UNIVERSIDADE D
COIMBRA

Rafael Gameiro Marques

Method for the rectification of 2D barcodes in curved surfaces

The use cases of QR Code and UniQode

Dissertação no âmbito do Mestrado em Engenharia Electrotécnica e de Computadores orientada pelo Professor Doutor Nuno Miguel Mendonça da Silva Gonçalves e apresentada ao Departamento de Engenharia Electrotécnica e de Computadores da Faculdade de Ciências e Tecnologia da Universidade de Coimbra.

Julho de 2023

1 2  9 0

UNIVERSIDADE D
COIMBRA

Method for the rectification of 2D barcodes in curved surfaces – the use cases of QR Code and UniQode

Supervisor:

Doutor Nuno Miguel Mendonça da Silva Gonçalves

Coimbra, July 2023

Acknowledgments

I would like to express my deepest gratitude to my family and friends for their unwavering support and encouragement throughout my academic journey. Your love, understanding, and belief in me have been invaluable, and I am truly grateful for having you by my side.

I would also like to extend my sincere appreciation to my teacher Nuno Gonçalves, for his guidance, patience, and support throughout the creation of this thesis.

Thank you.

Abstract

Two-dimensional machine readable codes, such as the QR Code, have become an essential part of several industries due to their very good portability and the ability to effectively store data. These codes are used in a variety of applications, such as advertising, inventory tracking, mobile payments and even in security applications. However, the decoding of such codes can be challenging when the codes are distorted, for example, due to printing errors, camera angle, or lens distortion.

One particular type of distortion that can affect these codes is cylindrical distortion, which occurs when a code is printed or placed on a cylindrical surface, such as a can or bottle, causing the code to be distorted when viewed through a camera lens. This distortion can make it difficult for code scanners to decode the information stored in the code and is amplified when the camera is not perpendicular in relation to the code.

In recent years, different types of these two-dimensional machine readable codes emerged, each with their particular advantages and disadvantages. In this thesis we explore a method to correct the distortion in cylindrical surfaces of the QR Code, which is the most known and used two-dimensional code, and the Graphic Code (UniQode system), a new code developed by researchers from the Institute of Systems and Robotics of University of Coimbra.

The method proposed is tested on two different datasets that cover virtual and real-life scenarios codes with cylindrical distortion. To replicate real-life situations more accurately and increase the difficulty level, we also introduce different angles of slope, rotation, and tilt during the testing process. The results of this research can have significant practical applications, especially in industries that heavily rely on QR codes for inventory management and tracking. A more efficient and accurate method for decoding cylindrical distorted QR codes can lead to improved productivity and cost savings for businesses.

Resumo

Os códigos bidimensionais de leitura ótica, tais como o QR Code, tornaram-se uma parte essencial de várias indústrias e mesmo da nossa vida devido à sua portabilidade e grande capacidade de armazenar dados de uma forma eficaz. Estes códigos são utilizados numa variedade de aplicações, como em publicidade, em controlo de inventário, em pagamentos móveis e até em aplicações de segurança. No entanto, a descodificação destes códigos é por vezes um desafio quando os códigos estão distorcidos devido a, por exemplo, erros de impressão, ao ângulo da câmara ou da distorção da lente.

Um tipo particular de distorção que pode afetar estes códigos é a distorção cilíndrica, que ocorre quando um código é impresso ou colocado numa superfície cilíndrica, como uma lata ou garrafa, criando uma distorção no código quando visto através da lente de uma câmara. Esta distorção pode dificultar a descodificação da informação armazenada no código por leitores de códigos e é amplificada quando a câmara não está perpendicular ao código.

Nos últimos anos, surgiram diferentes tipos destes códigos bidimensionais de leitura ótica, cada um com as suas vantagens e desvantagens específicas. Nesta tese exploramos um método para corrigir a distorção em superfícies cilíndricas do QR Code, que é o código bidimensional mais conhecido e utilizado, e do Graphic Code (sistema UniQode), um novo código desenvolvido por investigadores do Instituto de Sistemas e Robótica da Universidade de Coimbra.

O método proposto é testado em dois conjuntos de dados diferentes que abrangem códigos em cenários virtuais e reais com distorção cilíndrica. Para reproduzir situações reais com maior precisão e aumentar o nível de dificuldade, introduzimos também diferentes ângulos de inclinação e rotação durante o processo de teste. Os resultados desta investigação podem ter aplicações práticas significativas, especialmente nas indústrias que dependem fortemente dos códigos QR para a gestão e rastreio de inventários. Um método mais eficiente e preciso para descodificar códigos QR cilíndricos distorcidos pode levar a uma maior produtividade e a poupanças de custos para as empresas.

Contents

List of Figures

List of Tables

1 Introduction

1.1 Background and Motivation

Machine readable codes, or MRCs for short, have become increasingly popular in recent years due to their ability to efficiently store and transfer large amounts of data. The first MRC that gained widespread adoption was the UPC[1], a one-dimensional barcode which can store small amounts of data and is very rapidly decoded. As a result, it is now extensively utilized for labelling products in various industries (Fig. 1.1).

Figure 1.1: The most widely used 1-D Barcode, the EAN-13 [1].

Later, two-dimensional codes were introduced like the QR Code[4], DataMatrix[5] and Aztec Code[6]. Some of these codes, like the QR Code not only can store more information, but also come with in-built error correction. In this thesis we will focus on two-dimensional barcodes: the QR Code and the Graphic Code (UniQode)[7], that can be scanned using a smartphone camera or a dedicated reader. Once scanned, the code can direct the user to a website, display product information, provide contact details, or even display alphanumeric or binary information.

One of the primary reasons for the widespread use of these codes is their versatility, they can be used in a variety of settings and for a wide range of purposes, from marketing and advertising to even certify the authenticity and traceability of objects (e.g. wine bottles, tobacco products).

Over time, different types of machine-readable codes have emerged, they are similar to the QR Code but are designed to be visually appealing and integrate with a company's branding or marketing materials. They typically have a more customized design and may incorporate

(a) Aztec Code (b) DataMatrix (c) Graphic Code (UniQode)

Figure 1.2: Different types of 2-D Machine Readable Codes

images, logos, or other graphics and even the capacity to store more information. One example of this is Graphic Codes which use UniQode, which are more visually appealing and customizable while also increasing their storage capacity in relation to other MRCs [7].

However, with their increased prevalence comes the risk of deformities in the codes, which can hinder their functionality and reliability. In order to improve the decoding rate of deformed QR codes, it is necessary to apply pre-processing techniques to correct them.

This thesis focuses on exploring a method to correct QR codes that have a cylindrical deformation, which typically occurs when they are posted to cylindrical surfaces.

Utilizing Computer Vision techniques, this method has the capability to enhance the decoding rate not only for QR codes but also for other two-dimensional barcodes.

The latest smartphones with iOS 13 and above and Android 9 and above come equipped with advanced QR Code readers. These readers are already very powerful and able to successfully correct and detect QR Codes which have some type of distortion, however they lack the ability to detect different MRCs like the Aztec Code and UniQode, and their code is proprietary which means that very little documentation can be found of their implementation. In this thesis, a methodology is presented that aims to replicate the outcomes achieved by these advanced QR code readers, whilst also being able to interpret additional 2-D MRCs.

1.2 Objectives

For the study of the new proposed method we set several main objectives:

- Creation of a big and robust dataset that accurately represents real life scenarios with cylindrical distortion in combination with sloping and tilting.
- Adapt and improve a cylindrical distortion correction method to increase decoding rate.

- Adapt and improve a cylindrical distortion correction method to work with multiple two dimensional machine readable codes like the Graphic Code (UniQode).
- Study the effects on the decoding rate of this distortion with a basic and an advanced QR Code Reader using the new dataset
- Using the created dataset, obtain comparison results between basic/advanced QR code readers and ours.

1.3 Scientific Contributions

The main contributions of this thesis can be summarized as follows:

- **Method for Cylindrical Distortion correction:** a new method is introduced based on previous work, that is designed specifically to address the challenge of cylindrical distortion in two-dimensional machine-readable codes. The proposed method explores traditional approaches and leverages advanced image processing techniques and geometric transformations to accurately correct the distortion and improve decoding rate.
- **Open Source Nature:** One of the key contributions of this research is its open source nature. Most widely used QR Code decoders are the ones we have available on our smartphones, and most of them already possess some type of pre and post processing that corrects some distortions, however the methods used are proprietary.
- **Diverse Datasets:** The proposed method is rigorously evaluated through extensive experiments using diverse datasets containing cylindrical distorted codes. A synthetic and a Real Images dataset was created, on which a wide range of different angles and cylindrical distortion rates were tested in order to properly evaluate our method.
- **Application Potential and Impact:** The correction of cylindrical distortion in two-dimensional machine-readable codes has wide-ranging practical applications. Industries such as logistics, manufacturing, retail, and transportation heavily rely on machine-readable codes for efficient data capture and processing. The improved detection rate achieved through this research has the potential to improve the reliability and usability of codes applied on cylindrical surfaces.

In summary, this thesis makes relevant scientific contributions by introducing a novel open source method for correcting cylindrical distortion in two-dimensional machine-readable codes.

The method's effectiveness in improving the detection rate, its open source nature, and the extensive experimental evaluation demonstrate its practical applicability and potential impact.

1.4 Organization of Dissertation

This thesis is organized in 7 different chapters. In Chapter two a review is done in machine readable codes, mainly focusing on QR Codes and Graphic Codes (UniQode), where we give an overview of their history, their functionalities and the differences between them. In Chapter 3 an extensive review of the existing state of art is done, focusing on some different QR Code localization methodologies and existing methods to correct cylindrical distortion on QR Codes. This chapter is followed by Chapter 4 where we give a brief overview of the existing method we adapted and explain section by section the methodology behind our proposed method. After our proposed method is explained in depth, Chapter 6 gives us a detailed overview of how we built our datasets, detailing datasets composition and size and the different components we used to create it. To analyze our proposed method, in Chapter 6 we present the results obtained after extensive testing the method with the datasets we created, and finally in Chapter 7, a final overview of the work is done and possible future work is discussed.

2 Materials

2.1 QR Codes

2.1.1 History of QR Codes

QR code, short for Quick Response code, is a type of two-dimensional barcode that was first invented in 1994 by a Japanese company named Denso Wave[8]. The company's aim was to create a more efficient way of tracking inventory for their automobile parts manufacturing process with the development focus being on high speed readability. The QR code is a variation of the traditional barcode, but it can store much more information, including both alphanumeric and binary data.

Over time, the use of QR codes spread to other countries and industries, particularly after receiving international ISO recognition in 2000. Since then, QR codes have been used for a variety of purposes, including advertising, ticketing, and identification.

Today, QR codes can be seen in many different areas from product packaging, advertising and to even authenticate the traceability of objects. With the rise of mobile technology, QR codes have become an essential tool for businesses and organizations looking to connect with their customers and provide them with quick access to information.

2.1.2 Structure of QR Code

The basic structure of a QR Code (Fig. 2.1) consists of an array of black and white squares arranged in a square matrix. The structure of a QR code can be categorized into two main sections: the encoding region and the function parameters.

The function patterns are used to locate and identify the QR code and consists of the finder pattern, separator, timing patterns, and alignment patterns. These patterns work together to ensure accurate scanning and decoding of the QR code.

The encoding region contains all of the actual data stored and respective error correction words which are used to recover the data when some part of the code is damaged, as well as

additional information such as the format and version of the code.

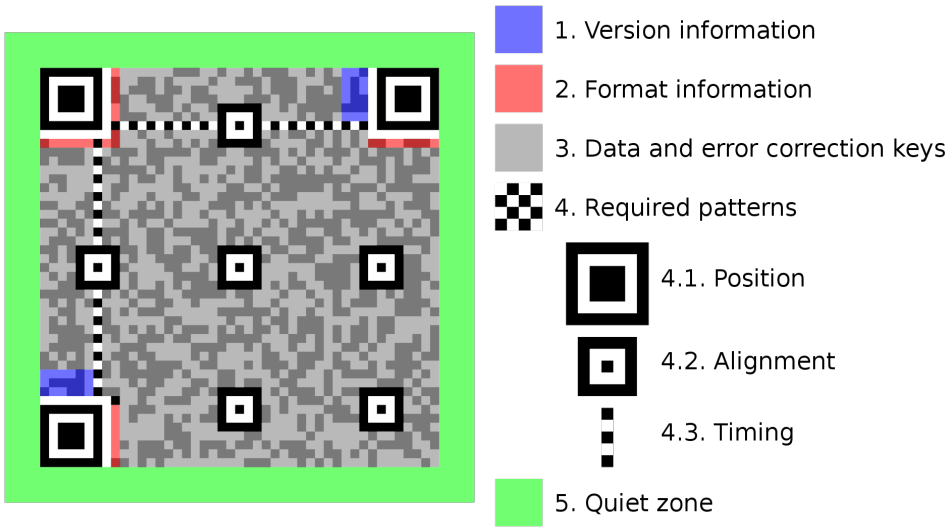


Figure 2.1: Structure of a QR Code.

