Experiment No: 8

Name: Polomi Adak

Roll No: 20

Aim: To Detecting and Recognizing Objects

Objective: Object Detection and recognition techniques HOG descriptor The Scale issues The location issue Non-maximum (or non-maxima) suppression vector machine people detection

Theory:

Object detection and recognition Techniques:-

Object detection is a computer vision technique for locating instances of objects in images or videos. Object detection algorithms typically leverage machine learning or deep learning to produce meaningful results.

Object detection and recognition are fundamental techniques in computer vision. Object detection involves identifying and locating objects within an image or video stream, typically using bounding boxes to indicate their positions. Modern object detection techniques often rely on deep learning approaches, such as Convolutional Neural Networks (CNNs), which can handle complex scenes and a wide variety of objects.

Object recognition, on the other hand, goes a step further by not only detecting objects but also identifying their specific class or category. This process requires training models on labeled datasets and is commonly achieved using deep learning frameworks like YOLO (You Only Look Once) and Faster R-CNN. These techniques find applications in numerous fields, including autonomous vehicles, surveillance, facial recognition, and augmented reality, enabling computers to interpret and interact with the visual world in a manner that mirrors human perception.

HOG descriptors:-

HOG (Histogram of Oriented Gradients) descriptors are a prominent feature extraction technique in computer vision. They capture information about the distribution of local gradient orientations in an image. HOG descriptors are commonly used for object detection and pedestrian recognition tasks. By dividing an image into small cells and calculating gradient orientation histograms within each cell, HOG descriptors effectively capture shape and texture information. They are robust to changes in lighting and contrast, making them useful for various real-world applications. HOG descriptors have been instrumental in the development of object detection algorithms, including the popular "SVM + HOG" approach, which is widely used in pedestrian and face detection systems.

The scale issue:-

The scale issue refers to the challenge of recognizing objects or patterns across different scales within an image. Objects can appear larger or smaller due to their proximity to the camera or the angle at which they are viewed. This variation in size can make it difficult for computer vision algorithms to detect and identify objects accurately. To address the scale issue, techniques such as image pyramid representations and scale-invariant features are used. These methods involve creating multiple image scales or extracting features that are resilient to scale changes, enabling more robust object detection and recognition in diverse visual contexts. Overcoming the scale issue is crucial for tasks like object detection, image classification, and scene understanding in computer vision applications.

The Location issue:-

The location issue is a fundamental challenge related to object recognition and tracking within a visual environment. It refers to the problem of accurately determining the position, orientation, and scale of objects or features within an image or a scene. Addressing the location issue is crucial for various applications, including robotics, augmented reality, and autonomous vehicles, as it impacts the system's ability to interpret and interact with the real world effectively. Computer vision algorithms often rely on techniques such as feature detection, matching, and pose estimation to tackle this issue, ensuring that objects can be accurately located and identified within the visual data, enabling machines to understand and interact with their surroundings.

Non-maximum(or Non-maxima)Suppression:-

Non-maximum suppression is a crucial techniqueused to reduce the number of local maxima or detections in an image. It is commonly applied in object detection and feature extraction processes. When detecting features like edges or keypoints, the algorithm identifies local maxima in response values, and non-maximum suppression helps eliminate redundant or non-essential detections. It works by considering each pixel's strength or response and comparing it to its neighbors in a local window. If a pixel is not the maximum within the neighborhood, it is suppressed or set to zero. This ensures that only the most prominent and distinct features are retained, improving the accuracy and efficiency of various computer vision tasks.

Support vector machines:-

Support Vector Machines (SVMs) in computer vision are powerful machine learning models used for tasks like object classification and image segmentation. SVMs excel in finding the optimal decision boundary to separate different objects or regions in images. They are particularly effective in scenarios with complex data distributions and have been widely adopted in image recognition, face detection, and image-based classification tasks, providing robust and accurate results in the field of computer vision.

Code:

```
from google.colab.patches import cv2_imshow

pip install opencv-python

import cv2
filename = 'pedestrians_2.jpg'

def main():

 # Create a HOGDescriptor object
 hog = cv2.HOGDescriptor()

 # Initialize the People Detector
 hog.setSVMDetector(cv2.HOGDescriptor_getDefaultPeopleDetector())

 # Load an image
 image = cv2.imread(filename)



 (bounding_boxes, weights) = hog.detectMultiScale(image,
                                    winStride=(4, 4),
                                    padding=(8, 8),
                                    scale=1.05)

 # Draw bounding boxes on the image
 for (x, y, w, h) in bounding_boxes:
  cv2.rectangle(image,
          (x, y),
          (x + w, y + h),
```

```
                (127, 0, 255),
                 4)

    # Create the output file name by removing the '.jpg' part
    size = len(filename)
    new_filename = filename[:size - 4]
    new_filename = new_filename + '_detect.jpg'

    # Save the new image in the working directory
    cv2.imwrite(new_filename, image)



    cv2_imshow(image)



    cv2.waitKey(0)

    # Close all windows
    cv2.destroyAllWindows()

main()
```

Input:

Output: