OBJECT DETECTION

APPLICATION OF DIGITAL IMAGE PROCESSING

AGENDA

01
Artifical Intelligence

Breif Introduction to AI

02
Digital Image Processing

What is Digital Image Processing?

03
Applications Of DIP

Various ways DIP is implemented in

04
Object Detection

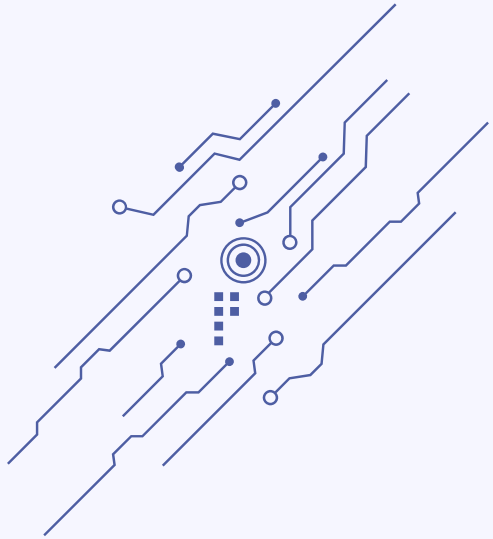
Recognition of objects in images and videos

01

ARTIFICAL INTELLIGENCE

WHAT IS ARTIFICIAL INTELLIGENCE?

Artificial intelligence (AI) refers to the simulation of human intelligence in machines that are programmed to think like humans and mimic their actions.



APPLICATIONS OF AI

- advanced web search engines (e.g., Google)
- recommendation systems (used by YouTube, Amazon and Netflix)
- understanding human speech (such as Siri and Alexa)
- self-driving cars (e.g., Tesla)
- automated decision-making,
- image processing (facial recognition used by smartphones)

02

DIGITAL IMAGE PROCESSING

WHAT IS DIGITAL IMAGE PROCESSING?

Digital Image Processing is the use of a digital computer to process digital images through an algorithm.

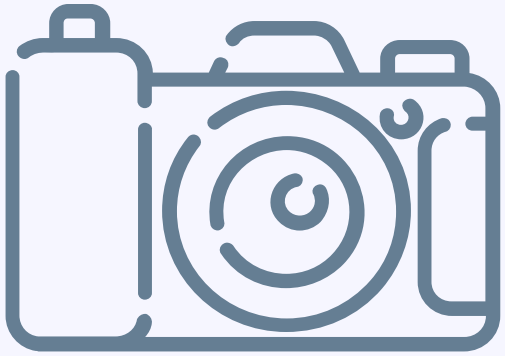
It is the manipulation of an image in order to enhance it or extract information from it.

WHAT IS DIGITAL IMAGE PROCESSING?

Machines can be taught to interpret images the same way our brains do and to analyze those images much more thoroughly than we can.

Image processing with artificial intelligence can power face recognition for detecting and recognizing objects and patterns in images and videos, and so on.

The input is an image, however, the output may be an image or information associated with that image, such as data on features, characteristics, bounding boxes, or masks.

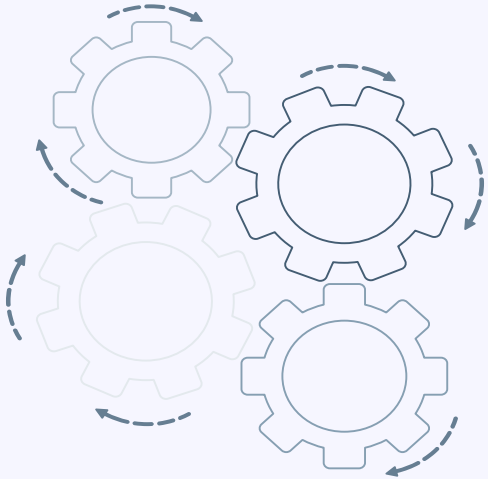


03

APPLICATIONS OF DIP

APPLICATIONS OF DIP

Image processing is serving a key role in many industries like medical, computer vision, and AI.



