Machine Vision and Applications (2021) 32:118
https://doi.org/10.1007/s00138-021-01244-z

ORIGINAL PAPER

Defect segmentation for multi-illumination quality control systems

David Honzátko[1,2] · Engin Türetken[1] · Siavash A. Bigdeli[1] · L. Andrea Dunbar[1] · Pascal Fua[2]

Received: 6 November 2020 / Revised: 8 June 2021 / Accepted: 16 August 2021

Abstract
Thanks to recent advancements in image processing and deep learning techniques, visual surface inspection in production lines has become an automated process as long as all the defects are visible in a single or a few images. However, it is often necessary to inspect parts under many different illumination conditions to capture all the defects. Training deep networks to perform this task requires large quantities of annotated data, which are rarely available and cumbersome to obtain. To alleviate this problem, we devised an original augmentation approach that, given a small image collection, generates rotated versions of the images while preserving illumination effects, something that random rotations cannot do. We introduce three real multi-illumination datasets, on which we demonstrate the effectiveness of our *illumination preserving rotation* approach. Training deep neural architectures with our approach delivers a performance increase of up to 51% in terms of AuPRC score over using standard rotations to perform data augmentation.

Keywords Defect detection · Multi-illumination · Deep learning · Data augmentation · Photometric stereo · Quality control

1 Introduction

Defect detection has long been central to a wide range of industrial vision systems that are required to provide reliable, high-speed, and cost-effective quality checks at frequent intervals. Automated visual inspection has recently emerged as a viable alternative to traditional inspection by human experts. However, the challenge to detect thin defects on surfaces with complex reflectance properties still remains.

Defects are often only visible under specific lighting conditions. Therefore, acquiring several images while varying the light orientation and intensity without moving the camera is an effective way to reveal these defects, see Fig. 1. This idea is widely used in industrial inspection systems. It usually involves first pre-processing the stack of images generated in this manner to estimate the surface properties (e.g., normals, or gradients), as in photometric stereo [29]. These properties can then be used for detection purposes. Unfortunately, most current systems are designed for very specific surface or defect types, which makes them over-specialized. Furthermore, they only use a small number of different light directions, which is in general suboptimal.

Addressing these limitations with a generic deep-learning approach is not a simple matter of running a standard model on the image stacks. This is essentially because capturing the defects requires using a large number of images, which contain much redundant information. Thus, training a deep network given only limited amounts of annotated data, which is the norm due to the scarcity of defective samples, results in overfitting. Theoretically, this can be addressed by various data augmentation techniques, including applying random photometric and geometric transformations such as brightness changes, translations, or rotations to the images. For convolutional neural networks (CNNs), rotation-based augmentations are particularly effective, as these networks are not rotation invariant. However, applying random rotations to the image stacks needlessly disrupts the connection between

✉ David Honzátko
david.honzatko@csem.ch; david.honzatko@epfl.ch

Engin Türetken
engin.tueretken@csem.ch

Siavash A. Bigdeli
siavash.bigdeli@csem.ch

L. Andrea Dunbar
andrea.dunbar@csem.ch

Pascal Fua
pascal.fua@epfl.ch

1 Centre Suisse d'Electronique et de Microtechnique (CSEM), Neuchâtel, Switzerland

2 École polytechnique fédérale de Lausanne (EPFL), Lausanne, Switzerland

Published online: 23 September 2021

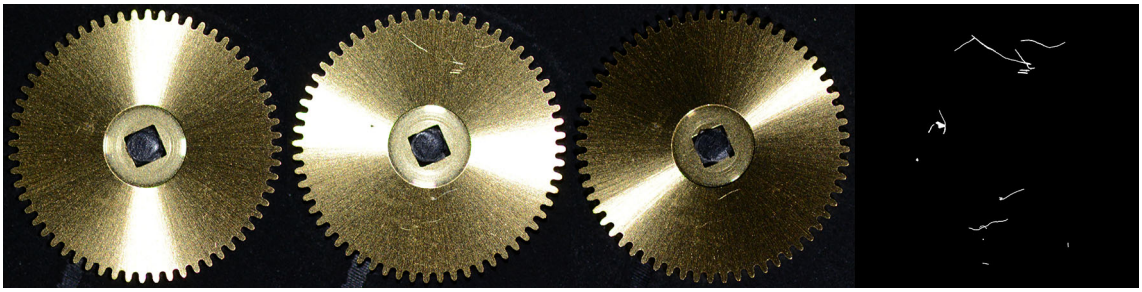


Fig. 1 Importance of illumination. Defective sample illuminated from different directions along with ground-truth binary mask representing the defects, showing that defect visibility varies strongly with illumination

the position of an image in the stack and the corresponding illumination setting and yields suboptimal results.

