Attention Mechanism Driven YOLOv3 on FPGA Acceleration for Efficient Vision Based Defect Inspection

Longzhen Yu
College of Economics and Management, Qingdao University of Science and Technology, Qingdao Shandong, China
yulongzhen@qust.edu.cn

Qian Zhao*
Department of Creative Informatics, Kyushu Institute of Technology, Fukuoka Japan
cho@ai.kyutech.ac.jp

Zhixian Wang
College of Economics and Management, Qingdao University of Science and Technology, Qingdao Shandong China
wangzhixian62@126.com

ABSTRACT

In this study, an efficient vision-based industry defect inspection system using attention mechanism driven YOLOv3 on FPGA acceleration is proposed. First, an attention mechanism is employed to improve YOLOv3 for the target defect inspection application. Image preprocessing named CZS (Cut, Zoom, and Splice) operation is used to reconstruct product images for selectively concentrating on the pre-defined detection regions. Then we optimize the backbone network of YOLOv3 according to defect size in images. Second, we use the PYNQ-Z2 FPGA board to deploy the proposed defect inspection system. The optimized YOLOv3 is deployed on the programmable logic through Xilinx DNNDK, which is a low-latency, low-cost, and low-power consumption hardware platform for industrial defect inspection. Experimental results showed that the achieved defect inspection accuracy was 99.2% with a processing speed of 1.54 FPS.

CCS CONCEPTS

• **Computing methodologies**; • **Artificial intelligence**; • **Computer vision**; • **Computer vision problems**; • **Object detection**;

KEYWORDS

Defect inspection, YOLO, FPGA

ACM Reference Format:
Longzhen Yu, Qian Zhao*, and Zhixian Wang. 2021. Attention Mechanism Driven YOLOv3 on FPGA Acceleration for Efficient Vision Based Defect Inspection. In *The 5th International Conference on Computer Science and Application Engineering (CSAE 2021), October 19–21, 2021, Sanya, China.* ACM, New York, NY, USA, 5 pages. https://doi.org/10.1145/3487075.3487165

1 INTRODUCTION

Vision-based product defect inspection is essential to ensure the quality of parts and the whole manufacturing process. In recent years, with the emergence of deep learning-based high-accuracy image recognition technology, conventional manual operations in vision-based defect inspection process can be replaced by automation to achieve better quality control and low cost. There are two types of deep learning techniques that can be used for vision-based defect inspection, the classification-based models and the regression-based models [1]. The classification based models are represented by R-CNN series. Generally, R-CNN (Region-based Convolutional Neural Network) needs two-stage processing for region extraction and object classification, which requires a larger amount of operations, so that the detection speed is lower. Redmon et al. [2] proposed a regression-based end-to-end object detection, namely YOLO (You Only Look Once). YOLOv3 is one of the most widely used YOLO models. Based on the YOLOv3 model, Jing et al. [3], Du et al. [4] etc. have proposed surface defect inspection methods for fabric, PCB board, pavement, etc.

Vision-based product defect inspection differs from the other object detection problems in two special aspects, which provide us with optimization opportunities for such applications. First, the recognition region on an image is predictable. As the example of a normal welding image and a defect welding image shown in Figure 1, the image capture angle of a specific product is relatively fixed, so the actual focus of the image to be processed is limited in a predefined region (e.g. the red box in the figure). Second, the system will be widely deployed at the production site. The processing accuracy, speed performance, power consumption, stability, and scalability must be considered. The target application requirements for this work are as follows. The defect inspection accuracy and image processing speed should not be less than 96% and 1 FPS, respectively. And the power consumption of single equipment should not exceed 100W.

In this work, an efficient defect inspection system using FPGA accelerated YOLOv3 is proposed to meet the above requirements. The main contributions are as follows.

We propose an attention mechanism to optimize YOLOv3 for defect inspection applications. A CZS image preprocessing method and YOLOv3 backbone network tailoring method are proposed according to the defect target character.

