Homework 1
Image Manipulation and Periodic Sequences

- Deadline: Wednesday, 2019-11-6, 23:45
- Submission: Upload your code on Cody Coursework™ Mathworks platform https://grader.mathworks.com/courses/9867-mdsp-ws-2019-20 according to the instructions given there for all problems marked with (Cx.y).
- For all problems marked with (Ax.y) refer to the quiz questions at https://www.moodle.tum.de/course/view.php?id=50238. There you can also find the homework rules and more information.
- Result verification: Check your functions as often as you want with Cody Coursework™.
- Lab session for MATLAB questions – see website, room 0943
- Notation: `variable name`, *`file name`*, `MATLAB_function()`

1 Image manipulation

1. (A1.1) Load image *`pears.jpg`* to variable `im` and analyze it: Determine data-type, dimension and plot the color histogram of the image. Calculate its minimum, maximum, mean and standard deviation. How do you interpret these values? *(4 points)*
2. (A1.2) Now copy `im` to `imd`, and convert the data-type of `imd` to *double*, which is MATLAB's default type required by many built-in functions. Also, by definition, image data of this type must lie in the range $[0; 1]$. Use this convention as the default for all your homework submissions. Display `imd` using at least 3 different MATLAB functions. What is the difference if you display `im` instead of `imd` using `imshow()`? *(3 points)*
3. (C1.1) Implement a function to draw a 2 pixel wide black border within the *double*-typed image, using sub-matrix addressing. Do not change its size. Verify your implementation using hidden and visible tests at Cody Coursework platform. *(5 points)*
4. (A1.3) Cody Coursework platform accepts a tolerance of 0.01 for the values of `imd_frame`. Add Gaussian noise with $\sigma = 0.005$ to that image and retry the tests. How many pixels (matrix elements) do you expect to lie outside the tolerance range? *(5 points)*
5. (C1.2) Implement a function to copy each second row and each third column (both starting at index 1) of the (noiseless) framed image and output it. *(4 points)*
6. (A1.4) Draw a red frame around the image you just displayed with the `plot()` function. How is this approach different to changing pixel values in `im`? Try out different options of the `plot()` function and add some text to the image with the `text()` function. *(4 points)*
7. (C1.3) In the following, you will practice sub-matrix addressing and use the Kronecker product to create colorized copies of one grayscale image, similar to "Warhol-style" images.
   Use the grayscale image `imd` from above as the input image. The output will be a color image of larger size. Calculate each color channel separately, using:

$$imk_i = c_i \otimes im \qquad \forall i \in \{R, G, B\} \text{ with}$$

$$c_R = \begin{bmatrix} 1 & 0 & 1 \\ 0.3 & 1 & 0 \\ 0 & 0.3 & 1 \end{bmatrix}, \qquad c_G = \begin{bmatrix} 0 & 1 & 1 \\ 0.3 & 0 & 1 \\ 0.7 & 0.3 & 0 \end{bmatrix}, \qquad c_B = \begin{bmatrix} 0 & 0 & 0 \\ 1 & 1 & 0 \\ 0.7 & 1 & 0 \end{bmatrix}$$

   Merge these channels to a color image and analyze it by showing on the screen. Do not use the `kron()` function – rather use sub-matrix addressing and for-loops stepping through all elements of $c_{R,G,B}$. (Usually, you should avoid for-loops in MATLAB for better performance.) *(7 points)*

2 Periodic Sequences

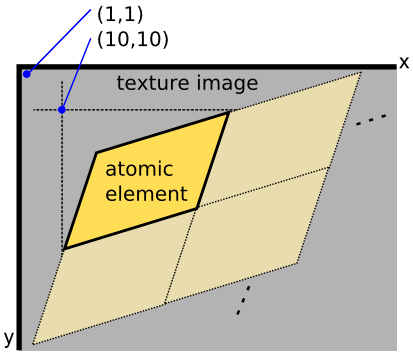

Figure 1: Placing and repetition of the atomic element in the original texture image