In this paper, we, therefore, introduce a novel data augmentation procedure that is specifically designed for multi-illumination image stacks, and that can handle an arbitrary number of illuminations. It relies on *illumination-preserving rotations* that amount to virtually acquiring physically rotated samples along the camera's principal axis. This involves first rotating all images by an integer multiple of the angle between two adjacent illuminators and then re-ordering them in the stack. Figure 2 illustrates this process.

We demonstrate the power and generality of our illumination-preserving rotations on three very distinct datasets that we acquired using a light-dome system [8], and that we will make public. Hence, our contribution is both a new approach to data augmentation and the release of these new datasets in an area that lacks them.

2 Related work

Detecting defects from images captured under different lighting conditions is an idea as old as the photometric stereo itself [28]. Thanks to its high speed, low costs, and effectiveness at capturing defects on various surfaces, multi-illumination systems are widely used today in the industry. Applications include detecting casting defects on steel and other metallic surfaces [7,9,11,12,21,26], cracks in paintings [4,16], connection defects in electronic circuit boards [23], or quality control of carbon-fiber materials [18]. Due to the widely varying reflectance properties of the materials involved, defect detection techniques vary substantially in terms of their algorithmic steps. Existing methods fall into three main categories depending on how they extract descriptive features from the input images.

Surface normals The first class of methods are those that first estimate the surface shape or surface normals [9,16,21]. The defects are then often defined as deviations from the expected surface normals, which often involves a single threshold, which makes these methods brittle.

Hand-crafted features A second class uses a hand-crafted set of image processing operations to produce feature maps and then use them for defect detection [7,11,18,19]. The operations used include edge-detection, thresholding, as well as various element-wise operations across the images such as maximum, subtraction, or division. The feature maps are then fed into a classifier or directly thresholded. It is almost universally accepted in the computer vision community that such hand-crafted features are less effective than those learned by deep networks when there is enough annotated data to train the networks. Hand-crafted methods are often overly specific for particular surfaces and defects and therefore cannot be reused for others.

Deep learning The third class includes methods that rely on convolutional neural networks and, hence, do not require any explicit pre-processing. The first such method [26], used only two images with different illuminations, which were stacked and fed to a very shallow network to classify small 16×16 image patches. Similarly, the method of [23] relies on two illuminations only, but it involves feeding each image individually to a Siamese network with shared weights. However, the experimental results presented in the paper demonstrate that it is better to process the two images together as much as possible. The approach of [4] works along the same lines. It accepts a six-image stack as input and uses a voting scheme to combine the output generated from each image. In this scheme, to balance the number of negative and positive samples, only pixels that are marked as edges by an edge-detector are considered. The only published end-to-end method [12] we know of that uses truly deep network stacks three illuminations in the RGB channels and passes it to the YOLO object detection network [20]. However, it does not provide details about the datasets, annotations, and training procedure. We could not find any method that uses more than six illuminations in the literature, even though it brings tangible benefits as we will show in the results section.