We use the TUL PYNQ-Z2 FPGA board to deploy the proposed defect inspection system. The optimized YOLOv3 is deployed on the programmable logic through Xilinx DNNDK, while the host program for CZS and database operations is running on the processing system, which is a low-latency, low-cost, and low-power consumption hardware platform for industrial defect inspection.

The rest of this paper is organized as follows. The YOLOv3 with the proposed attention mechanism for defect inspection is explained in section 2. In section 3, we describe FPGA implementation of the proposed system. Experimental results are shown in section 4. The last gives conclusions.

CSAE 2021, October 19–21, 2021, Sanya, China

ACM ISBN 978-1-4503-8985-3/21/10…$15.00
https://doi.org/10.1145/3487075.3487165

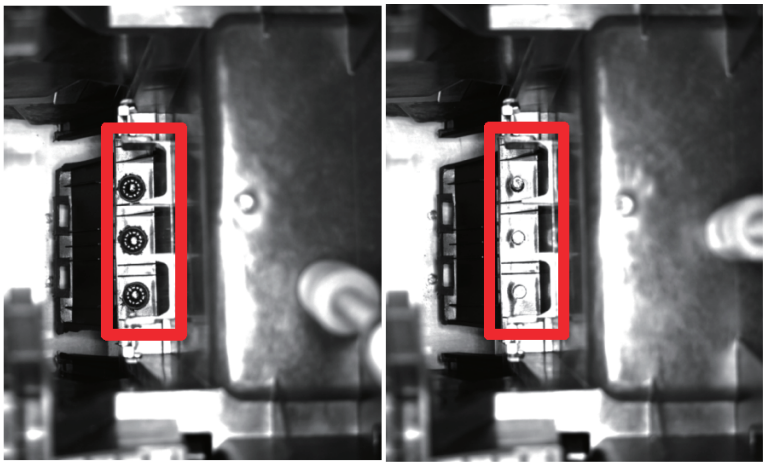

Figure 1: Normal Welding and Defect Welding.

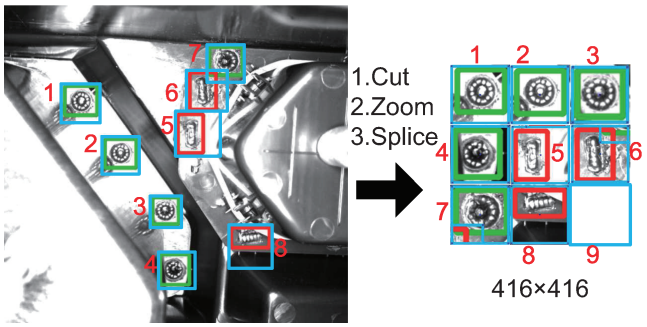

Figure 2: Proposed Image Preprocessing for Attention Enhancement.

2 YOLOV3 WITH ATTENTION MECHANISM

2.1 CZS Operation

Attention mechanism allows modeling of dependencies without regard to their distance in the input or output sequences [5]. For defect inspection, the image capture angle of a specific product is relatively fixed. Therefore, we can pay attention to the predefined region for inspection, and extraction features only from such regions. Attention mechanism focuses on the key information of input and ignores irrelevant information, whis is achieved by image preprocessing. Image preprocessing is as shown in Figure 2. The whole process consists of three steps, cut, zoom and splice. We name these operations CZS operation. The blue boxes represent the cutting regions, green boxes and red boxes represent two kinds of defect markup regions, respectively. Region numbers 1 to 8 of the left original image corresponding to the splice regions of the right image.

