Spatial Filtering and Convolution

Machine Vision

Dr. Mostapha Kalami Heris

✉️ m.k.heris@shu.ac.uk

School of Engineering and Built Environment

Sheffield Hallam University

Learning Outcomes

By the end of this lecture, you should be able to:

- Explain the idea of **spatial filtering** as a neighborhood-based image operation.
- Differentiate **correlation** from **convolution** and describe kernel flipping.
- Predict qualitative effects of common linear kernels (identity, blur, sharpen, Sobel).
- Apply common **MATLAB filtering functions** (`imfilter`, `fspecial`, etc.).

Roadmap for Today

- Motivation & Fundamentals
- Linear Filtering (Correlation & Convolution)
- Linear Kernel Examples
- Summary & Discussion + Teaser

Motivation: Why Do We Filter Images?

Sheffield Hallam University

Opening Questions

These are some questions for you. Think about them for a moment.

Q1: Suppose we have an image corrupted by **random noise**. How can we remove it?

Q2: If we want to detect **edges** or **boundaries** of objects, what property of pixels should we examine?

Q3: If we want to make an image **smoother**, what operation might help?

🗣️ **Let's discuss.**

What do these tasks have in common?

What kind of information might we need from the image?

Discussion Summary

Q1. Noise removal:

Noise behaves like **outliers**.

So we need to **identify and replace abnormal pixels**.

Q2. Edge detection:

Edges correspond to large intensity **differences** between neighboring pixels.

So we need to **compare pixels in a local region**.

Q3. Smoothing:

Smooth images have less local variation.

In order to reduce local variations, we can compute **averages** over a neighborhood.

Summary of Common Themes

We discussed three common image processing tasks:

- **Noise removal**, which involves identifying and correcting outlier pixels.
- **Edge detection**, which focuses on finding significant intensity changes.
- **Smoothing**, which aims to reduce local variations by averaging pixel values.

All these problems rely on **information from neighboring pixels**.

They can all be expressed as **mathematical operations** over a local region.

We call these operations **spatial filters**.

From Questions to Mathematics

Every operation above can be written as a **mathematical function** acting on a local window:

$$g(x, y) = f(\text{pixels in a neighborhood around } (x, y))$$

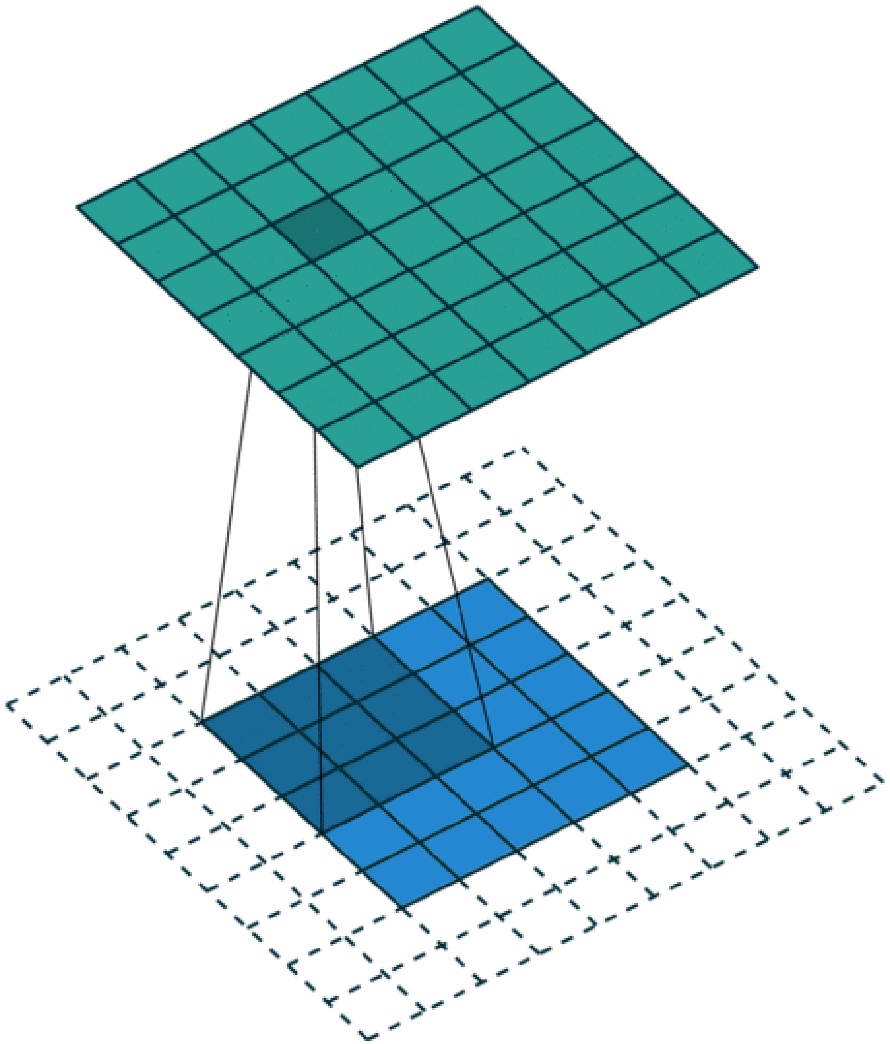
Where:

- $g(x, y)$ is the output pixel value at (x, y).
- f is some function that combines the values of pixels in a local neighborhood around (x, y).

