22437 - Industrial Vision
Lab 1: Introduction to Matlab and the Image Processing Toolbox

Jaume Taberner Ferrer

Universitat de les Illes Balears

Matrices and Matlab

Before beginning this lab, READ the first two tutorials of the Get Started with MATLAB[1], available online. For obtaining information about the functions available in Matlab to work with matrices, use the following command:

```
>> help elmat
```

Now perform the tasks written below. These tasks should be solved, in most cases, using only one Matlab command:

1. Create a 5×5 matrix of ones (A).
2. Create a 5×3 matrix of ones (B).
3. Create a 3×3 matrix of zeros (C).
4. Create a 4×4 matrix with equal row, column, and diagonal sums (D).
5. Create a 4×4 random matrix whose values are uniformly distributed between 0 and 1 (E).
6. Create a 5×5 identity matrix (F).
7. Sum the matrices A and F.
8. Subtract the matrices A and F.
9. Sum the matrices A and C. Is it possible?.
10. Compute D^{20}.
11. Compute $F \times 2$.
12. Compute $A \times F$.
13. Compute $A \times F$, element by element.
14. Compute the transpose matrix of E.
15. Compute the inverse matrix of E.
16. Compute the determinant of matrix C.
17. Store the size of matrix B in two variables, $rows$ and $cols$.
18. Given the matrix F, obtain the indices of the elements whose value is not zero.
19. Set the detected values in the previous point to -1.
20. Given a 10×10 matrix, set the values of the even columns to zero.

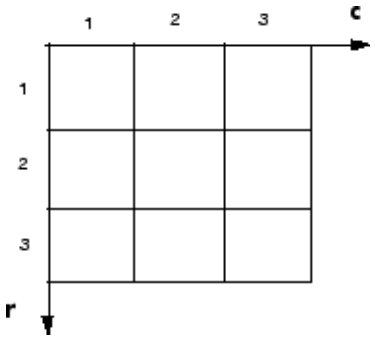
[1] https://es.mathworks.com/help/matlab/getting-started-with-matlab.html?lang=en

21. Create a 10×10 matrix where the values of each row coincide with the row number.
22. Given a 10×10 matrix, set the values of the fourth row to zero.
23. Given a 10×10 matrix, set the values of the second column to zero.
24. Given a 10×10 matrix, set the values of the fifth column to the values of the first column.
25. Given a 10×10 matrix, set all the values to zero, except the rows and columns in the edges of the matrix.

Introduction to Image Processing Toolbox

Useful functions: *imread*, *imwrite*, *im2double*, *mat2gray*, *rgb2gray*, *mesh*, *surf*, *gray2ind*.

A gray-scale image can be defined as a two-dimensional function $f(x, y)$, where x and y are called *spatial coordinates*, and the value of f at each pair (x, y) is called the *intensity* of the image at this point. In Matlab, a digital image can be represented as a matrix, where each element of this array is called *picture element*, or *pixel*. The coordinate conventions are defined as:



where r and c are the row and column numbers, respectively. Note that the indices start from 1 and not 0 like other programming languages. For a reference on image handling in Matlab, use the following command:

```
>> help images
```

Perform the following tasks using Matlab:

1. Open and display the image *landscape.jpg*. Determine the dimensions of the image. Is this a color image?
2. Convert the image to gray-scale, and display the result. Then, save this resulting image into another file called *landscapegray.jpg*. What is the data type of the pixels?
3. Rescale the pixel values to the range [0, 1] and convert the image to *double* precision.
4. Display the gray scale image as a three-dimensional plot. Compare the results using two functions: *mesh* and *surf*.
5. Convert the gray scale image to an indexed image with a colormap of 16 components and display the result. Do you observe differences between the original and the indexed images?
6. Write a Matlab script for generating the negative of the image *moon.bmp* without using any Matlab function. The final results should look like this:

7. Write a Matlab script which flips the image *moon.bmp* vertically without using any Matlab function. The final result should look like this: