beeldverwerken 2

Danny Dijkzeul 10554386
Joram Wessels 10631542

April 25, 2016

1

$$(g * f)(k) = \sum_l f(k-l)g(l)$$

1. $g = \{0\ 0\ 0\ 0\ \underline{1}\ 1\ 1\ 1\ 1\}$, $f = \{1\ \underline{2}\ 1\}$
   $(g * f)(-4) = f(-1)g(-4-(-1)) + f(0)g(-4-0) + f(1)g(-4-1) = 1 \cdot 0 + 2 \cdot 0 + 0 \cdot 0 = 0$
   $(g * f)(-3) = f(-1)g(-3-(-1)) + f(0)g(-3-0) + f(1)g(-3-1) = 1 \cdot 0 + 2 \cdot 0 + 0 \cdot 0 = 0$
   $(g * f)(-2) = f(-1)g(-2-(-1)) + f(0)g(-2-0) + f(1)g(-2-1) = 1 \cdot 0 + 2 \cdot 0 + 0 \cdot 0 = 0$
   $(g * f)(-1) = f(-1)g(-1-(-1)) + f(0)g(-1-0) + f(1)g(-1-1) = 1 \cdot 1 + 2 \cdot 1 + 0 \cdot 0 = 1$
   $(g*f)(0) = f(-1)g(0-(-1)) + f(0)g(0-0) + f(1)g(0-1) = 1 \cdot 1 + 2 \cdot 1 + 1 \cdot 0 = 3$
   $(g*f)(1) = f(-1)g(1-(-1)) + f(0)g(1-0) + f(1)g(1-1) = 1 \cdot 1 + 2 \cdot 1 + 1 \cdot 1 = 4$
   $(g*f)(2) = f(-1)g(2-(-1)) + f(0)g(2-0) + f(1)g(2-1) = 1 \cdot 1 + 2 \cdot 1 + 1 \cdot 1 = 4$
   $(g*f)(3) = f(-1)g(3-(-1)) + f(0)g(3-0) + f(1)g(3-1) = 1 \cdot 1 + 2 \cdot 1 + 1 \cdot 1 = 4$
   $(g*f)(4) = f(-1)g(4-(-1)) + f(0)g(4-0) + f(1)g(4-1) = 1 \cdot 1 + 2 \cdot 1 + 1 \cdot 1 = 4$
   $(g * f) = [0, 0, 0, 1, 3, 4, 4, 4, 4]$
   As for border cases, the missing values beyond the border are assumed to continue the last element, since it resembles a signoid funcion.

2. Since convolution is both associative and commutative, $(f * g)$ would result in exactly the same equations as as $(g * f)$. Hence, the equations above apply here as well.

3. $g = \{0\ 0\ 0\ 0\ \underline{1}\ 1\ 1\ 1\ 1\}$, $f = \{-1\ \underline{1}\}$
   For $k = -4$ through $k = -1$, $(g * f) = 0$
   $(g * f)(0) = f(-1)g(0 - (-1)) + f(0)g(0 - 0) = 1$
   For $k = 1$ through $k = 4$, $(g * f) = 0$
   Indices beyond the borders are assumed to continue the last element, since it resembles a signoid function.

4. $$(g * f)(k) = \sum_{l'=-\infty}^{\infty} g(l')f(k - l')$$

   $l = k - l' \ \rightarrow \ l' = k - l$

   $$\sum_{l=\infty}^{-\infty} g(k - l)f(l) = \sum_{l=-\infty}^{\infty} f(l)g(k - l) = (f * g)$$

   From ∞ to $-\infty$ or the other way around yield the same results. Ergo: $(f * g) = (g * f)$

5. $$((f*g)*h)(k) = \sum_{l=-\infty}^{\infty} (f*g)(l)h(k-l) = \sum_{l=-\infty}^{\infty} [\sum_{m=-\infty}^{\infty} f(m)g(l-m)]\, h(k-l)$$

6. g = $\{\underline{1}\}$

7. g = $\{\underline{3}\}$

8. $g = \begin{pmatrix} 0 & 0 & 0 & \underline{0} \\ 1 & 0 & 0 & 0 \end{pmatrix}$

9. No rotating an image around an arbitrary angle isn't possible, as it would be dependent on the size of the convolution matrix.

10. $$g = \begin{pmatrix} \frac{1}{9} & \frac{1}{9} & \frac{1}{9} \\ \frac{1}{9} & \underline{\frac{1}{9}} & \frac{1}{9} \\ \frac{1}{9} & \frac{1}{9} & \frac{1}{9} \end{pmatrix}$$

11. No, it's not possible to get the median of a 3 by 3 neighbourhood, because it is not possible to see which values are higher than others.

