A Laboratory Manual for

Computer Vision
(3171614)

B.E. Semester7

L D College of Engineering, Ahmedabad

Directorate of Technical Education, Gandhinagar,Gujarat

L.D College of Engineering, Ahmedabad

Certificate

This is to certify thatMr./Ms. _____ _____ Enrollment No. _____ of B.E. Semester 7(Branch _____)of this Institute (GTU Code: _____) has satisfactorily completed the Practical / Tutorial work for the subject**Computer Vison(3171614)** for the academic year 2022-23.

Place: _____

Date: _____

Name and Sign of Faculty member

Head ofthe Department

Preface

Main motto of any laboratory/practical/field work is for enhancing required skills as well as creating ability amongst students to solve real time problem by developing relevant competencies in psychomotor domain.By keeping in view, GTU has designed competency focused outcome-based curriculum for engineering degree programs where sufficient weightage is given to practical work. It shows importance of enhancement of skills amongst the students and it pays attention to utilize every second of time allotted for practical amongst students, instructors and faculty members to achieve relevant outcomes by performing the experiments rather than having merely study type experiments. It is must for effective implementation of competency focused outcome-basedcurriculum that every practical is keenly designed to serve as a tool to develop and enhance relevant competency required by the various industry among every student. These psychomotor skills are very difficult to develop through traditional chalk and board content delivery method in the classroom. Accordingly, this lab manual is designed to focus on the industry defined relevant outcomes, rather than old practice of conducting practical to prove concept and theory.

By using this lab manual students can go through the relevant theory and procedure in advance before the actual performance which createsan interest and students can have basic idea prior to performance.This in turn enhances pre-determined outcomes amongst students.Each experiment in this manual begins with competency, industry relevant skills, course outcomes as well as practical outcomes (objectives). The students will also achieve safety and necessary precautions to be taken while performing practical.

This manual also provides guidelines to faculty members to facilitate studentcentric lab activities through each experiment by arranging and managing necessary resources in order that the students follow the procedures with required safety and necessary precautions to achieve the outcomes. It also gives an idea that how students will be assessed by providing rubrics.

Computer vision is a professional elective course which deals with principles of image formation, image processing algorithms and recognition from single or multiple images (video). This course emphasizes the core vision tasks of scene understanding and recognition. Applications to object recognition, image analysis, image retrieval and object tracking will be discussed.

Utmost care has been taken while preparing this lab manual however always there is chances of improvement. Therefore, we welcome constructive suggestions for improvement and removal of errors if any.

Practical – Course Outcome matrix

Course Outcomes (COs):

1. **Learn fundamentals of computer vision and its applications**
2. **Understand the basic image processing operations to enhance, segment the images.**
3. **Understand the analyzing and extraction of relevant features of the concerned domain problem.**
4. **Understand and apply the motion concepts and its relevance in real time applications**
5. **Apply the knowledge in solving high level vision problems like object recognition, image classification etc.**

Sr. No.	Objective(s) of Experiment	CO 1	CO 2	CO 3	CO 4	CO 5
1.	Implementing various basic image processing operations in python/MATLAB/open-CV: Reading image, writing image, conversion of images, and complement of an image	√				
2.	Implement contrast adjustment of an image. Implement Histogram processing and equalization.		√			
3.	Implement the various low pass and high pass filtering mechanisms.			√		
4.	Use of Fourier transform for filtering the image.		√			
5.	Utilization of SIFT and HOG features for image analysis.			√		
6.	Performing/Implementing image segmentation		√			
7.	Implement optical flow computation algorithm.				√	
8.	Demonstrate the use of optical flow in any image processing application				√	
9.	Object detection and Recognition on available online image datasets					√
10.	Character or digit or face classification project					√

Industry Relevant Skills

The following industry relevant competencies are expected to be developed in the student by undertaking the practical work of this laboratory.

1. Will be able to solve open design problems
2. Will be able to apply the knowledge, techniques, skills and modern tools to become successful professionals in computer vision industries.

Guidelines for Faculty members

1. Teacher should provide the guideline with demonstration of practical to the students with all features.
2. Teacher shall explain basic concepts/theory related to the experiment to the students before starting of each practical
3. Involve all the students in performance of each experiment.
4. Teacher is expected to share the skills and competencies to be developed in the students and ensure that the respective skills and competencies are developed in the students after the completion of the experimentation.
5. Teachers should give opportunity to students for hands-on experience after the demonstration.
6. Teacher may provide additional knowledge and skills to the students even though not covered in the manual but are expected from the students by concerned industry.
7. Give practical assignment and assess the performance of students based on task assigned to check whether it is as per the instructions or not.
8. Teacher is expected to refer complete curriculum of the course and follow the guidelines for implementation.

Instructions for Students

1. Students are expected to carefully listen to all the theory classes delivered by the faculty members and understand the COs, content of the course, teaching and examination scheme, skill set to be developed etc.
2. Students shall organize the work in the group and make record of all observations.
3. Students shall develop maintenance skill as expected by industries.
4. Student shall attempt to develop related hand-on skills and build confidence.
5. Student shall develop the habits of evolving more ideas, innovations, skills etc. apart from those included in scope of manual.
6. Student shall refer technical magazines and data books.
7. Student should develop a habit of submitting the experimentation work as per the schedule and s/he should be well prepared for the same.

Common Safety Instructions

1) Switch on the PC carefully (not to use wet hands)
2) Shutdown the PC properly at the end of your Lab
3) Carefully Handle the peripherals (Mouse, Keyboard, Network cable etc)
4) Use Laptop in lab after getting permission from Teacher

Index
(Progressive Assessment Sheet)

Sr. No.	Objective(s) of Experiment	Page No.	Date of performance	Date of submission	Assessment Marks	Sign. of Teacher with date	Remarks
1.	Implementing various basic image processing operations in python/MATLAB/open-CV: Reading image, writing image, conversion of images, and complement of an image						
2.	Implement contrast adjustment of an image. Implement Histogram processing and equalization.						
3.	Implement the various low pass and high pass filtering mechanisms.						
4.	Use of Fourier transform for filtering the image.						
5.	Utilization of SIFT and HOG features for image analysis.						
6.	Performing/Implementing image segmentation						
7.	Implement optical flow computation algorithm.						
8.	Demonstrate the use of optical flow in any image processing application						
9.	Object detection and Recognition on available online image datasets						
10.	Character or digit or face classification project						
Total							

Experiment No: 1

Implementing various basic image processing operations in python/ MATLAB/ open-CV: Reading image, writing image, conversion of images, and complement of an image

Date:

Competency and Practical Skills:

This practical is expected to develop following skills in you

1. Reading an image.
2. Writing an image.
3. Conversion of an image.
4. Complement of an image.

Relevant CO: CO1

Objectives:

1. To understand basic image processing operations.
2. To implement basic image processing operations.

Equipment/Instruments: Computer systems equipped with python/matlab/open-CV

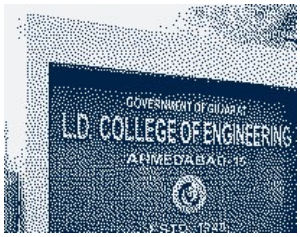
Theory:

Read Image: Function to read image essentially takes the grey values of all the pixels in the greyscale image and puts them all into a matrix. This matrix now becomes a variable of the programming platform we use namely Matlab, Python or Open-CV. Size of such matrix for a greyscale image will be MxN. In case of a color image with RGB (Red, Green, Blue) color palette the size of the matrix becomes 3 * (MxN). Here MxN is the resolution of the image.In general, the read function reads the pixel values from an image file, andreturns a matrix of all the pixel values.

Write Image: Once we have captured image data i.e matrix with MxN resolution either by digitally capturing it, extracting it from a video sequence or by processing any input image, we would intend to save the image on computer i.e writing the image. Write function will enable us to write this data from the matrix variable onto the hard disk at desired location and in corresponding file format as the matrix. A MxN matrix can generate a greyscale image while a 3 * (MxN) matrix contains data from R, G and B color palettes can generate a color image.

Image Conversion: Following are the types of digital images:

1. Binary images
2. Greyscale images
3. RGB images



(a) Binary Image

(b) Greyscale Image

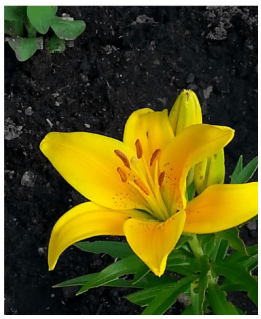
(c) RGB Image

Figure 1.1: Types of digital images

Binary as the name suggests is the image with either black or white pixels. Greyscale images have pixel values from 0 (black) to 255 (white). RGB images are the true color images with value between 0 to 255 for each of Red, Green and Blue components. Within the limitations of arithmetic conversions, we can convert images from one image type to another. RGB to Grey and Grey to RGB are examples of such image conversions.

Image Complement: In the complement of a binary image, zeros become ones and ones become zeros. Black and white are reversed. In the complement of a grayscale or color image, each pixel value is subtracted from the maximum pixel value supported by the class (or 1.0 for double-precision images). The difference is used as the pixel value in the output image. In the output image, dark areas become lighter and light areas become darker. For color images, reds become cyan, greens become magenta, blues become yellow, and vice versa.

(a) Original and Complement of an Intensity Image



(b) Original and Complement of a Color Image

Figure 1.2: Image Complement

Safety and necessary Precautions:

1. Do not alter the installed libraries of python/matlab/open-CV

Procedure:

1. Image read :
   matrix variable = read image (From_Location)
   Display the image
2. Image Write: write image(To_Location, matrix variable)
3. Image conversion:
   y=rgb2gray(x);
   y=gray2rgb(x)
4. Complement of Image:
   for each value of x in the image
   y= 255-x
   save y

Program:

// Write code of your program here

Output:

// Write output of your program here

Conclusion:

// Write conclusion here

Quiz:

1. If you have access to a digital cameracapable of capturing images with 1024x768 resolution for a fixed scene, using all possiblecamera settings what is the smallest file you can create?
2. Will it be possible to convert an original greyscale image to rgb ?

Suggested Reference:

1. Digital Image Processingby S. Sridhar. Oxford Press.
2. https://www.mathworks.com/help/matlab/ref/imwrite.html

References used by the students:

Rubric wise marks obtained:

Program	**(Excellent)(4)**	**(Good)(3)**	**(Fair)(2)**	**(Beginning)(1)**
Program execution	Program executes correctly with no syntax or runtime errors	Program executes with a minor error	Program executes with multiple minor (easily fixed error)	Program does not execute (0-1)
Design-Correctness of output	Program displays correct output with no errors	Output/design of output has minor errors	Output/Design of output has multiple errors	Output is incorrect (0-1)
Design of logic	Program is logically well designed	Program has slight logic errors that do no significantly affect the results	Program has significant logic errors	Program is incorrect (0-1)
Standards	Program is stylistically well designed	Few inappropriate design choices (i.e. poor variable names, improper indentation)	Several inappropriate design choices (i.e. poor variable names, improper indentation)	Program is poorly written (0-1)
Documentation	Program is well documented	Missing one required comment	Missing two or more required comments	Most or all documentation missing (0-1)

Criteria	**1**	**2**	**3**	**4**	**5**	**Total**
Marks						

Experiment No: 2

Implement contrast adjustment of an image. Implement Histogram processing and equalization

Date:

Competency and Practical Skills:

This practical is expected to develop following skills in you

1. Image Enhancement by adjusting image contrast
2. Image analysis using Histogram

Relevant CO: CO2

Objectives:

1. Image enhancement using contrast adjustment and histogram equalization

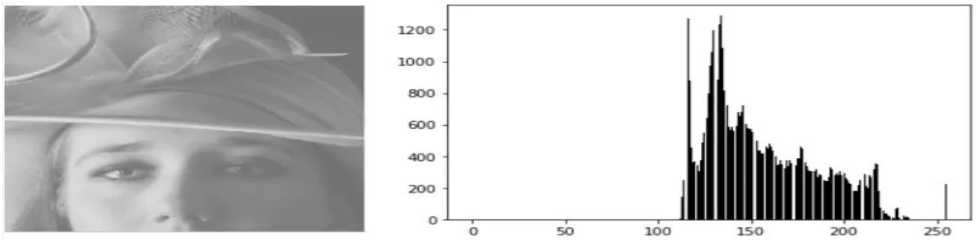
Equipment/Instruments: Computer systems equipped with python/ matlab/ open-CV

Theory:

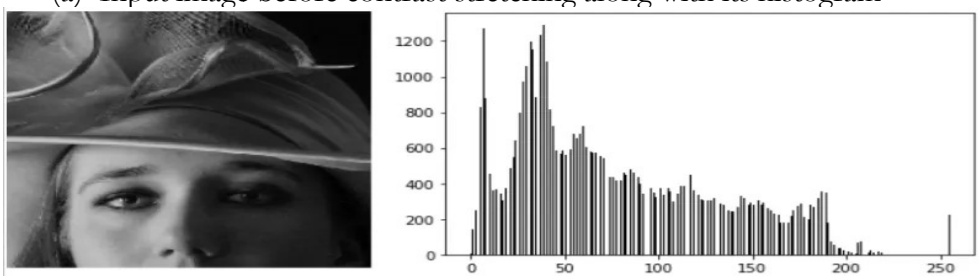
Contrast stretching: It is an image enhancement technique that tries to improve the contrast by stretching the intensity values of an image to fill the entire dynamic range. The transformation function used is always linear and monotonically increasing.If the minimum intensity value(r_{min}) present in the image is 100 then it is stretched to the possible minimum intensity value 0. Likewise, if the maximum intensity value(r_{max}) is less than the possible maximum intensity value 255 then it is stretched out to 255.0–255 is taken as standard minimum and maximum intensity values for 8-bit images. General Formula for Contrast Stretching is given by equation (2.1).

$$s = (r - r_{min})\frac{(I_{max} - I_{\min})}{(r_{max} - r_{\min})} + I_{min} \qquad \text{eq. (2.1)}$$

where,r = current pixel intensity value ;r_{min} = minimum intensity value present in the whole image ; r_{max} = maximum intensity value present in the whole image. I are the intended values and s is the resultant value of the intensity.



