in my CV class new topic taught is neighbourhood operations

I will assume **introductory-to-intermediate Computer Vision knowledge**, typical of a university CV course (you likely know pixels, images as matrices, and basic filtering ideas but not the deeper math yet). I'll explain this **step-by-step**, moving **simple** → **medium** → **slightly advanced**, with examples and exam-oriented clarity.

---

1. What are *Neighbourhood Operations*? (Core Idea)

Neighbourhood operations are image processing operations where the **new value of a pixel depends on its neighboring pixels**, not just itself.

- Contrast with **point operations**:
  - Point operation → output pixel depends only on the same pixel
  - Neighbourhood operation → output pixel depends on nearby pixels

Formal idea:

$$g(x,y) = f\big(\text{neighbourhood of } (x,y)\big)$$

---

2. What is a Neighbourhood?

A **neighbourhood** is a small window (mask / kernel) centered on a pixel.

Common neighbourhood sizes:

- **3×3** (most common)
- 5×5
- 7×7

Example: 3×3 neighbourhood

```css
a  b  c
d  e  f   ← e is the center pixel
g  h  i
```

Copy code

3. Why Do We Use Neighbourhood Operations?

They are used to:

- Remove **noise**
- Detect **edges**
- **Smooth** images
- **Sharpen** images
- Extract **features**

4. Types of Neighbourhood Operations (Very Important)

A. Linear Neighbourhood Operations

- Output pixel = **weighted sum** of neighbors
- Use **convolution**

Examples:

- Mean filter
- Gaussian filter
- Laplacian filter
- Sobel filter

B. Non-Linear Neighbourhood Operations

- Output pixel = **non-linear function** of neighbors
- No simple multiplication + addition

Examples:

- Median filter
- Max filter
- Min filter
- Morphological operations

5. Linear Neighbourhood Operation – Example

Mean (Averaging) Filter

Kernel (3×3):

$$\frac{1}{9}\begin{bmatrix}1 & 1 & 1\\1 & 1 & 1\\1 & 1 & 1\end{bmatrix}$$

What it does:

- Replaces pixel value with **average of neighbors**
- Smooths image
- Reduces noise
- Blurs edges

---

6. Non-Linear Neighbourhood Operation – Example

Median Filter

Steps:

1. Take neighborhood values
2. Sort them
3. Pick the **middle value**

Example:

```makefile
Neighbourhood: [12, 15, 16, 18, 255, 14, 17, 19, 13]
Sorted:        [12, 13, 14, 15, 16, 17, 18, 19, 255]
Median = 16
```

Copy code

Why median is important:

- Excellent for **salt-and-pepper noise**
- Preserves edges better than mean filter

---

7. Edge Detection as Neighbourhood Operation

Edges = sudden change in intensity

Detected using **derivative-based neighbourhood masks**

Example: Sobel Operator

Horizontal Sobel:

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

Vertical Sobel:

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

8. Boundary Problem (Exam Favorite)

When neighbourhood goes **outside image border**, what do we do?

Common solutions:

- Zero padding
- Replication
- Reflection
- Ignoring boundary pixels

9. Summary (Exam-Ready)

- Neighbourhood operations use **neighboring pixels**
- Depend on **mask / kernel**
- Two types:
    - Linear (mean, Gaussian, Sobel)
    - Non-linear (median, min, max)
- Used for:
    - Smoothing
    - Noise removal
    - Edge detection
- Boundary handling is important

10. One-Line Definition (Perfect for Exams)

> *Neighbourhood operations are image processing techniques in which the output pixel value is computed using the intensity values of its neighboring pixels within a defined window.*