Chapter 9
Research on FPGA Hardware Acceleration for Real-Time Detection of Mobile Phone Lens Defects

Xidong Wang, Guopeng Wang, Baochang Wang, and Hengtao Wang

Abstract The current machine vision method for mobile phone lens defect detection has the shortcomings of high delay, high power consumption, high cost, and harsh deployment conditions. In this article, optimization strategies such as parameter reordering, dynamic quantization, loop expansion, and block segmentation are adopted to design a software and hardware collaborative processing platform based on FPGA hardware acceleration technology, and real-time detection of mobile phone lens defects is realized. The experimental process is that, firstly, the data set samples of mobile lens defect detection are sent to the YOLOv2 model for iterative training, and the optimal weight is obtained by analyzing the performance curve of the training process. Then, the YOLOv2 model is transplanted to the PYNQ-Z2 development board. Finally, the real-time detection of the validation set samples is accelerated by the integrated neural network acceleration IP of the platform. The experimental results show that the FPGA complete real-time detection of an image takes 2 s, and the total power consumption is 2.953 W, which is 97 times higher than that of CPU. The positioning accuracy of the defect is high, and the detection accuracy is 90.80%. It meets the real-time detection requirements of low cost, low power consumption, and convenient deployment of small mobile terminals.

9.1 Introduction

With the vigorous development of convolutional neural network in the field of target detection and image recognition [1], the target detection model YOLO has become a hotspot in application research with the advantages of fast algorithm implementation, flexibility, and excellent generalization performance [2].

X. Wang (✉) · G. Wang · H. Wang
College of Computer and Information, China Three Gorges University, Yichang 443002, China
e-mail: xdwang@ctgu.edu.cn

B. Wang
College of Science, China Three Gorges University, Yichang 443002, China

G. Liu and F. Cen (eds.), *Advances in Precision Instruments and Optical Engineering*,
Springer Proceedings in Physics 270, https://doi.org/10.1007/978-981-16-7258-3_9

FPGA is a semi-custom circuit, which has programmable input and output units, embedded RAM, dedicated wiring resources, configurable logic blocks. FPGA parallelizes massive data to achieve accelerated processing, and the power consumption is extremely low. Compared with CPU, FPGA has unique advantages in accelerating convolution neural network algorithm [3], which is conducive to deployment to small mobile devices.

At present, most of the domestic mobile phone lens manufacturing factories use artificial visual method to detect lens defects. This method has low efficiency and high false detection rate, which is susceptible to the subjective factors of the test workers, and the labor cost is also high [4], which does not adapt to the current automation trend. In recent years, with the development of machine vision technology, its application in lens defect detection has gradually deepened [5]. Zhu et al. used blackbody as a dark background to improve the contrast of lens defects, and used image analysis method for detection. The detection method takes 5 s per piece [6]. Zhu optimized the point source projection lighting scheme of precision microscope and other instruments to achieve the optimal imaging effect of surface defects, analyzed the shape and size of lens defects, and wrote corresponding image processing algorithms for each defect type to identify. The cost of this method was high [7]. Zhu trained a degenerated YOLO network based on deep learning, and realized the PC-controlled detection system to identify various defects in resin lens samples. The detection power consumption of this method was high [8].

On the basis of the above research on detection methods, this article adopts optimization strategies such as parameter reordering, dynamic quantization, loop expansion and block segmentation to design a software and hardware collaborative processing platform based on FPGA hardware acceleration technology, and real-time detection of mobile phone lens defects is realized. The experimental results show that the platform has low real-time detection delay, low power consumption, low cost, high accuracy, and high defect positioning accuracy, which proves the effectiveness and practicability of the accelerated detection of this platform.

9.2 Material and Methods

9.2.1 Selection of Network Model

The R-CNN series algorithm is a two-stage structure network, and its detection speed cannot meet the real-time requirements. YOLOv1 [9] is not suitable for the defect detection of mobile phone lenses due to the immature prediction frame mechanism and loss calculation, which leads to the low positioning accuracy of the target with small feature size. YOLOv2 [10] is a single-stage structural network proposed by Redmon et al. in 2017. Its core idea is to transform the target detection problem into a classification regression problem [11], and introduce the Anchor mechanism to improve the recall rate in the process of model training and enhance the classification

ability of fine-grained image features. It is suitable for detecting targets with small feature size fast detection speed, which can achieve real-time detection of targets.

Due to the low cost of FPGA development board and the relatively limited computing resources, YOLOv2 with high detection accuracy for small size targets, low real-time detection delay, and lightweight network structure is selected.

9.2.2 YOLOv2 Network Structure

The overall structure of YOLOv2 network is shown in Fig. 9.1.

