applied sciences

MDPI

Article

Data Extraction Method for Industrial Data Matrix Codes Based on Local Adjacent Modules Structure

Licheng Liao [1,2], Jianmei Li [1,2,*] and Changhou Lu [1,2]

[1] School of Mechanical Engineering, Shandong University, Jinan 250061, China; 201913918@mail.sdu.edu.cn (L.L.); luchh@sdu.edu.cn (C.L.)
[2] Key Laboratory of High-Efficiency and Clean Mechanical Manufacture, Shandong University, Ministry of Education, Jinan 250061, China
* Correspondence: lijianmei@sdu.edu.cn

Abstract: A 2D barcode is a reliable way to provide lifetime traceability of parts that are exposed to harsh environments. However, there are considerable challenges in adopting mobile cameras to read symbols directly marked on metal surfaces. Images captured by mobile cameras are usually of low quality with poor contrast due to the reflective surface of 2D barcode symbols. To deal with this problem, a novel laser-marked Data Matrix symbols reading method based on deep learning is proposed for mobile phone captured images. Utilizing the barcode module features, we train different convolutional neural network (CNN) models to learn the colors of two adjacent modules of a Data Matrix symbol. Depending on whether the colors of the two adjacent modules are the same or not, an edge image is transformed from a square grid, which is the same size as the barcode. A correction method based on the KM algorithm is used to get a corrected edge image, which helps to reconstruct the final barcode image. Experiments are carried out on our database, and the results show that the proposed algorithm outperforms in high accuracy of barcode recognition.

Keywords: metal surface; barcode; data matrix; data extraction; deep learning

Citation: Liao, L.; Li, J.; Lu, C. Data Extraction Method for Industrial Data Matrix Codes Based on Local Adjacent Modules Structure. *Appl. Sci.* **2022**, *12*, 2291. https://doi.org/10.3390/app12052291

Academic Editors: Byung-Gyu Kim, Antonio Fernández-Caballero and Hugo Pedro Proença

Received: 21 December 2021
Accepted: 18 February 2022
Published: 22 February 2022

1. Introduction

Two-dimensional barcode symbols have been widely used in automated identification applications, such as industrial components retrospect [1], mobile robots [2], and Construction Safety Management [3]. Compared to the 1D barcode, 2D barcode symbols have more popular advantages, such as large data capacity, compact size, and strong fault tolerance [4]. With the rapid advancement of sensing capability and computational power in consumer electronic devices, the use of mobile cameras for barcode recognition has attracted significant attention from both academia and industry [5]. The mobile device can easily interact with the internet systems by identifying the physical code tag. Another important application is data transfer by applying barcodes captured by mobile phones [6]. At present, the most common printing 2D barcode symbols in daily life can be decoded rapidly and accurately by mobile phones.

Many industrial Data Matrix barcodes are marked onto reflective, curved, or uneven surfaces, which can be challenging for barcode readers to identify reliably. Images captured by mobile cameras usually being of low quality with poor contrast, as well as low image contrast and complex code background, also make the problem of automatic information extraction extremely challenging. Decoding the industrial Data Matrix symbols quickly and reliably has become an important topic in the machine vision community. Most industrial Data Matrix symbols suffer from problematic artifacts, including exposure to aggressive environments, uneven lighting, curved surface, low contrast, rich texture, and blur, as shown in Figure 1.

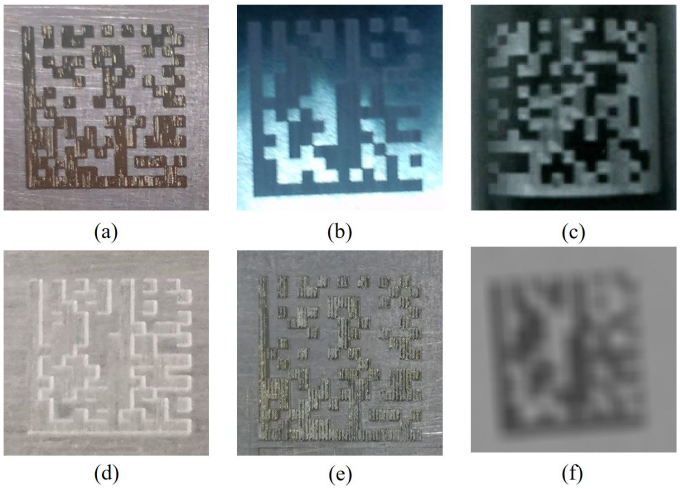


Figure 1. Some low-quality Data Matrix symbols: (**a**) exposure to aggressive environments; (**b**) uneven lighting; (**c**) curved surface; (**d**) low contrast; (**e**) rich texture; (**f**) blur.

To address this issue, we introduce a data extraction method for processing low-quality DM images. Different convolutional neural network (CNN) models are used to recognize the black and white modules of a barcode. Unlike the traditional methods that predict the module color based on pixels of each module, we train convolutional neural networks to learn and verify the colors of two adjacent modules. The relations of modules are used to transform a square grid image into an edge image. After correcting the edge image, the barcode symbol is reconstructed, and the final result will be read by decoders. The advantages of the proposed method are selecting effective features without losing too much information when labeling modules.

Our contributions include: (1) comparing different CNN models that help extract information from low-quality DM images; (2) unlike the traditional methods that predict the DM features based on pixels of each module, we use features of pairs of adjacent modules of a DM image to increase the accuracy of the data extraction; and (3) edge images are constructed and corrected by the proposed correction method based on the KM algorithm.

The remaining parts of the paper are organized as follows: Section 2 introduces related works. Section 3 presents the graphic structure of the DM symbol. Section 4 introduces barcode image pre-processing, and Section 5 describes the proposed data exaction method. The experiment results and analysis are presented in Section 6. Finally, the conclusion of this work is drawn in Section 7.

2. Related Works

To the best of our knowledge, known decoding algorithms, such as ZBar, ZXing, and libdmtx, cannot directly recognize degraded barcode symbols. So, some studies grade barcode quality through quality evaluation methods and then directly reject recognizing low-quality barcodes by decoding software. ISO-15415 [7] and ISO-16022 [8] classify the quality of 2D Data Matrix symbols into 5 grades based on some hand-crafted features. Chen et al. proposed a no-reference quality measure method for 2D barcodes captured by the mobile phone in order to reject the low-quality symbols [9]. They further improved their work and proposed a method of reduced-reference quality metric [10]. However, these methods are limited to barcode symbols with blur and illumination artifacts. Che et al. [5] classified the distortion types and quality levels of low-quality barcode images by deep learning networks and then used different pre-processing methods to improve barcode recognition efficiency.

Some traditional decoding apparatuses based on machine vision, such as KEYENCE SR-1000 [4], and COGNEX DataMan [11], have proposed a few pre-processing methods suitable for low-quality barcode symbols. For example, KEYENCE SR-1000 captures sample images from it by adjusting exposure time, optical gain, and polarization filter. Then, each

filtered image, which is processed by common image filters, is passed to the decoding step to judge if the symbol can be recognized successfully.

Binarization is the most common pre-processing method for barcode images. Using a suitable binarization algorithm to pre-process the barcode image can improve the quality of the barcode image and facilitate the extraction of barcode data. Ouaviani et al. [12] used the average gray level in a neighborhood surrounding each pixel as the threshold on the basis of modifying the Niblack algorithm. Thielemann et al. [13] enhanced image contrast by Niblack algorithm, followed by thresholding using Ridler and Calvard's method. Liu et al. [6] employed a threshold to process the barcode video images. Parikh et al. [14] divided the barcode image into four blocks and normalized each block, accounting for the varying lighting conditions. Then, a fixed threshold was obtained to binarize the normalized image. Yang et al. [15] proposed a binarization method using the local window of adaptive location and size for low-quality barcode images captured by mobile phones. The latest binarization algorithm has been proposed. Chen et al. [16] presented a binarization based on pre-processing and fuzzy C-means (FCM) clustering for low-quality ESPI fringe patterns. Sukanthi et al. [17] proposed a modified bi-level thresholding (MBET) algorithm based on contrast enhancement and edge filtering. Castellanos et al. [18] combined neural networks and DA in order to carry out unsupervised document binarization. The above binarization algorithms can improve the quality of printed barcodes with non-uniform illumination, but they are not effective for low-quality barcodes marked on metal surfaces.

3. The Graph Structure of the DM Symbol

A DM symbol is a square array composed of black and white modules. The structure of the symbol comprises Finder Pattern, Timing Pattern, and data region, as shown in Figure 2. The Finder Pattern is composed of two solid adjacent borders in an "L" shape, and the Timing Pattern consists of alternating black and white modules along the right and top edges. The Finder Pattern can determine the barcode position and orientation, and the Timing Pattern provides a count of the number of rows and columns in the symbol. The data region contains encoded text or numeric data. As more data is encoded in a barcode, the number of modules increases. Barcode symbol sizes vary from 10 × 10 to 144 × 144 in the new version ECC200. By adding error correction codes, the symbols can be read even if they are partially damaged.

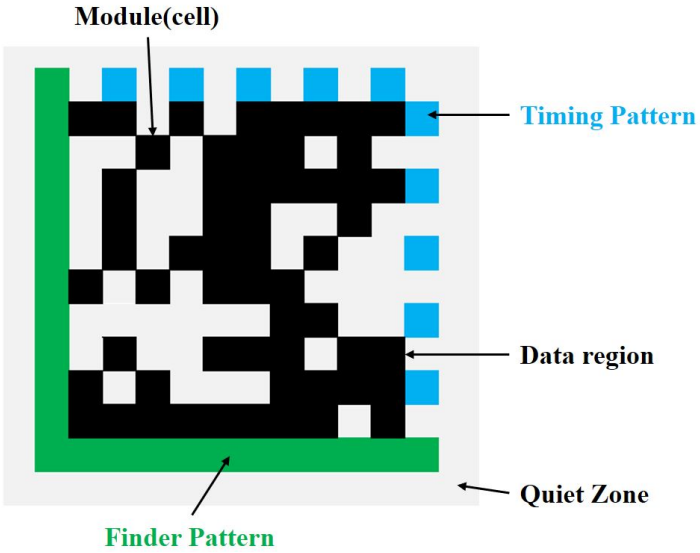


Figure 2. The structure of the DM symbol.

4. Barcode Image Pre-Processing

The general processing steps of barcode recognition consist of image pre-processing and data exaction. In the pre-processing stage, the Hough transform algorithm [19] is used to correct barcode images with geometric deformations. Then, an evenly spaced grid is overlaid onto the barcode by using the method [20] of the horizontal and vertical edge projections from gradient direction images, which divides the barcode into modules. The color of each module may be black or white, and a total number of modules can be calculated by the barcode size. Figure 3 shows the process of image pre-processing.

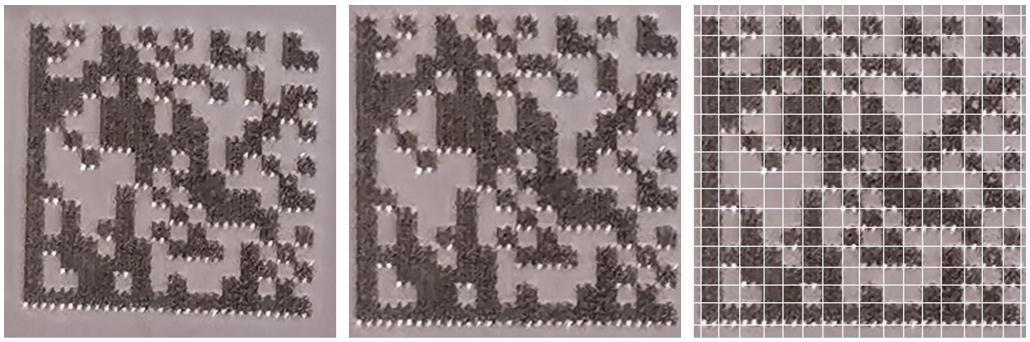


Figure 3. Image pre-processing procedure: (**a**) an original barcode image with geometric deformations; (**b**) barcode image after deformation correction; (**c**) the barcode image with an aligned grid.

5. Barcode Data Extraction

The traditional technology for extracting barcode information based on a single module divided by a barcode grid is shown in Figure 4. It is susceptible to the inaccurate placement of the gridline because of not wholly correcting the barcode with geometric deformations. The variance of pixel values in a module cannot be stable in a specific range. Therefore, it is a challenge to extract useful features for a low-quality barcode image to determine the colors of modules. Different from the present method, we utilize pairs of adjacent modules from the barcode, as shown in Figure 5a. The colors of modules of a pair are the same or different, which represents two category labels, as shown in Figure 5b. The comparison of the gray value of the two modules and whether there is an edge between the two modules are the main features to achieve classification. The process of the proposed data extraction method is divided into four steps, as shown in Figure 6.

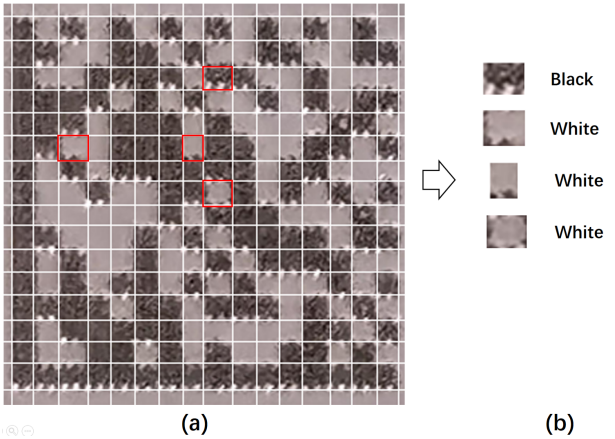


Figure 4. Traditional sampling: (**a**) sample by a single module; (**b**) sample examples and color labels.

