

Bilateral Filtering

Computer Vision I (CSCI 3240U)

Faisal Z. Qureshi

<http://vclab.science.ontariotechu.ca>



Week 5 at a glance

- ▶ **Lecture 1 (Oct 6): Bilateral filtering (Sec. 3.3.1, 3.3.2)**
 - ▶ The problem with linear smoothing
 - ▶ Non-linear filters: the median
 - ▶ The bilateral filter
 - ▶ Why it is not a convolution
- ▶ **Lecture 2 (Oct 8): Midterm 1**

Reading: Szeliski Sec. 3.3.1, 3.3.2. Exercise w05a. **No lab this week.**

Midterm 1 covers Weeks 1–4. Today's material is **not** on it; it will be examined on Midterm 2 instead.

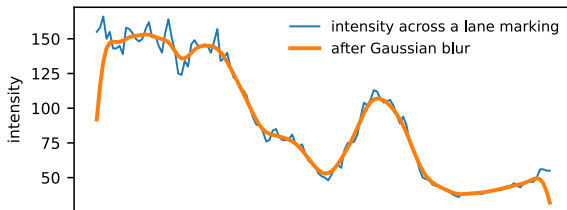
The problem with linear smoothing

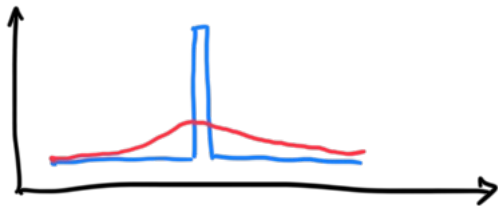
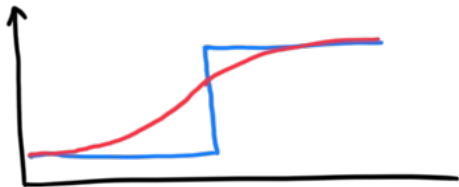
A Gaussian filter removes noise by averaging over a neighbourhood. But it averages *everything* in that neighbourhood, including pixels on the other side of an edge. So:

- ▶ noise goes down, **and**
- ▶ edges get blurred.

The two are inseparable for a linear shift-invariant filter, because the same fixed kernel is applied everywhere regardless of content.

To do better, the filter has to **look at the values**, not just the positions.

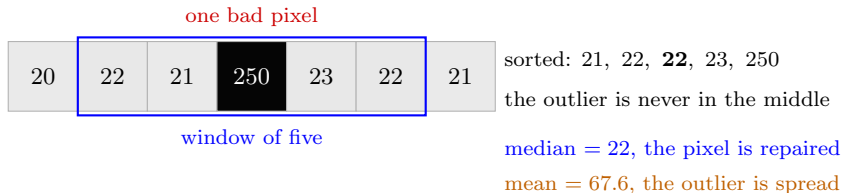




A first non-linear filter: the median

Replace each pixel by the **median** of its neighbourhood.

- ▶ Excellent against **salt-and-pepper** noise: an isolated extreme value is never the median, so it vanishes completely.
- ▶ Preserves step edges: on one side of an edge the majority of the window is still on that side.
- ▶ But it destroys fine texture and rounds corners.



It is non-linear: $\text{med}(a + b) \neq \text{med}(a) + \text{med}(b)$. Non-linearity is what buys us edge preservation, and it costs us the whole convolution toolkit.

20

22	21	22 250	23	22	21
----	----	----------------------	----	----	----

21, 22, 22, 23, 250

223 220

210	210 5	250
-----	---------------------	-----

137 199

3, 210, 250



20 22 21 250 23 22 21

21 97 98 99 22

21 22 23 23 22

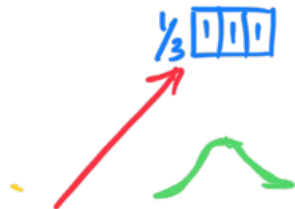
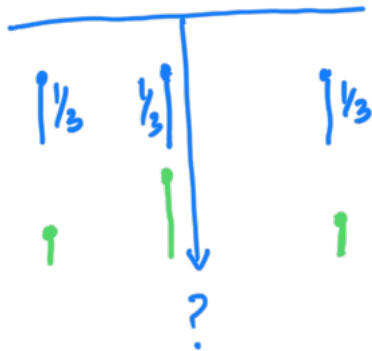
$\frac{1}{3}$