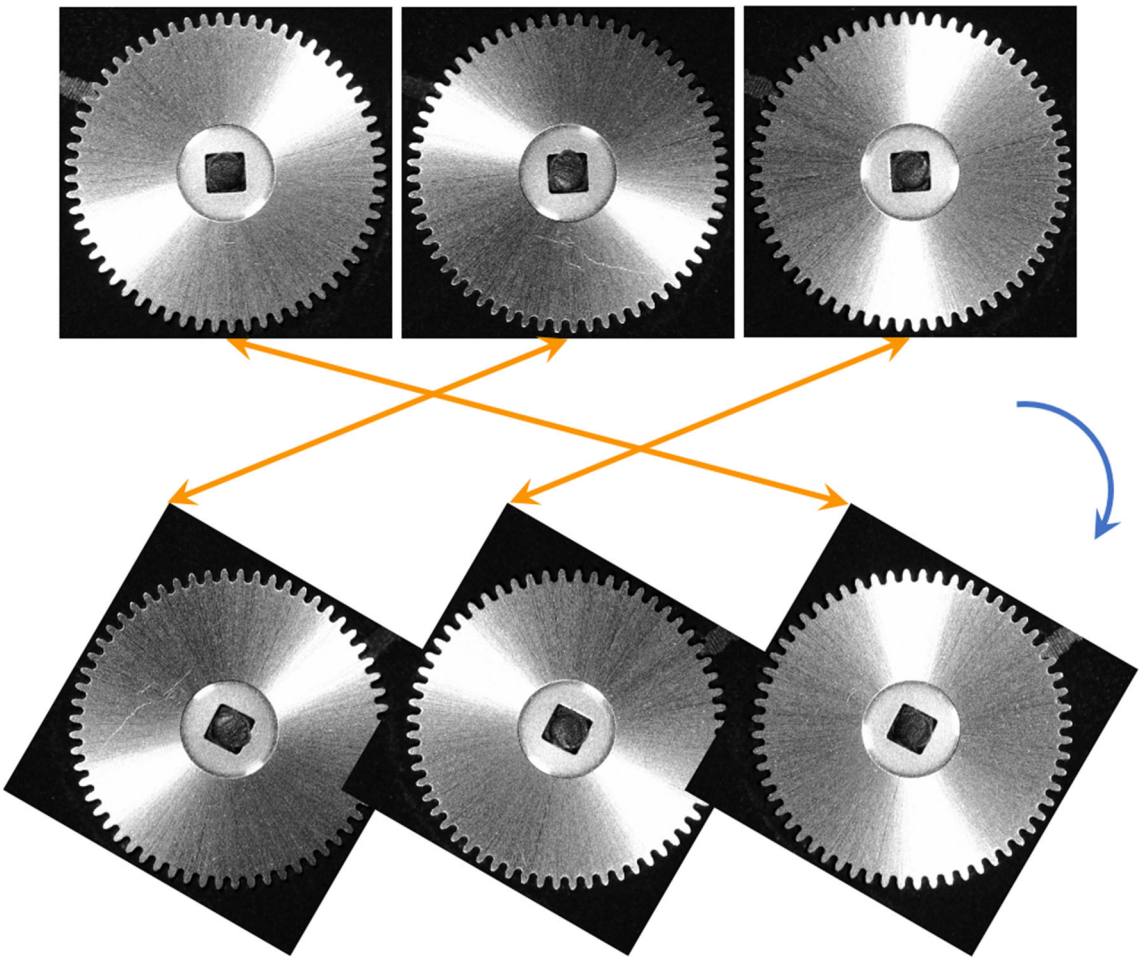


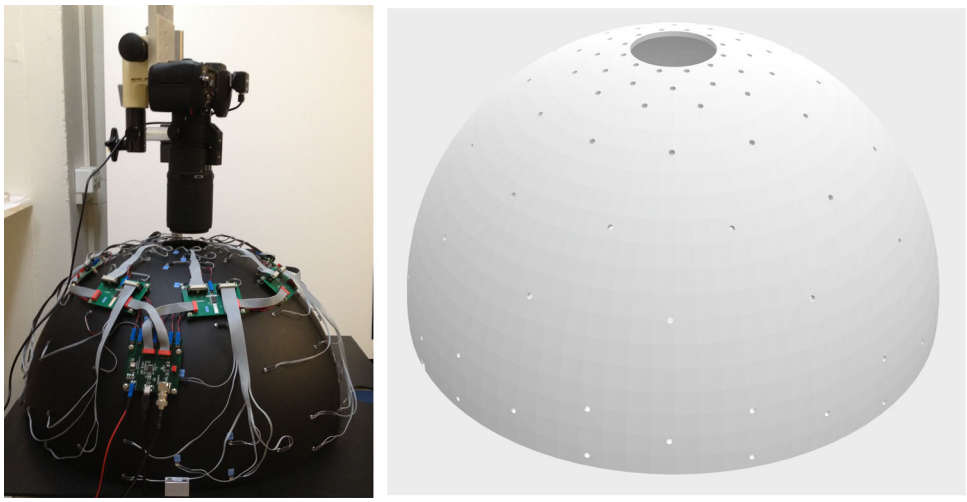
Fig. 2 Illumination-preserving rotations. In this example, we rotate a 3-image stack by 120°