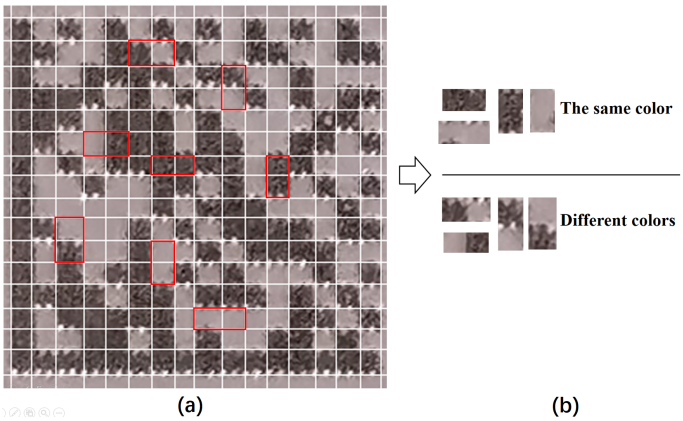


Figure 5. Proposal sampling: (**a**) sample by two adjacent modules; (**b**) samples and the classification labels.

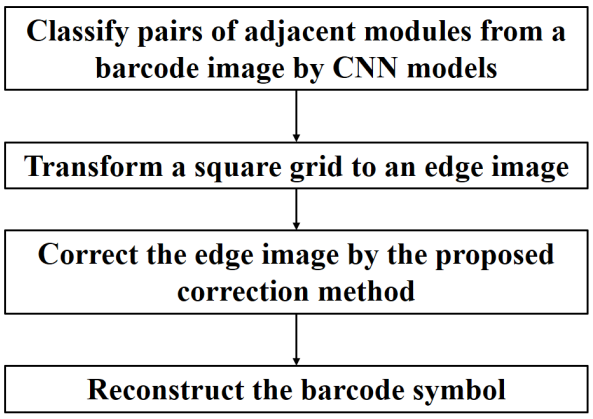


Figure 6. The flow chart of the proposed barcode data extraction method.

5.1. Classify Pairs of Adjacent Modules by CNN

Traditional image classification methods need handcrafted features and classifiers for training features, such as SVM and KNN, which have low classification accuracy, unsatisfactory effects, and weak adaptive ability. With the development of CNN, computational power, and computer vision, deep learning methods for image classification tasks have surpassed traditional algorithms in accuracy and real-time performance. Deep learning methods make it easy to analyze images with complex backgrounds for some specific tasks. A CNN system may have somewhere between 3 to 150 or even more layers and CNN layers can be of four main types: Convolution Layer, ReLu Layer, Pooling Layer, and Fully-Connected Layer.

Five CNN architectures are chosen for our task based on their present performance in image classification [21], including AlexNet [22], GoogleNet [23], VGG16 [24], VGG19 [24], and ResNet50 [25]. Table 1 shows their performance in ILSVRC challenge.

Table 1. Comparison between different CNN architectures for image classification.

ILSVRC Architectures	Number of Layers	Top 5 Error Rate	Training Dataset	Execution Environment
AlexNet (2012)	8	15.3%	ImageNet	Two GTX 580 GPUs 3Gg
GoogleNet (2014)	22	6.67%	ImageNet	CPU
VGGNet (2014)	16-19	6.8%	ImageNet	Four NVIDIA Titan Black GPU
ResNet (2015)	18-34-50-101-152	3.57%	ImageNet	Two GPUs

In order to sample pairs of adjacent modules from a barcode image, a sliding window algorithm is applied in the barcode image, which size is the same as the size of the two adjacent cells of the barcode grid. The sliding window with one grid stride scans the barcode image according to the position of the grid vertices. The sliding window has two states: horizontal and vertical. The horizontal sliding window in Figure 7a samples horizontal pairs of adjacent modules, and the vertical sliding window in Figure 7c samples vertical pairs of adjacent modules.

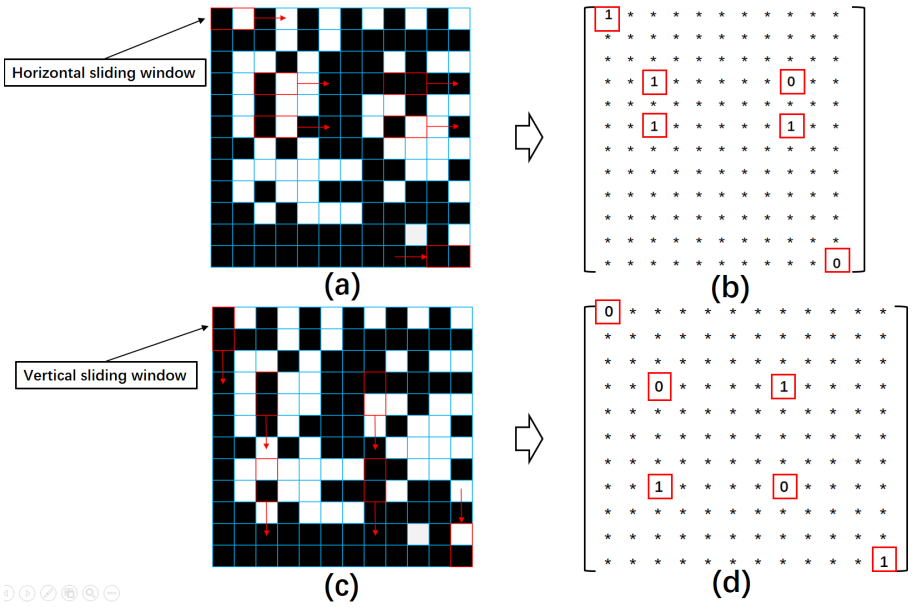


Figure 7. Demonstration of the inputs and outputs of the CNN model for a 12 × 12 DM barcode. (**a**,**c**) The horizontal and vertical sliding window sampling in the grid format of the barcode image, respectively; (**b**,**d**) the label matrix for the horizontal and vertical pairs, respectively.

We train a CNN network to learn the colors of a pair of two adjacent modules and classify the pairs into two categories according to whether the two adjacent modules have the same color or not. So, we use the pairs as the input of CNN models and use labels 0 and 1 as the output of CNN models. It is defined that label 0 represents the class of a pair with the same color, and label 1 represents the other class of a pair with different colors. Two matrices are used to save labels of horizontal and vertical pairs, respectively. The labels are stored in the matrix in sequence according to the original position of the sampled pairs of adjacent modules. Figure 7b,d show the horizontal and the vertical pair label matrix, respectively.

5.2. Transform a Square Grid Format Image to an Edge Image

Based on the two label matrices produced by the CNN models, a new square grid is constructed and transformed, which size is the same as the barcode size. The common gridlines of the two adjacent cells are transformed by using a sliding window algorithm again in the grid. A horizontal sliding window with the size of 1 × 2 cells is applied to scan the grid horizontally, as shown in Figure 8. We look up the label values in the horizontal pair label matrix for every step of the sliding window. If a label value is 0, the common edge of the two adjacent cells is removed. If a label value is 1, no additional operation is needed.