1	1	1
---	---	---

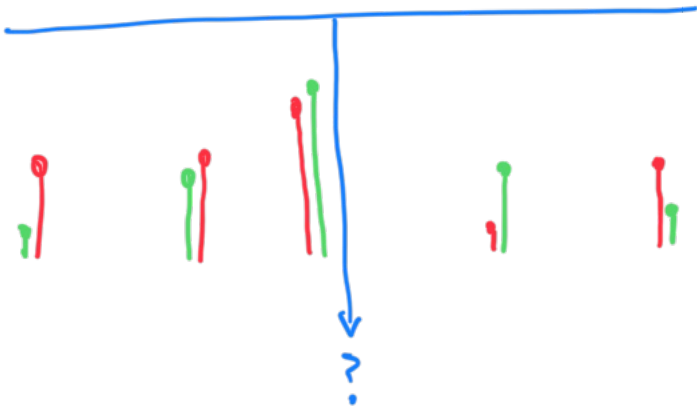


← median

20 22 21 250 23 22 21



20 22 21 250 23 22 21



20

22

21

20

23

22

21



20

22

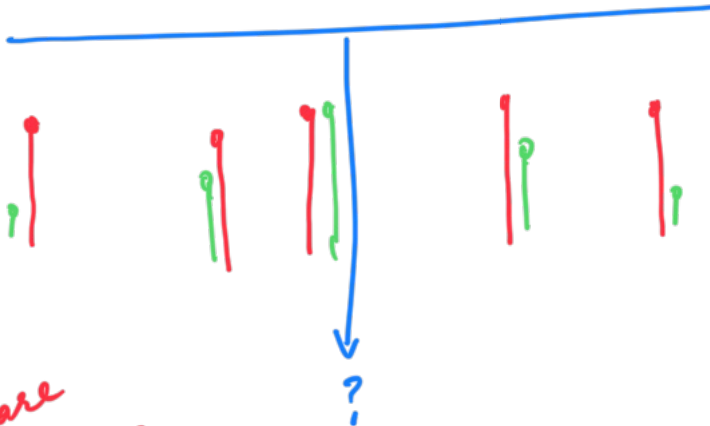
21

23

23

22

21



Edge-aware
filtering.

The bilateral filter: the idea

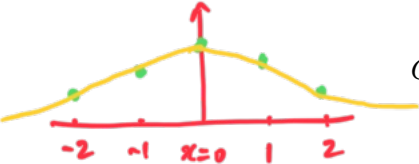
Keep the Gaussian's spatial weighting, but **multiply in a second weight** that depends on how similar the intensities are.

- ▶ A neighbour that is **nearby** *and* **similar in value** contributes a lot.
- ▶ A neighbour that is nearby but **very different in value**, i.e. across an edge, contributes almost nothing.

So averaging happens **within** a region and stops at its boundary.

The Gaussian, written out

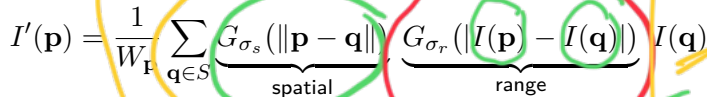
Both weights we are about to multiply are the same function, the 1D Gaussian:


$$G_{\sigma}(x) = \frac{1}{\sigma\sqrt{2\pi}} \exp\left(-\frac{x^2}{2\sigma^2}\right)$$

- ▶ σ sets the width: about 68% of the weight lies within $\pm\sigma$ of the centre, and 95% within $\pm 2\sigma$.
- ▶ The constant in front makes it integrate to 1.
- ▶ It is largest at $x = 0$ and falls off smoothly, never reaching zero.