Our defect segmentation method belongs to the third class. Similar to other end-to-end deep learning methods, it can adapt to different types of surfaces. However, unlike existing approaches does not depend on image size and can jointly process an arbitrarily large number of illuminations, which is required to capture all the relevant surface defects in industrial environments.

Due to the scarcity of the defects and the resulting lack of annotated data available for training, deep learning methods often suffer from overfitting to the training data, which results in poor generalization. Current methods often compensate for that by using data augmentation to enrich the training datasets. Standard augmentations used in computer vision [24] include color space transformations, noise injection, random erasing, conditional image generation, image mix-ups [5], or geometric transformations such as flipping, rotating, zooming, skewing, or wrapping. When it comes to multi-illumination defect detection, existing methods mainly rely on geometric transformations [23,26]. Unfortunately, some geometric transformations, such as rotations, may degrade instead of improve the detection performance in the multi-illumination setting because they disrupt the relationship between illumination angles and images. We propose a novel rotation-based augmentation technique that preserves this relation and thus, improves performance.

3 Method

We have used a simple, generic, hardware setup [8] that allows us to illuminate the sample from 108 different angles. It consists of a half-spherical *light-dome* depicted by Fig. 3a that filters out all the ambient light. A camera is placed on top of the dome, and 108 point-lights are laid out on the inner side of the dome forming nine circles with twelve equidistantly placed LEDs, each circle being centered around the camera's principal axis. The circles are placed at different elevation levels and slightly rotated in order to capture more than 12 azimuths. The LED placement is shown in Fig. 3b. This setup lets us acquire multiple images of a static object placed at the center under different lighting conditions in a time-multiplexed manner. For each image, only a single point-light is used to illuminate the sample. For the purpose of



(a) Prototype **(b)** LED positions

Fig. 3 Light dome system prototype used to capture a surface under different point-light illuminations

deep learning, we stack the images along the channel dimension to form a multi-illumination stack. In practice, despite acquiring RGB images, we only use the luminance channels as the defects are not color-related.

The cost of such a system is dominated by the price of the single camera and lens used. Compared to a stereo or a multi-camera defect detection system, it can therefore be produced at a substantially lower cost. With respect to timing, our system takes approximately a second to acquire an image but is not optimized for speed. This can be done to allow for batch processing [15], albeit it would still lead to system that is slightly slower than one that uses a single, fixed exposure.

Formally, let $\mathbf{I}$ be the image-stack acquired using the light-dome hardware setup. We can write it as:

$$\mathbf{I} = \left\{ I_{\theta,\phi} \middle| \theta \in \left[0, \frac{\pi}{2}\right), \phi \in [0, 2\pi) \right\}, \tag{1}$$

where ϕ denotes the azimuth angle and θ the elevation angle of a light source with respect to the intersection of the camera's principal axis and the surface of the part being inspected. For each value of θ, the set of azimuths ϕ is uniformly distributed in the interval $[0, 2\pi)$ with an angular step of α, as illustrated by the simplified setup of Fig. 4. For our hardware setup, $\alpha = 30°$ when using all 12 different azimuth values and more when using fewer.

Given such an input image stack, we aim to generate a binary image I_s in which the defects are highlighted. A naive approach would be to train a deep network to output I_s given $\mathbf{I}$. However, as for all deep learning algorithms, the problem becomes one of providing enough annotated data to properly train the network. Furthermore, in our case, images acquired with spatially nearby or directly opposite point-lights are very correlated, which reduces their usefulness for training purposes. In this section, we first describe our data augmentation procedure that we advocate to overcome this difficulty. We then introduce our deep learning architecture.

