

Article

Mask-Guided Generation Method for Industrial Defect Images with Non-uniform Structures

Jing Wei ^{1,2} , Zhengtao Zhang ^{1,2,3} , Fei Shen ^{1,2,3} and Chengkan Lv ^{1,2,3,*} ¹ Institute of Automation, Chinese Academy of Sciences, Beijing 100190, China² The School of Artificial Intelligence, University of Chinese Academy of Sciences, Beijing 100049, China³ CASI Vision Technology Co., Ltd., Luoyang 471000, China

* Correspondence: chengkan.lv@ia.ac.cn

Abstract: Defect generation is a crucial method for solving data problems in industrial defect detection. However, the current defect generation methods suffer from the problems of background information loss, insufficient consideration of complex defects, and lack of accurate annotations, which limits their application in defect segmentation tasks. To tackle these problems, we proposed a mask-guided background-preserving defect generation method, MDGAN (mask-guided defect generation adversarial networks). First, to preserve the normal background and provide accurate annotations for the generated defect samples, we proposed a background replacement module (BRM), to add real background information to the generator and guide the generator to only focus on the generation of defect content in specified regions. Second, to guarantee the quality of the generated complex texture defects, we proposed a double discrimination module (DDM), to assist the discriminator in measuring the realism of the input image and distinguishing whether or not the defects were distributed at specified locations. The experimental results on metal, fabric, and plastic products showed that MDGAN could generate diversified and high-quality defect samples, demonstrating an improvement in detection over the traditional augmented samples. In addition, MDGAN can transfer defects between datasets with similar defect contents, thus achieving zero-shot defect detection.

Keywords: industrial manufacturing; deep learning; data augmentation; defect generation; defect detection



Citation: Wei, J.; Zhang, Z.; Shen, F.; Lv, C. Mask-Guided Generation Method for Industrial Defect Images with Non-uniform Structures. *Machines* **2022**, *10*, 1239. <https://doi.org/10.3390/machines10121239>

Academic Editor: Dimitrios Manolakos

Received: 6 November 2022

Accepted: 15 December 2022

Published: 18 December 2022

Publisher's Note: MDPI stays neutral with regard to jurisdictional claims in published maps and institutional affiliations.



Copyright: © 2022 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

1. Introduction

Methods based on machine learning and deep learning have remarkably improved industrial defect detection performance [1–3]. However, practical industrial scenarios pose challenges to the current detection methods, such as data problems. Acquiring large datasets for manufacturing applications remains a challenging proposition, due to the time and costs involved [4]. The small number of defect samples and data imbalances can lead to overfitting during the training of supervised deep-learning methods and poor performance in testing [5].

Data augmentation is a very powerful method for building useful deep-learning models and for reducing validation errors [6]. Image data augmentation mainly includes traditional and learning-based methods. Traditional methods can increase the number of samples, but cannot create new defect samples. In contrast, learning-based methods such as GAN (Generative Adversarial Nets) [7], AAE (Adversarial AutoEncoders) [8], and VAE (Variational Auto-Encoder) [9] can model the distribution of a real dataset and synthesize new samples that are different from the original dataset, which increases both the number and the diversity of the dataset. Based on cutting-edge work in image synthesis [10–13], industrial image generation can be carried out, to achieve the augmentation of few-sample datasets. Currently, diverse learning-based augmentation methods are emerging, to alleviate data problems in industrial defect detection [14–20].

However, there are still some challenges that need to be addressed in the current learning-based defect image augmentation methods.

Insufficient retention of realistic background textures. Textures provide important and unique information for intelligent visual detection and identification systems [5]. In industrial defect detection, any slight change to real textures can disturb the detection results. Researchers usually perform defect image generation based on non-defective samples, due to their easy availability in industrial manufacturing. This requires generation methods that preserve the real normal backgrounds to the maximum extent possible. Many works [15–17,19] have used CycleGAN [21] to generate a defective image for an input normal image, where normal backgrounds may be excessively falsified, since they do not constrain the treatment of the normal background.

The independent controls of the normal backgrounds, defect shapes, and defect textures are rarely considered. If independent control and operation of the three are achieved, then they can be arbitrarily combined to obtain an infinite number of defective images from a normal image. However, current effective methods such as SDGAN [15], Defect-GAN [16], and SIGAN [17] control the three as a whole and can only obtain one defect image from a normal image based on well-trained models, whose randomness and diversity are insufficient. Moreover, the pixel-level annotations of the generated defect images can be acquired if we separately process the normal backgrounds and defect regions. Defect-GAN generates a spatial distribution map, to indicate what is modified in the source image compared with the generated image, but it does not decouple the backgrounds and defect regions and cannot obtain accurate binary annotations from the spatial distribution map.

Lack of exploration of the generation of non-uniform complex structure defects with binary annotations. There are multiple semantic regions in a non-uniform structure image, where texture features are different and the corresponding defects are distinct. As shown in Figure 1, there are two types of textures in a zipper image: fabric and zipper teeth, whose defect contents are significantly disparate. In terms of such non-uniform structure images, networks must generate conforming defects at the specified locations, to obtain realistic synthetic results. In addition, obtaining binary annotations for the generated complex texture defect images is also a challenging task. Shuanlong N. et al. [18] used random seeds to construct input masks, while this is limited to simple stripe defects in some uniform textures. Du-Ming T. et al. [19] adopted two CycleGANs to preserve normal backgrounds and threshold segmentation to obtain binary annotations. However, this work also only synthesized defects with uniform textures, and the overall networks contained four generators and four discriminators, which were too complicated.

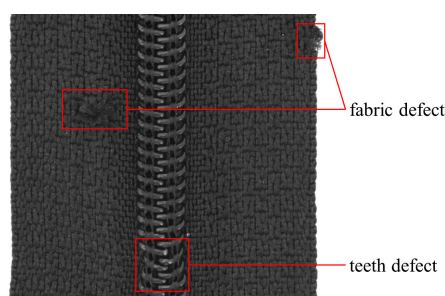


Figure 1. An example of a defect sample with non-uniform structures. The top two boxes indicate a type of fabric defect. The bottom box indicates a type of teeth defect.

To tackle these challenges, we proposed a MDGAN (mask-guided defect generation adversarial network) based on CGAN [22]. The MDGAN can generate realistic defects in regions specified by the input binary mask. First, we introduce a BRM (background replacement module) to extract normal backgrounds using a binary mask to replace the contents at the corresponding positions in the feature maps. The BRM achieves the preservation of the normal backgrounds and facilitates the separate control of the normal backgrounds and

the shape of defects. In addition, the generated defect textures can be controlled by training MDGAN separately for different types of defects. Second, we proposed a DDM (double discrimination module), to extract the defect features from the whole feature map with the guidance of binary masks and measure the authenticity of the whole and the local based on one discriminator. In addition, we constructed a pseudo-normal background for each defect image, to provide paired training inputs. This preprocess ensures MDGAN generates defects according to normal features in the same regions, thus enabling generation of defects with non-uniform structures. Finally, the outputs of MDGAN and the input binary masks were combined to construct our pixel-level annotated synthetic datasets.

In summary, the main contributions of this work are as follows.

- (1) We constructed corresponding pseudo-normal backgrounds for defective images, which solves the problem of the lack of paired training inputs in industrial defect generation and avoids the dependence on CycleGAN.
- (2) We proposed a MDGAN, to achieve independent control of the normal backgrounds, defect shapes, and defect textures of images. The addition of BRM achieves the preservation of normal backgrounds and enables the acquisition of binary annotations. Our DDM focuses on the defect region and the whole image simultaneously, ensuring the quality of the generated results.
- (3) Since BRM can achieve total preservation of the normal background in the generated defect images, our MDGAN can also achieve defect transfer between datasets with similar defect contents.

The subsequent sections of this article are organized as follows: Section 2 proposes the MDGAN. Section 3 introduces the related datasets used in this work. Section 4 details the generation, ablation, comparison, and segmentation experiments. Finally, we summarize our work in Section 5.

2. Methods

This section introduces the construction of the paired training inputs, the architectures, and the training process of MDGAN.