- **Position Detection Patterns:** Position Detection Patterns, also known as Finder Patterns, were designed to maintain the same ratio regardless of the angle of scanning. This ratio is 1:1:3:1:1 (B:W:BBB:W:B) and it can be used for the fast detection of the QR Code.
- **Alignment Pattern:** This is an additional element which acts like a smaller Finder Pattern and it can be used to increase accuracy when detecting the orientation of the code, especially in large QR codes.
- **Timing Patterns:** The scanner relies on these markers to ascertain the dimensions of the data matrix, enabling it to pinpoint the central coordinates of the data cell even in the presence of errors.
- **Version Information:** These specify the used QR Code version, which can range from version 1 to 40, which are respectively $21 * 21$ modules to $177 * 177$ modules. Version 40 is able to store up to 7089 numeric characters.
- **Format Information:** The format patterns contain the error correction level of the QR Code. This can range from Low to High. QR codes that have a high error tolerance can correct data damage up to 30%.
- **Data and Error Correction Codes:** This area of the code is the one that actually holds the data of the QR code.
- **Quiet Zone:** The quiet zone is the white area around the QR Code, this spacing is very important for facilitating decoders to correctly locate the QR Code.

2.1.3 Error Correction Level

Error correction is an essential aspect of QR codes, which are designed to be highly reliable and readable even in less than ideal conditions. QR codes contain built-in error correction mechanisms that allow them to withstand damage or distortion that may occur during printing, scanning, or transmission.

QR codes use Reed-Solomon error correction codes, which allows for the detection and correction of errors within the code. Reed-Solomon codes are capable of correcting up to 30% of the code's data without losing any information (Fig. 2.2). This means that even if a QR code is partially damaged or distorted, it can still be successfully scanned and decoded, as long as enough of the code remains intact.

Error Correction Level	Error tolerance (%)
L (Low)	7%
M (Medium)	15%
Q (Quartile)	25%
H (High)	30%

Figure 2.2: Error Correction Levels.

When creating a QR code, the level of error correction can be adjusted to suit the intended use and the expected level of damage or distortion that the code may experience, however higher levels of error correcting may lead to the need of a bigger QR Code to accommodate more error correcting pixels. QR codes have four levels of error correction, ranging from Level L (low) to Level H (high) (Fig. 2.3), with each level offering progressively more robust error correction.

Figure 2.3: Error Correction in a QR code.

2.2 Graphic Codes (UniQode)

2.2.1 Creation of the Graphic Codes

In collaboration with INCM, a group of researchers from the ISR of University of Coimbra proposed a novel Machine Readable code called Graphic Code[7]. Like QR codes, Graphic Codes can be employed for advertising, ticketing, and identification of various products. This code utilizes a symmetric cryptographic system that utilizes a dictionary and a single key for encryption and decryption. The code can be encoded into pixel-based or icon-based images while maintaining decoding rate, this allows for more aesthetically pleasing codes.

2.2.2 Encoding and details

To create and encrypt the content Graphic Code (UniQode), each code uses a dictionary, which associates a symbol to a pattern (Fig. 2.4). Each symbol is assigned to different patterns of pixels, these can be of any size, however the most common pattern is a 3x3 cell.

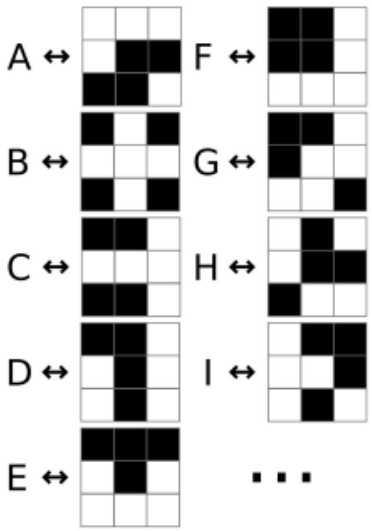


Figure 2.4: Example of a dictionary.

The process starts by encoding a quantum value (regular division of the grayscale range by 10, for a $3*3$ cell it's value is $0-9$) to each grid cell ($3*3$) of the base image. Then, based on the quantum value, candidate cells are chosen, these cells are the one which have enough quantities of different patterns to be used with the chosen dictionary. Then, one cell per character of the message is chosen and replaced by the respective pattern in the dictionary, while the remaining cells are all replaced using non-dictionary patterns according to their quantum value (Fig. 2.5).

(a) Base Image (b) Candidates (c) Chosen Cells (d) Graphic Code

Figure 2.5: Coding pipeline of enconding a Graphic Code (UniQode).

A singular Graphic Code (UniQode) can have more than one dictionary, this can be useful to restrict groups of users to certain information. Unlike the QR Code, a Graphic Code doesn't have any size restrictions, which means that even if, for the same size, a QR Code can contain more information, a Graphic Code has the potential for storing more data simply because it has no size limitations.

For the purpose of this thesis we will use Graphic Codes created using a black square as the

base image, these codes still follow the same encoding pipeline and creates a code that looks like the one in Fig.2.6 . This code still uses the same basic concepts of the UniQode system, it just doesn't take advantage of the ability to make the codes more visually appealing.

Figure 2.6: Example of Graphic Code which uses a black square as the base image.

3 State of Art

3.1 Localization Methods

In order to correct a QR Code, it is necessary to first correctly locate it. There has been extensive work on this subject, and the techniques used can be divided in three different localization methods.

3.1.1 Finder Pattern-Based Localization Methods

These localization methods use the unique ratio of the finder pattern (Fig.2.1), found in the QR Code to accurately find the code vertices. This ratio is defined in pixels as 1:1:3:1:1 (B:W:BBB:W:B), and it's true regardless of the scanning angle.

Lin and Fuh [9] propose that the binary image is systematically scanned both horizontally and vertically to identify specific points that adhere to the ratios of 1:1:3:1:1 and therefore find all potential finder patterns in the image. The angle between these is checked to find the three correct finder patterns.

In 2017 Tribak and Zaz[10] utilize a two-step approach to localize QR Code patterns. Firstly, preliminary localization is achieved through horizontal and vertical scans. Subsequently, the principal component analysis (PCA) method is employed to effectively eliminate any false positives.

In the research conducted by Li et al. [11], the connected components present in a binary image are compressed using run-length coding on a row-by-row basis. Subsequently, the typical ratio associated with finder patterns (1:1:3:1:1) are sought after in each row. Once these points are identified, a minimal containing region is established and a modified Knuth-Morris-Pratt algorithm is employed, allowing for efficient computation of the center coordinates of the finder patterns.

Belussi and Hirata [12] propose a methodology that involves utilizing a cascade of weak classifiers trained specifically on the finder patterns of the QR Code. The Viola-Jones' rapid

object detection method and Haar-like features, extracted using a floating window approach, are used in their approach. The angles between the detected candidate finder patterns are then calculated to assess if any subgroup of candidate finder patterns form a square.

3.1.2 Machine Learning Based Localization Methods

Some work has also been done using deep learning tecnhiques in order to correctly locate QR Codes. Hansen et al.[13] proposes an adaptation to a well know deep learning based object detection algorithm (YOLO) to detect barcodes.

Chou et al.[14] introduce an algorithm that uses convolutional neural networks and majority voting to correctly locate QR Codes. In 2019, Zharkov and Zagaynov [15], also propose a new deep learning detector based on semantic segmentation.

3.1.3 Localization Methods based on connected components

This last methodology takes advantage that the QR Code, like most other two dimensional machine readable codes consist of an array of black and white squares. Some work has been done using image morphology. In 2013, Kong et al. [16] used Harris Corner Detection method in junction with the convex hull algorithm to get an accurate outline of a QR Code. Gaur and Tiwari[17] propose the use of a Canny Edge detector followed by dilation operation to get the minimal rectangle containing the QR Code.

3.2 Existing methods for Geometric Correction

After locating the QR Code, it still needs to be correctly decoded and when a QR code is on a non-planar surface, such as a curved surface, it needs to be corrected in order to achieve accurate recognition. So far, only a small number of research work tackle this problem especially when the QR Code is posted on a cylindrical surface. In 2013, K. Suran et al. [16] proposes a method which uses the convex hull algorithm. Firstly, Suran uses local threshold and mathematical morphology methods to binarize the QR code image with uneven light. Then, the outline of the QR code is found and the distorted image is recovered by using perspective collineation. Finally, the convex hull algorithm is applied for the geometrical correction of image. M. Li et al. [18] proposes the pre-processing of the image using the image gray feature to then get accurate vertices and apply accurate image correction based on projection transformation. This work is particularly notable because not only it is able to tackle the angular distortion of the image but also the light distortion. In 2015, K. Lay et al. [2], uses a new method based on Conic

Segmentation to solve this problem and successfully increases the probability of success in the decoding of QR code from a common QR Reader.

4 Methods

4.1 Existing method to correct cylindrical distortion in QR Codes – Conic Segmentation

In 2015, K. Lay et al. [2] proposed a conic segmentation (CS) scheme to correct the distortion in QR images that are posted on cylindrical surfaces. The CS method involves using conic curves to segment the distorted QR image into an array of either more white or more black cells. The author proposes an adaptive method that uses two boundaries of the QR Code to create two conic curves, these curves are then thickened by the technique of dilation and iterated over the entire code. In this iteration the author groups all edge points that intersect with the thickened curve until this curve is not intersecting any edge point, using the edge points of the group a new curve is formed and is used to iterate over the next edge points as seen in Fig 4.1.

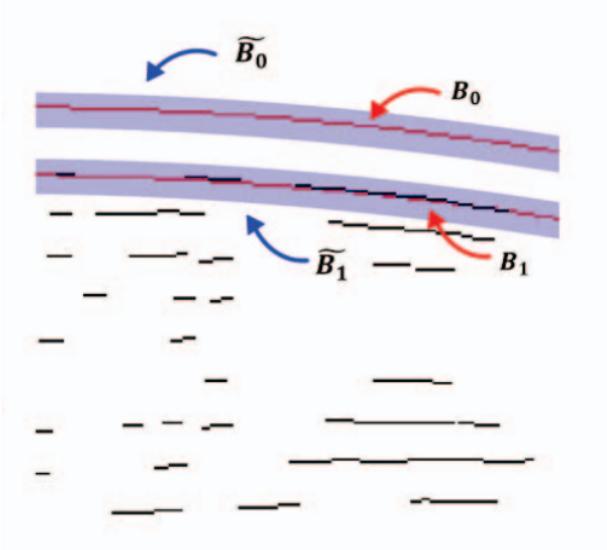


Figure 4.1: Conic segmentation - Edge hitting[2].

In his method these conic curves change shape during this process to adapt to the edges of the QR Code. With these two conic curves mapped to the QR code a matrix can be arranged

and cells formed, these cells are then converted into pure white or pure black modules and mapped onto a blank QR template. (Fig. 4.2) The proposed scheme utilizes various digital image processing techniques to implement this rectification process. Experimental results demonstrate that the incorporation of this CS-based rectification significantly improves the success rate of QR Code decoding for images posted on cylindrical surfaces in a small database of synthetic QR Codes.