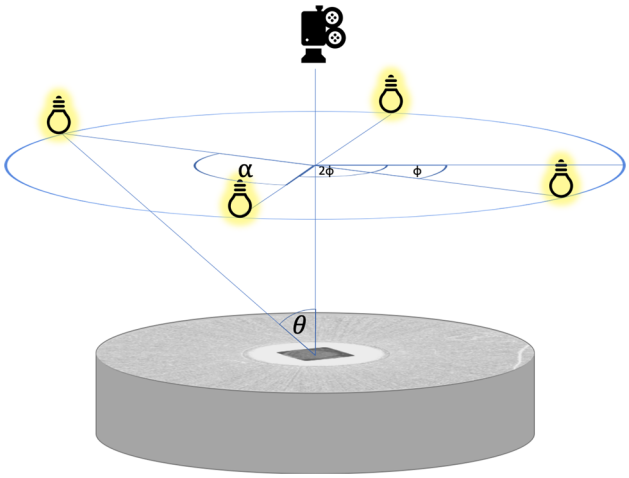


Fig. 4 Light and camera setup. At any given elevation θ, there is a set of uniformly distributed point-lights laying on a circle centered around the camera axis

3.1 Illumination-preserving rotations

Data augmentation is key to handling small datasets of high-dimensional samples, on which it considerably improves generalization performance. Its objective is to enlarge the dataset with new samples that are created from the existing ones by a set of random photometric and geometric transformations. To have a significant effect on training, these newly created samples should be considerably different from the existing ones, yet theoretically acquirable by the hardware setup used. For the datasets captured in a controlled environment, such as the one described earlier, the number of suitable transformations is therefore often very small.