2.1. Pseudo-Normal Background

Image-to-image (I2I) translation [23] is the most effective method to convert images in the source domain into the target domain, where paired source-target images are needed in training. Defect synthesis based on normal images is an I2I task, where the source domain consists of normal industrial images and the defect images constitute the target domain. However, it is almost impossible to obtain exactly corresponding normal-defective pairs in industrial manufacturing. Many works relied on CycleGAN to avoid this problem. Unfortunately, CycleGAN lacks randomness and cannot retain the background, as shown in Section 1. Therefore, we construct pseudo-normal backgrounds for the defect images. First, we select a similar normal image N for the defect image D , whose binary mask is M . N contains areas where the defect contents in D appears. Then calibrate N to obtain N^c by affine transformation matrix T ,

$$N_{i,j}^c = T * N_{i,j} \quad (1)$$

$$T = \begin{bmatrix} \cos\theta & -\sin\theta & T_x \\ \sin\theta & \cos\theta & T_y \\ 0 & 0 & 1 \end{bmatrix} \quad (2)$$

where $N_{i,j}^c, N_{i,j}$ are points of homogeneous form in N^c and N , respectively, θ is the rotation angle in the anticlockwise direction, and T_x and T_y are translation parameters. Affine transformation can ensure that the area used for filling in the normal image is aligned with the defect area. Finally, use N^c to replace the defect regions in D ,

$$B = (D \odot M_-) \oplus (N^c \odot M) \quad (3)$$

where M_- means $(1-M)$, and $\odot$, $\oplus$ means element-level multiplication and addition, respectively. Then B is used as the pseudo-normal background in the source domain to train the MDGAN.

2.2. MDGAN

As shown in Figure 2, MDGAN consists of a generator and a discriminator. A BRM is proposed to modulate the background in the generator using binary masks, as a way to avoid the modification of the background and control the generated defect shapes. A DDM is proposed, to divide the feature map into a whole feature branch and a defect feature branch, which can guide the discriminator to focus on both the whole and the local regions. Only a discriminator is needed to facilitate the generator to output high-quality defect images. Overall, MDGAN is able to generate images with defects appearing in regions specified by binary masks and preserve the normal backgrounds, combining them with input masks to obtain defect samples with pixel-level annotations.

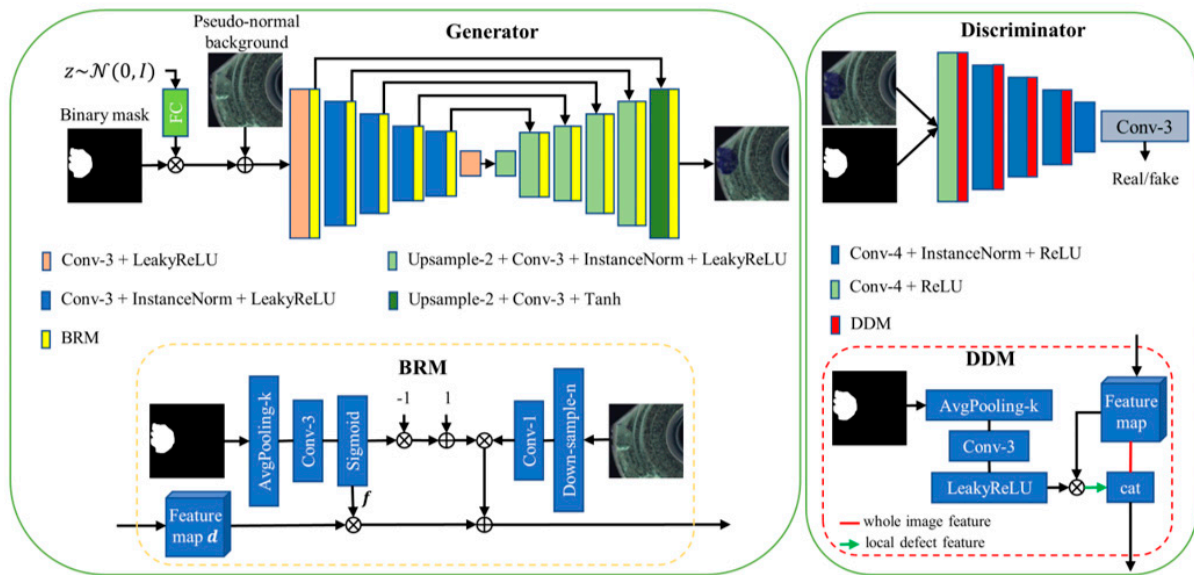


Figure 2. Architectures of MDGAN. *Conv-3* means the convolution kernel size is 3×3 , *Upsample-2* means the feature map is upsampled two times. *AvgPooling-k* means the average pooling kernel size is $k \times k$, which depends on the size of d . *cat* means the channel-level connection of inputs.

2.2.1. Architectures

As shown in Figure 2, the generator is a UNet-like [24] architecture, whose inputs are Gaussian noise, pseudo-normal background, and a 0-1 binary mask. The size of the output is same as the input images. In the output image, the defective contents are at the “1” positions specified by the mask, and the original normal backgrounds are at the “0” positions. The discriminator adopts patchGAN [23] to process the input defect image and binary mask, and its output indicates the probability that the input defect image annotated by the mask is real.

Background Replacement Module (BRM). As shown in Figure 2, BRM employs a binary mask to fuse the real background to the specific positions of the feature map. First, the input binary mask M is average pooling down-sampled, convolved, and activated by Sigmoid, to obtain a weight map f with values of $[0, 1]$. f has the same size as the feature map d of the current layer. The input normal background B is downsampled and convolved with 1×1 kernels, to obtain a background feature map B' with the same size as f . B' is modulated into the generator to obtain the input of the next layer and the skip-connected F ,

$$F = (B' \odot f_-) \oplus (d \odot f), \quad (4)$$

Double Discrimination Module (DDM). As shown in Figure 2, DDM divides the current feature map of the discriminator into whole image features and local defect features for processing. First, M is processed to obtain a weight map with the same size as the feature map, and then the weight map is used to extract the local defect information. Finally, the extracted contents are connected with the original feature map and input into the next layer. Thus, DDM assists the discriminator in processing both the local and global information and ensures the realism of the generated defects. Therefore, we need two discriminators to achieve the above tasks without DDM, which slows down the training and makes the architectures more complex.

2.2.2. Training Objectives

First, a binary mask M ; a real defect image D , which is the ground truth of the generator; and a constructed pseudo-normal background B are given, and we sample random noise z from the Gaussian distribution $\mathcal{N}(0, I)$ as code from latent space. Next, z is mapped and reshaped to obtain x_z with the same size as B . The input of the first convolution layer of the generator is

$$x = x_z \odot M \oplus B, \quad (5)$$

x , M , and B are processed by the generator to obtain the output $D^r = G(M, z, B)$ and the input feature map d_l^r of the last BRM.

Defect region losses. First, we calculate the defect reconstruction loss of D^r and D ,

$$\mathcal{L}_{r1} = 5 \| (D^r - D) \odot M \|_1, \quad (6)$$

where $\| \cdot \|_1$ is the L1 loss. In order to improve the diversity of the generated defects, we sample another $z' \sim \mathcal{N}(0, I)$ to obtain the new outputs $D' = G(M, z', B)$ and d_l' , and calculate the diversity loss of the defect regions,

$$\mathcal{L}_{div} = - \| (D^r - D') \odot M \|_1. \quad (7)$$

Essentially, D^r and D' are synthetic defect images with the same background and annotations, but different defect contents.

Normal background losses. According to the above description, we calculate the reconstruction loss of the normal backgrounds,

$$\mathcal{L}_{r2} = \| (D^r - D) \odot M_- \|_1 + \| (D' - D) \odot M_- \|_1. \quad (8)$$

In addition, when the normal background of d_l (input of the last BRM) is as similar as possible to D , the last BRM can only modulate the details of D into d_l . On the contrary, MDGAN will depend on the last BRM to substantially replace the normal background of d_l , which may lead to a large incoherence between the normal and the defect regions. Therefore, in order to avoid a dependence on the last BRM and obtain a more coherent transition between the defect contents and the normal background of the final output, we calculate $\mathcal{L}_{r3}$ to constrain the backgrounds of d_l to be as similar as possible to D ,

$$\mathcal{L}_{r3} = \| (d_l^r - D) \odot M_- \|_1 + \| (d_l' - D) \odot M_- \|_1, \quad (9)$$

Whole image losses. To ensure the coherence of the defect and background regions after replacement, the gradient loss [25] at the boundary between the two regions is calculated,

$$\mathcal{L}_{grad} = \| F(\nabla M)_{10 \rightarrow 1} \odot (\nabla D - \nabla D^r) \|_1 + \| F(\nabla M)_{10 \rightarrow 1} \odot (\nabla D - \nabla D') \|_1, \quad (10)$$

where ∇ is the gradient and $F(X)_{10 \rightarrow 1}$ sets all the non-zero elements in X to 1.