12. It is not possible to get het minimum value of a 5 by 5 neighbourhood, because it is not possible to see which values are higher than others.

13. $g = \begin{pmatrix} \frac{1}{5} & \frac{1}{5} & \frac{1}{5} & \frac{1}{5} & \underline{\frac{1}{5}} \end{pmatrix}$

14. with $\sigma = 0.5$ pixels, $g = \begin{pmatrix} 0.017 & 0.017 & 0.017 \\ 0.017 & \underline{0.862} & 0.017 \\ 0.017 & 0.017 & 0.017 \end{pmatrix}$

15. $g = \begin{pmatrix} -1 & -1 & -1 \\ -1 & \underline{8} & -1 \\ -1 & -1 & -1 \end{pmatrix}$

16. $g = \frac{1}{2}\begin{pmatrix} -1 & \underline{0} & 1 \end{pmatrix}$

17. $g = \frac{1}{4}\begin{pmatrix} 1 & 0 & \underline{-2} & 0 & 1 \end{pmatrix}$

18. Zooming in isn't possible because the convolutional kernel is the same for every cell, whereas a zoom would require a convolution that only uses values in the direction of the origin.

19. No if-statements can be used in a convolutional kernel, and thus thresholding isn't possible.

Matlab examples:

```
F = [1 2 3 ; 2 3 4 ; 3 4 5];
Translation:
G = [0 0 ; 1 0];
imfilter(F, G, 'conv', 'replicate')
ans =
    2 3 3
    3 4 4
    4 5 5

Motion blur:
G = [0.5 0.5];
imfilter(F, G, 'conv', 'replicate')
ans =
    1.5 2.5 3.0
    2.5 3.5 4.0
    3.5 4.5 5.0

Unsharp image masking:
G = [-1 -1 -1 ; -1 8 -1 ; -1 -1 -1];
imfilter(F, G, 'conv', 'replicate')
ans =
    -6 -3  0
    -3  0  3
     0  3  6
```

20. Smoothness of a function is the amount of continuous derivatives it has. Because its derivatives have a periodic progression, the Gaussian has a smoothness of arbitrary order; any order derivative is continuous.

21. So that both the derivative as the Gaussian blur convolution kernals can be merged into a derivative Gaussian convolution that can be applied to all images.

22. Given that 1) taking derivatives can be represented as a convolution, 2) convolutions are associative, and 3) convolutions are commutative, one can deduce that $(D * I) * G = (D * G) * I$, where $(D * G)$ is the Gaussian derivative δG in convolution form.

23. A normal Gaussian: $G(x, \sigma) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{x^2}{2\sigma^2}}$
    The derivative of a Gaussian:
    $\frac{\delta G(x,\sigma)}{\delta x} = \frac{x}{\sigma^3\sqrt{2\pi}} e^{-\frac{x^2}{2\sigma^2}} = -\frac{x}{\sigma^2}\frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{x^2}{2\sigma^2}} = -\frac{x}{\sigma^2} G(x,\sigma)$

24. $\frac{\delta^2 G(x,\sigma)}{\delta x^2} = \frac{\delta}{\delta x}\frac{\delta G(x,\sigma)}{\delta x} = \frac{\delta}{\delta x}(-\frac{x}{\sigma^2} G(x,\sigma)) = (\frac{\delta}{\delta x}(-\frac{x}{\sigma^2}))G(x,\sigma) + (-\frac{x}{\sigma^2})\frac{\delta G(x,\sigma)}{\delta x}$
    $= (-\frac{1}{\sigma^2})G(x,\sigma) + (-\frac{x}{\sigma^2})(-\frac{x}{\sigma^2}G(x,\sigma)) = -\frac{1}{\sigma^2}G(x,\sigma) + \frac{x^2}{\sigma^4}G(x,\sigma)$
    $= (\frac{x^2}{\sigma^4} - \frac{1}{\sigma^2})G(x,\sigma)$

25. As convolutions are associative and commutative, smoothing an image twice is equivalent to smoothing it with the convolution of two Gaussians. The convolution of two Gaussians is another Gaussian with $\sigma = \sigma_1 + \sigma_2$.

26.

