Journal of Information Hiding and Multimedia Signal Processing ©2021 ISSN 2073-4212
Ubiquitous International Volume **12**, Number **4**, December **2021**

Multi Barcode Scanning and Decoding Technology Based on AR Smart Glasses

Charles Chen
School of Computer Science and Engineering
Minnan Normal University
No. 36 Xianqianzhi Street, Xiangcheng District, Zhangzhou City, Fujian Province, China. 363000
charles.chen@mnnu.edu.cn; 2440216302@qq.com

Sheng-Lung Peng
Department of Creation Technology and Product Design
National Taipei University of Business
No. 321, Sec. 1, Jinan Rd., Zhongzheng District, Taipei, 100, Taiwan
slpeng@ntub.edu.tw

Received September 2021; revised November 2021

Abstract. *This research mainly develops the technology of multi barcode location, extraction and decoding on Android AR smart glasses, and realizes it with Python, OpenCV and Zbar. This technology can be applied to manufacturing industry, combining shop floor control system (SFCS) and execution manufacturing system (MES), to improve the production process and reduce the production cycle. In addition, due to the spread of COVID-19, in order to avoid infection of COVID-19, the business mode of unmanned shopping malls is getting highly valued. Our research result provides a feasible solution to unmanned shopping malls.*
Keywords: Barcode; Python, OpenCV; Zbar; Augmented Reality.

1. **Introduction.** Our research focuses on the application of AR smart glasses and the development of multi barcodes locating and decoding technology. At present, many materials are needed in the production process. Operators often need to hold the materials in one hand and scan the barcode of the materials with the code scanner in the other hand, so as to input them into the shop floor control system (SFCS). In addition, in the production of back-end packaging operations, we must first scan the barcode on the carton, and then scan the barcode on the product box loaded in the carton, which often requires many times of code scanning action, which will reduce the production efficiency. How to free hands and let operators focus on the production line is the purpose of this research project. This research combines AR smart glasses and multi barcodes locating and decoding technology to develop multi barcode scanning technology which is coded by Python. Our result can be applied to the manufacturing industry, especially in the shop floor control system or execution manufacturing system (MES), to improve the production process and reduce the production cycle of products. Through the camera on the AR smart glasses, the video of the object is transmitted to the wearable microcomputer connected with the smart glasses. The video visual analysis program on the microcomputer will analyze the transmitted video information, first locate multiple bar codes, and then analyze the information contained in each bar code. Through the wireless device on the wearable microcomputer, the analyzed information is transmitted to the back-end

"shop flow control system" or "manufacturing execution system". When the back-end system detects the error or has any indication, it can also send the information to the wearable microcomputer and present it on the display of glasses, so that the workshop staff can take corrective actions in time. Figure 1 is the proposed system architecture of this research. Regarding the program of scanning multiple barcodes, the Python3.7 programming language with OpenCV module and Zbar module is used under the platform of PC Windows 7 system. Then, the developed Python program is moved to Android based AR smart glasses via Qpython. Our experiment results show EAN8, EAN13, Code39, Code128, I25, CODEBAR, UPC-A, UPC-E and QR Code barcodes can be scanned and decoded by our Python program. This is the contribution of this paper. The structure of this paper is scheduled as follows. Section 2 describes the related works. Section 3 is the research method. Section 4 is the Result Analysis and Conclusion.