> The specific form of f determines the type of filtering (e.g., averaging, edge detection).

Sheffield Hallam University

How is this done in practice?



- We define a **small window** (e.g., 3x3, 5x5) that moves across the image.
- For each pixel, we apply a **function** to the pixels in the window to compute a new value.
- This process is repeated for every pixel in the image.
- The window is often called a **kernel** or **mask**.
- The function can be **linear** (e.g., weighted sum) or **nonlinear** (e.g., median).

Sheffield Hallam University

Linear Filtering

Sheffield Hallam University

Types of Linear Filters

Linear filters can be categorized into two main types based on the mathematical operation used:

1. **Correlation-based filters** (denoted by $\star$)

2. **Convolution-based filters** (denoted by $*$)

Both types involve applying a kernel to the image, but they differ in how the kernel is applied.

> **Main Difference:**
>
> In convolution, the kernel is flipped before application, while in correlation, it is used as is.

Correlation-based Filtering

147	163	169	122	51
152	168	148	71	48
176	167	97	44	57
185	121	45	50	63
132	55	43	61	67

Pixels of the Source Image with their numerical values.

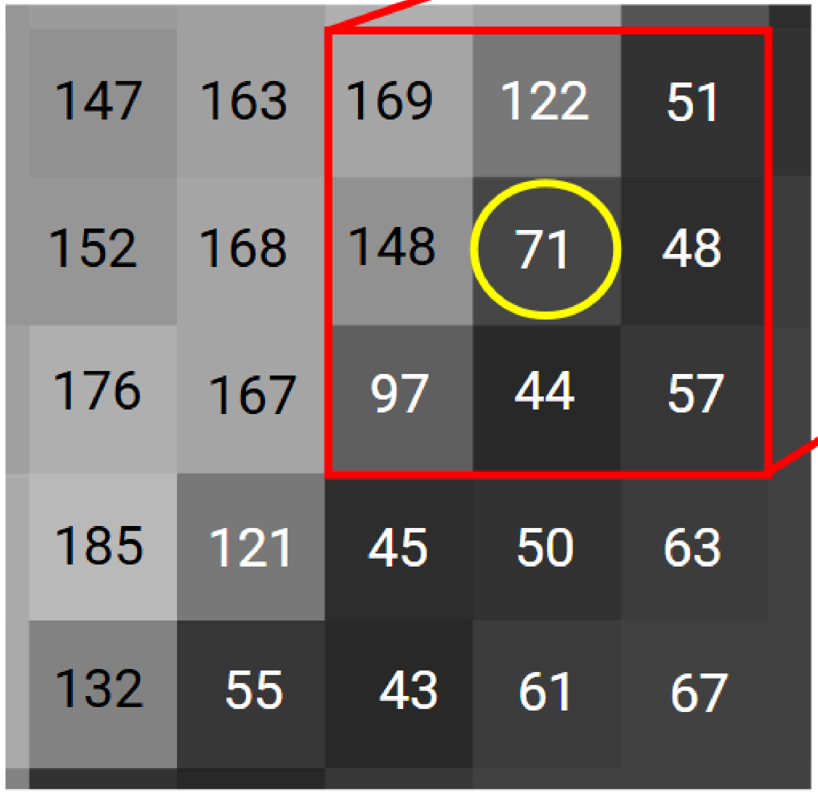
1	2	1
0	0	0
−1	−2	−1

A 3x3 Kernel (Mask) with its weights.

Do you have any idea what would be the value of the output pixel at the center?

Sheffield Hallam University

Correlation Operation



1	2	1
0	0	0
−1	−2	−1

We multiply each pixel in the window by the corresponding weight in the kernel.

169	244	51
0	0	0
−97	−88	−57

And then we sum all these products:

$$\text{Output} = 169 + 244 + 51 - 97 - 88 - 57 = 222$$

Sheffield Hallam University

Final Output of Correlation

After applying the correlation operation to the entire image, we get the following output.