In order to stabilize the training process, we add the gradient penalty of WGAN-GP [26,27] to the discriminator. We construct two hybrid samples based on the real and the generated defect samples,

$$D_{\alpha}^r = \alpha_1 D + (1 - \alpha_1) D', \quad (11)$$

$$D_{\alpha}' = \alpha_2 D + (1 - \alpha_2) D', \quad (12)$$

where α_1 and α_2 are random numbers $[0, 1]$ and the discriminator needs to process the two hybrid samples, which only back-propagate gradients to the discriminator Dis . The gradient penalty loss is

$$\mathcal{L}_{gp} = \| Dis(D_{\alpha}^r, M) \|_2 + \| Dis(D_{\alpha}', M) \|_2. \quad (13)$$

Then we calculate the adversarial loss, to constrain the reality of the images,

$$\mathcal{L}_{adv} = 2 \| Dis(D, M) \|_2 + \| Dis(D^r, M) - 1 \|_2 + \| Dis(D', M) - 1 \|_2. \quad (14)$$

Full objective. Ultimately, we obtain the final training objective

$$G, Dis = \min_G \max_{Dis} \lambda_r \sum_{i=1}^3 \mathcal{L}_{ri} + \lambda_d \mathcal{L}_{div} + \lambda_g \mathcal{L}_{grad} + \mathcal{L}_{adv} + \lambda_{gp} \mathcal{L}_{gp} \quad (15)$$

where λ_r , λ_d , λ_g , and λ_{gp} control the contribution of each loss to the whole. Based on the trained MDGAN, inputting the normal background and the binary mask to the generator, the output is the synthesized defect image whose pixel-level annotation is the binary mask.

3. Datasets

3.1. Training Sets

MVTec-AD [28] is a commonly used public dataset in industrial vision tasks. MVTec-test contains multiple classes of defect images with pixel-level annotations, and MVTec-train contains many normal images. Therefore, we employed MVTec-AD to construct the training and testing sets of MDGAN.

In this work, experiments were conducted on four items, including the grid, zipper, capsule, and metal nut in MVTec-AD. Figure 3 shows the training images cropped from the original large images. It can be seen that there are multiple complex texture regions in these items. In addition, the phone band images from the actual production line were used in this work. We classified the phone band defects into dirty, roll, and scratch, to facilitate the network in reducing unnecessary blending. In addition, the defect samples for subsequent segmentation testing needed to be pre-preserved. The numbers of the relevant datasets are shown in Table 1. The training sets for generation and segmentation were taken from the same original images. *Zipper-combined* means there are multiple classes of defects in one image, which were only used in segmentation testing. As shown in Table 1, we replaced some long defect names with initials, to simplify the description.

3.2. Testing Sets

We employed two methods to obtain binary masks to construct the rich MDGAN test sets. First, MVTec-AD provided many binary masks of defect samples, which characterized the shapes of industrial defects and could be cropped as inputs for MDGAN. Second, to generate more defects with different shapes, we acquired a large number of binary masks based on Perlin noise [29]. Figure 4 shows the process of Perlin-noise-based mask generation. The normal images and binary masks were cropped to obtain image pairs. Based on the above two methods, we could obtain an arbitrary number of test sets.

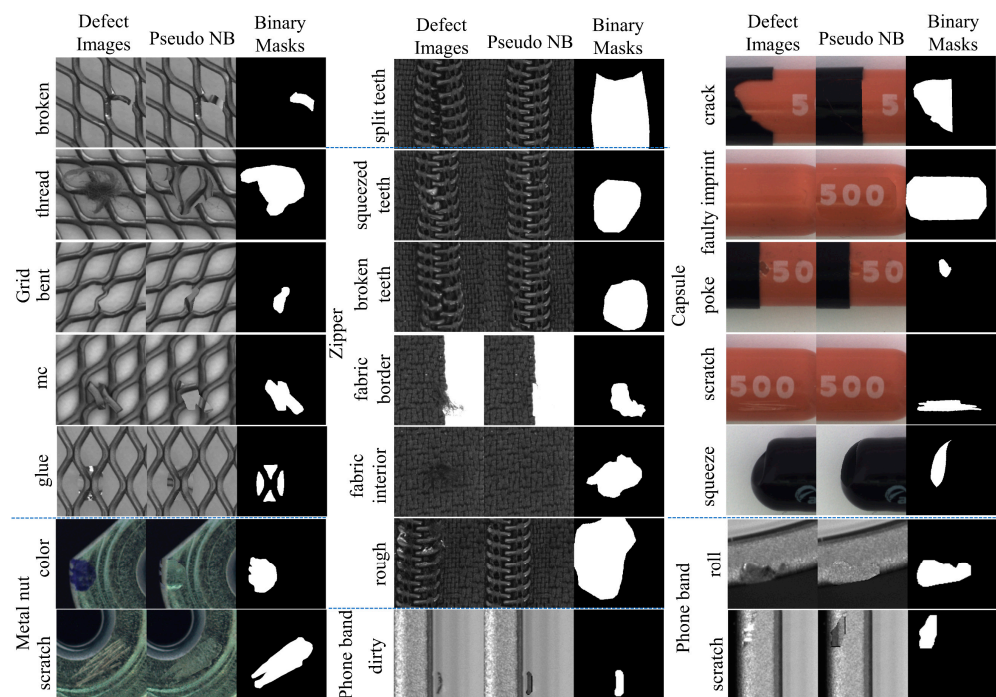


Figure 3. Non-uniform structure images in training sets. *Defect Images* (256^2) are cropped from the original real images. *Pseudo NB* (256^2) are the cropped constructed pseudo-normal backgrounds. *Binary Masks* (256^2) are the cropped binary annotation images. In particular, the size of training images in *Capsule* and *phone band* were 320^2 and 128^2 , respectively.

Table 1. The number of datasets (original/cropped images). *Training* means the training sets of MDGAN. *EL/AUG* means the number of augmented samples from MDGAN (*EL*) and traditional methods (*AUG*). *RAW* and *Seg-test* are the real training and testing sets of segmentation, respectively.

Item Defect	Dataset	Training	EL/AUG	RAW	Seg-Test
Metal nut	scratch	18/1282	900	18/848	5/23
	color	17/1108	800	17/919	5/19
	bent			19/656	6/23
	flip			19/931	4/19
Grid	bent	10/820	500	10/507	2/26
	broken	10/775	500	10/500	2/33
	glue	9/542	700	9/323	2/29
	mc	9/580	800	9/200	2/21
	thread	8/677	500	8/493	3/20
Zipper	broken teeth (bt)	14/771	800	14/597	5/23
	fabric border (fb)	12/817	900	12/521	5/44
	fabric interior (fi)	11/741	1000	11/465	5/38
	rough	12/1105	600	12/814	5/14
	split teeth (spt)	13/867	800	13/620	5/37
	squeezed teeth (sqt)	12/658	1000	12/479	4/22
Capsule	combined				16/87
	crack	23/643	700	23/459	5//12
	faculty imprint (fm)	22/672	900	22/309	4/7
	poke	21/518	800	21/358	4/13
	scratch	23/806	600	23/509	4/19
Phone band	squeeze	20/861	500	20/650	4/25
	scratch	24/496	550	24/258	5/11
	roll	52/1031	550	52/924	12/27
	dirty	49/497	550	49/344	12/24

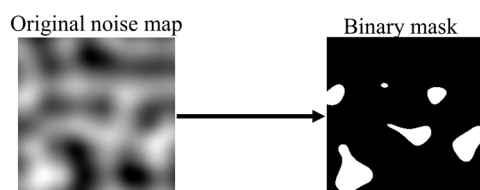


Figure 4. Example of generating a binary mask from Perlin noise. Set the values greater than 0.4 in the *Original noise map* (256^2) to 1 and other values to 0 to obtain a *Binary mask* (256^2).

