Experiment-2.3

Aim of the Experiment : Human face detection using Haar features.

Problem Description :

Human face detection using Haar feature-based cascades involves the use of a machine learning approach to detect faces in images or video streams. This method utilizes Haar-like features and a cascade classifier, typically implemented using OpenCV, a popular computer vision library.

Haar features are patterns of intensities in an image that are useful for distinguishing objects. The Haar cascade classifier is a machine learning-based approach that utilizes these features to detect objects in images. Here's a basic overview of the process:

1) Haar-like features: These are digital image features used in object recognition. They are simple rectangular patterns that are calculated over specific regions of an image. These features are selected based on their ability to distinguish between the object of interest (e.g., a human face) and the background.

2) Training the classifier: The Haar cascade classifier is trained using a large dataset of positive and negative examples. Positive examples are images containing the object to be detected (e.g., human faces), while negative examples are images without the object. During training, the classifier learns to differentiate between these examples based on the Haar-like features.

3) Cascade classifier: The trained classifier is organized into multiple stages, each containing a series of weak classifiers. These weak classifiers are simple decision rules based on Haar-like features. The cascade structure allows for fast rejection of non-object regions in the image, reducing the computational load.

4) Detection: To detect human faces in an image or video stream, the Haar cascade classifier is applied by sliding a window of varying sizes across the image. At each window position, the Haar-like features are calculated, and the cascade classifier evaluates whether the region contains a human face based on the learned decision rules.

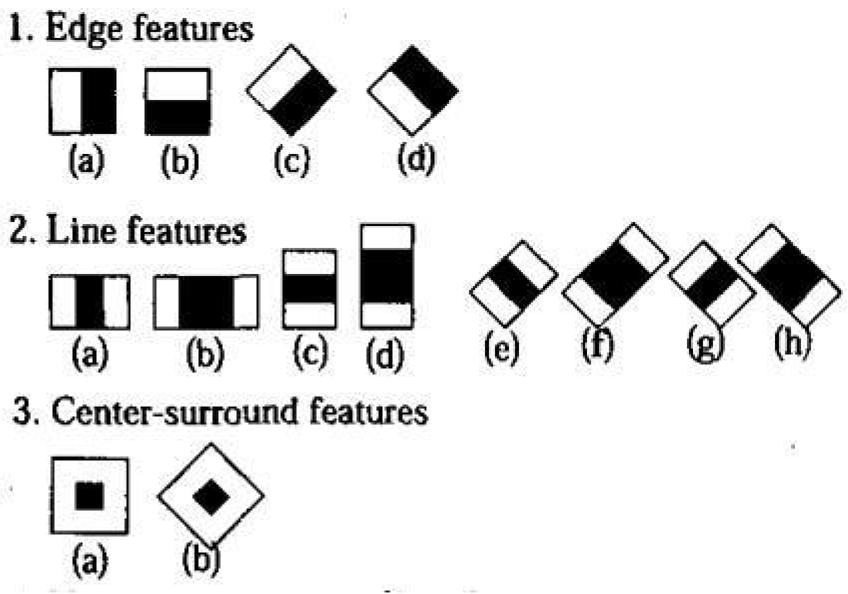
5) Post-processing: After detection, post-processing steps such as non-maximum suppression may be applied to remove duplicate or overlapping detections and refine the final output.

There are three primary types of Haar-like features:

1) Edge features: These features capture changes in intensity between adjacent rectangular regions. They are defined by two adjacent rectangles with opposite intensities.

2) Line features: These features capture changes in intensity along a line. They are defined by three adjacent rectangles: two with the same intensity and one with the opposite intensity.

3) Center-surround features: These features capture differences in intensity between a central region and its surrounding areas. They are defined by a pair of adjacent rectangles with the central region having a higher intensity.