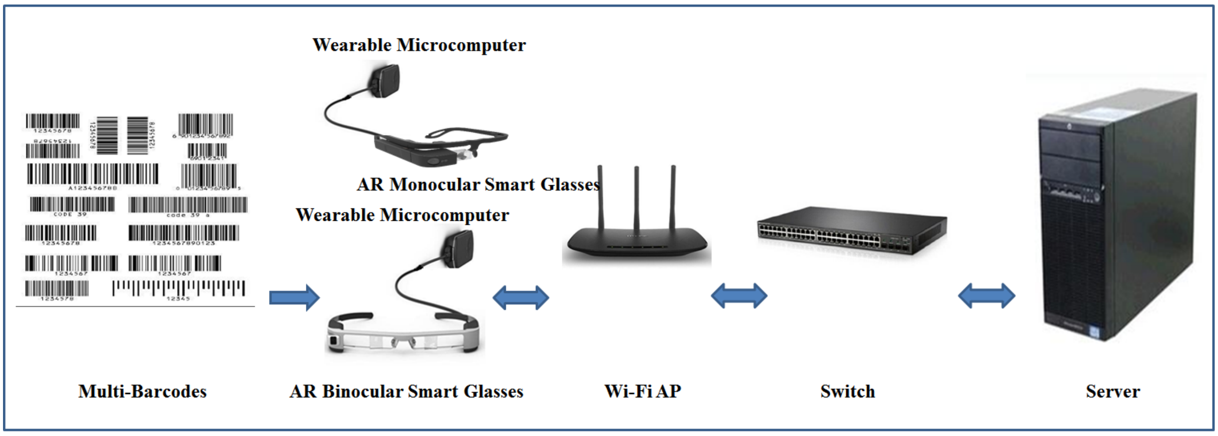

FIGURE 1. System architecture

2. Related Work.

2.1. AR/VR.

The origin of AR/VR technology can be traced back to the Sensorama simulator invented by Morton Heilig in the 1950s and 1960s. He is a philosopher, filmmaker and inventor. He used his film experience to design a machine called Sensorama simulator. Sensorrama simulator can use images, sound, fragrance and vibration to let users feel the scene of riding a motorcycle on the streets of Brooklyn, New York. This invention was very advanced at that time. Taking this opportunity, AR also started its development history. Among the most widely accepted definitions, researcher Ron Azuma said that augmented reality (AR) has three key requirements [1]:
(1) It combines real and virtual content
(2) Real time interaction
(3) Register in 3D
These three characteristics also define the technical requirements of AR system, that is, it must have a display that can combine real and virtual images, a computer system that can generate interactive graphics and make real-time response to user input, and a computer system that can determine the user's viewpoint position and make the virtual image display in the real world. Generally speaking, the key of AR technology lies in the perception and understanding of the device to the surrounding environment: the most basic is to determine the spatial location of the device itself; the more advanced is to reconstruct the environment in real time (Simultaneous Localization and Map Building, SLAM); the more advanced is related to recognition, cognition and interaction [2].

From the perspective of user experience, AR development mainly includes the following four aspects:
(1) From content display to content interaction. Now more and more AR applications begin to transition from display experience to interactive experience. AR is no longer a simple model presentation, but gradually evolving to an interactive system. This trend, especially in AR games, is well reflected.
(2) From entertainment application to practical function. With the deepening of users' awareness of AR, AR applications with entertainment display can not meet the needs of users. Taking mobile AR products as an example, most of the applications on the market are model display, scene interaction, games, etc. But now, AR real measurement, AR furniture placement, AR real translation, AR navigation, AR remote guidance machine maintenance, AR telemedicine consultation and other functional products are gradually emerging. This kind of functional AR application will become a trend, will become more and more, solve the main point of every aspect of users' daily life.
(3) From single person experience to multi person interaction. AR content experience with a single capability is only an intermediate state under the current technical background. The ideal AR is to build a copy of the real physical world and realize digital AR cloud content sharing, continuous experience and multi person interaction. At present, there are many technical reserves, which can realize multi person interactive experience, including the collaborative technology of ARKit [3] and the cloud anchors capability of ARCore [4].
(4) From camera enhancement to environment enhancement. With the development of SLAM, image recognition and tracking technology, AR based on real environment perception will further realize the matching between virtual information and real physical world, realize environment understanding, realize virtual and real occlusion, and present the corresponding AR information in the appropriate location.

2.2. **Barcode.** Barcodes are something that everybody sees every day; so common as to be taken for granted and normally unnoticed. Readable, no one reads them. They are used to allow machines to identify a wide variety of non-electronic, real life objects. Barcode is one of the earliest types of what is now called "Automatic Identification and Data Capture" (AIDC), meaning "data was transmitted into whatever system by something other than typing or hand-writing" [5].
Barcode systems belong to the "optical" category of AIDC. It is very old in usage as these technologies go, having first been patented in 1949. The initial uses were in the early 1950s and diversity of use is ever increasing as people find new ways to make this versatile old technology work. Generally speaking, barcode is composed of bars and spaces with different width and reflectivity, which are compiled according to certain coding rules to express a group of digital or alphabetic symbol signals [6].
The process of barcode-based systems generally includes two stages: one is barcode localization, and another is barcode decoding [7]. There are a lot of proposed barcode location algorithms [8-12]. These methods extract handcrafted barcode features via corner detection, edge detection, Hough transform, morphological operations.
Regarding the barcode detection methods, some have used machine learning and deep learning in recent years. Zamberletti et al. [13] used a multilayer perceptron (MLP) network to identify 1D barcode in Hough Transform space. The deep neural network was first used to detect barcode by Gr'osz et al. [14]. In their work, Deep Rectifier Neural Networks (DRN) has been utilized to classify whether the pixel is QR code or background. In term of universal object detection, many CNN-based methods have been proposed [15, 16, 17, 18]. In addition, arbitrary-oriented object detection methods have

been studied in scene text detection [19, 20]. These methods solve the issue that the text may appear in different scenes with different angles. Zhang et al. [7] propose a region proposal network (RPN) based barcode detection network to finely detect and classify multi-class barcode in complex environments. The proposed method includes a region proposal network, a multi-scale spatial pyramid pooling (MSPP) layer, a quadrilateral bounding box regression layer, and multi-class barcode classification layer.