The cut operation takes the smallest square containing the defect markup region as the clipping region. Assuming that the width and height of the original image are w_s and h_s respectively, the proportions of the width, height, x-coordinate and y-coordinate of the center point of the defect markup region to the whole image are w_{mr}, h_{mr}, x_{mr} and y_{mr} respectively. Then the values of width, height, x-coordinate and y-coordinate of the center point of the defect markup region are $w_m = w_s \times w_{mr}$, $h_m = h_s \times h_{mr}$, $x_m = w_s \times x_{mr}$ and $y_m = h_s \times y_{mr}$; The width, height, x-coordinate and y-coordinate of the upper left corner of the clipping region are w_c, h_c, x_c and y_c respectively.

$$w_c = h_c = max(w_m, h_m) \times \alpha \tag{1}$$

$$x_c = (x_m + \frac{w_c}{2} \leq w_s)?x_m - \frac{w_c}{2} : w_s - w_c \tag{2}$$

$$y_c = (y_m + \frac{h_c}{2} \leq h_s)?y_m - \frac{h_c}{2} : h_s - h_c \tag{3}$$

The α represents an expansion coefficient that taking a value between 1 and 2, meaning that the area of the clipping region is 1 to 2 times of the defect markup region. In order to ensure that the clipping region can contain the defect markup region, the clipping region should be slightly larger than the defect markup region. The ternary operator (*Logical expression*) ?*True operation: False operation* in formulas (2) and (3) means that when the defect markup region is close to the boundary of the picture, the upper left corner of the clipping region should be extended correspondingly.

The zoom operation is to scale all the clipping regions on an image to the same size, so that can be fully accommodated by the new 416×416 image defined as $W = 416$ and $H = 416$, which is a YOLOv3 standard. If the number of defect markup regions on an image is known to be N_l, the number of clipping regions that can be accommodated in a row or column of a new image is calculated according to formula (4). The target size that the clipping region should be scaled is calculated according to formula (5). The scaling factor β is calculated according to formula (6).

$$N_{row} = N_{col} = ceil(sqrt(N_l)) \tag{4}$$

$$w_z = h_z = floor(\frac{W}{N_{row}}) \tag{5}$$

$$\beta = \frac{w_z}{w_c} \tag{6}$$

The splice operation is to combine several clipping regions after the scaling operation into a picture. It mainly involves two algorithms, one is mapping clipping regions to splice regions; the other is mapping markup regions to splice regions. For algorithm 1, we first arrange the clipping regions from small to large according to the x_c value. If the x_c values of the two clipping regions are the same, then we arrange them from small to large according to y_c value. Assuming that a clipping region is sorted as N_o, the width, height, x-coordinate and y-coordinate of the upper left corner of the region in the new picture are taken as w_d, h_d, x_d and y_d. // represents the integer operation and % represents the remainder operation.

$$n_{row} = (N_o\%N_{col} \neq 0)?N_o//N_{col} + 1 : N_o//N_{col} \tag{7}$$

$$n_{col} = (N_o\%N_{col} \neq 0)?N_o\%N_{col} : N_{col} \tag{8}$$

$$w_d = h_d = w_z \tag{9}$$

$$x_d = (n_{col} - 1) \times w_z \tag{10}$$

$$y_d = (n_{row} - 1) \times h_z \tag{11}$$

For algorithm 2, the size ratio of the width, height, x-coordinate and y-coordinate of the center point of the defect markup region to the new picture is w_{nr}, h_{nr}, x_{nr} and y_{nr}.

$$w_{nr} = \frac{w_m \times \beta}{W} \tag{12}$$

$$h_{nr} = \frac{h_m \times \beta}{H} \tag{13}$$

$$x_{nr} = \frac{(n_{col} - 1) \times w_z + (x_m - x_c) \times \beta}{W} \tag{14}$$

$$y_{nr} = \frac{(n_{row} - 1) \times h_z + (y_m - y_c) \times \beta}{H} \tag{15}$$

The new image generated may have some blank regions, which are filled with 0 to complete the image preprocessing.

2.2 YOLOv3 Backbone Network Tailoring

In response to CZS operation, the backbone network of YOLOv3 can also be optimized to detect the defect regions more targeted.

The YOLOv3 backbone network includes 53 layers, called Darknet-53. The final feature map of YOLOv3 has three sizes, 13×13 has better support for big object recognition, while 52×52 have better support for small object recognition and 26×26 for medium size recognition.