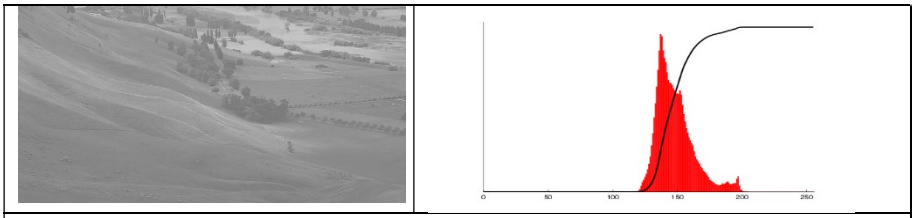
(a) Input image before contrast stretching along with its histogram



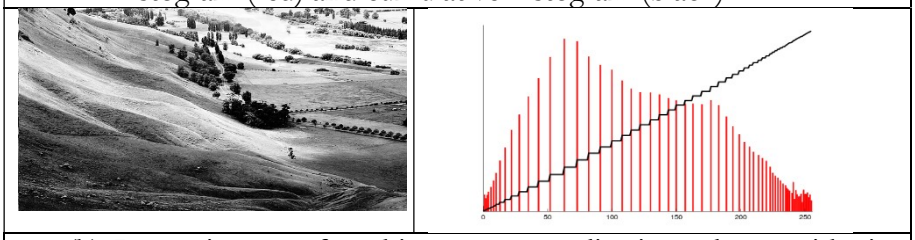
(b) Input image after contrast stretching along with its histogram

Figure 2.1: Results of contrast stretching

Histogram Equalization: It is an automatic enhancement technique which produces an output (enhanced) image that has a near uniformly distributed histogram**.** The idea is to change the histogram to one which is uniform; that is that every bar onthe histogram is of the same height, or in other words that each grey level in the image occurs withthe same frequency. In practice this is generally not possible, although we shall see that the result ofhistogram equalization provides very good results.Through this adjustment, the intensities can be better distributed on the histogram utilizing the full range of intensities evenly. This allows for areas of lower local contrast to gain a higher contrast. Histogram equalization accomplishes this by effectively spreading out the highly populated intensity values which are used to degrade image contrast.The method is useful in images with backgrounds and foregrounds that are both bright or both dark.



(a) Input image before histogram equalization along with its histogram (red) and cumulative histogram (black)



(b) Input image after histogram equalization along with its histogram (red) and cumulative histogram (black)

Figure 2.2: Results of Histogram Equalization

Procedure in Matlab:

Contrast stretching:

```
I = imread('<input image>');
figure
imshow(I)
J = imadjust(I,stretchlim(I),[]);
figure
imshow(J)
```

Histogram Equalization:

```
I = imread('<input image>');
figure
subplot(1,3,1)
imshow(I)
subplot(1,3,2:3)
imhist(I)
J = histeq(I);
figure
subplot(1,3,1)
imshow(J)
subplot(1,3,2:3)
imhist(J)
```

Program:

// Write code of your program here

Output:

// Write output of your program here

Conclusion:

// Write conclusion here

Quiz:

1. Differentiate between contrast stretching and histogram equalization
2. Is it possible to re-tract to original image after in both contrast stretching and histogram equalization

Suggested Reference:

1.Digital Image Processingby S. Sridhar. Oxford Press.
2.https://uotechnology.edu.iq/ce/lecture%202013n/4th%20Image%20Processing%20_Lectures/DIP_Lecture5.pdf
3. https://en.wikipedia.org/wiki/Histogram_equalization

References used by the students:

Rubric wise marks obtained:

Program	**(Excellent)(4)**	**(Good)(3)**	**(Fair)(2)**	**(Beginning)(1)**
Program execution	Program executes correctly with no syntax or runtime errors	Program executes with a minor error	Program executes with multiple minor (easily fixed error)	Program does not execute (0-1)
Design-Correctness of output	Program displays correct output with no errors	Output/design of output has minor errors	Output/Design of output has multiple errors	Output is incorrect (0-1)
Design of logic	Program is logically well designed	Program has slight logic errors that do no significantly affect the results	Program has significant logic errors	Program is incorrect (0-1)
Standards	Program is stylistically well designed	Few inappropriate design choices (i.e. poor variable names, improper indentation)	Several inappropriate design choices (i.e. poor variable names, improper indentation)	Program is poorly written (0-1)
Documentation	Program is well documented	Missing one required comment	Missing two or more required comments	Most or all documentation missing (0-1)

Criteria	**1**	**2**	**3**	**4**	**5**	**Total**
Marks						

Experiment No: 3

Implement the various low pass and high pass filtering mechanisms.

Date:

Competency and Practical Skills:

This practical is expected to develop following skills in you

1. Image filtering using low pass filters
2. Image filtering using high pass filters

Relevant CO: CO3

Objectives:

1. Image enhancement such as smoothing, sharpening and edge enhancement using various filters.

Equipment/Instruments: Computer systems equipped with python/ matlab/ open-CV

Theory:

Filtering is a technique for modifying or enhancing an image. For example, you can filter an image to emphasize certain features or remove other features. Image processing operations implemented with filtering include smoothing, sharpening, and edge enhancement.

Low pass filter (smoothing): Low pass filter is the type of frequency domain filter that is used for smoothing the image. It attenuates the high-frequency components and preserves the low-frequency components. High frequency content corresponds to boundaries of the objects. An image is smoothed by decreasing the disparity between pixel values by averaging nearby pixels. The low-pass filters usually employ moving window operator which affects one pixel of the image at a time, changing its value by some function of a local region (window) of pixels. The operator moves over the image to affect all the pixels in the image.

Mean filtering: It is used as a method of smoothing images, reducing the amount of intensity variation between one pixel and the next resulting in reducing noise in images. The idea of mean filtering is simply to replace each pixel value in an image with the mean (average) value of its neighbors, including itself. This has the effect of eliminating pixel values which are unrepresentative of their surroundings.

Median filter: Median filtering is a nonlinear operation often used in image processing to reduce "salt and pepper" noise. Median filter replaces the pixel at the center of the filter with the median value of the pixels falling beneath the mask. Median filter does not blur the image but it rounds the corners.

