Mathematisch- Naturwissenschaftliche
Fakultät

Fachbereich Informatik
Arbeitsbereich Visual Computing

Eberhard Karls
Universität
Tübingen

Bildverarbeitung I (Prof. Schilling)
WS2017/2018

Assignment 2, Due: November 16th, 2017

Remarks

Please submit your exercises in ILIAS before 23:50 on the closing date. *Each* member of the group must be able to explain *each* exercise. Groups and members will be chosen at random and asked to present an exercise as a representative of the whole group. You should be prepared to explain any exercise at our biweekly tutorial.

Exercise 3: Multi-Image Denoising [1 point]

Use `my_loadImage` from the first assignment to load images for this and all upcoming assignments.

a) Use the provided image sequence a (cups) for this exercise. The provided images were shot with high ISO settings and contain serious amounts of noise. Implement the function `my_nFilter` to generate a denoised image from the set of five images.

b) Use the provided image sequence b (tree) for this exercise. The set contains five crops of an image sequence. You can find one of the original images on this page. Look at the result and discuss why the approach performs poorly for this sequence of images.

Exercise 4: Denoising [4 points]

The provided script loads two image files (`imageFile0` and `imageFile1`). Use the predefined filter parameters for `imageFile0` and adjust the parameters for `imageFile1` in such a way that the best possible results are achieved (you will have to modify the script `exercise04` directly).

a) Mean filter [1 point]: Write a function `my_meanFilter` and implement a mean filter as described in the lecture.

b) Median filter [1 point]: Write a function `my_medianFilter` and implement a median filter as described in the lecture.

c) Gaussian filter [2 points]: Write a function `my_gaussFilter` and implement a Gaussian filter as described in the lecture. It might be useful to define (local functions), e.g. to define a kernel.

Hint: The solutions do not care about boundary condition. Can you do better? :)

Reminder: Try to avoid for loops in order to prevent excessive runtime of your functions! Try to vectorize simple for loops and use the parallelization toolbox, if you have access to it. You could have a look at: `parfor`, `reshape` and `repmat`.

Exercise 5: Bilateral Filtering [4 points]

Read the provided paper thoroughly. Write a function `my_bilateralFilter` and implement bilateral filtering as described in the paper. Optimize the parameters `w`, `sigma0` and `sigma1` to achieve the best possible results.