As for the case vision-based defect inspection, the backbone network can be tailored according to the scale of the defect markup region. Taking the defect inspection of automobile rubber and plastic parts as an example, the small-scale network can be simplified due to the moderate size and obvious characteristics of rubber and plastic parts in the inspection photos. Taking PCB solder joint missing detection as another example, the large-scale network can be simplified for that the solder joint layout is fine. To sum up, the specific judgment formula is as follows.

$$tailor(yolo_{52\times52}), if(\underset{i\in N}{each}(w_i > \frac{W}{26} \cap h_i > \frac{H}{26})) \tag{16}$$

$$tailor(yolo_{13\times13}), if(\underset{i\in N}{each}(w_i < \frac{W}{26} \cap h_i < \frac{H}{26})) \tag{17}$$

The *tailor* function is used to simplify the network $yolo52 \times 52$. represents the small-scale detection network, $yolo13 \times 13$ represents the large-scale detection network, *each* function is used to take each detection markup region, and N represents the total number of detection markup regions on an image. w_i and h_i represent the width and height of the i-th detection markup region respectively, and W and H represent the width and height of the image respectively.

In the following case study, the 52×52 small scale feature recognition network is simplified. Specifically, the improvement of YOLOv3 is to simplify the third down-sampling in the backbone network, that is the first eight repeats part. Seven repeats of which are deleted, so the backbone network is reduced from 53 convolution layers to 39 convolution layers.

3 FPGA IMPLEMENTATION

3.1 Overall Framework

The proposed vision-based defect inspection system is accelerated and deployed with MPSoC (Multi-processor system-on-a-chip) FPGA, which allows using low-power and customizable FPGA hardware to replace high power-consumption general-purpose deep learning workstations. Figure 3 shows the overall framework. In general, the whole system includes a main host program running on processing system (PS) and a deep learning processing unit (DPU) implemented on programmable logic (PL). The general processing flow is described as follows. The main program captures product image from the camera, performs CZS operation according to the inspection region information in database, and then sends preprocessed image to DPU for defect recognition.

All these functions are deployed on the TUL PYNQ-Z2 board, which equipped with a Xilinx ZYNQ 7020 MPSoC FPGA. The PYNQZ2 supports python programming on PS and supports DNNDK (Deep Neural Network Development Kit) framework to deploy DPU on PL for deep learning model acceleration. The power consumption of PYNQ-Z2 is less than 10w, which has much better energy efficiency than general-purpose CPU and GPU.

3.2 Deployment

We employs the B1152 DPU architecture Xilinx DNNDK 3.0 framework. It can efficiently process YOLOv3 on 416×416 images with about 3.5 FPS, which can fully meet our mentioned speed performance target.

We perform network training, pruning, quantization and compiling on a workstation computer. The training dataset is prepared using actual product images from production site, we manually labeled types of defects. We use PyTorch to build and train the optimized YOLOv3 model, and then export it to TensorFlow with ONNX to bridge the PyTorch and DNNDK flow. Next, DNNDK performs model optimization and compilation for FPGA deployment. It performs pruning is to get a new network from the trained network, replace the variable nodes with the constant values of the current session, and delete redundant network branches that do not affect the output. Then data quantization is performed to convert floating-point numbers into fixed-point numbers. The first reason for quantization is to use short data types instead of long data types, for example, to replace 32-bit floating-point types with 8-bit integers, so that the whole network can achieve the purpose of compression and reduce storage space; the second is that the DSP units of FPGA are fixed-point processing units, which are good