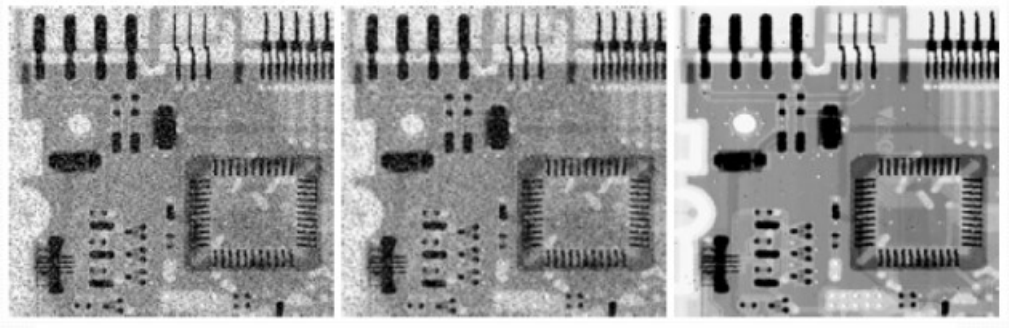


Figure 3.1: Original image, mean filtered output and median filtered output in the order of left to right

High pass filter (sharpening and edge enhancement): High pass filter is the type of frequency domain filter that is used for sharpening the image. It attenuates the low-frequency components and preserves the high-frequency components. A high-pass filter can be used to make an image appear sharper. These filters emphasize fine details in the image - the opposite of the low-pass filter. High-pass filtering works in the same way as low-pass filtering; it just uses a different convolution kernel. Prewitt and Sobel are derivative filters used as edge detectors.

Laplacian filter: One of the most known high-pass filters is the Laplacian edge enhancement. Its meaning can be thus understood: We subtract the image from a blurred version of itself created from the averaging of the four nearest neighbours. This enhances edges and isolated pixels with extreme values. This method being very sensitive to noise, Laplacian of Gaussian (LoG) is used.

Procedure in Matlab:

Low Pass Filters:

```
I = imread('<input image>');
h = 1/3*ones(3,1);
H = h*h';
imfilt = filter2(H,I); // Mean filter for 3x3
J = medfilt2(I) // Median filter
```

High Pass Filters:

```
I = imread('<input image>');
BW1 = edge(I,'Sobel');
BW2 = edge(I,'Prewitt');
BW3 = edge(I,'Canny');
BW4 = edge(I,'log');
imshowpair(BW1,BW2,'montage')
imshowpair(BW3,BW4,'montage')
```

Program:

// Write code of your program here

Output:

// Write output of your program here

Conclusion:

// Write conclusion here

Quiz:

1. Compare low pass filters and high pass filters
2. For noise removal which type of filter will be used?
3. It is necessary to use Gaussian smoothing before using Laplacian filter. Justify.

Suggested Reference:

1.Digital Image Processing by S. Sridhar. Oxford Press.
2.https://rpg.ifi.uzh.ch/docs/teaching/2017/04_filtering.pdf
3.https://www.bogotobogo.com/Matlab/Matlab_Tutorial_Digital_Image_Processing_6_Filter_Smoothing_Low_Pass_fspecial_filter2.php

References used by the students:

Rubric wise marks obtained:

Program	**(Excellent)(4)**	**(Good)(3)**	**(Fair)(2)**	**(Beginning)(1)**
Program execution	Program executes correctly with no syntax or runtime errors	Program executes with a minor error	Program executes with multiple minor (easily fixed error)	Program does not execute (0-1)
Design-Correctness of output	Program displays correct output with no errors	Output/design of output has minor errors	Output/Design of output has multiple errors	Output is incorrect (0-1)
Design of logic	Program is logically well designed	Program has slight logic errors that do no significantly affect the results	Program has significant logic errors	Program is incorrect (0-1)
Standards	Program is stylistically well designed	Few inappropriate design choices (i.e. poor variable names, improper indentation)	Several inappropriate design choices (i.e. poor variable names, improper indentation)	Program is poorly written (0-1)
Documentation	Program is well documented	Missing one required comment	Missing two or more required comments	Most or all documentation missing (0-1)

Criteria	**1**	**2**	**3**	**4**	**5**	**Total**
Marks						

Experiment No: 4

Use of Fourier transform for filtering the image.

Date:

Competency and Practical Skills:

This practical is expected to develop following skills in you

1. Understanding Discrete Fourier Transform
2. Implementation of Fourier Transform in digital image

Relevant CO: CO2

Objectives:

1. Use of Fourier transform in digital image processing

Equipment/Instruments: Computer systems equipped with python/ matlab/ open-CV

Theory:

Fourier Transform is an important image processing tool which is used to decompose an image into its sine and cosine components. The output of the transformation represents the image in the *Fourier* or frequency domain, while the input image is the spatial domain equivalent. In the Fourier domain image, each point represents a particular frequency contained in the spatial domain image. The Fourier Transform is used in a wide range of applications, such as image analysis, image filtering, image reconstruction and image compression.

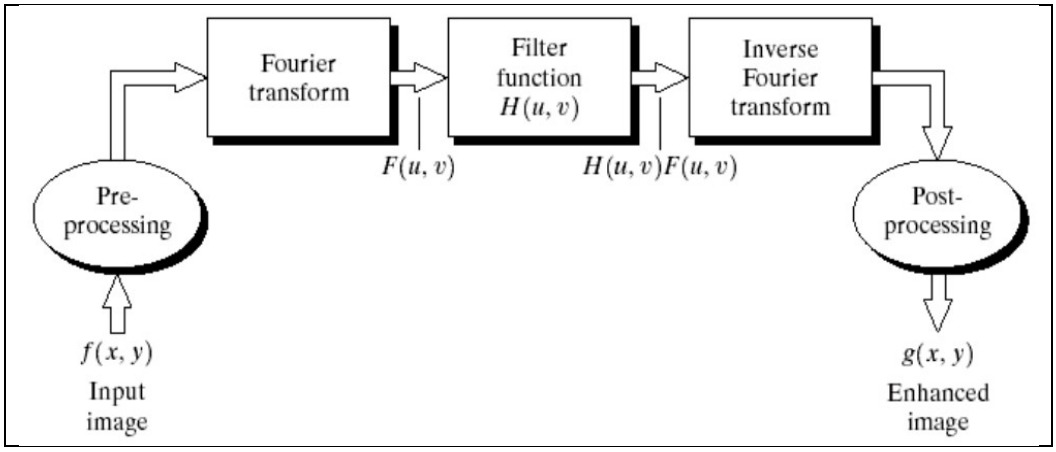


Figure 3.1: Frequency domain filtering

Procedure in Matlab:

1. fft which takes the DFT of a vector
2. ifft which takes the inverse DFT of a vector
3. fft2 which takes the DFT of a matrix
4. ifft2 which takes the inverse DFT of a matrix
5. fftshift which shifts a transform

Program:

// Write code of your program here

Output:

// Write output of your program here

Conclusion:

// Write conclusion here

Quiz:

1. Discuss Properties of Fourier Transform

Suggested Reference:

1.Digital Image Processingby S. Sridhar. Oxford Press.
2. https://www.ece.mcmaster.ca/~shirani/ip12/chapter4
3. https://vincmazet.github.io/bip/filtering/fourier.html

References used by the students:

Rubric wise marks obtained:

Program	**(Excellent)(4)**	**(Good)(3)**	**(Fair)(2)**	**(Beginning)(1)**
Program execution	Program executes correctly with no syntax or runtime errors	Program executes with a minor error	Program executes with multiple minor (easily fixed error)	Program does not execute (0-1)
Design-Correctness of output	Program displays correct output with no errors	Output/design of output has minor errors	Output/Design of output has multiple errors	Output is incorrect (0-1)
Design of logic	Program is logically well designed	Program has slight logic errors that do no significantly affect the results	Program has significant logic errors	Program is incorrect (0-1)
Standards	Program is stylistically well designed	Few inappropriate design choices (i.e. poor variable names, improper indentation)	Several inappropriate design choices (i.e. poor variable names, improper indentation)	Program is poorly written (0-1)
Documentation	Program is well documented	Missing one required comment	Missing two or more required comments	Most or all documentation missing (0-1)

Criteria	**1**	**2**	**3**	**4**	**5**	**Total**
Marks						

Experiment No: 5

Utilization of SIFT and HOG features for image analysis.

Date:

Competency and Practical Skills:

This practical is expected to develop following skills in you

1. Understanding SIFT and HOG features
2. Implementation of SIFT and HOG features

Relevant CO: CO3

Objectives:

1. Use of SIFT and HOG feature extraction in digital image processing

Equipment/Instruments: Computer systems equipped with python/ matlab/ open-CV

Theory:

Scale-Invariant Feature Transform (SIFT): SIFT is invariance to image scale and rotation. In general, SIFT algorithm can be decomposed into four steps as (a) Feature point (also called keypoint) detection (b) Feature point localization (c) Orientation assignment and (d) Feature descriptor generation.

Major advantages of SIFT are

- **Locality:** features are local, so robust to occlusion and clutter (no prior segmentation)
- **Distinctiveness:** individual features can be matched to a large database of objects
- **Quantity:** many features can be generated for even small objects
- **Efficiency:** close to real-time performance
- **Extensibility:** can easily be extended to a wide range of different feature types, with each adding robustness

SIFT helps to reduce the dimensions of the feature space by removing the redundant features, which highly impact the training of the machine learning used in large scale applications

Figure 5.1: SIFT features visual

Histogram of Oriented Gradients (HOG): This feature descriptor is used for the purpose of object detection. The technique counts occurrences of gradient orientation in localized portions of an image. This method is similar to that of edge orientation histograms, scale-invariant feature transform

descriptors, and shape contexts, but differs in that it is computed on a dense grid of uniformly spaced cells and uses overlapping local contrast normalization for improved accuracy. The HOG feature vector is arranged by HOG blocks. The cell histogram, $H(C_{yx})$, is 1-by-NumBins. The figure below shows the HOG feature vector with a 1-by-1 cell overlap between blocks.

.

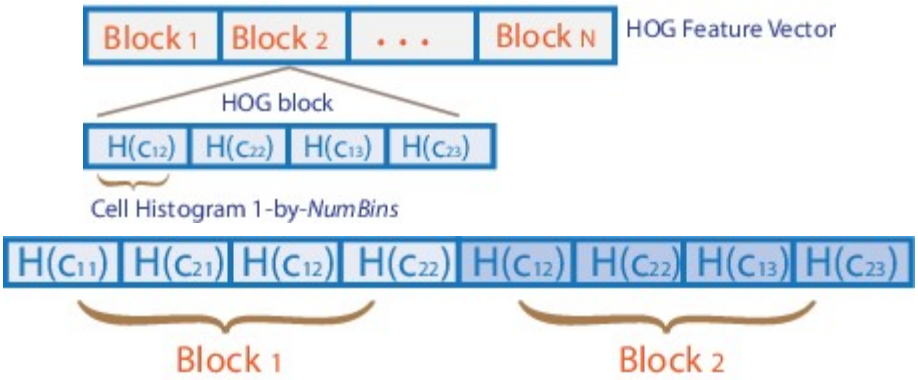


Figure 5.2: HOG Feature Vector

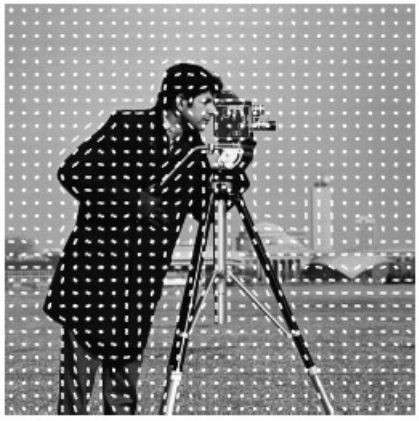


Figure 5.3: HOG features visual

Procedure in OpenCV:

SIFT:

```
img = imread('image_name')
imgGray = cv2.cvtColor(img,cv2.COLOR_BGR2GRAY)
sift = cv2.SIFT_create()
keypoints,descriptors = sift.detectAndCompute(img, None)
sift_image = cv2.drawKeypoints(imgGray, keypoints, img)
```

HOG:

```
img = cv2.imread('image_name')
(hog, hog_image) = feature.hog(img, orientations=9,pixels_per_cell = (8,8),
cells_per_block=(2,2),block_norm='L2-Hys', visualize=True, transform_sqrt=True)
cv2.imshow("Ori", img)
cv2.imshow('HOG IMAGE', hog_image)
```

Program:

// Write code of your program here

Output:

// Write output of your program here

Conclusion:

// Write conclusion here

Quiz:

1. Compare HOG and SIFT feature descriptors

Suggested Reference:

1.Digital Image Processing by S. Sridhar. Oxford Press.
2. https://in.mathworks.com/help/vision/ref/extracthogfeatures.html
3. https://towardsdatascience.com/hog-histogram-of-oriented-gradients-67ecd887675f

References used by the students:

Rubric wise marks obtained:

Program	**(Excellent)(4)**	**(Good)(3)**	**(Fair)(2)**	**(Beginning)(1)**
Program execution	Program executes correctly with no syntax or runtime errors	Program executes with a minor error	Program executes with multiple minor (easily fixed error)	Program does not execute (0-1)
Design-Correctness of output	Program displays correct output with no errors	Output/design of output has minor errors	Output/Design of output has multiple errors	Output is incorrect (0-1)
Design of logic	Program is logically well designed	Program has slight logic errors that do no significantly affect the results	Program has significant logic errors	Program is incorrect (0-1)
Standards	Program is stylistically well designed	Few inappropriate design choices (i.e. poor variable names, improper indentation)	Several inappropriate design choices (i.e. poor variable names, improper indentation)	Program is poorly written (0-1)
Documentation	Program is well documented	Missing one required comment	Missing two or more required comments	Most or all documentation missing (0-1)

Criteria	**1**	**2**	**3**	**4**	**5**	**Total**
Marks						

Experiment No: 6

Performing/Implementing image segmentation.