2.2.1. *Classification of Barcode.* According to the dimension of pattern, barcode can be simply classified into one-dimension barcode and two-dimension barcode [7]. The popular use of one-dimension barcode includes EAN barcode, UPC barcode, Code25, Code39, Code128, Codabar barcode, etc. UPC/EAN barcode often use in the commerce logistic system. Code25 is usually used in the warehouse and logistics management system. Code39/Code128 are widely used in enterprise internal management, production process and logistics control system. Codabar barcode is mainly used for automatic identification in medical and health, library and information, materials and other fields [7, 20].
According to structure of barcode, two-dimension barcode has 2D Stacked Code and 2D Matrix Code. 2D Stacked Code is a code system based on one-dimensional bar code, which can be pushed into two or more lines as needed. Its coding design, verification principle and reading equipment are compatible with one-dimensional barcode. Representative two-dimensional barcodes are Code16k, Code49, PDF417, etc. [8]. 2D Matrix Code is in the matrix corresponding element position, with the point (square point, dot or other shape) appear to represent binary "1", the point does not appear to represent binary "0". The arrangement and combination of points determine the meaning of matrix type two-dimensional code. 2D Matrix bar code is a new type of automatic recognition and processing code system based on computer image processing technology and combination coding principle. Representative matrix two-dimensional bar codes include Code One, Maxicode, QR code, Data Matrix, etc. [21].

2.2.2. *Merit of Barcode Technology.* Bar code is the most economical and practical automatic identification technology so far. Bar code technology has the following advantages [6, 11]:
(1) Fast input speed: compared with keyboard input, bar code input speed is 100 times of keyboard input, and can realize "real-time data input".
(2) Long-proven technology with many specialized and more general applications on the market affording a wide range of programmatic, format and vendor choices.
(3) large degree of freedom. The degree of freedom of the relative position between the identification device and the barcode label is much greater than that of OCR.
(4) Unique identification for each record/object averts confusion between similar documents and meets many regulatory requirements for that "uniqueness" with no further effort.
(5) High reliability: the error rate of keyboard input data is one in three hundred, the error rate of optical character recognition technology is one in ten thousand, and the error rate of bar code technology is less than one in one million.
(6) One-to-one correspondence between records and metadata linked by the unique identifier.
(7) Large amount of information collection: the traditional one-dimensional bar code can collect dozens of characters at a time, and the two-dimensional bar code can carry thousands of characters of information, and has a certain ability of automatic error correction.
(8) Flexible and practical: bar code identification can be used alone as an identification means, can also form a system with relevant identification equipment to realize automatic

identification, and can also be connected with other control equipment to realize automatic management.
(9) Machine-readable format reduces human error based on misreading record identification and/or content (e.g., title) potentially improving both storage and retrieval accuracy.
(10)Improved information lifecycle and inventory management achieved.
(11)The equipment is simple. The bar code symbol recognition equipment has the advantages of simple structure, easy operation and no special training.
(12)Space management improvements and associated cost savings proven.

2.3. **OpenCV.** The full name of OpenCV is open source computer vision library. OpenCV was established by Intel in 1999 and is currently supported and maintained by a non-profit foundation (OpenCV.org) [22]. OpenCV is a cross platform computer vision library based on BSD license (open source). The original version was published in IEEE Computer Vision and pattern recognition in 2000, and can run on Linux, windows, Android and Mac OS operating systems. It is lightweight and efficient, which is composed of a series of C functions and a small number of C++ classes. It also provides the interfaces of python, ruby, MATLAB and other languages, and realizes many general algorithms in image processing and computer vision [23]. OpenCV includes many functions about computer image conversion, image processing and other mathematical operation processing. The main modules of OpenCV3.20 are shown in Table 1 [23].

2.4. **OpenCV-Python.** OpenCV-Python is the API interface of OpenCV for python. It has the functions of OpenCV C++ API and the features of Python language. Although Python language runs slower than C++, it is simple, short and fast to learn. At the same time, Python is also very convenient to call the components developed by C++. Such high-performance functions can be realized by C++. In this way, we can not only run code as fast as C++, but also easily use Python language to develop software functions, which is why we need to implement the OpenCV-Python interface. At the same time, OpenCV-Python also implements the interface specification with Numpy library, which is very convenient to use Numpy in Python. For example, the data structure of Numpy can be transferred to OpenCV, and the data structure of OpenCV can also be transferred to Numpy. In addition, it can be used in collaboration with SciPy and Matplotlib, which can be used in a wider range. Therefore, OpenCV-Python is a very suitable tool for developing visual prototypes and experiments.

2.5. **Zbar and Python-Zbar.** Zbar is a bar code scanning and decoding library. A comprehensive software suite for reading barcodes. Supports EAN/UPC, Code 128, Code 39, Interleaved 2 of 5 and QR Code. Zbar includes libraries and applications for decoding captured barcode images and using a video device (eg, webcam) as a barcode scanner.

