RWTH Aachen University
Faculty of Mathematics, Computer Science and Natural Sciences
Chair of Computer Science 13 (Computer Vision)
Prof. Dr. Bastian Leibe

Seminar Report

Linear and Nonlinear Filters

Alexander Skretting
Matriculation Number: 445457

Jose Rigel Soeryo Soebandoro
Matriculation Number: 444345

June 2023

Advisor: George Lydakis

Abstract

Contents

1 Introduction

2 Linear Filters

With **Linear Filtering**, or specifically linear spatial filtering, the function which is used to pass the image through must be linear and shift invariant. The function $\mathbf{L}$ is considered linear if there exists two constants a and b for any two input pixels $f_1(m,n)$ and $f_2(m,n)$ such that

$$\begin{aligned}&\implies g_1(m,n) = \boldsymbol{L}[f_1(m,n)] \land g_2(m,n) = \boldsymbol{L}[f_2(m,n)]\\&\implies a \cdot g_1(m,n) + b \cdot g_2(m,n) = \boldsymbol{L}[a \cdot f_1(m,n) + b \cdot f_2(m,n)]\end{aligned}$$

also called the *superposition property of linear systems*[BA09]. Whereas a function is considered *shift invariant* when

$$\begin{aligned}&\implies g(m,n) = \boldsymbol{L}[f(m,n)]\\&\implies g(m-p,n-q) = \boldsymbol{L}[f(m-p,n-q)]\end{aligned}$$

p and q implies a spatial shift to both the input and output pixels[BA09]. An intuitive way to think of these two properties is that the function has to behave the same throughout the entire picture.

2.1 Correlation and Convolution Filtering

A common formula for linear filtering is the *Correlation Filtering*[BA09].

$$g(i,j) = \sum_{l \in \mathcal{M}} \sum_{k \in \mathcal{N}} f(i+k, j+l) \cdot h(k,l)$$

or commonly notated as $g = f \otimes h$.

The desired output pixel $g(i,j)$, where i and j specify the coordinates of it, is based on a $M \times N$ sized neighborhood, meaning not only does one pixel define an output pixel, but also a specified number of its neighbors. The influence of each pixel in the neighborhood is defined by the filter coefficient $h(k,l)$, also called its *kernel* or *mask*. [Sze22]

$$\underbrace{\begin{bmatrix}128 & 34 & 123\\ 68 & 54 & 73\\ 100 & 95 & 17\end{bmatrix}}_{\text{input neighborhood}} \otimes \underbrace{\begin{bmatrix}0.1 & 0.1 & 0.1\\ 0.1 & 0.2 & 0.1\\ 0.1 & 0.1 & 0.1\end{bmatrix}}_{\text{kernel}} = 75$$

As the above example with a 3×3 kernel, a total of 9 pixels is needed to calculate a single output pixel.

Another common variant on the formula is having the signs of the offsets reversed.

$$g = f * h$$

$$g(i,j) = \sum_{l \in \mathcal{M}} \sum_{k \in \mathcal{N}} f(i-k, j-l) \cdot h(k,l)$$

With this formula, $*$ is called the *convolutional* operator, and the kernel h is called the *impulse response function*. An interesting note is that, when the kernel h is convolved with an impulse signal δ (an image with 0 everywhere except the origin), it reproduces the kernel itself $h * \delta = h$, whereas with correlational filtering, it produces the reflected signal (inverted signal in both dimensions).[Sze22] In cases where the kernel is symmetrical on both axis (e.g. box blur), the result of convolutional and correlational is the same.

(a) correlation (b) convolution

Figure 1: Difference between Correlation and Convolution Filtering by translating 32 pixels

An apparent problem from neighborhood filtering is that on the edges, the neighbors simply does not exist in one or two directions (e.g. a 1000×1000 image passed through a 3×3 kernel would produce a 998×998 image). There are a couple method to alleviate the calculation of the nonexistent neighbors.

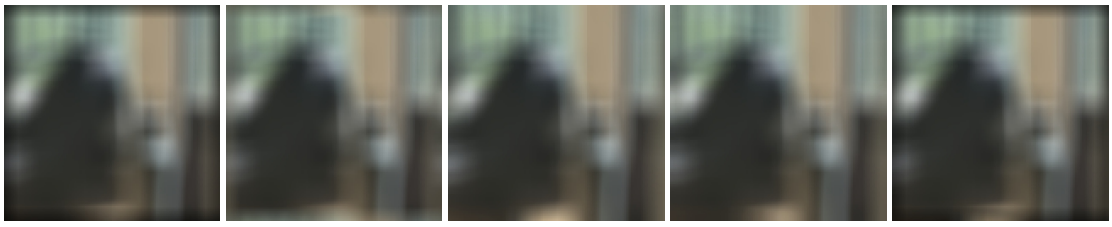
2.2 Padding (Border Effects)

(a) zero (b) wrap (c) clamp (d) mirror (e) extend

Figure 2: Different types of border padding

- **Zero:** Set all pixels outside the region to zero.
- **Constant:** This is similar to Zero, but instead a specific color is chosen to replace the out of bound pixels.
- **Clamp:** The edge pixels are repeated indefinitely.
- **(Cyclic) Wrap (Repeat or Tile):** The entire image is repeated similar to tiling it infinitely.
- **Mirror:** The image is mirrored accordingly on each of the edges.
- **Extend:** This is a combination of clamping and mirroring, where clamped image is subtracted with the mirrored image.

This of course not only affects the unseen pixels, but the effects can bleed into the pixels inside the frame.