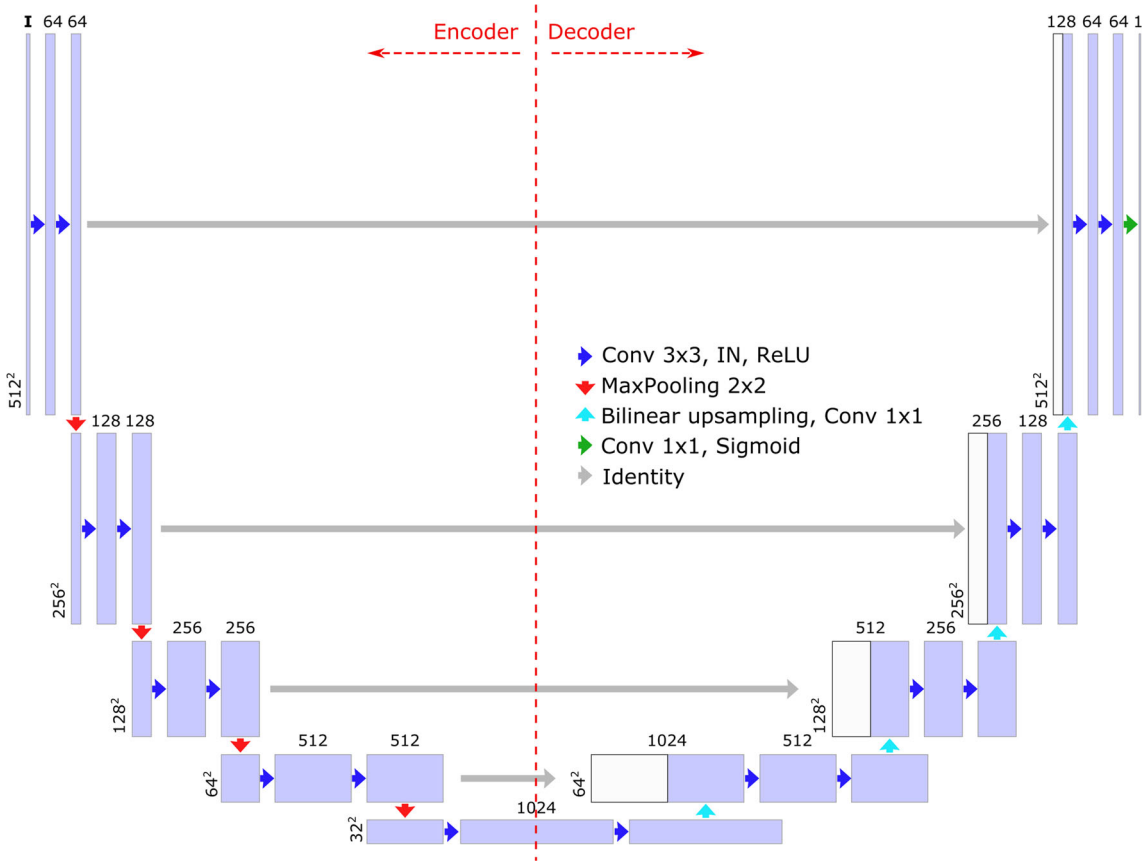


Fig. 5 Modified U-Net architecture. The decoder relies on bilinear upsampling instead of deconvolution

Rotation-based augmentations are generally very effective, particularly if the samples in a dataset do not have a fixed orientation. However, applying an unconstrained rotation to all images of the stack needlessly breaks the binding between the position of the image in the stack and the illumination angle used during the acquisition. A stack rotated in this way cannot actually be acquired by the same setup as the original stack, and therefore applying such augmentations during training harms the performance due to the sample domain mismatch between training and inference stages.

We address this problem by introducing *illumination-preserving rotations*. Given a stack of images of a sample, it produces a new stack as if it captured the same sample physically rotated in the acquisition device. We denote the illumination-preserving rotation function as $\mathcal{R}_{k\cdot\alpha}(\mathbf{I})$, which effectively rotates an image stack by an integer multiple k of angle α while preserving the relative ordering of images in the stack by permuting them. This is illustrated by Fig. 2.

More specifically, we first rotate each image $I_{\theta,\phi}$ individually by $k\cdot\alpha$ degrees, which generates a new stack where the indices ϕ no longer correspond to the illumination angles in the acquisition setup. To fix that, we then reorder the images in the stack. In other words, given an input stack $\mathbf{I}$, we create the rotated stack $\mathbf{I}' = \mathcal{R}_{k\cdot\alpha}(\mathbf{I})$ in which the individual images are obtained with

$$I'_{\theta,\phi} = R_{k\cdot\alpha}(I_{\theta,(\phi-k\cdot\alpha \mod 2\pi)}). \tag{2}$$

3.2 Deep learning architecture

Figure 5 depicts the neural network architecture we use. It is based on the U-Net [22], which is one of the most widely used architectures that has shown good performance on many fine image segmentation tasks. It consists of an encoder that compresses the input image stack $\mathbf{I}$ into a feature representation of low resolution, which allows the network to capture high-level features of a scene, and a decoder that given this feature representation predicts the segmentation mask I_s of the original resolution. To facilitate the segmentation of fine details, skip connections are introduced between blocks of layers of the same spatial resolution.

The encoder receives the input image stack $\mathbf{I}$ as a four-dimensional tensor, where the 3 dimensions stand for batch, width, and height, and the channel dimension contains individual gray-scaled observations. The encoder comprises four blocks of two convolutional layers followed by a max-pooling layer, and a final block of two convolutional layers. Inversely, the decoder is composed of four blocks of an

upsampling layer followed by two convolutional layers. The last layer of the decoder is a 1×1 convolutional layer followed by a sigmoid activation function that outputs a single-channel image I_s that can be interpreted as the likelihood of defects being present at individual spatial positions.

The skip connections are added between the encoder and decoder so that the output from the last convolution of each encoder block is concatenated with the output of the upsampling layer that has the same resolution. The first convolutional layer of the encoder is set to output 64 feature maps, and in each subsequent block the number of feature maps is doubled. Conversely, the number of output feature maps of each upsampling layer in each block of the decoder is halved with respect to its input. All the convolutional layers except the last one use kernels of size 3×3 and zero padding to preserve the resolution of the feature maps. They omit the bias term and are followed by ReLU nonlinearities. The max-pooling layers reduce the resolution by a factor of two while the upsampling ones increase it by the same factor.

