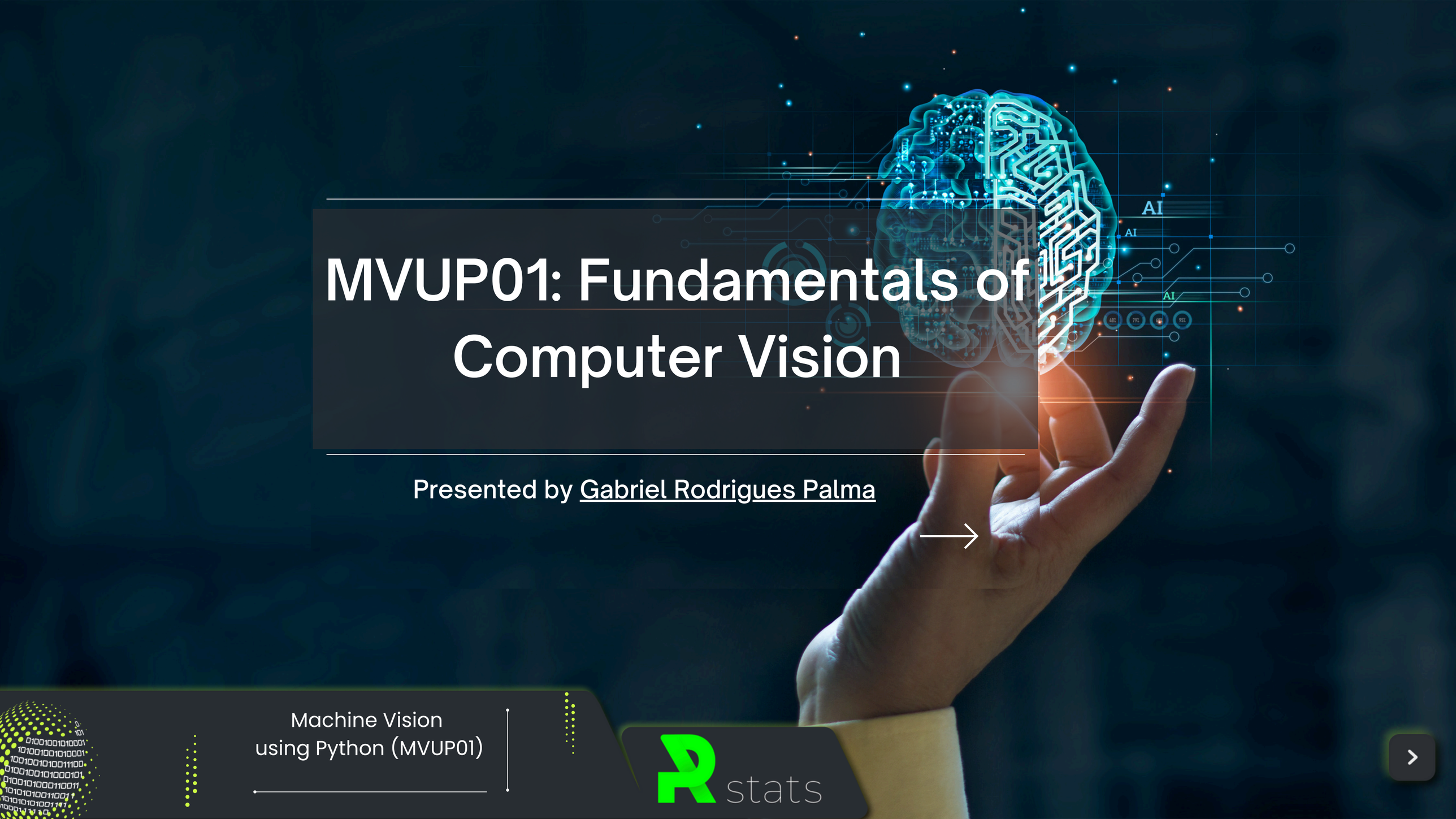
MVUP01: Fundamentals of Computer Vision

Presented by Gabriel Rodrigues Palma

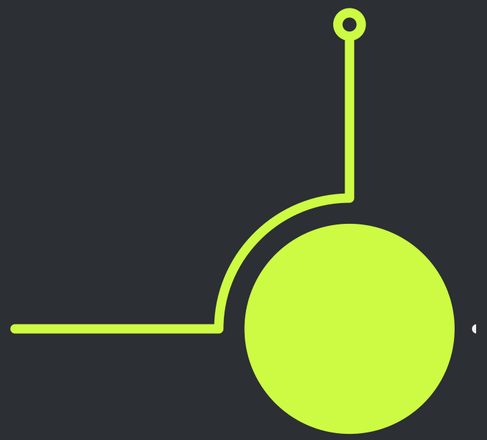


Machine Vision
using Python (MVUP01)

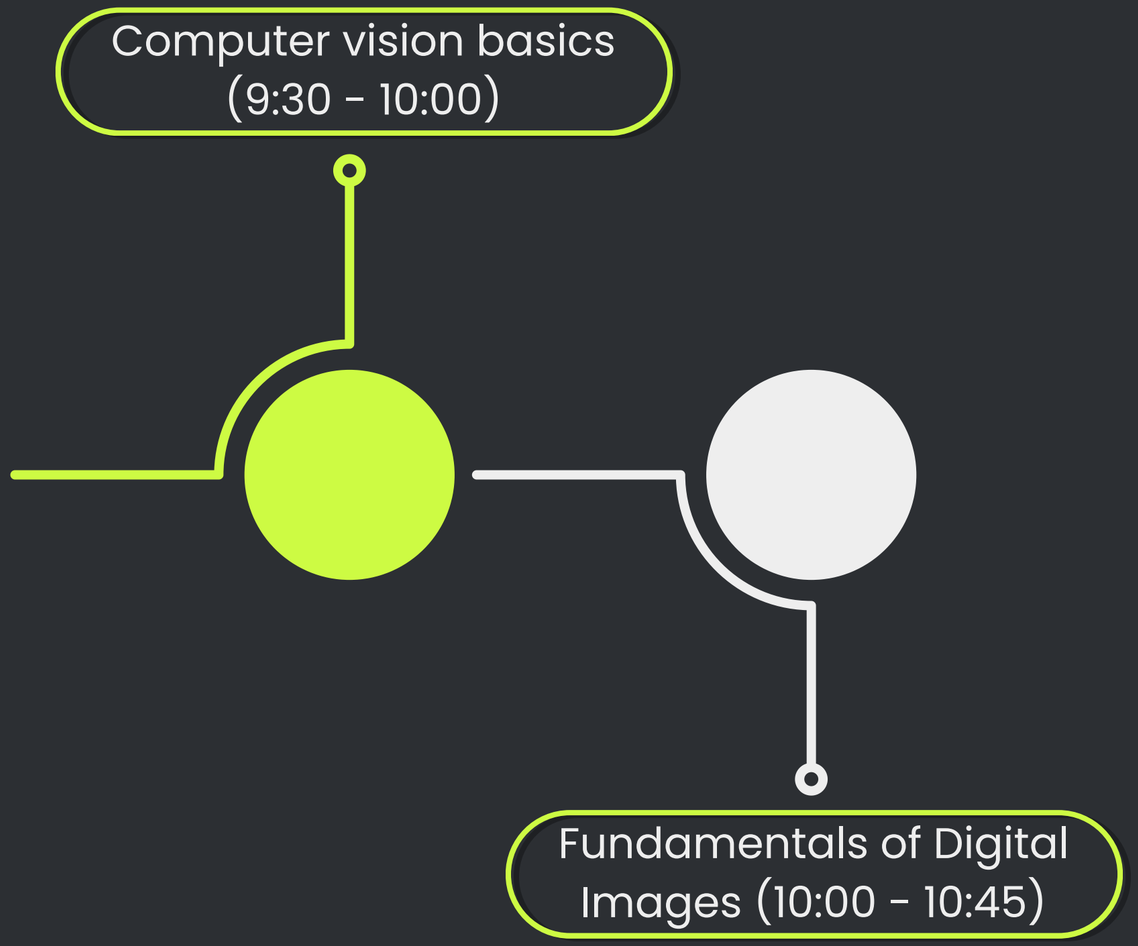
stats

Morning Session (9:30 - 12:00)

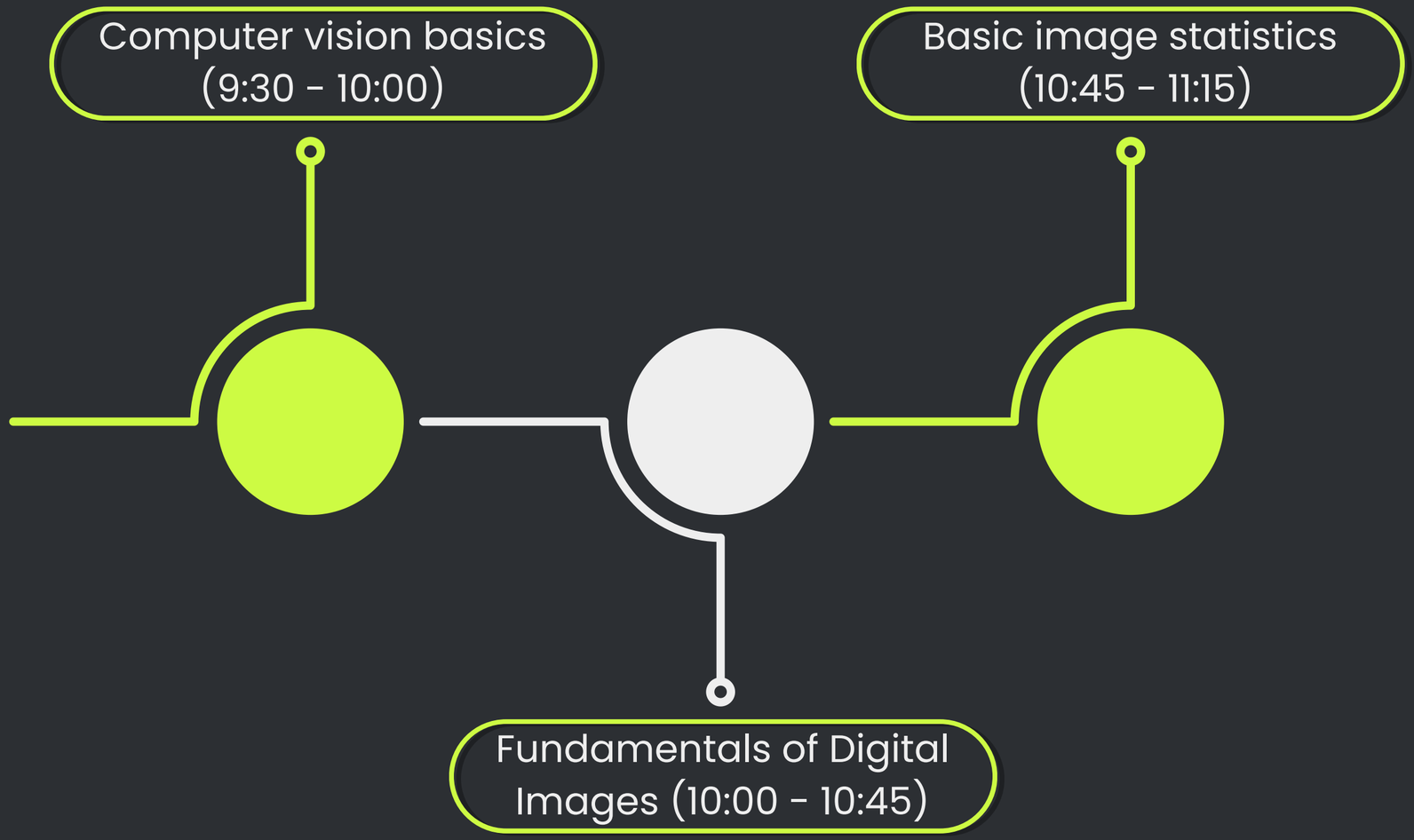
Computer vision basics
(9:30 - 10:00)



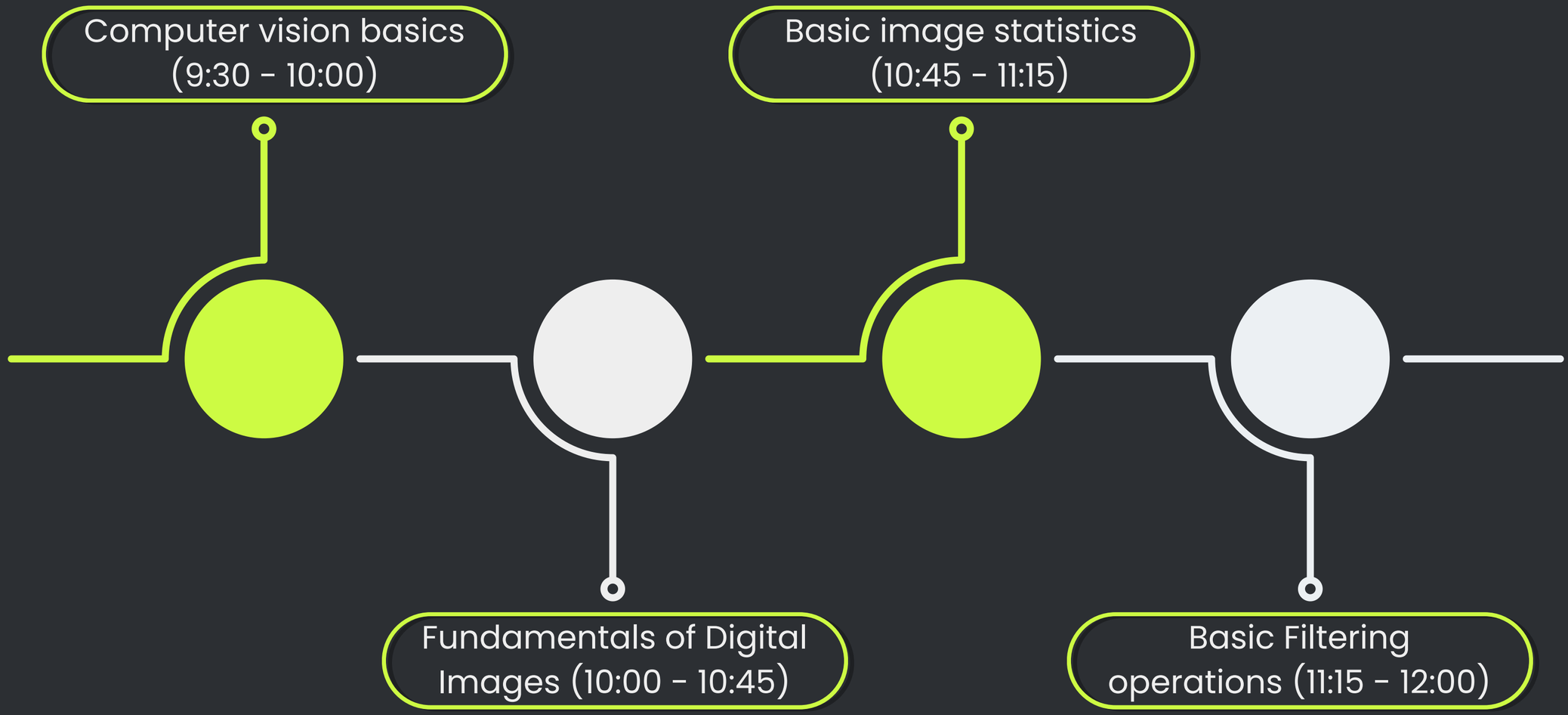
Morning Session (9:30 - 12:00)