5	22	143	254	158
−69	35	218	222	55
−39	164	274	130	−13
190	322	203	10	−40
389	281	36	−54	−32

Sheffield Hallam University

Correlation: Input vs. Output

147	163	169	122	51
152	168	148	71	48
176	167	97	44	57
185	121	45	50	63
132	55	43	61	67

Input Image

5	22	143	254	158
−69	35	218	222	55
−39	164	274	130	−13
190	322	203	10	−40
389	281	36	−54	−32

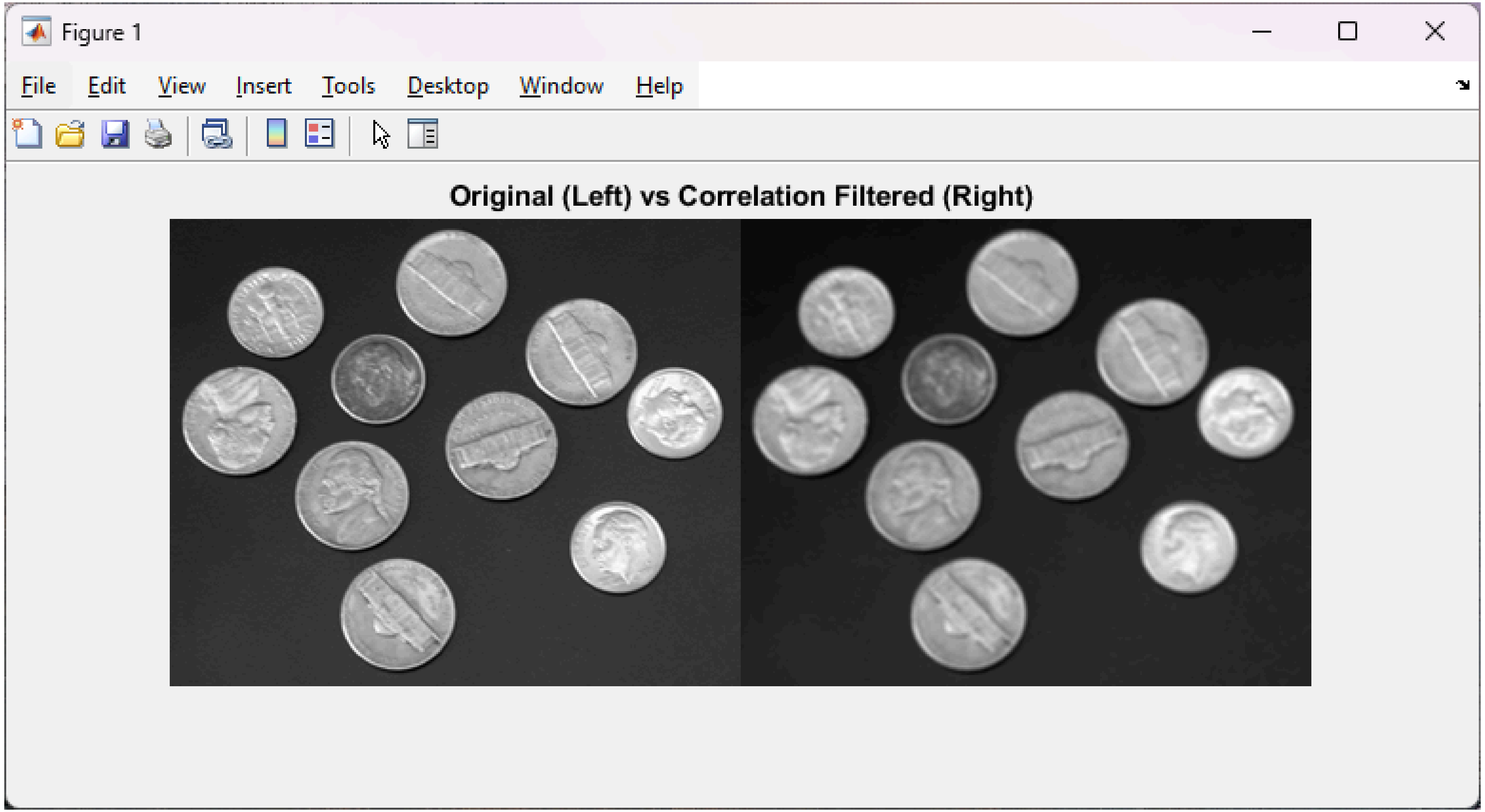
Output Image

MATLAB Example: Correlation Filtering

```matlab
% Read the image
I = imread('coins.png');

% Define a simple averaging kernel (3x3)
h = fspecial('average', [3 3]);

% Apply correlation filtering
J = imfilter(I, h, 'corr', 'replicate');

% Display the original and filtered images
imshowpair(I, J, 'montage');
title('Original (Left) vs Correlation Filtered (Right)');
```

Sheffield Hallam University

Result of the MATLAB Code



Convolution-based Filtering

147	163	169	122	51
152	168	148	71	48
176	167	97	44	57
185	121	45	50	63
132	55	43	61	67

Pixels of the Source Image with their numerical values.

0	−1	0
−1	0	1
0	1	0

A 3x3 Kernel (Mask) with its weights.

Convolution is similar to correlation, but the kernel is flipped both horizontally and vertically before applying it to the image.