4. Experiments

MDGAN was used to synthesize defect images based on the above datasets. Except for the metal nut and capsule, which involved three-channel color images, the images consisted of single-channel grayscale images. Section 4.2 shows the generated quality, diversity, and annotation accuracy of MDGAN. The effectiveness of BRM and DDM are demonstrated in Section 4.3. We also compare MDGAN with the most commonly used CycleGAN in Section 4.5, to certify the effectiveness of our methods. Then we assess the advantages of our synthetic samples over traditional augmented results. Finally, we explore the possibility of defect transfer using MDGAN in Section 4.6.

4.1. Implementations

To construct the pseudo-normal backgrounds, we first obtain the main area where defects may occur in defect image D and select normal image N by thresholding, and draw the minimum external rectangular boxes of these areas. Then we conduct affine transformation for D and N , so that the length and width of the two boxes are parallel and the center points are overlapped. Finally, the defect contents in D are filled by N , and the filled image is reversely transformed to the attitude of the original D . These two affine transformations can be simplified as Equations (1)–(3). We performed the above processes based on the OpenCV library, where functions such as `cv2.getRotationMatrix2D()` and `cv2.warpAffine()` were used.

MDGAN was trained for each type of defect under each item. We augmented the training image pairs by rotation, flipping, and random cropping. The binary mask was normalized to $[-1, 1]$ when inputting to the MDGAN and $[0, 1]$ when calculating losses. The number of output channels of the 12 convolution layers of the generator from input to output were 64, 128, 256, 256, 512, 512, 512, 256, 256, 128, and 64, respectively; those of the discriminator were 32, 128, 256, 256, 512, and 1, respectively. The dimension of latent space was 8, and the hyperparameters of Equation (15) were set as $\lambda_r = 10$, $\lambda_d = 15$, $\lambda_g = 10$, and $\lambda_{gp} = 10$. The Adam optimizer was used with $\beta_1 = 0.5$, $\beta_2 = 0.999$ to train MDGAN, with a batch size = 20 and learning rate = 0.0004. We trained the MDGAN for 500 iterations on one NVIDIA GeForce RTX 3090 GPU of a server with Intel(R) Xeon(R) Gold 622306R CPU @ 2.90 GHz. During training, backgrounds could be retained without the last BRM in some datasets, which could simplify the architecture and obtain a better transition between normal backgrounds and defect regions. Therefore, we eliminated the last BRM in the defects of the grid.

4.2. Synthetic Results

Some synthetic defect samples are shown in Figures 5 and 6. As shown in Figure 5, MDGAN achieved defect image generation for all categories. Good synthesis results were obtained for complex and weak defects, such as the fine-grained texture defects (zipper-fb, zipper-fi, grid-thread, etc.), color defects (metal nut-color), metallic defects (grid-mc), and weak defects (phone band). As seen from the *Defect contents* in Figure 5, the generated defect contents were realistic and accurately distributed in the locations marked by the mask, achieving an accurate correspondence between the synthetic defect images and the input binary masks. In addition, MDGAN preserved the background structures in the *Grid* reasonably well, despite canceling the last BRM. In summary, with the help of BRM and DDM, MDGAN was able to generate realistic defect images, while preserving

the original real backgrounds outside the annotations and achieved a natural transition between generated defects and real background, so that the generated defect images were accurately labeled by input binary masks.

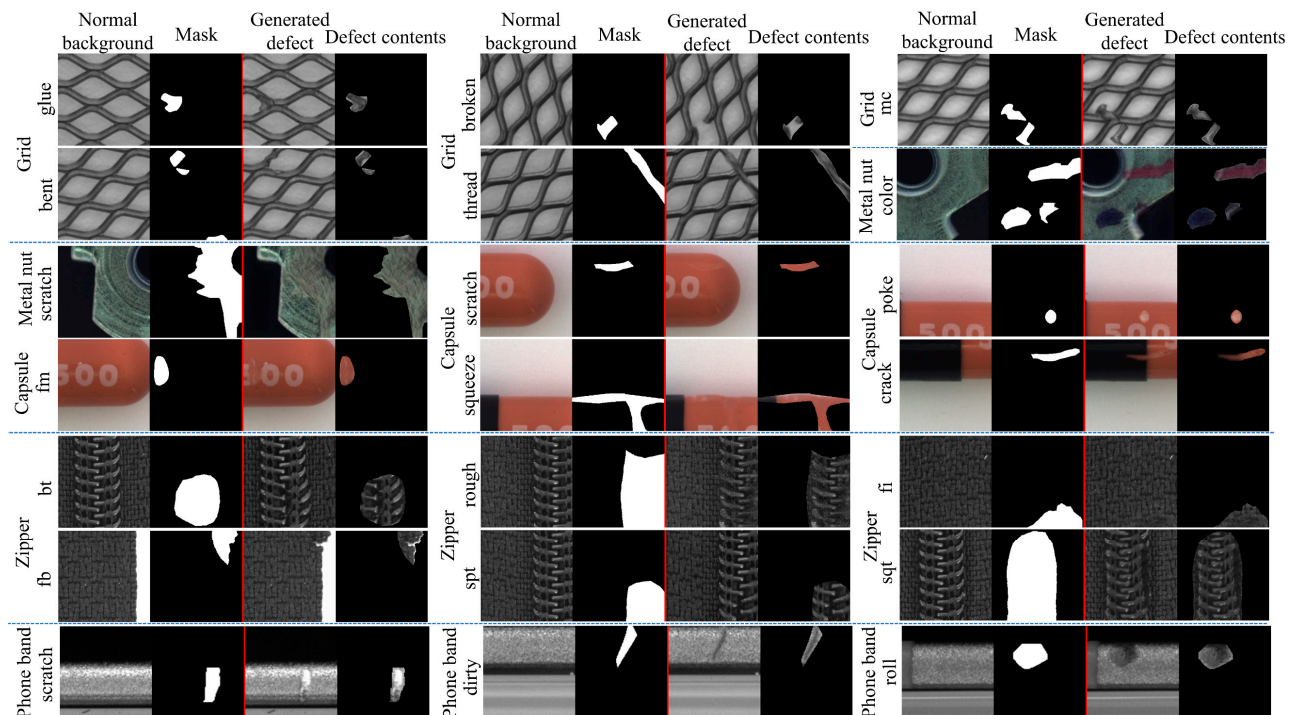


Figure 5. Synthesis results for each item. *Normal background* indicates the real background image, *Mask* is the corresponding binary mask, and both were input into the generator to obtain the synthetic defect image *Generated defect*. *Defect contents* were segmented by the *Mask* from the *Generated defect*.

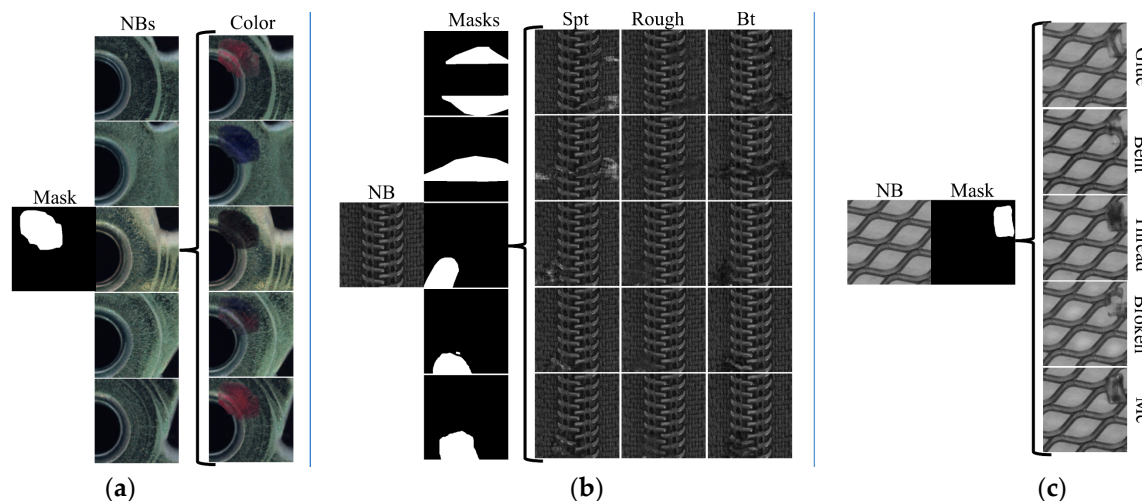


Figure 6. Examples of one-to-many generation. *NB* and *mask* denote the normal background and the binary mask input into MDGAN, respectively. (a) Background variation. *Color* denotes the generated metal nut color defects. (b) Defect shape and texture variation. *Spt*, *Rough*, and *Bt* denote the three types of zipper defects generated from the same background. (c) Defect texture variation.