3. **Research Methods.** Our research method is based on Python, OpenCV and Zbar under the platform of PC Windows 7 system and QPython for Android system AR smart glasses. The developed program mainly includes the following functions.
1) Locating the barcode from the video data coming from the AR intelligent glasses lens.
2) Extraction and decoding of barcode.
3) After decoding the bar code, the information is displayed on the silicone display screen of AR smart glasses. Through the wireless transmission devices (such as WiFi, Bluetooth, SIM card) on the wearable micro-computer, the decoded information is transmitted to

the back-end enterprise management system (such as shop flow control system, manufacturing execution system) database.
4) The enterprise management system (such as shop flow control system and manufacturing execution system) will compare with the data set by the system. If any error is found, the correct or error message will be sent to the wearable microcomputer through the wireless network device, and displayed on the display of AR smart glasses to remind the field operators to confirm the normal operation or take corrective actions.
Regarding the back-end enterprise management system (such as shop flow control system, manufacturing execution system), this research will not be developed completely, but will design and develop a simple database and application program, mainly to verify the correctness of barcode data. Figure 2 shows the system function operation process flow.

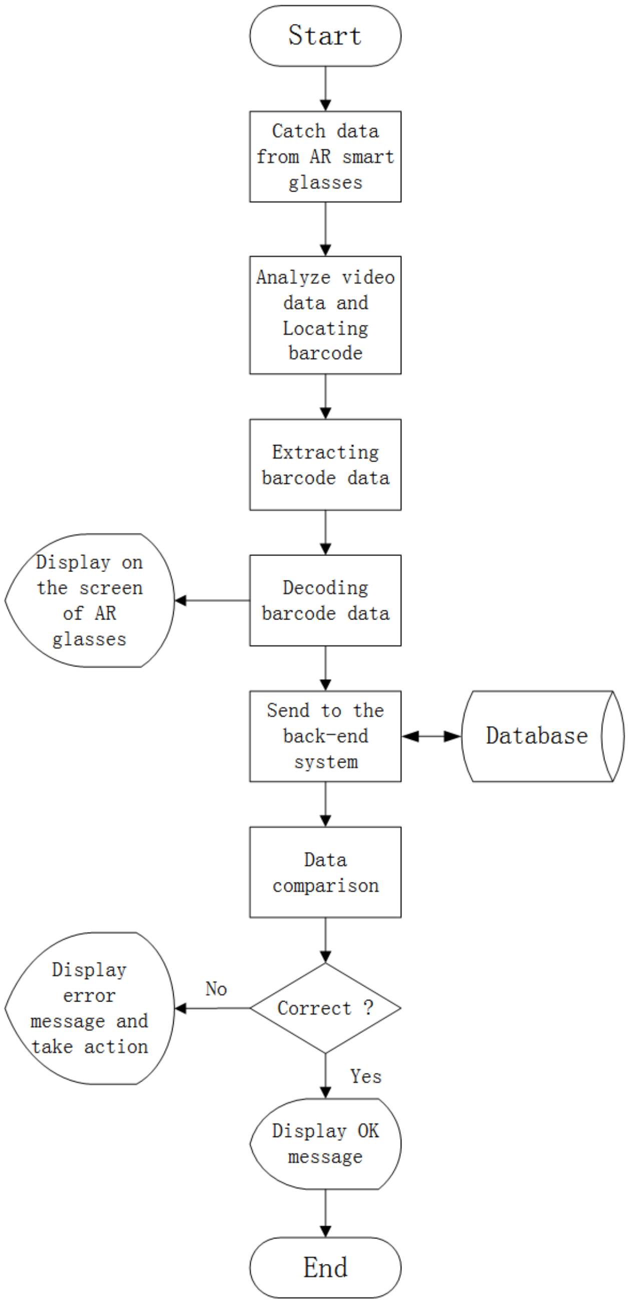

FIGURE 2. The system operation process flow

3.1. Dependent Packages Installation. First, we have to install dependency packages in PC in order to code and run python program. Figure 3 is the dependency packages which need to install.

```
pip install pillow
pip install opencv-python
pip install opencv-contrib-python
pip install numpy
pip instal pyzbar
```

FIGURE 3. Dependent packages installation

3.2. Camera Test. In order to open the camera for testing, we need to import Opencv-python. Figure 4 is the Python sample code.

```
import numpy as np
import cv2
capture = cv2.VideoCapture(0,cv2.CAP_DSHOW)
capture.set(3,640)
capture.set(4,480)
while True:
   # Read the image in the camera.
   ret,img = capture.read()
   cv2.imshow('frame', img)
   k = cv2.waitKey(1)
   if k == 27:    # 'ESC'关闭
      break
```

FIGURE 4. Sample code for camera test

After the camera open test is ok, we start the formal coding. Figure 5 is Python sample code.

```
import cv2
from PIL import Image, ImageDraw, ImageFont
import csv
import pyzbar.pyzbar as pyzbar
import numpy
import time
```

FIGURE 5. Sample code to import OpenCV-Python and Python-Zbar

3.3. Image preprocessing. After AR smart glassed fetching the image, in order to get better verification, it may need to implement the preprocess. Generally, the preprocess stages of barcode are shown as Figure 6.

