



Third Lab Assignment - Lab 3: Image Processing in Matlab

Objectives: This assignment intends to apply and consolidate the knowledge gained concerning image processing techniques to process and improve images.

Introduction

The objective of this laboratory assignment is to allow the student to experiment further digital processing techniques of visual signals, using Matlab.


To conduct the proposed experiments, Matlab scripts and images available on Moodle will be used. Images will be used as input data to those scripts. The output will consist of processed versions of those images, which should be analysed by the student to interpret the effects of the conducted processing operations. The majority of those images has the bitmap format (.bmp) which means that each pixel is represented by three RGB eight bit values, so in total 24 bits per pixel.

Note: the symbol  means that you should include in your report graphics or images resulting from the operated processing or code that you may have developed. The symbol  indicates that you should include in your report a brief analysis of the results you have obtained.

Work to be developed

1. *Blue background extraction*

On TV weather news, what the viewer normally sees is the weather forecast person standing in front of maps. In traditional news production, they just stand in front of a blue wall. Put in simple manner, a special system extracts the person (the “weatherman”) from the images and adds her/him in front of the real weather related images. The basic idea of this system is to split an image into RGB channels, create a mask based on the information of the blue channel, and use this mask to extract images. In this assignment, you will develop a script that is able to do just this.

1.1. Basic segmentation: based on the script “segmentBB.m” and the set of Matlab built-in functions already learnt, write a um script Matlab  that:

- imports a coloured image with a blue background and presents that image on the screen;
- separates each RGB component in a different matrix and visualises each one on the screen;
- uses the matrix with the B component to identify the foreground objects (the jumping man or the Christmas bulbs or the bird). One possibility for doing this is to inspect the values of the B matrix: high values will correspond to the background. If we set up a threshold with a value just below those higher values, all the pixels that have a value

lower than the threshold in principle will belong to the foreground (they will represent the foreground objects, i.e., the jumping man, the christmas bulbs or the bird). Using that threshold, copy from the B matrix to a new matrix (with the same dimensions) only the pixel values that are below that threshold. When doing that you may put those pixels with the value 255 and all the others with value 0 (you will create a black and white picture). To decide on the threshold, instead of looking directly at the values of the B matrix, you may generate the histogram with the built-in function `imhist` and by inspecting visually the histogram decide on a suitable threshold value (the script may ask the user to input that value).

- shows the black and white segmented image on the screen and creates in the disk a new file with that image.
- Make experiments with different threshold values and with different images. Interpret the results and comment the results taking into consideration that you have used only the blue channel. 🤖 Could there be low blue value in zones of the background? Could there be high blue values in some parts of the foreground objects? Is it always true that a pixel with a high value in the B component is always blue?

1.2. Alternative segmentation: based on the script “segmentBB.m” and the set of Matlab built-in functions already learnt, write a um script Matlab 📄 that:

- imports a coloured image with a blue background and presents that image on the screen;
- separates each RGB component in a different matrix and visualises each one on the screen;
- Considering that a pixel is really blue if it has high values in the B component and low values in the other components, computes the “blueness” of a pixel using the equation $\text{blueness} = B - \max(R, G)$.
- selects a threshold based on the blueness factor (adopt the procedure explained above) and creates a new black and white image with the pixels of the foreground objects with value 255 and all the other with value zero (adopt the procedure explained above).
- shows the black and white segmented image on the screen and creates in the disk a new file with that image.

2. Adding objects to another image

In this part you will use the segmented images that have been generated by your previous scripts to create a new image with the superimposition of the segmented image with a new image.

- Use the Matlab script “imfuse” to achieve that. Use the command “help imfuse” at the Matlab command prompt to learn how the script works.
- modify your previous script(s) to generate a coloured segmented image and not only a black and white version.

Report should be delivered no later than March 28 in Moodle.