Controllability of the backgrounds, defect shapes, and defect textures. We separately processed these three aspects, to verify that MDGAN could generate various defect images for a normal background and obtain accurate annotations. Figure 6 shows the generated results when normal backgrounds, defect shapes, and defect textures were changed, respectively. First, the defect images in Figure 6a shared the same binary mask. This shows

that BRM added different backgrounds to the generator, to obtain multiple defect samples, whose annotations were the same, and the defects varied with the background. Second, Figure 6b shows the generated defect images of the zipper, with the same background *NB* and five different masks. Defect images in the same row shared the same mask on the left side. It can be seen that the BRM could replace different background regions with different masks, which assisted MDGAN in generating multiple defects with different shapes and contents on the same normal image. Third, Figure 6c shows multiple categories of generated defects in the grid, where the defect images shared the same normal background (*NB*) and annotations (*Mask*), but had different defective textures. As seen from Figure 6c and each row in Figure 6b, MDGAN could produce multiple types of defect in the same specified region of the same normal image when the training sets were different.

In summary, based on BRM and DDM, MDGAN achieved independent control of the background, defect shape, and defect texture, and was able to generate a huge number of diverse and high-quality defect samples for a normal image. In reality, various defects may appear in a normal image, and our experimental results fit actual situations. Moreover, since MDGAN accurately controls defect shapes and preserves backgrounds using the given binary mask, the output defect regions are precisely labeled by the mask. Thus, our synthetic samples can be used to train segmentation models, which is beneficial for detecting defects.

4.3. Ablation Experiments

To verify the effect of BRM on the background retention and DDM on generated quality, we separately removed the two modules and performed ablation experiments on the above datasets, where the remaining settings were exactly the same as in the formal experiments.

Without BRM. Figure 7 shows the comparison results of the with/without (wo) BRM for the same pairs of test images. As can be seen, the results generated without BRM have drastically and unreasonably modified backgrounds, while MDGAN retained the details of the backgrounds well. In particular, substantial modifications to the original backgrounds led to the loss of real structures in the *phone band*. The generated defect contents did not appear accurately in the annotated positions without BRM, resulting in inaccurate binary annotations.

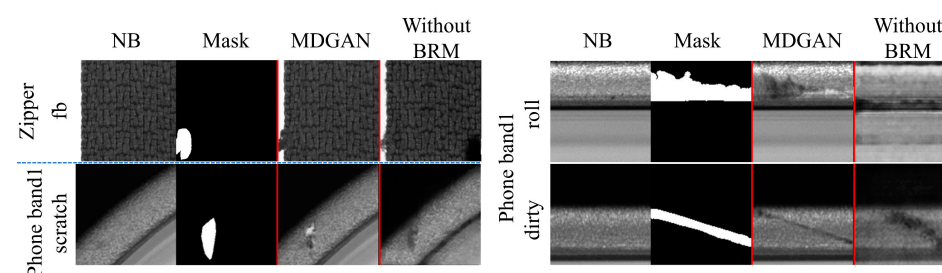


Figure 7. Comparison of with and without BRM. *Background* and *Mask* are input normal backgrounds and binary masks, *MDGAN* is the synthetic result of MDGAN. *Without BRM* is the synthetic result after canceling BRM.

In order to quantify the background retention ability of BRM, we employed the structure similarity index measure (SSIM) to evaluate the background similarity. The SSIM value is between [0, 1], and the larger the value, the more similar the two images. Using the same test set, and based on the with/wo BRM models, to synthesize 1000 defective samples, the SSIM between the normal background areas of the output and the input was calculated. The mean SSIM (mSSIM) is shown in Table 2. It can be seen that the backgrounds generated by MDGAN were highly similar to the real backgrounds. Both the qualitative and quantitative results showed that BRM can modulate the real backgrounds in the feature map of the generator, which restricted the location of the generated defects, avoided the loss of backgrounds in the training, and finally facilitated MDGAN in generating defect samples with realistic backgrounds and accurate binary annotations.

Table 2. Quantitative results of the ablation experiments. Bold fonts indicate better results.

Dataset		mSSIM		Dataset		mSSIM	
		MDGAN	wo BRM			MDGAN	wo BRM
Capsule	crack	0.998	0.915	Zipper	fb	0.996	0.736
	fm	0.998	0.888		fi	0.990	0.793
	poke	0.998	0.921		rough	0.890	0.832
	scratch	0.998	0.937		bt	0.932	0.849
	squeeze	0.998	0.927		sqt	0.996	0.800
Grid	bent	0.975	0.917	Metal nut	spt	0.899	0.813
	broken	0.977	0.927		color	0.997	0.716
	glue	0.964	0.935	Phone band	scratch	0.997	0.725
	mc	0.977	0.913		dirty	0.996	0.734
	thread	0.972	0.902		roll	0.997	0.707
					scratch	0.996	0.665

Without DDM. All DDMs were removed in the discriminator. The mask and image were channel-level connected and input into the discriminator. To match the formal experiments, the number of output channels of the first three convolutional layers was doubled. As shown in Figure 8, the discriminator's binding on the defect quality decreased after removing the DDM. As shown in Figure 8a, the generated defects without DDM contained unrealistic stripes and the zipper teeth were not smooth enough, being totally different from the real images. In addition, as shown in Figure 8b, there were unreasonable black contents in the generated *capsule-crack* without DDM, while MDGAN smoothly stripped the black blocks and generated clear red defects on them. On *capsule-fm*, without DDM covered the annotated region with only fuzzy uniform white blocks, while MDGAN generated realistic and detailed scratches. That is, the networks could not inherit and generate the real defects without DDM. Overall, the addition of DDM assisted the discriminator in focusing on both on the whole image and the local defects, to improve judgment and enable MDGAN to accurately capture the stripes and grayscale distribution of defects and generate more realistic and higher quality images.

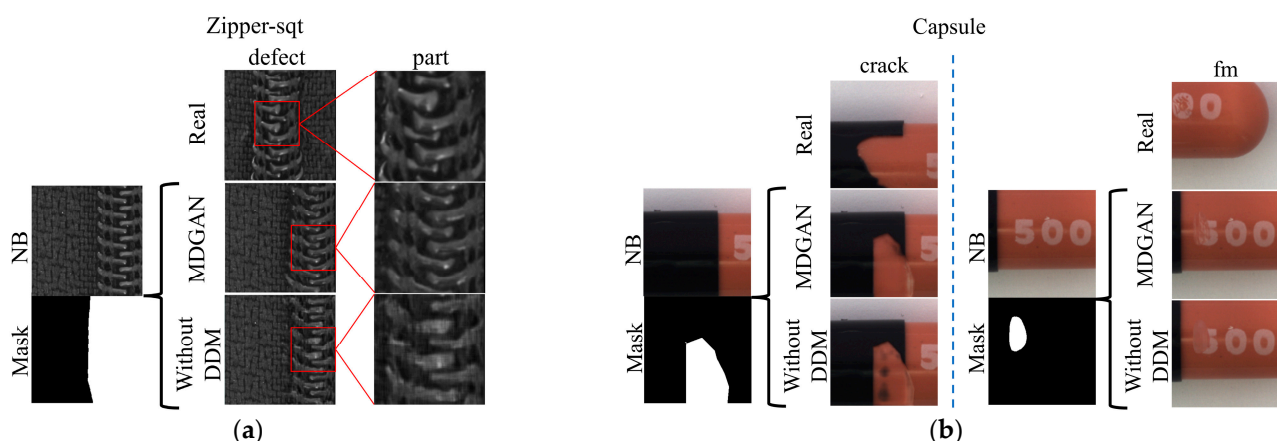


Figure 8. Comparison of with and without DDM. *Real* is the real defect image. *MDGAN* and *Without DDM* are the generated defect samples with and without DDM, respectively. (a) One type of effect without DDM. *part* is the enlarged defect in the red box of the *defect*. (b) Another effect.