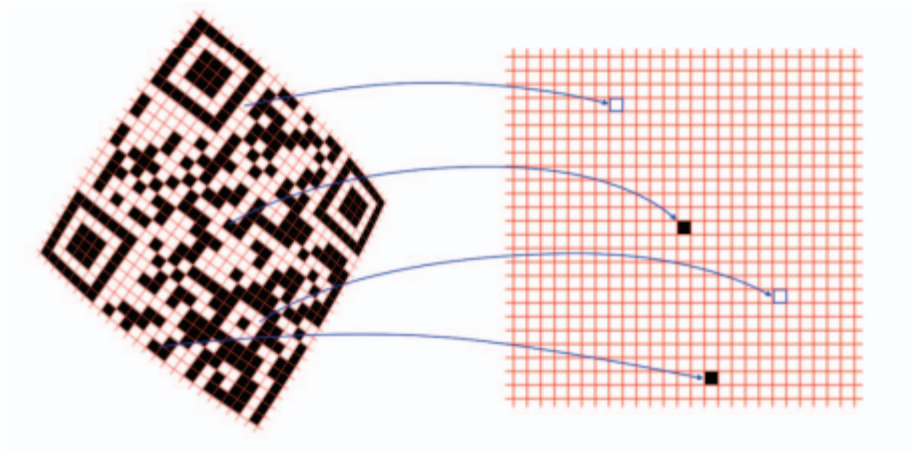


Figure 4.2: Conic segmentation [2].

However, since the proposed method is dependent on edge hitting it is very sensitive to noise, and the authors have only tested it on a synthetic database of small size (100 samples). Since the method is also reliant on the Position Detection Patterns it is not able to read other 2-D barcodes (Graphic Code, DataMatrix, Aztec Code, etc). In this thesis we propose changes to the method to increase robustness to noise and stains, and also changes to the localization method of the code so it can read other MRCs like the Graphic Code.

4.1.1 Overall overview of changes to method

Since the creation of the QR code, several other 2-D MRCs were created, this other 2D Codes don't have Positional Detection Patterns like the QR Code, however, what almost all of them have in common is that they consist of an arrangement of small black and white squares or pixels. Knowing this we propose changes to find accurate vertices of the code by using morphological erosion and dilation to get the four corners of the code, instead of using the Positional Detection Patterns that are exclusive to the QR Code. With the corners determined we can apply Projective Transformation (PT) to align any rotating deformation the code might have. After applying PT to the code, we extract the four boundaries (top, bottom, left and right) which should be roughly aligned with the horizontal and vertical axis after the PT, and use them to

apply an altered conic segmentation algorithm. This altered algorithm uses four boundaries of the code (top, right, bottom and left boundaries) to create four conic curves which are then mapped to the code without any adaptation. With the conic curves mapped, the code is divided in four sections, and for each section a matrix composed of cells is formed using the mapped conic curves. These cells are then converted into pure white or pure black modules and mapped onto a blank map. The four sections are then stitched together to create the final corrected code. The workflow of the method is shown in Fig. 4.3.

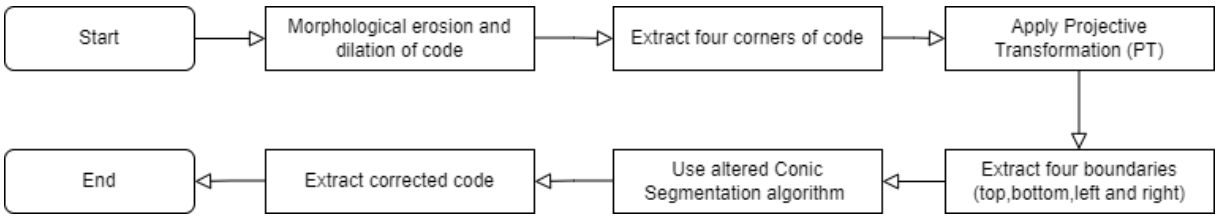


Figure 4.3: Flowchart of proposed method.

4.2 Pre-Processing of the QR Code

To obtain an accurate QR code image from a photo taken by a mobile phone or camera, it is necessary to address the issue of background interference. Although these images capture the QR code pattern, they also include the background image, which must be removed to facilitate the acquisition of the QR code's four vertices. In order to extract the QR code from the background image, various pre-processing techniques are utilized, including greying the image, binarizing the image, applying dilation and erosion operations, and eliminating irrelevant connected areas. The workflow of this pre-processing is shown in Fig. 4.4

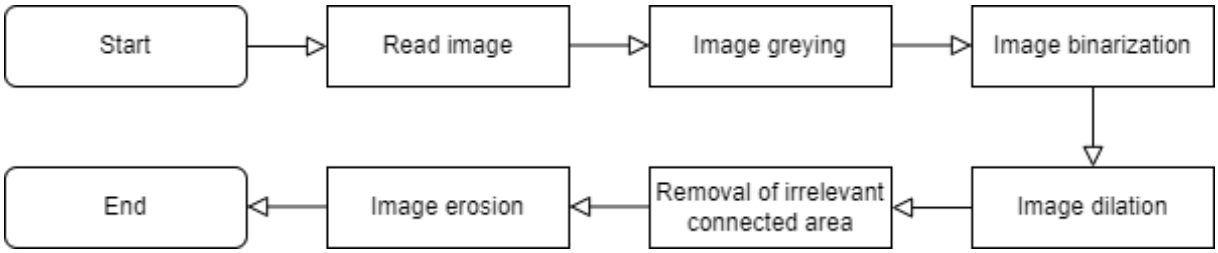
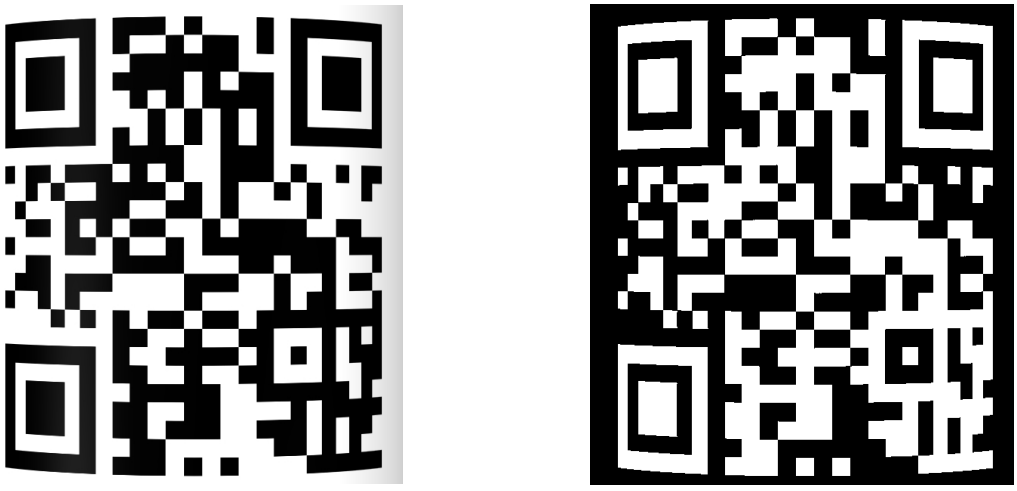


Figure 4.4: Flowchart of the pre-processing operation.

4.2.1 Converting image to grayscale and Binarization

A typical image captured by a smartphone is stored as an RGB image. An RGB image consists of three channels: red, green, and blue. By converting the image to grayscale we transform the original image with 3 colours, into a grayscale version that only includes shades of grey.

Converting images to black and white through binarization simplifies the process of analysing images, facilitating the identification of features in the image, as well as subsequent operations such as edge detection. Furthermore, because binary images require fewer computations when compared to grayscale or colour images, binarization can significantly speed up image processing operations. To binarize the image the threshold for each pixel is computed using the local mean intensity around the neighbourhood of the pixel, this technique is also known as Bradley's method [19].



(a) Before Binarization (b) After Binarization

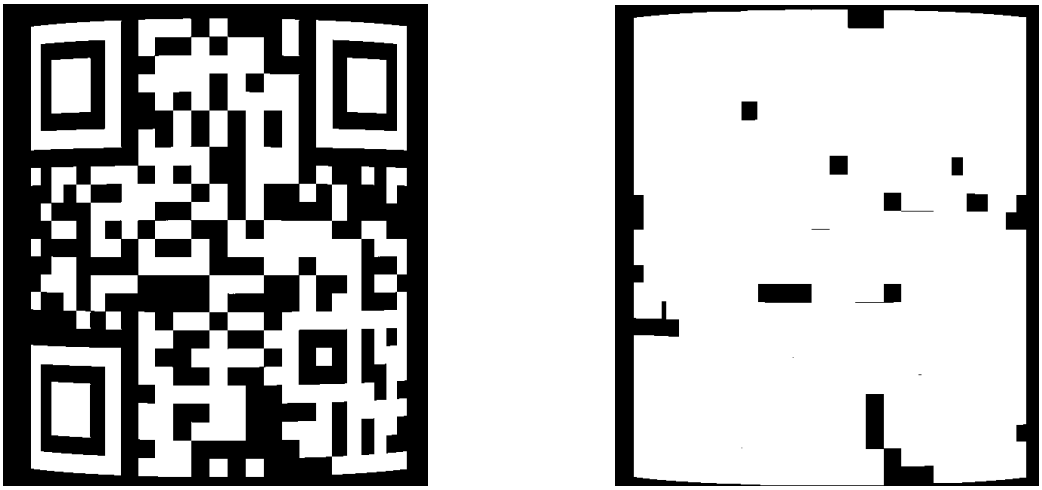
Figure 4.5: The process of binarization

4.2.2 Image Dilation

Since QR Codes consist of small white and black squares and image dilation is a morphological operation used to fill small holes and narrow gulfs in objects, using the inverse of our binarization we can see the black squares as small holes and use the technique of image dilation to fill them. Image dilation (usually represented by $\oplus$) is one of the basic operations in mathematical morphology[20]. The dilation operation usually uses a structuring element for probing and expanding the shapes contained in the input image.

$$A \oplus B = \bigcup_{b \epsilon B} A_b \tag{4.1}$$

The binary dilation of A by B is denoted as A $\oplus$ B and is defined in equation 4.1. This operation will, for each pixel in A that has a value of 1, superimpose B, with the center of B aligned with the corresponding pixel in A. For the purpose of this thesis, the optimal size of B is a 25 by 25 square structuring element and A is our input image and an example of the results obatined after this operation is shown in Fig. 4.6



(a) Before (b) After

Figure 4.6: The process of image dilation

4.2.3 Removal of irrelevant connected area

After image dilation there are some isolated areas remaining inside the dilated square. These areas are removed using an algorithm based on morphological reconstruction[21] using pixel connectivity.

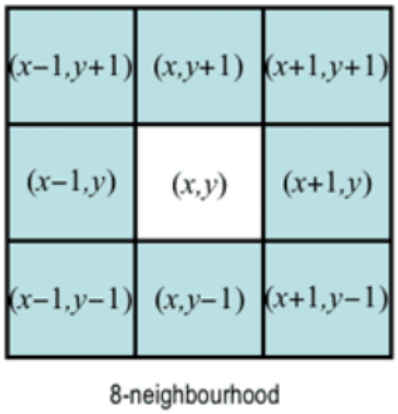
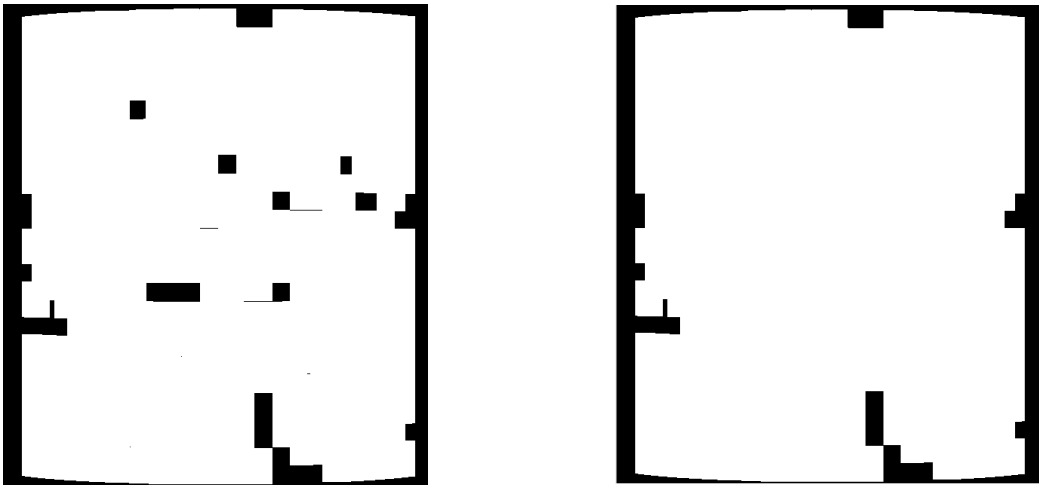


Figure 4.7: 8-neighbourhood example.