The standard U-Net decoder often produces checkerboard artifacts in its predictions [17]. To eliminate them, we rely on bilinear upsampling followed by a 1×1 convolutional layer instead of deconvolutional upsampling. Furthermore, as image samples from industrial datasets do not significantly differ one from each other, we process the samples as a whole without cropping and with instance normalization layers instead of the commonly used batch normalization ones. Due to the memory constraints, such processing often considerably limits the batch size. Nevertheless, in Sect. 4.5.3, we empirically prove that it is a reasonable choice.

4 Experiments

In this section, we first introduce the datasets we use. We then present our training procedure and finally, our results.

4.1 Datasets

As there are no publicly available datasets for multi-illumination defect detection, we used the *light-dome* system [8] shown in Fig. 3a to acquire two datasets of parts used in high-end mechanical watches, a gear and a screw, and one dataset of flat washers. These datasets feature typical defects that occur in small-scale production industries, where inspection by humans still prevails. Even though all parts are metallic, their surface texture, and the nature of defects they exhibit, differ substantially.

We will refer to these datasets as **gears**, **screws**, and **washers**. They are depicted by Figs. 6, 7, and 8, and we will make them publicly available.[1] More specifically, the watch-parts datasets, **gears** and **screws**, comprise image stacks depicting 35 defective samples with corresponding hand-labeled binary masks representing where the defects are and 35 non-defective samples. The **washers** dataset comprises 70 defective samples with hand-labeled masks and no intact samples. For each sample, we also provide an automatically extracted foreground segmentation mask. For evaluation purposes of defect detection performance, we randomly split the datasets into training and test sets. The training set contains 16 defective and 16 intact samples for the **gears** and **screws** and 32 defective samples for the **washers**, with the remaining samples forming the test set. In addition to these labeled stacks, **gears** and **screws** also contain several hundreds of unlabeled samples that we do not use in this work but could be useful for future semi-supervised learning research.

In **gears**, the surface is relatively rough with a square hole for an axis in its center. The most common defects are scratches, while a few more others are caused by oil pollution. In the images of **screws**, the top surface of the samples is relatively smooth with a single indentation forming the slot. The defects are visibly much more diverse and can be scratches of different sizes or inclusions, among various other patterns. In **washers**, some defects, such as notches and holes, are visible in most images but in a different way each time, while others, such as scratches, are only visible in a few.

As most defects are thin elongated structures, downsampling images often leads to their disappearance. We quantify this in Fig. 9, where we plot the percentage of reduction in the defect coverage ratio with respect to the original image resolution after downsampling and binarization. To this end, we started from the original ground-truth segmentation masks of size 512×512 and progressively downsampled them after Gaussian smoothing. For all three datasets, the ratio decreases with resolution, but the effect is particularly strong in the case of **gears**, where after 8x downsampling the defect coverage ratio is almost 6x smaller. This can be attributed to the high percentage of long and thin scratch defects in this dataset.

In any event, only rarely can defects be detected from a single image, which is especially true for thin scratches. This makes the defect detection task on these datasets challenging, especially given that

- each stack consists of $512 \times 512 \times 108$ pixels (assuming only the luminance channels are used) and is therefore large;
- the metallic surfaces produce many specular reflections that sometimes saturate the camera sensors;
- the annotations are not very precise because the exact extent of defect contours is always subjective;
- the defects are very sparse also in the spatial dimensions: they cover only about 0.2% of the total image area in **gears**, 0.8% in **screws**, and 1.4% in **washers**;

