Solutions to Problem Set 6

Fundamentals of Simulation Methods 8 ECTS
Heidelberg University WiSe 20/21

Elias Olofsson
ub253@stud.uni-heidelberg.de

December 16, 2020

1 FFT-based convolution (8pts)

The normalization of the kernel

$$W(r) = k \begin{cases} 1 - 6\left(\frac{r}{h}\right)^2 + 6\left(\frac{r}{h}\right)^3 & \text{for } 0 \leq \frac{r}{h} < 1/2 \\ 2\left(1 - \frac{r}{h}\right)^3 & \text{for } 1/2 \leq \frac{r}{h} < 1 \\ 0 & \text{otherwise} \end{cases} \tag{1}$$

is done quick and easy through normal integration of $\int W(|\vec{r}|)\mathrm{d}^2\tilde{\mathbf{r}} = 1$. By noticing that the kernel is only dependent on radial distance, we can switch to polar coordinates and do a variable substitution $u = r/h$ and obtain

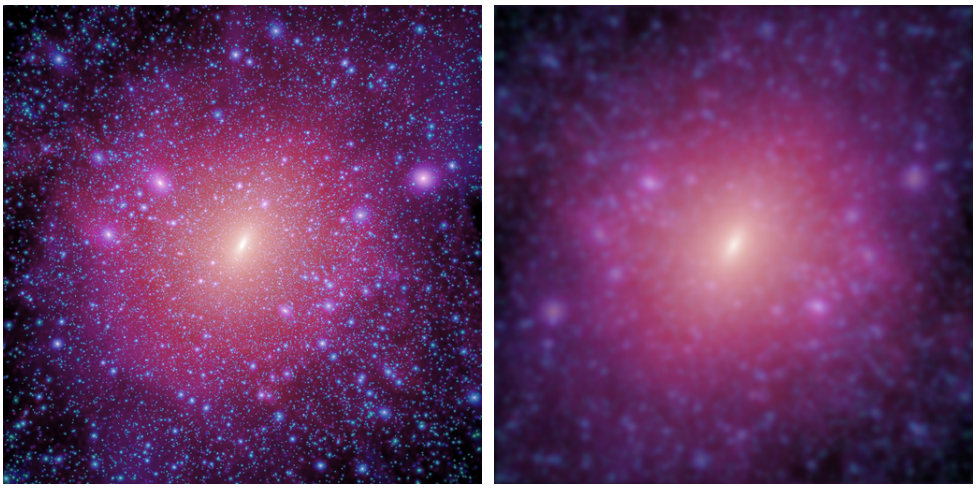
$$\begin{aligned} \int W(|\vec{r}|)\mathrm{d}^2\tilde{\mathbf{r}} &= 2\pi \int_0^\infty W(r) r dr && (2) \\ &= 2\pi h^2 \left[\int_0^{1/2} W_1(uh) u du + \int_{1/2}^1 W_2(uh) u du \right] && (3) \\ &= 2\pi k h^2 \left[\int_0^{1/2} (1 - 6u^2 + 6u^3) u du + \int_{1/2}^1 2(1-u)^3 u du \right] && (4) \\ &= \pi k h^2 \frac{7}{40} && (5) \\ \implies \quad k &= \frac{40}{7\pi h^2}, && (6) \end{aligned}$$

which then properly normalizes the kernel.

For implementation specific details regarding the convolution of the given image with the kernel in eq.(1), please see the attached code `smooth_image.c`. Here, I have set up the kernel in real space such that it is centered at the pixel with indices $i = j = 0$, but mirrored "backwards" in each dimension, such that pixels in each corner are effectively the closest ones to each other. Using this configuration, I seem to get good smoothing results of the original image. A comparison of the supplied image and my smoothed version of it can be seen in fig.(1). However, one can not that there seems to be some mirroring going on where distinct points of light at one side of the original

image, suddenly has a "twin" opposite itself in the smoothed version. Thus, the smoothing has not only a blurring effect, but also mirrors objects relative to the center of the image.

I'm not sure if this is due to an incorrect implementation on my part, or if this is inherent in the theory of DFT, and the specific way it is used here. Unfortunately, I did not have time to conduct a better and deeper analysis into why this is the case, or whether if I have made a mistake. My intuition however is that most likely it was an error on my behalf, since doing a Fourier transform back and fourth, even if I did multiply it with the transformed kernel in between, should conserve most, if not all, of the real space information.



(a) **(b)**

Figure 1 – The original and the smoothed image.

A further comparison was done by printing the summed pixel values of each color channel for the original image and the smoothed version. The printouts can be seen below.

```
Sum of pixel values per color channel, original image.
Red   : 25182443.00
Green : 12287845.00
Blue  : 24795222.00

Sum of pixel values per color channel, smoothed image.
Red   : 25182345.90
Green : 12287797.62
Blue  : 24795126.40
```

Looking at these values and comparing the sums, it conforms with what we can visually see in the images, that both of the images overall are very

similar. There is however a slight difference in these summed values, and I am not fully sure if this is a numerical artifact or an error of my implementation, as discussed above. Nonetheless, similarly to above I am more leaning towards that this is a mistake on my part. I believe that a correct implementation should yield exactly (or extremely close to) the same summed value, since otherwise this would physically imply that charge or mass was destroyed or created during the process of the Fourier transformations, if this was describing a mass density or charge density field instead of a color image. I did not however have more time to delve into this deeper, and I didn't even have time to do the second exercise of this problem solving set. Sorry about that.