IT203: DM Project Report: QR code implementation using Hamming Codes

Aadil Zubair Khalifa - 191IT101
Information Technology
National Institute of Technology Karnataka
Surathkal, India 575025
Email: aadilkhalifa@gmail.com

Harini Thirunavukkarasan- 191IT221
Information Technology
National Institute of Technology Karnataka
Surathkal, India 575025
Email: harinithiru.191it221@nitk.edu.in

Prasanthi Bolimera - 191IT240
Information Technology
National Institute of Technology Karnataka
Surathkal, India 575025
Email: prasanthibolimera.191it240@nitk.edu.in

I. Introduction

QR or quick response code is a 2D matrix code, that is used for storing data larger than its predecessor- the barcode, having features such as error correction and different encoding techniques. The idea behind the development of the QR code is the limitation of the barcode information capacity (can only hold 20 alphanumeric characters). The QR code was designed to allow its contents to be decoded at high speed. There are many different versions of the qr code ranging from the qr code version 1, i.e. a 21x21 grid having the lowest storage capacity, to much higher storage capacity qr code version 40 with a 177x177 grid. QR codes are designed to handle error correction in case the code gets damages and the level of error correction depends on the length of the data being stored.

The code consists of black modules arranged in a square pattern on a white background. The QR code can be thought of as a sequence of binary numbers inserted into a 21x21 grid (in case of version 1 QR code), in a particular format. The black modules represent '1' bit and the white modules represent '0' bit.

Fig. 1: Different QR code versions.

While they were developed for tracking parts in vehicle manufacturing, QR codes now are used in many other fields, from commercial tracking to entertainment, in-store product labeling, and in those applications that are aimed at smartphone users. Users may open URL; receive text after scanning QR codes. By using QR code generating sites or apps, users can generate and print their own QR codes for others to scan and use.

The QR code system consists of a QR code encoder and decoder. The encoder is responsible for encoding data and generation of the QR Code, while the decoder decodes the data from the QR code.

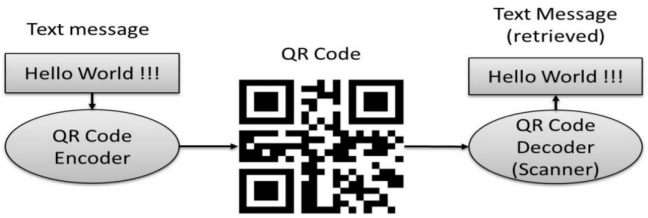

Fig. 2: Working of QR codes.

II. Work Done

We have chosen to work with QR code version 1, that deals with a 21x21 grid. We chose byte encoding as our primary encoding mode. We have chosen hamming code error correction as our error detection and correction method.

A. Hamming Code

1) Hamming code: It is a linear error detection algorithm that is capable of detecting up to two simultaneous bit errors and correcting single-bit errors. In this coding method, the

source encodes the message by inserting redundant bits/parity bits within the message.

- A parity bit is a bit appended to a data of binary bits to ensure that the total number of 1's in the data is even or odd. There are two types of parity bits: Odd and Even parity bits. In this project, we have implemented using Even parity bit.
- Even parity: In the case of even parity, for a given set of bits, the number of 1's are counted. If that count is odd, the parity bit value is set to 1, making the total count of occurrences of 1's an even number. If the total number of 1's in a given set of bits is already even, the parity bit's value is 0.

Positions whose iterations are powers of 2 are considered as Parity bits whereas the other positions are for data bits. For eg:- P1 P2 D3 P4 D5 D6 D7 P8 D9 .... where P represents parity bit and D represents data bit. Each data bit is included in a unique set of 2 or more parity bits, as determined by the binary form of its bit position.

Bit position		1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	
Encoded data bits		p1	p2	d1	p4	d2	d3	d4	p8	d5	d6	d7	d8	d9	d10	d11	p16	d12	d13	d14	d15	
Parity bit coverage	p1	✗		✗		✗		✗		✗		✗		✗		✗		✗		✗		
	p2		✗	✗			✗	✗			✗	✗			✗	✗			✗	✗		
	p4				✗	✗	✗	✗					✗	✗	✗	✗					✗	
	p8								✗	✗	✗	✗	✗	✗	✗	✗						
	p16																✗	✗	✗	✗	✗	

Fig. 3: Parity bit coverage

For example if the transmitted data is 1001, then it is stored as follows:

P1	P2	D3	P4	D5	D6	D7
0	0	1	1	0	0	1

where P1 = 0 since the total number of 1's in the positions D3, D5, D7 are even.
P2 = 0 since the total number of 1's in the positions D3, D6,D7 are even.
P4 = 1 since the total number of 1's in the positions D5, D6, D7 are odd.