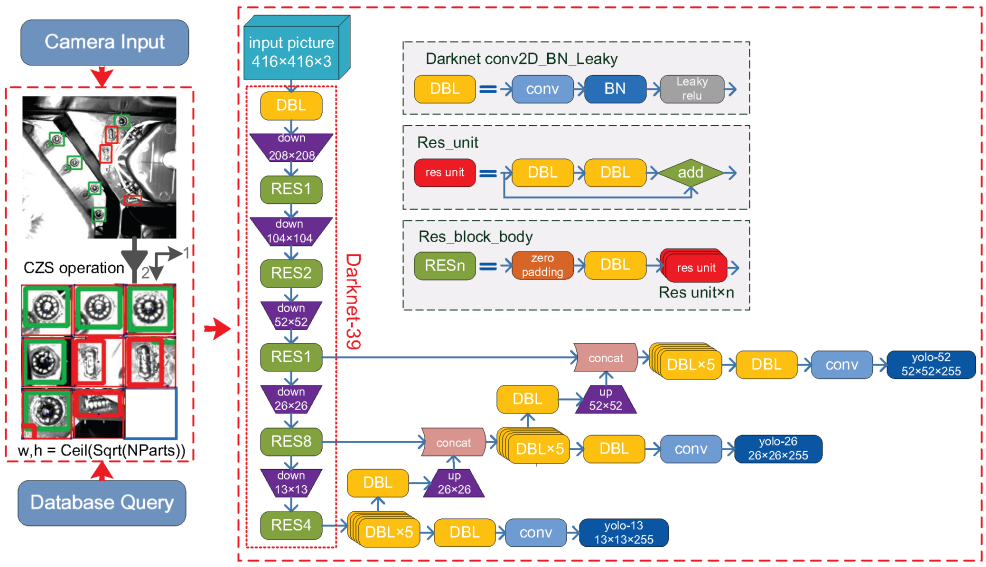

Figure 3: Overall Framework of the Proposed Defect Inspection System.

at fixed-point number operation. At last, the compiler transforms neural network model into binary instructions recognized by DPU, which is the combination of interpreter, optimizer and code generator. The interpreter is responsible for parsing the model and converting it into IR (Intermediate Representation); the optimizer optimizes IR; and the code generator maps the optimized IR to DPU instructions. After compiling, a file containing the optimized YOLOv3 model structure recognized by DPU is generated. Deploy this file to PYNQ-Z2 and load it in the main host program, the deployment on FPGA is completed.

4 EXPERIMENTAL RESULTS

4.1 Experiment Design

We collected experimental data from a representative automobile parts manufacturer. A total of 630 photos of the original samples were collected by 10 industrial cameras with fixed angles. They were 63 photos of the same type of rubber and plastic parts, all 1600×1200 pixels and 8-bit depth. There are 16 kinds of defects markups. The ratio of good and defect products in the original photos is about 9:1. The workstation computer used in the experiment consists of AMD R9 3900X CPU, 64GB DDR4 memory, NVIDIA RTX3090 GPU. The FPGA for deployment was a TUL PYNQ-Z2 board.

We performed data augmentation on the original photos, and achieved the training set composed of 36,139 photos, and the training epoch was set to 300. On the workstation computer side, the trained optimized YOLOv3 model shows an accuracy of 99.2%. And the inspection time is about 0.01 second. As shown in Table 1, the inspection time of our model is shortened by 0.004s, and the inspection accuracy is improved by 4.2%. Compared with GIoU, our model has better convergence performance. Compared with other indicators, our model basically coincides with YOLOv3, indicating that our model has no loss in performance due to tailoring the backbone network.

Table 1: Comparison between Our Model and YOLOv3

	Training	Inspection	Accuracy
YOLOv3	28 FPS	0.014 s	95.0%
Our Model	28 FPS	0.010 s	99.2%

Table 2: Processing Time Comparison between Workstation Computer and PYNQ-Z2

	Preprocess	1600 × 1200	416 × 416
Host	0.24 s	0.01 s	0.01 s
PYNQ-Z2	0.31 s	1.20 s	0.65 s

4.2 Deployment Result

The experimental results on PYNQ-Z2 deployment also showed high accuracy. The processing time comparison between workstation computer and PYNQ-Z2 are shown in Table 2. Through the artificial random validation with 100 photos, the DPU inference results presents the same inference results as the workstation computer side. In addition, the MPSoC shows a reasonable processing speed with much lower power consumption than a workstation computer. The performance on PYQN-Z2 can meet our design target mentioned in Section 1.

