Software Methodologies COMP2231

2018/2019

Image processing assignment

Part A - Bilateral filter

Bilateral Filter general description and main properties

Bilateral filtering is a procedure that takes a picture and produces a smoothed version of that picture whilst preserving its edges. The algorithm that it uses consists of taking each pixel and replacing it by a weighted mean of nearby pixels, so that neighbours that are further away or have different intensities get penalized and hence have a lower impact on the processed pixel. So, the 2 parameters that are passed to the bilateral filter function are σ_s -the spacial component (e.g. Euclidian distance) and σ_r -the range component (describes pixel intensity). A combination of these two parameters guarantees that only pixels close in terms of proximity and intensity values contribute to the final output.

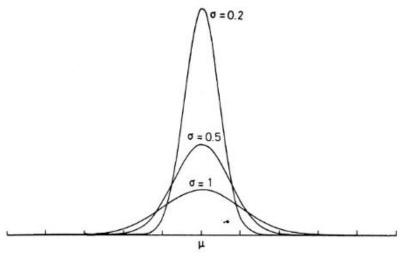
The equation below describes the bilateral filter:

$$BF[I]_{\mathbf{p}} = \underbrace{\frac{1}{W_{\mathbf{p}}}}_{\text{normalization}} \sum_{\mathbf{q}\in S} \underbrace{G_{\sigma_s}(\|\mathbf{p}-\mathbf{q}\|)}_{\text{space}} \underbrace{G_{\sigma_r}(|I_{\mathbf{p}}-I_{\mathbf{q}}|)}_{\text{range}} I_{\mathbf{q}}$$

where G(sigma_s) is a spacial Gaussian (depends on pixel location) that deacreases the influence of distant pixels and G(sigma_r) is a range Gaussian (depends on pixel values) that decreases the influence of pixel q with an intensity value different from I_p.

The Gaussian function is a normal distribution function (bell shaped).
$G(\sigma_s)$ and $G(\sigma_r)$ are Gaussians dependant on parameters σ_s and σ_r

- As the range parameter (sigma r) increases the Gaussian becomes flatter, so the bilateral filter becomes close to a Gaussian blur (so we almost ignore the range gaussian because it becomes constant).
- Increasing the spatial parameter (sigma s) helps smoothing out larger features.



In the program that I implemented, the bilateral filter is applied to each BGR colour channel separately with the same parameter sigma for all 3 channels. So, when the original pixel is compared to a pixel in the neighbourhood, the intensity values (for each colour channel) are subtracted and passed to the Gaussian function that returns higher value for intensity differences closer to zero and that will therefore have a bigger impact on the original pixel (assuming the pixels are spatially close to each other).

Choosing parameters for real-time applications:
Sigma values: For simplicity, in the program that I have implemented, I set the sigma values to be similar (sometimes the same). By trial and error method, I have observed that if the sigma values are small (< 10), the filter will not have a big effect, whereas if the values are large (> 150), the effect will be much stronger, producing a more “cartoonish” looking picture.

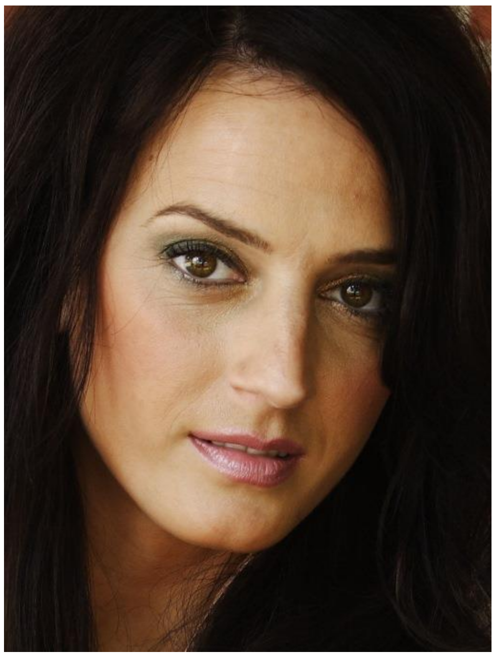
Filter size: Large filters (d > 5) are very slow, so my program mostly uses d=5 and d=7 as neighbourhood diameter for testing images in this coursework.