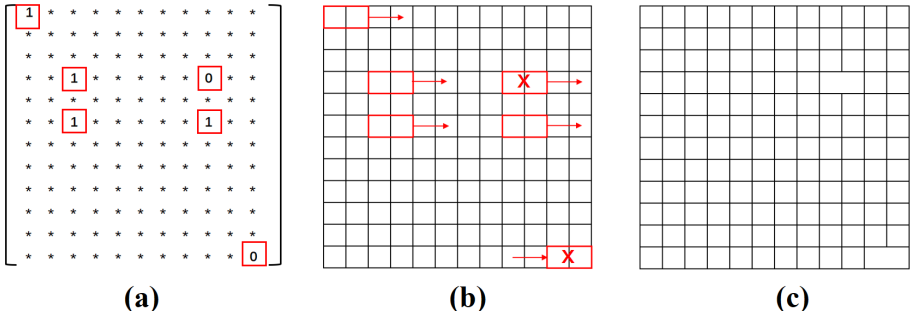


Figure 8. Demonstration of the first step of transforming the grid: (**a**) the horizontal pair label matrix; (**b**) scan and transform the grid; (**c**) the result of transformation.

After the horizontal scanning process, a vertical sliding window continues to work on the grid. As shown in Figure 9, a vertical sliding window with the size of 2 × 1 cells is used. The operation in the vertical scanning process is similar to the horizontal one. After implementing the scanning process in two directions, an edge image for the barcode is formed, as shown in Figure 10a. Then, the outermost borders of the grid in the edge image are removed or retained by using the information of the DM Timing Pattern, as shown in Figure 10b. In addition, Figure 10c shows the transformed edge image.

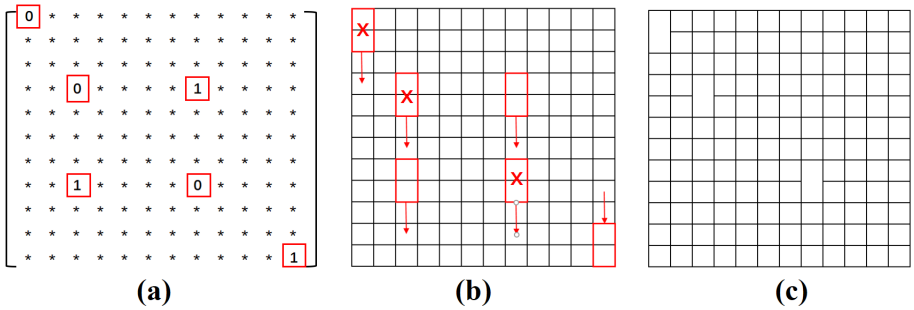


Figure 9. Demonstration of the second step of transforming the grid: (**a**) the vertical pair label matrix; (b) scan and transform the grid; (**c**) the result of transformation.

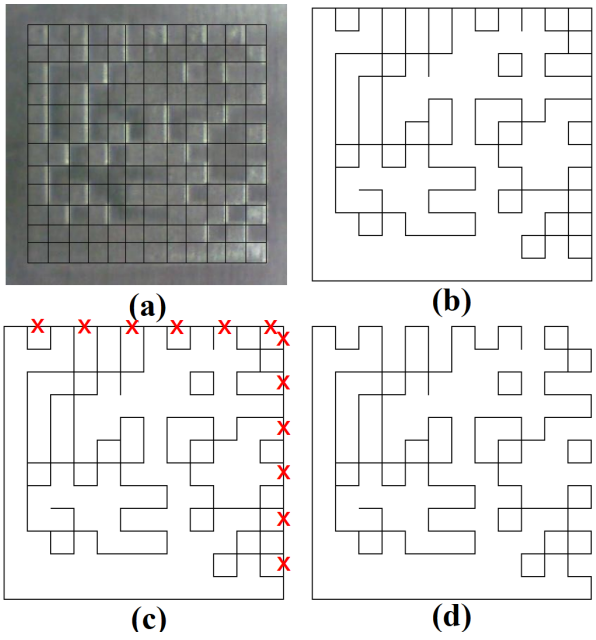


Figure 10. An edge image generation process: (**a**) a barcode image; (**b**) the edge image with solid line border; (**c**) modify the outermost border of the grid; (**d**) the modified edge image.

5.3. Transform a Square Grid Format Image to an Edge Image

Adding or removing the edge between two adjacent vertices will lead to increasing or decreasing degrees of the vertices in the edge image. It means that the wrong edges in the edge image are related to the degrees of the vertices, and a correction algorithm is proposed. The degree of any vertex in the standard barcode edge image can be only 0, 2, or 4, as shown in Figure 11b. When the edge image is not standard, there are some odd-degree vertices, such as 1 or 3, as shown in Figure 11c. Therefore, the proposed correction algorithm can correct the edge image by eliminating these odd-degree vertices. The process of the proposed correction algorithm is divided into two steps: (1) pairing odd-degree vertices in the edge image; and (2) eliminating odd-degree vertices in the edge image.

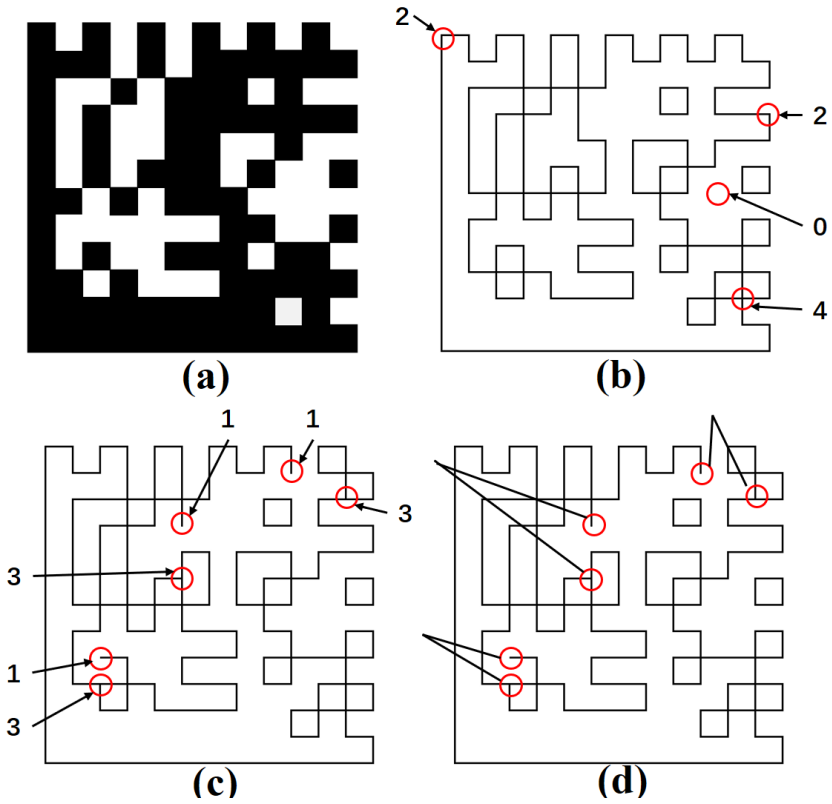


Figure 11. (**a**) A barcode image; (**b**) a standard barcode edge image; (**c**) odd-degree vertices in the edge image; (**d**) pair odd-degree vertices in the edge image.