Date:

Competency and Practical Skills:

This practical is expected to develop following skills in you

1. Understanding Image Segmentation
2. Implementing Image Segmentation

Relevant CO: CO2

Objectives:

1. Use of segmentation in digital image processing

Equipment/Instruments: Computer systems equipped with python/ matlab/ open-CV

Theory:

Segmentation: Instead of processing the entire image, a common practice is to extract the Region of Interest (RoI). Image segmentation is a method of dividing a digital image into subgroups called image segments, reducing the complexity of the image and enabling further processing or analysis of each image segment. Technically, segmentation is the assignment of labels to pixels to identify objects, people, or other important elements in the image. Image segmentation could involve separating foreground from background, or clustering regions of pixels based on similarities in color or shape. For example, a common application of image segmentation in medical imaging is to detect and label pixels in an image or voxels of a 3D volume that represent a tumor in a patient's brain or other organs. Image segmentation is typically used to locate objects and boundaries (lines, curves, etc.) in images. Types of segmentation are as below:

Edge-Based Segmentation: This technique identifies the edges of various objects in a given image. It helps locate features of associated objects in the image using the information from the edges. Edge detection helps strip images of redundant information, reducing their size and facilitating analysis. Edge-based segmentation algorithms identify edges based on contrast, texture, color, and saturation variations. They can accurately represent the borders of objects in an image using edge chains comprising the individual edges.

Threshold Based: It is the simplest image segmentation method, dividing pixels based on their intensity relative to a given value or threshold. It is suitable for segmenting objects with higher intensity than other objects or backgrounds. The threshold value T can work as a constant in low-noise images. In some cases, it is possible to use dynamic thresholds.

Region-based Segmentation: It involves dividing an image into regions with similar characteristics. Each region is a group of pixels, which the algorithm locates via a seed point. Once the algorithm finds the seed points, it can grow regions by adding more pixels or shrinking and merging them with other points.

Once the mask is ready then the RoI can be segmented out of the given image with the help of the mask.

Procedure:

1. Create mask for desired RoI
2. Mask it with the original image
3. The resultant segments can further be used for post-processing

Program:

// Write code of your program here

Output:

// Write output of your program here

Conclusion:

// Write conclusion here

Quiz:

1. Discuss applications of different segmentation techniques

Suggested Reference:

1.Digital Image Processing by S. Sridhar. Oxford Press.
2. https://www.tensorflow.org/tutorials/images/segmentation

References used by the students:

Rubric wise marks obtained:

Program	**(Excellent)(4)**	**(Good)(3)**	**(Fair)(2)**	**(Beginning)(1)**
Program execution	Program executes correctly with no syntax or runtime errors	Program executes with a minor error	Program executes with multiple minor (easily fixed error)	Program does not execute (0-1)
Design-Correctness of output	Program displays correct output with no errors	Output/design of output has minor errors	Output/Design of output has multiple errors	Output is incorrect (0-1)
Design of logic	Program is logically well designed	Program has slight logic errors that do no significantly affect the results	Program has significant logic errors	Program is incorrect (0-1)
Standards	Program is stylistically well designed	Few inappropriate design choices (i.e. poor variable names, improper indentation)	Several inappropriate design choices (i.e. poor variable names, improper indentation)	Program is poorly written (0-1)
Documentation	Program is well documented	Missing one required comment	Missing two or more required comments	Most or all documentation missing (0-1)

Criteria	**1**	**2**	**3**	**4**	**5**	**Total**
Marks						

Experiment No: 7

Implement optical flow computation algorithm.

Date:

Competency and Practical Skills:

This practical is expected to develop following skills in you

1. Understanding optical flow computation algorithm
2. Implementation of optical flow computation algorithm

Relevant CO: CO4

Objectives:

1. Understanding motion estimation and implementing optical flow computation algorithm

Equipment/Instruments: Computer systems equipped with python/ matlab/ open-CV

Theory:

Optical flow: Itis the motion of objects between consecutive frames of sequence, caused by the relative movement between the object and camera. The problem of optical flow may be expressed by following figure:

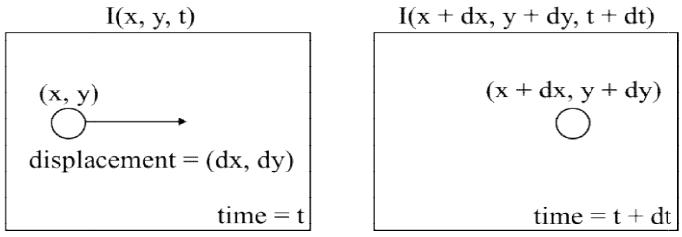


Figure 7.1: Optical flow

where between consecutive frames, we can express the image intensity (I) as a function of space (x,y) and time (t). In other words, if we take the first image I(x,y,t) and move its pixels by (dx,dy) over t time, we obtain the new image I(x+dx, y+dy, t+dt).

Optical flow works on following assumptions:

1. The pixel intensities of an object do not change between consecutive frames.
2. Neighbouring pixels have similar motion.

Differential methods of estimating optical flow, based on partial derivatives of the image signal and/or the sought flow field and higher-order partial derivatives, such as:

1. Lucas–Kanade method – regarding image patches and an affine model for the flow field
2. Horn–Schunck method – optimizing a functional based on residuals from the brightness constancy constraint, and a particular regularization term expressing the expected smoothness of the flow field
3. Buxton–Buxton method – based on a model of the motion of edges in image sequences
4. Black–Jepson method – coarse optical flow via correlation

General variational methods – a range of modifications/extensions of Horn–Schunck, using other data terms and other smoothness terms.

Procedure for Lucas-KanadeSparse Optical Flow method with OpenCV:

1. Setting up your environment and pen sparse-starter.py with your text editor
2. Configuring OpenCV to read a video and setting up parameters
3. Grayscaling
4. Shi-Tomasi Corner Detector - selecting the pixels to track
5. Tracking Specific Objects
6. Lucas-Kanade: Sparse Optical Flow
7. Visualizing

Program:

// Write code of your program here

Output:

// Write output of your program here

Conclusion:

// Write conclusion here

Quiz:

1. Compare Sparse vs. Dense Optical Flow

Suggested Reference:

1.The computation of optical flow by S. S. Beauchemin and J. L. Barron. ACM digital Library.
2. https://nanonets.com/blog/optical-flow/#what-is-optical-flow

References used by the students:

Rubric wise marks obtained:

Program	**(Excellent)(4)**	**(Good)(3)**	**(Fair)(2)**	**(Beginning)(1)**
Program execution	Program executes correctly with no syntax or runtime errors	Program executes with a minor error	Program executes with multiple minor (easily fixed error)	Program does not execute (0-1)
Design-Correctness of output	Program displays correct output with no errors	Output/design of output has minor errors	Output/Design of output has multiple errors	Output is incorrect (0-1)
Design of logic	Program is logically well designed	Program has slight logic errors that do no significantly affect the results	Program has significant logic errors	Program is incorrect (0-1)
Standards	Program is stylistically well designed	Few inappropriate design choices (i.e. poor variable names, improper indentation)	Several inappropriate design choices (i.e. poor variable names, improper indentation)	Program is poorly written (0-1)
Documentation	Program is well	Missing one	Missing two or	Most or all

	documented	required comment	more required comments	documentation missing (0-1)

Criteria	1	2	3	4	5	Total
Marks						

Experiment No: 8

Demonstrate the use of optical flow in any image processing application.

