**Session 2025-2026**

|  |  |
| --- | --- |
| **Vision:** Dream of where you want. | **Mission:** Means to achieve Vision |

**Program Educational Objectives of the program (PEO):** (broad statements that describe the professional and career accomplishments)

|  |  |  |  |
| --- | --- | --- | --- |
| PEO1 | **Preparation** | **P: Preparation** | **Pep-CL abbreviation**  **pronounce as Pep-si-lL easy to recall** |
| PEO2 | **Core Competence** | **E: Environment (Learning Environment)** |
| PEO3 | **Breadth** | **P: Professionalism** |
| PEO4 | **Professionalism** | **C: Core Competence** |
| PEO5 | **Learning Environment** | **L: Breadth (Learning in diverse areas)** |

**Program Outcomes (PO):** (statements that describe what a student should be able to do and know by the end of a program)

**Keywords of POs:**

Engineering knowledge, Problem analysis, Design/development of solutions, Conduct Investigations of Complex Problems, Engineering Tool Usage, The Engineer and The World, Ethics, Individual and Collaborative Team work, Communication, Project Management and Finance, Life-Long Learning

**PSO Keywords:** Cutting edge technologies, Research

“I am an engineer, and I know how to apply engineering knowledge to investigate, analyse and design solutions to complex problems using tools for entire world following all ethics in a collaborative way with proper management skills throughout my life.” *to contribute to the development of cutting-edge technologies and Research*.

**Integrity:** I will adhere to the Laboratory Code of Conduct and ethics in its entirety.

**Name and Signature of Student and Date**

**Krishna Mishra**

(Signature and Date in Handwritten)

