31051 | 0 | 0.029094 | 1 | 14 |
| lena\_5.gif | 0.31051 | 0 | 0.029094 | 1 | 14 |
|  |  |  |  |  |  |
| squares\_1.gif | 9.7495 | 0 | -4.5947 | 0 | 11 |
| squares\_2.gif | -10.1055 | 0 | -4.0995 | 0 | 11 |
| squares\_3.gif | -10.2864 | 0 | -3.7359 | 0 | 11 |
| squares\_4.gif | -10.4003 | 0 | -3.4807 | 0 | 11 |
| squares\_5.gif | -10.483 | 0 | -3.2985 | 0 | 11 |

## Exercício 4

### Função codeCardGenerato

* **A qual gera uma imagem colorida, contendo um cartão de códigos, com conteúdo aleatório, de forma matricial, tal como se apresenta na figura. Apresente cinco imagens diferentes geradas com o método proposto.**
* **Descrição da metodologia**

Este exercício tem como objetivo gerar uma imagem colorida com conteúdo aleatório na forma matricial.

1. Gerar um valor (inteiro) aleatório para a largura N e um valor (caracter) para altura M.
2. Adicionar a um contento os valores de 1 a N com espaçamento de 3.
3. Criar um objeto imagem (*Graphics2D*)*.* Ter cuidado com o tipo de fonte para que o espaçamento não fique desorganizado.
4. Gerar um valor aleatório para a escolha da cor, a cor é introduzida por linha.
5. Escrever no objeto imagem o conteúdo do contentor com a cor gerada.
6. Gerar M linhas com N valores (inteiros) aleatórios.
7. Cada linha gerada é adicionada ao contentor, é gerada também uma cor aleatória. Repetir o ponto 5.
8. Preencher o fundo da imagem com uma cor.

* **Apresentação de resultados**



## Exercício 5

As imagens utilizadas estão presentes em BinaryAndGrayscaleImages.zip.

* **Realize coloração das imagens através das técnicas de intensity slicing e intensity to RGB transform; indique os critérios e as funções usadas para a atribuição de cores.**

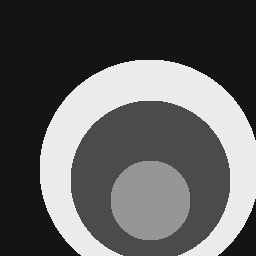
As técnicas recebem imagens monocromáticas e realização a sua coloração. É útil para visualizar imagens médicas / científicas / vegetação, pois é de interesse realçar certos valores de intensidade para serem mais percetíveis ao sistema visual humano.

A técnica de intensity slicing começa por dividir a resolução em profundidade da imagem recebida por um valor (escala) obtendo o número de intervalos. O valor escala é calculado consoante os valores de níveis de cinzento que a imagem original usa. Logo, o número de intervalos é igual ao número de cores da imagem original que por sua vez é igual ao número de cor que a imagem final vai ter.

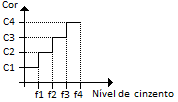
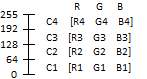
Para cada intervalo é construído um valor RGB. No final a imagem monocromática de dimensão passa para .

Exemplo:

Imagem monocromática (circles.bmp) com resolução em profundidade n = 8 bit/pixel



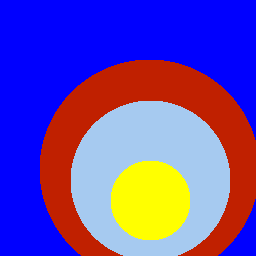
A escala é 4 porque a imagem apenas usa quatro níveis cinzentos.



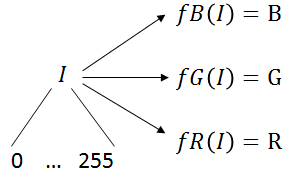


Quando o valor do pixel está mais perto de 0 a cor RGB atribuída tem um tom frio, e quando o valor do pixel está mais perto de 255 a cor RGB atribuída tem um tom quente. Foi utilizada uma tabela de lookup.

Resultado:



A técnica intensity to RGB transform aplica três funções diferentes sobre a imagem monocromática recebida. Cada função gera a componente azul, a componente verde e a componente vermelha.



* **Comente qual das técnicas aplicadas produz melhores resultados.**

A técnica de intensity slicing apresenta melhor resultados, pois se a função escolhida na técnica intensity to RGB transform não for adequada pode fazer o oposto de salientar detalhes.

Exemplo:

intensity slicing: intensity to RGB transform