It can be seen from Fig. 9.1 that YOLOv2 network is composed of CBL, MCN, Route, and Reorg. CBL is the smallest component in the YOLOv2 network structure, which is composed of Conv, BN, and Leaky ReLU. The function of Conv (convolution operation) is to realize feature extraction and obtain feature map. Batch normalized BN layer is a linear transformation of feature image elements to accelerate the training convergence process [12]. The activation function is to make a nonlinear transformation of the feature image elements to enhance the nonlinear fitting ability of the network. Except that the final output layer uses Sigmoid, the rest are Leaky ReLU, as shown in (9.1).

$$f(x) = \begin{cases} x \times 0.1, & x < 0 \\ x, & x \geq 0 \end{cases} \tag{9.1}$$

MCN is a down-sampling component, which is composed of the maximum pooling layer MAX and N CBLs. The function of the maximum pooling layer is to realize the down-sampling of the feature map, reduce its size and keep its functional characteristics unchanged.

Route is the routing layer, which integrates multi-dimensional feature information.

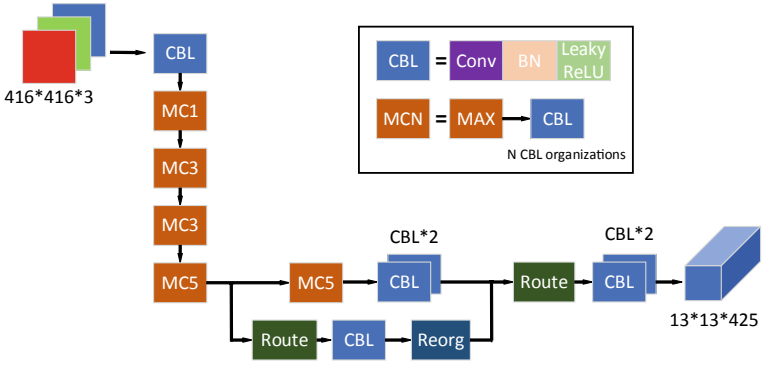


Fig. 9.1 Overall structure of YOLOv2 network

Reorg is a reordering layer that enables reordering of features. The large size feature map is divided into several small size feature maps.

9.2.3 *Accelerator Framework*

In the acceleration scheme, the part with large amount of calculation and redundant calculation structure in the neural network is implanted into FPGA, and the dynamically configurable neural network acceleration IP is designed. The PS terminal stores the structural parameters of the neural network, and the calculation results are obtained by calling the neural network acceleration IP. The overall framework of neural network accelerating IP design is shown in Fig. 9.2.

It can be seen from Fig. 9.2 that the input and output modules are designed in the FPGA, and the convolution calculation, pooling calculation and reorder calculation are encapsulated in the IP module, and the AXI-lite and AXI-full interfaces are encapsulated. The PS terminal configures the calculation unit through AXI-lite, and the weight parameters and sample feature data are introduced into the convolution neural network to accelerate the IP through AXI-full, and then the calculated results are received by AXI-full.

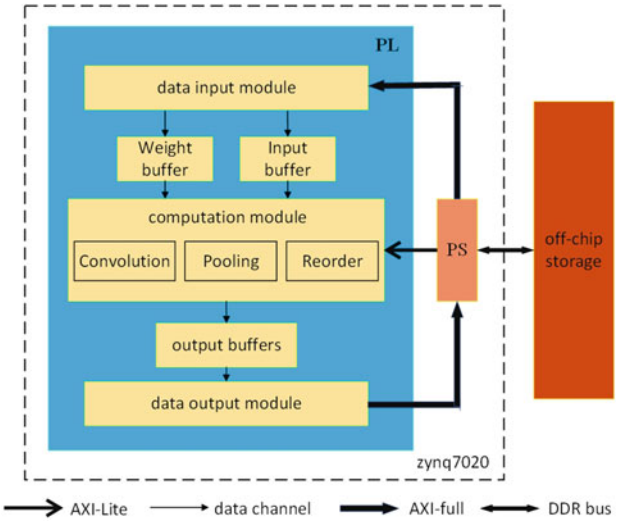


Fig. 9.2 Neural network accelerating IP framework

9.2.4 Integrated Convolutional Neural Network IP

Input and Output Module

The input and output module is used as the main device, and its interface design adopts AXI-full interface, and DDR3 at the PS terminal is used as the slave device. Considering the limited BRAM resources of FPGA, the idea of loop block segmentation is used for expansion, as shown in (9.2) and (9.3).

$$Tci = (Tco - 1) \times S + K \tag{9.2}$$

$$Tri = (Tro - 1) \times S + K \tag{9.3}$$

Among them, Tci and Tri are the size of input feature graph, Tco and Tro are the size of output feature graph, S is the step size of convolution kernel, K is the size of convolution kernel. The process of expanding the input and output modules in depth is shown in Fig. 9.3.

In Fig. 9.3, Tn is the input depth, and Tm is the output depth obtained. The buffer design of double buffers and pipeline operation are adopted to increase the throughput of data and avoid the performance bottleneck for the subsequent convolution acceleration module. The pipelined input and output structure of double buffer is shown in Fig. 9.4.