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Session** | **2025-26 (EVEN)** | | **Course Name** | **LAB: Sensor Data Analytics** | |
| **Semester** | **6th** | | **Course Code** | **23ADS1602** | |
| **Roll No** | **42** | | **Name of Student** | **Krishna Mishra** | |
|  |  | |  |  |  |
| Practical Number | | **6** | | | |
| Course Outcome | | CO1 Apply image enhancement and smoothing techniques to improve image quality for further analysis.  CO2 Extract meaningful features from images using descriptors such as HOG and SIFT. | | | |
| Aim | | Implement R-CNN algorithms for object detection**.** | | | |
| Problem Definition | | To implement an instance segmentation system using a pretrained Mask R-CNN model that detects objects in an input image, classifies them, generates bounding boxes, and produces pixel-level segmentation masks for each detected object. | | | |
| Theory  (100 words) | | The objective of this experiment is to implement instance segmentation using the Mask R-CNN deep learning model. The model should detect multiple objects present in an image, classify them into predefined categories, generate bounding boxes for localization, and produce accurate pixel-level segmentation masks for each object instance. The final output should visualize the segmented objects along with their labels and confidence scores. | | | |
| Procedure and Execution  (100 Words) | | Algorithm:  Start  Load pretrained **Mask R-CNN**  Read input image  Convert image to tensor  Perform model inference  Extract boxes, labels, scores, and masks  Apply confidence threshold  Draw bounding boxes and overlay masks  Display final output  Stop | | | |
| import torch  import torchvision  from torchvision.models.detection import maskrcnn\_resnet50\_fpn  from torchvision.models.detection import MaskRCNN\_ResNet50\_FPN\_Weights  import torchvision.transforms.functional as F  import cv2  import matplotlib.pyplot as plt  import numpy as np  from google.colab import files  device = torch.device("cuda" if torch.cuda.is\_available() else "cpu")  print("Using Device:", device)  weights = MaskRCNN\_ResNet50\_FPN\_Weights.DEFAULT  model = maskrcnn\_resnet50\_fpn(weights=weights)  model.to(device)  model.eval()  print("Mask R-CNN Model Loaded Successfully!")  uploaded = files.upload()  image\_path = list(uploaded.keys())[0]  image = cv2.imread(image\_path)  if image is None:      raise ValueError("Image not found! Please upload valid image.")  image\_rgb = cv2.cvtColor(image, cv2.COLOR\_BGR2RGB)  height, width, \_ = image\_rgb.shape  plt.figure(figsize=(8,6))  plt.imshow(image\_rgb)  plt.title("Original Image")  plt.axis("off")  plt.show()  grid\_image = image\_rgb.copy()  rows = 8  cols = 8  row\_step = height // rows  col\_step = width // cols  for r in range(rows):      y = r \* row\_step      cv2.line(grid\_image, (0, y), (width, y), (0,255,0), 1)  for c in range(cols):      x = c \* col\_step      cv2.line(grid\_image, (x, 0), (x, height), (0,255,0), 1)  plt.figure(figsize=(8,6))  plt.imshow(grid\_image)  plt.title("Grid Representation (Image as Matrix)")  plt.axis("off")  plt.show()  image\_tensor = F.to\_tensor(image\_rgb).to(device)  with torch.no\_grad():      outputs = model([image\_tensor])  boxes = outputs[0]["boxes"].cpu().numpy()  scores = outputs[0]["scores"].cpu().numpy()  labels = outputs[0]["labels"].cpu().numpy()  masks = outputs[0]["masks"].cpu().numpy()  coco\_labels = weights.meta["categories"]  threshold = 0.5  keep = scores >= threshold  boxes = boxes[keep]  scores = scores[keep]  labels = labels[keep]  masks = masks[keep]  print("\nDetected Objects:", len(boxes))  for i in range(len(masks)):      mask = masks[i, 0]      plt.figure(figsize=(4,4))      plt.imshow(mask, cmap='gray')      plt.title(f"Mask: {coco\_labels[labels[i]]} | Score: {scores[i]:.2f}")      plt.axis("off")      plt.show()  final\_image = image\_rgb.copy()  for i in range(len(boxes)):      x1, y1, x2, y2 = boxes[i].astype(int)      cv2.rectangle(final\_image, (x1,y1), (x2,y2), (0,255,0), 2)      label\_text = f"{coco\_labels[labels[i]]}: {scores[i]:.2f}"      cv2.putText(final\_image, label\_text,                  (x1, y1-5),                  cv2.FONT\_HERSHEY\_SIMPLEX,                  0.5,                  (0,255,0),                  2)      mask = masks[i, 0]      mask = mask > 0.5      colored\_mask = np.zeros\_like(final\_image)      colored\_mask[mask] = [255, 0, 0]  # Red mask      final\_image = cv2.addWeighted(final\_image, 1, colored\_mask, 0.5, 0)  plt.figure(figsize=(10,8))  plt.imshow(final\_image)  plt.title("Final Output: Mask R-CNN Instance Segmentation")  plt.axis("off")  plt.show() | | | |
| Output: | | | |
| Link of student Github profile where lab assignment has been uploaded | | [krish-na45/ComputerVision\_Practicals](https://github.com/krish-na45/ComputerVision_Practicals) | | | |
| Conclusion | | In this practical, we successfully implemented instance segmentation using Mask R-CNN with a ResNet-50 FPN backbone in Python using PyTorch and Torchvision. The pre-trained model was loaded and executed on an input image to detect objects, generate bounding boxes, classify them using COCO dataset labels, and produce pixel-level masks for each detected object.  The image was first preprocessed and visualized as a grid to understand its matrix representation. After passing the image through the Mask R-CNN model, the system predicted object locations, confidence scores, and segmentation masks. By applying a confidence threshold, only reliable detections were kept. Each detected object was displayed with its mask, bounding box, and label, demonstrating how deep learning models can perform object detection and segmentation simultaneously.  This experiment helped in understanding:   * How pre-trained deep learning models can be used for computer vision tasks * The working of instance segmentation compared to simple object detection * Image representation as matrices and tensors in deep learning frameworks * Visualization of masks, bounding boxes, and class predictions   Overall, the practical demonstrated the effectiveness of Mask R-CNN for accurate object detection and pixel-level segmentation, which is widely used in real-world applications such as autonomous driving, medical imaging, surveillance, and robotics. | | | |
| Date | | **28-02-2026** | | | |