As the processing time comparison of different models on PYNQZ2 shown in Table 3, CZS operation and tailoring YOLOv3 can both reduce the processing time. Table 4 shows the performance comparison with other representative industry defect inspection studies. Nico et al. [6] used image segmentation based model. Ting et al. [7] and Chunyang et al. [8] used deep learning based model but YOLO. Junfeng et al. [3] also used classical YOLOv3. Compared with these studies, our approach has achieved better accuracy on

Table 3: Processing Time Comparison of Different Models on PYNQ-Z2

	SetImage	RunTask	Deal
No preprocess, YOLOv3	0.71 s	0.45 s	1.20 s
Preprocess, YOLOv3	0.13 s	0.45 s	0.73 s
Preprocess, our model	0.13 s	0.38 s	0.65 s

Table 4: Performance Comparison with Other Representative Industry Defect Inspection Studies

	Accuracy	1/FPS	mAP_{bbox}
Nico Prappacher [6]	98%	0.153	-
Ting He [7]	98.7%	0.007	-
Chunyang Xia [8]	98.4%	-	-
Junfeng Jing [3]	98%	0.046	-
Our model	99.2%	0.010	0.991

industry defect inspection with comparatively shorter inspection times (on workstation computer). In addition, our approach can be promoted in the enterprise level application to complete the edge detection work in a cheap, stable and green way.

5 CONCLUSION

In this paper, we propose a novel efficient vision-based defect inspection YOLOv3 model improved with attention mechanism and evaluated its FPGA implementation. The experimental result shows that the accuracy reached 99.2%, the processing time of each image is less than 1 second, and the power consumption is lower than 10w. The proposed framework is expected to be widely deployed in industrial field as a low-cost defect inspection solution.

REFERENCES

[1] Li Liu,Wanli Ouyang, XiaogangWang, Paul Fieguth, Jie Chen, Xinwang Liu, and Matti Pietikinen(2020). Deep Learning for Generic Object Detection: A Survey. International Journal of Computer Vision 128, 2 (1), 261–318.

[2] Joseph Redmon and Ali Farhadi (2018). YOLOv3: An Incremental Improvement. arXiv.org (April 2018), 1–6. arXiv:1804.02767v1 [cs.CV] http://arxiv.org/abs/1804.02767v1

[3] Junfeng Jing, Dong Zhuo, Huanhuan Zhang, Yong Liang, and Min Zheng (2020). Fabric defect detection using the improved YOLOv3 model. Journal of engineered fibers and fabrics 15, 1 (1), 155892502090826.

[4] Yuchuan Du, Ning Pan, Zihao Xu, Fuwen Deng, and Hua Kang (2020). Pavement distress detection and classification based on YOLO network. International Journal of Pavement Engineering (1), 1–14.

[5] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin (2017). Attention Is All You Need. arXiv:1706.03762 [cs.CL]

[6] Prappacher Nico, Bullmann Markus, Bohn Gunther, Deinzer Frank, and Linke Andreas (2020). Defect Detection on Rolling Element Surface Scans Using Neural Image Segmentation. Applied Sciences 10, 9 (1). https://doi.org/10.3390/app10093290

[7] Ting He, Ying Liu, Yabin Yu, Qian Zhao, and Zhongkang Hu (2019). Application of Deep Convolutional Neural Network on Feature Extraction and Detection of Wood Defects. Measurement 152 (1), 107357.

[8] Chunyang Xia, Zengxi Pan, Zhenyu Fei, Shiyu Zhang, and Huijun Li (2020). Vision based defects detection for Keyhole TIG welding using deep learning with visual explanation. Journal of Manufacturing Processes 56 (1), 845–855.