5.3.1. Pair Odd-Degree Vertices in the Edge Image

A proper path is chosen to eliminate two odd-degree vertices in the edge image, which begins at one vertex and ends at another vertex. In the meantime, we should modify the edge image as little as possible, which means that the sum of the lengths of all paths in the edge image is minimum. Based on this principle, the knowledge of graph theory is used to pair these odd-degree vertices. Assuming that the number of these vertices is 2N, we calculate distances between these vertices and draw a weighted undirected graph. It is known that N edges can cover all vertices in the graph according to the edge-covering number of an undirected graph. In order to minimize the sum of the lengths of N edges, the proposed pairing method is used and costs less time than the exhaustive method, which takes the split point operation to turn a weighted undirected graph into a weighted bipartite graph and the KM (Kuhn–Munkres) algorithm [26] to find an approximate solution.

The flow of the KM algorithm is as follows.

For a bipartite graph $G = (V, E)$, V can divide two disjoint subsets (X, Y), and E is a set of edges.

1. Starting from any feasible vertex labeling L, determine the equality subgraph G_L of L and select the matching M in G_L. If M is a perfect matching, the algorithm stops. Otherwise, go to step 2.

2. Hungarian algorithm begins. When $N_{G_L}(S) = T$, with $S \in X$ and $T \in Y$, the Hungarian algorithm stops. Calculate αL. Determine a new feasible vertex labeling L'. Replace L with L', replace G_L with G, and then move to step 1.

In the KM algorithm, the Hungarian algorithm [27] is used. In addition, the Hungarian algorithm can solve the perfect matching of a bipartite graph. Its algorithm flow is as follows.

1. Select any matching M of G. If any vertex of X is the saturated vertex of M, the algorithm stops. Otherwise, take a point x of X and $x \notin M$. Let $S = \{x\}$ and $T = \emptyset$.

2. If $N(S) = T$, the algorithm stops. There is no perfect matching in G. If $N(S) \neq T$, take $y \notin N(S) \setminus T$.

3. If y is the saturated vertex of M, there exist $z \notin X \setminus S$ and $yz \in M$. Replace S with $S \cup \{z\}$, and replace T with $T \cup \{y\}$. Turn to step 2. If y is not the saturated vertex of M, there is an augmenting path P of M with x as the starting point and y as the ending point in G. Replace M with $M' = M \triangle E(P)$ and turn to step 1.

The KM algorithm pairs the odd-degree vertices in the edge image, as shown in Figure 11d.

5.3.2. Correct the Edge Image by Eliminating Pairs of Odd-Degree Vertices

We eliminate a pair of odd-degree vertices by selecting a proper path between them in the edge image. The proper path along the gridline to connect the pair. An XY coordinate system is used to describe positions of vertices in the edge image, as shown in Figure 12a. Suppose the points (X_1, Y_1) and (X_2, Y_2) belong to a pair of odd-degree vertices, the horizontal distance between the pair is $\mid X_2 - X_1 \mid$, and the vertical distance between the pair is $\mid Y_2 - Y_1 \mid$. If the horizontal distance or the vertical distance is zero, the proper path is the connection line between the pair. If the horizontal distance and the vertical distance are non-zero, the proper path is chosen by the conditions. The conditions are as follows: (1) the length of the path is minimum; and (2) the weight of the path is smallest when weight the edges of the grid in the edge image. The weight values are calculated when CNN models classify pairs of adjacent modules in the barcode image.

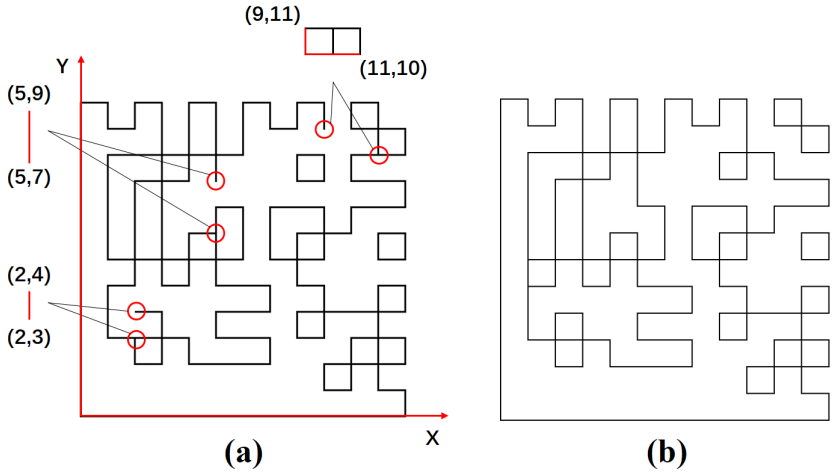


Figure 12. A final edge image forming process: (**a**) path selection in the edge image; (**b**) the corrected edge image.

After obtaining the proper path of any pair of odd-degree vertices, we should change the property of the path in the edge image. If there is an edge in the path, the edge will be removed, and, if there is not an edge in the path, an edge will be added. The final edge image is obtained after changing the property of all paths in the edge image, which does not include any odd-degree vertex, as shown in Figure 12b.

5.4. Barcode Reconstruction and Recognition

In the edge image, we determine the colors of modules of the Finder Pattern and the Timing Pattern based on the knowledge of the DM scheme, as shown in Figure 13a. Colors of modules of the data region can be recognized by the topological relationship in the edge image. The reconstruction result is shown in Figure 13b. In addition, Figure 13c shows some incorrect modules in the reconstructed barcode image. The black modules are correct, and the red modules are incorrect.

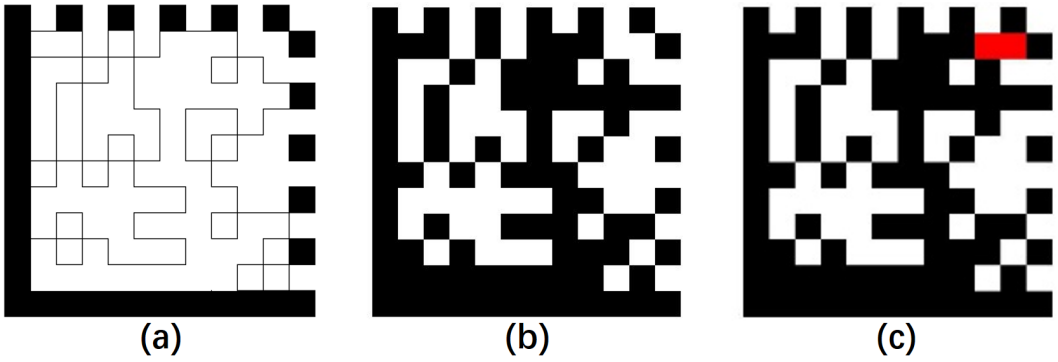


Figure 13. Barcode reconstruction. (**a**) Colors of modules of the Finder Pattern and the Timing Pattern; (**b**) the reconstructed barcode image; (**c**) incorrect modules in the reconstructed barcode image.

The required Reed–Solomon error correction built into Data Matrix ECC200 is able to reconstruct and verify the data scanned for improved accuracy. In addition, the decoding software can correctly decode these symbols with the incorrect modules. Table 2 shows the size, capacity, and error correction features of each Data Matrix format. If the number of error character codes of a barcode reconstruction symbol is not more than the max correctable error of the format, the barcode symbol can be correctly identified.

Table 2. The size, capacity, and error correction features of each Data Matrix format.

