Individual based Image processing project

G.A.T.L Ranasinghe, 21/ENG/033, Department Of Electrical and Electronic Engineering, University Of Sri Jayewadenepura, Sri Lanka

This manuscript was compiled on July 29, 2024

The role of haarcascade classifier

In order to detect face clearly and separately within all other objects ,code should need to be highly trained. Also proper training dataset should be embedded into the code. Paul Viola and Michael Jones presented an efficient object recognition technique in their paper "Rapid Object Detection using a Boosted Cascade of Simple Features" (2001) that uses Haar feature-based cascade classifiers. This method is based on machine learning, and it involves training a cascade function with a large number of both positive and negative images. Next, it's applied to identify things in more pictures.

Feature selection. Many features are now calculated using all available kernel locations and sizes. Just consider how much computing that requires. In a 24x24 window, more than 160000 features are displayed. To calculate each feature, the total of the pixels under the black and white rectangles must be determined. They introduced the integral picture as a solution. It lowers the computations for a given pixel to an operation requiring just four pixels, regardless of the size of the image. It produces results quite quickly.Unfortunately some of features are irrelevant. Sometimes eyes are frequently darker than nose and cheeks. But the same windows applied to cheeks or any other area is meaningless. So how do we select the best features out of 160000+ features? It is achieved via Adaboost.

Adaboost. We do this by applying every feature to every training image. It determines the optimal threshold for each attribute to categorize the faces as positive or negative. There will undoubtedly be mistakes or incorrect classifications. The features that classify faces and non-faces images the most accurately are those with the lowest error rate, or minimum error rate features. (The method is not as straightforward as this). At first, the weight of each image is the same. The weights of images that are incorrectly classified are raised after each classification. The same procedure is then carried out. New error rates are calculated. Also new weights. The process is continued until the required accuracy or error rate is achieved or the required number of features are found.

Furthermore. The final classifier is a weighted sum of these weak classifiers. It is called weak because it alone can't classify the image, but together with others forms a strong classifier. The paper says even 200 features provide detection with 95 percent accuracy. Final step is to reduce the setup into 6000 features.

Since the majority of an image is not a face region, it is better to have a straightforward method to determine whether a window is not a face region. If it is not, discard it in one go and don't process it again; otherwise, concentrate on regions where a face could be present. In order to do this, they introduced the concept of Cascade of Classifiers, which groups features into different stages of classifiers and applies them one at a time, rather than applying all 6000 features on a window; typically, the first few stages will have a relatively small number of features. If a window fails the first stage, it is discarded without taking into account the features that remain on it.

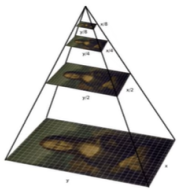

Fig. 1. Image pyramid

Python code explanation

First of all it is required to import haarcascade classifier to the code through cascade = cv2.CascadeClassifier(path). This path depicts the relevant folder which shows the exact location of haarcascade classifier. Purpose of importing this classifier is important because it contains the training dataset in order to detect faces. Then webacam of user will be initiated through video = cv2.VideoCapture(0) and this line of code will act as a reference to the webcam. In order to detect faces it is vital to implement a loop which continuously turn on the camera.

Another line of code which reads the current frame is success, video.read(). Next image should be converted to gray scale image to reduce complexities.Colour image includes the Red, Green and blue colour intensities. Then it is difficult to extract features of face with those blend of colours. Hence by using cv2.cvtColor method colour image get converted to gray scale.Then faces are detected through cascadedetectMultiScale(gray$_scale, 1.2, 4)lineofcode.$

This third parameter acts as a scale to the image pyramid which is a multi-scale representation of an image. At each layer of the image pyramid the image is downsized and (optionally) smoothed via a Gaussian filter. This scale parameter controls the factor in which image is resized at each layer of the image pyramid. Ultimately influencing the number of levels in the image pyramid. A smaller scale will increase the number of layers in the image pyramid and increase the amount of time it takes to process the image.

Defining dimensions of the image . Width of the image is defined as W while height of the image defined as h. Also each face should have a location coordinates which are specified as x,y. Then the result is displayed through this line of code.cv2.imshow("img", image). Face is recognized as blue colour rectangle which has a thickness of 2mm.