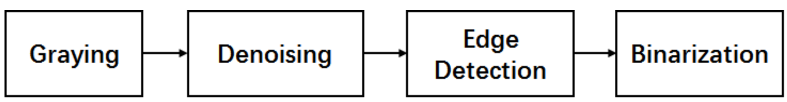

FIGURE 6. The flow of preprocess

3.3.1. *Graying.* Most of the images captured by the camera function terminal are RGB images. The method of converting RGB images via a certain calculation method into a grayscale image is called graying process. The images can be equalized to three matrices R, G and B. The value of each element of each matrix is between 0-255. After graying, the amount of image data is greatly reduced. This can improve the processing speed. Figure 7 is the graying program.

```
gray = cv2.cvtColor(frame,cv2.COLOR_BGR2GRAY)
```

FIGURE 7. Graying program of OpenCV-Python

3.3.2. *Denoising.* Image noise is the random change of image quality. It is the quality change of the object itself and the redundant interference information. This noise is caused by (1) image acquisition process: digital camera sensors and circuits, non-standard operation, (2) image signal transmission process: defects of transmission medium and recording equipment. Therefore, image may contain impulse noise, particle noise, Gaussian noise and other types of noise. In the image preprocessing stage, we need to remove these noises by means of filtering. There are many kinds of filtering methods such as mean filter, Wiener filter, median filter, Gaussian filtering and wavelet denoising, are used to remove all kinds of noises. Considering the noise type of the image acquired by AR smart glasses and the requirement of barcode location and recognition, in the denoising process stage, the gray image is filtered by Gaussian filter. Figure 8 is the denoising process program.

```
blur = cv2.GaussianBlur(gray,(5,5),0)
```

FIGURE 8. Denoising process program

3.3.3. *Edge detection.* Edge detection is an important part of digital image processing. If the gray value of the pixel in the image has a step change, the set of these pixels will be changed. The bar code is composed of a series of black-and-white "bars" and "blanks", and the edge is the edge of the image. Because of the strong edge attribute, the use of edge detection method is conducive to the positioning of bar code. In the field of digital image processing, Sobel, Kirsch, Prewitt, Canny and Laplacian can be used for edge detection. We will use Sobel to do the edge detection. Technically, it is a discrete difference operator, which is used to calculate the approximate gray value of image brightness function. Using this operator at any point of the image will produce the corresponding gray vector or its normal vector. Figure 9 is the edge detection program.

```
gradX = cv2.Sobel(blur, ddepth = cv2.CV_32F, dx = 1, dy = 0, ksize = -1)
gradY = cv2.Sobel(blur, ddepth = cv2.CV_32F, dx = 0, dy = 1, ksize = -1)
gradient = cv2.subtract(gradX, gradY)
```

FIGURE 9. Edge detection program

3.3.4. *Binarization.* Due to the influence of light and other factors in the process of image acquisition, the image contains some effective information and background information. In order to extract the effective information from the image, the binary algorithm can be implemented. In this processing, that is, using threshold selection algorithm Select the threshold T and process the gray image according to the following formula to make the image appear Obvious black and white effect. Binarization can quickly distinguish the foreground and background of the image scene, get the feature area of interest. Figure 10 is the binarization program.

```
(_, thresh) = cv2.threshold(gradient, 250, 255, cv2.THRESH_BINARY)
cv2.imshow("threshold_Image", thresh)
```

FIGURE 10. Binarization program

3.4. **Barcode location and interception.** Because there is a certain distance between the AR smart glasses and the barcode, there is useless background information around the barcode in the image, which increases the difficulty of barcode recognition. By locating and intercepting the barcode, the background can be removed, and only the barcode region can be extracted for subsequent decoding, so as to improve the recognition rate of the barcode.
Many scholars have proposed a lot of barcode location and interception methods, such as edge line extraction, cosine discrete transform and so on, which are suitable for different application scenarios. Due to the single environment in the workshop, the bar code itself "bar" and "empty" constitute a rectangular area, so we choose a more convenient morphological processing method. Morphological processing refers to the use of methods such as erosion, dilation, thinning, open operation and closed operation to process images, in which erosion and dilation are two basic operations. Figure 11 is the dilation program.

```
# Morphological processing, the definition of rectangular structure
kernel = cv2.getStructuringElement(cv2.MORPH_RECT, (10, 5))

# Expanding image, connecting breakpoints
closed = cv2.dilate(thresh, kernel, iterations = 1)
```