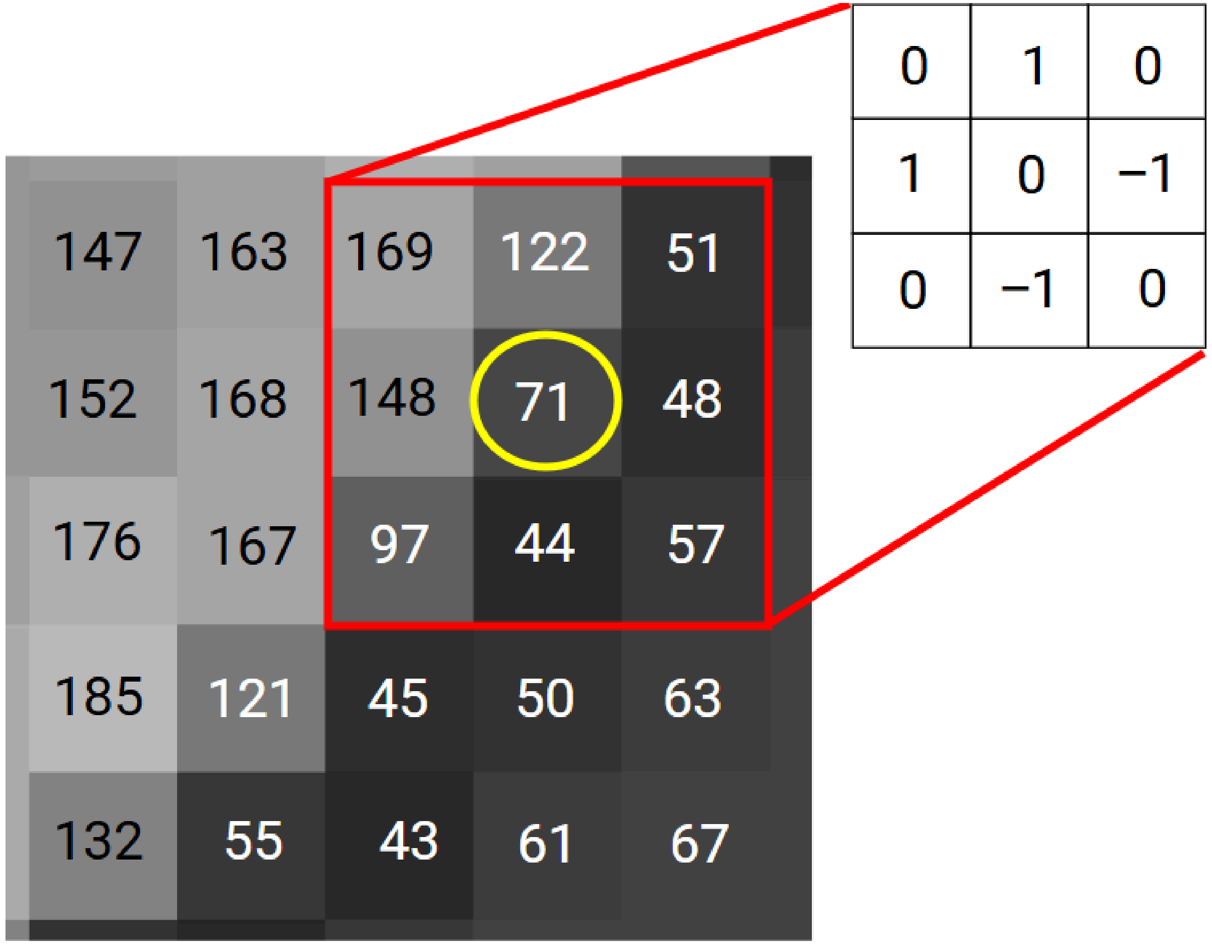
Kernel Flipping (Rotation)

In convolution, the kernel is flipped both horizontally and vertically before applying it to the image.

This is equivalent to rotating the kernel by 180 degrees.

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

Convolution Operation



We multiply each pixel in the window by the corresponding weight in the flipped kernel.

0	122	0
148	0	−48
0	−44	0

And then we sum all these products:

$$\text{Output} = 122 + 148 - 48 - 44 = 178$$

Final Output of Convolution

After applying the convolution operation to the entire image, we get the following output.