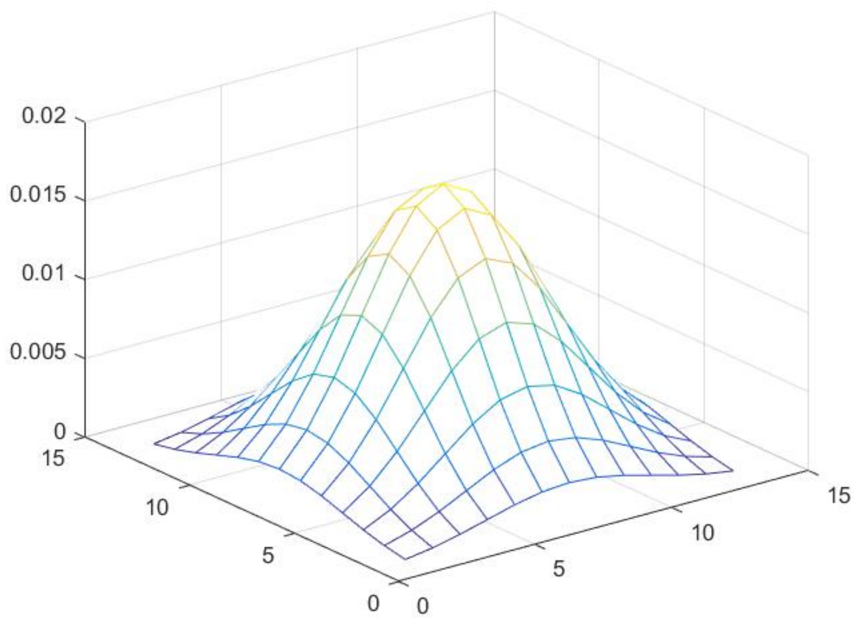
27.

2 Implementation of Gaussian Derivatives

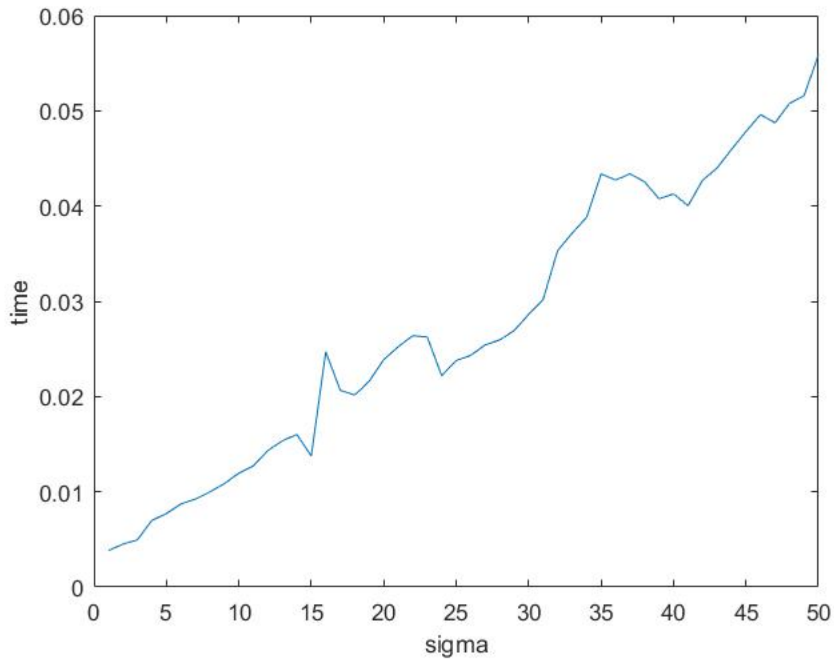
-

- You would expect it to equal 1 as the area undearneath a Gaussian always adds up to 1. The Matlab calculation approaches, but doesn't equal, 1. This can be solved by distributing the area beneath the infinite sides among the last samples.

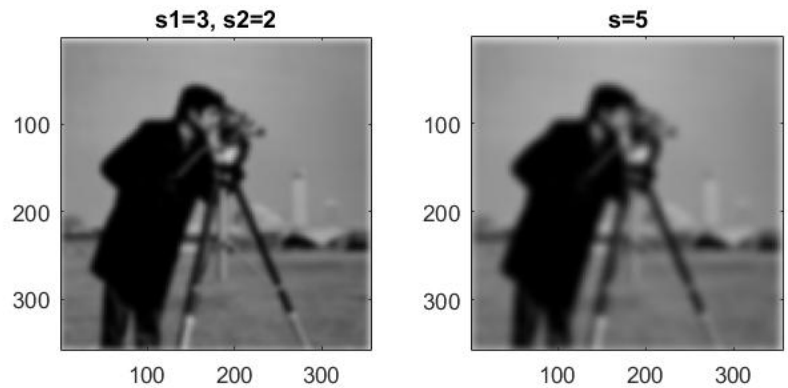
- mesh(Gauss(3))



- The physical unit of σ is pixels.
- Using the average of 10 samples per sigma on **cameraman.jpg**:



- The order of computational complexity in terms of the scale is $O(sigma)$, as the time increases linearly with the sigma.
- Blur of a blur, and the double blur



3 The Canny Edge Detector