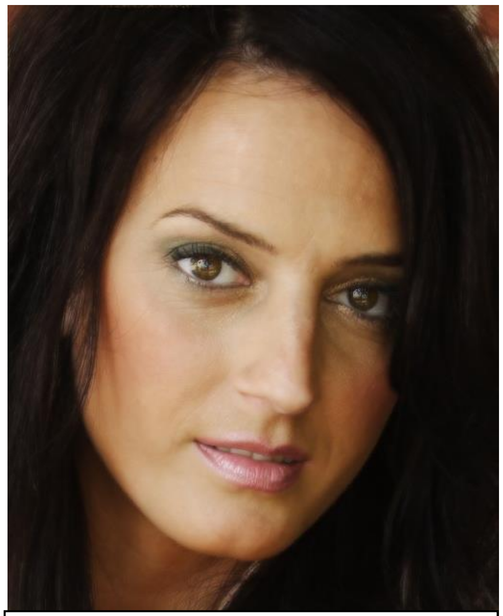
<u>Disadvantages:</u>
Bilateral filtering is a time-consuming technique as at each stage, the pixel intensity is calculated as an weighted average over a large neighbourhood. The bilateral filter is a nonlinear filter which means it can’t be implemented efficiently using different speedy techniques (like Fourier transform). However there exist some accelerated numerical schemes that have been specifically developed for Bilateral Filter (e.g. Gauss-Seidel iterations).

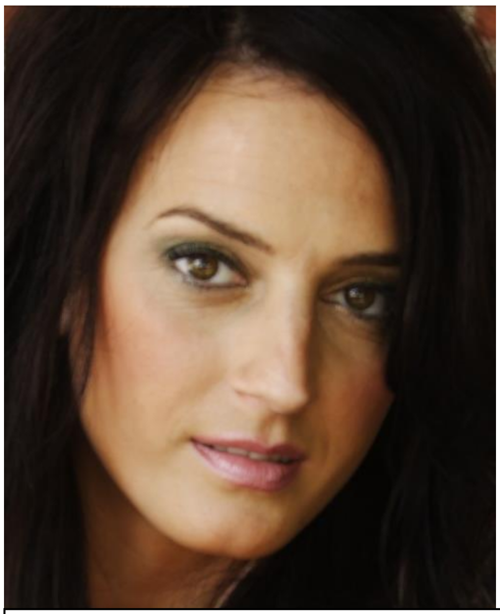
One simple and obvious characteristic of the bilater filtering is that when weights are multiplied if one of the weights' value is close to zero, no smoothing happens. For example, if a large spacial Gaussian is combined with a very small range Gaussian, it will produce very limited smoothing, if any, although the spacial weight has high potential. One interesting feature of the bilateral filter is that when applied iteratively, it can achive cartoon-like instances of pictures.

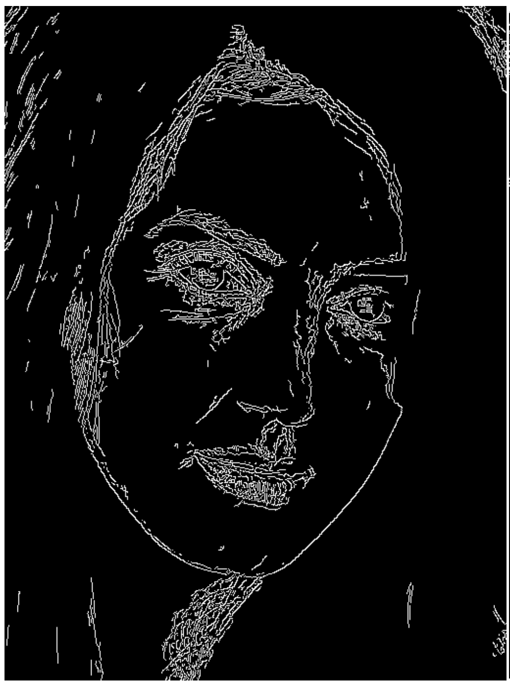
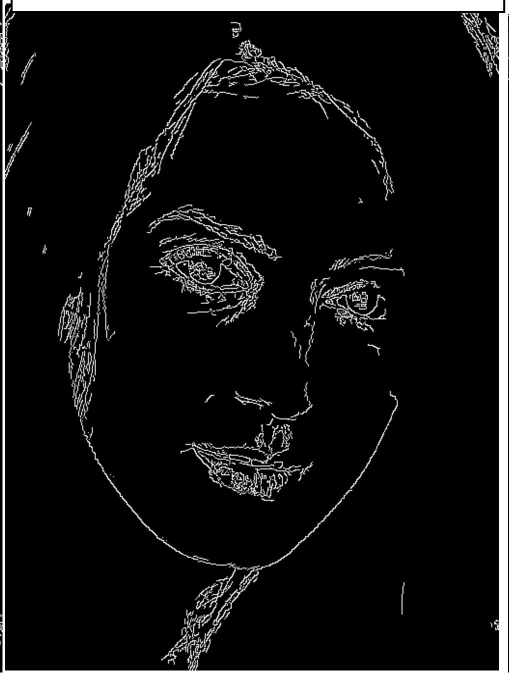
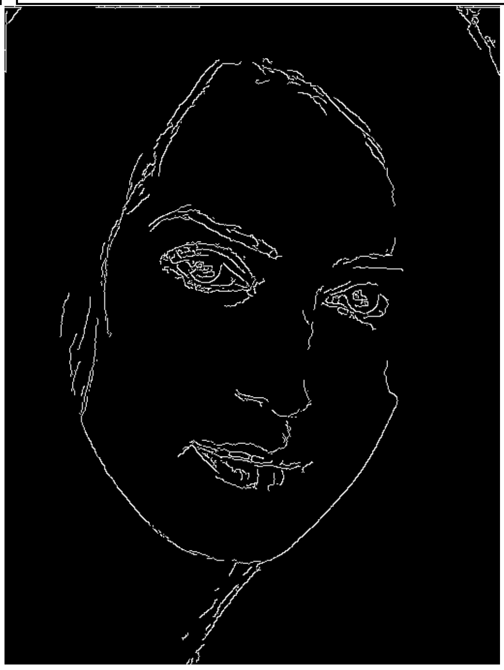
For filtering the test images I have implemented my own version of the bilateral filter function and I have compared them to the OpenCV results. In this section I will analyse the effects of the bilateral filter implemented by me (analogous to the OpenCV one) and compare them to the original input. As a confirmation that my bilateral filter function is doing what's it's meant to, I have additionally checked the edges of the images produced against the edges of the original ones using the OpenCV Canny function, the results of which I will present in this section.