−3	−28	69	211	108
−43	0	169	178	3
−39	126	226	61	−36
97	252	125	−35	−26
249	170	−20	−41	−13

Convolution: Input vs. Output

147	163	169	122	51
152	168	148	71	48
176	167	97	44	57
185	121	45	50	63
132	55	43	61	67

Input Image

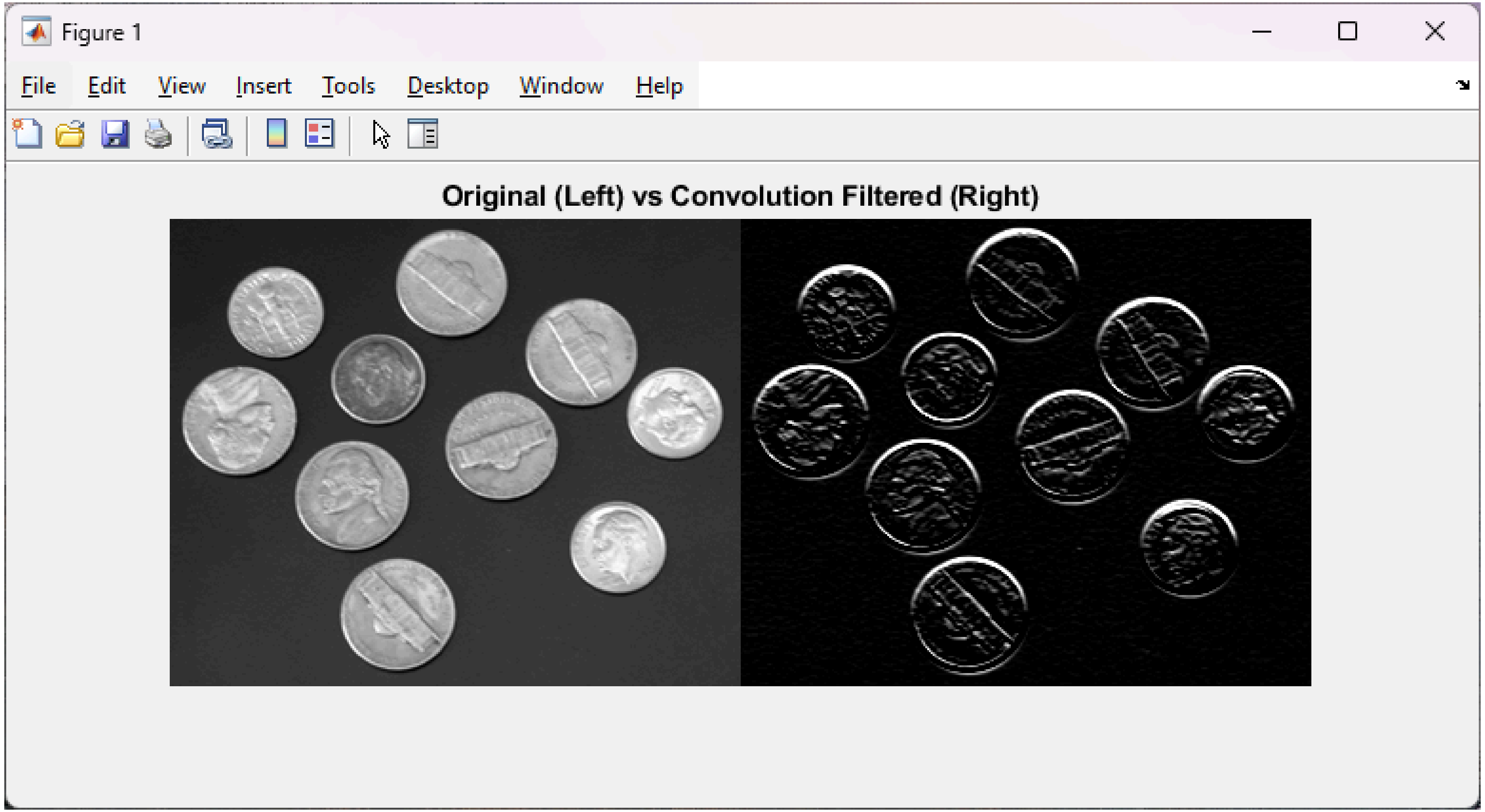
−3	−28	69	211	108
−43	0	169	178	3
−39	126	226	61	−36
97	252	125	−35	−26
249	170	−20	−41	−13

Output Image

MATLAB Example: Convolution Filtering

```matlab
% Read the image
I = imread('coins.png');

% Define a Sobel kernel for edge detection
h = fspecial('sobel');

% Apply convolution filtering
J = imfilter(I, h, 'conv', 'replicate');

% Display the original and filtered images
imshowpair(I, J, 'montage');
title('Original (Left) vs Convolution Filtered (Right)');
```

Result of the MATLAB Code



Group Activity: Kernel Matching Challenge

Group Activity: Kernel Matching

🧩 The Goal

You will work in **small groups** to identify **which image** corresponds to **which kernel**.

Each group will receive:

- **One A4 sheet** with **5 kernel matrices** and **empty boxes** beside them (you will place images there)
- **A set of printed images — 7 in total** (5 real matches + 2 distractors)

> Your task is to **match** each image to the kernel that most likely produced it. and
>
> **Discuss your reasoning** with your group.

Sheffield Hallam University

During the Activity

⏱️ **You have about 5 minutes.**

Work collaboratively and share your reasoning.