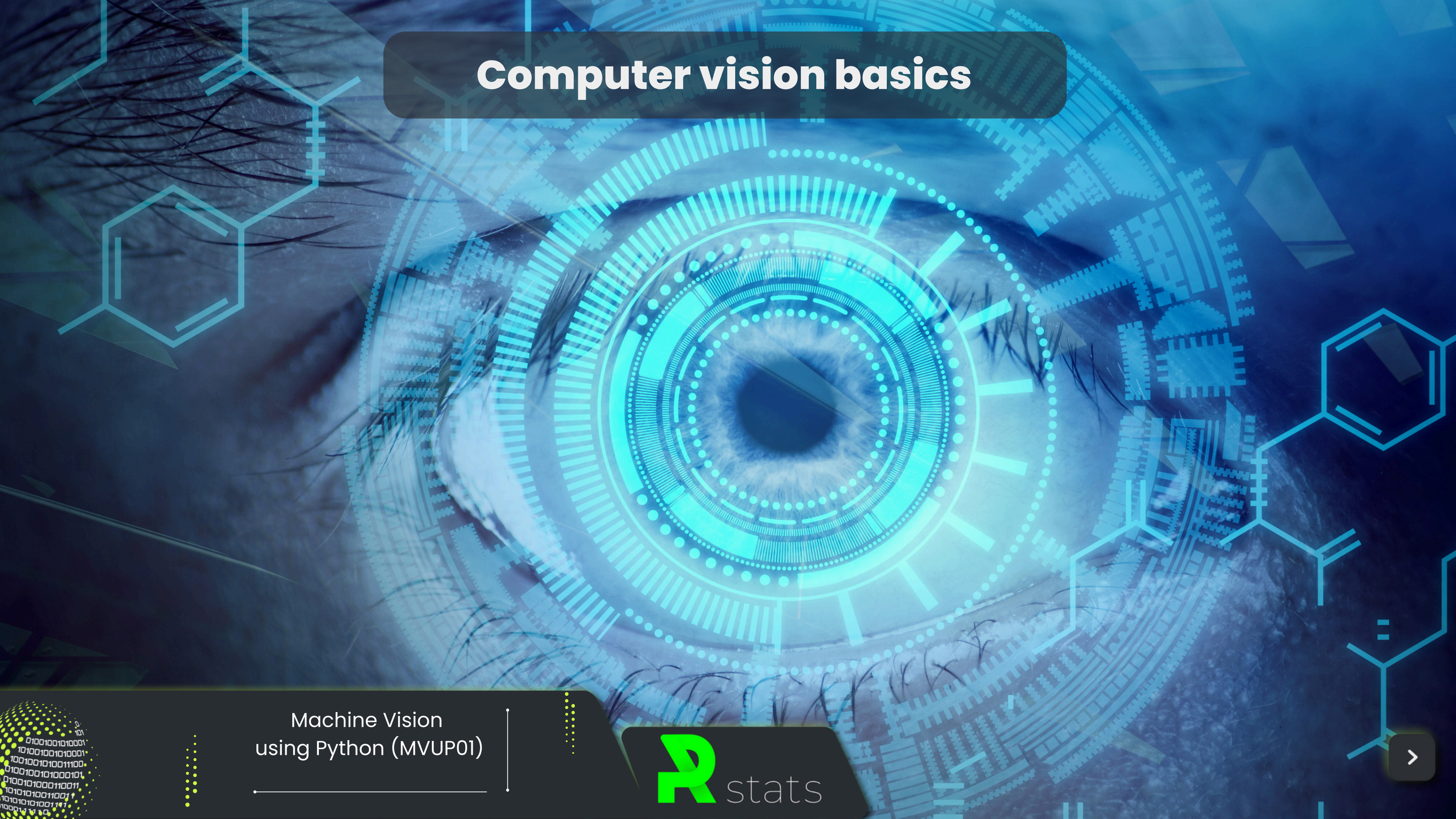
Morning Session (9:30 - 12:00)



Morning Session (9:30 - 12:00)



Computer vision basics



Machine Vision
using Python (MVUP01)

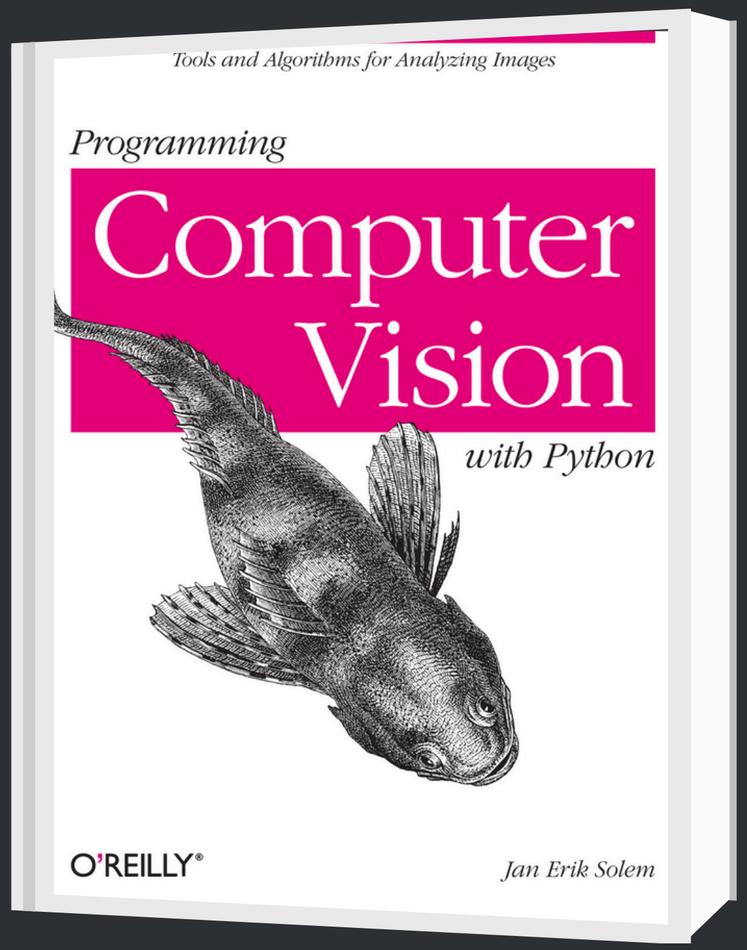
stats

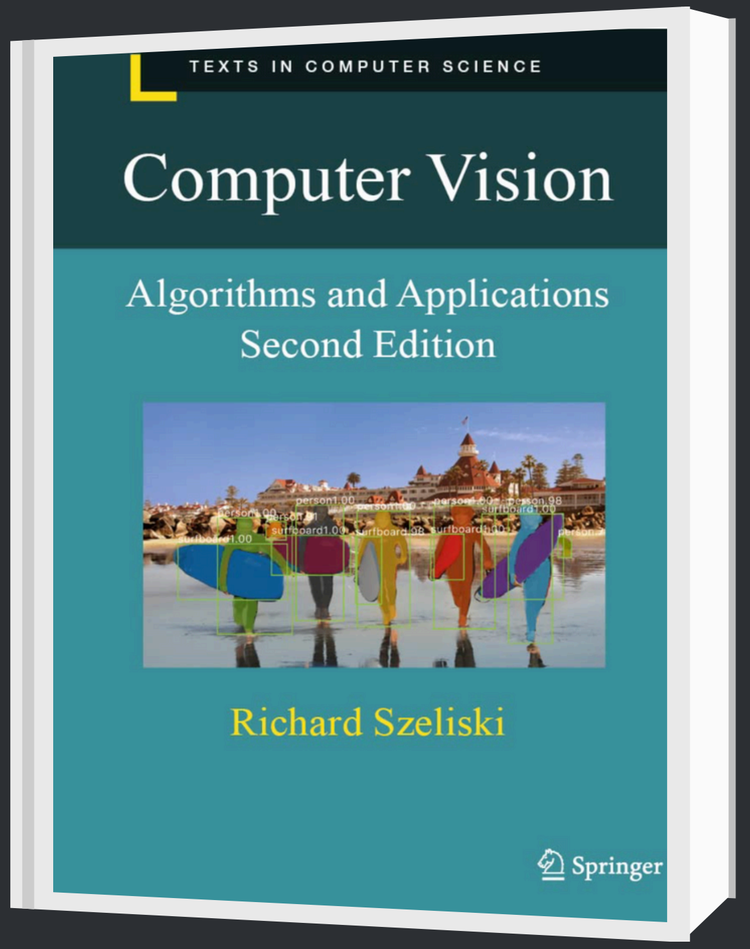
Computer vision basics

Machine Vision
using Python (MVUP01)

stats

Computer vision basics





Computer vision basics

“In computer vision, we are trying to describe the world that we see in one or more images and to reconstruct its properties, such as shape, illumination, and colour distributions.” (Szeliski at al., 2022)

Computer vision basics

Machine vision extends the focus to the integration and automation of the techniques, converting an image into useful information that can be obtained using a standard workflow

Computer vision basics

"In computer vision, we are trying to describe the world that we see in one or more images and to reconstruct its properties, such as shape, illumination, and colour distributions." (Szeliski at al., 2022)

Computer vision is being used today in a wide variety of real-world applications, which include

Computer vision basics

Machine Vision
using Python (MVUP01)

stats

Computer vision basics

Machine Vision
using Python (MVUP01)

stats

Computer vision basics

Machine Vision
using Python (MVUP01)

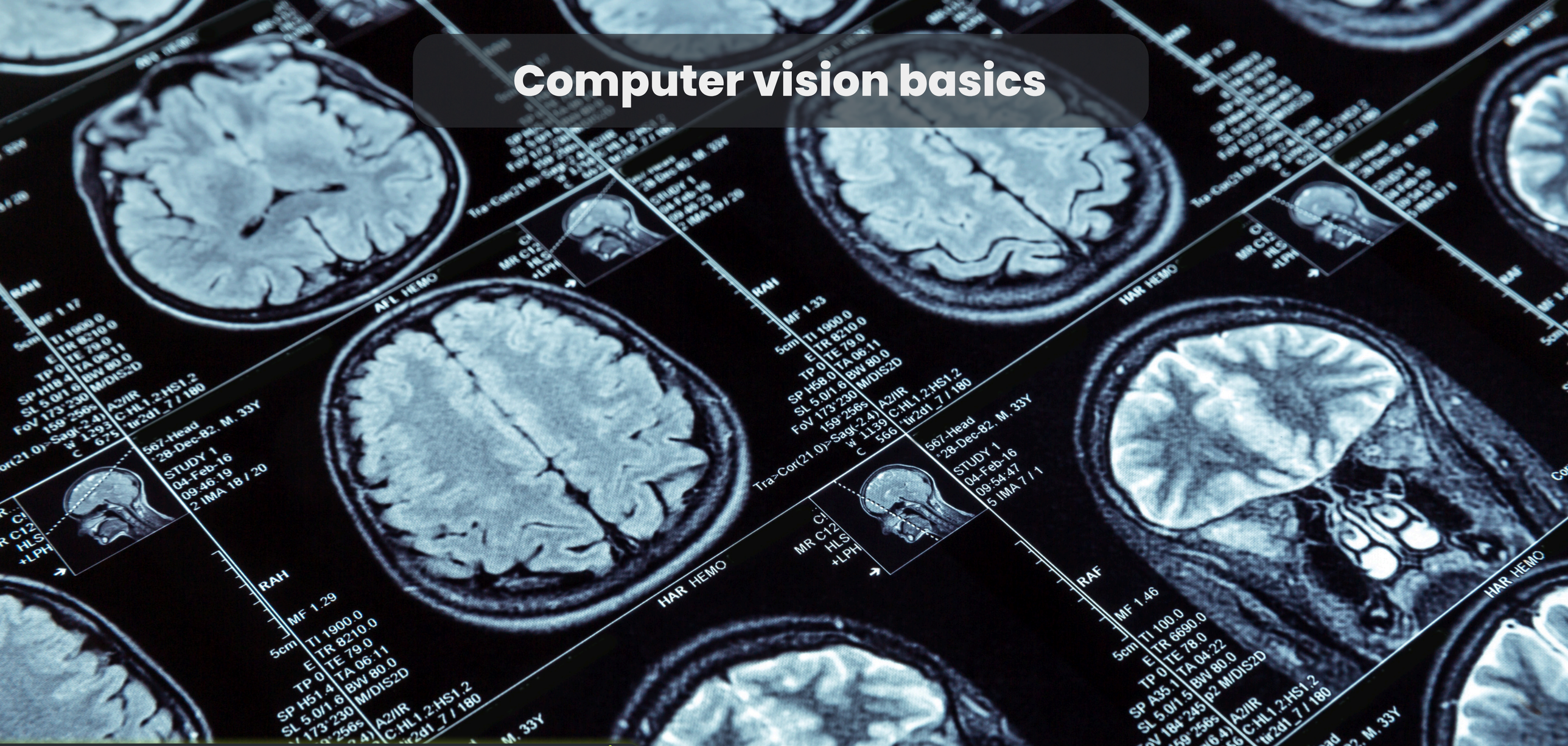
stats

Computer vision basics

Machine Vision
using Python (MVUP01)

R stats

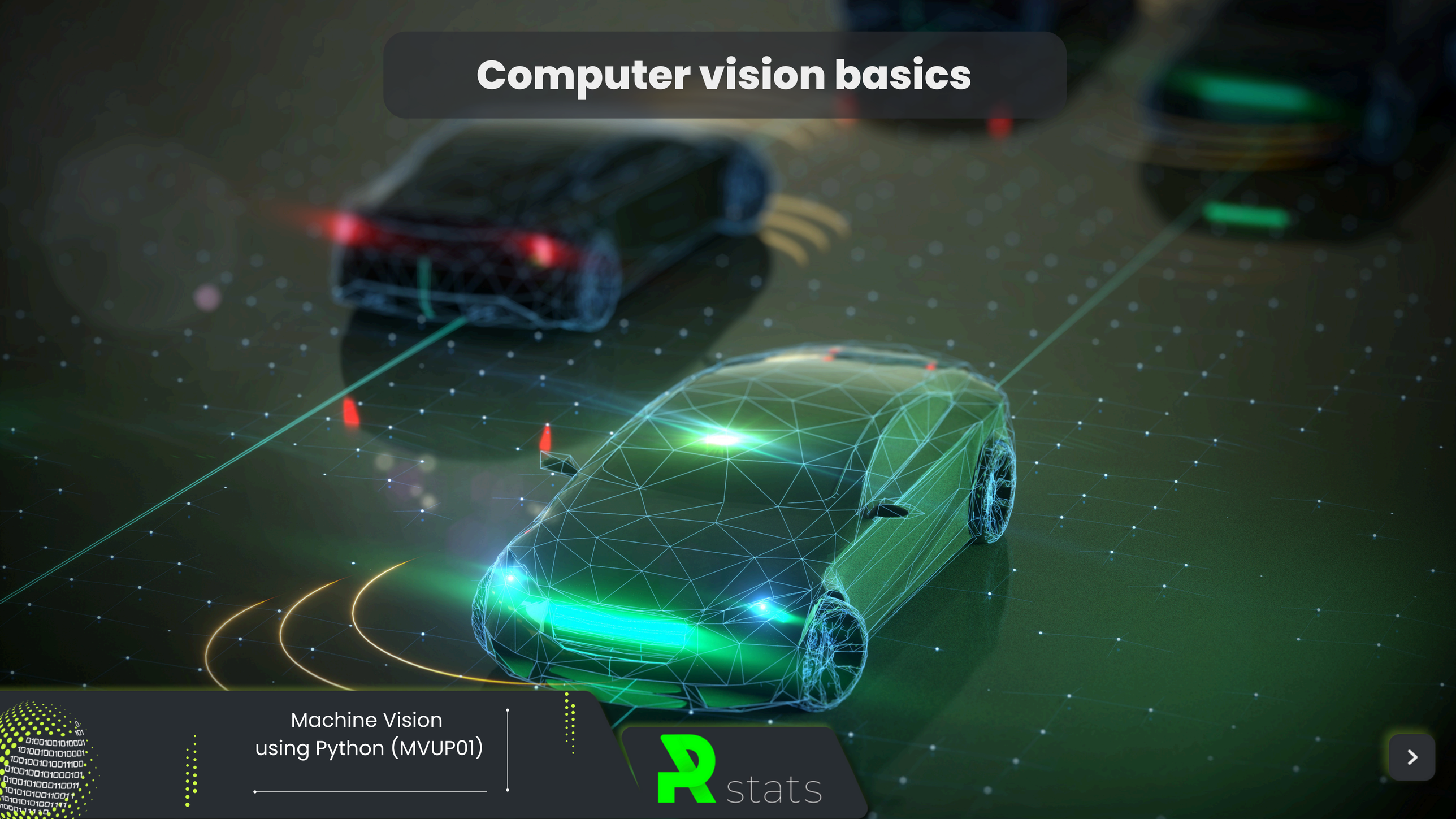
Computer vision basics



Machine Vision
using Python (MVUP01)