4.4. Comparison with CycleGAN

CycleGAN-based methods are leading the way in defect synthesis. To verify the advantages of MDGAN over other methods, CycleGANs were trained on the above datasets.

To help CycleGAN retain the normal background, we added a L1 loss between the input and output to the original losses [17]. Some of results generated by CycleGAN and MDGAN based on the same backgrounds are shown in Figure 9. Despite the addition of the L1 loss, CycleGAN still modified the normal backgrounds, and it could convert the input normal images into defect images for the zipper-fi, zipper-sqt, zipper-spt, and grid-broken. Moreover, there were stretches and artifacts in the generated results of the capsule and phone band. This indicated that CycleGAN could only convert the source image into the most similar target image seen during the training for few-sample datasets, resulting in either a failed generation or loss of structures in the source image. In contrary, MDGAN generated realistic defect images for each category of input normal backgrounds and retained the original normal textures.

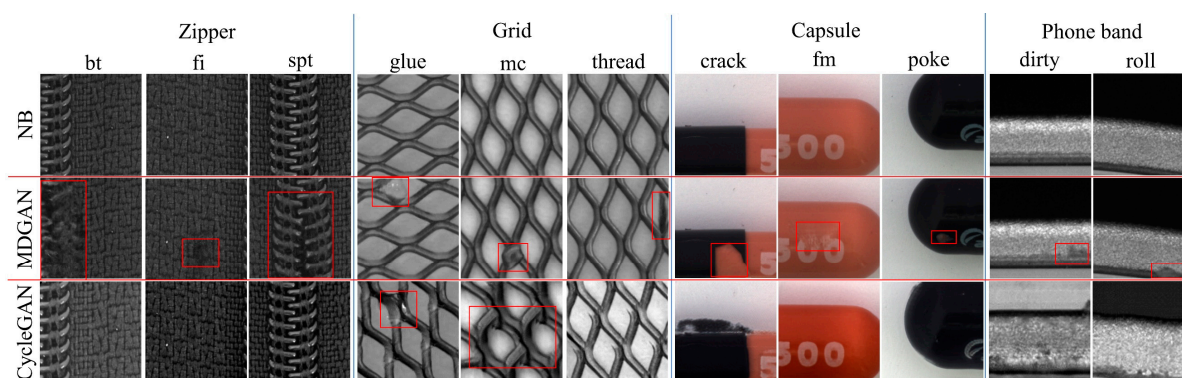


Figure 9. Generated results of MDGAN and CycleGAN with L1 loss based on the same normal backgrounds NB. MDGAN was generated from the NB and a binary mask by MDGAN, and CycleGAN was generated from the NB by CycleGAN. Defects are indicated by red boxes.

To quantify the generation quality, 1000 samples were generated on the same testing set based on CycleGAN and MDGAN, and the FID [30] between the generated and the real defect datasets was calculated. The lower the FID, the closer the generated and real features. As shown in Table 3, MDGAN obtained smaller FID than CycleGAN for most defect types. Nevertheless, CycleGAN had a lower FID for the grid-glue, grid-bent, and zipper-fb. In grid-glue, as shown in Figure 9, CycleGAN simply memorized training sets and translated all test images into the most similar training images, resulting in a low FID. In the other two items, MDGAN generated defects in the given annotated regions, and when the mask shapes used in testing differed significantly from the real, the generated features were a little far away from the real features, resulting in a high FID.

Table 3. FID between generated results and real images. Bold fonts indicate better results.

Dataset	Model	MDGAN	CycleGAN + L1	Dataset	Model	MDGAN	CycleGAN + L1
Capsule	squeeze	81.72	119.2	Zipper	bt	45.69	50.67
	crack	51.99	111.45		rough	48.81	56.47
	fm	39.99	94.96		fb	88.08	86.93
	scratch	45.02	99.39		fi	54.96	59.15
	poke	60.86	97.01		sqt	29.13	136.25
Grid	bent	78.84	63.90	Metal nut	spt	38.4	42.93
	broken	75.98	112.13		color	69.93	95.52
	glue	114.23	94.76	Phone band	scratch	66.90	81.85
	mc	99.39	102.92		dirty	115.31	129.83
	thread	79.74	89.63		scratch	112.54	169.62
					roll	107.88	147.69

Overall, MDGAN constructed pseudo-normal images, to efficiently acquire pairs of inputs without relying on CycleGAN and obtained a higher quality than CycleGAN for most items. Moreover, MDGAN imports random noise to construct latent representations for real defects; in contrast with CycleGAN, which improves both the randomness of the generation and the diversity of results, and is thus more suitable for few-sample synthesis. Furthermore, due to the background preservation with BRM and the constraints on quality from DDM, MDGAN successfully generated defect images with pixel-level annotations, while preserving the real normal backgrounds. Meanwhile, CycleGAN generated samples with only image-level annotations, which could not be used for defect segmentation experiments.

4.5. Detection Performance

To verify the advantages of MDGAN for detection over traditional augmentation, synthetic samples from MDGAN were added to the real segmentation training set RAW, to construct EL; and the traditional brightness adjustment, rotation, and noise injection were adopted to obtain the training set AUG. The number of datasets is shown in Table 1. Two types of segmentation network, UNet and sResNet, were trained using the above three training sets. sResNet is a UNet-like segmentation model, where the skip-connections are removed and the convolution layers are replaced by the Res-blocks in ResNet [31]. UNet consists of six downsampling and six upsampling layers whose number of output channels are 32, 64, 128, 256, 512, 256, 512, 256, 128, 64, 32, and 1. sResNet consists of four downsampling layers, four res-blocks, and four upsampling layers, whose number of output channels are 32, 64, 128, 256, 256, 256, 256, 128, 64, 32, and 1. The output of the two networks is a single-channel map with the same size as the input. We used cross entropy loss to train the two models. The Adam optimizer was used with $\beta_1 = 0.5$, $\beta_2 = 0.999$, batch size = 50, and learning rate = 0.0005 on an NVIDIA GeForce RTX 3090 GPU. The mIoU (mean Intersection over Union) and F1 coefficient were calculated on the same test set at 500th iterations.

The mean testing results of each item are shown in Table 4, where higher values indicate a better detection performance. It can be seen that EL greatly improved the detection performance in contrast to AUG and RAW. The mean results were substantially improved in EL. Among the mean results of the various items, EL outperformed RAW and AUG, with an mIoU improvement up to 7.3% (UNet-capsule) and F1 up to 6.2% (sResNet-capsule). On the contrary, AUG reduced the overall detection results (zipper). Figure 10 shows a qualitative comparison of the segmentation results; the results of EL had less over-kill and escape versus RAW and AUG and were closer to the ground truths.

Table 4. Comparison of the mean test results for the three training sets. Bold fonts indicate the optimal IoU and F1 among the three results. R and A denote RAW and AUG, respectively.