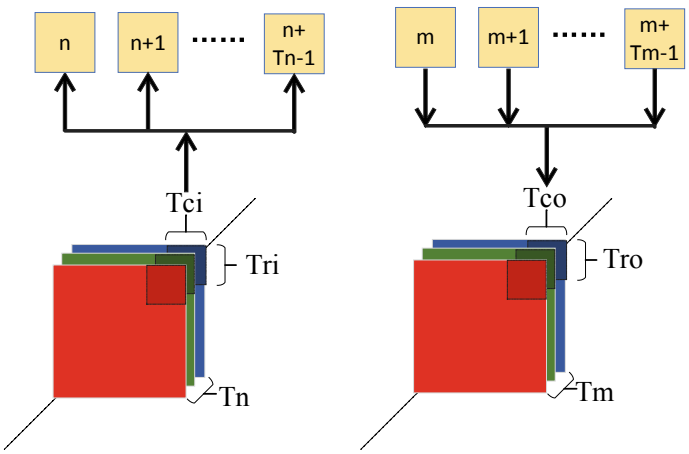


Fig. 9.3 Input and output block segmentation diagram

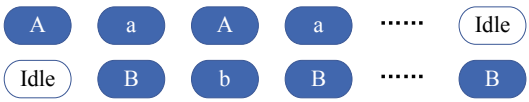


Fig. 9.4 Pipelined input and output structure of double buffer

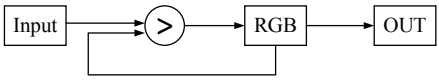


Fig. 9.5 Hardware structure corresponding to maximum pooling

Figure 9.4 for the input structure, A is to pass data from DDR3 to cache Buffer1, a is to pass data from DDR3 to cache Buffer2, B is to pass data from Buffer1 to on-chip cache, and b is to pass data from Buffer2 to on-chip cache. Figure 9.4 for the output structure, A is to pass the data from the on-chip output cache to the cache Buffer1, a is to pass the data from the on-chip output cache to the cache Buffer2, B is to pass the data from Buffer1 to DDR3, and b is to pass the data from Buffer2 to DDR3.

Convolution Module

Convolution calculation occupies the vast majority of the calculation in the neural network, and its essence is multiplication and accumulation. In order to improve the parallelism, the two optimization strategies of expansion and block segmentation are adopted. The input and output feature maps are expanded in two dimensions. Each time, Tm output feature maps and Tn input feature maps are parallelly calculated. The input feature maps are divided into blocks, and only one block is calculated each time. Reduce FPGA requirements for BRAM, reuse computing data, reduce the number of read and write data from off-chip storage.

Maximum Pooling Module

Consider the maximum pooling as a special convolution. It does not require weight parameters, only pooling the input feature map of a channel, and its corresponding operation unit is a comparator. The corresponding hardware structure is shown in Fig. 9.5.

When the 2×2 maximum pooling is used in Fig. 9.5, the REG is initially zero. When the four numbers are input, in turn, the comparator inputs a large number into the REG register. After 2×2 clock cycles, the comparator completes the comparison, outputs the largest number, and clears the REG. Since the resources consumed by comparators are mostly LUT, the pooling design adopts the block segmentation optimization strategy.

Reordering Module

Reordering within FPGA can be seen as a multiplexer, as shown in Fig. 9.6.

In Fig. 9.6, one input buffer corresponds to four output buffers to achieve 2×2 reordering.

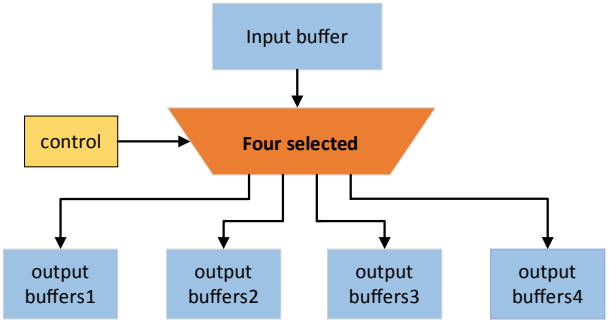


Fig. 9.6 The corresponding structure of reordering

9.2.5 Production and Training of Defect Lens Data Set

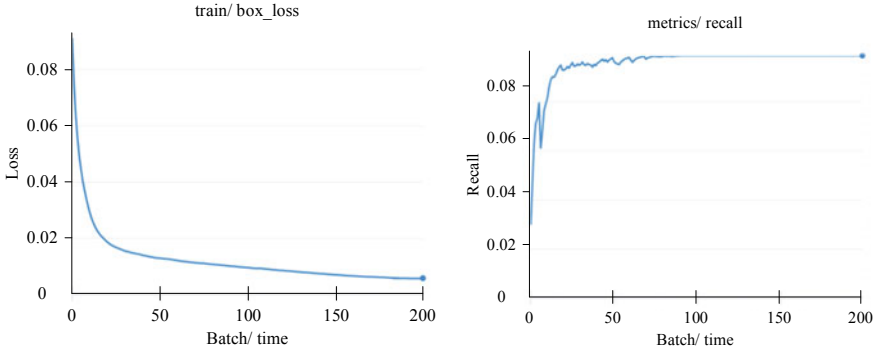
The four common defects (petal injury, glue hole, crack, membrane crack) of mobile phone lens are selected as the research objects to make data sets. The labeling software is used to mark mobile phone lens defect detection dataset. Due to the limited number of actual samples, the number of samples is expanded through data enhancement (rotation transformation, affine transformation, image enhancement, and noise addition). In this way, the purpose of increasing the number of samples is realized, the direction and size of defects are changed, and the robustness of image noise is enhanced [13].