Here the constant does not matter. Every weight is divided by W_p , so any scale factor cancels. Implementations write $\exp(-x^2/2\sigma^2)$ and let the normaliser sort it out.

The bilateral filter: the maths



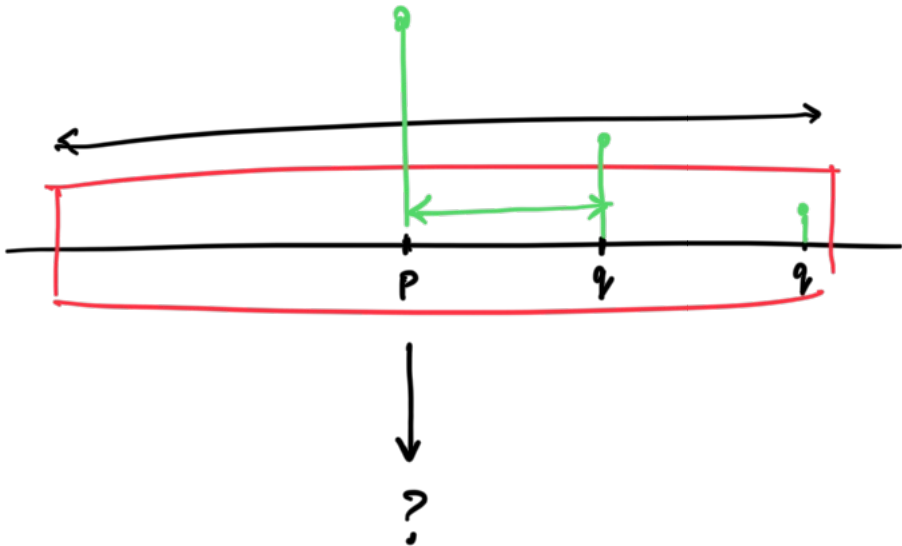
The diagram shows the bilateral filter equation with several annotations. A large yellow oval encircles the entire equation. A green oval encircles the spatial kernel term $G_{\sigma_s}(\|\mathbf{p} - \mathbf{q}\|)$, which is labeled 'spatial' with a bracket underneath. A red oval encircles the range kernel term $G_{\sigma_r}(\|I(\mathbf{p}) - I(\mathbf{q})\|)$, which is labeled 'range' with a bracket underneath. Within the red oval, the terms $I(\mathbf{p})$ and $I(\mathbf{q})$ are each individually circled in green. The summation is over $\mathbf{q} \in S$, and the entire expression is multiplied by $I(\mathbf{q})$.

$$I'(\mathbf{p}) = \frac{1}{W_{\mathbf{p}}} \sum_{\mathbf{q} \in S} \underbrace{G_{\sigma_s}(\|\mathbf{p} - \mathbf{q}\|)}_{\text{spatial}} \underbrace{G_{\sigma_r}(\|I(\mathbf{p}) - I(\mathbf{q})\|)}_{\text{range}} I(\mathbf{q})$$