💡 Think about:

- Does the image look smoother or sharper?
- Are any directions emphasized (horizontal or vertical)?
- Do bright and dark areas look exaggerated?

🗣️ **Discuss and agree as a group.**

Discussion: What Did You Observe?

Let us hear from a few groups.

Which image matched each kernel?

What patterns or clues did you use?

Were any cases confusing or ambiguous?

Linear Kernel Examples

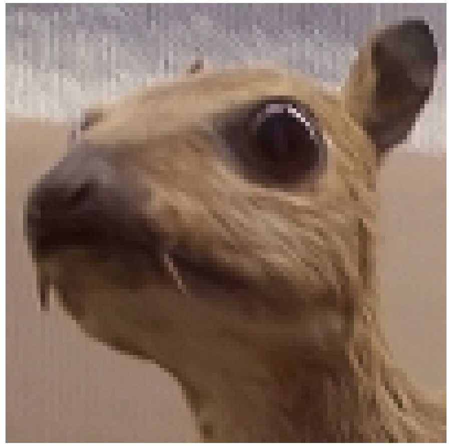
Identity Kernel

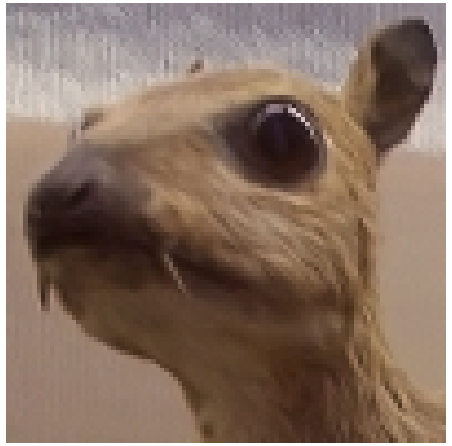
Identity Kernel is defined as:

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

Applying this kernel to an image leaves the image unchanged.

> **Why?** Can you explain this?





Sheffield Hallam University

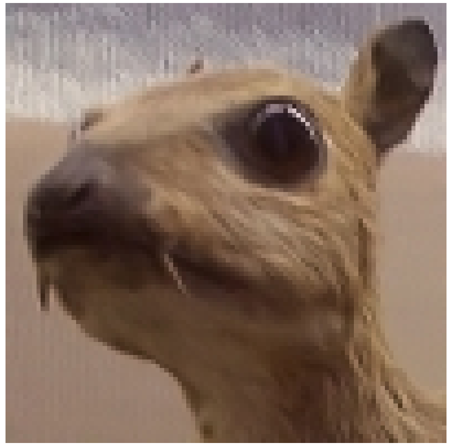
Sharpening Kernel

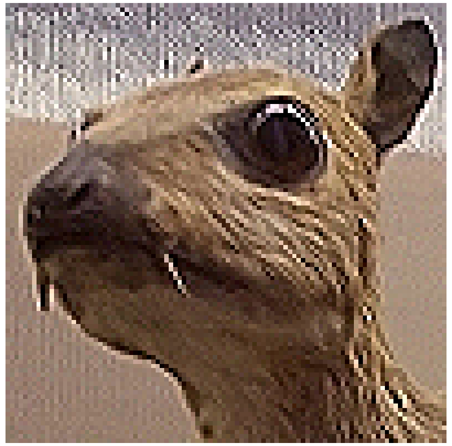
Sharpening Kernel is defined as:

$$\begin{bmatrix} 0 & -1 & 0 \\ -1 & 5 & -1 \\ 0 & -1 & 0 \end{bmatrix}$$

Applying this kernel to an image enhances edges and fine details.

> **Can you explain why this happens?**



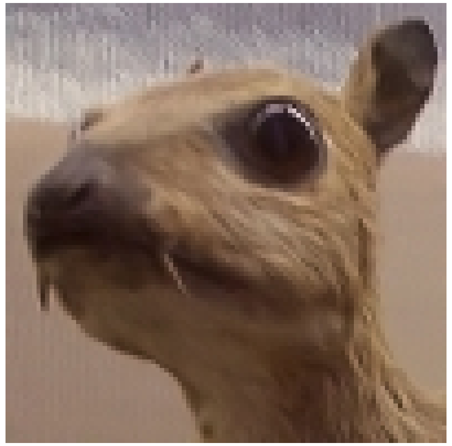


Box Blur Kernel

Box Blur Kernel is defined as:

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

Applying this kernel to an image smooths it by averaging pixel values in a local neighborhood.