YOLOv2 training network parameters: In order to balance the training effect and the pressure of memory occupation, the number of pictures sent to the network in batches at each iteration is 64, and the number of copies sent to the trainer in batches at each iteration is 8. In order to improve the training speed and avoid overfitting, the momentum constant is 0.9 and the weight attenuation regularization coefficient is 0.0005. With the increase of iterations, in order to make model learning more effective, the initial learning rate is 0.001, and the total number of iterations is 200. The learning rate adjustment strategy is steps. When the number of iterations reaches 160 and 180, the learning rate is reduced to 0.0001 and 0.00001, respectively. The number of categories in region layer is changed to 4, and the number of convolution kernels in the last convolution layer is changed to 45.

In the training process, the change curves of Loss and recall rate of the model are shown in Fig. 9.7a, b, respectively.

It can be seen from Fig. 9.7a that when the training model is iterative to 90 times, the loss function converges to 0.01. It can be seen from Fig. 9.7b that when the training model is iterative to 70 times, the recall rate is close to 1. Based on the analysis of training process performance curve, the weight effect of YOLOv2 network model is ideal. Use right re-results detect the samples of the validation set, and the results are shown in Table 9.1.

Table 9.1 shows that the model detection accuracy is 96.13%, which proves that the training effect of the model is ideal and can be transplanted in the next step.



(a) Loss curve. (b) Recall rate curve.

Fig. 9.7 Change curve of training process

Table 9.1 Test results of model validation set

Class	Number	Accurate number	Accuracy (%)
Petal injury	1152	1092	94.79
Glue hole	960	928	96.67
Crack	192	192	100
Membrane crack	96	95	98.96
Total	2400	2307	96.13

9.2.6 Transplantation of YOLOv2

Rewrite YOLOv2 feature extraction network darknet framework source code, the weight file is separated into two files of weight and offset parameters. Parameter mapping is shown in (9.4).

$$y = A \times X + B \tag{9.4}$$

In the above equation, A and B are mapping coefficients. The weight and offset parameter expressions of convolution kernel are shown in (9.5) and (9.6).

$$\text{weight}_i = \text{weight}_i \times A_i \tag{9.5}$$

$$\text{bias}_i = \text{bias}_i \times A_i + B_i \tag{9.6}$$

In the above equations, where A_i and B_i are the mapping coefficients of characteristic figure i. Store the mapped values in binary form for the separated files, reducing the amount of computation and increasing the computation speed [14].

Quantification of Data

Before quantization, each parameter in the weight and offset data occupies 32 bits, which requires high bandwidth of the transmission bus and indirectly limits the operation rate. In this article, the dynamic fixed-point 16-bit quantization [15] is used to reduce the bit width of the parameters. The optimal order of weight for each layer is searched through traversal, as shown in (9.7).

$$\exp_w = \arg\min \sum_{i=0}^{n} \left| w^i_{float} - w^i_{(bw,\exp_w)} \right| \tag{9.7}$$

In the above equation, $\exp_w$ is the order code of the minimum sum of the absolute value of the loss accuracy of all parameters in this layer after quantization, n is the number of parameters to be quantified in this layer,w^i_{float} is the original floating point value of the first parameter, $w^i_{(bw,\exp_w)}$ is the number of fixed points of the first parameter under the bit width bw and the order code $\exp_w$, and then the number of floating points is converted.

Weight Reordering

When the reordering module is designed, the idea of block segmentation is used to optimize the convolution operation. When the designed algorithm is transformed into RTL circuit by HLS, the simulation results show that the weight parameters are stored separately, resulting in the burst transmission of AXI-full, the size is only K*K, which cannot make full use of the transmission bandwidth of DRAM. In order to make full use of the transmission bandwidth, the weight parameters need to be reordered, as shown in Fig. 9.8.