The method employed is an 8-neighbour approach, whereby pixels are considered connected if they are situated one pixel of distance in the horizontal, vertical, or diagonal directions, as illustrated in Fig. 4.8.



(a) Before (b) After

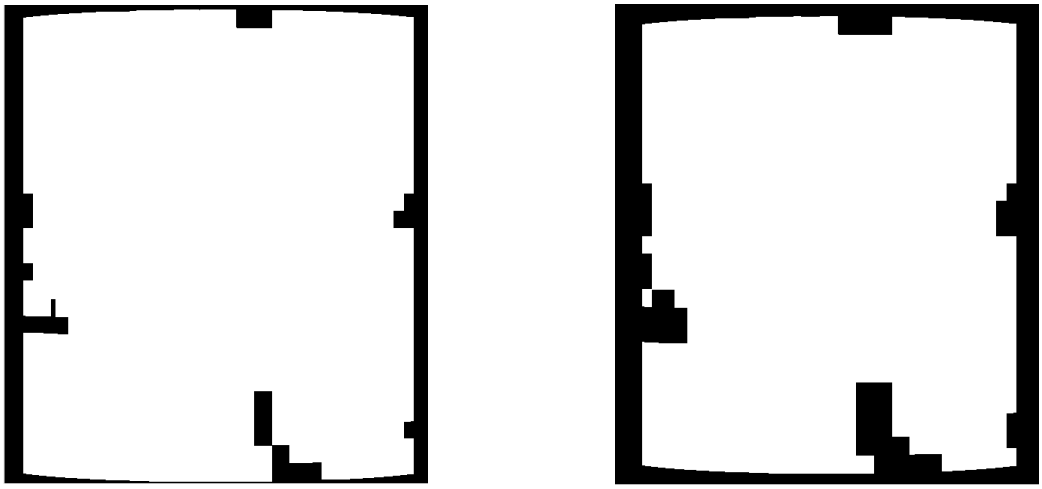
Figure 4.8: Removing irrelevant connected area

4.2.4 Image Erosion

To preserve the size of the original code which has been enlarged by image dilation, image erosion is applied. Image erosion[20], represented by $\ominus$, is one of two fundamental operations (the other being dilation) in morphological image processing from which all other morphological operations are based. The erosion operation uses a structuring element for probing and reducing the shapes contained in the input image [12].

$$A \ominus B = \bigcap_{b \epsilon B} A_{-b} \tag{4.2}$$

The erosion of the binary image A by the structuring element B is defined in equation 4.2. This operation will, for each pixel in A, superimpose the origin of B, if B is completely contained by A the pixel is retained, else deleted. The size of the structuring element is the same as used in the process of image dilation to maintain the correct Code size.



(a) Before (b) After

Figure 4.9: The process of image erosion

4.2.5 Finding QR Code vertices

Once the image of the code has been pre-processed, obtaining the four accurate vertices, C_1, C_2, C_3, C_4, can be achieved with relative ease by utilizing a corner detection technique such as the Harris Corner method or a similar approach. By detecting the majority of corner points and calculating the four outermost corners, the vertices of the code can be accurately determined.

4.2.6 Using projective transformation to correct tilting deformation

Most code images will have some sort of tilting and/or rotation related distortion as they are usually captured by mobile cameras, often from smartphones. To correct this distortion and align the four captured vertices in the horizontal and vertical axis, projective transformation is used. Projective transformation can be used to rectify images of planar scenes (e.g., building facades) to frontoparallel view[22]. Equation 4.3 demonstrates the process of perspective transformation, where (u, v) represents the coordinate of the original captured image, (u', v') represents the new coordinates of the QR code vertices after the transformation and H is the homography matrix.

$$\alpha \begin{bmatrix} u' \\ v' \\ 1 \end{bmatrix} = \begin{bmatrix} h_{11}\, h_{12}\, h_{13} \\ h_{21}\, h_{22}\, h_{23} \\ h_{31}\, h_{32}\, h_{33} \end{bmatrix} \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} \tag{4.3}$$

Since the homography matrix has 8 degrees of freedom, we need at least four pairs of corresponding points to solve for the values of h_{11}, h_{12}, h_{13}, h_{21}, h_{22}, h_{23}, h_{31}, h_{32}. Using the points calculated on 3.2.2, and writing the problem as a linear system of equations, we reach equation 4.4 .

$$\begin{bmatrix} u_1 & v_1 & 1 & 0 & 0 & 0 & -u'_1u_1 & u'_1v_1 \\ 0 & 0 & 0 & u_1 & v_1 & 1 & -v'_1u_1 & -v'_1v_1 \\ u_2 & v_2 & 1 & 0 & 0 & 0 & -u'_2u_2 & u'_2v_2 \\ 0 & 0 & 0 & u_2 & v_2 & 1 & -v'_2u_2 & -v'_2v_2 \\ u_3 & v_3 & 1 & 0 & 0 & 0 & -u'_3u_3 & u'_3v_3 \\ 0 & 0 & 0 & u_3 & v_3 & 1 & -v'_3u_3 & -v'_3v_3 \\ u_4 & v_4 & 1 & 0 & 0 & 0 & -u'_4u_4 & u'_4v_4 \\ 0 & 0 & 0 & u_4 & v_4 & 1 & -v'_4u_4 & -v'_4v_4 \end{bmatrix} \begin{bmatrix} h_{11} \\ h_{12} \\ h_{13} \\ h_{21} \\ h_{22} \\ h_{23} \\ h_{31} \\ h_{32} \end{bmatrix} = \begin{bmatrix} u'_1 \\ v'_1 \\ u'_2 \\ v'_2 \\ u'_3 \\ v'_3 \\ u'_4 \\ v'_4 \end{bmatrix} \tag{4.4}$$

This form allows us to solve for the H vector using the Least Square method, and subsequently improve the estimation using any robust estimation for this linear system of equations. With the homography matrix calculated the rest of the points can be calculated using interpolation and any tilting and/or rotation deformation on the code image is corrected.

4.3 Conic Segmentation

4.3.1 Basic idea behind conic segmentation

The idea of conic segmentation comes from fitting conic curves to the code in order to create a matrix that, for each cell, contains the entirety of the black or white square from the code. As the code is distorted these squares are deformed and they normally are a quadrilateral of various forms depending on the amount and type of distortion and therefore the use of straight lines to create this matrix is not possible. Instead, the boundaries, B_T, B_B, B_L, B_R, will be used as the base for the construction of our matrix.

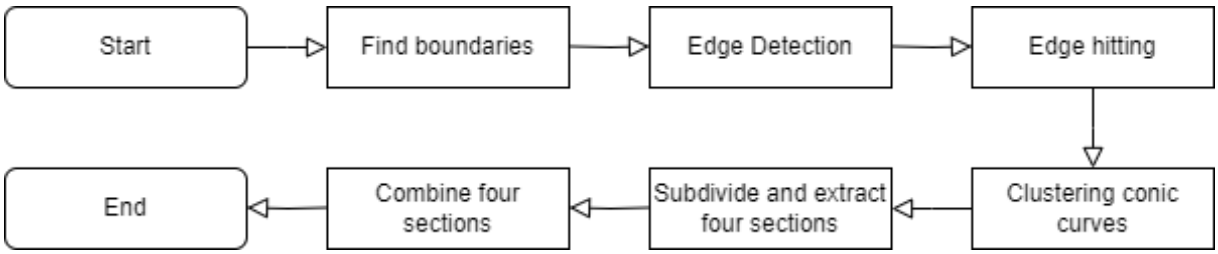


Figure 4.10: Flowchart of conic segmentation process.

4.3.2 Finding boundaries

With the code roughly aligned with the horizontal and vertical axis, the same methodologies used in sections 3.2.1.3-3.2.1.5 are applied to the now rectified image from section 3.2.3, this means using basic mathematical morphology of image dilation and erosion and the removal of

irrelevant connected areas to get a rough outline of the QR Code. Using this outline, the process from 3.2.2 is repeated and we obtain the 4 vertices, C_1, C_2, C_3, C_4. To get approximate values for the top, bottom, left and right boundaries, B_T, B_B, B_L, B_R, we select a small area in the middle of each pair of corner points and find the pixel farthest from the center of the QR Code, which we will call as auxiliary points, A_{12}, A_{23}, A_{34}, A_{41}.

Figure 4.11: Middle area (in blue) and calculaled top pixel (in red).

In Fig.4.11 this process is shown for the top boundary, where a middle area is determined (in blue) by dividing the distance between the two top corners, C_1, C_2, and the top-most pixel is selected as the auxiliary point, A_{12}. This process is repeated for all four corner points pairs to get four auxiliary points. To get approximate boundaries a 2-nd degree polynomial is fitted between each corner point pair and their corresponding auxiliary point, see Fig4.12.

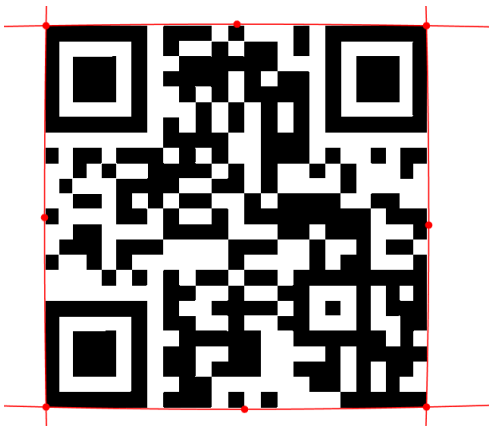


Figure 4.12: Four calculated boundaries.

4.3.3 Edge Detection

With the QR Code roughly aligned in the horizontal and vertical axis, the use of the sobel operator (or other edge detection method) can be used to highlight lines aligned in the vertical and horizontal axis. The operator consists of a pair of 3×3 convolution kernels as shown in Fig. 4.13. One kernel is simply the other rotated by 90°.

$$G_x = \begin{bmatrix} 1 & 0 & -1 \\ 2 & 0 & -2 \\ 1 & 0 & -1 \end{bmatrix}$$

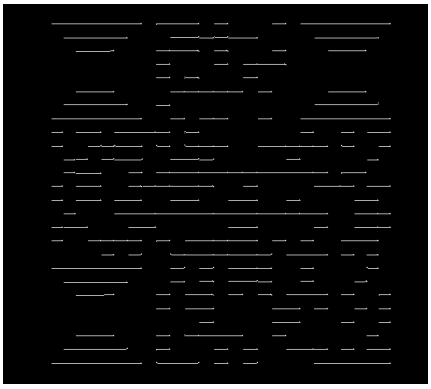
(a) Sobel Filter in horizontal direction

$$G_y = \begin{bmatrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix}$$

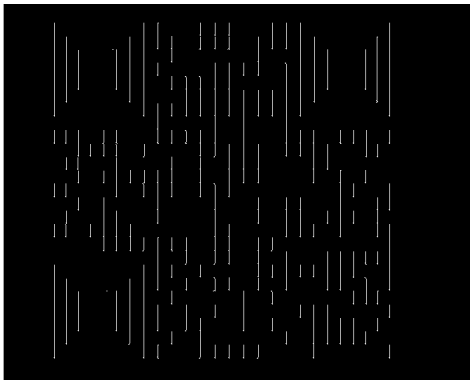
(b) Sobel Filter in vertical direction

Figure 4.13: Sobel Filters

The purpose of these kernels is to detect edges that run vertically and horizontally in relation to the pixel grid. To accomplish this, there are two separate kernels, each designed to respond maximally to one of the two perpendicular orientations. By applying these kernels separately to the input image, we can generate two distinct measurements of the gradient component in each orientation, as seen in Fig. 4.14.



(a) Image after applying horizontal sobel filter.

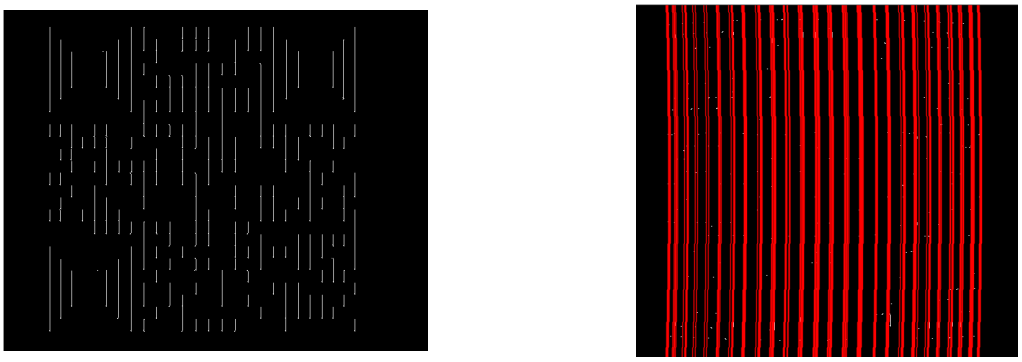


(b) Image after applying vertical sobel filter.

Figure 4.14: QR Code image after applying sobel filters

4.3.4 Edge hitting

To fit the conic curves on the QR Code, we use the boundaries BT, BB, BL, BR and the edge points obtained from the previous section. So, for example for the right boundary, we start by shifting it a small amount to the right, we'll call it BL0. Then, iteratively we will shift it pixel by pixel over the vertical edge points (see Fig.4.15a) and record every value for the boundary when it hits over a certain threshold of edge points, we'll denote them as BLn and the entire group of conic curves as BLall. In the end of this process, the entire group of conic curves is obtained and can be seen as the red lines in Fig.4.15b . We repeat the same sequence of action with all four boundaries and get BTall, BBall, BLall, BRall.



(a) Before

(b) After

Figure 4.15: The process of edge hitting

4.3.5 Clustering conic curves

In the previous section the positions of the conic curves were calculated, however the results contain conic curves very close to each other, in order to correct this, we apply a clustering algorithm based on DBSCAN.

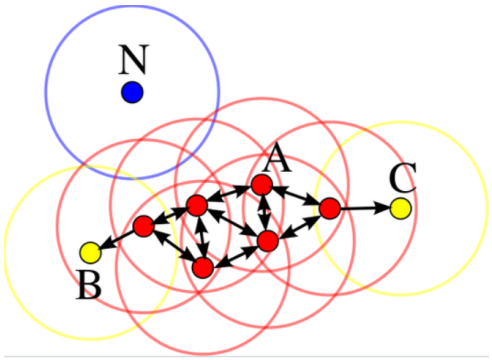
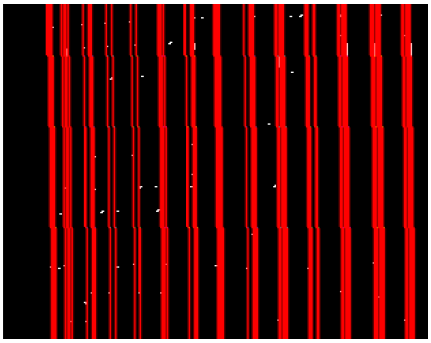
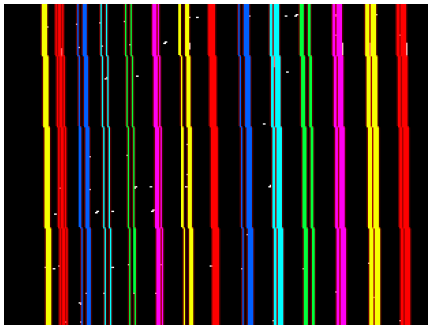


Figure 4.16: DBSCAN method.

DBSCAN (Density-Based Spatial Clustering of Applications with Noise) is a data clustering algorithm that was introduced in 1996 [23]. It identifies clusters of points based on their proximity to one another in space. Essentially, DBSCAN groups points that are densely packed together and have many nearby neighbours, without requiring any prior knowledge about the number or shape of the clusters in the data.



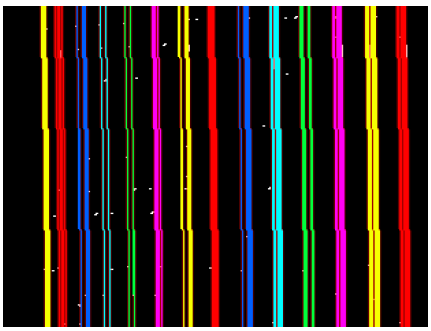
(a) Unclustered Grid



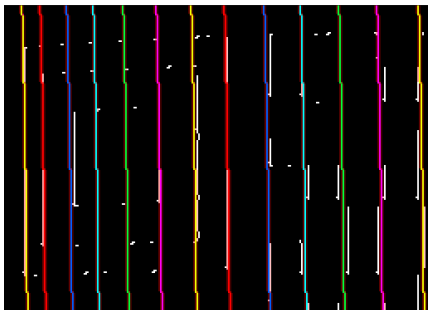
(b) Clustered Grid

Figure 4.17: The process of clustering, in (a) it's shown B_{Rall}, containing several conic curves, a lot of them very close to each other. By applying our clustering method results like (b) are obtained.

We repeat this same process for B_{Tall}, B_{Ball}, B_{Lall}, B_{Rall}, and filter each cluster so only the conic curve closest to the mean of each cluster is kept, this process is shown in Fig. 4.17.



(a) Clustered Grid - Before Filtering

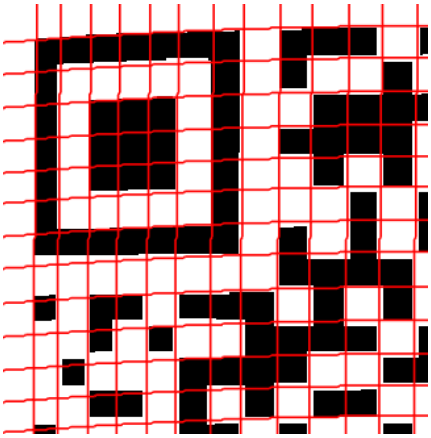


(b) Clustered Grid - After Filtering

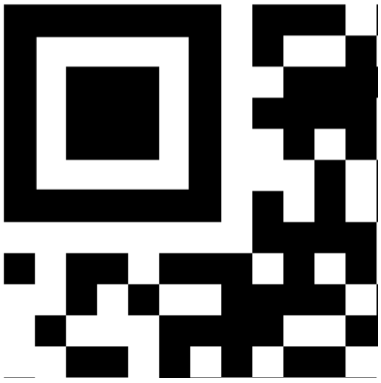
Figure 4.18: The process of filtering

4.3.6 Subdividing the QR Code

After clustering and filtering the fitted conic curves we have four different grids, each fitted to the QR Code. Different combinations between the horizontal (conic curves fitted using the top and bot boundaries), and vertical (conic curves fitted using the right and left boundaries), can be used to create a grid-like matrix that correctly divides each QR Code cell. Using these combinations, we can construct a new matrix with the corrected QR Code. We do this by subdividing the QR Code in four sections: top-left, top-right, bottom-left, bottom-right. For example, for the top-left section we use the conic curves fitted using the top and left boundaries as is shown in Fig 4.19.



(a) Conic Segmentation - Before



(b) Conic Segmentation - After

Figure 4.19: The process of conic segmentation

By the end of this process we end up with four sections (top-left, top-right, bottom-left, bottom-right) which are each respectively 1/4 of the original code. So in order to get the original undistorted code these four sections are stitched together, creating our final result, a completely undistorted QR Code as seen in Fig. 4.20.

Figure 4.20: Final QR.

In this chapter we used a QR Code to explain the methodology behind our method, however, it is important to note that this same methodology can be applied to any other 2-D MRC that consists of black and white squares since it doesn't rely on the Positional Detection Patterns to correctly locate and decode the code.

5 Experiments

5.1 Experiment Overview

The quality of the dataset we created to evaluate the method was crucial, as it could greatly affect the results and the conclusions drawn from them. To create a strong dataset, it was necessary to ensure that it accurately reflected the problem being solved and was also large enough to provide a detailed evaluation of the method's performance. When capturing images of codes affixed to cylindrical surfaces, the QR Code will obviously suffer from cylindrical distortion, however this distortion is amplified when combined with the tilting and/or sloping caused by the angle of the camera in relation to the code. Tilting distortion refers to when the code is tilted in relation to the camera, while sloping distortion occurs when the code or camera is inclined. To test our method to cylindrical distortions of this nature and assess its effectiveness, we devised datasets that simulate cylindrical distortion in combination with several ranges of sloping and tilting of the camera.

5.2 Cylindrical Distortion Ratio

A QR Code that is posted on a cylinder with a larger radius will experience less distortion than a QR Code posted on a cylinder with a smaller radius. Consequently, to quantify this distortion, a ratio can be created between the width/height of the QR Code and the radius of the cylinder it is posted on. We derived the equation 5.1 to express the ratio between of the two variables, code width and the radius of the cylinder it's posted on.

$$Ratio = \frac{CodeWidth}{CylinderRadius} \tag{5.1}$$

Both the synthetic and Real image images dataset uses this ratio to properly test the accuracy of our method. Examples are presented in Fig. 5.1 : (a) demonstrates a small ratio of 1, indicating minimal cylindrical distortion, while (i) illustrates a larger ratio of 3, indicating

significant distortion.

(a) Ratio of 1 (b) Ratio of 1,25 (c) Ratio of 1,5

(d) Ratio of 1,75 (e) Ratio of 2 (f) Ratio of 2,25

(g) Ratio of 2,5 (h) Ratio of 2,75 (i) Ratio of 3

Figure 5.1: Example images of cylindrical distortion ratio with an increasing ratio from 1 to 3.

5.3 Dataset Overview

To ensure a comprehensive evaluation of our method, we created two distinct datasets - one comprising synthetic images, and the other featuring Real image images. In total, our datasets

contain 603 images, providing ample coverage of a diverse set of scenarios.

Both datasets incorporate cylindrical distortion, and a wide range of tilting and sloping to create a wider range of scenarios and difficult cases. This approach can more effectively simulate real-world scenarios and we can better test the robustness of our method.

Our use of synthetic and Real image images also allows for a more nuanced assessment of the method's effectiveness, as it can perform differently under varying conditions. Further elaboration on the composition and conditions utilized in the creation of these datasets can be found in the subsequent subsections.

5.3.1 Synthetic Dataset

A synthetic dataset is a data set that is generated artificially, rather than being sourced from real-world data. Because synthetic datasets are generated artificially, it is easy to replicate the specific types of distortion that we want to test our method on. To create the dataset, we used a software called DesignSpark Mechanical. DesignSpark Mechanical is a 3D modelling software used for mechanical design and engineering. It allows to create 3D models of designs and prototypes, which can be exported in various file formats.

However, the true strength of this software lies in its ability to enable us to design and export objects in perspective view, thereby allowing us to simulate the cylindrical distortion we intend to evaluate. This dataset contains:

- 45 images of a QR Code with a cylindrical distortion ratio of 2 in a wide variety of tilting and sloping angles (Table 5.1).
- 9 images of a QR Code with a cylindrical distortion ratio of 2 with a varying degree of rotation (Table 5.2).
- 9 images of a QR Code with a varying amount of cylindrical distortion ratio (Table 6.9).

Syntethic Dataset Overview Tilt and Slope Combinations					
	Slope				
$Tilt_{(clockwise)}$	0°	5°	10°	15°	20°
-20°	✓	✓	✓	✓	✓
-15°	✓	✓	✓	✓	✓
-10°	✓	✓	✓	✓	✓
-5°	✓	✓	✓	✓	✓
0°	✓	✓	✓	✓	✓
5°	✓	✓	✓	✓	✓
10°	✓	✓	✓	✓	✓
15°	✓	✓	✓	✓	✓
20°	✓	✓	✓	✓	✓

Table 5.1: Syntethic Dataset Overview Tilt and Slope Combinations

Syntethic Dataset Overview - Rotation									
Rotation	-60°	-45°	-30°	-15°	0°	15°	30°	45°	60°
(clockwise)	✓	✓	✓	✓	✓	✓	✓	✓	✓

Table 5.2: Synthetic Dataset Overview - Rotation

Syntethic Dataset Overview - Cylindrical Distortion Ratio									
Ratio	1	1,25	1,5	1,75	2	2,25	2,5	2,75	3
	✓	✓	✓	✓	✓	✓	✓	✓	✓

Table 5.3: Syntethic Dataset Overview - Cylindrical Distortion Ratio

In total the dataset comprises a total of 72 images that feature various combinations of cylindrical distortion rotation ratios and different types of tilting, sloping, and rotation. Some of these images are shown in Fig.5.2.

(a) Only cylindrical distortion.