Network		UNet						sResNet					
Index		mIoU		F1		mIoU		F1					
Dataset	R	A	EL	R	A	EL	R	A	EL	R	A	EL	
Metal nut	0.742	0.748	0.768	0.846	0.848	0.86	0.733	0.738	0.751	0.844	0.84	0.847	
Grid	0.619	0.603	0.652	0.755	0.714	0.782	0.636	0.652	0.657	0.769	0.783	0.786	
Zipper	0.732	0.719	0.775	0.970	0.959	0.975	0.745	0.723	0.764	0.977	0.959	0.979	
Capsule	0.548	0.571	0.621	0.692	0.713	0.748	0.472	0.483	0.561	0.618	0.627	0.689	
Phone band	0.656	0.646	0.666	0.979	0.985	0.987	0.65	0.653	0.681	0.984	0.981	0.985	

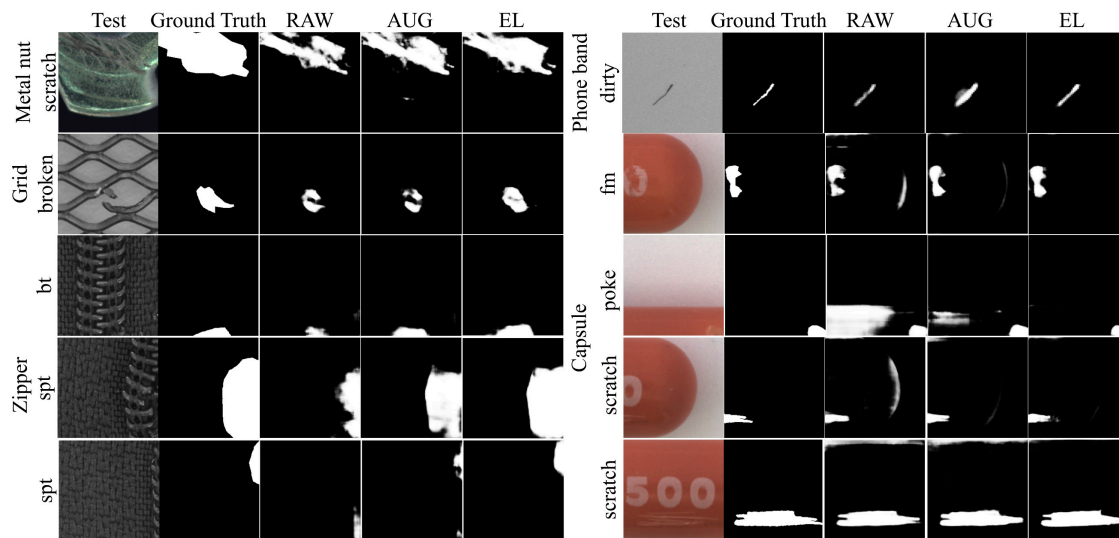


Figure 10. Comparison of UNet-based segmentation results. *Test* and *Ground Truth* denote the test images and the corresponding ground truth (256^2). *RAW*, *AUG*, and *EL* denote the test results obtained by the three training sets (256^2).

Overall, the inclusion of synthetic samples alleviates the data problems of class imbalance, lack of diversity, and few samples, and improves the detection performance. Compared with the traditional augmented samples, there were various backgrounds, defect contents, and shapes of binary annotations in our synthesized datasets, which were very different from the original training sets and could assist the networks in seeing and remembering richer defect information during training. When testing, the EL-trained models learned more knowledge and were more conducive to detecting unseen samples. This demonstrated that the synthetic samples from MDGAN could be used to train the supervised segmentation networks and that our work has practical application value.

4.6. Defect Transfer

It can be seen from the above that MDGAN achieved defect image generation with complete retention of backgrounds. Therefore, we could try to employ the MDGAN-trained source dataset to perform defect synthesis on a target dataset and used only the synthetic target defect images to train models to detect real defects in the target dataset, thus achieving defect transfer and zero-shot detection of the target dataset.

Figure 11 shows the related images and procedure of defect transfer, where the first and second rows in the *Source* are from phone bands on the reverse side (phone band1) and curved glass, respectively, and those in *Target* are from phone band on the front side (phone band2) and phone cover glass, respectively. It can be seen that the *Source* and *Target* have different structures, but their defect contents were similar. In consequence, we trained MDGAN based on phone band1 and curved glass and tested it on phone band2 and phone cover glass. The generated results are shown in the middle parts of Figure 11. It can be seen that, despite the differences in the backgrounds on the two sides of the defect transfer, MDGAN still generated defects in the annotated areas (*Defect contents*) and obtained defect images (*GD*) similar to the real ones. As shown in *Res*, MDGAN only modified the annotated regions and fully retained the targeted normal backgrounds after transfer.

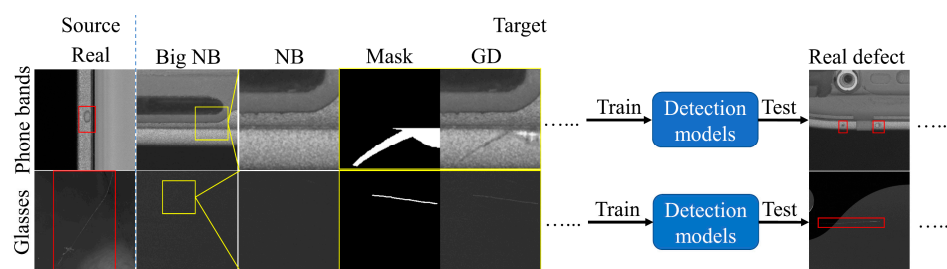


Figure 11. The first row shows the two types of *Phone bands*, and the second is the two types of *glass*. Red boxes indicate real defects. *Source* is the defect image used for training MDGAN. *Target* means the images to be transferred. *NB* is the normal background cut from *Big NB*. *Mask* is the input binary mask. *GD* is the generated target defect image. *Real defect* is defect image of the target datasets to be tested.

In order to verify the effect of the transferred defect samples, we only used the transferred defect samples to train the segmentation networks and adopted the real images as the test set. The number of datasets and the test results are shown in Table 5. It can be seen that good results were obtained on the real test set based on segmentation models trained only using transferred defect samples. The AUC (area under curve) was up to **0.971** (UNet) for phone band1 detection and **0.991** (sResNet) for phone cover glass detection.

Table 5. The numbers of transferred training sets and real test sets, and the test results of segmentation.

Datasets	Train	Test	Models	mIoU	F1	AUC
Phone band2	2508	186	UNet	0.464	0.622	0.971
			sResNet	0.425	0.579	0.969
Phone cover glass	1302	115	UNet	0.458	0.636	0.987
			sResNet	0.439	0.619	0.991

Based on the generated samples from MDGAN trained using another dataset, we achieved defect detection on two real industrial surface defect datasets, showing that MDGAN could achieve defect transfer between datasets with similar defects but different backgrounds. Hence, we could train MDGAN based on the existing source defect datasets and then synthesize a large number of defect samples on the new target normal backgrounds when the type of products changed. As long as the defect textures are similar, the resource consumption for recollecting and labeling datasets can be greatly reduced by MDGAN, which is highly valuable for intelligent manufacturing.

5. Conclusions

This paper proposes MDGAN, to tackle the problems of falsifying backgrounds, lack of pixel-level annotations, and less attention being given to non-uniform complex structures in the current defect synthesis methods. Guided by binary masks, MDGAN modulates the real background into the generator using BRM and employs DDM to achieve discrimination of both local and global information. Due to the background-preserving effect of BRM and the quality constraint of DDM, MDGAN solves the problem of falsifying backgrounds and enriches the diversity of datasets. Finally, defect samples with accurate pixel-level annotations on multiple datasets with complex textures were synthesized using MDGAN. In addition, the qualitative and quantitative results showed that MDGAN obtained a better quality than the commonly used CycleGAN. The segmentation results demonstrated that the synthetic samples from MDGAN greatly improved the detection performance, with an improvement of IoU up to 7.3% and F1 up to 6.2%. Furthermore, based on the excellent background retention capability of MDGAN, we successfully synthesized target defect images using MDGAN trained on a source dataset and achieved the defect detection of real target samples, based only on the synthesized samples.

There are some aspects that need to be improved in our work. Since the binary masks are directly given by the test sets, the feature and the diversity of the generated defect shapes can be limited. In follow-up work, we will explore the synthesis method of producing both defect images and annotations using networks, as a way to enrich the defect shapes.

Author Contributions: Conceptualization, Z.Z. and F.S.; Formal analysis, J.W. and C.L.; Investigation, J.W. and C.L.; Methodology, J.W., Z.Z. and F.S.; Resources, Z.Z.; Software, Z.Z.; Writing—original draft, J.W.; Writing—review and editing, J.W., Z.Z., F.S. and C.L. All authors have read and agreed to the published version of the manuscript.

