Qr generator, reader and comparator

Repository with the code:
https://github.com/Yaiza0706/DigitalImageProcessing

- **Motivation**
- **Problem description**
- **Background**
- **Methodology**
- **Results**
- **Conclusions**
- **Hints for future research directions**
- **Relevant references.**

Yaiza Rubio Chavida
Fidan Rustambayli
Hugo Coutier
Date: 13/12/2020
Digital Image Processing project

MOTIVATION

We would like to generate different QR which will be analyzed by a QR code reader and comparator.

Our idea is to assign each person a QR code in a company, university ... This QR will contain personal information of that person: name, last name, telephone or email and what is its position in the company. Then, each person will receive its personalized QR code and they will have to show their code to our detector to get permission before entering a class / office.

The program will compare if this code is in the corresponding folder for the room, if it found a similar image, it will allow the person to enter and it will save the date, hour and how long this person stayed there. Also, it will decode the QR to save the name and last name of this person. If the images are not similar, the person won't be able to enter.

To achieve points exposed before we will analyze different softwares to see which fits better for our project. We will compare and learn about QR image new tools

PROBLEM DESCRIPTION

The main problem our program will solve is to control which people have entered to some place and at what time they did it. More in detail, we will focus on two objectives:

- Prevent people entering rooms without belonging to the company / university. In this way, only people we want will have access to some place.
- Control people who have been together in some place. Now it has special importance due to the COVID-19 pandemic. It can be very useful in the case that some of these people test positive in covid, the detector will help us to know how many people, how much time and when they have been in contact with. The company or the university will warn these people to stay in self-isolation.

The advantages of our idea is that people will have zero contact, and it is an easy implementation to solve two of the biggest problems nowadays. Although many people have tested similar codes by themselves, they have not yet been joined for a similar purpose.

METHODOLOGY

The first thing we discussed about what software tool was the best for our project. This task was a bit difficult because each of us came from a different university, so we had studied different software tools.

Finally we chose Python 3, using Jupyter Notebook. All of us worked with it this year in other courses. We liked it because we can compile and run step by step without the necessity of running the complete code and also because we can add comments between the code for a better understanding of what each of us made. Then, analysing this tool we found that it has a QR library, it facilitated a lot of our work. Moreover, after looking for more useful libraries [1], we finally used these libraries: cv2, numpy, pandas and pyzbar.

To share our project we used GitHub.

Then we divided the code part in three very well separated:

- ➔ First one, was QR generation for each person that has access to the place we are preparing our system for.
- ➔ Second one, was the comparison part. The program will compare the data with the database and give an access to the person to enter if it finds matching information. If the person is in the list, the computer will allow him to enter, if not it will show a negative feedback.
- ➔ Last one was the QR decoding part. If the person got the permission to enter, the computer will decode the QR and save the name and surname as well as date, time and how long that person will be there.

To do this, we looked for some codes that can help us in the development of our idea. We tried most of them, and saw the advantages and disadvantages of each one and we finally achieved our final result.

BACKGROUND

We have looked for different programs that have been tested and we will take them as a reference to learn different things from them .

First of all we saw different examples in different languages in github [4], it was quite good because it gave us an overall idea and helped us to start trying different codes. Then we found a very recent report explaining how to read and generate different QR codes using python, qrtools library. The main advantage of this report is that it was quite updated.[5]

After that, we got into more detail and looked for the zbar library and ways to install it. The main advantage in this article is that an overall of what we have to do was also given including screenshots to see how it is really done in a real code. [2]

Finally, we watched a youtube video that explains the relevant parts of the QR code, its size and discusses how much data a QR-code can hold. [6]

RESULTS

Our first objective was to learn QR tools to complete our program. We will show our results separated in the three parts exposed before:

➔ QR GENERATION

For this part, our code will firstly ask for the personal data of the new person. It will be asking as shown below.

```
: print("You are going to generate a new QR")
  name = input("Introduce your name please: ")
  lastname= input("Lastname: ")
  contact= input("Email/phone number:")
  position = input("Student/teacher/other(specify): ")
  newperson = People(name,lastname,contact,position)
```

