Machine Learning

Computer Vision

Salymbekov University

Miss Aliia Beishenalieva

aliya.beiwenalieva@gmail.com

How the pixel information is used in CV?

- **Canny Edge Detection**

```
import numpy as np
import cv2 as cv
from matplotlib import pyplot as plt

img = cv.imread('messi5.jpg', cv.IMREAD_GRAYSCALE)
assert img is not None, "file could not be read, check with os.path.exists()"
edges = cv.Canny(img,100,200)

plt.subplot(121),plt.imshow(img,cmap = 'gray')
plt.title('Original Image'), plt.xticks([]), plt.yticks([])
plt.subplot(122),plt.imshow(edges,cmap = 'gray')
plt.title('Edge Image'), plt.xticks([]), plt.yticks([])

plt.show()
```

https://docs.opencv.org/4.x/da/d22/tutorial_py_canny.html

Object detection

❑ Haar cascades

```python
import cv2
import matplotlib.pyplot as plt

# Load the pre-trained Haar cascade classifier for face detection
face_cascade = cv2.CascadeClassifier(cv2.data.haarcascades + 'haarcascade_frontalface_default.xml')

# Read the image
image = cv2.imread('path_to_your_image.jpg')
gray_image = cv2.cvtColor(image, cv2.COLOR_BGR2GRAY)  # Convert to grayscale

# Detect faces in the image
faces = face_cascade.detectMultiScale(gray_image, scaleFactor=1.1, minNeighbors=5, minSize=(30, 30))

# Draw rectangles around detected faces
for (x, y, w, h) in faces:
    cv2.rectangle(image, (x, y), (x + w, y + h), (255, 0, 0), 2)  # Blue rectangle with thickness 2

# Convert image color to RGB (from BGR) for displaying with matplotlib
image_rgb = cv2.cvtColor(image, cv2.COLOR_BGR2RGB)

# Display the output using matplotlib
plt.imshow(image_rgb)
plt.axis('off')  # Hide axes
plt.show()
```

Object detection

- **Make mini-project using traditional object detection**

Thank you for your attention

Computer Vision

Salymbekov University

Miss Aliia Beishenalieva

aliya.beiwenalieva@gmail.com