Symbol Size	Max Numeric Capacity	Max Alphanumeric Capacity	Max Binary Capacity	Max Correctable Error/Erasure
10 × 10	6	3	1	2
12 × 12	10	6	3	3
14 × 14	16	10	6	5/7
16 × 16	24	16	10	6/9
18 × 18	36	25	20	7/11

6. Experiment Results

Our algorithm experiments related to this article use MATLAB 2020a on Win10 operation system with Intel(R) Core(TM) i5-9400 and CPU @2.90GHz. The decoding software is Intellect 1.5.1, a machine vision software from COGNEX.

Samples used in this experiment come from the database of barcodes marked in the aluminum alloy by our laboratory. The marking equipment is Nd: YAG-T80C laser. The type of barcode is the Data Matrix ECC200 barcode. The barcode images are captured by mobile phones under indoor lighting in a lab. In addition, the camera's pixels are 20 million. We chose RGB as the image format, and the size of the images is 400 × 400. The size of our database is 250, including 52 barcodes with the size of 12 × 12, 37 barcodes with the size of 14 × 14, and 161 barcodes with the size of 16 × 16. We divide the samples into a training set and a test set. The size of the training set is 50, and the size of the test set is 200. Some samples of our database are shown in Figure 14.

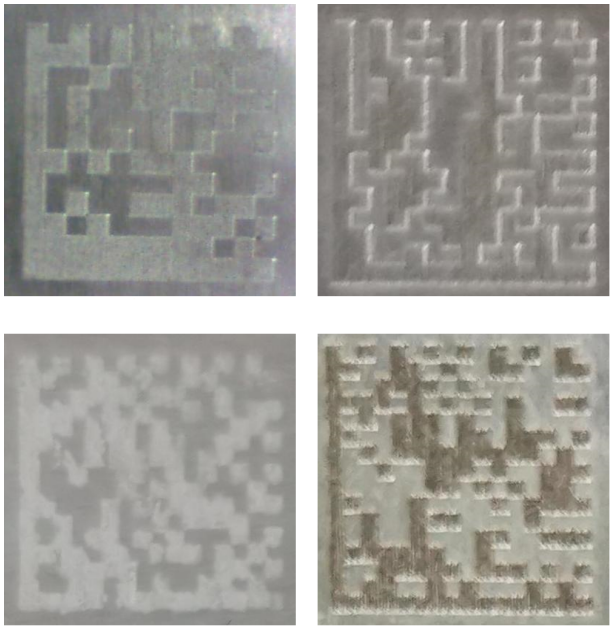


Figure 14. Some samples of our database.

6.1. Training and Verification of CNN

The training set size in this experiment is 50, and pairs of adjacent modules from the 50 barcode images consist of a CNN training and validation set. The number of total pairs of adjacent modules of a barcode image can be calculated by the formula: F = n*(n − 1)*2, where n is the size of the barcode. The CNN training and validation set, finally, contains 20,810 samples. Then, we manually mark these samples into two categories. One has 10,040 samples, and the other has 10,770 samples.

We use five CNN models for the CNN training and validation set, such as AlexNet, GoogleNet, VGG16, VGG19, and ResNet50. The period of the training experiment is 5, and the learning rate is 0.01. In addition, the batch size is 128. The classifier is tested in five-fold cross-validation. To evaluate the loss value of the training period and the accuracy of verification, they are displayed graphically, as shown in Figure 15. It can be found that the validation score follows the training score, which indicates that our results avoid overfitting in the training procedure and retain the generalization characteristics of the CNN.

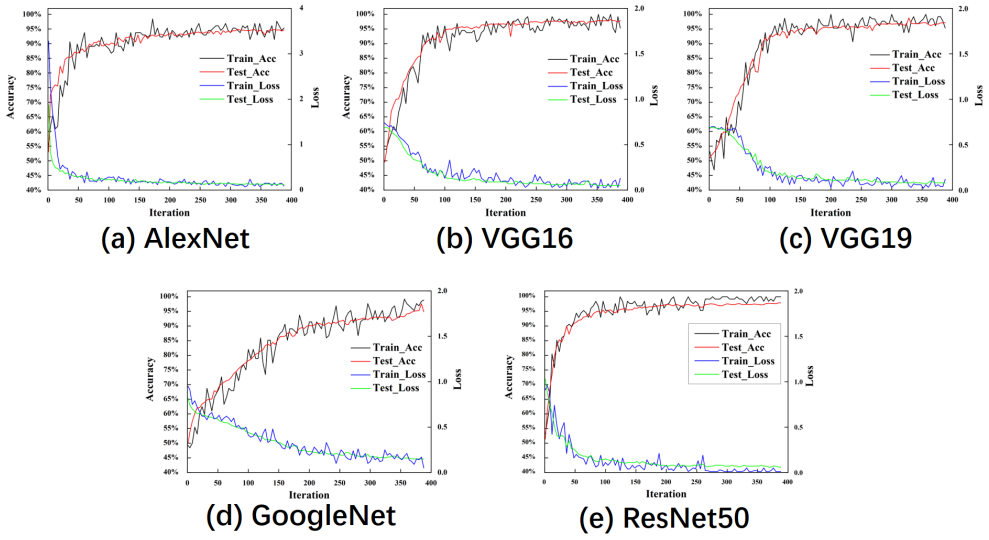


Figure 15. Change in accuracy and loss of five CNN models.

6.2. Test Results of CNN

The size of the test set for this experiment is 200, and the CNN test set contains 82,636 pairs of adjacent modules sampled from the test set. The recognition results of the CNN test set are shown in Table 3. All five CNN network models are used and achieved high accuracy rates. VGG19 achieves the highest accuracy rate of 99.5%, and GoogleNet achieves the lowest accuracy. Although the difference between GoogleNet and VGG19 is 1.6%, it is enough to impact barcode reconstruction and recognition significantly.

Table 3. The recognition results in the CNN test set.

Type of CNN Models	Total Number of Samples	The Number of Correct Identification	The Number of Incorrect Identification	Accuracy (%)
AlexNet	82,636	81,312	1324	98.4
GoogleNet	82,636	80,899	1737	97.9
VGG16	82,636	82,082	554	99.3
VGG19	82,636	82,183	453	99.5
ResNet50	82,636	81,208	1428	98.3

The performance of the five CNN models in the test set is discussed below. The classification results are used for edge images, and the number of wrong edges in an edge image compared with the standard barcode edge image can be used as an evaluation parameter for the performance. Figure 16 shows the distribution of the wrong edge number in the test set when using different CNN models. When the wrong edge number varies from 1 to 9, the five models have almost the same sample proportions. VGG19 has the largest sample proportion, of 45.5%, when the wrong edge number is 0. When the wrong edge number is over 10, GoogleNet has the largest sample proportion of 32.5%, while VGG19 has the smallest sample proportion of 4%. It shows that VGG19 has the best performance. This is because the fewer number of wrong edges in an edge image, the better it is for barcode reconstruction and recognition.