> **How can we use this operation in practice?**

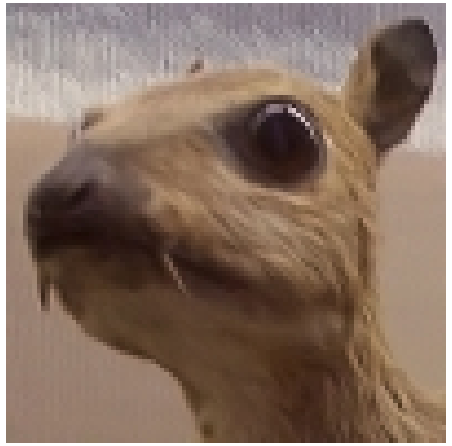
5x5 Box Blur Kernel

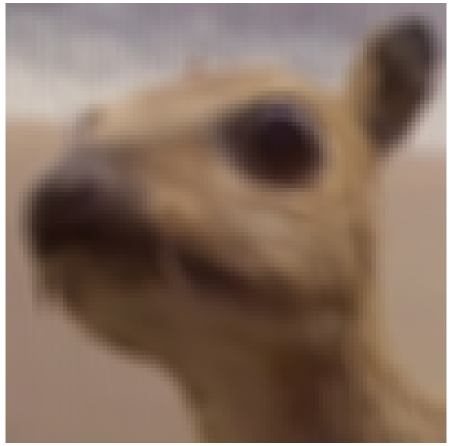
5x5 Box Blur Kernel is defined as:

$$\frac{1}{25}\begin{bmatrix}1 & 1 & 1 & 1 & 1\\1 & 1 & 1 & 1 & 1\\1 & 1 & 1 & 1 & 1\\1 & 1 & 1 & 1 & 1\\1 & 1 & 1 & 1 & 1\end{bmatrix}$$

Applying this kernel to an image results in a stronger smoothing effect compared to the 3x3 box blur.

> **What might be a downside of using this?**





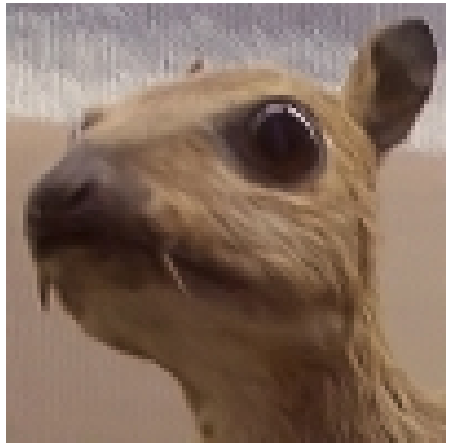
Sheffield Hallam University

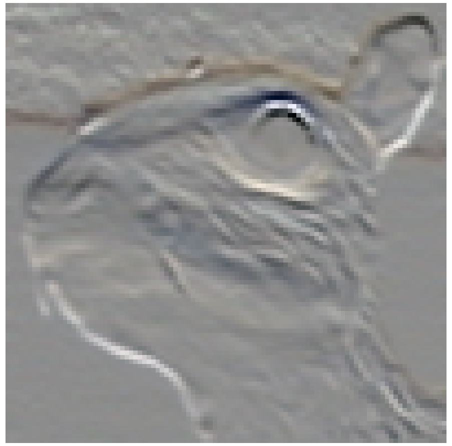
Horizontal Sobel Kernel

Horizontal Sobel Kernel (aka. Sobel-X) is defined as:

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

Applying this kernel to an image emphasizes horizontal edges.





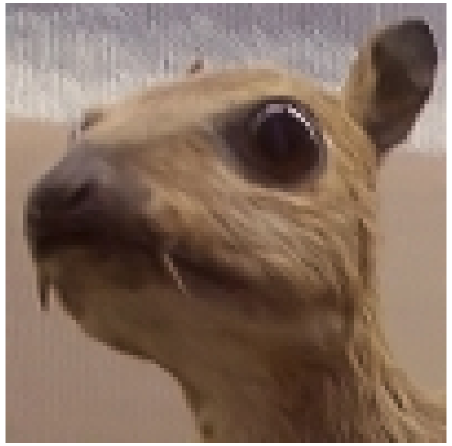
> **How does this kernel work?**
>
> What about vertical edges?

Vertical Sobel Kernel

Vertical Sobel Kernel (aka. Sobel-Y) is defined as:

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

Applying this kernel to an image emphasizes vertical edges.