PS and PL Terminal Collaborative Real-time Detection

The functional design of PS terminal is mainly to store the neural network structure, drive the USB camera, call the integrated neural network to accelerate IP, preprocess the images collected by the camera, and post-process the final output of the integrated

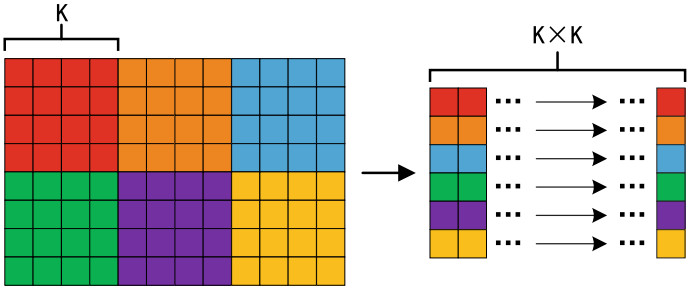


Fig. 9.8 Weight parameter reordering

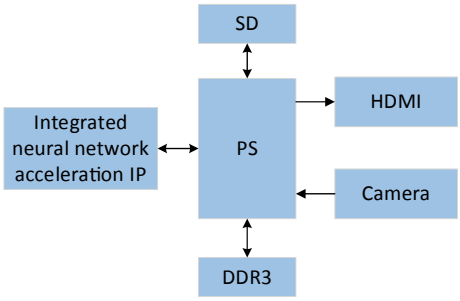


Fig. 9.9 PS and PL terminal synergy framework

neural network to accelerate IP. The coordination design framework is shown in Fig. 9.9.

It can be seen from Fig. 9.9 that the PS terminal first reads the weight and offset files stored in the SD card and puts them to the external memory DDR3. Then, it drives the USB camera to collect real-time video frames, caches the collected images, and then preprocesses them, mainly including the shaping of the collected images and pixel normalization. Finally, it reads the neural network structure, and configures the integrated neural network to accelerate IP according to the input and output layers and order codes.

OpenCV is called to drive the camera, and its built-in function will cache 4 frames of video. In order to solve the problem that continuous acquisition will cause non-real-time video frames, set the program segment to delete the cache frame, which will affect the delay of lens detection.

In order to simplify the detection platform, HDMI is used to display, DIGILENT's RGB2DVI open-source IP core is used to complete data bit conversion, and XILINX's VTC IP core is used to control video timing. In order to solve the problem of video timing, the original data of the video should be input. After the USB camera collects the image data, the data are stored in the external memory DDR3. Since the DDR3 controller is on the PS terminal, the video data need to be transmitted to the PL terminal through the AXI-full interface. In order to improve the video transmission rate and simplify the workload of the PS terminal, the VDMA IP core is called on the PL terminal. The DDR3 data is read by AXI-full and converted into AXI-stream format. The AXIS2VIDEO IP core is called. The data transmitted by the AXI-stream bus is converted into 24-bit RGB data and the video sequence received from the VTC IP core is output to the RGB2DVI IP core. AXIS2VIDEO IP core interacts with VDMA IP core through AXI-stream bus.

When displaying real-time detection results, taking into account the PS terminal's limited computing power, in order to reduce the display time, the ipywidgets library is used. The processed image is displayed directly. If there are defects in the image, it is stored. If there are no defects, it is directly covered to the next frame.

The FPGA hardware accelerator lens defect real-time detection process is shown in Fig. 9.10.

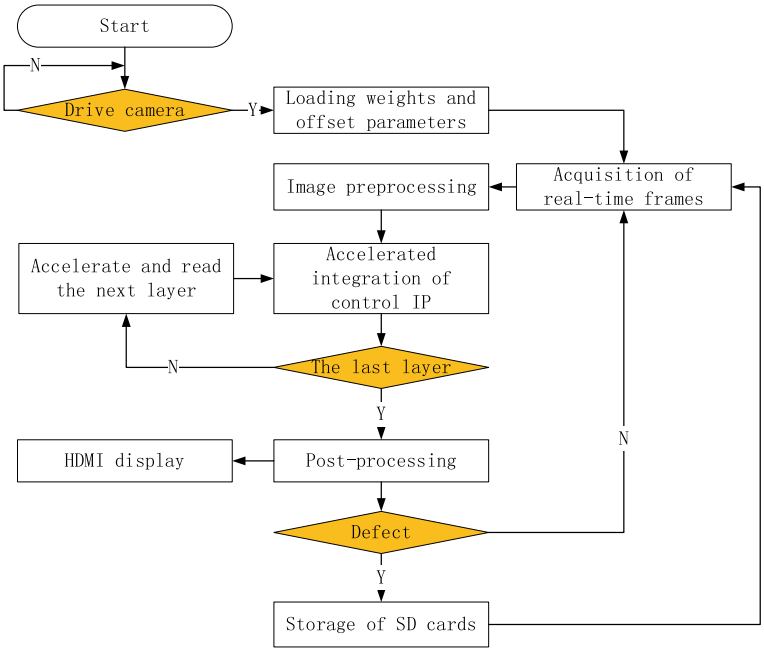


Fig. 9.10 FPGA hardware acceleration lens defect real-time detection

In Fig. 9.10, the convolution layer, the maximum pooling layer, and the reordering layer are accelerated in the integrated neural network acceleration IP according to the division of labor. The routing layer records the address information of input and output. The post-processing is mainly the final detection layer, and the PS terminal is used to realize the positioning and category calibration of the lens defect prediction box.

9.3 Results

9.3.1 Hardware and Software Platform Environment

The hardware and software platform environment used in the experiment is shown in Table 9.2.