FIGURE 11. Binarization program

3.5. **Barcode decoding.** Because the bar code coding rules are clear, and there are open source modules for barcode decoding, the use of such open source modules can greatly reduce the difficulty of programming, without additional familiarity with all kinds of coding rules, high decoding accuracy and speed. Therefore, after the above series of preprocessing, positioning and intercepting the barcode, the images taken in the AR smart glasses can be decoded by using Python-Zbar and the barcode number, type and other data will be returned after decoding. Figure 12 is the decoding method of Python-Zbar.

```
barcodes = pyzbar.decode(closed)
```

FIGURE 12. Binarization program

3.6. **Move Python program to AR smart glasses.** For easily coding and testing, we develop the Python program in PC environment via PyCharm IDE. After finishing protype, we will move the program source to Android based AR smart glasses via QPython application. Figure 13 is the UI of QPython. We can use FTP function to move source code from PC to AR smart glasses. We also have to install OpenCV-Python, Python-Zbar in the AR smart glasses via PIP Console or QPYPI. Then we can test run Python program on AR smart glasses.

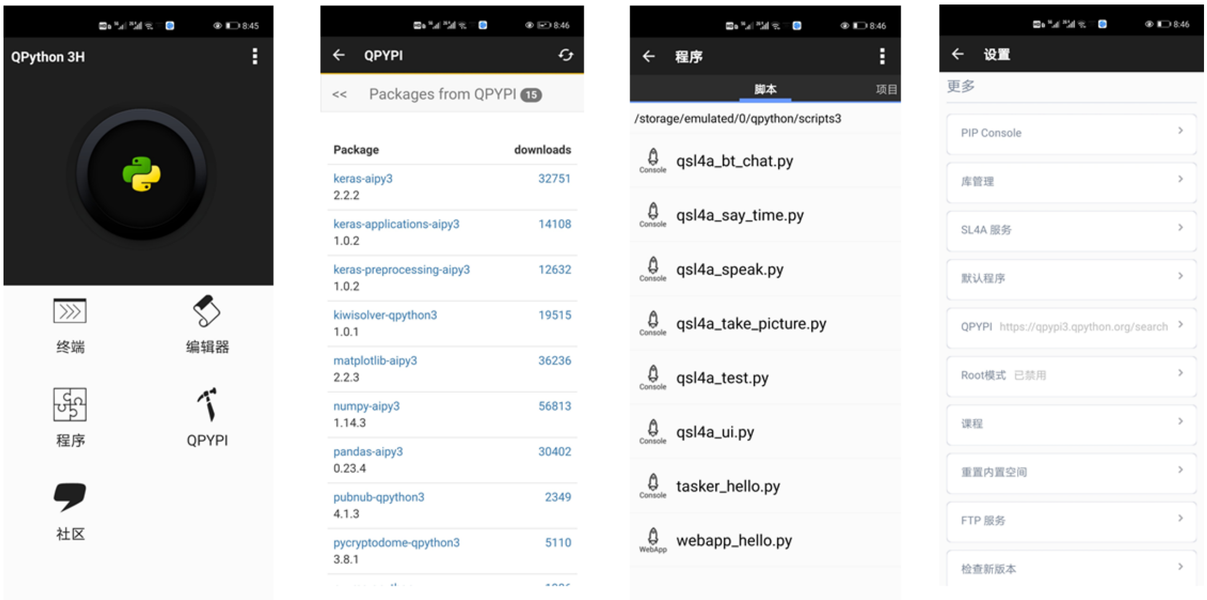

FIGURE 13. Binarization program

4. **Result Analysis and Conclusion.** After testing the different barcodes, only EAN8, EAN13, Code39, Code128, I25, CODEBAR, UPC-A, UPC-E and QR Code can be correctly read out. Figure 14 to Figure 16 are the tested results. However, the types of UPC-A and UPC-E are verified as EAN13. The reasons that tested barcodes are not clear. It also may have a lot of noise in the barcode, so that our program can not read out. We will look for more fine 1-D barcodes such as Code25 and 2-D barcode such as Code16k, Code49, PDF417, Code One, Maxicode, Data Matrix, Aztec, Vericode, Ultracode for testing and fix out the decoding problems via image preprocess. The successful rate can also be measured for each barcode in the near future. Although we have designed and developed a multiple barcode scanned system, we have not practiced and tested with SFCS and MES. We will cooperate with a manufacturing plant that has SFCS and MES to see the implemented results in the near future. For unmanned shopping malls, the barcodes recognition using AR smart glasses provides a solution for customers to quick check out by themselves.

```
Type: I25  Results: 10855996005624   Datetime: 2021/09/12-12:51:09
Type: I25  Results: 10855996005624   Datetime: 2021/09/12-12:51:09
Type: I25  Results: 10855996005327   Datetime: 2021/09/12-12:51:09
Type: I25  Results: 10855996005327   Datetime: 2021/09/12-12:51:09
Type: I25  Results: 10855996005327   Datetime: 2021/09/12-12:51:09
Type: I25  Results: 10855996005327   Datetime: 2021/09/12-12:51:09
Type: EAN8  Results: 69012341   Datetime: 2021/09/12-13:00:08
Type: EAN8  Results: 69012341   Datetime: 2021/09/12-13:00:08
Type: EAN8  Results: 69012341   Datetime: 2021/09/12-13:00:08
Type: EAN13  Results: 6901234567892   Datetime: 2021/09/12-13:00:08
Type: EAN13  Results: 6901234567892   Datetime: 2021/09/12-13:00:08
Type: EAN8  Results: 69012341   Datetime: 2021/09/12-13:00:08
Type: EAN13  Results: 0089000006007   Datetime: 2021/09/12-13:00:08
Type: EAN13  Results: 0089600124569   Datetime: 2021/09/12-13:00:08
```