Image result

Role of canny edge detection. An edge is defined as a sudden change in pixel intensity within an image; edges represent the boundaries between distinct objects or regions with varying intensity levels. Canny edge detection is a popular and widely used edge detection technique that aims to identify and extract the edges of objects within an image. It was developed by John

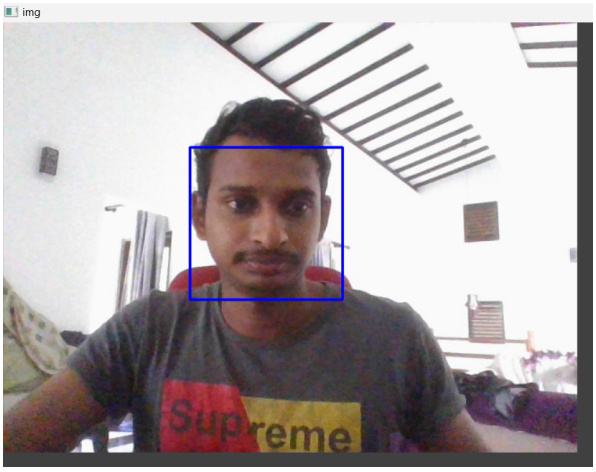

Fig. 2. Placeholder image of a frog with a long example legend to show justification setting.

$$G(x, y) = \frac{1}{2\pi\sigma^2} \cdot \exp\left(-\frac{x^2 + y^2}{2\sigma^2}\right)$$

F. Canny in 1986 and has since become a fundamental tool in computer vision and image analysis. “Edge detection” is the term given to this technique in image processing. Formal steps to implement the method

1. Gray scale conversion
2. Noise reduction
3. Gradient calculation
4. Non maximum suppresion
5. Double thresholding
6. Edge tracking by hystersis

Noise reduction Gaussian filtering the input image is the initial stage in the Canny edge detection algorithm. One smoothing technique that lowers noise in an image is the Gaussian filter. False edges can be introduced by noise, which could reduce the edge detection method’s precision. By convolving the image with a Gaussian kernel, the Gaussian filter smooths the image and reduces high-frequency noise while maintaining the sharpness of the edges. Gaussian kernel is depicted as below

- x and y are the spatial coordinates of the kernel.
- is the mathematical constant Pi (approximately 3.14159).
- is the standard deviation, controlling the width of the Gaussian distribution.

Gradient calculation The gradient of the smoothed image is computed using the Canny approach after first minimizing noise. The gradient conveys the rate at which each pixel’s intensity changes. The method uses the Sobel operator, a derivative concept, to determine the gradient magnitude and orientation for each pixel in the image. The gradient magnitude represents the strength of the intensity change, whereas the gradient orientation denotes the direction of the steepest shift.

Double thresholding Double thresholding is used in the following stage to divide edges into three groups: non-edges, weak edges, and strong edges.For this, two thresholds are used: a high threshold and a low threshold. Strong edges, or pixels exhibiting notable changes in intensity, are defined as having gradient magnitudes higher than the high threshold. Weak edges are defined as pixels having gradient magnitudes between low and high thresholds. These weak edges require more confirmation because they could be genuine edges or noise. Pixels that possess gradient magnitudes lower than the threshold are deemed non-edges and are subsequently removed.

Libraries and datasets. In order to implement the face detection code by using python certain libraries are useful. OpenCV is one of vital library to implement CV command for this project. This openCV library is initiated through import cv2 line of code. Another crucial feature that is used to complete this project is Haarcascade classifier file which contains lots of training dataset.

References. [1]Samina, “Educative Answers - Trusted Answers to Developer Questions,” *Educative*. Available: https://www.educative.io/answers/what-is-canny-edge-detection. [Accessed: Jul. 29, 2024] [2]N. Barney, “What is Face Detection and How Does It Work?,” *SearchEnterpriseAI*, Apr. 23, 2023. Available: https://www.techtarget.com/searchenterpriseai/definition/face-detection. [Accessed: Jul. 29, 2024] [3]OpenCV, “OpenCV: Cascade Classifier,” *docs.opencv.org*. Available: https://docs.opencv.org/3.4/db/d28/tutorial_cascade_classifier.html