INSTITUTO FEDERAL
Mato Grosso
Campus Cuiabá
Cel. Octayde Jorge da Silva

Departamento da Área de Informática
Curso: Bacharelado em Engenharia da Computação **Semestre:** 9
Curso: Bacharelado em Engenharia de Controle e Automação **Semestre:** Optativa
Disciplina: Processamento Digital de Imagens.
Professor: Esp. Giuliano Robledo Zucoloto Moreira.

ESTUDO DIRIGIDO 01
Introdução ao GNU Octave aplicado à disciplina de PDI

Grande parte deste material está baseada na documentação oficial das das versões 4.2.1 e 6.1.0 do *software* que, na período desta redação, foi encontrada no endereço eletrônico: `https://octave.org/doc`**.**

Todos os algoritmos devem conter instruções para:

- fechar todas as janelas que foram abertas por script/comando em execução/ões anteriores;
- limpar o prompt de comando;
- limpar todas variáveis.

1. **COMO ABRIR UMA IMAGEM?** Quando se abre uma imagem é necessário ter em mente que o *GNU Octave* não abre a imagem original, mas sim uma cópia dos dados da imagem para uma variável especificada pelo programador. A função utilizada para realizar tal operação é a função ***imread***. A quantidade de parâmetros de retorno desta função depende do tipo de imagem a ser "aberta". Este exemplo trata superficialmente da "abertura" à exibição de imagens.

```
%Fechar todas as janelas da execução anterior.
close all;
%Limpar o prompt de comando.
clc;
%Limpar todas as variáveis.
clear all;

% > > > > > Utilização da função imread < < < < <
%Se for utilizar imagem fora do diretório do script o caminho
%(path) da imagem é necessário no parâmetro da função.

%"Abrir" uma imagem e armazenar na variável Ibin.
%Neste caso a imagem FORNECIDA é BINÁRIA.
Ibin = imread('nota_01_figura_01_bin.png');

%"Abrir" uma imagem e armazenar na variável Imono.
```

```
%Neste caso a imagem FORNECIDA é MONOCROMÁTICA.
Imono = imread('nota_01_figura_01_cinza.png');

%"Abrir" uma imagem e armazenar na variável Irgb.
%Neste caso a imagem FORNECIDA é COLORIDA (RGB).
Irgb = imread('nota_01_figura_01_rgb.png');

Ibin    %Sem ponto e vírgula imprime o conteúdo de Ibin.
Imono   %Sem ponto e vírgula imprime o conteúdo de Imono.
Irgb    %Sem ponto e vírgula imprime o conteúdo de Irgb.

size(Ibin)  %Imprime as dimensões da matriz Ibin.
size(Imono) %Imprime as dimensões da matriz Imono.
size(Irgb)  %Imprime as dimensões da matriz Irgb.

%O comando subplot divide a área da janela em uma matriz M x N
%(dois primeiros parâmetros) e permite inserir uma imagem ou
%gráfico dentro de uma posição no intervalo de 1 a MN da
%divisão da tela (último parâmetro do argumento da função).
subplot(2,2,1);
%O comando imshow mostra uma imagem da matriz informada
%no parâmetro. Os colchetes no parâmetro 2 são utilizados
para configuração automática do intervalo de exibição.
imshow(Ibin,[]);
subplot(2,2,2);
imshow(Imono,[]);
subplot(2,2,3);
imshow(Irgb,[]);
```

2. **COMO SALVAR UMA IMAGEM?** Relembrando: quando se abre uma imagem é necessário ter em mente que o *GNU Octave* não abre a imagem original, mas sim uma cópia dos dados da imagem para uma variável especificada pelo programador. O conteúdo de tal variável, modificado ou não, pode ser salvo em arquivo, até mesmo sobrescerver o arquivo da imagem original. A função utilizada para realizar a gravação do conteúdo da variável em um arquivo de imagem é a função ***imwrite***. Esta função tem basicamente dois parâmetros: o primeiro é a variável que contém a imagem e o segundo, o nome do arquivo de imagem acompanhado da extensão. O exemplo a seguir mostra uma linha de comando onde a função ***imwrite*** é utilizada para salavar o conteúdo da variável I no arquivo foto.png:

```
imwrite(I, "foto.png");
```

   O polimorfismo permite a adição de parâmetros ao argumento da função. Talvez estes sejam apresentados neste documento em outro momento.