(b) Cylindrical distortion with only tilting.

(c) Cylindrical distortion with tilting and sloping

Figure 5.2: Example images of synthetic dataset.

5.3.2 Real image Dataset

Although synthetic datasets are very useful for testing our algorithm, a Real image dataset is also of extreme importance for evaluating the effectiveness in real-world scenarios. Real images contain a diverse range of distortions that are not present in synthetic datasets, such as lighting variations, occlusions, and noise.

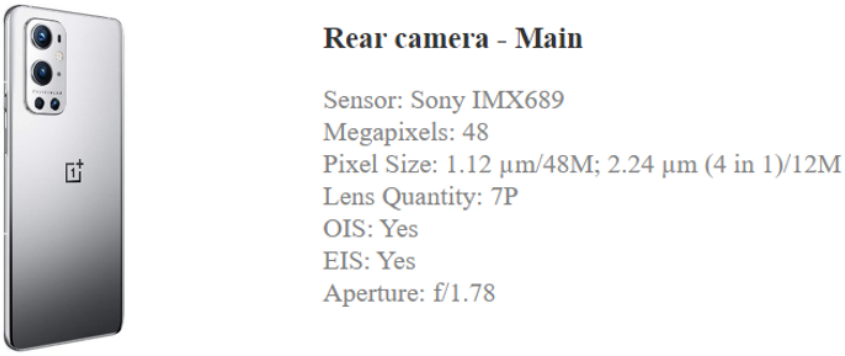


Figure 5.3: Camera specifications of One Plus Pro 8, taken from [3].

To create this database a rigorous process was determined and followed carefully to ensure the quality of the final dataset. To capture the images, we used a smartphone One Plus Pro 8, whose camera specifications can be seen in Fig.5.3 and a phone support which can be seen in Fig.5.4.

(a) Phone Support used.

(b) Phone Support in use in the lab.

Figure 5.4: Phone Support used to create Real Images dataset

The dataset was all captured in the ISR Computer Vision laboratory under the same conditions of light, exposure and background to ensure reliable data could be extracted from it. This dataset contains:

- 450 images of a QR Code with a cylindrical distortion ratio of 2 in a wide variety of tilting and sloping angles (Table 5.4):
  - 225 images using a QR Code width of 14mm.
  - 225 images using a QR Code width of 21mm.
- 90 images of a QR Code with a varying amount of cylindrical distortion ratio (Table 5.5).

Real image Image Dataset Overview - Tilt and slope combinations					
	Slope				
$Tilt_{(clockwise)}$	0°	5°	10°	15°	20°
-20°	5	5	5	5	5
-15°	5	5	5	5	5
-10°	5	5	5	5	5
-5°	5	5	5	5	5
0°	5	5	5	5	5
5°	5	5	5	5	5
10°	5	5	5	5	5
15°	5	5	5	5	5
20°	5	5	5	5	5

Table 5.4: Real image Image Dataset Overview - Tilt and slope combinations

Real image Image Dataset Overview - Cylindrical Distortion Ratio									
Ratio	1	1,25	1,5	1,75	2	2,25	2,5	2,75	3
	10	10	10	10	10	10	10	10	10

Table 5.5: Real image Image Dataset Overview - Cylindrical Distortion Ratio

(a) Only cylindrical distortion.

(b) Cylindrical distortion with only tilting.

(c) Cylindrical distortion with tilting and sloping

Figure 5.5: Example images of real images dataset.

5.4 Different Readers used

To properly evaluate our method we used two other QR Code readers as benchmarks. Since these readers only read QR Codes, these tests were only performed with QR Codes and not with the Graphic Codes (UniQode) in which we used another methodology to validate and evaluate our results. We used two readers, a basic, and an advanced one:

- readbarcode() from Matlab
- In-built Apple reader

Readbarcode() from Matlab is a very basic QR Code reader and does not perform any correcting to the QR Codes and only reads them if they are aligned in the vertical or horizontal axis. The in-build apple QR Code Reader however, is very capable and has proven to be successful in decoding QR Codes which have been severely deformed, this capacity is not publicized by Apple in any way and no documentation is available to the public since their methods are all proprietary. For this reason we believe it is important to compare our results with this benchmark in order to present solutions available to everyone that replicate this reader's success.

6 Results

6.1 Synthetic database

To start we tested the entirety of our synthetic database using the 3 methods mentioned before, the very simple readbarcode(), Apple's integrated method and ours.

6.1.1 Cylindrical Ratio

Within this section we test the impact of increasing the cylindrical ratio (defined in equation 5.1) in the decoding rate of the decoders used.

Syntethic Dataset Overview - Cylindrical Distortion Ratio									
Ratio	1	1,25	1,5	1,75	2	2,25	2,5	2,75	3
readbarcode()	✓	✓	✓	✓	✗	✗	✗	✗	✗
Proposed method	✓	✓	✓	✓	✓	✓	✓	✓	✗
Apple Method	✓	✓	✓	✓	✓	✓	✗	✗	✗

Table 6.1: Syntethic Dataset Overview - Cylindrical Distortion Ratio

From the results obtained we can see that the effects of cylindrical distortion start to decrease the decoding rate at around the value of 1,75. Apple's method shows some ability to reconstruct the distorted code, however our method is capable to handle more severe amounts of cylindrical distortion, being able to successfully decode a QR code with a 2,75 ratio of distortion.

6.1.2 Ratio + Tilting

After completing the experiments with variations of only the cylindrical ratio, we now want to investigate the effects of combining cylindrical distortion with some tilting, which is produced

when the camera has some angle in relation to the code. For this section we test a range of 40 degrees (-20°to 20°) in the horizontal axis and the code captured has a cylindrical distortion ratio of 2.

Syntethic Dataset - Ratio=2,0 Tilting= [-20°, 20 °] Sloping=0°			
Tilt (clockwise)	readbarcode()	Proposed Method	Apple's Method
-20°	✗	✗	✗
-15°	✗	✓	✗
-10°	✗	✓	✓
-5 °	✗	✓	✓
0 °	✓	✓	✓
5 °	✗	✓	✓
10 °	✗	✓	✓
15 °	✗	✓	✗
20 °	✗	✗	✗

Table 6.2: Syntethic Dataset - Ratio=2,0 Tilting= [-20°, 20 °] Sloping=0°

6.1.3 Ratio + Tilting + Sloping

To effectively represent real life scenarios where the camera capturing the code is often miss-aligned with the center of the code, in this section we test the capture of the code with the addition of tilting and sloping. We tested a range of tilting of 40°, as in the above section with the addition of a range of 20 degrees (0°to 20°) of sloping. The code captured has a cylindrical distortion ratio of 2.

Sloping of 5°

Syntethic Dataset - Ratio=2,0 Tilting= [-20°, 20 °] Sloping=5°			
Tilt(clockwise)	readbarcode()	Proposed Method	Apple Method
-20°	✗	✗	✗
-15°	✗	✓	✗
-10°	✗	✓	✗
-5 °	✗	✓	✓
0 °	✗	✓	✓
5 °	✗	✓	✓
10 °	✗	✓	✗
15 °	✗	✓	✗
20 °	✗	✗	✗

Table 6.3: Syntethic Dataset - Ratio=2,0 Tilting= [-20°, 20 °] Sloping=5°

Sloping of 10°

Syntethic Dataset - Ratio=2,0 Tilting= [-20°, 20 °] Sloping=10°			
Tilt(clockwise)	readbarcode()	Proposed Method	Apple Method
-20°	✗	✗	✗
-15°	✗	✗	✗
-10°	✗	✓	✗
-5 °	✗	✓	✓
0 °	✗	✓	✓
5 °	✗	✓	✗
10 °	✗	✓	✗
15 °	✗	✗	✗
20 °	✗	✗	✗

Table 6.4: Syntethic Dataset - Ratio=2,0 Tilting= [-20°, 20 °] Sloping=10°

Sloping of 15°

Syntethic Dataset - Ratio=2,0 Tilting= [-20°, 20 °] Sloping=15°			
Tilt(clockwise)	readbarcode()	Proposed Method	Apple Method
-20°	✗	✗	✗
-15°	✗	✗	✗
-10°	✗	✗	✗
-5 °	✗	✗	✗
0 °	✗	✓	✗
5 °	✗	✗	✗
10 °	✗	✗	✗
15 °	✗	✗	✗
20 °	✗	✗	✗

Table 6.5: Syntethic Dataset - Ratio=2,0 Tilting= [-20°, 20 °] Sloping=15°

We didn't include the table containing the results with a sloping of 20°since neither method achieved a single successful decoding in that scenario. With the results obtained it is obvious that the introduction of sloping to the codes highly affects the decoding rate, since the cylindrical distortion is highly amplified with the addition of sloping and tilting. It's, however, clear that our method produced significantly better results than the tested counterparts.

6.2 Real Images Dataset - QR Codes

6.2.1 Cylindrical distortion - Ratio

To start testing with our dataset comprising of real images captured by a smartphone camera, we first test the influence that cylindrical distortion ratio has in the decoding rate of the three methods.

Real Dataset Overview - Cylindrical Distortion Ratio									
Ratio	1	1,25	1,5	1,75	2	2,25	2,5	2,75	3
readbarcode()	100%	100%	80%	90%	20%	0%	0%	0%	0%
Proposed method	100%	100%	100%	100%	100%	100%	80%	20%	0%
Apple's Method	100%	100%	100%	100%	100%	40%	0%	0%	0%

Table 6.6: Real Dataset Overview - Cylindrical Distortion Ratio

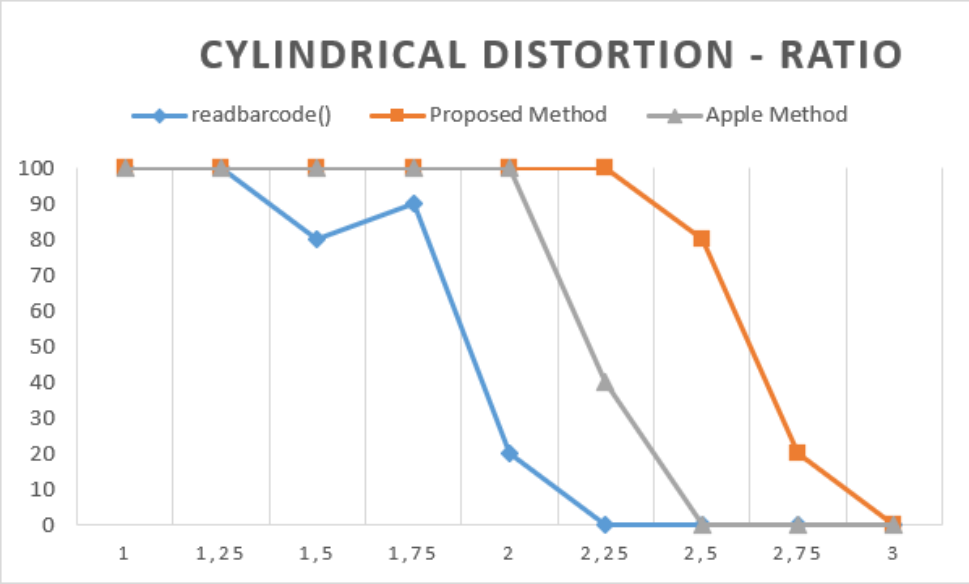


Figure 6.1: Real Dataset Overview - Cylindrical Distortion Ratio

6.2.2 Ratio + Tilting

Within this section, we conduct a comprehensive assessment of QR Codes featuring two different widths (14mm and 21mm). The QR Codes in this section all have been distorted with a cylindrical ratio of 2, this means that, for example a QR code with a width of 14mm has been posted on a cylinder with a 7mm radius as per equation 5.1. We aim to explore the impact of tilting on these QR Codes across a wide spectrum of angles, ranging from -20 to 20 degrees.