This experiment uses TUL's PYNQ-Z2 development board based on zynq7020 chip. The main resource allocation of zynq7020 chip is shown in Table 9.3.

PYNQ-Z2 supports the open-source framework PYNQ, which is simply understood as combining Python with ZYNQ. The library module of Python language can be called under the framework to achieve efficient embedded development and provide full play to the advantages of software and hardware collaboration

Table 9.2 Hardware and software platform configuration

Platform	Configuration
FPGA	PYNQ-Z2(zynq7020)
CPU	Intel(R) Core(TM) i5-10,500 CPU @ 3.10 GHz
Frame	PYNQ
Development tools	Vivado 2020
Operating system	Ubuntu 20.04 LTS
Language	Python 3.9.5

Table 9.3 Zynq7020 partial main resource allocation

Parameter	Configuration
ARM chip	Dual-core ARM Cortex-A9
Maximum frequency	667 MHz
RAM in the SCM	256 KB
Logical unit	85 K
LUT	53.2 K
Trigger	106 K
DSP computing unit	220
BRAM36K	140

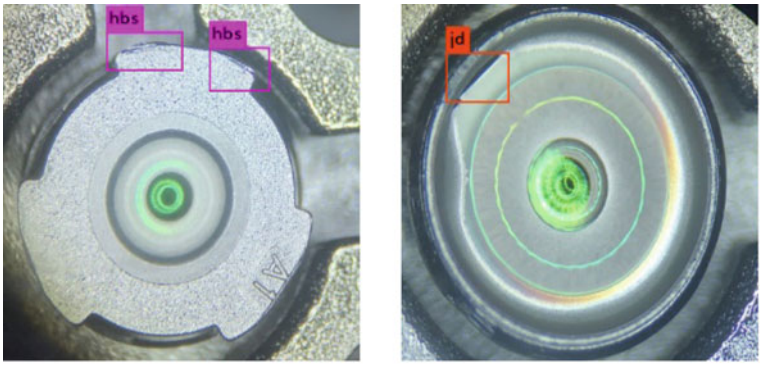
[16]. Although RTL programming cannot be directly carried out in Python environment, the PYNQ open-source framework provides convenience for transplanting the YOLOv2 model.

9.3.2 Real-Time Detection

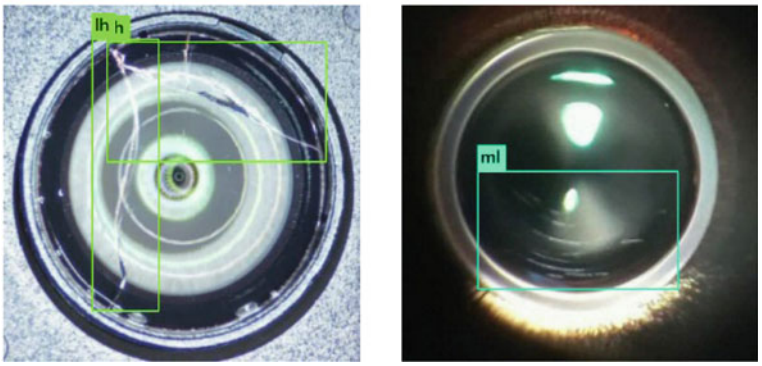
Using the panel display mobile phone lens defect verification set sample for real-time detection, the effect is shown in Fig. 9.11.

Figure 9.11a shows two petal defects in the model correct box, Fig. 9.11b shows one glue hole defect in the model correct box, Fig. 9.11c shows two crack defects in the model correct box, and Fig. 9.11d shows one membrane crack defect in the model correct box. The data above the target box are labels for the prediction category. From the test results, the model can predict and distinguish four kinds of defects. At the same time, in the case of multi-defect coexistence, there is basically no defect missing.

In the process of real-time detection, the adjustment of camera focal length directly affects the image quality of real-time video frames, which indirectly affects the detection results. The real-time detection log is shown in Fig. 9.12.



(a) Petal injury samples. (b) Glue hole samples.



(c) Crack samples. (d) Membrane crack samples.

Fig. 9.11 Real-time detection effect

```
Open pictrue success!
pictrue size: (640, 480)
pictrue mode: RGB
yolov2_image copy ok

FPGA_Accelerate_Completed!!
845
name: jiaodong 65.2170420121
fpga_process_time        : 0.997957706451416
```

Fig. 9.12 Real-time detection log

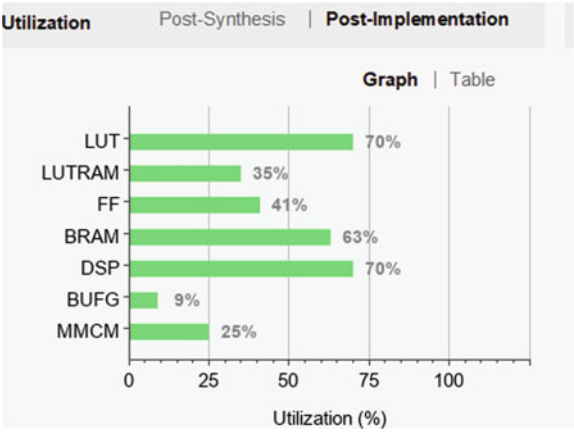


Fig. 9.13 Hardware resource consumption

It can be seen from Fig. 9.12 that the FPGA acceleration process can be completed within 1 s. With the time-consuming of pre-processing, post-processing and PS terminal real-time detection and display results, it can be concluded that it takes 2 s for FPGA to complete the detection of a real-time image by detecting multiple samples and taking the average value respectively.