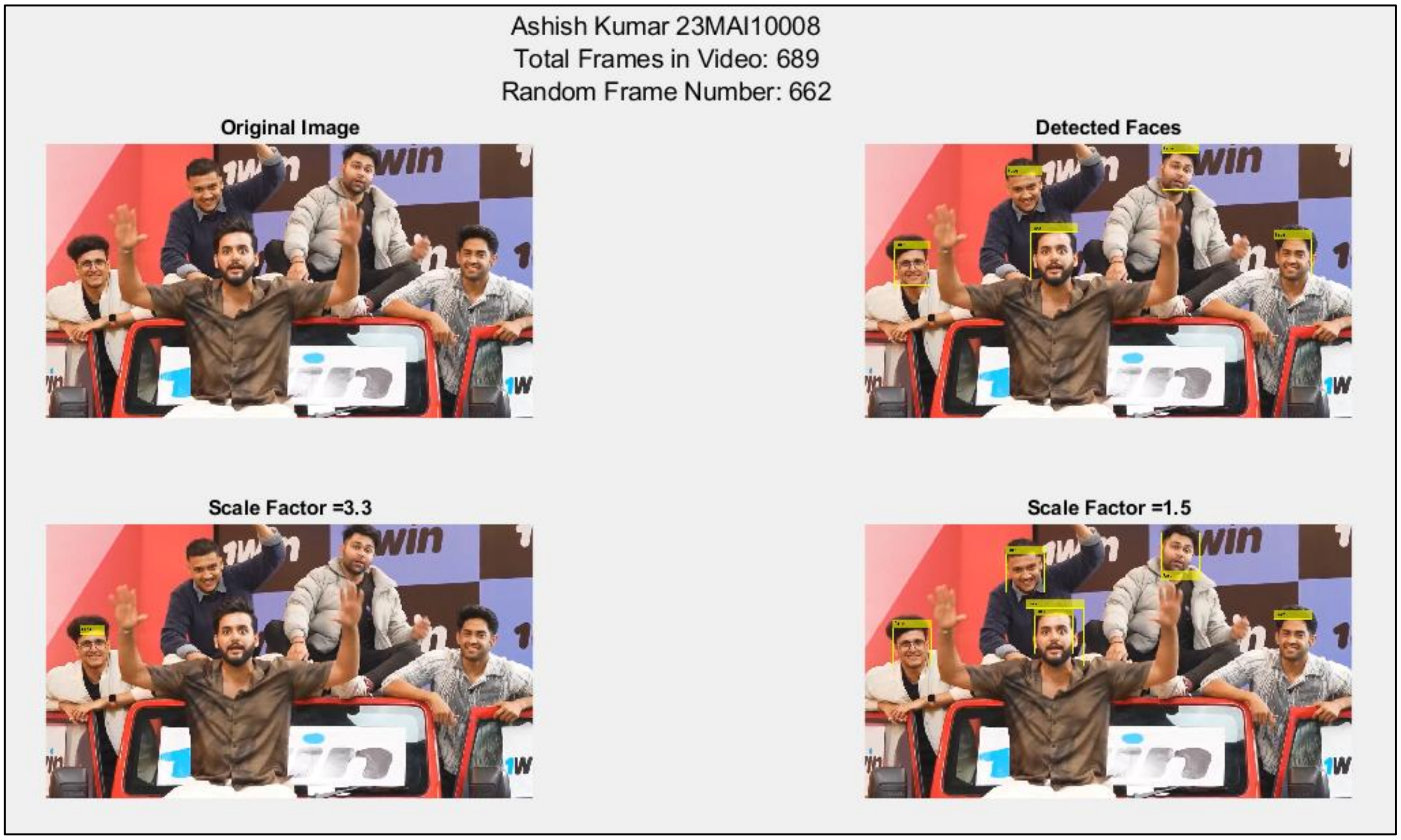


Code :

```
% Create a video reader
VideoObj = VideoReader('Match.mp4');

% Count Total frames in video
framecount=VideoObj.NumFrames;
X = ['Total Frames in Video: ',num2str(framecount)];

% Select a Random frame
Rframe = randi([1,framecount]);
Y = ['Random Frame Number: ',num2str(Rframe)];
disp(Y)
image = read(VideoObj,Rframe);
```

```
% Display the original image
subplot(2,2,1)
imshow(image);
title('Original Image');


% Load the Haar cascade classifier for face detection
faceDetector = vision.CascadeObjectDetector();

% Detect faces in the image
bbox = step(faceDetector, image);

% Annotate the detected faces on the image
detectedImage = insertObjectAnnotation(image, 'rectangle', bbox, 'Face');

% Display the image with annotated faces
subplot(2,2,2)
imshow(detectedImage);
title('Detected Faces');


% Customize face detection parameters

% Adjust the scale factor for face detection
faceDetector.ScaleFactor = 3.3;
% Detect faces with the updated parameters
bboxUpdated = step(faceDetector, image);
% Annotate the detected faces on the image
detectedImageUpdated = insertObjectAnnotation(image, 'rectangle', bboxUpdated, 'Face');

% Display the image with annotated faces after customization
subplot(2,2,3)
imshow(detectedImageUpdated);
title('Scale Factor =3.3');


% Adjust the scale factor for face detection
faceDetector.ScaleFactor = 1.5;
% Detect faces with the updated parameters
bboxUpdated = step(faceDetector, image);
% Annotate the detected faces on the image
detectedImageUpdated = insertObjectAnnotation(image, 'rectangle', bboxUpdated, 'Face');

% Display the image with annotated faces after customization
subplot(2,2,4)
imshow(detectedImageUpdated);
title('Scale Factor =1.5');

sgtitle(["Ashish Kumar 23MAI10008",X,Y])
```

Output :



Learning outcomes :

1. Learnt about the concept of Face Detection.
2. Learnt about how Haar features work.
3. Learnt about how to make matched faces using Haar features.
4. Learnt about how to detect faces using various scales.
5. Learnt about how to detect Face in a Video.