At a first glance, we can observe that the appeareance of wrinkles in the test picture below has been considerably smoothed, whilst the main edges have been preserved. This demonstrates that the bilateral filter is working well.


Original picture


The sigma values and radius dimension used:
sigma_r=80 sigma_s=80 radius=5

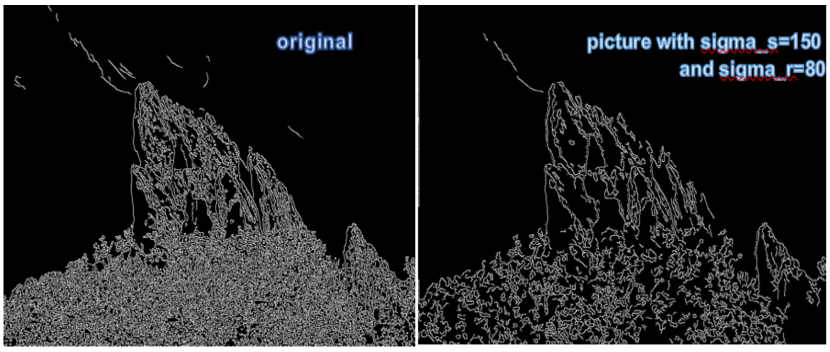

The sigma values and radius dimension used:
sigma_r=150 sigma_s=80 radius=5

Radius= 5, sigma_s=40, sigma_r=80

Radius= 5, sigma_s=80, sigma_r=80

Radius= 5, sigma_s=150, sigma_r=80



Experimentation
By trial and fail method, and experimenting with different parameters, I have managed to convince myself of the properties of the bilateral filter. We can see that by choosing sigma_r relatively small, the edge stopping function will suitably choose the weights of the pixel in the neighbourhood (will preserve the edges well), whilst by setting sigma_r big enough, it will start resembling a Gaussian blur. By increasing sigma_s, we can smooth out larger features. In the edge detected pictures we can observe that the main edges are preserved whilst finer ones get blurred out (like the wrinkles on the woman's face in the first picture and the branches of the tree in the second one)

Part B - Joint bilateral filter description, main properties and its applications

In order to understand how the joint bilateral filter works, I will first examine the difference between the two input images. The illumination in flash picture intensifies the signal-to-noise ratio (SNR) and offers a better high-frequency detail estimate. The flash image demonstrates a better SNR than the ambient image, because the scene brightness is much higher in the flash picture (even though not uniform). So, following from this remark, the goal is to alter the bilateral filter described above to calculate the edge-stopping function G(sigma r) using the flash image F instead of the ambient one A.

The equation below describes the joint bilateral filter:

$$\mathrm{JBF}\,[I]_{\mathbf{p}} = \underbrace{\frac{1}{W_{\mathbf{p}}}}_{\text{normalization}} \sum_{\mathbf{q}\in S} \underbrace{G_{\sigma_s}(\|\mathbf{p}-\mathbf{q}\|)}_{\text{space}} \underbrace{G_{\sigma_r}(|F_{\mathbf{p}}-F_{\mathbf{q}}|)}_{\text{range}} I_{\mathbf{q}}$$

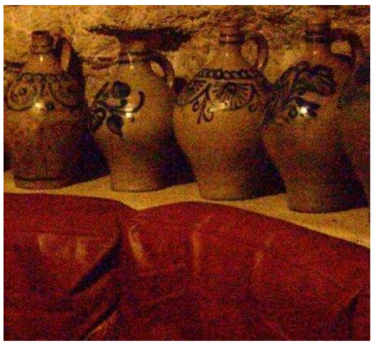
Where F(p) and F(q) are the intensities of original and neighbouring pixels on the flash image and I(q) the intensity of neighbouring pixel on the non-flash, ambient image.
By setting sigma s the same way we did for the bilateral filter and assuming the flash image has little noise, a small sigma r value can be set, and we can still guarantee that the edge-stopping function $G_{\sigma r}(F_p-F_q)$ will appropriately chose the weights of the neighbourhood pixels.