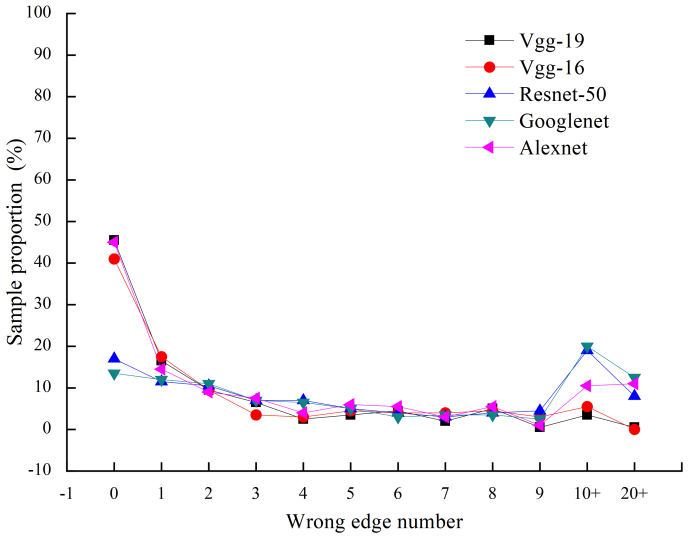


Figure 16. The performance of five CNN models in the test set.

6.3. The Performance of the Proposed Correction Algorithm in the Edge Images

After correcting the edge images, the sample proportions for five CNN models increase to more than 70%, of which, for VGG19, is 90%, when the wrong edge number is 0, as shown in Figure 17. The proposed correction algorithm fails to get the sample proportion of 100% for five CNN models, which means that some edge images still cannot be fully corrected and still contain wrong edges.

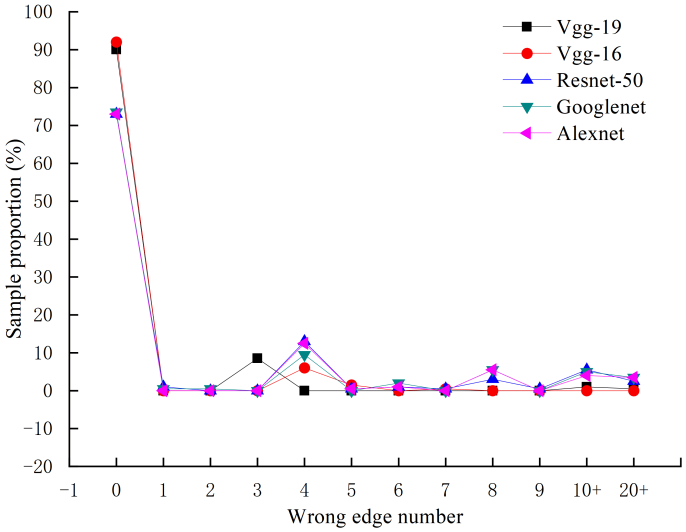


Figure 17. The performance of the correction algorithm in edge images.

6.4. The Result of Barcode Reconstruction and Recognition

Completely corrected edge images can reconstruct barcode symbols with no incorrect module. However, a few reconstructed barcode symbols which are reconstructed by the edge images with wrong edges have some incorrect modules. Figure 18 shows their distributions. If the incorrect module number is less than 9, the reconstructed barcode images have a high probability of being recognized by Reed–Solomon error correction. If the incorrect module number is more than 10, the reconstructed barcode images are difficult to identify. The best performance is based on VGG19, which hardly generates the reconstructed barcode images with more than ten modules.

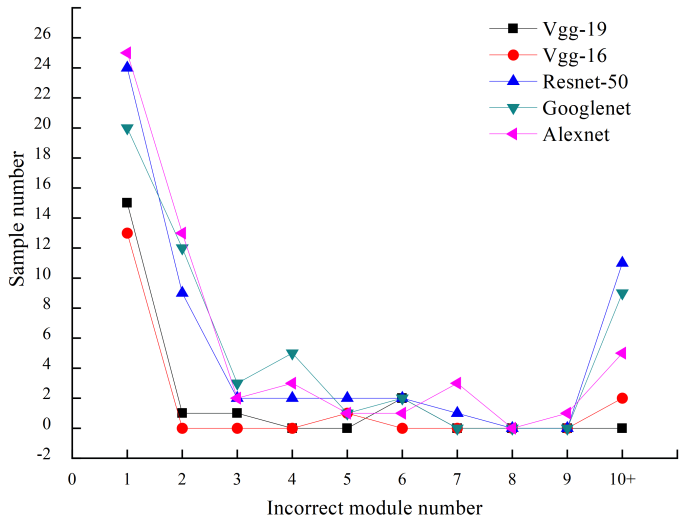


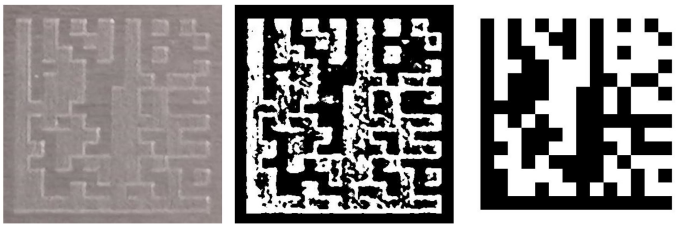
Figure 18. The distribution of incorrect module numbers in barcode reconstruction images with incorrect modules.

In Table 4, there are our results compared against competitive DXM decoding software (open-source and also commercial). These software examples do not have a high recognition rate for low-quality barcode images. However, our methods can achieve a recognition rate of more than 96%, and we obtain the recognition rate of 99.5% based on VGG19. One suggestion that can be adopted is to integrate our method into these softwares as a last resort in order to achieve better performance. Before the barcode image is input into the decoding software, Yang et al. [15] binarizes the image to remove the influence of illumination. The binarization result is input into Dynamsoft Barcode, and the decoding

rate increased from 51.5% to 78.5%. However, its recognition rate is lower than our method because barcode images marked on metal surfaces have worse quality, compared to the printed barcode images in [15]. The barcode pattern and background are often uneven, which is determined by the metal material properties, the surface texture, and the high reflectivity. Figure 19 shows that the grayscale of the barcode pattern is uneven due to texture, which affects the effect of binarization. However, our method has strong stability in extracting features.

Table 4. The recognition results in the CNN test set.

Software	The Size of the Set	The Number of Recognized Samples	Recognition Rate (%)
Google ZXing	200	3	1.5
Onbarcode.NET	200	10	5
Dynamsoft barcode	200	103	51.5
LEADTOOLS	200	63	31.5
Libdmtx	200	17	8.5
Inlite barcode	200	3	1.5
Yang [15]+Dynamsoft	200	175	78.5
Our solution (four networks)	200	>192	>96.5
our solution (VGG19)	200	199	99.5



(a) (b) (c)

Figure 19. (**a**) Original barcode image; (**b**) binarization result of Yang [15]; (**c**) our result.

7. Conclusions and Future Work

In this paper, a novel data extraction method based on local adjacent modules for images of industrial Data Matrix barcodes captured by smartphone cameras is presented. Different CNN models are established to learn the colors of the two adjacent modules and label them depending on whether they have the same color or not. Based on the label matrix of a Data Matrix image, an edge image is produced based on the KM algorithm. A Data Matrix symbol is, finally, reconstructed and decoded. Based on the experiments with our barcode image database, we achieve a 99.5% recognition rate by using CNN network VGG19. The execution environment of our experiment is the CPU. If GPU is used, the time-consuming issue can be kept within an acceptable range. The proposed method has good generalizability to images captured by different smartphone cameras and performs well under different lighting conditions. Experimental results on Data Matrix images demonstrate that our data extraction method can also be extended to barcode patterns with similar features. One interesting future work is to focus our method on extremely poor quality modules in barcode images, where other high-contrast modules are identified by a binarization algorithm to save decoding time.