1. (C2.1) In this problem you will need to write a function to compute atomic image elements from the texture image ***texture1.jpg***. The atomic element is fully specified by the periodicity matrix N, the texture image, and its location (x, y). In (x, y), x is the location of the atomic element's leftmost pixel within the texture image, and y is the location of the topmost pixel, see Figure 1. Make sure the matrices for your atomic elements are of the minimal required size (with the given periodicity matrix). Verify that you create correct atomic image elements for $(x, y) = (1, 1)$ (MATLAB coordinates), $(x, y) = (10, 10)$ and $(x, y) = (50, 20)$ with following periodicity matrices:

$$N_A = \begin{bmatrix} 200 & 0 \\ 0 & 200 \end{bmatrix}, \qquad N_B = \begin{bmatrix} 200 & 200 \\ 0 & 200 \end{bmatrix}, \qquad N_C = \begin{bmatrix} 200 & 200 \\ -200 & 200 \end{bmatrix}.$$

   The function should return atomic image elements and masks (if used) for the given pair (x, y) as `double` images[1].
   Hints: You should work with binary image masks for non-rectangular elements. Create the required parallelograms using MATLAB's `poly2mask()` function (see `http://de.mathworks.com/help/images/ref/poly2mask.html`). When using `poly2mask()` pay attention that it uses a sub-pixel grid to decide which pixels are inside the specified periodicity matrix-based polygon. See Figure 2 for illustration. *(18 points)*

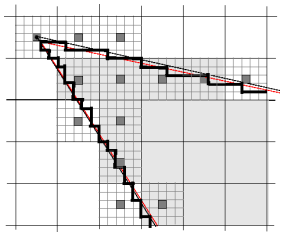

Figure 2: Illustration how `poly2mask()` determines which pixels are inside the region (taken from `de.mathworks.com`)

2. (C2.2) Here, you would need to implement an inverse operation: generate a periodic image for the given atomic element. For this you are given as input arguments an atomic element, atomic mask, periodicity matrix, base location and size of the image. The size of the periodic image should be at least the same size as the original texture image. *(18 points)*

[1] When saving a `double` image, make sure to use the correct scaling!

In the following and future problems, the normalized cross correlation (NCC) between two matrices i_1,i_2 is used to measure the similarity between two images. The coefficient is a scalar and generally calculated as follows:

$$\mathrm{ncc}(i_1, i_2) = \frac{1}{N} \sum_{x,y} \frac{(i_1(x,y) - \overline{i_1})(i_2(x,y) - \overline{i_2})}{\sigma_{i_1} \sigma_{i_2}} \tag{1}$$

With N — number of elements, $\bar{i}$ — mean, σ — standard deviation. Note that some MATLAB functions evaluate the NCC over a shift parameter by default.

3. (C2.3) Write a function to calculate NCC between two equal-sized periodic sequences (images in this case). Use as little for-loops as possible. Refrain from using the built-in MATLAB functions for computation of cross-correlation. *(8 points)*
4. (C2.4) In this task you will write a function to find the atomic element of a given naturally periodic texture image using exhaustive search and correlation. Work with image ***texture2.jpg*** with $(x, y) = (1, 1)$ and search for quadratic atomic elements of edge length $s \in [1, 300]$ pixels, using the following steps:
   - Create a quadratic atomic element of edge length s
   - Generate a periodic image from the atomic element
   - Compare the generated periodic image to the original image using NCC
   - Repeat until all edge lengths are tested

   Write a function that computes NCCs and returns them in ☞ `ncc_range` (length: 300 elements). Also, the function has to search for (the correct) local maxima in `atom_ncc` and save the corresponding edge lengths to ☞ `ncc_max_loc`. Check the correctness of your results by inspection. *(19 points)*
5. (A2.1) Why do you observe several maxima in the NCC values of the whole image? How can this be explained by the image properties? *(5 points)*