stats

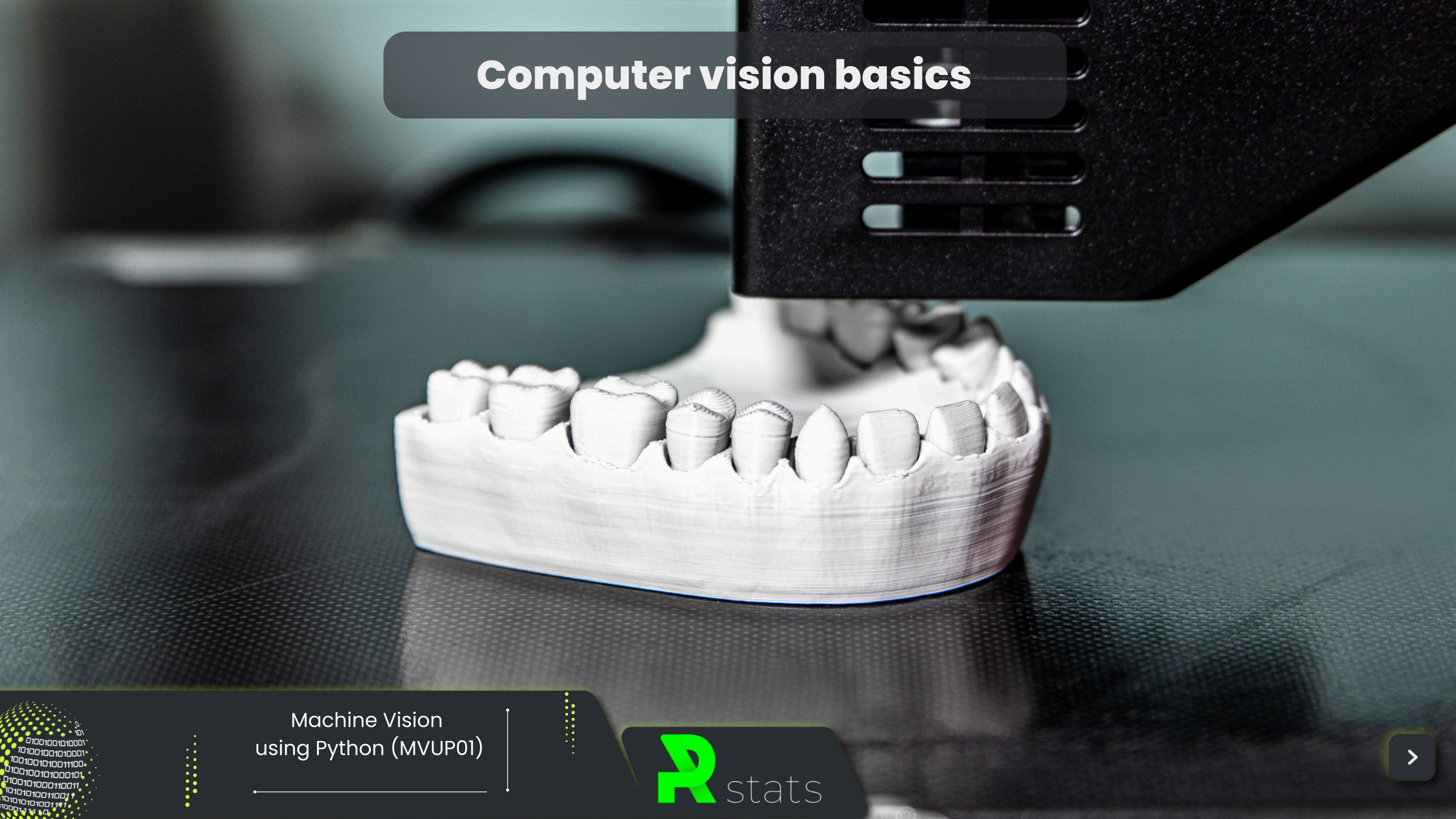
Computer vision basics



Machine Vision
using Python (MVUP01)

Rstats

Computer vision basics



Machine Vision
using Python (MVUP01)

stats

Computer vision basics

Machine Vision
using Python (MVUP01)

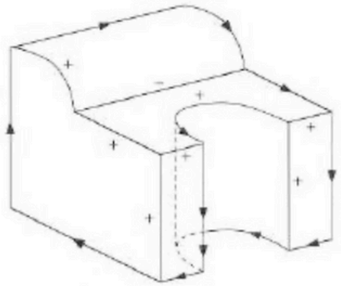
stats

Computer vision basics

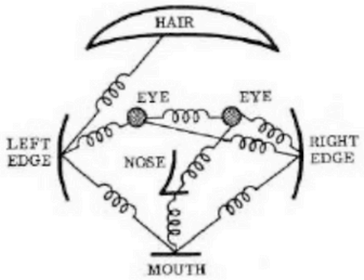
1970s

Foundational concepts in computer vision

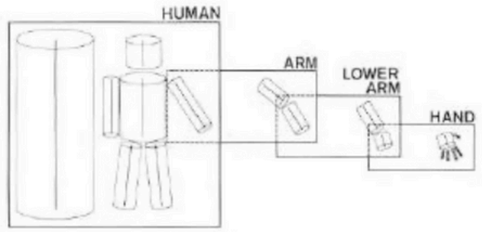
Computer vision basics



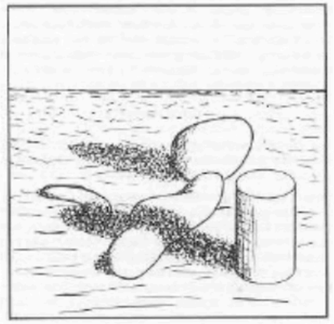
(a)



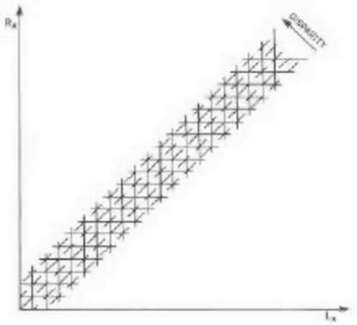
(b)



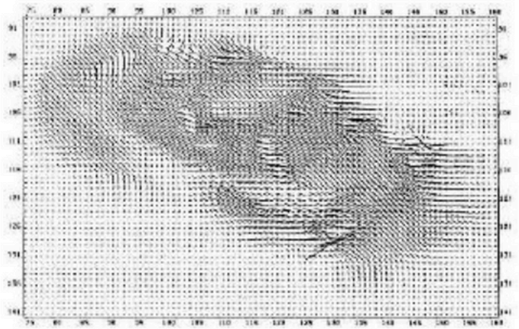
(c)



(d)



(e)



(f)

Computer vision basics

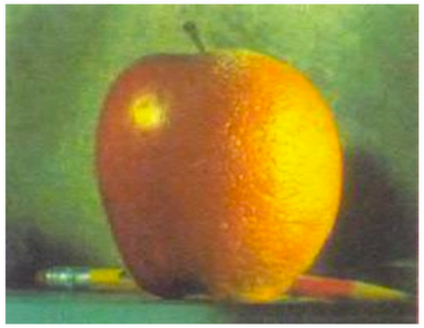
1970s

Foundational concepts in computer vision

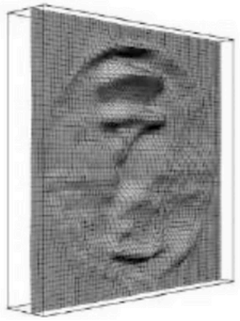
1980s

Advancements in motion and shape analysis

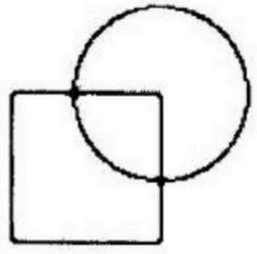
Computer vision basics



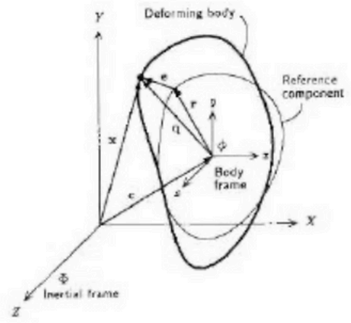
(a)



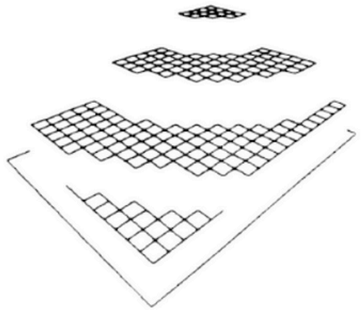
(b)



(c)



(d)



(e)

(f)

Computer vision basics

1970s

Foundational concepts in computer vision

1980s

Advancements in motion and shape analysis

1990s

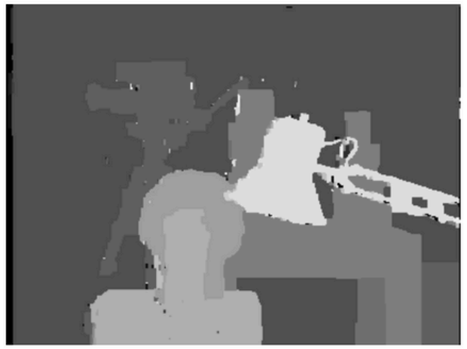
Rise of geometry-based modelling techniques

2000s

Expansion into physics-based vision and applications

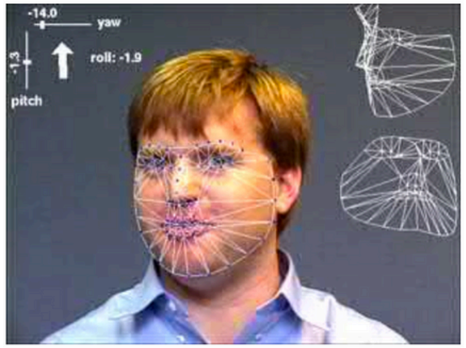
Computer vision basics

(a)

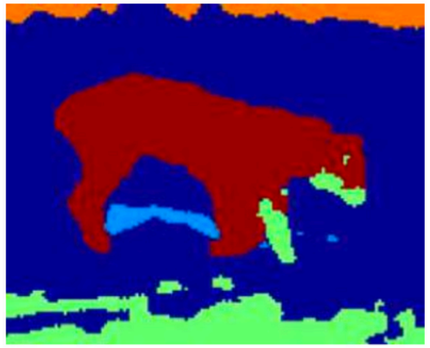


(b)

(c)



(d)



(e)

(f)

Computer vision basics

1970s

Foundational concepts in computer vision

1980s

Advancements in motion and shape analysis

1990s

Rise of geometry-based modelling techniques

2000s

Expansion into physics-based vision and applications

2010s

Integration of machine learning and human modelling

Computer vision basics

1970s

Foundational concepts in computer vision

1980s

Advancements in motion and shape analysis

1990s

Rise of geometry-based modelling techniques

2000s

Expansion into physics-based vision and applications

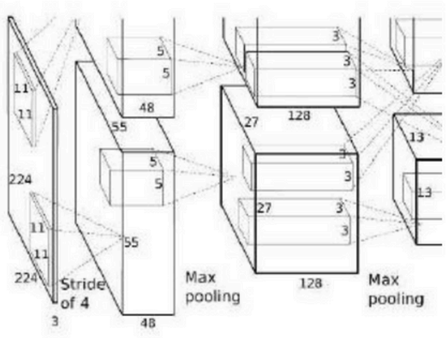
2010s

Integration of machine learning and human modelling

2020

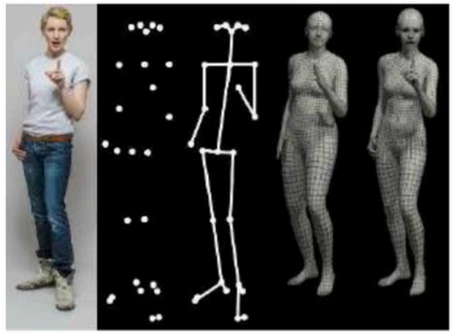
Culmination in deep learning and vision-language integration

Computer vision basics

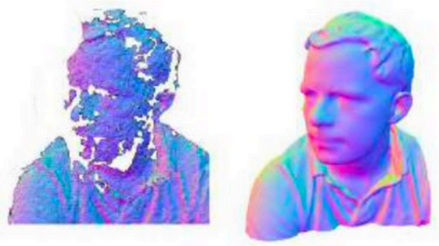


(a)

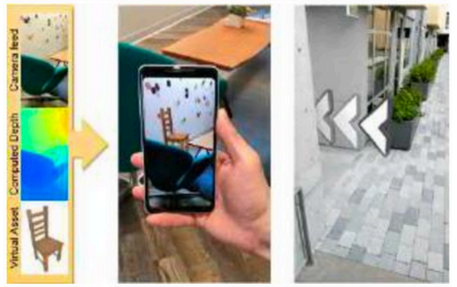
(b)



(c)