Author Contributions: Conceptualization, L.L. and J.L.; project administration, C.L.; resources, C.L.; supervision, L.L. and J.L.; writing—review and editing, J.L. and C.L.; writing—original draft, L.L. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Not applicable.

Acknowledgments: This work was supported by the key R&D plan of Shandong Province (2019GGX104098).

Conflicts of Interest: The authors declare that there is no conflict of interest regarding the publication of this paper.

References

1. Xiao, Y.; Ming, Z. 1D barcode detection via integrated deep-learning and geometric approach. *Appl. Sci.* **2019**, *9*, 3268. [CrossRef]
2. Karrach, L.; Pivarčiová, E.; Bozek, P. Recognition of perspective distorted QR codes with a partially damaged finder pattern in real scene images. *Appl. Sci.* **2020**, *10*, 7814. [CrossRef]
3. Kim, J.S.; Yi, C.Y.; Park, Y.J. Image Processing and QR Code Application Method for Construction Safety Management. *Appl. Sci.* **2021**, *11*, 4400. [CrossRef]
4. Nadabar, S.G.; Desai, R. Method and Apparatus Using Intensity Gradients for Visual Identification of 2D Matrix Symbols. U.S. Patent 6,941,026, 6 September2005.
5. Che, Z.; Zhai, G.; Liu, J.; Gu, K.; Le Callet, P.; Zhou, J.; Liu, X. A blind quality measure for industrial 2d matrix symbols using shallow convolutional neural network. In Proceedings of the 2018 25th IEEE International Conference on Image Processing (ICIP), Athens, Greece, 7–10 October 2018; pp. 2481–2485.
6. Liu, X.; Doermann, D.; Li, H. VCode-Pervasive data transfer using video barcode. *IEEE Trans. Multimed.* **2008**, *10*, 361–371. [CrossRef]
7. *ISO/IEC-15415*; Information Technology-Automatic Identification and Data Capture Techniques-Bar Code Symbol Print Quality Test Specification-Two-Dimensional Symbols. ISO: Geneva, Switzerland, 2011; Volume 2.
8. *ISO/IEC-16022*; Information Technology-Automatic Identification and Data Capture Techniques-Data Matrix Bar Code Symbology Specification. ISO: Geneva, Switzerland, 2006; Volume 2.
9. Chen, C.; Kot, A.C.; Yang, H. A two-stage quality measure for mobile phone captured 2D barcode images. *Pattern Recognit.* **2013**, *46*, 2588–2598. [CrossRef]
10. Kang, L.; Ye, P.; Li, Y.; Doermann, D. Convolutional neural networks for no-reference image quality assessment. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Columbus, OH, USA, 23–28 June 2014; pp. 1733–1740.
11. Natsukari, C.; Nakata, H. Two-Dimensional Code Reader Setting Method, Two-Dimensional Code Reader, Two Dimensional Code Reader Setting Program and Computer Readable Recording Medium. U.S. Patent 6,983,886, 10 January 2006.
12. Ottaviani, E.; Pavan, A.; Bottazzi, M.; Brunelli, E.; Caselli, F.; Guerrero, M. A common image processing framework for 2-D barcode reading. *IEE Conf. Publ.* **1999**, *465*, 652–655.
13. Thielemann, J.T.; Schumann-Olsen, H.; Schulerud, H.; Kirkhus, T. Handheld PC with camera used for reading information dense barcodes. In Proceedings of the IEEE Int. Conference on Computer Vision and Pattern Recognition, Demonstration Program, Washington, DC, USA, 27 June–2 July 2004.
14. Parikh, D.; Jancke, G. Localization and segmentation of a 2D high capacity color barcode. In Proceedings of the 2008 IEEE Workshop on Applications of Computer Vision, Copper Mountain, CO, USA, 7–9 January 2008; pp. 1–6.
15. Yang, H.; Kot, A.C.; Jiang, X. Binarization of Low-Quality Barcode Images Captured by Mobile Phones Using Local Window of Adaptive Location and Size. *IEEE Trans. Image Process.* **2011**, *21*, 418–425. [CrossRef] [PubMed]
16. Chen, L.; Tang, C.; Xu, M.; Lei, Z. Binarization for low-quality ESPI fringe patterns based on preprocessing and clustering. *Appl. Opt.* **2021**, *60*, 9866–9874. [CrossRef] [PubMed]
17. Sukanthi.; Murugan, S.S.; Hanis, S. Binarization of Stone Inscription Images by Modified Bi-level Entropy Thresholding. *Fluct. Noise Lett.* **2021**, *20*, 2150054. [CrossRef]
18. Castellanos, F.J.; Gallego, A.J.; Calvo-Zaragoza, J. Unsupervised neural domain adaptation for document image binarization. *Pattern Recognit.* **2021**, *119*, 108099. [CrossRef]
19. Mukhopadhyay, P.; Chaudhuri, B.B. A survey of Hough Transform. *Pattern Recognit.* **2015**, *48*, 993–1010. [CrossRef]
20. Sun, H.; Uysalturk, M.C.; Karakaya, M. Invisible data matrix detection with smart phone using geometric correction and Hough transform. *Opt. Pattern Recognit. XXVII Int. Soc. Opt. Photonics* **2016**, *9845*, 98450P. [CrossRef]
21. Chelghoum, R.; Ikhlef, A.; Hameurlaine, A.; Jacquir, S. Transfer learning using convolutional neural network architectures for brain tumor classification from MRI images. In *IFIP International Conference on Artificial Intelligence Applications and Innovations*; Springer: Berlin, Germany, 2020; pp. 189–200.
22. Krizhevsky, A.; Sutskever, I.; Hinton, G.E. Imagenet classification with deep convolutional neural networks. *Adv. Neural Inf. Process. Syst.* **2012**, *25*, 1097–1105. [CrossRef]
23. Szegedy, C.; Liu, W.; Jia, Y.; Sermanet, P.; Reed, S.; Anguelov, D.; Erhan, D.; Vanhoucke, V.; Rabinovich, A. Going deeper with convolutions. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Boston, MA, USA, 7–12 June 2015; pp. 1–9.
24. Simonyan, K.; Zisserman, A. Very deep convolutional networks for large-scale image recognition. *arXiv* **2014**, arXiv:1409.1556

25. He, K.; Zhang, X.; Ren, S.; Sun, J. Deep residual learning for image recognition. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 27–30 June 2016; pp. 770–778.
26. Kuhn, H.W. The Hungarian method for the assignment problem. *Nav. Res. Logist. Q.* **1955**, *2*, 83–97. [CrossRef]
27. Munkres, J. Algorithms for the assignment and transportation problems. *J. Soc. Ind. Appl. Math.* **1957**, *5*, 32–38. [CrossRef]