(a) zero (b) wrap (c) clamp (d) mirror (e) extend

Figure 3: Effects of different paddings with box blur

There are other ways to handle the borders such as two-phase duplication, which is similar to clamp but alternates between the last 2 pixels. A change in the filter function per individual basis can also be implemented, but it might not be practical, as large amounts of additional logic may be needed[BA18].

2.3 Separable Filtering

The time complexity to calculate a single pixel can be said to be $O(M \cdot N)$ as it is dependent on the size of the kernel. For each of the pixel in the kernel, it must be both multiplied and summed together with its neighbors. This can be in practice sped up by separating the horizontal and vertical operations, hence performing a one dimensional convolution horizontally, then one additional operation for the vertical convolution. This kernel in this case is the outer product of two one dimensional kernels[Sze22].

$$\boldsymbol{K} = v \otimes h = \begin{bmatrix} v_1h_1 & v_1h_2 & v_1h_3 & \dots & v_1h_n \\ v_2h_1 & v_2h_2 & v_2h_3 & \dots & v_2h_n \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ v_mh_1 & v_mh_2 & v_mh_3 & \dots & v_mh_n \end{bmatrix}$$

Also notated as $K = vh^{\boldsymbol{T}}$, where $\boldsymbol{T}$ denotes tranposing h.
A method to check whether a given Kernel is separable is to use *Singular Value Decomposition (SVD)*[Sze22]

$$\boldsymbol{K} = \sum_i \sigma_i \cdot u_i \cdot v_i^{\boldsymbol{T}}$$

or more generally

$$\begin{aligned} \boldsymbol{K} &= \mathbf{U}_{m\times p}\boldsymbol{\sum}_{p\times p}\mathbf{V}^{\boldsymbol{T}}_{p\times n} \\ \boldsymbol{K} &= \begin{bmatrix} u_0 & \dots & u_{p-1} \end{bmatrix} \begin{bmatrix} \sigma_0 & & \\ & \ddots & \\ & & \sigma_{p-1} \end{bmatrix} \begin{bmatrix} v_0 \\ \vdots \\ v_{p-1} \end{bmatrix} \end{aligned}$$

Note that it requires the number of rows and columns must be the same, we can also alleviate this by putting 0 appropriately, to allow the kernel to be a square matrix.Then the first singular value σ_0 is checked. If it is the only one that is non zero, and the rest is zero, then the kernel is separable. As an example, the sobel filter, which is used for edge detection, can be separated into two separate operations. In this case, this is for the horizontal derivative approximation.

$$\frac{1}{8}\begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix} = \frac{1}{8}\begin{bmatrix} -1 & 0 & 1 \end{bmatrix}\begin{bmatrix} 1 \\ 2 \\ 1 \end{bmatrix}$$

The normalization of the operators are easily calculated by dividing each operator with the absolute sum of the entries.

$$\begin{bmatrix} -1 & 0 & 1 \end{bmatrix} \Rightarrow |-1| + |0| + |1| = 2 \Rightarrow \frac{1}{2}$$

Then the normalized operator is

$$\frac{1}{2}\begin{bmatrix} -1 & 0 & 1 \end{bmatrix}$$

it applies to the vertical operator as well

$$\frac{1}{4}\begin{bmatrix} 1 \\ 2 \\ 1 \end{bmatrix}$$

The following are more examples of common separable filtering.

Box	Bilinear	Gaussian	Sobel	Corner
$\frac{1}{K^2}\begin{bmatrix} 1 & 1 & \dots & 1 \\ 1 & 1 & \dots & 1 \\ \vdots & \vdots & \ddots & \vdots \\ 1 & 1 & \dots & 1 \end{bmatrix}$	$\frac{1}{16}\begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix}$	$\frac{1}{256}\begin{bmatrix} 1 & 4 & 6 & 4 & 1 \\ 4 & 16 & 24 & 16 & 4 \\ 6 & 24 & 36 & 24 & 6 \\ 4 & 16 & 24 & 16 & 4 \\ 1 & 4 & 6 & 4 & 1 \end{bmatrix}$	$\frac{1}{8}\begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix}$	$\frac{1}{4}\begin{bmatrix} 1 & -2 & 1 \\ -2 & 4 & -2 \\ 1 & -2 & 1 \end{bmatrix}$
$\frac{1}{K}\begin{bmatrix} 1 & 1 & \dots & 1 \end{bmatrix}$	$\frac{1}{4}\begin{bmatrix} 1 & 2 & 1 \end{bmatrix}$	$\frac{1}{16}\begin{bmatrix} 1 & 4 & 6 & 4 & 1 \end{bmatrix}$	$\frac{1}{2}\begin{bmatrix} -1 & 0 & 1 \end{bmatrix}$	$\frac{1}{2}\begin{bmatrix} -1 & 2 & 1 \end{bmatrix}$

Note that for Corner filtering, the kernel is not normalized. With some filters, it is not always possible to separate it into just two operations, therefore it can be transformed into a series of 1 dimensional operations. However, this may be impractical as it increases the time and space complexity.

References

[BA09] Alan C. Bovik and Scott T. Acton. Chapter 10 - basic linear filtering with application to image enhancement. In Al Bovik, editor, *The Essential Guide to Image Processing*, pages 225–239. Academic Press, Boston, 2009.

[BA18] Donald G. Bailey and Anoop S. Ambikumar. Border handling for 2d transpose filter structures on an fpga, Nov 2018.

[Sze22] Richard Szeliski. *3.2 Linear filtering*, page 119–131. Springer, 2nd edition, 2022.