In the program that I implemented the Joint Bilateral Filter works the same way as the normal Bilateral filter with the correction that the Gaussian of sigma r is passed the parameters for the colour intensities from the **flash image**, passing each BGR channel value of the difference between 2 pixels intensity separately and multiplying the final result by the spacial Gaussian and the neighbourhood pixel from the **non-flash** image.

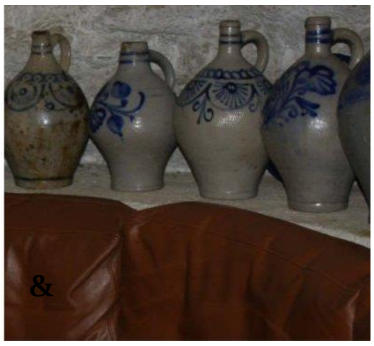
One direct consequence of using joint bilateral filter is that it fails in the shadowy regions of the picture that only appear in the flash image, causing it to under-blur or over-blur the ambient image in these regions. This can be solved by detecting flash shadows and applying a mask produced by the detection algorithm.

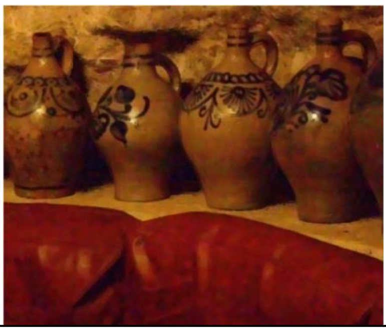
One application area of the joint bilateral filter that I found particularly interesting is the image and video abstraction. One implementation is the Kuwahara filtering. In this technique, the details in high contrast regions are removed and the shape boundaries in the low-contrast regions are preserved so that we achieve cartoon illustrations of the input image/video in real-time. Initially a structure tensor is calculated and smoothed with a Gaussian mask. Structure-aware smoothing is performed using joint bilateral filtering, getting an outcome defined as a weighted average, where the most amount of weight is given to averages with low standard deviation.

Inputs

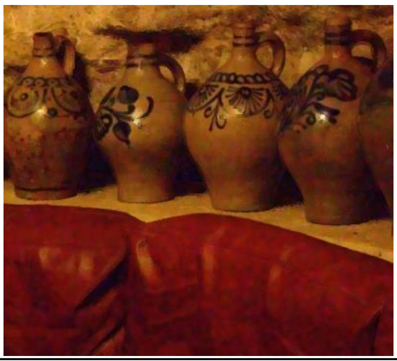


&

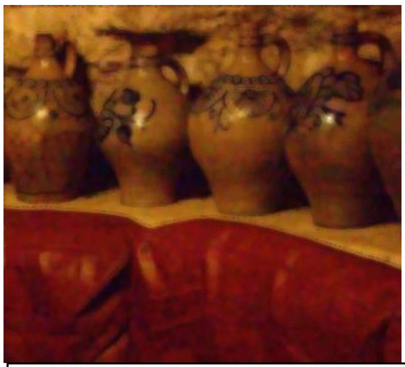




Sigma s and sigma r and radius values used:
sigma_r=5 sigma_s=60 radius=7



Sigma s and sigma r and radius values used:
sigma_r=5 sigma_s=120 radius=7



Sigma s and sigma r and radius values used:
sigma_r=120 sigma_s=120 radius=7

Experimentation:
As we can observe, from applying the joint bilateral filter, a visually more pleasing image is obtained by combining a filtered no-flash image with the flash residual to achieve the desired illumination with the high-quality structure. The edge detection function applied on the flash image produces the structure component that is then merged with the illumination component of the non-flash picture and generates a relighted and enhanced picture. The output of the test3a and test3b show that when combining a flash and no-flash picture, the extracted output picture has both the warm lighting of the no-flash picture and the crisp detail of the flash image. The same properties of sigmas apply as in the bilateral filater case.

References:
A Gentle Introduction to Bilateral Filtering and its Applications
(https://people.csail.mit.edu/sparis/bf_course/course_notes.pdf)
Bilateral Filter Wikipedia
(https://en.wikipedia.org/wiki/Bilateral_filter)
Image and Video abstraction
(https://pdfs.semanticscholar.org/a42e/d2cb3fbf1f9eb21ff469d40974006bd1ebdf.pdf