Almost in every field, digital image processing puts a live effect on things and is growing with time to time and with new technologies.

Today, image processing is widely used in medical visualization, biometrics, self-driving vehicles, gaming, surveillance, law enforcement, and other spheres.

APPLICATIONS OF DIP

01

Visualization

Giving visual form to objects that aren't visible

02

Image Sharpening and Restoration

Improve the quality of processed images

03

Image Retrieval

Help with image search

04

Object Measurement

Measure distance of objects in an image

APPLICATIONS OF DIP

05

Pattern Recognition

Distinguish and classify objects in an image

06

Object Detection

Recognizing objects which comes in way of something

07

Vision of Machines and Robot

Enhancing the vision ability by making different variation in its processing

08

Remote Sensing Field

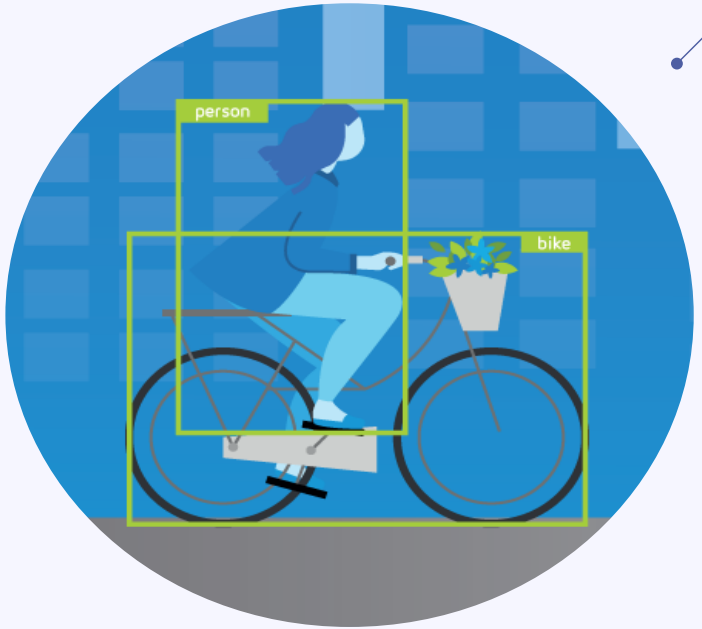
Analysis of Earth's surface using satellites

04

OBJECT DETECTION

LET'S TAKE A LOOK INTO WHAT OBJECT DETECTION IS

Detection of Obstacles or also known as, Hurdle detection, it is one of the common application that have been done through image processing.



Object Detection is a computer technology related to image processing that deals with detecting instances of semantic objects of a certain class (such as humans, buildings, or cars) in digital images and videos.

MORE ABOUT OBJECT DETECTION

Object detection algorithms typically leverage machine learning or deep learning to produce meaningful results.

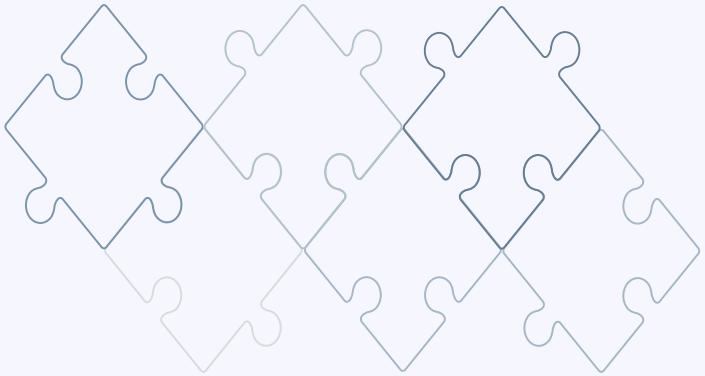
When humans look at images or video, we can recognize and locate objects of interest within a matter of moments.

The goal of object detection is to replicate this intelligence using a computer.

Well-researched domains of object detection include face detection, vehicle tracking and pedestrian detection.

IMPORTANCE OF OBJECT DETECTION

Object detection is a key technology behind advanced driver assistance systems (ADAS) that enable cars to detect driving lanes or perform pedestrian detection to improve road safety. Object detection is also useful in applications such as video surveillance or image retrieval systems.