Date:

Competency and Practical Skills:

This practical is expected to develop following skills in you

1. Use of optical flow in any image processing application

Relevant CO: CO4

Objectives:

1. Applying optical flow in image processing

Equipment/Instruments: Computer systems equipped with python/ matlab/ open-CV

Theory:

Optical flow has many applications in areas like:

1. Structure from Motion
2. Video Compression
3. Video Stabilization

We can create a simple application which tracks some points in a video. To decide the points, we use **cv.goodFeaturesToTrack()**. We take the first frame, detect some Shi-Tomasi corner points in it, then we iteratively track those points using Lucas-Kanade optical flow. For the function **cv.calcOpticalFlowPyrLK()** we pass the previous frame, previous points and next frame. It returns next points along with some status numbers which has a value of 1 if next point is found, else zero. We iteratively pass these next points as previous points in next step.

Figure 8.1: Demonstration output

Procedure with OpenCV:

```
#include <iostream>
#include <opencv2/core.hpp>
#include <opencv2/highgui.hpp>
#include <opencv2/imgproc.hpp>
#include <opencv2/videoio.hpp>
#include <opencv2/video.hpp>
using namespace cv;
using namespace std;
int main(int argc, char **argv)
{
    const string about =
        "This sample demonstrates Lucas-Kanade Optical Flow calculation.\n"
        "The example file can be downloaded from:\n"
"
https://www.bogotobogo.com/python/OpenCV_Python/images/mean_shift_tracking/slow_traffic
_small.mp4";
    const string keys =
"{ h help |      | print this help message }"
"{ @image | vtest.avi | path to image file }";
CommandLineParserparser(argc, argv, keys);
parser.about(about);
    if (parser.has("help"))
    {
parser.printMessage();
        return 0;
    }
    string filename = samples::findFile(parser.get<string>("@image"));
    if (!parser.check())
    {
parser.printErrors();
        return 0;
    }
VideoCapture capture(filename);
    if (!capture.isOpened()){
        //error in opening the video input
cerr<< "Unable to open file!" <<endl;
        return 0;
    }
    // Create some random colors
    vector<Scalar> colors;
    RNG rng;
for(int i = 0; i< 100; i++)
    {
        int r = rng.uniform(0, 256);
        int g = rng.uniform(0, 256);
        int b = rng.uniform(0, 256);
colors.push_back(Scalar(r,g,b));
    }
    Mat old_frame, old_gray;
    vector<Point2f> p0, p1;
    // Take first frame and find corners in it
    capture >>old_frame;
```

```
cvtColor(old_frame, old_gray, COLOR_BGR2GRAY);
goodFeaturesToTrack(old_gray, p0, 100, 0.3, 7, Mat(), 7, false, 0.04);
   // Create a mask image for drawing purposes
   Mat mask = Mat::zeros(old_frame.size(), old_frame.type());
   while(true){
      Mat frame, frame_gray;
      capture >> frame;
      if (frame.empty())
         break;
cvtColor(frame, frame_gray, COLOR_BGR2GRAY);
      // calculate optical flow
      vector<uchar> status;
      vector<float> err;
TermCriteria criteria = TermCriteria((TermCriteria::COUNT) + (TermCriteria::EPS), 10, 0.03);
calcOpticalFlowPyrLK(old_gray, frame_gray, p0, p1, status, err, Size(15,15), 2, criteria);
      vector<Point2f>good_new;
for(uinti = 0; i< p0.size(); i++)
      {
         // Select good points
         if(status[i] == 1) {
good_new.push_back(p1[i]);
            // draw the tracks
            line(mask,p1[i], p0[i], colors[i], 2);
circle(frame, p1[i], 5, colors[i], -1);
         }
      }
      Mat img;
add(frame, mask, img);
imshow("Frame", img);
      int keyboard = waitKey(30);
      if (keyboard == 'q' || keyboard == 27)
         break;
      // Now update the previous frame and previous points
old_gray = frame_gray.clone();
      p0 = good_new;
   }
}
```

Program:

// Write your modified code for above program here

Output:

// Write output of your program here

Conclusion:

// Write conclusion here

Quiz:

1. Is it necessary to detect corner points in particular intervals ?

Suggested Reference:

1.The computation of optical flow by S. S. Beauchemin and J. L. Barron. ACM digital Library.
2. https://nanonets.com/blog/optical-flow/#what-is-optical-flow

3. https://docs.opencv.org/3.4/d4/dee/tutorial_optical_flow.html

References used by the students:

Rubric wise marks obtained:

Program	**(Excellent)(4)**	**(Good)(3)**	**(Fair)(2)**	**(Beginning)(1)**
Program execution	Program executes correctly with no syntax or runtime errors	Program executes with a minor error	Program executes with multiple minor (easily fixed error)	Program does not execute (0-1)
Design-Correctness of output	Program displays correct output with no errors	Output/design of output has minor errors	Output/Design of output has multiple errors	Output is incorrect (0-1)
Design of logic	Program is logically well designed	Program has slight logic errors that do no significantly affect the results	Program has significant logic errors	Program is incorrect (0-1)
Standards	Program is stylistically well designed	Few inappropriate design choices (i.e. poor variable names, improper indentation)	Several inappropriate design choices (i.e. poor variable names, improper indentation)	Program is poorly written (0-1)
Documentation	Program is well documented	Missing one required comment	Missing two or more required comments	Most or all documentation missing (0-1)

Criteria	**1**	**2**	**3**	**4**	**5**	**Total**
Marks						

Experiment No: 9

Object detection and Recognition on available online image datasets

Date:

Competency and Practical Skills:

This practical is expected to develop following skills in you

1. Perform object detection
2. Perform object recognition

Relevant CO: CO5

Objectives:

1. Detect multiple objects from online dataset

Equipment/Instruments: Computer systems equipped with python/ matlab/ open-CV

Theory:

Object detection and object recognition are similar techniques for identifying objects, but they vary in their execution. Object detection is the process of finding instances of objects in images. In the case of deep learning, object detection is a subset of object recognition, where the object is not only identified but also located in an image. This allows for multiple objects to be identified and located within the same image. Object recognition is a key technology behind driverless cars, enabling them to recognize a stop sign or to distinguish a pedestrian from a lamppost. It is also useful in a variety of applications such as disease identification in bioimaging, industrial inspection, and robotic vision.