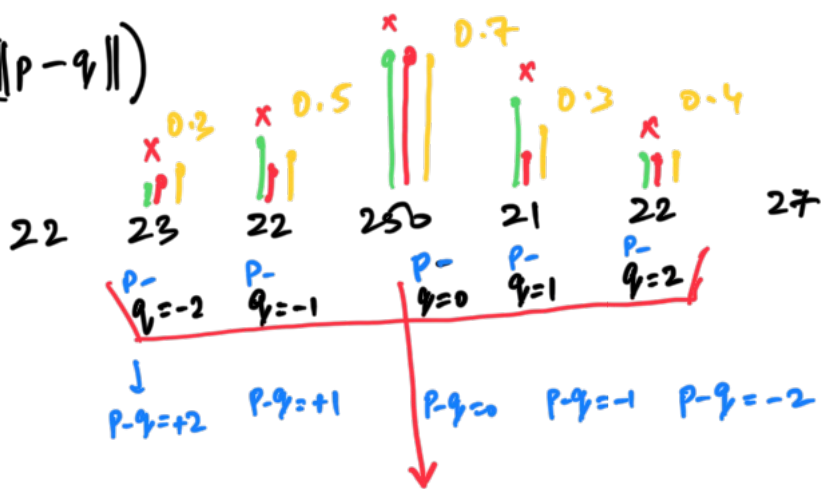
with the normaliser

$$W_{\mathbf{p}} = \sum_{\mathbf{q} \in S} G_{\sigma_s}(\|\mathbf{p} - \mathbf{q}\|) G_{\sigma_r}(\|I(\mathbf{p}) - I(\mathbf{q})\|)$$

$W_{\mathbf{p}}$ depends on $\mathbf{p}$. It cannot be folded into the kernel as a constant, the way it can for a Gaussian, because the weights differ at every pixel.



$$G_s(\|p-q\|)$$

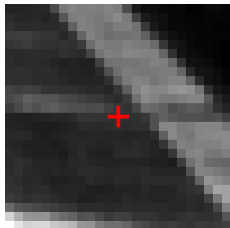


$$\|p-q\|=2 \quad \|p-q\|=1 \quad \|p-q\|=0 \quad \|p-q\|=1 \quad \|p-q\|=2$$

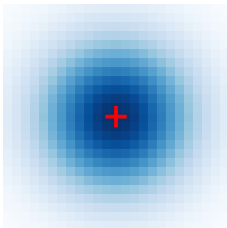
$$G = \frac{1}{\sqrt{2\pi}} e^{-\frac{x^2}{2s^2}}$$

The weights at one pixel

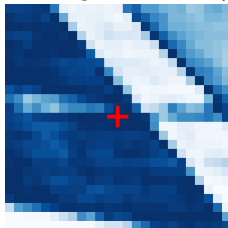
the window



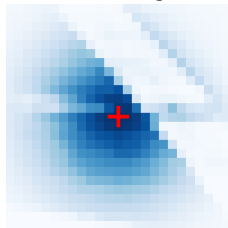
spatial, $\sigma_s = 6$



range, $\sigma_r = 20$



product: the weights used



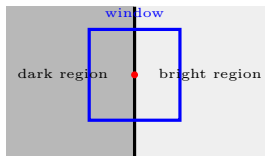
A window straddling a lane marking, centred on the red cross. Darker means a larger weight. The spatial term knows only distance; the range term knows only intensity, and it switches off the bright paint. Their product is the kernel actually applied **at this pixel alone**: move one pixel across the edge and it changes shape.

The two parameters

- ▶ σ_s : the **spatial** scale. How far to look. Behaves like the σ of a Gaussian blur.
- ▶ σ_r : the **range** scale. How different two intensities may be before they stop influencing each other.

Two limits worth knowing:

- ▶ $\sigma_r \rightarrow \infty$: the range weight $\rightarrow 1$ everywhere and we recover the **ordinary Gaussian filter**.
- ▶ $\sigma_r \rightarrow 0$: nothing but the pixel itself contributes; the image is unchanged.



spatial weight: symmetric range weight: kills the far side product: one-sided

What σ_r does

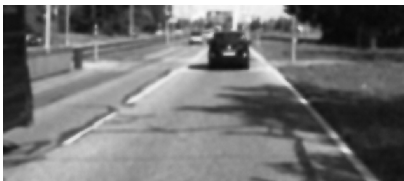
noisy input



bilateral, $\sigma_r = 15$



bilateral, $\sigma_r = 150$



Gaussian, $\sigma = 2$



Same window, same σ_s . At $\sigma_r = 15$ the noise goes and the lane markings stay sharp. At $\sigma_r = 150$ every neighbour counts again, and the result is the Gaussian beside it: mean $|I_x|$ of 18.9 against 18.6.