3. **COMO PRODUZIR UMA IMAGEM ALEATÓRIA, SINTÉTICA E BINÁRIA?** Considerando as características apresentadas e que as dimensões da imagem sejam M e N podemos gerar tal imagem (**Ibin**) através do código:

```
M = 16;
N = 9;
```

```
Ibin = randi([0 1], M, N);
```

A função ***randi*** gera valores aleatórios inteiros. O primeiro parâmetro tem a função de limitar o maior número permitido para ser sorteado aleatoriamente ou de estabelecer um intervalo, entre colchetes, como foi feito no código de exemplo "**[0 1]**". Se não especificado o limite inferior num intervalo, o menor número a ser sorteado aleatoriamente é o número "**1**". Os demais parâmetros apresentados são respectivamente as dimensões de uma matriz onde M representa o número de linhas desta matriz e N o número de colunas.

4. **COMO PRODUZIR UMA IMAGEM ALEATÓRIA, SINTÉTICA E MONOCROMÁTICA?** Considerando as características apresentadas, que as dimensões da imagem sejam M e N e que n seja o número de bits que estabelece a faixa dinâmica (L) da imagem, podemos gerar tal imagem (**Imono**) através do código:

```
n = 8;
L = [0 (2^n)-1];
M = 16;
N = 9;
Imono = randi(L, M, N);
```

Como dito, a função ***randi*** gera valores aleatórios inteiros e o primeiro parâmetro tem a função de limitar o maior número permitido para ser sorteado aleatoriamente ou de estabelecer um intervalo para realização do sorteio. O intervalo da faixa dinâmica L foi criado de 0 a $2^n - 1$ atendendo a especificidade da teoria e também ajustando o limite inferior de sorteio da função ***randi*** de 1 para 0. Os demais parâmetros apresentados são respectivamente as dimensões de uma matriz onde M representa o número de linhas desta matriz e N o número de colunas.

5. **COMO PRODUZIR UMA IMAGEM ALEATÓRIA, SINTÉTICA E COLORIDA (RGB)?** Considerando as características apresentadas, que as dimensões da imagem sejam M e N e que n seja o número de bits que estabelece a faixa dinâmica (L) da imagem, podemos gerar tal imagem (**Irgb**) através do código:

```
n = 8;
L = [0 (2^n)-1];
M = 16;
N = 9;
C = 3;
Irgb = randi(L, M, N, C);
```

Considerando que para este código são respeitadas as peculiaridades já apresentadas em códigos anteriores fica codificado o intervalo da faixa dinâmica L de 0 a $2^n - 1$ e as dimensões da matriz, onde M e N representam respectivamente o número de linhas de colunas da matriz. A novidade é o número de canais **C** da imagem codificado igual a **3**, o que irá gerar a terceira dimensão da matriz da imagem **Irgb** com três camadas, sendo uma camada para cada canal de cor, respectivamente ***RGB***.

6. **COMO PRODUZIR UMA IMAGEM ALEATÓRIA, SINTÉTICA E COLORIDA (INDEXADA)?** Considerando as características apresentadas e que as dimensões da imagem sejam M e N, podemos gerar tal imagem (**Iind**) através do código:

```
M = 16;
N = 9;
C = 3;
Iind = rand(M, N, C, "single");
```

A função ***rand*** gera números aleatórios de precisão simples (*single*) ou dupla (*double*) num intervalo de zero e um, inclusive. As dimensões da matriz são M e N que representam respectivamente seu número de linhas de colunas e **C** é o número de canais da imagem codificado igual a **3**. O código irá gerar uma matriz tridimensional com dimensões $MxNxC$ preenchida com intensidades indexadas.

