MANJARA CHARITABLE TRUST
RAJIV GANDHI INSTITUTE OF TECHNOLOGY, MUMBAI

(Permanently Affiliated to University of Mumbai) JUHU VERSOVA LINK ROAD, ANDHERI (W), MUMBAI-53 **DEPARTMENT OF ELECTRONICS AND TELECOMMUNICATION ENGINEERING**

University of Mumbai

Academic Year: 2020-21

A Project Report

On

QR Code & Bar Code Scanner using Python and Open-CV

Submitted in partial fulfilment of the requirements of

Subject:

"Image Processing and Machine Vision"

Third Year of Engineering In

Electronics and Telecommunication Engineering

BY

SAYLEE SURVE	B-648
AADITYA AUTI	B-658
BHARAT KOLEKAR	B-659
TEJAS LAD	B-660

Under the Guidance of

Prof. Arvind Sangle

Abstract

Identifying the QR code and Bar code using Open-CV and python is the necessity of today's world. This function used by so many robot development organizations for more versatile robot development. This report describes the face detection and recognition mini-project undertaken for the visual perception and autonomy module at Plymouth university. It reports the technologies available in the Open-Computer-Vision (OpenCV) library and methodology to implement them using Python. For code detection, pillow and pyzbar were used and for code recognition Eigenfaces, Fisher faces and Local binary pattern histograms were used. The methodology is described including flow charts for each stage of the system. Next, the results are shown including plots and screen-shots followed by a discussion of encountered challenges. The report is concluded with the authors' opinion on the project and possible applications.

Introduction

The following document is a report on the mini project for Robotic visual perception and autonomy. It involved building a system for face detection and face recognition using several classifiers available in the open computer vision library (OpenCV). Barcode and qr code recognition are a non-invasive identification system and faster than other systems since multiple faces can be analysed at the same time.

The History of Face Recognition

A QR code (abbreviated from Quick Response code) is a type of matrix barcode (or two-dimensional barcode) invented in 1994 by the Japanese automotive company Denso Wave. A barcode is a machine-readable optical label that contains information about the item to which it is attached. In practice, QR codes often contain data for a locator, identifier, or tracker that points to a website or application. A QR code uses four standardized encoding modes (numeric, alphanumeric, byte/binary, and kanji) to store data efficiently; extensions may also be used.

The Quick Response system became popular outside the automotive industry due to its fast readability and greater storage capacity compared to standard UPC barcodes. Applications include product tracking, item identification, time tracking, document management, and general marketing.

A QR code consists of black squares arranged in a square grid on a white background, which can be read by an imaging device such as a camera, and processed using Reed–Solomon error correction until the image can be appropriately interpreted. The required data is then extracted from patterns that are present in both horizontal and vertical components of the image.

QR CODE DETECTION

he decoding function will be doing mainly three things, and can be listed as follows:

- Recognizing and decoding the barcode/QR code that we will be showing to the camera.
- Adding the stored information as a text on the recognized barcode/QR code.
- And lastly, exporting the stored information as a text document.

```
#import libraries

import cv2
from pyzbar import pyzbar
```

Now, let's write the function. Instead of adding part by part, I will share the whole function with you. Since, indentation matters when writing in python, I don't want to disorganize things by ruining the structure of the code. I will add my comments below the code.

```
def read_barcodes(frame):
    barcodes = pyzbar.decode(frame)
    for barcode in barcodes:
        x, y , w, h = barcode.rect

        #1
        barcode_info = barcode.data.decode('utf-8')
        cv2.rectangle(frame, (x, y),(x+w, y+h), (0, 255, 0), 2)

        #2
        font = cv2.FONT_HERSHEY_DUPLEX
        cv2.putText(frame, barcode_info, (x + 6, y - 6), font, 2.0,
(255, 255, 255), 1)

        #3
        with open("barcode_result.txt", mode ='w') as file:
            file.write("Recognized Barcode:" + barcode_info)

    return frame
```

Understanding the function:

- Firstly, we are decoding the information from the barcode or QR code. And then drawing a rectangle around it. This helps us to see if our machine has detected the barcode/Qr code.
- Secondly, we are adding text on top of the rectangle that was created. The text will show the decoded information.
- Thirdly, we are exporting the information into a text document. If you are planning to test with multiple barcodes or QR codes, I recommend changing the document name otherwise it will overwrite.