Funding: This work was supported by the Youth Innovation Promotion Association, CAS (2020139).

Data Availability Statement: Not applicable.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Tabernik, D.; Šela, S.; Skvarč, J.; Skočaj, D. Segmentation-based deep-learning approach for surface-defect detection. *J. Intell. Manuf.* **2019**, *31*, 759–776. [\[CrossRef\]](#)
2. Ren, R.; Hung, T.; Tan, K.C. A Generic Deep-Learning-Based Approach for Automated Surface Inspection. *IEEE Trans. Cybern.* **2017**, *48*, 929–940. [\[CrossRef\]](#) [\[PubMed\]](#)
3. Chen, Y.; Ding, Y.; Zhao, F.; Zhang, E.; Wu, Z.; Shao, L. Surface Defect Detection Methods for Industrial Products: A Review. *Appl. Sci.* **2021**, *11*, 7657. [\[CrossRef\]](#)
4. Saksham, J.; Gautam, S.; Paruthi, A.; Soni, U.; Kumar, G. Synthetic data augmentation for surface defect detection and classification using deep learning. *J. Intell. Manuf.* **2022**, *33*, 1007–1020.
5. Czimmermann, T.; Ciuti, G.; Milazzo, M.; Chiurazzi, M.; Roccella, S.; Oddo, C.M.; Dario, P. Visual-Based Defect Detection and Classification Approaches for Industrial Applications—A SURVEY. *Sensors* **2020**, *20*, 1459. [\[CrossRef\]](#) [\[PubMed\]](#)
6. Shorten, C.; Khoshgoftaar, T.M. A survey on Image Data Augmentation for Deep Learning. *J. Big Data* **2019**, *6*, 60. [\[CrossRef\]](#)
7. Goodfellow, I.; Pouget-Abadie, J.; Mirza, M.; Xu, B.; Warde-Farley, D. Generative adversarial nets. *Adv. Neural Inf. Process. Syst.* **2014**, *15*, 2672–2680.
8. Makhzani, A.; Shlens, J.; Jaitly, N.; Goodfellow, I.; Frey, B. Adversarial autoencoders. *arXiv* **2015**, arXiv:1511.05644.
9. Kingma, D.P.; Welling, M. Auto-encoding variational bayes. *arXiv* **2013**, arXiv:1312.6114.
10. Karras, T.; Laine, S.; Aila, T. A Style-Based Generator Architecture for Generative Adversarial Networks. In Proceedings of the 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Long Beach, CA, USA, 15–20 June 2019; pp. 4396–4405. [\[CrossRef\]](#)
11. Karras, T.; Laine, S.; Aittala, M.; Hellsten, J.; Lehtinen, J.; Aila, T. Analyzing and improving the image quality of stylegan. In Proceedings of the 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Seattle, WA, USA, 16–18 June 2020; pp. 8110–8119. [\[CrossRef\]](#)
12. Choi, Y.; Choi, M.; Kim, M.; Ha, J.W.; Kim, S.; Choo, J. Stargan: Unified generative adversarial networks for multi-domain image-to-image translation. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–23 June 2018; pp. 8789–8797.
13. Choi, Y.; Uh, Y.; Yoo, J.; Ha, J.W. Stargan v2: Diversified image synthesis for multiple domains. In Proceedings of the 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Seattle, WA, USA, 13–19 June 2020; pp. 8188–8197.
14. Liu, J.; Wang, C.; Su, H.; Du, B.; Tao, D. Multistage GAN for Fabric Defect Detection. *IEEE Trans. Image Process.* **2019**, *29*, 3388–3400. [\[CrossRef\]](#) [\[PubMed\]](#)
15. Niu, S.; Li, B.; Wang, X.; Lin, H. Defect Image Sample Generation with GAN for Improving Defect Recognition. *IEEE Trans. Autom. Sci. Eng.* **2020**, *17*, 1611–1622. [\[CrossRef\]](#)
16. Zhang, G.; Cui, K.; Hung, T.-Y.; Lu, S. Defect-GAN: High-Fidelity Defect Synthesis for Automated Defect Inspection. In Proceedings of the 2021 IEEE Winter Conference on Applications of Computer Vision (WACV), Waikoloa, HI, USA, 3–8 January 2021; pp. 2523–2533. [\[CrossRef\]](#)
17. Su, B.; Zhou, Z.; Chen, H.; Cao, X. SIGAN: A Novel Image Generation Method for Solar Cell Defect Segmentation and Augmentation. *arXiv* **2021**, arXiv:2104.04953.
18. Niu, S.; Li, B.; Wang, X.; Peng, Y. Region- and Strength-Controllable GAN for Defect Generation and Segmentation in Industrial Images. *IEEE Trans. Ind. Inf.* **2021**, *18*, 4531–4541. [\[CrossRef\]](#)
19. Tsai, D.-M.; Fan, S.-K.S.; Chou, Y.-H. Auto-Annotated Deep Segmentation for Surface Defect Detection. *IEEE Trans. Instrum. Meas.* **2021**, *70*, 1–10. [\[CrossRef\]](#)
20. Jalayer, M.; Jalayer, R.; Kabolli, A.; Orsenigo, C.; Vercellis, C. Automatic Visual Inspection of Rare Defects: A Framework based on GP-WGAN and Enhanced Faster R-CNN. In Proceedings of the 2021 IEEE International Conference on Industry 4.0, Artificial Intelligence, and Communications Technology (IAICT), Bandung, Indonesia, 27–28 July 2021; pp. 221–227. [\[CrossRef\]](#)

21. Zhu, J.-Y.; Park, T.; Isola, P.; Efros, A.A. Unpaired image-to-image translation using cycle-consistent adversarial networks. In Proceedings of the IEEE International Conference on Computer Vision, Venice, Italy, 22–29 October 2017; pp. 2223–2232.
22. Mirza, M.; Osindero, S. Conditional generative adversarial nets. *arXiv* **2014**, arXiv:1411.178.
23. Isola, P.; Zhu, J.-Y.; Zhou, T.; Efros, A.A. Image-to-Image Translation with Conditional Adversarial Networks. In Proceedings of the 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Honolulu, HI, USA, 21–26 July 2017; pp. 5967–5976. [[CrossRef](#)]
24. Ronneberger, O.; Fischer, P.; Brox, T. U-Net: Convolutional Networks for Biomedical Image Segmentation. *Int. Conf. Med. Image Comput. Comput. Assist.* **2015**, 9351. [[CrossRef](#)]
25. Zhang, L.; Wen, T.; Shi, J. Deep Image Blending. In Proceedings of the 2020 IEEE Winter Conference on Applications of Computer Vision (WACV), Snowmass, CO, USA, 1–5 March 2020; pp. 231–240. [[CrossRef](#)]
26. Arjovsky, M. Soumith Chintala, and Léon Bottou, Wasserstein Generative Adversarial Networks. In Proceedings of the 34th International Conference on Machine Learning, Sydney, NSW, Australia, 6–11 August 2017. [[CrossRef](#)]
27. Gulrajani, I.; Ahmed, F.; Arjovsky, M.; Dumoulin, V.; Courville, A.C. Improved Training of Wasserstein GANs. *Adv. Neural Inf. Process. Syst.* **2017**, 30. [[CrossRef](#)]
28. Bergmann, P.; Fauser, M.; Sattlegger, D.; Steger, C. MVTec AD—A Comprehensive Real-World Dataset for Unsupervised Anomaly Detection. In Proceedings of the 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Long Beach, CA, USA, 15–20 June 2019; pp. 9584–9592. [[CrossRef](#)]
29. Perlin, K. Improving noise. *ACM Trans. Graph.* **2002**, 21, 681–682. [[CrossRef](#)]
30. Heusel, M.; Ramsauer, H.; Unterthiner, T.; Nessler, B.; Hochreiter, S. Gans trained by a two time-scale update rule converge to a local nash equilibrium. *Adv. Neural Inf. Process. Syst. (NeurIPS)* **2017**, 30, 6629–6640. [[CrossRef](#)]
31. He, K.; Zhang, X.; Ren, S.; Sun, J. Deep residual learning for image recognition. In Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR), Las Vegas, NV, USA, 27–30 June 2016; pp. 770–778. [[CrossRef](#)]