9.3.3 Hardware Resource Consumption

The hardware resource consumption accelerated by porting the YOLOv2 network to the PYNQ-Z2 development board is shown in Fig. 9.13.

From Fig. 9.13, BRAM and DSP consume the most hardware resources. The reason for BRAM consumption is that the weight parameters and input–output feature maps are stored every time. The reason for DSP consumption is that in order to maximize the acceleration effect of a large number of multiplication and addition operations in the neural network, multiplication and addition operations are performed in parallel.

9.3.4 Detection Power Consumption

The power consumption comprehensive evaluation report of PYNQ-Z2 development board running YOLOv2 model on Vivado platform for lens defect detection is shown in Fig. 9.14.

It can be seen from Fig. 9.14 that the static power consumption is 0.198 W, the dynamic power consumption is 2.775 W, and the total power consumption is 2.953 W. It can be seen that the power consumption of FPGA is extremely low.

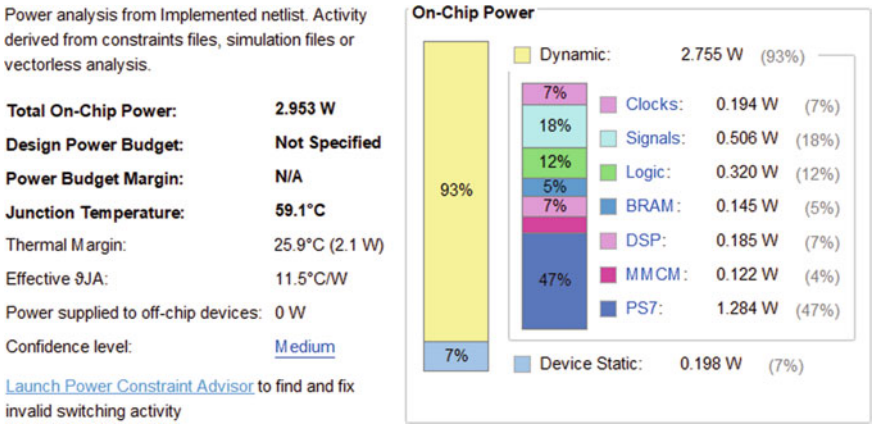


Fig. 9.14 Power consumption synthesis report

Table 9.4 The verification results of lens detection under FPGA

Class	Number	Accurate number	Accuracy (%)
Petal injury	240	208	86.7
Glue hole	200	187	93.5
Crack	40	40	100.0
Membrane crack	20	19	95.0
Total	500	454	90.8

9.3.5 Detection Accuracy

The focal length is adjusted. After the imaging is clear and stable, 500 samples of validation set are detected in real time. The statistical verification results are shown in Table 9.4.

The analysis of Table 9.4 shows that the accuracy of FPGA real-time detection is 90.80%, and the lowest accuracy is petal injury. Compared with other defects, petal injury has smaller feature size and smaller difference between feature shape and background, which leads to relatively simple feature extraction and low detection rate.

9.4 Discussion

The description of CPU detection environment in YOLO official network is shown in Fig. 9.15.

From Fig. 9.15, YOLO official CPU environment detection of an image takes 6–12 s.

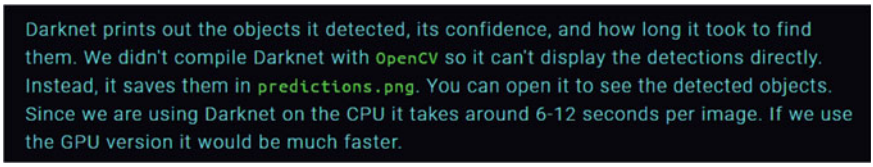


Fig. 9.15 YOLO official description

Table 9.5 Comparison of FPGA and CPU detection performance

Platform	Time-consuming (s)	Power loss (W)	Accuracy (%)
PYNQ-Z2(zynq7020)	2	2.953 W	90.80
Intel Core i5-10,500 CPU @ 3.10 GHz	7	81 W	96.13

FPGA versus CPU detection performance is shown in Table 9.5.

The analysis of Table 9.5 shows that the FPGA hardware acceleration lens defect real-time detection acceleration rate is 350% of the CPU environment, FPGA acceleration effect is significant. FPGA has excellent advantages in power consumption. For long-running equipment or instruments, it is more energy-saving to choose FPGA. The detection accuracy of FPGA is slightly lower than that of CPU environment. The main reason is that the weight and offset parameters are quantified by dynamic fixed-point 16 bits, and the decrease of bit width leads to the loss of accuracy and the decrease of accuracy.