2) Error Correction: During the transfer of qr code, it maybe subjected to damages hence one or two bits may be detected wrong. If the received data bits is 0011101 as per the previous example,
the error correction is done as follows,

P1	P2	D3	P4	D5	D6	D7
0	0	1	1	1	0	1

P1 D3 D5 D7 :- the number of 1's are not even hence P1=1 (error detected);
P2 D3 D6 D7 :- the number of 1's are even hence P2=0;
P4 D5 D6 D7 :- the number of 1's are not even hence P4=1 (error detected);

Therefore the error is in position, 101(Binary) – 5th bit (Decimal); Hence we flip the D5 bit and the corrected data bits is

P1	P2	D3	P4	D5	D6	D7
0	0	1	1	0	0	1

3) Types of Hamming code: There are different types of hamming code,

Parity bits	Total bits	Data bits	Name	Rate
2	3	1	Hamming(3,1) (Triple repetition code)	1/3 ≈ 0.333
3	7	4	Hamming(7,4)	4/7 ≈ 0.571
4	15	11	Hamming(15,11)	11/15 ≈ 0.733
5	31	26	Hamming(31,26)	26/31 ≈ 0.839
6	63	57	Hamming(63,57)	57/63 ≈ 0.905
7	127	120	Hamming(127,120)	120/127 ≈ 0.945
8	255	247	Hamming(255,247)	247/255 ≈ 0.969

Fig. 4: Possible Hamming codes.

B. For Encoding the QR code

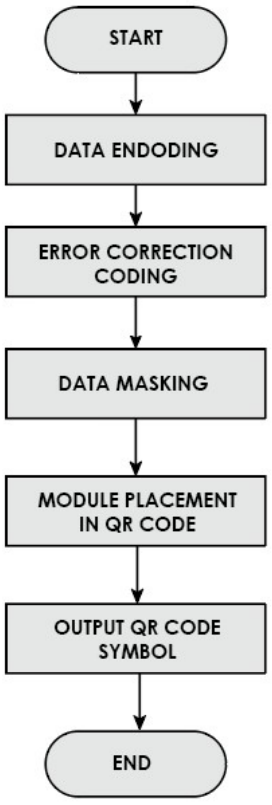

Fig. 5: QR code encoding.

1) **Data Encoding:** The inputted data is encoded using byte encoding, that is, each character that is present is converted to its Unicode value and that value is converted into 8-bit binary number. Hence using byte encoding mode, each character is encoded using 8 bits. The encoding type is specified using 4-bits and the length of the input data is specified using 8-bits.
The encoded data is formatted as follows:
Encoded data before error correction = encoding type (4-bit) + data length (8-bit) + byte encoding of data (8n-bits)

2) **Error Correction Coding:** Now that the data has been encoded using byte encoding, we need to add some error correction features, in case the QR code gets damaged. For this, we apply hamming code to this encoded data. The choice of hamming code is chosen based on the size of the data to maximize the error correction possible in the QR code. So, based on the length of the encoded data, 15-bit, 7-bit or 3-bit hamming code is applied on the encoded data where 3- bit hamming code has the max error correction level while it also consumes a huge amount of space for applying the hamming code.

3) **Data Masking:** Certain patterns in the QR code matrix can make it difficult for QR code scanners to correctly read the code. The data is masked to avoid regular patterns.

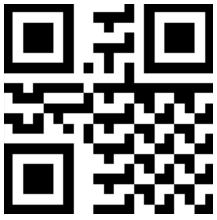

Fig. 6: One of the mask patterns used.

4) **Module Placement in QR code:** Here the data is structured and arranged in the appropriate QR code specification. The type of hamming code used and the type of masking used is specified within the format information locations. The required finder patterns and timing patterns are added.

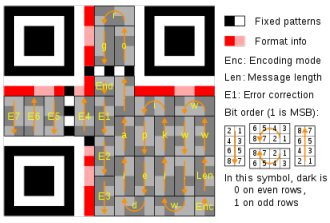

Fig. 7: Bit arrangement of a QR code.