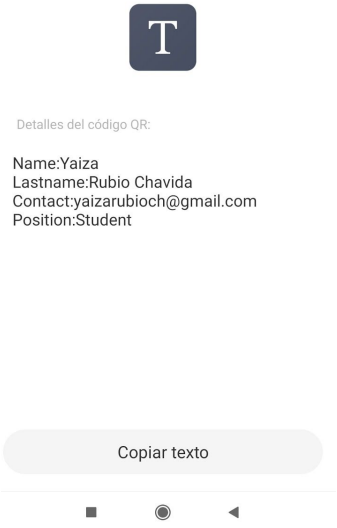
```
You are going to generate a new QR
Introduce your name please: Yaiza
Lastname: Rubio Chavida
Email/phone number:yaizarubioch@gmail.com

Student/teacher/other(specify): Student
```

If everything worked okay, it will show a message with the data that was encoded.

```
Great! You have created a new profile:
Name:Yaiza
Lastname:Rubio Chavida
Contact:yaizarubioch@gmail.com
Position:Student
```

Lastly, the QR with all the information will be saved as a .png image.



As we can see in the first image, the name of the file will be the name and last name of the person.

In the second one, we can see what you find when you decode the QR.

We created a few false profiles to test our code.

➔ QR COMPARISON

For this part, we created two demo QR, program finally will print:

```
These images are completally different images 18269.250780437043
```

Then we implemented two if functions. One for accepting the entrance and it will ask for the date, time and duration, in the same way it did in the previous part. It also will add this data to the previous information and will also save it in a dataframe.

The other one is for denying the entrance; in this case it will print an error message.

```
Sorry. You don´t have access to this room.
```

➔ QR DECODING

This part was one of the easiest because there was a function used for decoding.

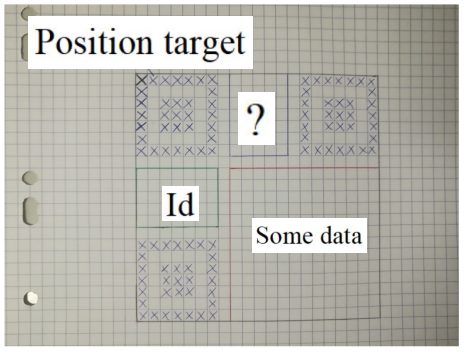
Our second objective was to know more about QR codes[3]. After a deep study, we want to expose our final discoveries:

- ➢ QR means quick response code and it is a 2 dimensional barcode (main differences with one dimensional barcode is that they can store a larger amount of information per area and they don't need a chunky hand-held scanner to scan them).
- ➢ It is a way for storing information in a dot matrix that can be read and take the user to a web page, application or person profile (like in our case).
- ➢ It was created by Denso Wave, a Japanese company, in 1994 .
- ➢ Looking at its appearance we realized that the code has always three squares filled in black color with a white border in the corners. This helps the reader to find the position of the code.
- ➢ Moreover, it is composed of alignment targets, lines, two format areas, and data (it saves one of this part for correction and detection).
- ➢ The main objective of this tool is reading its content at high speed.

SOME ATTEMPTS DID BEFORE THIS FINAL IMPLEMENTATION

Before getting to this code, we first tried to create our own QR-code system with Id and data coded in it as explained below:

We imagined a 20x20 qr-code with:



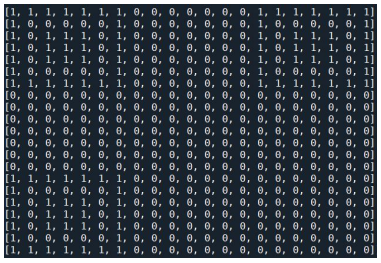
- 3 positionals targets as there are in all QRCodes
- 1 space to put a binary Id number to log into a database
- 1 space for some other data we can put
- 1 space where we didn't load anything but we can load here more data or error correction data or maybe one more target

The position targets help us to know what is the rotation of the qr-code when we scan it, then we can correct the rotation and extract data. In some QR-codes there are also some alignment targets

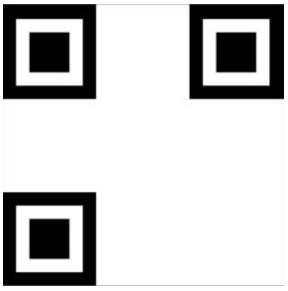
To realize this, we imagined the QR-code as an array where colors were represented with numbers (1 for black 0 for white), we have used python for its efficient tools to work on lists.

1) First step: Create an empty QR-code with only the three targets:

We named the function "generate_empty()" which doesn't take any parameter.