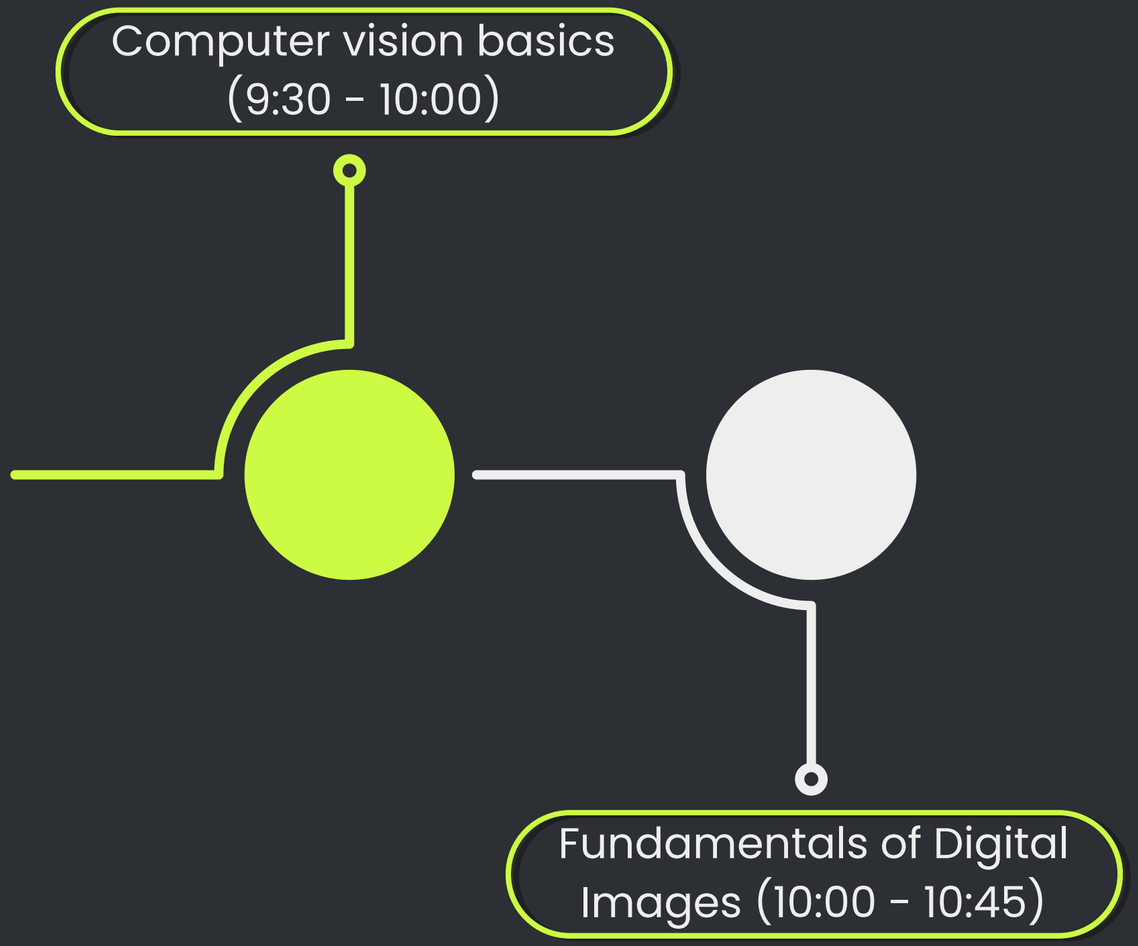
(d)



(e)

(f)

Morning Session (9:30 – 12:00)



Fundamentals of Digital Images

Machine Vision
using Python (MVUP01)

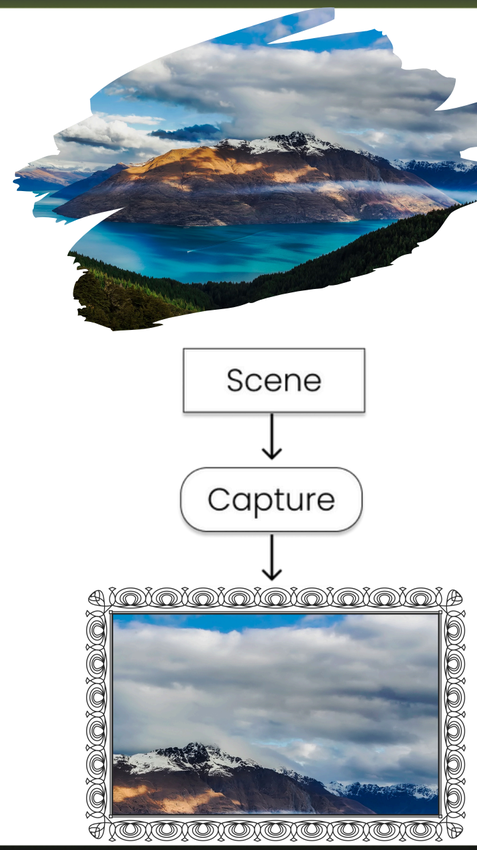
stats

Fundamentals of Digital Images

Machine Vision
using Python (MVUP01)

stats

Fundamentals of Digital Images



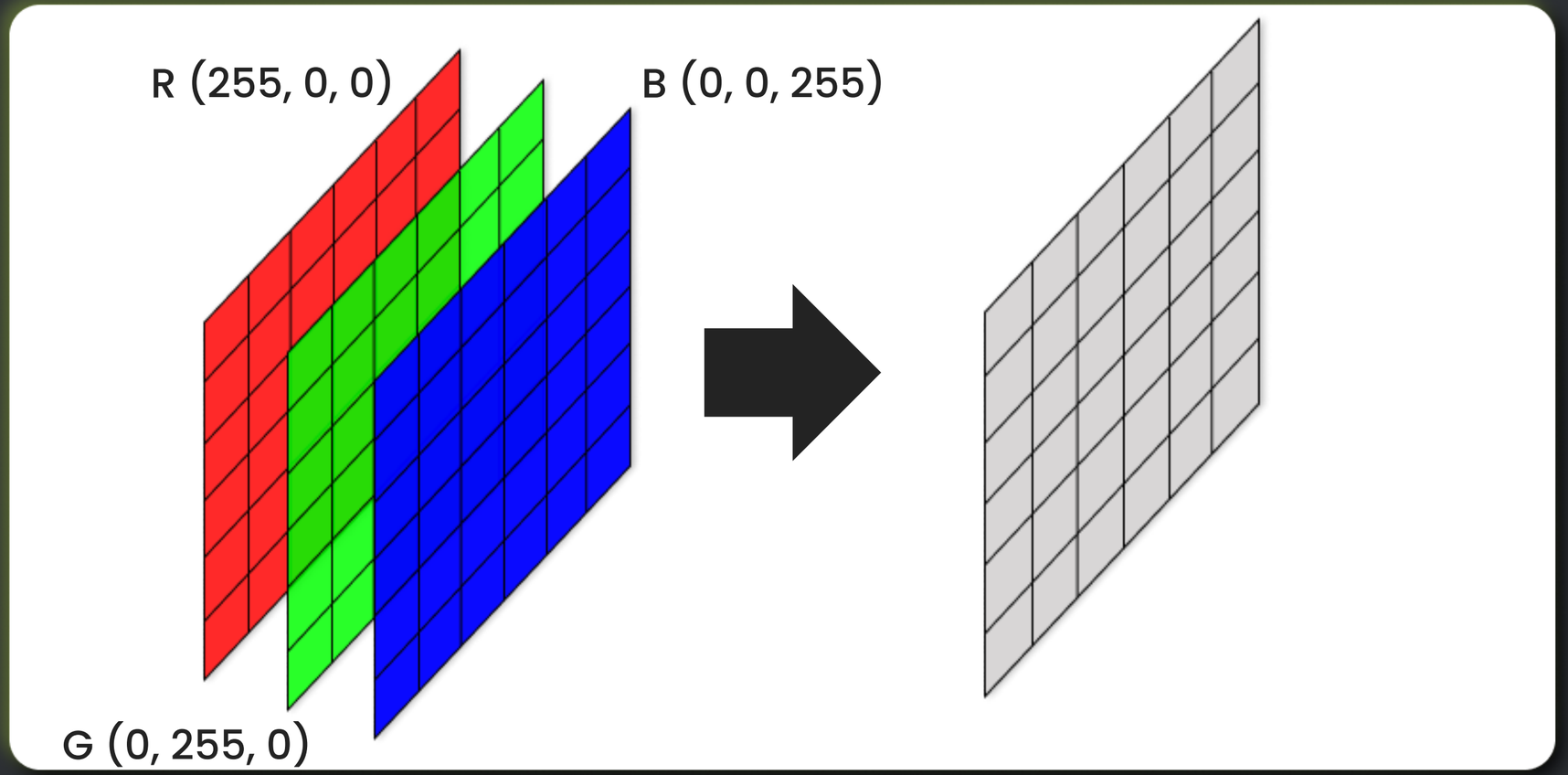
Fundamentals of Digital Images

Capture: This involves choosing and configuring suitable equipment to capture images of sufficient quality to pass into the machine vision pipeline. The main imaging devices and cameras used to capture images relating to pest species detection include microscopes, camera traps, mobile phones and SLR cameras. Other technologies, such as sonification, can also produce images

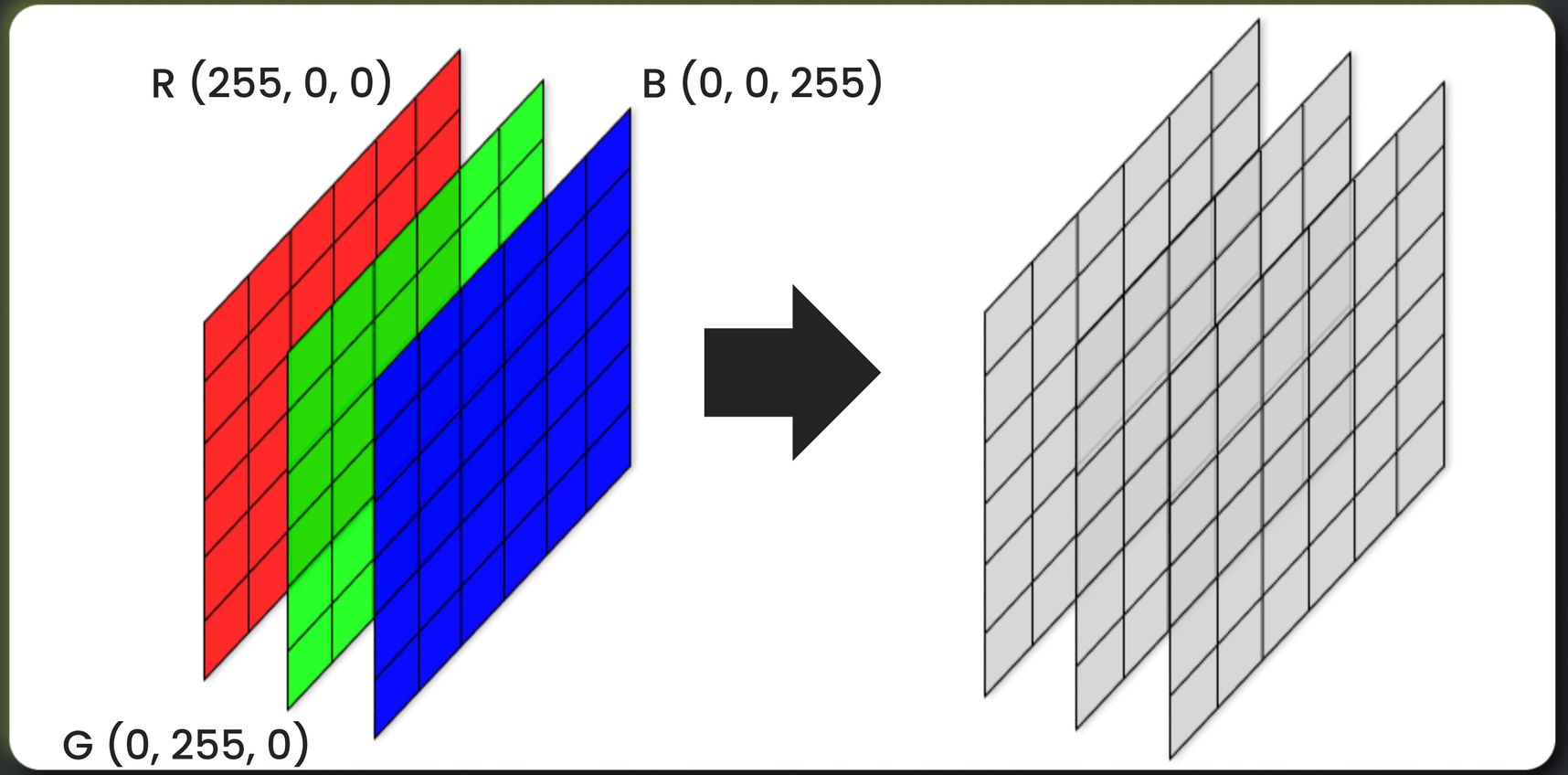
Fundamentals of Digital Images

Fundamentals of Digital Images

Fundamentals of Digital Images

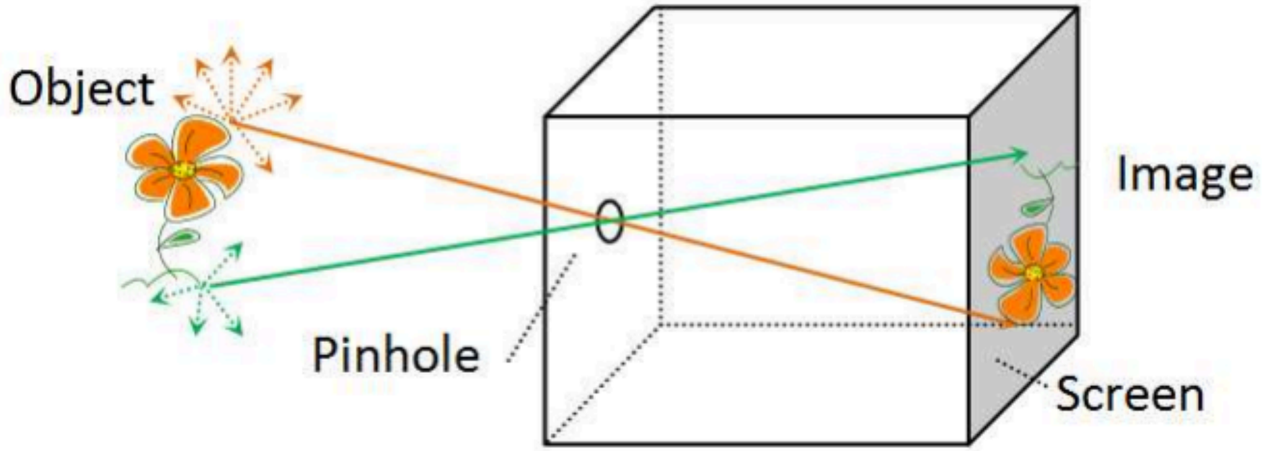


Fundamentals of Digital Images



Fundamentals of Digital Images

Fundamentals of Digital Images



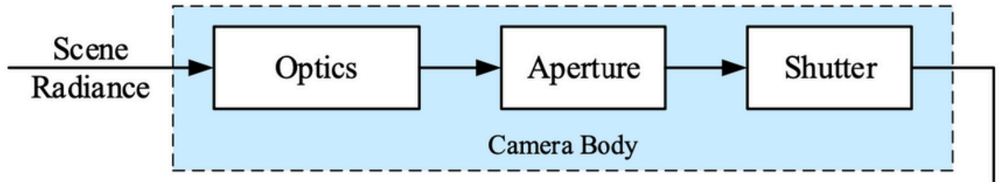
A pin hole camera producing a 2D image of a 3D object