7. **COMO DISPONIBILIZAR UMA CAIXA DE DIÁLOGO PARA ABRIR UMA IMAGEM?** A caixa de diálogo aberta através da função *uigetfile* é utilizada para abrir um arquivo. A função aciona uma caixa de diálogo que permite selecionar ou até mesmo informar um nome de arquivo e retorna respectivamente três parâmetros: o nome do arquivo com a extensão, o caminho (*path)* do arquivo e por último o índice selecionado no tipo de arquivo na caixa de diálogo. Para receber o retorno da função necessitamos de três variáveis, uma para cada parâmetro. Segue um exemplo, cuja ordem das variáveis que recebem o retorno deve ser respeitada:

```
[arq, cam, ind] = uigetfile();
```

Neste caso a variável ***arq*** receberá o nome do arquivo selecionado acompanhado da sua extensão, a variável ***cam*** receberá o caminho (*path)* do arquivo selecionado e a variável ***ind*** receberá o índice do tipo de arquivo indicado no campo "*Tipo de arquivo*" da caixa de diálogo.

O polimorfismo da função *uigetfile* possibilita que o programador faça alterações nos aspectos visuais e funcionais da caixa de diálogo. No exemplo a seguir as variáveis tem exatamente as mesmas funções do exemplo anterior, a variação ocorrerá no parâmetro da função:

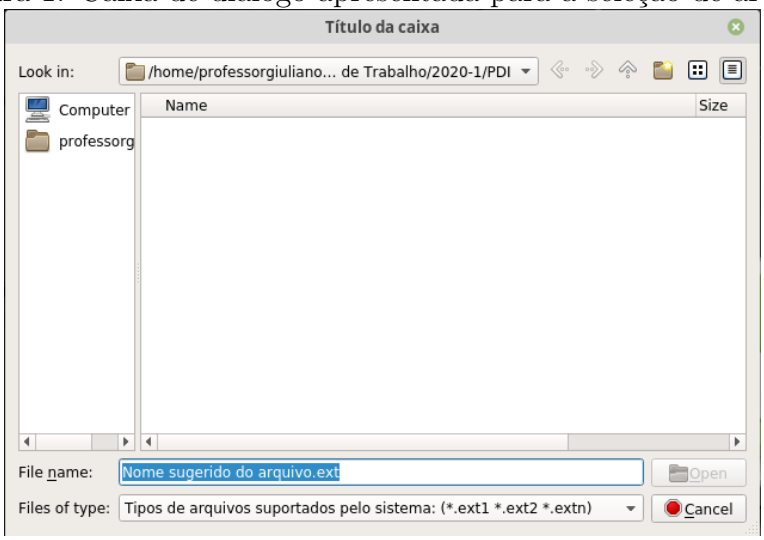
```
[arq, cam, ind] = uigetfile({"*.ext1;*.ext2;*.extn",
"Tipos de arquivos suportados pelo sistema:"}, "Título da caixa",
"Nome sugerido do arquivo.ext","MultiSelect","off");
```

O resultado da linha de comando supracitada é apresentado na Figura 1:

Analisando a função, o primeiro parâmetro, entre as chaves, está dividido em duas partes, cada parte entre chaves duplas e separadas por uma "," (vírgula); o conteúdo entre as primeiras aspas duplas permite informar os tipos de extensão cujos arquivos serão visíveis na caixa de diálogo, tais tipos precedidos de "*." (asterísco e ponto) e separados por ";" (ponto e vírgula); o conteúdo entre as segundas aspas duplas, após a vírgula, permite informar uma mensagem que será exibido junto às extensões listadas. O conteúdo entre as **chaves** será exibido na caixa de combinação (*combobox*) do campo **Tipo de arquivo** da caixa de diálogo.

O segundo parâmetro (após o fechamento da chave) é o "Título da caixa", onde o programador pode alterar o título da caixa de diálogo, o terceiro parâmetro, "Nome sugerido do arquivo.ext", é onde o programador pode alterar o nome de arquivo que é sugerido ao usuário no campo Arquivo da caixa de diálogo; O quarto parâmetro pode variar, neste caso foi adicionada a opção "MultiSelect", e esta desligada (***"off"***);

Figura 1: Caixa de diálogo apresentada para a seleção de arquivo



tal opção permite ou não ao usuário selecionar mais de um arquivo e na configuração apresentada o usuário não poderá selecionar mais de um arquivo, pois a multi-seleção está desligada. Para permitir ao usuário selecionar pamis de um arquivo na caixa de diálogo bastaria modificar a configuração **"*off*"** para **"*on*"**. Mais parâmetros podem ser adicionados à função sempre seguindo a lógica de informar entre aspas duplas o nome do parâmetro, seguido de vírgula, seguido da configuração desejada. É necessário analisar quais os parâmetros possíveis as opções parâmetro disponíveis para esta caixa de diálogo, sugere-se a leitura no manual do software para aperfeiçoamento.