Empty QR-code in python list



Empty QR-code drawn with turtle

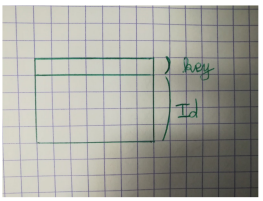
2) Second step: create the function to draw the QR-code (like previous image)

This step consists in going through the QR-code coded with python array and to draw for each value in this array, a square which is black if the value in the array is 1 and white if the value is 0. For that we created a function "trace(M,taille)" which means "draw(M,size)" but in french. Inside this function we used the module turtle: we created two loops to go through the matrix values and for each one we went to the place we had to draw the corresponding square and we drew a black or white square, depending on the value in the matrix.

3) Third step: load data into the QR-code

To do that we had to use the draw we presented at the beginning which represents the way we imagined we could store the data onto our QR_code.

We splitted this step into 3 substeps :
i) First substep: generate a QR-code with Id and key



We created a function “generate_Id_Clef()” wich mean generate ID and key which generate a random number between 0 and 2^28-1 (We decided to code the Id on 28 bits(or pixels) and return the Id in decimal value, in binary list, and the key in binary list, and in decimal value.
The key is the sum of decimal Id digits coded in binary is to verify if the QR-code is valid. After, we created a function called “generate_qr_avec_id_et_clef()” where we put the values of Id and key in the matrix as binary numbers (black pixel = 1, white = 0)

ii) Second substep: load data into the QR-code

For this step we created a function which takes as parameter an array of 169 numbers 0 or 1 and the function loads these data into the red “some data” area of the QR-code. This data can be a lot of things but we haven't decided yet what it is and in which order it is coded in the QR-code.

iii) Third substep: draw and save the QR-code

To realize this step we re-used the function we used in the second step drawing ones as black pixels and zeros as white. Then we had to save the resulting image in our computer, then it would allow us to generate QR_codes with a loop.
For that we used different function: **.getscreen()** which is a function of the module turtle of python to get all things present in the screen, then **.getcanvas()**, then **Canvas.postscript(file=”filename”)**, and finally we use **save_as_png** wich is a function we created to convert the file we get from **Canvas.postscript(file=”filename”)** to a png image and wich rename it with a name we pass as a parameter of the function.

Here is the result we obtained with the code we wrote.
There’s a place where there’s a blank but as we didn’t decide how we could organize data on this self made Qr-code, we chose not to fill with anything but of course we could load more things into it.

This analysis can be continued in a future.

CONCLUSIONS

Although some difficulties were presented, we finally were happy with our achievements. Some of these difficulties were: choosing the appropriate environment, analyzing the different functions of the libraries and maybe the biggest one, was a bit of confusion when joining different parts. However, we thought that we achieved our learning objectives and increased our knowledge of this famous and useful tool: QR code. Also our initial system was almost complete and we found very good results.

HINTS FOR FUTURE RESEARCH DIRECTIONS

We were able to complete most of our initial ideas, but there was a step that we couldn't complete because we didn't find a good way to implement it. This step missing was the way to scan the image from the user device to the computer.

Moreover, we thought about different improvements to be made:

- ➢ Add some encryption and decryption methods in the QR code to keep user privacy safe. We tried to add this part, but we weren't able to fix some errors, so finally we omitted it.
- ➢ Connect this system with an automatic door to lock and unlock according to the answer given by the comparator.
- ➢ Improve the admission part dividing people in different timetables. For example having only access to a specific timetable of the corresponding subject.

RELEVANT REFERENCES

[1] Kalebu Jordan. Become a pro python developer(2017)

https://kalebujordan.com/reading-bar-codes-python/

[2] Adrian Rosebrock. An Open CV barcode and QR code scanner with ZBar.(2018)

https://www.pyimagesearch.com/2018/05/21/an-opencv-barcode-and-qr-code-scanner-with-zbar/

[3] Jae Hwa Chang. An introduction to using QR codes in scholarly journals. (2014)

https://www.researchgate.net/publication/271098121_An_introduction_to_using_QR_codes_in_scholarly_journals

[4] GitHub code(2020)

https://github.com/nayuki/QR-Code-generator

[5] Reading and Generating QR codes in Python using QRtools (2020)

https://www.geeksforgeeks.org/reading-generating-qr-codes-python-using-qrtools/

[6] James. How QR codes are built.(2020)

https://www.youtube.com/watch?v=142TGhaTMtI