Figure 14. Recognized resulted of I25, EAN8 and EAN13

```
Type: EAN13  Results: 0089600124569   Datetime: 2021/09/12-13:00:08
Type: EAN13  Results: 0089000006007   Datetime: 2021/09/12-13:00:08
Type: EAN13  Results: 6901234567892   Datetime: 2021/09/12-13:00:08
Type: EAN8  Results: 69012341   Datetime: 2021/09/12-13:00:08
Type: EAN13  Results: 0089600124569   Datetime: 2021/09/12-13:00:08
Type: EAN13  Results: 0089000006007   Datetime: 2021/09/12-13:00:08
Type: EAN13  Results: 6901234567892   Datetime: 2021/09/12-13:00:08
Type: EAN8  Results: 69012341   Datetime: 2021/09/12-13:00:08
Type: CODABAR  Results: A12345678A   Datetime: 2021/09/12-13:03:19
Type: CODABAR  Results: A12345678A   Datetime: 2021/09/12-13:03:19
Type: CODABAR  Results: A12345678A   Datetime: 2021/09/12-13:03:19
Type: CODABAR  Results: A12345678A   Datetime: 2021/09/12-13:03:19
Type: CODABAR  Results: A12345678A   Datetime: 2021/09/12-13:03:19
Type: CODABAR  Results: A12345678A   Datetime: 2021/09/12-13:03:19
Type: QRCODE  Results: https://u.wechat.com/MLLY4ZKDd4E-kF0ubtfkmCo   Datetime: 2021/09/12-13:04:47
Type: QRCODE  Results: https://u.wechat.com/MLLY4ZKDd4E-kF0ubtfkmCo   Datetime: 2021/09/12-13:04:47
```

Figure 15. Recognized resulted of EAN13(UPCA), EAN13(UPCE), CODABAR and QRCode

```
Type: CODE39  Results: 3PRM8N   Datetime: 2021/09/12-22:50:16
Type: CODE39  Results: 3PRM8N   Datetime: 2021/09/12-22:50:16
Type: CODE39  Results: CODE39   Datetime: 2021/09/12-22:50:16
Type: CODE39  Results: CODE39   Datetime: 2021/09/12-22:53:48
```

Figure 16. Recognized resulted of Code39

5. **Acknowledgment.** This paper is supported by science and technology project of Zhangzhou City, Fujian Province. Project No.: ZZ2019J23.

Table 1. Modules of OpenCV3.20

Module	Function description
Core	The core functionality contains the definition of core data structure and function
Imgproc	Image Processing: graphical filtering, Geometry conversion, Shape analysis
imgcodecs	Image file reading and writing
Highgui	High-level GUI and Media I/O
Calib3d	Camera Calibration and 3D Reconstruction
Feartur2d	Feature Detection and Description
Videoio	Media I/O
Video	Video Analysis
Objectect	Object Detection
ML	Machine Learning
Flann	Clustering and Search in Multi-Dimensional Spaces
Photo	Computational Photography
CUDA	CUDA-accelerated Computer Vision
Cudaarithm	CUDA-accelerated Operations on Matrices
Cudabgsegm	CUDA-accelerated Background Segmentation
Cudacodec	CUDA-accelerated Video Encoding/Decoding
Cudafeatures2d	CUDA-accelerated Feature Detection and Description
cudafilters	CUDA-accelerated Image Filtering
cudaimgproc	CUDA-accelerated Image Processing
cudaoptflow	CUDA-accelerated Optical Flow
cudastereo	CUDA-accelerated Stereo Correspondence
cudawarping	CUDA-accelerated Image Warping
Shape	Shape Distance and Matching
Superres	Super Resolution
Videostab	Video Stabilization
Viz	3D Visualizer
datasets	Framework for working with different datasets. (The datasets module includes classes for working with different datasets: load data, evaluate different algorithms on them, contains benchmarks, etc)
face	Face Recognition
optflow	Optical Flow Algorithms
reg	Image Registration
rgbd	RGB-Depth Processing
Surface-Matching	Surface Matching
Text	Scene Text Detection and Recognition
tracking	Tracking API
xfeatures2	Extra 2D Features Framework
ximgproc	Extended Image Processing
xobjdetect	Extended object detection
xphoto	Additional photo processing algorithms

REFERENCES

[1] T. Ronald and A. Azuma, A survey of augmented reality, *Presence*, vol. 6, No. 4, pp.355–385, 1997.