QR Code - 14mmx14mm

QR Code 14mm - Ratio=2,0 Tilting= [-20°, 20 °] Sloping=0°			
Tilt(clockwise)	readbarcode()	Proposed Method	Apple's Method
-20°	0%	80%	100%
-15°	0%	100%	100%
-10°	0%	100%	80%
-5 °	0%	100%	100%
0 °	100%	100%	80%
5 °	60%	100%	100%
10 °	0%	80%	100%
15 °	0%	100%	100%
20 °	0%	80%	100%

Table 6.7: QR Code 14mm - Ratio=2,0 Tilting= [-20°, 20 °] Sloping=0°

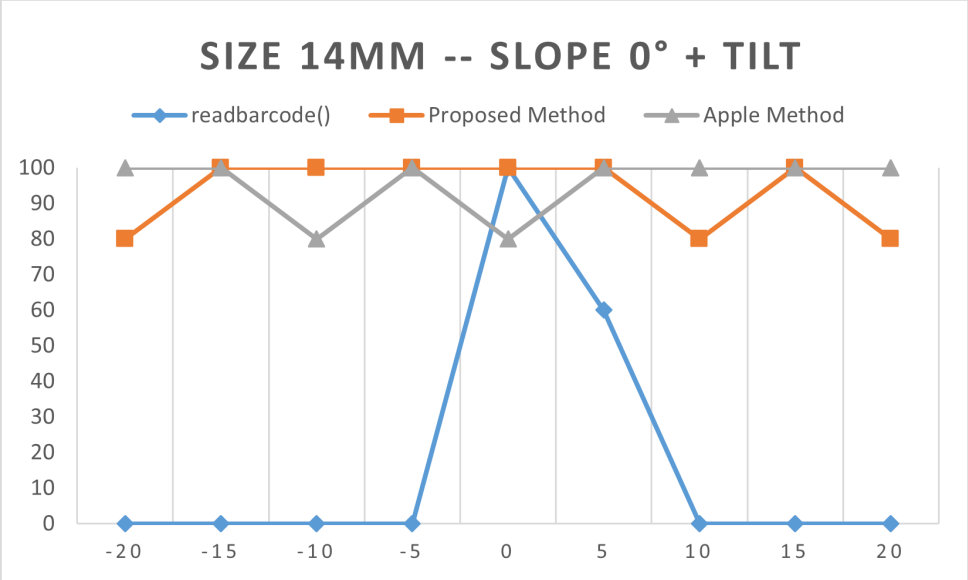


Figure 6.2: QR Code 14mm - Ratio=2,0 Tilting= [-20°, 20 °] Sloping=0°

QR Code - 21mmx21mm

QR Code 21mm - Ratio=2,0 Tilting= [-20°, 20 °] Sloping=0°			
Tilt(clockwise)	readbarcode()	Proposed Method	Apple's Method
-20°	0%	0%	20%
-15°	0%	40%	40%
-10°	0%	100%	60%
-5 °	0%	100%	80%
0 °	100%	100%	100%
5 °	0%	100%	80%
10 °	0%	100%	100%
15 °	0%	20%	40%
20 °	0%	0%	0%

Table 6.8: QR Code 21mm - Ratio=2,0 Tilting= [-20°, 20 °] Sloping=0°

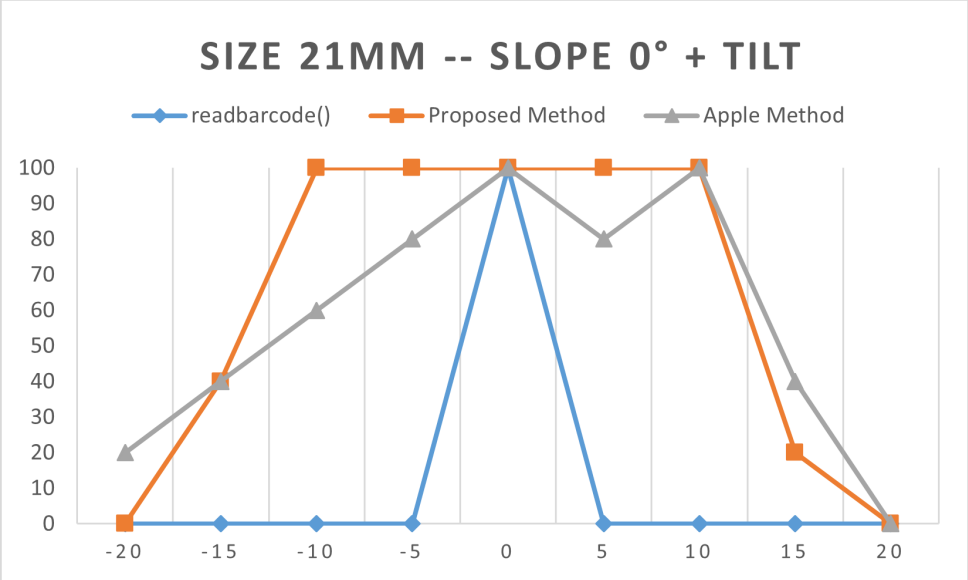


Figure 6.3: Graph: QR Code 21mm - Ratio=2,0 Tilting= [-20°, 20 °] Sloping=0°

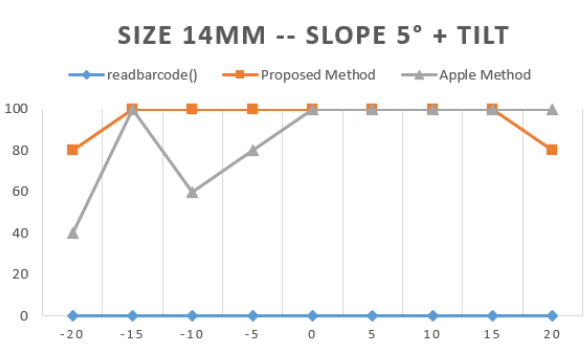
By the results presented above it is clear the effect that tilting as on the QR Code since both methods (ours and Apple's) have a drastic decrease in decoding rate by the ±10°tilting value. The cylindrical distortion on the code is amplified by the tilting which causes a lot of distortion on the edges of the code which sinks the decoding rate. Between the two methods in the smaller QR codes ($14mm * 14mm$) both performed very similar however in the larger QR Codes

$(21mm * 21mm)$ our method slightly outperforms Apple's one.

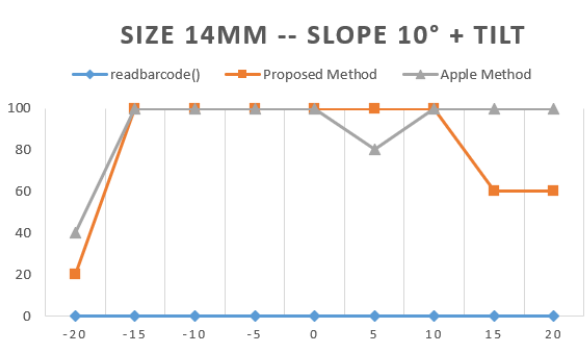
6.2.3 Ratio + Tilting + Sloping

Similar to the previous section, we now tested QR Codes with a width of 14mm and 21mm with a cylindrical ratio of 2 over a range of [-20°,20°] degrees of tilting and with sloping over a range of [5°,20°].

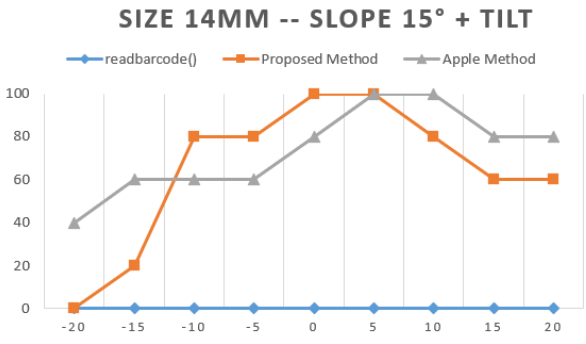
QR Code - 14mmx14mm



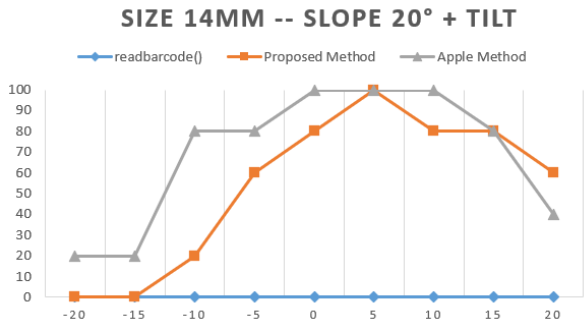
(a) QR Code - 14mmx14mm - Slope: 5°



(b) QR Code - 14mmx14mm - Slope: 10°



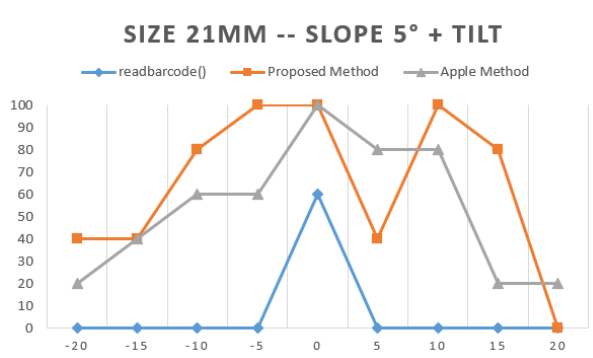
(c) QR Code - 14mmx14mm - Slope: 15°



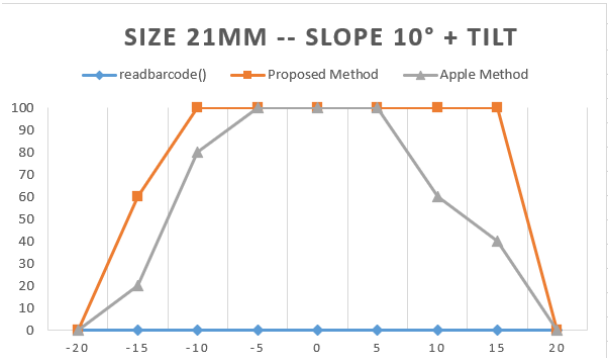
(d) QR Code - 14mmx14mm - Slope: 20°

Figure 6.4: QR Code - 14mmx14mm - Results obtained for different slope combinations.

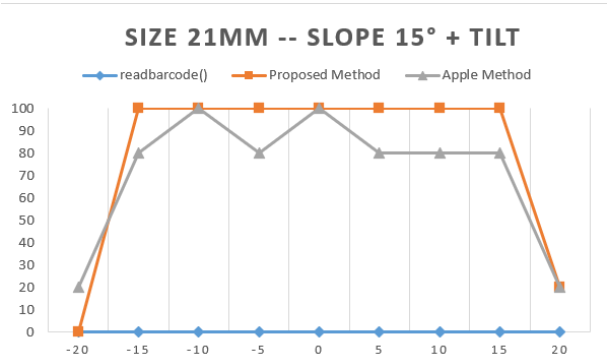
QR Code - 21mmx21mm



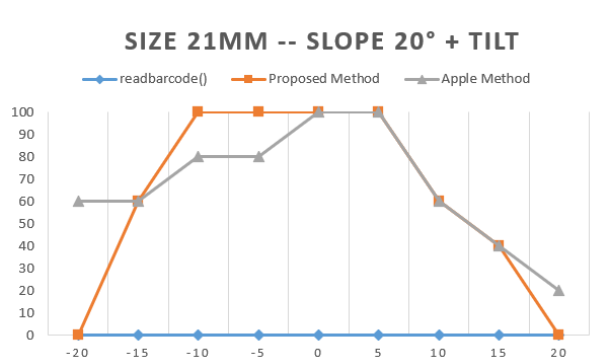
(a) QR Code - 21mmx21mm - Slope: 5°



(b) QR Code - 21mmx21mm - Slope: 10°



(c) QR Code - 21mmx21mm - Slope: 15°



(d) QR Code - 21mmx21mm - Slope: 20°

Figure 6.5: QR Code - 21mmx21mm - Results obtained for different slope combinations.

It is clear that with the introduction of sloping on these codes the decoding rate would decrease again. The junction of sloping and tilting on the cylindrical distorted QR Code creates significant distortion on the code and makes it very hard to decode them. We had mixed results between the two methods since Apple's method slightly outperformed ours in the smaller codes and ours outperformed Apple's method in the larger codes.

