Vectorizing

Matrices are matlab's forte, using them wisely and learning to vectorize to speed up your code is essential when handling for example big data or large images.

1. This function adds a black frame to an image.

```
function [im_out] = add_frame(im_in)
% adds a frame to an existing grayscale image
% im_in: original grayscale image
% im_out: black frame of 5 wide is added to im_in
[r,c]=size(im_in);
im_out=uint8(255*ones(r+10,c+10));
im_out(6:r+5,6:c+5)=im_in;
end
```

Have a look at https://nl.mathworks.com/help/matlab/matlab_prog/measure-performance-of-your-program.html

Adapt this function to a second version which uses two for-loops to add the frame to the image.

Make two scripts to time each functions for the image 'baboon.jpg'. Do you notice a difference. What do you notice when you run the scripts several times. What do you suggest to improve this test?

What do you need to change in the original function to have a white frame?

2. Choose two articles and apply the represented techniques on an image. A bonus if you find an extra article which proves to be interesting.

https://nl.mathworks.com/company/newsletters/articles/new-functions-for-vectorizing-operations-on-any-data-type.html

https://nl.mathworks.com/company/newsletters/articles/matrix-indexing-in-matlab.html

https://blogs.mathworks.com/loren/2019/09/25/which-way-to-compute-cellfun-or-for-loop/

https://blogs.mathworks.com/loren/2019/07/09/internet-of-things-how-to-get-started/

https://blogs.mathworks.com/loren/2019/05/29/big-data-in-mat-files/

https://nl.mathworks.com/company/newsletters/articles/an-adventure-of-sortsbehind-the-scenes-of-a-matlab-upgrade.html

https://nl.mathworks.com/company/newsletters/articles/two-new-functions-for-converting-datatypes-and-changing-byte-order.html

https://nl.mathworks.com/company/newsletters/articles/how-matlab-represents-pixel-colors.html