Fundamentals of Digital Images

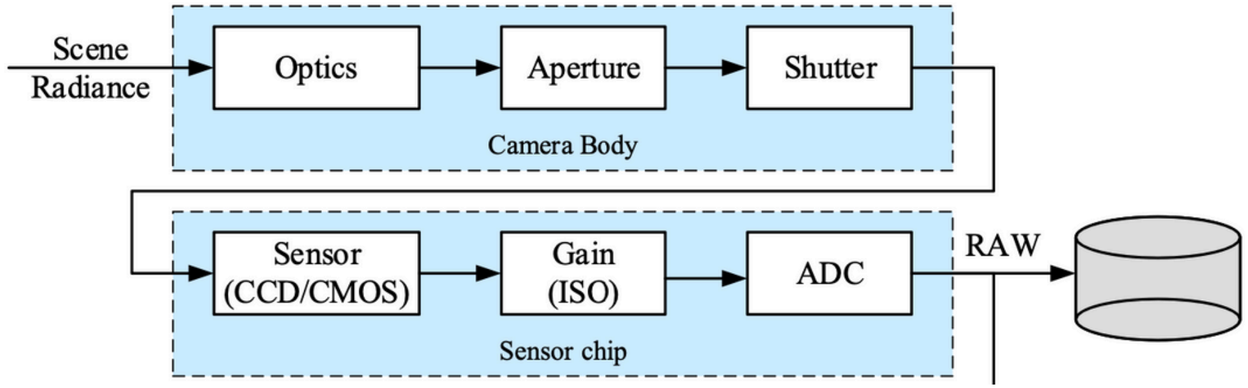
Machine Vision
using Python (MVUP01)

stats

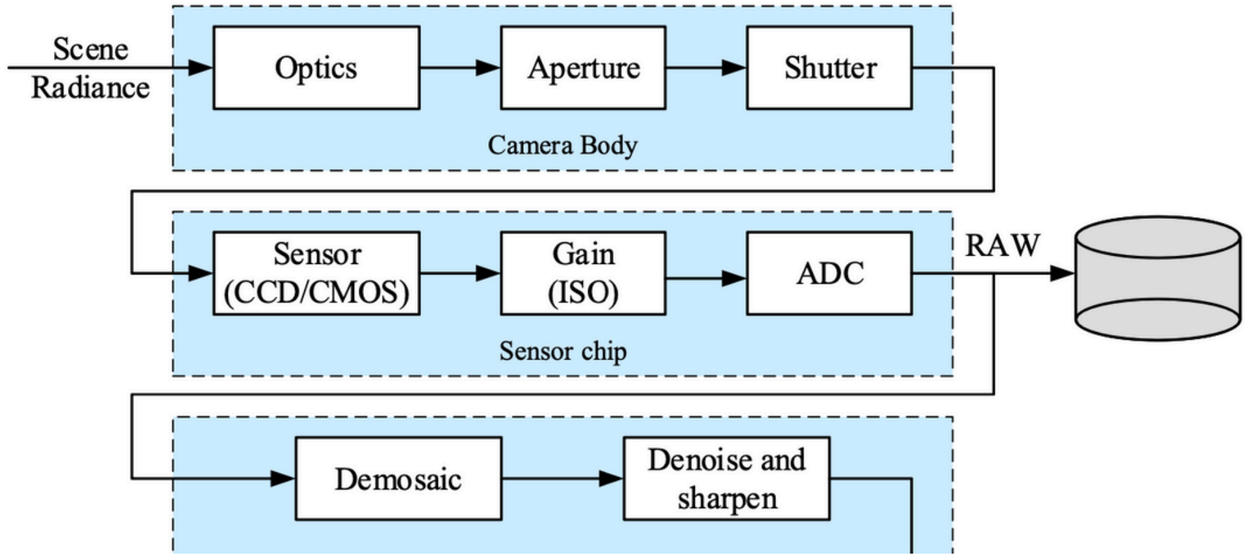
Fundamentals of Digital Images



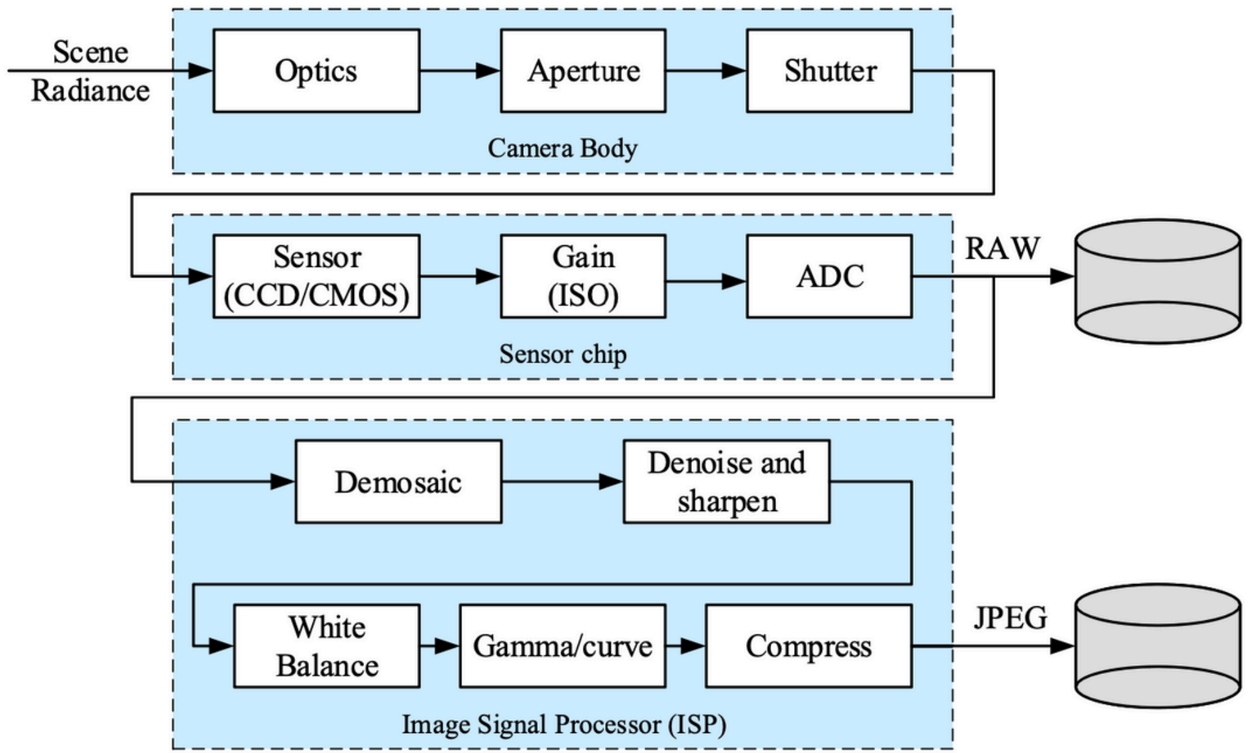
Fundamentals of Digital Images



Fundamentals of Digital Images



Fundamentals of Digital Images



Fundamentals of Digital Images

Machine Vision
using Python (MVUP01)

stats

Fundamentals of Digital Images

How can I read an image in python?

```
!pip install opencv-python
```

Fundamentals of Digital Images

How can I read an image in python?

```
!pip install opencv-python
```

```
Collecting opencv-python
  Downloading opencv_python-4.11.0.86-cp37-abi3-macosx_13_0_arm64.whl.metadata (20 kB)
Collecting numpy>=1.21.2 (from opencv-python)
  Downloading numpy-2.2.2-cp313-cp313-macosx_14_0_arm64.whl.metadata (62 kB)
Downloading opencv_python-4.11.0.86-cp37-abi3-macosx_13_0_arm64.whl (37.3 MB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 37.3/37.3 MB 30.0 MB/s eta 0:00:0000:0100:01
Downloading numpy-2.2.2-cp313-cp313-macosx_14_0_arm64.whl (5.1 MB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 5.1/5.1 MB 26.9 MB/s eta 0:00:00
Installing collected packages: numpy, opencv-python
Successfully installed numpy-2.2.2 opencv-python-4.11.0.86
```

Fundamentals of Digital Images

How can I read an image in python?

```
!pip install numpy
```

Fundamentals of Digital Images

How can I read an image in python?

```
!pip install numpy
```

```
Requirement already satisfied: numpy in
/Users/gabriel/miniforge3/envs/mvup/lib/python3.13/site-packages (2.2.2)
```

Fundamentals of Digital Images

How can I read an image in python?

```
import cv2
import numpy as np
```

Fundamentals of Digital Images

How can I read an image in python?

```
# Read image in BGR format (OpenCV default)
image = cv2.imread('CaputedImage.jpg')
```

Fundamentals of Digital Images

How can I read an image in python?

```
image.shape
```

Fundamentals of Digital Images

How can I read an image in python?

```
image.shape
```

```
(108, 192, 3)
```

Fundamentals of Digital Images

How can I read an image in python?

```
# Check pixel values at specific coordinates (row=100, column=50)
image[100, 50]
```

Fundamentals of Digital Images

How can I read an image in python?

```
# Check pixel values at specific coordinates (row=100, column=50)
image[100, 50]
```

```
array([24, 58, 48], dtype=uint8)
```

Fundamentals of Digital Images

How can I read an image in python?

```
b, g, r = image[100, 50] # OpenCV stores as BGR
print(f"BGR values at (100,50): Blue={b}, Green={g}, Red={r}")
```

Fundamentals of Digital Images

How can I read an image in python?

```
b, g, r = image[100, 50] # OpenCV stores as BGR
print(f"BGR values at (100,50): Blue={b}, Green={g}, Red={r}")
```

```
BGR values at (100,50): Blue=24, Green=58, Red=48
```

Fundamentals of Digital Images

How can I read an image in python?

```
# Split into individual color channels
blue_ch, green_ch, red_ch = cv2.split(image)
blue_ch
```

Fundamentals of Digital Images

How can I read an image in python?

```
# Split into individual color channels
blue_ch, green_ch, red_ch = cv2.split(image)
blue_ch
```

```
array([[231  232, 231, ..., 202  201  200],
       [229  228  230       196  197  194],
       [224  225, 227, ..., 201  204, 203],
       ...,
       [  0    4,   0, ...,  15    3,   0],
       [  0    0    0      ,  24    8    0],
       [ 13    2,   0, ...,  21    9,   2]], shape=(108, 192), dtype=uint8)
```

Fundamentals of Digital Images

How can I read an image in python?

```
# Visualize individual channels in color
zeros = np.zeros_like(blue_ch)
blue_display = cv2.merge([blue_ch, zeros, zeros])  # Blue channel in color
green_display = cv2.merge([zeros, green_ch, zeros]) # Green channel
red_display = cv2.merge([zeros, zeros, red_ch])     # Red channel
```

Fundamentals of Digital Images

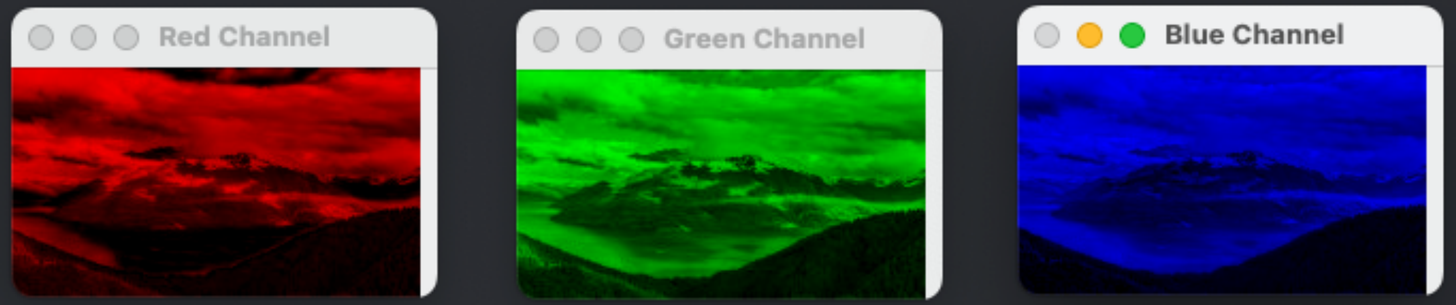
How can I read an image in python?

```
# Display channel visualizations
cv2.imshow('Blue Channel', blue_display)
cv2.imshow('Green Channel', green_display)
cv2.imshow('Red Channel', red_display)
cv2.waitKey(0)  # Wait for key press
cv2.destroyAllWindows()
cv2.waitKey(1)
```

Fundamentals of Digital Images

How can I read an image in python?

```
# Display channel visualizations
cv2.imshow('Blue Channel', blue_display)
cv2.imshow('Green Channel', green_display)
cv2.imshow('Red Channel', red_display)
cv2.waitKey(0)  # Wait for key press
cv2.destroyAllWindows()
cv2.waitKey(1)
```



Fundamentals of Digital Images

How can I read an image in python?

```
# Image visualisation with all channels
cv2.imshow('Original image', image)
cv2.waitKey(0)  # Wait for key press
cv2.destroyAllWindows()
cv2.waitKey(1)
```

Fundamentals of Digital Images

How can I read an image in python?

```
# Image visualisation with all channels
cv2.imshow('Original image', image)
cv2.waitKey(0)  # Wait for key press
cv2.destroyAllWindows()
cv2.waitKey(1)
```

Fundamentals of Digital Images

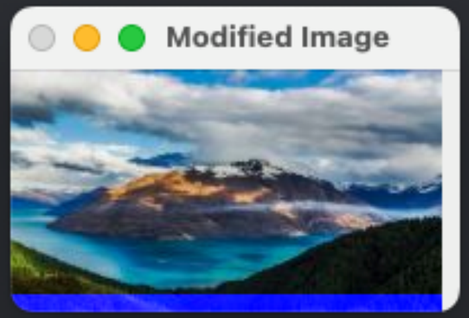
How can I read an image in python?

```
# Image slicing and modification
modified = image.copy()
modified[100:110, :, 0] = 255  # Set blue channel (rows 100-110) to max [3]
modified[200:210, :, 1] = 255  # Set green channel (rows 200-210)
modified[300:310, :, 2] = 255  # Set red channel (rows 300-310)

cv2.imshow('Modified Image', modified)
cv2.waitKey(0)
cv2.destroyAllWindows()
cv2.waitKey(1)
```