Table 9.6 compares the defect detection performance of this platform with the other three methods in references.

It can be seen from Table 9.6 that the platform in this article is superior to the other three machine vision methods in terms of detection delay, power consumption, cost and application scenarios, and meets the requirements of real-time detection of lens defects with low cost and low power consumption.

Table 9.6 Method performance comparison

Method	Time-consuming	Power loss	Cost	Application scene
[6]	5 s/piece	High	High	Real-time
[7]	High	High	High	Static state
[8]	High	High	High	Static state
This article	2 s/piece	Lower	Lower	Real-time

9.5 Conclusions

In order to solve the problems of high delay, high power consumption, high cost and harsh deployment conditions in current machine vision method for mobile phone lens defect detection, and meet the requirements of low delay, low power consumption, high accuracy and strong stability in real-time application scenarios of small mobile devices. In this article, optimization strategies such as parameter reordering, dynamic quantization, loop expansion, and block segmentation are adopted to design a soft and hard collaborative processing platform based on FPGA hardware acceleration technology, and real-time detection of lens defects is realized. The experimental results show that the FPGA complete real-time detection of an image takes 2 s, which is 3.5 times faster than the CPU performance, and the total power consumption is 2.953 W, which is equivalent to 3.6% of the CPU, and the performance power consumption ratio is increased by 97 times. The defect location accuracy of the detection method is high, and the detection accuracy is 90.80%, which proves that the detection platform has certain advantages in performance.

Limited by time and energy, this platform has the following limitations. Firstly, the current FPGA detection accuracy is 5.33% lower than that of CPU. Subsequently, the dynamic fixed-point 16-bit quantized model can be iteratively optimized by multiple trainings to reduce the loss of detection accuracy and control it within an acceptable range. Secondly, the post-processing flow in the real-time detection process of the platform is not accelerated by FPGA hardware, and the time consumption is about 0.5 s. The post-processing flow can be accelerated in FPGA, which can reduce the real-time detection delay to a certain extent.

References

1. O. Russakovsky, J. Deng, H. Su et al., ImageNet large scale visual recognition challenge. Int. J. Comput. Vision **115**(3), 211–252 (2015)
2. B. Wang, H.X. Le, W.J. Li, M.H. Zhang, Oil tank detection algorithm on remote sensing image using multi-scale parallel convolutional neural networks. Comput. Eng. Appl. **57**(08), 62–69 (2021)
3. X.F. Wang, P.L. Jiang, H. Zhou, X.B. Zhao, Design of FPGA accelerator with high parallelism for convolution neural network. Comput. Appl. **41**(03), 812–819 (2021)
4. L. Sun, C. Liu, H.B. Yao, Automatic detection of water-mask for resin glasses by machine vision. J. Jiangsu Univ. **39**(04), 425–430 (2018)
5. L. Lin, T.G. Zang, Edge detection of defect for resin lenses based on machine vision. Mach. Build. Automat. **50**(02), 230–232+240 (2021)
6. Y.D. Zhu, Y.X. Chen, Visual inspection method of detecting flaws on optical glasses surface. Appl. Opt. **41**(03), 553–558 (2020)
7. J.F. Zhu, Research on surface defect detection of resin lens based on machine vision technology. Jiangsu Univ. **2019**, 1–75 (2019)
8. H.Z. Zhu, The research on resin lens defect recognition method based on degenerate YOLO network. Harbin Inst. Technol. **2019**, 1–66 (2019)
9. J. Redmon, S. Divvala, R. Girshick et al., You only look once: unified, real-time object detection, in *Computer Vision & Pattern Recognition* (Las Vegas, 2016), pp. 779–788

10. J. Redmon, A. Farhadi, YOLO9000: better, faster, stronger, in *Computer Vision & Pattern Recognition* (Hawaii, 2017), pp. 6517–6525.
11. S.S. Xin, W.L. Zhao, Overview of UAV target detection in complex scenes based on Yolo series algorithms. Appl. Res. Comput. **37**(S2), 28–30 (2020)
12. J.W. Liu, H.D. Zhao, X.L. Luo, L. Xu, Research progress on batch normalization of deep learning and its related algorithms. Acta Automatica Sinica. **46**(06), 1090–1120 (2020)
13. B. Yu, A data augmentation method for image class imbalance problem using generative adversarial networks. South China Univ. Technol. **2018**, 1–73 (2018)
14. Y. Cui, G.H. Shu, D. Li, Research on binary neural network classification models based on the PYNQ development board. Electr. Automat. **041**(005), 53–56 (2019)
15. S. Lei, M. Zhang, D. Lin et al., A dynamic multi-precision fixed-point data quantization strategy for convolutional neural network, in *CCF National Conference on Computer Engineering and Technology* (Singapore, 2016), pp. 978–981
16. H.J. Liu, Y.J. Yue, Design of binocular vision ranging system based on pynq platform. Electron. World. **16**, 184–186 (2020)