A worked example

10, 12, 11, 10, **60**, 62, 61, 60

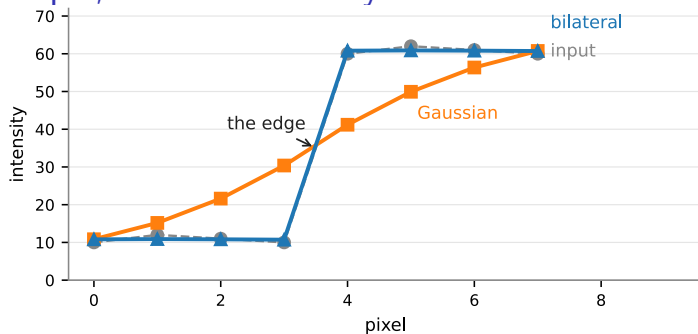
Filter the pixel with value 60, window ± 3 , $\sigma_r = 5$.

- ▶ Neighbours 62, 61, 60: within a few levels, range weight ≈ 1 . **They contribute.**
- ▶ Neighbours 10, 11, 12: about 50 levels away, i.e. $10\sigma_r$. Range weight $\approx e^{-50^2/(2 \cdot 5^2)}$, effectively zero. **They are suppressed.**

The output stays near 60: the step survives. A box or Gaussian filter over the same window would drag it toward the middle and smear the edge.

This is exercise w05a.

The worked example, filtered both ways



Same window, same $\sigma_s = 2$; the only difference is the range weight. The step is 50 levels in the input. The bilateral filter returns 50.1 of it, the Gaussian 10.8: it has spread the edge across six pixels.

Why it is not a convolution

Convolution is **linear** and **shift-invariant**: one fixed kernel, applied identically everywhere, independent of the image. The bilateral filter's weights depend on the intensities in the neighbourhood, so the effective kernel is **different at every pixel** and depends on the input. Consequences:

- ▶ **Not separable.** The $O(k^2) \rightarrow O(k)$ trick from Week 3 does not apply.
- ▶ **No FFT acceleration.**
- ▶ **No frequency-domain interpretation.**
- ▶ Naive implementation is genuinely slow; fast approximations exist and are an active topic.

Where it is used

- ▶ **Denoising** while keeping edges, the original motivation.
- ▶ **Detail enhancement:** bilateral-filter the image, subtract to get a detail layer, then amplify and add back.
- ▶ **Tone mapping** in HDR photography.
- ▶ **Joint (cross) bilateral filtering:** take the range weights from a *different* image, e.g. smooth a noisy depth map using edges from the colour image.

That last idea is the one that generalises: the range term does not have to come from the image you are filtering.

Week 5 recap

- ▶ Linear smoothing cannot separate noise from edges, because one fixed kernel is applied everywhere.
- ▶ Non-linear filters can. The median is the simplest.
- ▶ The bilateral filter multiplies a spatial Gaussian by a range Gaussian.
- ▶ $\sigma_r \rightarrow \infty$ recovers the Gaussian filter.
- ▶ It is neither linear nor shift-invariant, so it is not a convolution, and it loses separability and the FFT with it.

Midterm 1

Thursday Oct 8, covering **Weeks 1–4**:

- ▶ Pinhole camera, perspective projection, what projection destroys.
- ▶ Homogeneous coordinates, $\mathbf{K}$, $\mathbf{P} = \mathbf{K}[\mathbf{R}|\mathbf{t}]$, vanishing points.
- ▶ Convolution and correlation, boundary handling, Gaussians, separability, cost.
- ▶ Derivatives, Sobel, gradient magnitude and orientation, Canny, pyramids.

Today's bilateral filtering is **not** examinable on Midterm 1. The in-class exercises are the best guide to the style of question. Work w01a through w04c again without looking at your answers.

This week's work

- ▶ **Reading:** Szeliski Sec. 3.3.1, 3.3.2.
- ▶ **Exercise:** w05a.
- ▶ **No lab.** Use the time to prepare.

After the break: we stop processing images and start **fitting models** to what we have measured: least squares, and why it fails.

Copyright and License

©Faisal Z. Qureshi



This work is licensed under a Creative Commons Attribution-NonCommercial 4.0 International License.