Fundamentals of Digital Images

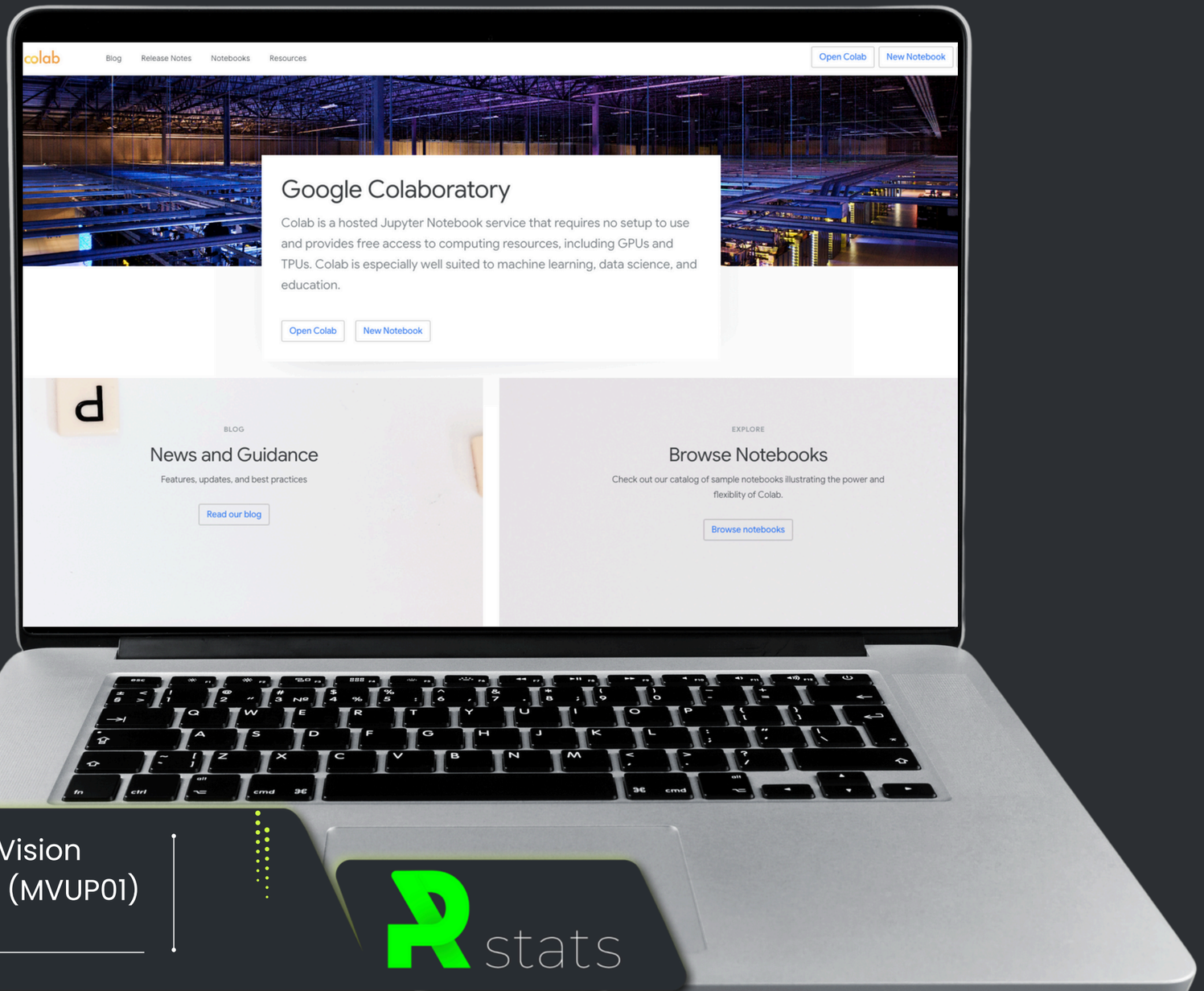
How can I read an image in python?

```python
# Image slicing and modification
modified = image.copy()
modified[100:110, :, 0] = 255  # Set blue channel (rows 100-110) to max [3]
modified[200:210, :, 1] = 255  # Set green channel (rows 200-210)
modified[300:310, :, 2] = 255  # Set red channel (rows 300-310)

cv2.imshow('Modified Image', modified)
cv2.waitKey(0)
cv2.destroyAllWindows()
cv2.waitKey(1)
```



Fundamentals of Digital Images

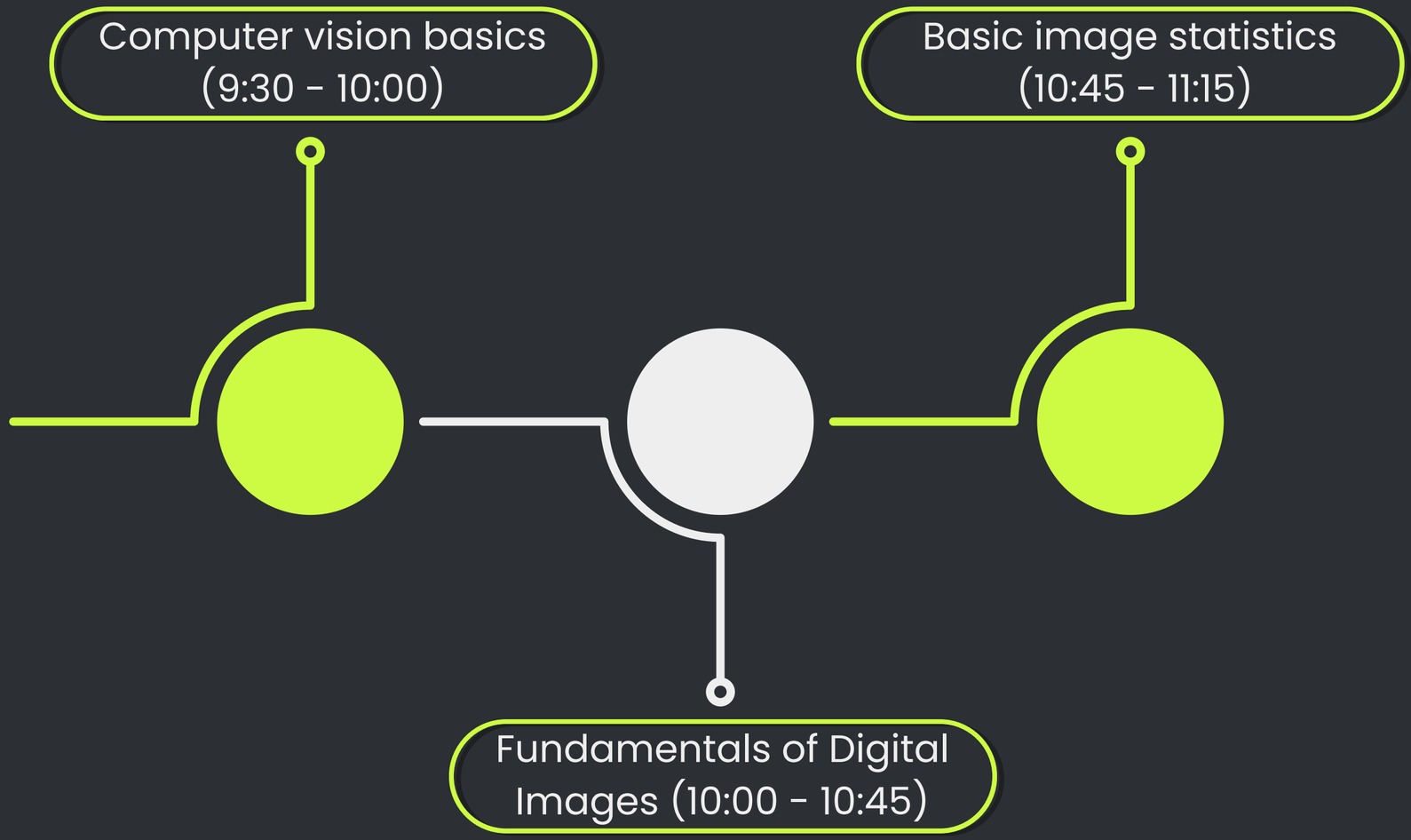
Hands-on activity:



Machine Vision
using Python (MVUP01)

stats

Morning Session (9:30 – 12:00)



Basic image statistics

Machine Vision
using Python (MVUP01)

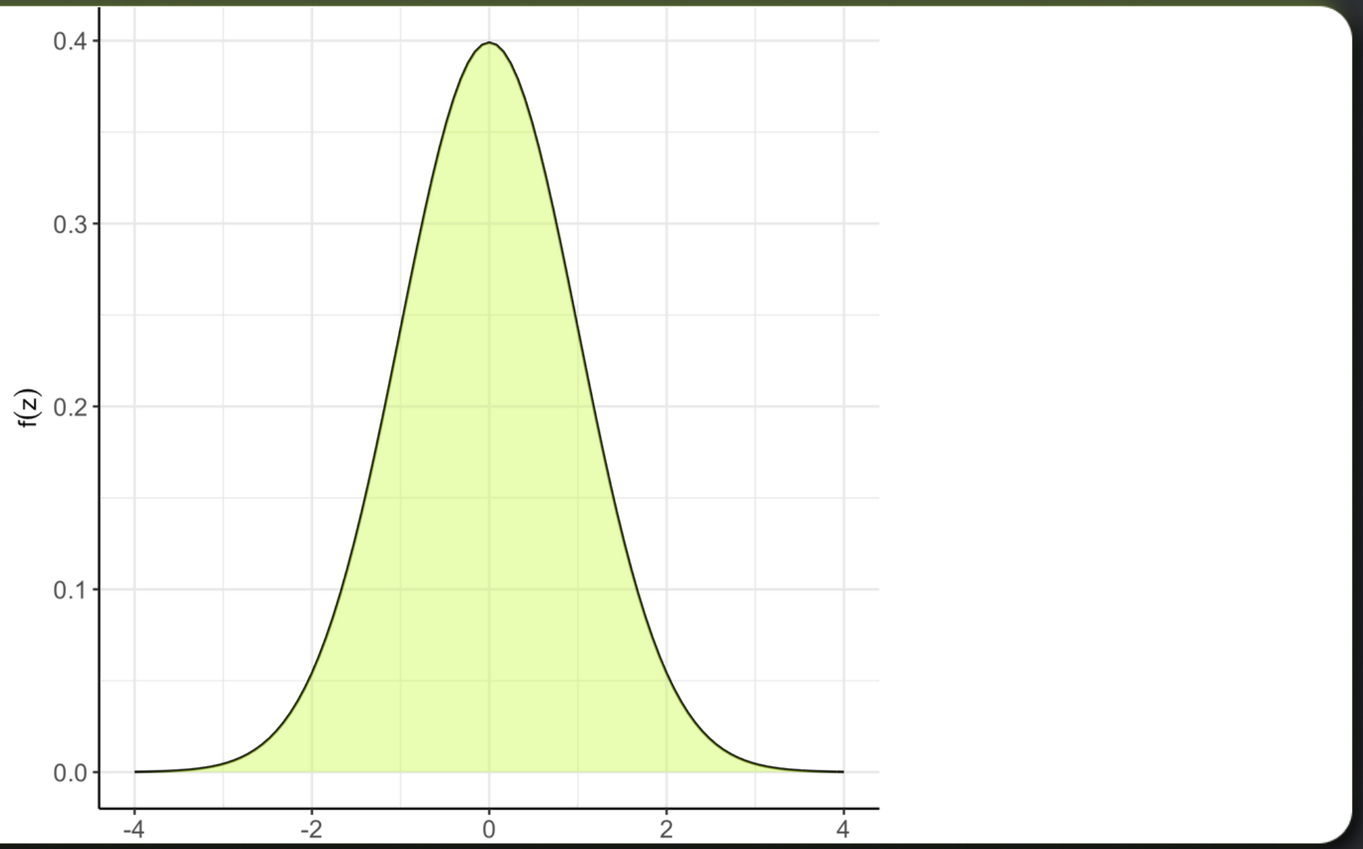
R stats

Basic image statistics

Machine Vision
using Python (MVUP01)

stats

Basic image statistics

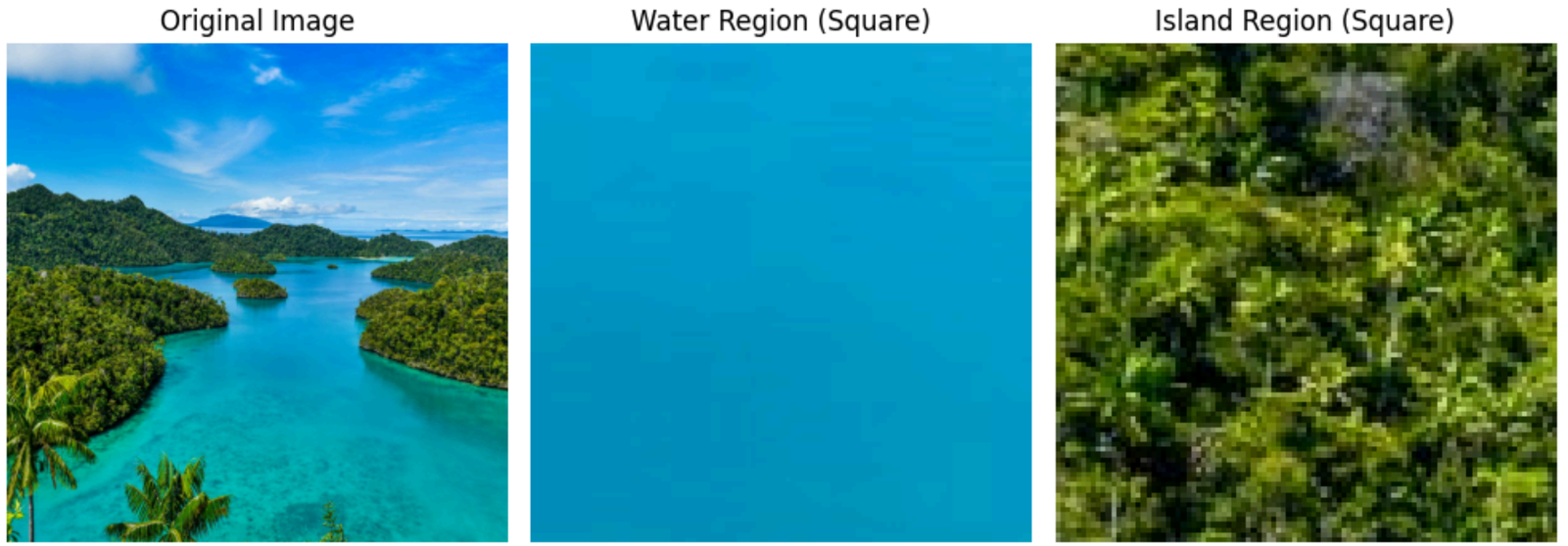


Basic image statistics

Machine Vision
using Python (MVUP01)

stats

Basic image statistics

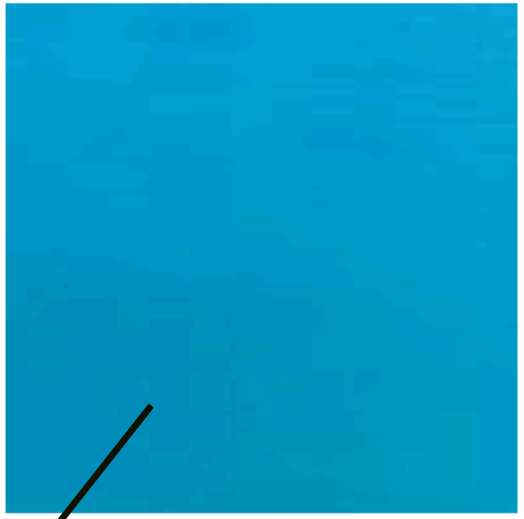


Basic image statistics

What does the B channel's statistics tell us?

Original Image

Water Region (Square)



Island Region (Square)

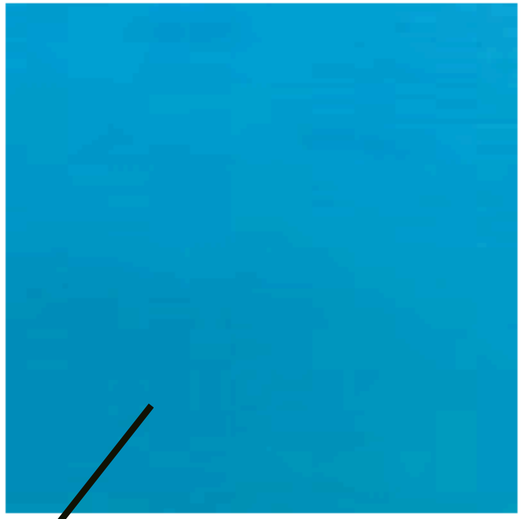
```
{'mean': np.float64(198.5987),
 'dispersion': np.float64(9.104200036796206),
 'percentile_2.5': np.float64(185.0)}
```

Basic image statistics

What does the B channel's statistics tell us?