HOW OBJECT DETECTION WORKS?

Every object class has its own special features that helps in classifying the class – for example all circles are round. Object class detection uses these special features.

For example, when looking for circles, objects that are at a particular distance from a point (i.e. the center) are sought. Similarly, when looking for squares, objects that are perpendicular at corners and have equal side lengths are needed.

A similar approach is used for face identification where eyes, nose, and lips can be found and features like skin color and distance between eyes can be found.

HOW OBJECT DETECTION WORKS?

Object detection can be performed using image processing techniques where they generally don't require historical data for training and are unsupervised in nature. OpenCV is a popular tool for image processing tasks.

Pro's: Hence, those tasks do not require annotated images, where humans labeled data manually (for supervised training).

Con's: These techniques are restricted to multiple factors, such as complex scenarios (without unicolor background), occlusion (partially hidden objects), illumination and shadows, and clutter effect.

USES OF OBJECT DETECTION

It is widely used in computer vision tasks such as:

- **image annotation**
- **vehicle counting**
- **activity recognition**
- **face detection**
- **face recognition**
- **video object co-segmentation**

It is also used in tracking objects, for example tracking a ball during a football match, tracking movement of a cricket bat, or tracking a person.

APPLICATIONS OF OBJECT DTECTECTION

People Detection in Security

Animal Detection in Agriculture

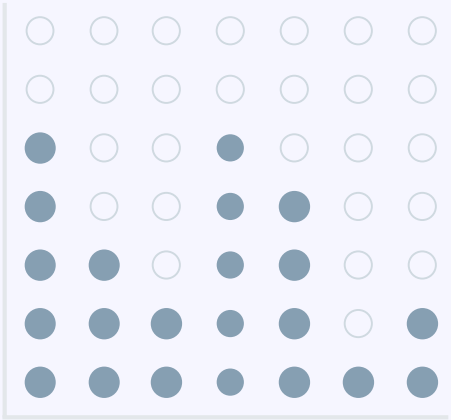
Object Detection in Retail

Autonomous Driving

Now, Let's Implement An Example Of OBJECT DETECTION

PARKING SPACE DETECTION!

OPENCV PROJECT FOR PARKING SPACE DETECTION



With this project, we will create a Parking Space Detection and Counter program where we can see how many total cars are present and how many spaces are vacant to park using basic Image Processing techniques.

REQUIRED PACKAGES ON PYCHARM

01

cv2

open-source library that can be used to perform tasks like face detection, objection tracking etc

02

pickle

used for serializing and de-serializing python object structures and converting them into byte streams

03

cvzone

computer vision package that makes us easy to run face detection, hand tracking, etc., and other AI functions

04

numpy

a Python library used for working with arrays.

CODE: main.py

```python
import cv2
import pickle
import cvzone
import numpy as np

# Video feed
cap = cv2.VideoCapture('carPark.mp4')

with open('CarParkPos', 'rb') as f:
    posList = pickle.load(f)

width, height = 107, 48


def checkParkingSpace(imgPro):
    spaceCounter = 0

    for pos in posList:
        x, y = pos
```

CODE: main.py

```
21
22          imgCrop = imgPro[y:y + height, x:x + width]
23          # cv2.imshow(str(x * y), imgCrop)
24          count = cv2.countNonZero(imgCrop)
25
26          if count < 900:
27              color = (0, 255, 0)
28              thickness = 5
29              spaceCounter += 1
30          else:
31              color = (0, 0, 255)
32              thickness = 2
33
34          cv2.rectangle(img, pos, (pos[0] + width, pos[1] + height), color, thickness)
35          cvzone.putTextRect(img, str(count), (x, y + height - 3), scale=1,
36                             thickness=2, offset=0, colorR=color)
37
38      cvzone.putTextRect(img, f'Free: {spaceCounter}/{len(posList)}', (100, 50), scale=3,
39                         thickness=5, offset=20, colorR=(0, 200, 0))
40
```