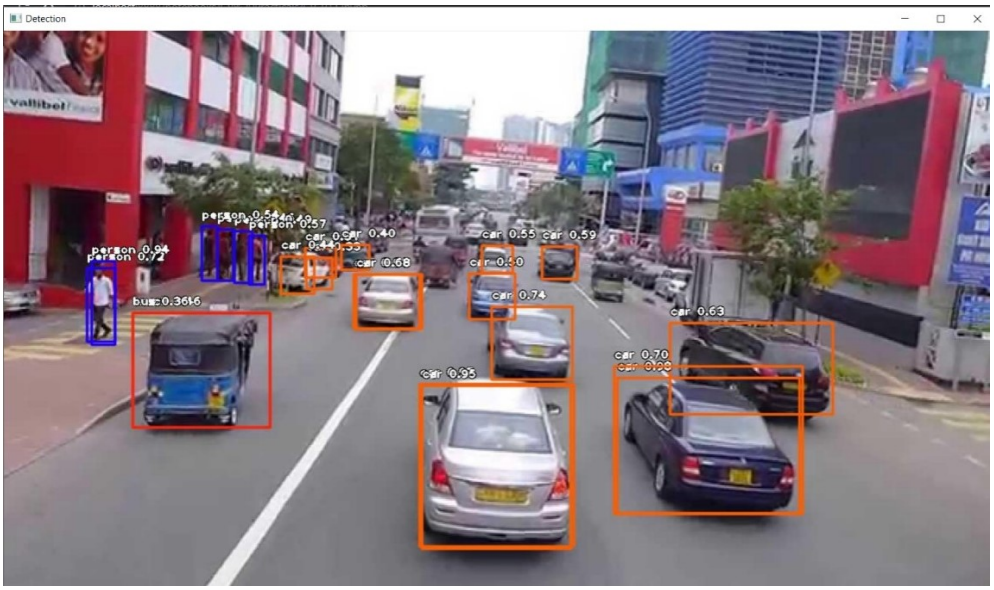


Figure 9.1 Sample object detection and recognition

Procedure:

```
from imageai.Detection import ObjectDetection
detector = ObjectDetection()
model_path = "yolo-tiny.h5" input_path = "cars.jpg" output_path = "output_image.jpg"
detector.setModelTypeAsTinyYOLOv3()
detector.setModelPath(model_path)
detector.loadModel()
detection = detector.detectObjectsFromImage(input_image=input_path,
output_image_path=output_path)
```

Program:

// Write code of your program here

Output:

// Write output of your program here

Conclusion:

// Write conclusion here

Quiz:

1. Differentiate between machine learning approach and deep learning approach for object recognition

Suggested Reference:

1. Computer Vision: Algorithms and Applications, R. Szeliski, Springer, 2011.
2. Introductory techniques for 3D computer vision, E. Trucco and A. Verri, Prentice Hall, 1998.
3. https://www.kaggle.com/getting-started/169984
4. https://www.geeksforgeeks.org/object-detection-vs-object-recognition-vs-image-segmentation/

References used by the students:

Rubric wise marks obtained:

Program	**(Excellent)(4)**	**(Good)(3)**	**(Fair)(2)**	**(Beginning)(1)**
Program execution	Program executes correctly with no syntax or runtime errors	Program executes with a minor error	Program executes with multiple minor (easily fixed error)	Program does not execute (0-1)
Design-Correctness of output	Program displays correct output with no errors	Output/design of output has minor errors	Output/Design of output has multiple errors	Output is incorrect (0-1)
Design of logic	Program is logically well designed	Program has slight logic errors that do no significantly affect the results	Program has significant logic errors	Program is incorrect (0-1)
Standards	Program is stylistically well designed	Few inappropriate design choices	Several inappropriate design choices	Program is poorly written (0-1)

		(i.e. poor variable names, improper indentation)	(i.e. poor variable names, improper indentation)	
Documentation	Program is well documented	Missing one required comment	Missing two or more required comments	Most or all documentation missing (0-1)

Criteria	**1**	**2**	**3**	**4**	**5**	**Total**
Marks						

Experiment No: 10

Character or digit or face classification project

Date:

Competency and Practical Skills:

This practical is expected to develop following skills in you

1. Problem solving

Relevant CO: CO5

Objectives:

1. Applying computer vision knowledge to solve real time problem

Equipment/Instruments: Computer systems equipped with python/ matlab/ open-CV

Theory:

Face detection, also called facial detection, is an artificial intelligence-based computer technology used to find and identify human faces in digital images and video. Face detection technology is often used for surveillance and tracking of people in real time. It is used in various fields including security, biometrics, law enforcement, entertainment and social media.

To perform the face recognition function, face detection is first performed to determine the position of the face in the picture. OpenCV performs functionalities for the same. It firstly extracts the feature images into a large sample set by extracting the face Haar features in the image and then uses the AdaBoost algorithm as the face detector. In face detection, the algorithm can effectively adapt to complex environments such as insufficient illumination and background blur, which greatly improves the accuracy of detection. For a set of training sets, different training sets are obtained for subsequent work by changing the distribution probabilities of each of the samples, and each training set is trained to obtain a weak classifier, and then these several classifiers are weighted.

Figure 10.1 Sample output of face recognition

Procedure:

Captureframe-by-frame from the input video
faces = faceCascade.detectMultiScale(gray,scaleFactor=1.1, minNeighbors=5, minSize=(30, 30),flags=cv2.CASCADE_SCALE_IMAGE)
Draw a rectangle around the faces for (x, y, w, h) in faces:
cv2.rectangle(frames, (x, y), (x+w, y+h), (0, 255, 0), 2)
Display the resulting frame cv2.imshow('Video', frames)

Program:

// Write code of your program here

Output:

// Write output of your program here

Conclusion:

// Write conclusion here

Quiz:

1. Compare approaches for face detection from images and videos

Suggested Reference:

1. https://www.hindawi.com/journals/js/2021/4796768/

References used by the students:

Rubric wise marks obtained:

Program	**(Excellent)(4)**	**(Good)(3)**	**(Fair)(2)**	**(Beginning)(1)**
Program execution	Program executes correctly with no syntax or runtime errors	Program executes with a minor error	Program executes with multiple minor (easily fixed error)	Program does not execute (0-1)
Design-Correctness of output	Program displays correct output with no errors	Output/design of output has minor errors	Output/Design of output has multiple errors	Output is incorrect (0-1)
Design of logic	Program is logically well designed	Program has slight logic errors that do no significantly affect the results	Program has significant logic errors	Program is incorrect (0-1)
Standards	Program is stylistically well designed	Few inappropriate design choices (i.e. poor variable names, improper indentation)	Several inappropriate design choices (i.e. poor variable names, improper indentation)	Program is poorly written (0-1)
Documentation	Program is well documented	Missing one required comment	Missing two or more required comments	Most or all documentation missing (0-1)

Criteria	1	2	3	4	5	Total
Marks						