Original Image

Water Region (Square)



Island Region (Square)

```
{'mean': np.float64(198.5987),
 'dispersion': np.float64(9.104200036796206),
 'percentile_2.5': np.float64(185.0)}
```

```
{'mean': np.float64(22.8878),
 'dispersion': np.float64(26.3196164706099),
 'percentile_2.5': np.float64(0.0)}
```

Basic image statistics

How can I obtain descriptive statistics in Python?

```
!pip install numpy
```

Basic image statistics

How can I obtain descriptive statistics in Python?

```
!pip install numpy
```

```
Requirement already satisfied: numpy in
/Users/gabriel/miniforge3/envs/mvup/lib/python3.13/site-packages (2.2.2)
```

Basic image statistics

How can I obtain descriptive statistics in Python?

```
import cv2
import numpy as np
```

Basic image statistics

How can I obtain descriptive statistics in Python?

```
# Split the image into its RGB channels
blue_channel, green_channel, red_channel = cv2.split(image)
```

Basic image statistics

How can I obtain descriptive statistics in Python?

```python
# Function to calculate basic statistics for a given channel
def calculate_statistics(channel):
    stats = {
        "mean": np.mean(channel),
        "std_dev": np.std(channel),
        "min": np.min(channel),
        "max": np.max(channel)
    }
    return stats
```

Basic image statistics

How can I obtain descriptive statistics in Python?

```
# Calculate statistics for each channel
blue_stats = calculate_statistics(blue_channel)
green_stats = calculate_statistics(green_channel)
red_stats = calculate_statistics(red_channel)
```

Basic image statistics

How can I obtain descriptive statistics in Python?

```
# Print the results
print("Blue Channel Statistics:")
print(blue_stats)
print("\nGreen Channel Statistics:")
print(green_stats)
print("\nRed Channel Statistics:")
print(red_stats)
```

Basic image statistics

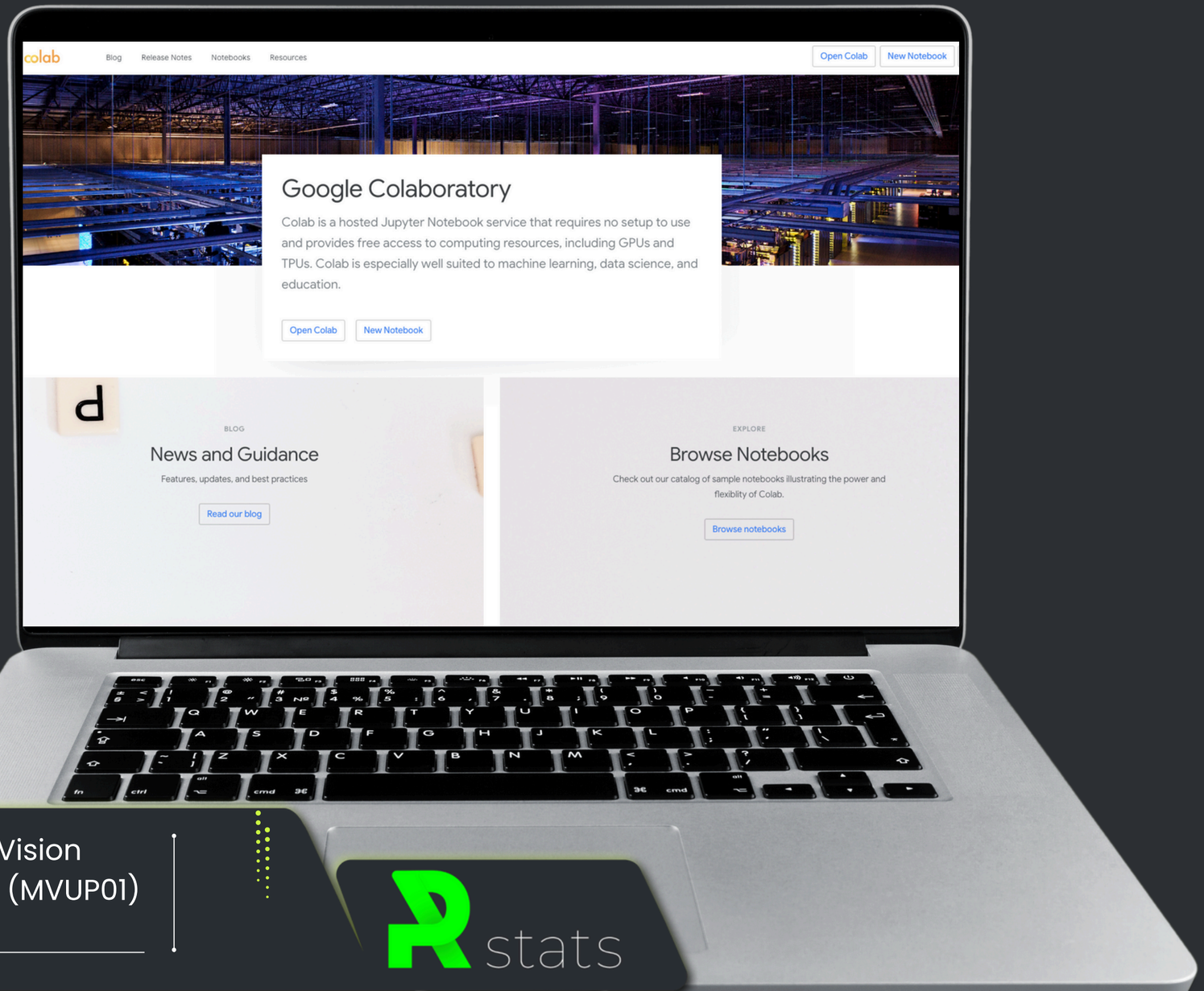
How can I obtain descriptive statistics in Python?

```
# Print the results
print("Blue Channel Statistics:")
print(blue_stats)
print("\nGreen Channel Statistics:")
print(green_stats)
print("\nRed Channel Statistics:")
print(red_stats)
```

```
Blue Channel Statistics:
{'mean': np.float64(152.496436042524), 'std_dev': np.float64(85.55766375060415), 'min': np.uint8(0),
'max': np.uint8(255)}

Green Channel Statistics:
{'mean': np.float64(137.12149691358024), 'std_dev': np.float64(47.37410983584465), 'min':
np.uint8(0), 'max': np.uint8(255)}

Red Channel Statistics:
{'mean': np.float64(39.272794924554184), 'std_dev': np.float64(51.199604768029104), 'min': np.uint8(0),
 'max': np.uint8(255)}
```

Basic image statistics

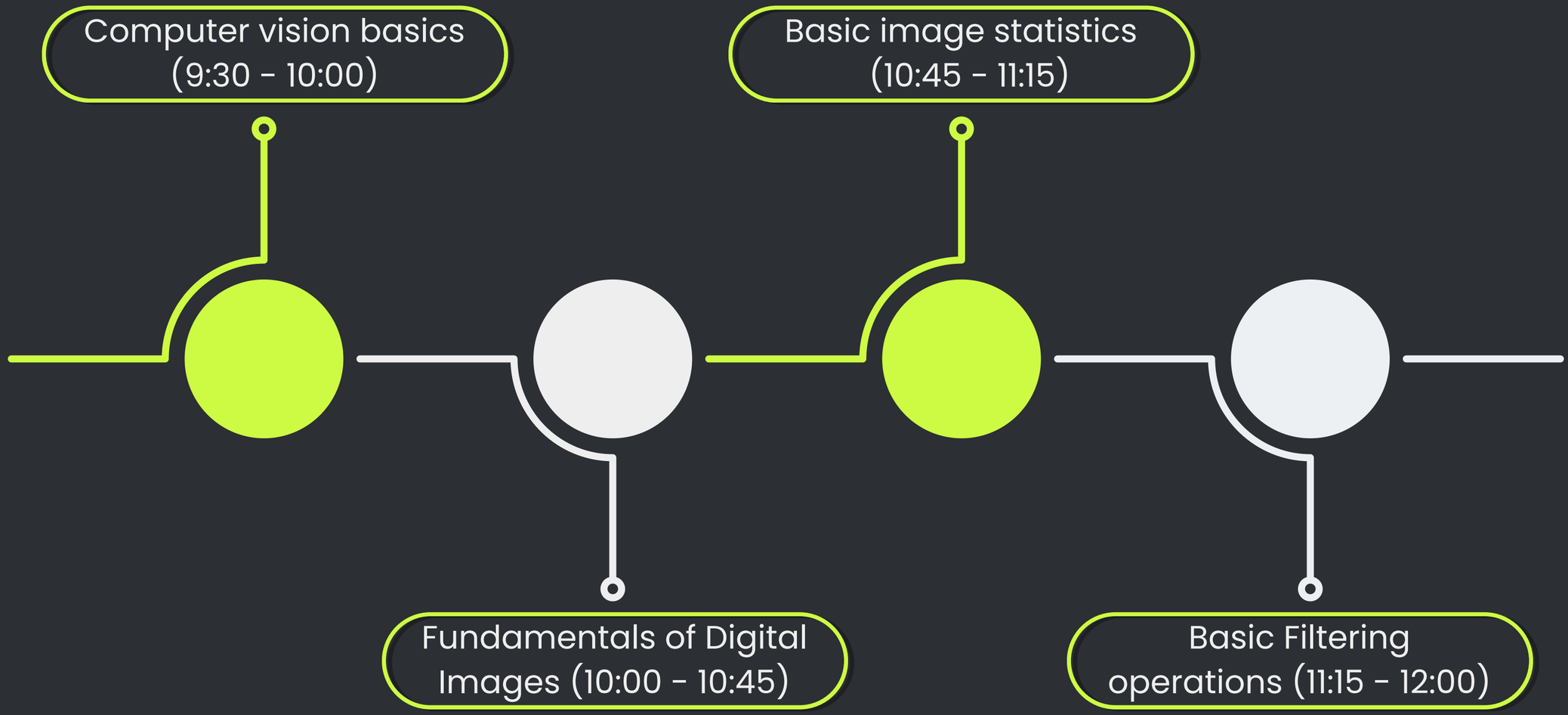
Hands-on activity:



Machine Vision
using Python (MVUP01)

stats

Morning Session (9:30 - 12:00)



Basic Filtering operations

Machine Vision
using Python (MVUP01)

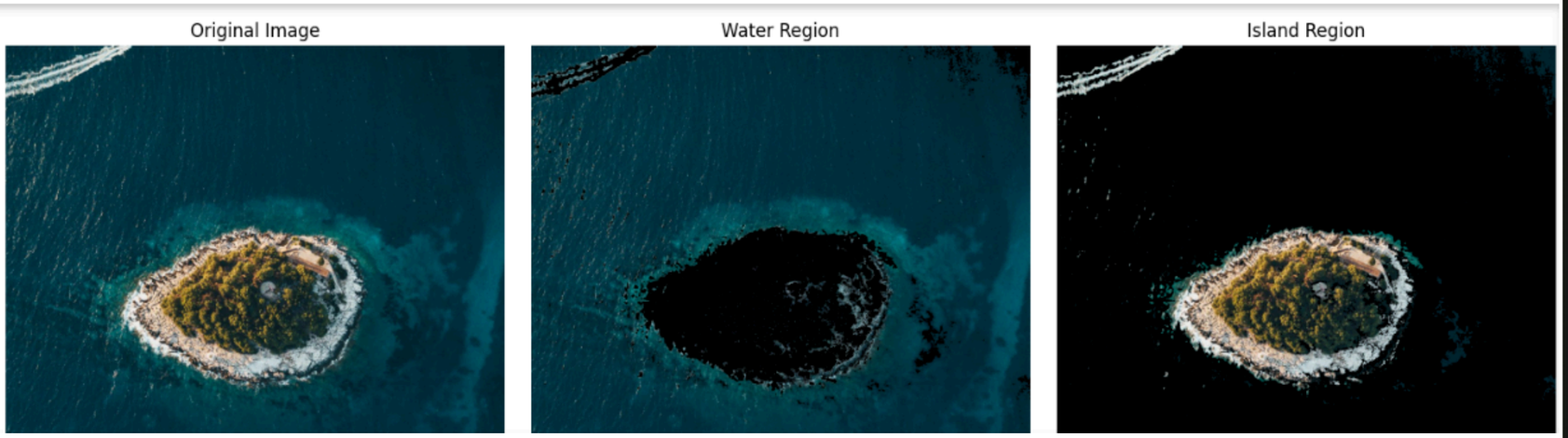
stats

Basic Filtering operations

Machine Vision
using Python (MVUP01)

Rstats

Basic Filtering operations



Basic Filtering operations

How can I filter images in Python?

```
# Load the image
    image = cv2.imread(image_path)
    image_rgb = cv2.cvtColor(image, cv2.COLOR_BGR2RGB)
```

Basic Filtering operations

How can I filter images in Python?

```
# Define thresholds for water (blueish regions)
    lower_blue = np.array([90, 50, 50])  # Adjust based on the image
    upper_blue = np.array([130, 255, 255])
```

Basic Filtering operations

How can I filter images in Python?

```
# Convert to HSV for better color segmentation
hsv_image = cv2.cvtColor(image, cv2.COLOR_BGR2HSV)
# Create a mask for water
water_mask = cv2.inRange(hsv_image, lower_blue, upper_blue)
```

Basic Filtering operations

RGB Color Model:

- **Additive colour system:** Combines red, green, and blue light in varying intensities (0-255 in 8-bit systems) to create colours
- **Device-dependent:** Colour appearance varies across displays due to different colour gamuts and calibration
- **Native display format:** Matches how screens physically produce colours using subpixels
- **Three correlated channels:** Changing one channel affects perceived brightness and colour balance

Basic Filtering operations

HSV Color Model:

- **Perceptual organisation**: Separates colour aspects into:
  - **Hue**: Dominant wavelength (0-179)
  - **Saturation**: Colour purity (0 represents grayscale, 255 is full saturation)
  - **Value**: Brightness (0 is black, 255 is maximum brightness)
- **Device-independent**: Represents colours in human-meaningful terms
- **Decoupled channels**: Allows independent adjustment of colour (hue), intensity (saturation), and brightness (value)

Basic Filtering operations

How can I filter images in Python?

```
# Invert the mask for the island
island_mask = cv2.bitwise_not(water_mask)

# Extract water and island regions
water = cv2.bitwise_and(image_rgb, image_rgb, mask=water_mask)
island = cv2.bitwise_and(image_rgb, image_rgb, mask=island_mask)
```

Basic Filtering operations

How can I filter images in Python?

```
# Plot the results
plt.figure(figsize=(15, 5))
plt.subplot(1, 3, 1)
plt.imshow(image_rgb)
plt.title('Original Image')
plt.axis('off')
```

Basic Filtering operations

How can I filter images in Python?

```
plt.subplot(1, 3, 2)
plt.imshow(water)
plt.title('Water Region')
plt.axis('off')
```

Basic Filtering operations

How can I filter images in Python?

```
plt.subplot(1, 3, 3)
plt.imshow(island)
plt.title('Island Region )
plt.axis('off')
plt.tight_layout()
plt.show()
```