[2] M. Billinghurst, A. Clark, and G. Lee, A Survey of Augmented Reality, *Foundations and Trend in Human-Computer Interaction*, vol. 8, No. 2–3, pp.273–272, 2014.

[3] U. Dilek and M. Erol, Detecting position using ARKit, *Physics Education*, vol. 53, No. 2, pp.1–4, 2018.

[4] X. Zhang, X. Yao, Y. Zhu and F. Hu, An ARCore Based User Centric Assistive Navigation System for Visually Impaired People, *Applied Sciences*, vol. 9, No. 5, pp.1–15, 2019.

[5] M.J. Keenen and A.W. Nusbaum, Barcode uses and abuses, *OSTI.GOV. https://www.osti.gov/biblio/755623*, 2000.

[6] S. Hong-Ying, The Application of Barcode Technology in Logistics and Warehouse Management, *First International Workshop on Education. DOI: 10.1109/ETCS.2009.698*, pp.735–735, 2009.

[7] J. Zhang, X. Min, and J. Jia, Fine localization and distortion resistant detection of multi-class barcode in complex environments, *Multimed Tools Applications*, vol. 80, No. 11, pp.16153–16172, 2020. https://doi.org/10.1007/s11042-019-08578-x.

[8] L. Belussi and N. Hirata, Fast QR code detection in arbitrarily acquired images, *In: SIBGRAPI conference on graphics, patterns and images (SIBGRAPI)*, pp.281–288, 2011.

[9] M. Katona and L.G. Ny'ul, A novel method for accurate and efficient barcode detection with morphological operations, *In: International conference on signal image technology and internet based systems (SITIS)*, pp.307–314, 2012.

[10] G. Soros and C. Florkemeier, Blur-resistant joint 1d and 2d barcode localization for smartphones, *In: International conference on mobile and ubiquitous multimedia (MUM)*, vol. 11, pp.1–8, 2013.

[11] I. Szentandr'asi, A. Herout, and M. Dubsk'a, Fast detection and recognition of QR codes in high-resolution images, *In: Conference on computer graphics (CCG)*, pp.129–136, 2012.

[12] W. Xu and S. McCloskey, 2D barcode localization and motion deblurring using a flutter shutter camera, *In: IEEE Workshop on applications of computer vision (WACV)*, pp.159–165, 2011.

[13] A. Zamberletti, I. Gallo, and S. Albertini, Robust angle invariant 1d barcode detection, *In: Asian conference on pattern recognition (ACPR)*, pp.160–164, 2013.

[14] T. Grosz, P. Bodnar P, L. Toth, and L.G. Nyul, QR code localization using deep neural networks, *In: IEEE International workshop on machine learning for signal processing (MLSP)*, pp.1–6, 2014.

[15] T.Y. Lin, P. Dollar, R. Girshick, K. He, B. Hariharan, and S. Belongie, Feature pyramid networks for object detection, *In: The IEEE conference on computer vision and pattern recognition (CVPR)*, pp.2117–2125, 2017.

[16] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C, Fu, and A.C. Berg, Ssd: single shot multibox detector, *In: European conference on computer vision (ECCV)*, pp.21–37, 2016.

[17] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, You only look once: unified real-time object detection, *In: The IEEE Conference on computer vision and pattern recognition (CVPR)*, pp.779–788, 2016.

[18] S. Ren, R, He, K. Girshick, and J. Sun, Faster r-cnn: towards real-time object detection with region proposal networks, *In: Neural information processing systems (NIPS)*, pp.91–99, 2015.

[19] M. Busta, L. Neumann,and J. Matas, Deep textspotter: an end-to-end trainable scene text localization and recognition framework, *In: Proceedings of the IEEE international conference on computer vision (ICCV)*, pp.2204–2212, 2017.

[20] L.M. Akanbi, M.T. Bashorun, U.A. Salihu, G.O. Babafemi, K. Sulaiman, and S.O. Kolajo, Application of Barcode Technology in Landmark University Centre for Learning Resources, *Omu-Aran Experience, Library Philosophy and Practice (e-journal)*, 2018, 2088. http://digitalcommons.unl.edu/libphilprac/2088

[21] H. Jin, S. Jin, and J.F. Wu, An Amorphous 2-Dimensional Barcode, *Journal of Cyber Security*, vol.2, No.1, pp.37–48, 2020.

[22] X.R. Wang, X.G. Wang, and Q. Li, *OpenCV and Visual Studio*, Posts and Telecom Press., 2019

[23] C. Ivan, A. David, P. Tomislav, D. Hrvoje, and C. Mario, A brief introduction to OpenCV, *2012 Proceedings of the 35th International Convention MIPRO*, pp.1725–1730, 21–25 May 2012.