Main Function

In this step, we will write the main function, where the application is prompt to work. The main function will turn on the video camera of the computer, and the then call the decoding function. Here is the code:

```
def main():

    #1
    camera = cv2.VideoCapture(0)
    ret, frame = camera.read()

    #2
    while ret:
        ret, frame = camera.read()
        frame = read_barcodes(frame)
        cv2.imshow('Barcode/QR code reader', frame)
        if cv2.waitKey(1) & 0xFF == 27:
            break

    #3
    camera.release()
    cv2.destroyAllWindows()

#4
if __name__ == '__main__':
    main()
```

Source Code for detection face mask

#import libraries

import cv2

from pyzbar import pyzbar

def read_barcodes(frame):

barcodes = pyzbar.decode(frame)

for barcode in barcodes:

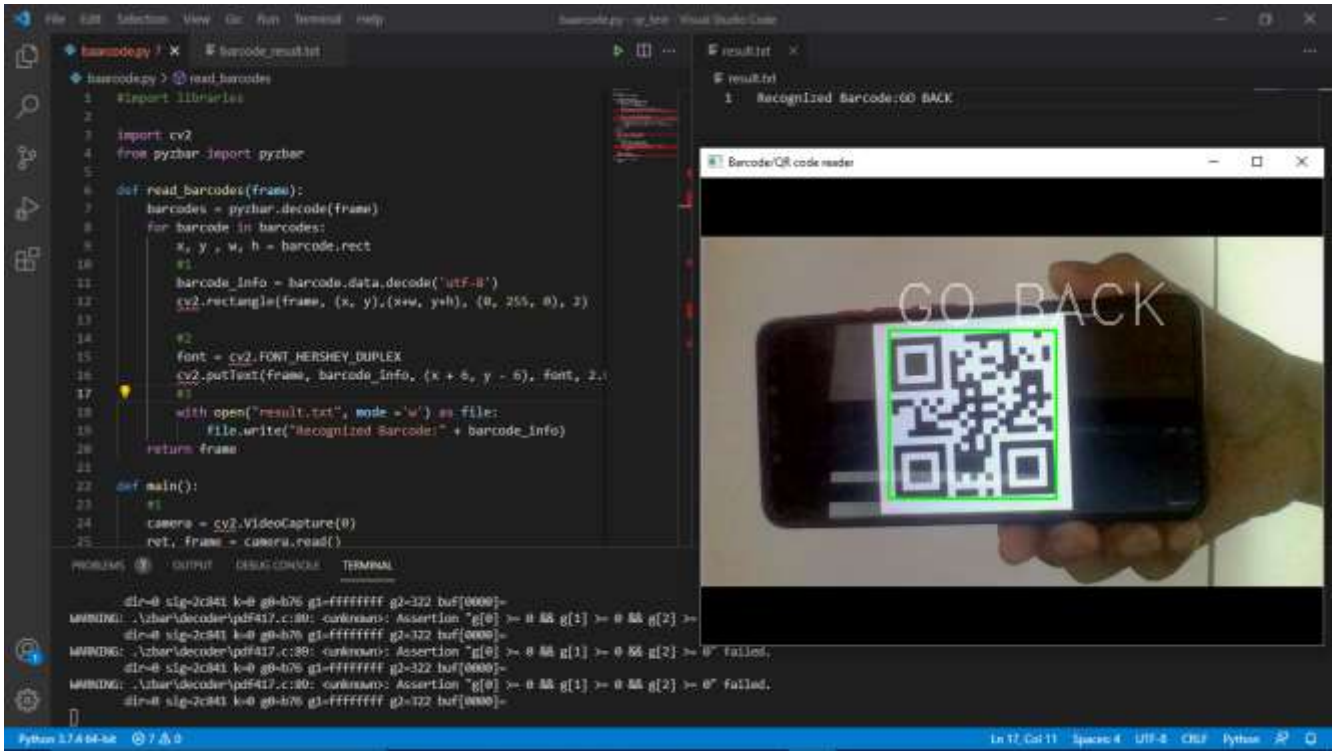
x, y , w, h = barcode.rect

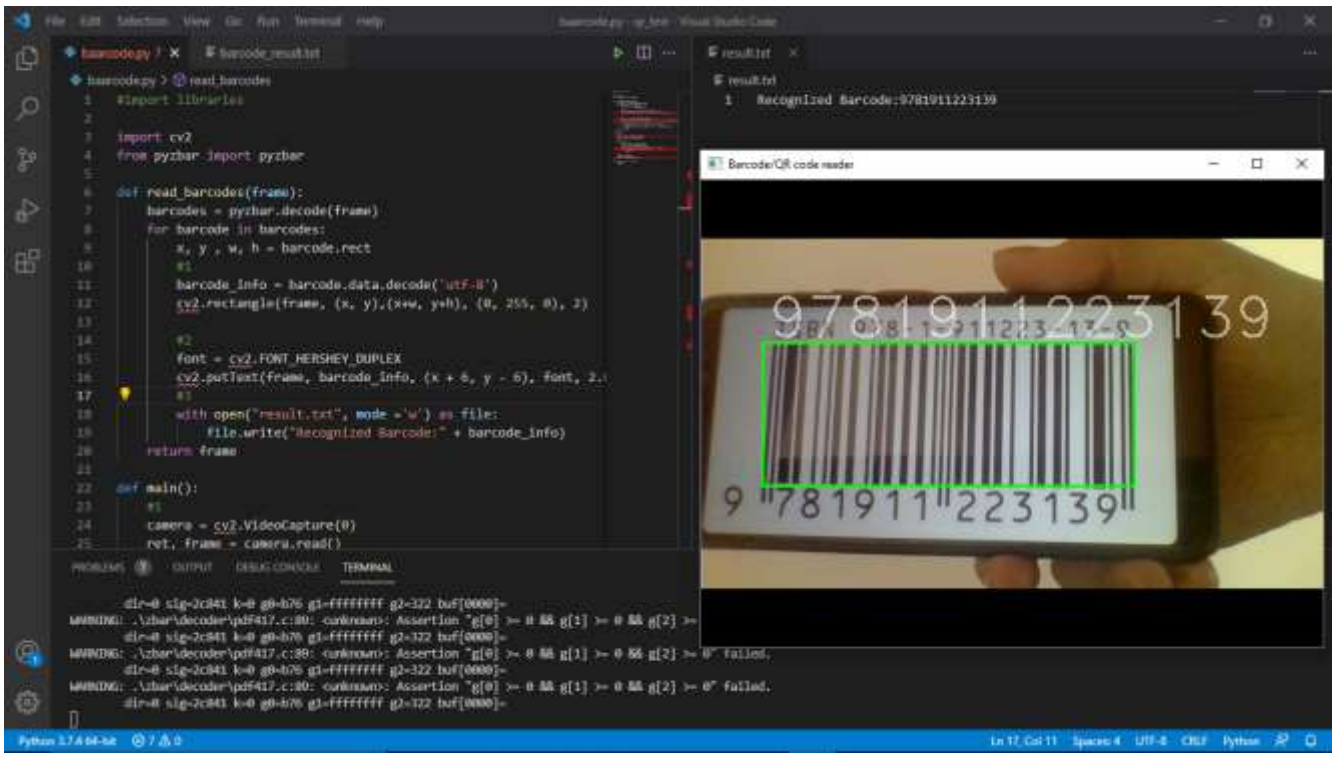
```
        #1
        barcode_info = barcode.data.decode('utf-8')
        cv2.rectangle(frame, (x, y),(x+w, y+h), (0, 255, 0), 2)


        #2
        font = cv2.FONT_HERSHEY_DUPLEX
        cv2.putText(frame, barcode_info, (x + 6, y - 6), font, 2.0, (255, 255, 255), 1)
        #3
        with open("result.txt", mode ='w') as file:
            file.write("Recognized Barcode:" + barcode_info)
    return frame


def main():
    #1
    camera = cv2.VideoCapture(0)
    ret, frame = camera.read()
    #2
    while ret:
        ret, frame = camera.read()
        frame = read_barcodes(frame)
        cv2.imshow('Barcode/QR code reader', frame)
        if cv2.waitKey(1) & 0xFF == 27:
            break
    #3
    camera.release()
    cv2.destroyAllWindows()
#4
if __name__ == '__main__':
    main()
```

Output


Recognized Barcode:GO BACK
Barcode/QR code reader
GO BACK


Recognized Barcode:9781911223139
Barcode/QR code reader
9781911223139

Conclusion

This paper describes the mini-project for visual perception and autonomy module. Next, it explains the technologies used in the project and the methodology used. Finally, it shows the results, discuss the challenges and how they were resolved followed by a discussion. Using pillows and pyzbar for code detection worked extremely well even when subjects wore spectacles. Real time video speed was satisfactory as well devoid of noticeable frame lag. Considering all factors, LBPH combined with Haar-cascades can be implemented as a cost-effective face recognition platform. An example is a system to identify known troublemakers in a mall or a supermarket to provide the owner a warning to keep him alert or for automatic attendance taking in a class.