C. For Decoding the QR code

Fig. 8: QR code decoding.

1) **Recognizing Modules:** Recognize black and white modules as an array of “0” and “1” bits and arrange the array in the appropriate order based on the QR code specification.

2) **Extract Format Information:** From the QR code get the type of masking performed on the data, and the type of hamming code applied on the data.

3) **Release Masking:** Based on the type of masking, the masking can be carried out again on the data to undo the masking.

4) **Error Detection and Correction:** The data that we have now has hamming code applied to it. So, using the rules of hamming code we can check if there are any errors present in the data, and if so, we can subsequently correct them. Then the data without the hamming code component can be taken.

5) **Error Decode Data:** According to the encoding type, which is specified in the first 4-bits of the encoded data,

and the length which is specified in the next 8-bits of the encoded data, the original message can be decoded using the appropriate method.

III. Results and Analysis

Here, we have successfully implemented, using python, the encoding and decoding of a version 1, 21x21 grid QR code using hamming codes for error correction. The extent of error correction possible is determined by the length of data that is being stored.

From the graph plotted we can clearly see, that the error correction that's possible is more when the length of the input data is less. This is because, when it input data is less, there is more space for error correnction to be added and hence a lower bit hamming code is applied which results in more possible error correction.

Fig. 9: QR code for "www.google.com".

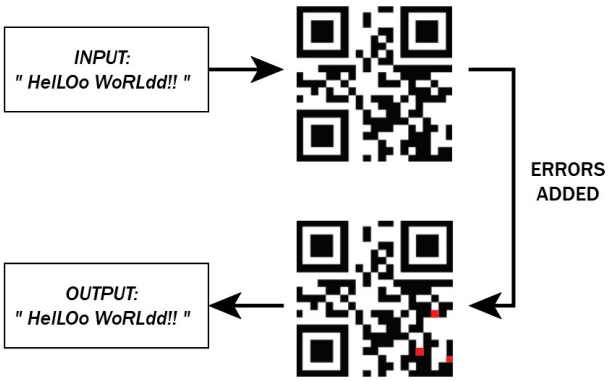

Fig. 10: Testing the error correction of the QR code by manually adding error.

Number of characters	Max bit corrections possible
1	20
2	28
3	36
4	44
5	52
6	60
7	68
8	19
9	21
10	23
11	25
12	27
13	29
14	12
15	12
16	13

Fig. 11: Comparisons between number of bit corrections possible vs number of characters.

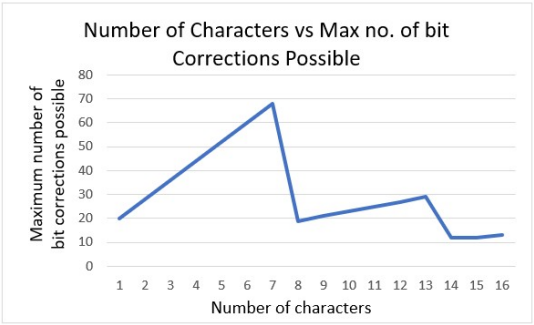

Fig. 12: Graph plotted for number of bit corrections possible vs number of characters.

IV. Conclusion

In this paper, we studied QR code technology, its benefits, application areas, and its impact on marketing and technological world. Initially, they were developed and used for inventory tracking stuff but these they find applications in many areas like marketing, advertising, education industries, contact-less payments, etc.

Adoption of the QR codes has been growing rapidly since past few years and its applications has increased wide and the number of users also increased exponentially. This is due to its features like easy and fast scanning, high capacity of data storage, error-correction and direct marking. It also has many placement opportunities like can be printed on different materials or displayed digitally. These days, most smartphones have in-built QR scanner which makes it even more easy to use the codes. This technology is anticipated to grow in the following years.

V. Implemented/Base paper

An Introduction To QR Code Technology by Sumit Tiwari,Dept. of Technical Education, SITS Educators Society, International Conference on Information Technology in 2016

VI. References

- https://en.wikiversity.org/wiki/Reed–Solomon_codes_for_coders
- https://ieeexplore.ieee.org/document/8735663
- https://en.wikipedia.org/wiki/QR_code
- https://en.wikipedia.org/wiki/Hamming_code