A respeito da abertura de arquivos, em termos de prática, há poucas mudanças, pois a função coleta apenas informações do arquivo selecionado e as escreve nas variáveis especificadas. A abertura do arquivo se dá pela função imread, já apresentada. A mudança ocorre na forma como é informado o nome do arquivo para a importação, conforme ilustra o exemplo:

```
[arq, cam, ind] = uigetfile({"*.ext1;*.ext2;*.extn",
"Tipos de arquivos suportados pelo sistema:"}, "Título da caixa",
"Nome sugerido do arquivo.ext","MultiSelect","off");
I = imread(strcat(cam, arq));
```

A função *strcat* efetua a junção do caminho do arquivo e do nome do arquivo acompanhado de sua extensão, formando o caminho completo do arquivo. Este valor é passado à função *imread* que efetua a importação, neste caso, para a variável ***I***.

8. **COMO DISPONIBILIZAR UMA CAIXA DE DIÁLOGO PARA SALVAR UMA IMAGEM?** O processo é semelhante à disponibilização de uma caixa para abrir uma imagem, a diferença está basicamente no nome da função que é utilizada para chamar a caixa de diálogo e ao invés de se fazer uso da função de leitura, faz-se uso da função de escrita de imagem. A caixa de diálogo aberta através da função *uiputfile* é utilizada para salvar um arquivo. A função aciona uma caixa de diálogo que permite informar um nome de arquivo e retorna respectivamente três parâmetros: o nome do arquivo com a extensão, o caminho (*path)* do arquivo e por último o índice selecionado no tipo de arquivo na caixa de diálogo. Para receber o retorno da função necessitamos de três variáveis, uma para cada parâmetro. Segue um exemplo, cuja ordem das variáveis que recebem o retorno deve ser respeitada:

```
[arq, cam, ind] = uiputfile();
```

Neste caso a variável ***arq*** receberá o nome do arquivo selecionado acompanhado da sua extensão, a variável ***cam*** receberá o caminho (*path)* do arquivo selecionado e a variável ***ind*** receberá o índice do tipo de arquivo indicado no campo "*Tipo de arquivo*" da caixa de diálogo.

O polimorfismo da função *uiputfile* possibilita que o programador faça alterações nos aspectos visuais e funcionais da caixa de diálogo. No exemplo a seguir as variáveis tem exatamente as mesmas funções dos exemplos anteriores, a variação ocorrerá no parâmetro da função:

```
[arq, cam, ind] = uiputfile({"*.ext1;*.ext2;*.extn",
"Tipos de arquivos suportados pelo sistema:"}, "Título da caixa",
"Nome sugerido do arquivo.ext");
```

Como os parâmetros são os mesmos do exemplo anterior, o resultado será o mesmo que é apresentado na Figura 1.

A respeito da gravação de arquivos, em termos de prática, há poucas mudanças, pois a função coleta apenas informações do arquivo selecionado e as escreve nas variáveis especificadas. A gravação do arquivo se dá pela função imwrite, já apresentada. A mudança ocorre na forma como é informado o nome do arquivo para a exportação, conforme ilustra o exemplo:

```
[arq, cam, ind] = uiputfile({"*.ext1;*.ext2;*.extn",
"Tipos de arquivos suportados pelo sistema:"}, "Título da caixa",
"Nome sugerido do arquivo.ext");
imwrite(I,strcat(cam, arq));
```

A função *strcat* efetua a junção do caminho do arquivo e do nome do arquivo acompanhado de sua extensão, formando o caminho completo do arquivo. Este valor é passado à função *imwrite* que efetua a exportação da variável ***I*** para o arquivo especificado.