6.2.4 Overall decoding rate

QR Code - 14mmx14mm

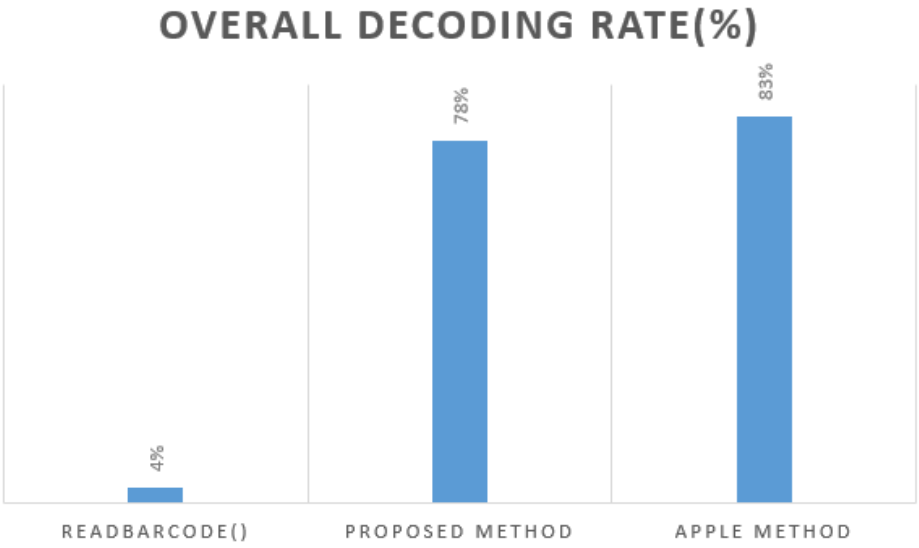


Figure 6.6: QR Code 14mm - Ratio=2 Overall Decoding rate

QR Code - 21mmx21mm

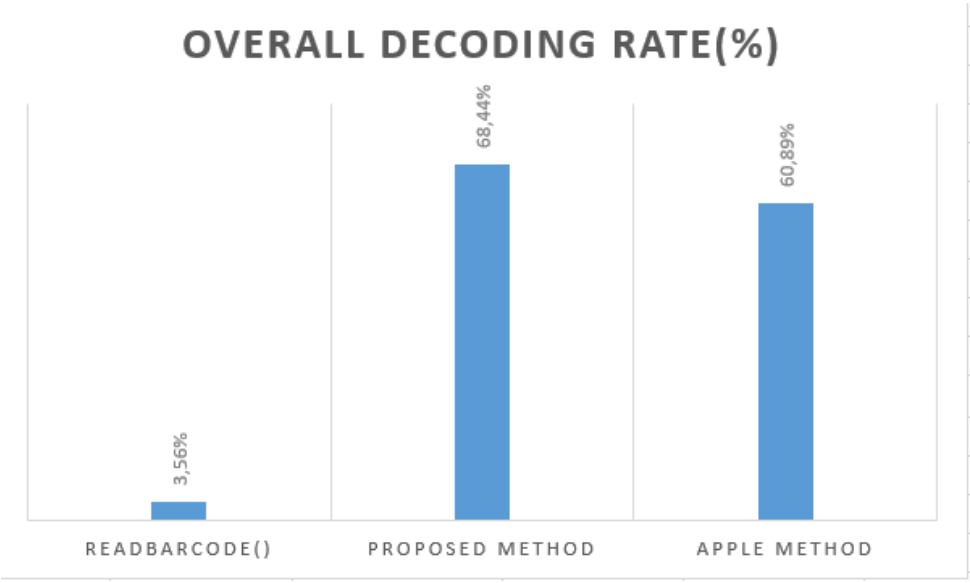


Figure 6.7: QR Code 21mm - Ratio=2 Overall Decoding rate

We note the interesting results obtained, it's unclear why both methods struggle with the larger codes, but we hypothesize that since the code is larger the tilting and sloping cause bigger distortions on the code. It's also clear by the results presented above that our method is slightly

better than Apple's but only by a small fraction, we believe this is to be a good result since Apple's method is proprietary, probably with several other pre and post processing techniques that enhance the method's accuracy and reliance. One of the initial objectives of this thesis was to build an open-source method which achieves state of the art accuracy, to decode 2D barcodes, like the QR Code and the Graphic Code (UniQode). The Apple's method, due to it's very high performance in real life scenarios, was a good goal to achieve.

6.3 Real Image Dataset - Tax Stamp Code Code

Since the Graphic is still relatively new, and the majority of Graphic Code decoders are not in the public domain, we created a simple algorithm to calculate the percentage of pixels our method would correctly find. Again, as this code is not widely used and is still in development we also couldn't compare the results with other decoders/methods but we believe some valuable information can be extracted from the following results.

6.3.1 Ratio + Tilting + Sloping

In this section we analyze the effects of only tilting (when the slope is equal to 0°) and a combination of tilting and sloping. The ranges used for these are the same as used in all previous sections, this is a range of [-20°,20°] degrees of tilting and sloping over a range of [0°,20°].

Tax Stamp Code Dataset Results – Cylindrical Distortion = 2					
	Slope				
Tilt	0°	5°	10°	15°	20°
-20°	94,2	93,05%	95,8%	81,66%	73,25%
-15°	90,8%	100%	99,88%	93,11%	76,96%
-10°	100%	100%	99,82%	99,05%	90,36%
-5°	100%	100%	100%	95,95%	95,77%
0°	100%	100%	99,91%	95,95%	91,6%
5°	100%	100%	99,88%	95,86%	89,32%
10°	100%	99,85%	99,56%	98,88%	81,8%
15°	99,97%	99,76%	77,87%	84,73%	75,3%
20°	99,91%	84,05%	81,6%	75,36%	74,67%

Table 6.9: Tax Stamp Code Dataset Results – Cylindrical Distortion = 2

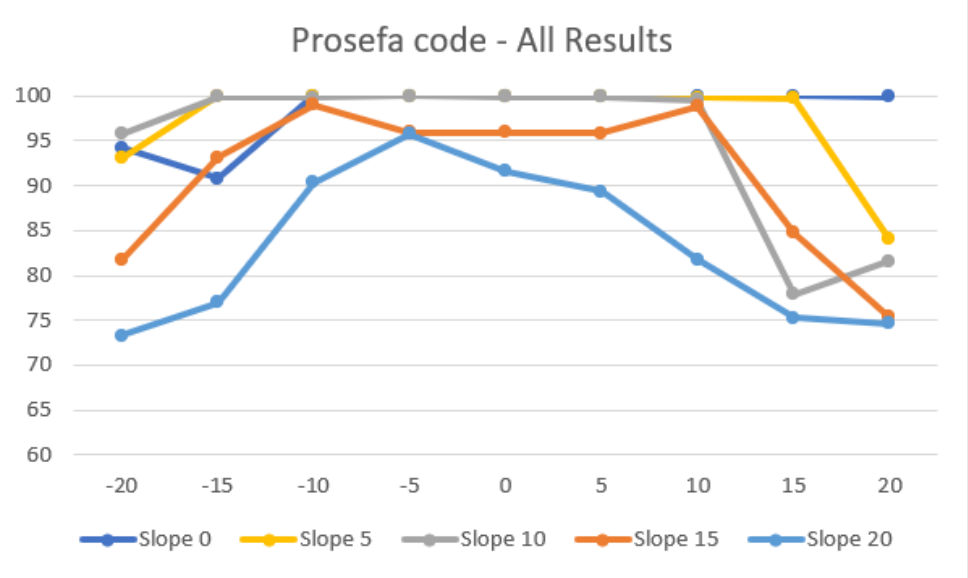


Figure 6.8: Tax Stamp Code Dataset all results

It is important to note that the reason we obtain such an high accuracy rate in the Graphic Code compared to the QR Code, is because the pixels that the method starts to not correctly find are usually the ones in the borders of the code. Since QR Code decoders need to use the Positional Detection Patterns to correctly decode them, the accuracy drops very fast even though most pixels are recovered.

7 Conclusion

Machine-readable codes are increasingly utilized in various industries and applications, including retail, logistics, security, healthcare, and transportation and therefore, the accurate decoding of these codes holds immense importance and possesses the potential to greatly improve their utilization in these sectors. In this thesis we have presented a new method for correcting cylindrical distortion in QR Codes and other types of machine-readable codes. Some of the existing methods for distortion correction have shown limitations when applied to cylindrical surfaces, often resulting in inaccurate decoding and compromised data integrity. Other methods have shown some capacity in correcting this type of distortion however their methodology is closed source.

The proposed method leverages computer vision techniques to accurately identify and rectify the cylindrical distortion present in QR Codes. The proposed method also has the ability to work with other machine readable codes like the Graphic Code (UniQode) which is a limitation most of the already existing methods don't tackle.

In our work we did extensive experimental tests and our results demonstrate the effectiveness of the proposed method in correcting cylindrical distortion, achieving comparable and even slightly improved decoding accuracy compared to existing closed-source techniques and greatly surpassing the accuracy achieved by some commonly used decoders.

7.1 Future Work

We believe future work can still be done using different computer vision techniques in order to achieve even better results, however it is the use of machine learning approaches that can potentially greatly enhance the performance and robustness of the distortion correction process. Even though the use of machine learning for this purpose was not tested in this thesis due to the need of a lightweight method that could be used on cellphones, with the recent advances in deep learning techniques, these methods have been becoming more lightweight and so their reliability for this purpose is increasing.

Bibliography

[1] E. Basker and T. Simcoe, "Upstream, downstream: Diffusion and impacts of the universal product code," *Journal of Political Economy*, vol. 129, pp. 1252–1286, 4 2021.

[2] K.-T. Lay, L.-J. Wang, P.-L. Han, and Y.-S. Lin, "Rectification of images of qr codes posted on cylinders by conic segmentation," pp. 389–393, IEEE, 10 2015.

[3] "One plus pro 8 phone specifications."

[4] "Information technology - automatic identification and data capture techniques - qr code bar code symbology specification," *ISO/IEC 18004:2015*.

[5] "Information technology - automatic identification and data capture techniques - data matrix bar code symbology specification," *ISO/IEC 16022:2006*.

[6] "Information technology - automatic identification and data capture techniques - aztec code bar code symbology specification," *ISO/IEC 24778:2008*.

[7] L. Cruz, B. Patrao, and N. Goncalves, "Graphic code: A new machine readable approach," pp. 169–172, IEEE, 12 2018.

[8] "History of the qr code."

[9] J.-A. Lin and C.-S. Fuh, "2d barcode image decoding," *Mathematical Problems in Engineering*, vol. 2013, pp. 1–10, 2013.

[10] H. Tribak and Y. Zaz, "Qr code recognition based on principal components analysis method," *International Journal of Advanced Computer Science and Applications*, vol. 8, no. 4, 2017.

[11] S. Li, J. Shang, Z. Duan, and J. Huang, "Fast detection method of quick response code based on run-length coding," *IET Image Processing*, vol. 12, pp. 546–551, 4 2018.

[12] L. Belussi and N. Hirata, “Fast qr code detection in arbitrarily acquired images,” pp. 281–288, IEEE, 8 2011.

[13] D. K. Hansen, K. Nasrollahi, C. B. Rasmusen, and T. B. Moeslund, “Real-time barcode detection and classification using deep learning,” pp. 321–327, SCITEPRESS - Science and Technology Publications, 2017.

[14] T.-H. Chou, C.-S. Ho, and Y.-F. Kuo, “Qr code detection using convolutional neural networks,” pp. 1–5, IEEE, 5 2015.

[15] A. Zharkov and I. Zagaynov, “Universal barcode detector via semantic segmentation,” pp. 837–843, IEEE, 9 2019.

[16] S. Kong, “Qr code image correction based on corner detection and convex hull algorithm,” *Journal of Multimedia*, vol. 8, 11 2013.

[17] P. Gaur and S. Tiwari, “Recognition of 2 d barcode images using edge detection and morphological operation,” 2014.

[18] M. Li, P. Cao, L. Feng, L. Yu, J. Chen, and J. Wang, “The research of qr code image correction based on image gray feature,” pp. 1–5, IEEE, 6 2017.

[19] D. Bradley and G. Roth, “Adaptive thresholding using the integral image,” *Journal of Graphics Tools*, vol. 12, pp. 13–21, 1 2007.

[20] R. Gonzalez and R. Woods, “Erosion and dilation,” 1992.

[21] S. P., “Morphological reconstruction,” 1999.

[22] M. Sonka, V. Hlavac, and R. Boyle, *Image Processing, Analysis and Machine Vision*. Springer US, 1993.

[23] A. Ram, S. Jalal, A. S. Jalal, and M. Kumar, “A density based algorithm for discovering density varied clusters in large spatial databases,” *International Journal of Computer Applications*, vol. 3, pp. 1–4, 6 2010.