Code: main.py

```
41
42  while True:
43
44      if cap.get(cv2.CAP_PROP_POS_FRAMES) == cap.get(cv2.CAP_PROP_FRAME_COUNT):
45          cap.set(cv2.CAP_PROP_POS_FRAMES, 0)
46      success, img = cap.read()
47      imgGray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
48      imgBlur = cv2.GaussianBlur(imgGray, (3, 3), 1)
49      imgThreshold = cv2.adaptiveThreshold(imgBlur, 255, cv2.ADAPTIVE_THRESH_GAUSSIAN_C,
50                                           cv2.THRESH_BINARY_INV, 25, 16)
51      imgMedian = cv2.medianBlur(imgThreshold, 5)
52      kernel = np.ones((3, 3), np.uint8)
53      imgDilate = cv2.dilate(imgMedian, kernel, iterations=1)
54
55      checkParkingSpace(imgDilate)
56      cv2.imshow("Image", img)
57      # cv2.imshow("ImageBlur", imgBlur)
58      # cv2.imshow("ImageThres", imgMedian)
59      cv2.waitKey(10)
```

OUTPUT

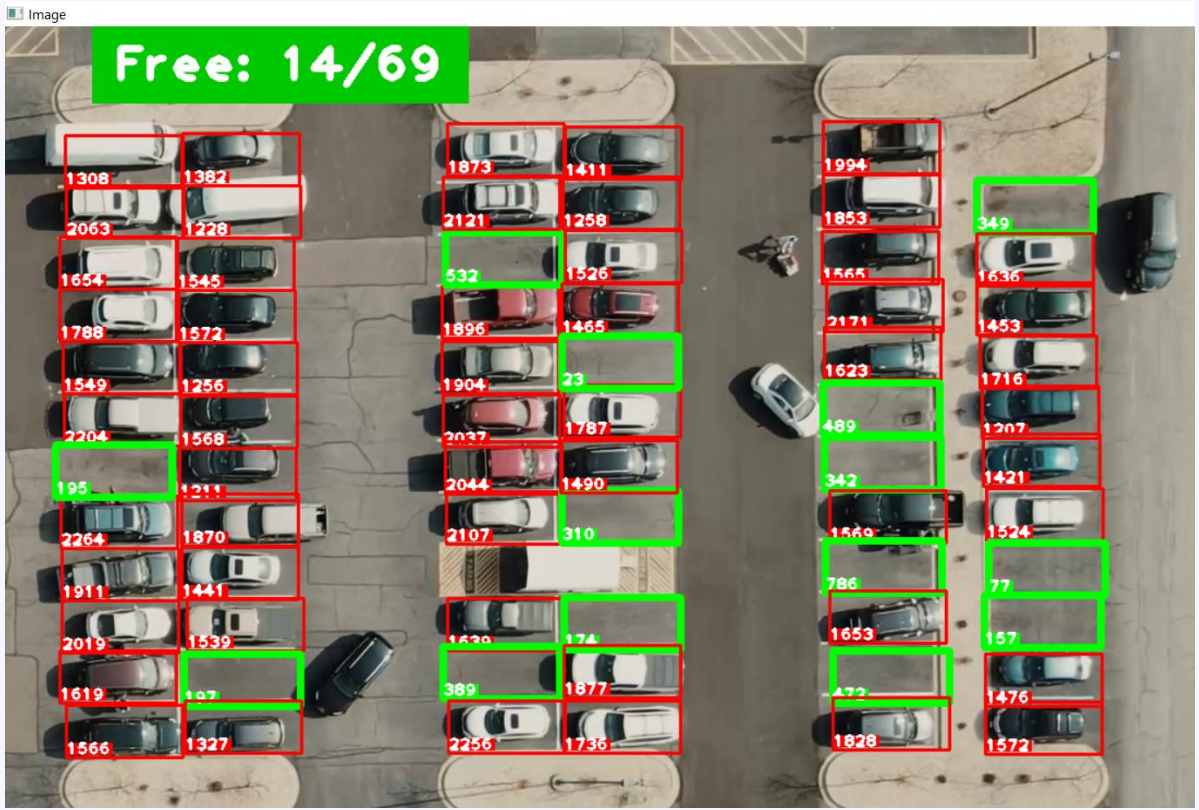
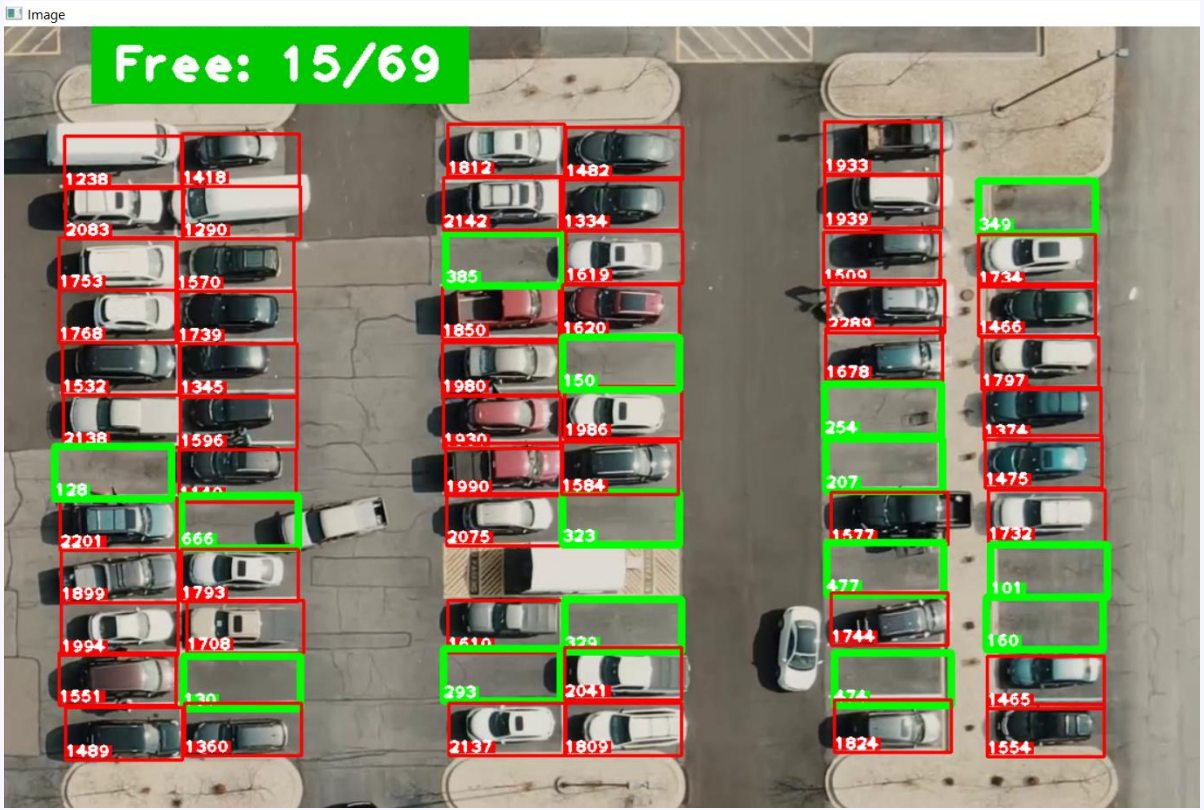


Image
Free: 14/69
1308
1382
2063
1228
1654
1545
1788
1572
1549
1256
2204
1568
195
2264
1870
1911
1441
2019
1539
1619
1566
1327
1873
1411
2121
1258
532
1526
1896
1465
1904
23
1787
2044
1490
2107
310
174
389
1877
2256
1736
1994
1853
1565
2171
1623
489
342
1569
786
1653
1828
1572
349
1636
1453
1716
1207
1421
1524
77
157
1476

OUTPUT



And thus, this is how **Object Detecting** using **Digital Image Processing** would help you find an empty parking space next time you go out!

the end,
thankyou !

presentation by:

saachi shinde
bsc cs hons. aiml
roll no.: 12