Basic Filtering operations

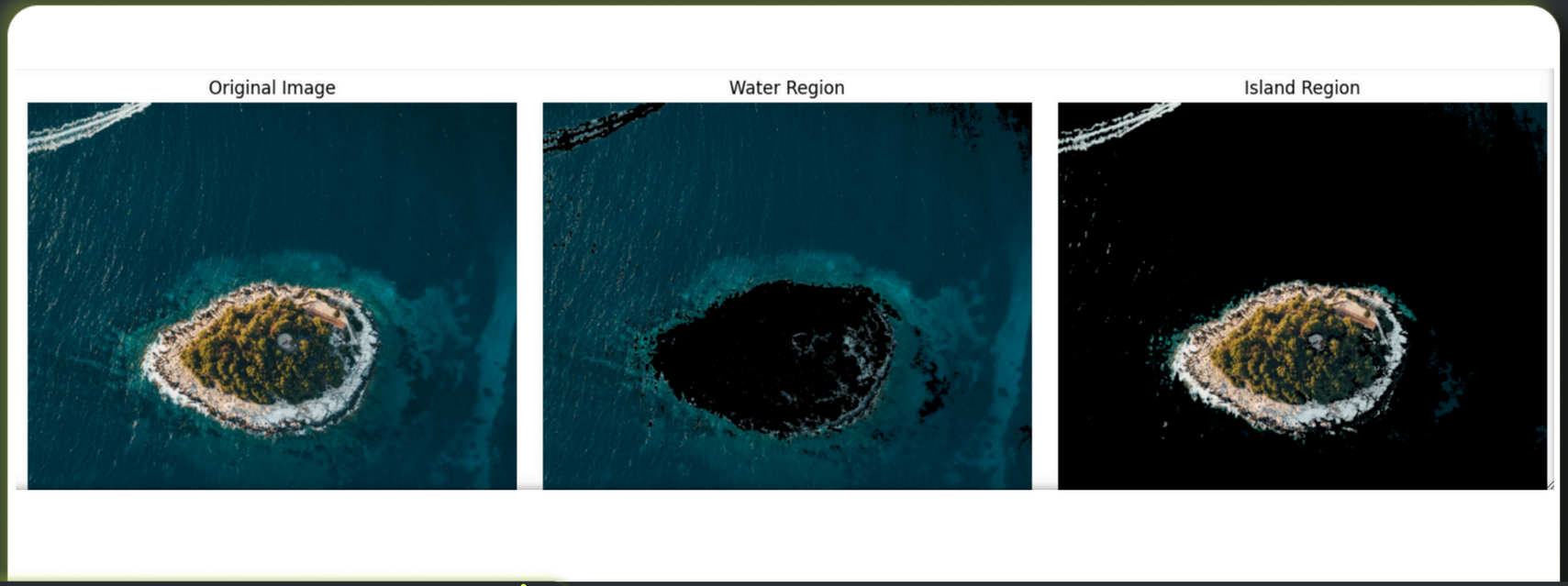
How can I filter images in Python?

```
# Example usage
separate_water_island('Island1.jpg')
```

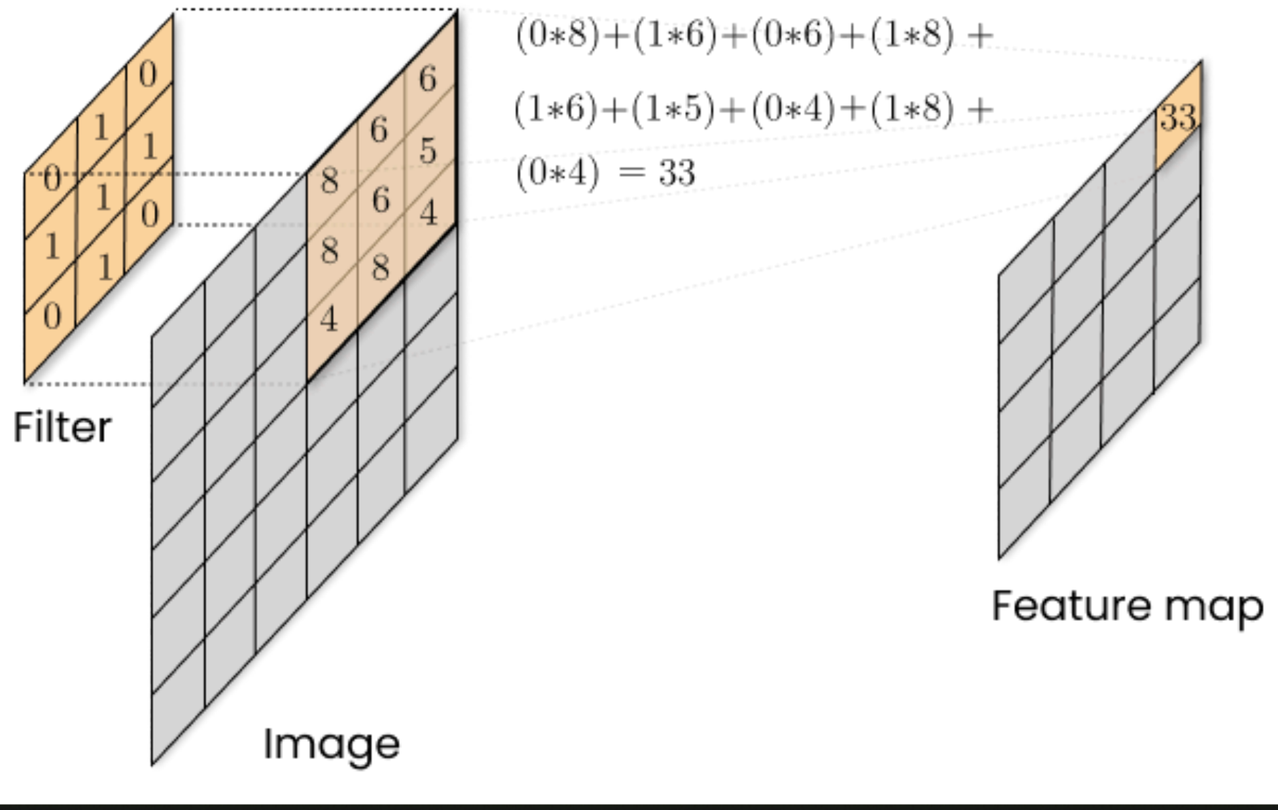
Basic Filtering operations

How can I filter images in Python?

```
# Example usage
separate_water_island('Island1.jpg')
```



Basic Filtering operations



Basic Filtering operations

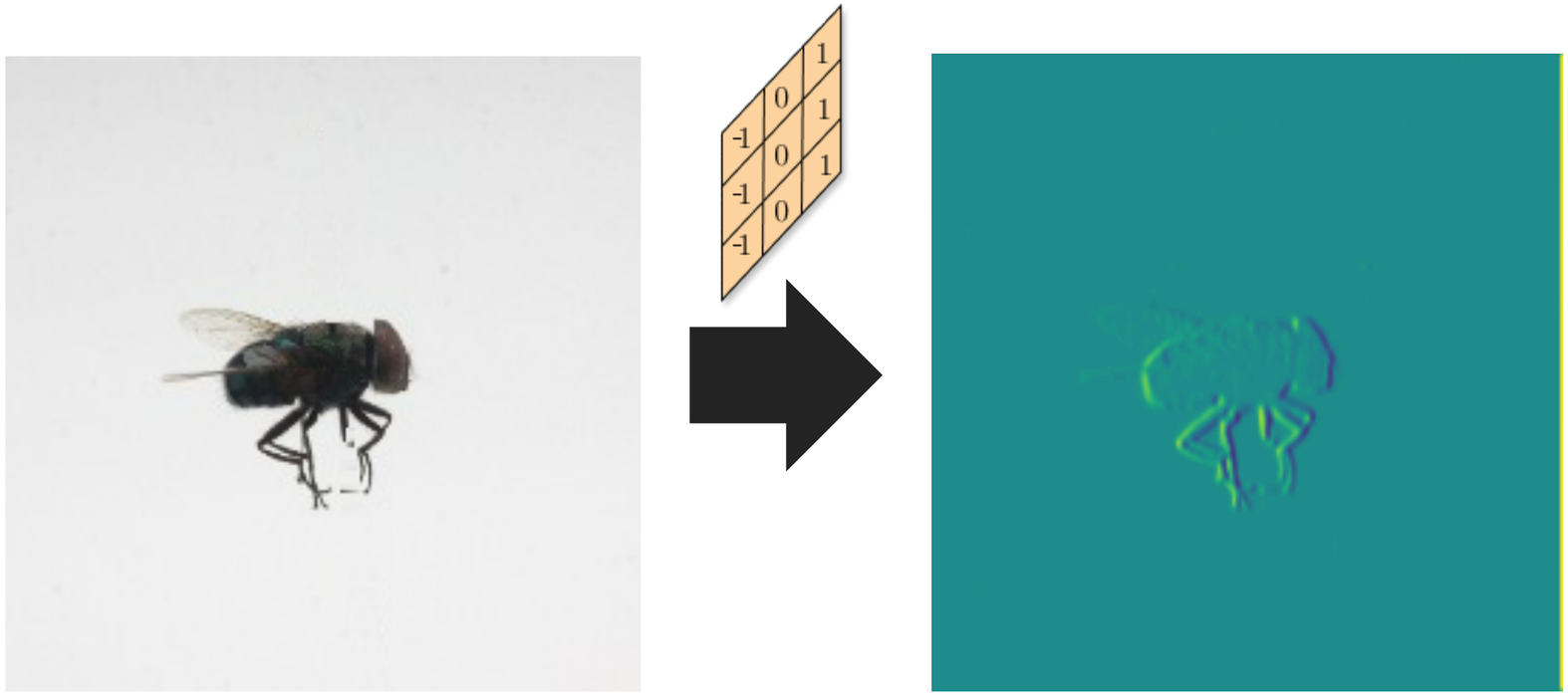
Two common choices for D_x and D_y are the *Prewitt filters*

$$D_x = \begin{bmatrix} -1 & 0 & 1 \\ -1 & 0 & 1 \\ -1 & 0 & 1 \end{bmatrix} \text{ and } D_y = \begin{bmatrix} -1 & -1 & -1 \\ 0 & 0 & 0 \\ 1 & 1 & 1 \end{bmatrix},$$

and *Sobel filters*

$$D_x = \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix} \text{ and } D_y = \begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{bmatrix}.$$

Basic Filtering operations

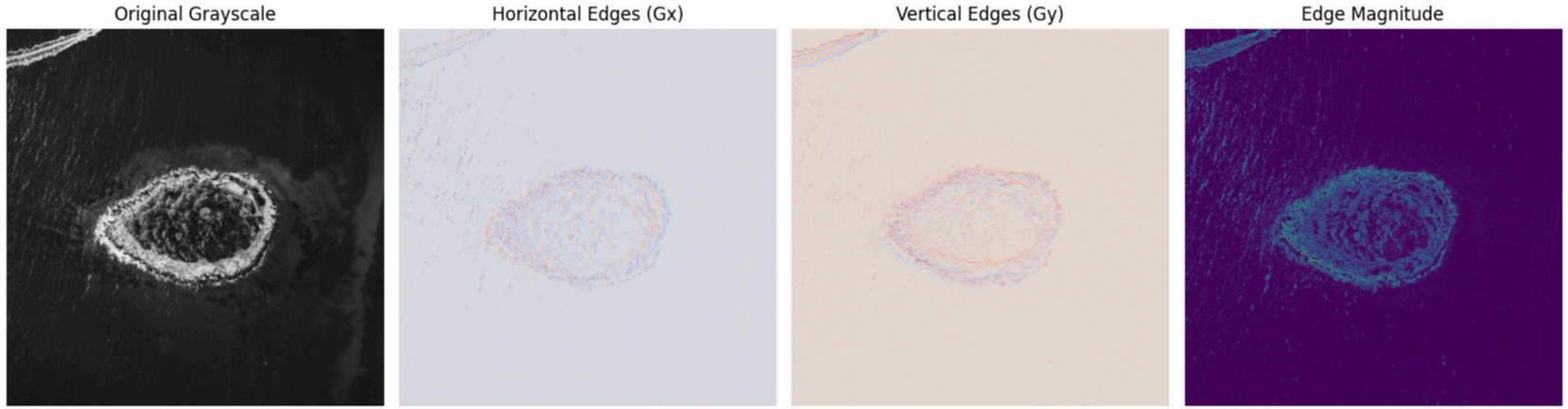


Basic Filtering operations

Machine Vision
using Python (MVUP01)

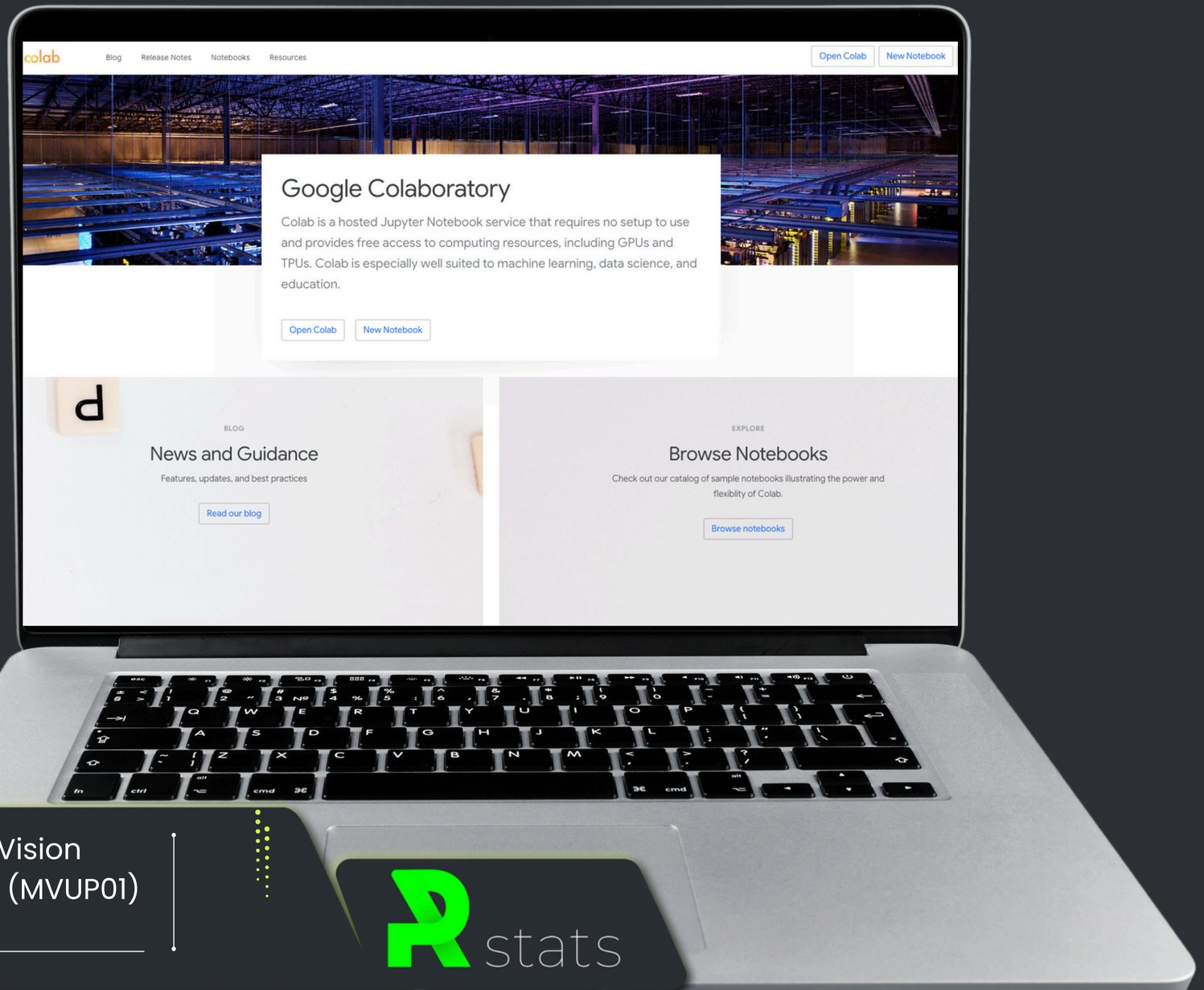
stats

Basic Filtering operations



Basic image statistics

Hands-on activity:



stats

Lunch time

Machine Vision
using Python (MVUP01)

stats