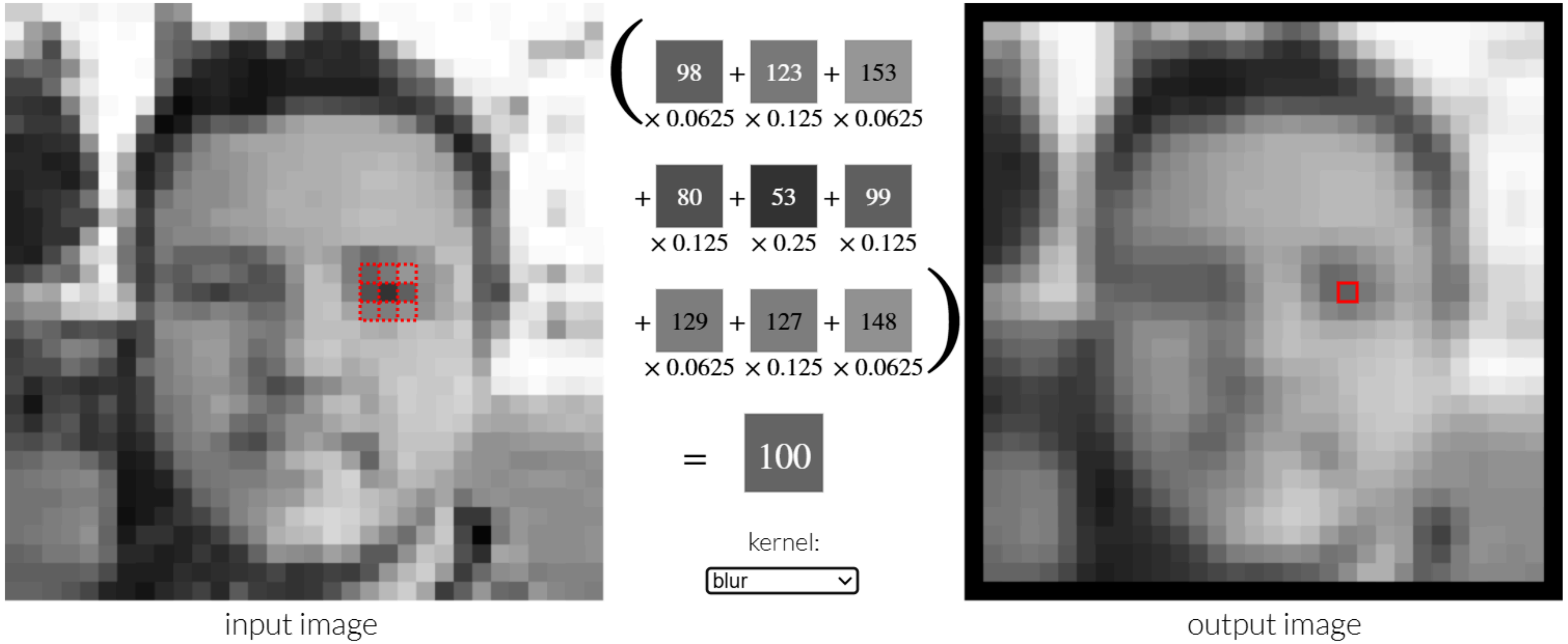


> **How does this kernel work?**
>
> What about edges in other directions?

Image Kernels Explained Visually

Following online tool, developed by **Victor Powell**, allows you to visualize different kernels on images.

🔗 Image Kernels Explained Visually



Summary & References

Sheffield Hallam University

Summary of Key Points

- **Spatial filtering** involves applying a function to a local neighborhood of pixels.
- **Linear filters** can be correlation-based or convolution-based, differing in kernel application.
- Common linear filters include identity, sharpening, box blur, and Sobel filters.
- MATLAB provides built-in functions for both correlation- and convolution-based filtering.
- Understanding these concepts is crucial for effective image processing and analysis.

Mini-Challenge

Let's discuss.

How would the 3x3 box blur handle a single bright pixel in a dark area?

What might happen if one pixel is an extreme outlier?

What do you expect at the image edges?

Think About Upcoming Topics

These are questions to ponder for next time.

Q1: Can we extend the concept of linear filtering to non-linear operations?

Q2: What about the boundary pixels? How should we handle them during filtering?

Q3: How do we choose the right kernel for a specific image processing task?

References

- Gonzalez, R. C., & Woods, R. E. (2018). *Digital Image Processing* (4th ed.). Pearson.
- Gonzalez, R. C., Woods, R. E., & Eddins, S. L. (2020). *Digital Image Processing Using MATLAB* (3rd ed.). Gatesmark Publishing.
- Forsyth, D. A., & Ponce, J. (2011). *Computer Vision: A Modern Approach* (2nd ed.). Pearson.
- Szeliski, R. (2010). *Computer Vision: Algorithms and Applications*. Springer.
- MathWorks. (n.d.). *Image Processing Toolbox – MATLAB*. Retrieved from https://mathworks.com/help/images/index.html

Exit Ticket

Scan the QR code or visit the link below.

Before you leave, please complete the online form and provide your feedback.

1. **One clear concept** from today
2. **One confusing concept**
3. **One thing to try in MATLAB** this week

forms.office.com/e/YvWnAr66yJ

Sheffield Hallam University

Questions & Support

- You are encouraged to **ask questions anytime** 💡
  - During the lecture
  - In lab sessions
  - By email → m.k.heris@shu.ac.uk
- **No question is too simple** — asking questions:
  - Helps you learn faster
  - Builds confidence
  - Improves understanding for the whole class

👉 If something is unclear, **just ask!**