[1] The dataset is available at https://doi.org/10.5281/zenodo.5513768

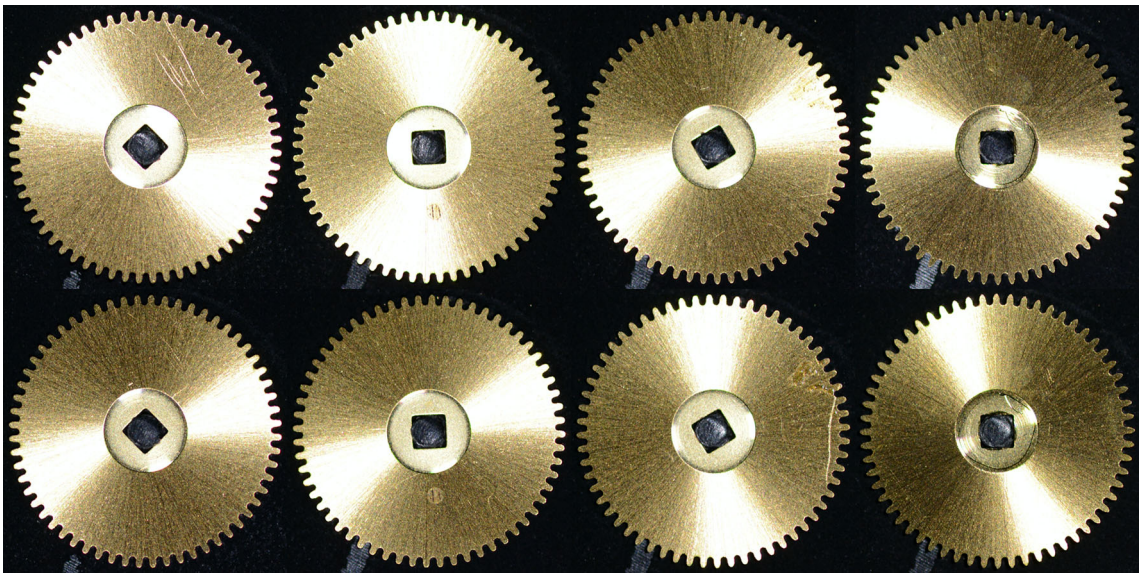


Fig. 6 Samples from the gears dataset. Each column depicts a single gear illuminated from two different angles

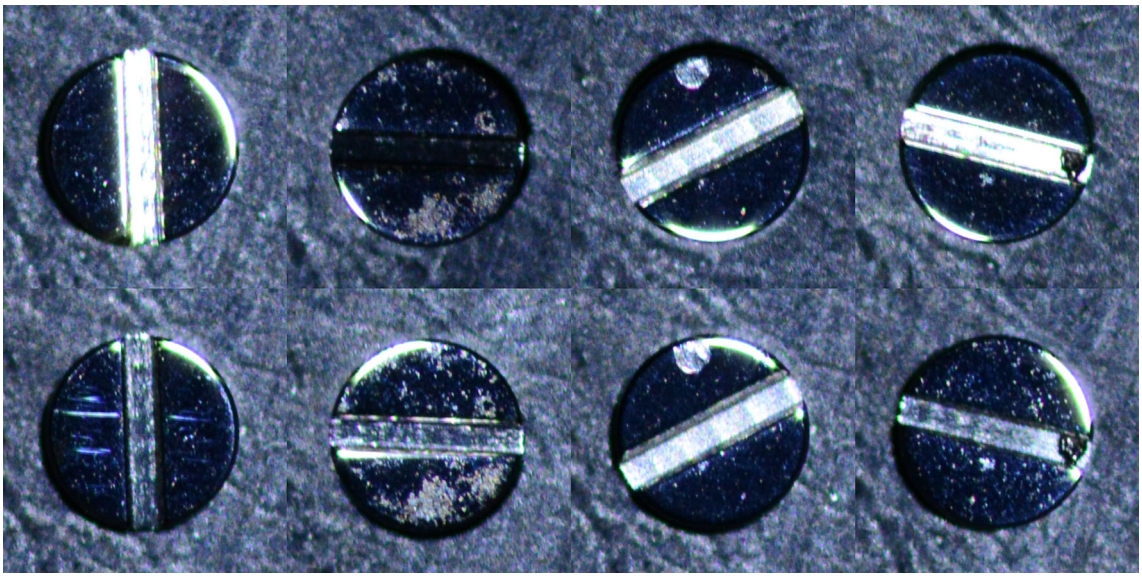


Fig. 7 Samples from the screws dataset. Each column depicts a single screw illuminated from two different angles

– many defects, such as scratches and notches, are thin elongated structures, and downscaling the images would erase them as shown in Fig. 9.

4.2 Training

To demonstrate the benefit of using illumination-preserving rotations, we train the network of Sect. 3.2 in three different ways:

No rotations We pass the training image stacks to the network without any data augmentation.

Random rotations On each iteration of the optimization, we augment the training stacks by rotating all the images in them, along with the segmentation masks, by a random angle between 0° and 360°. Also, with a 50% chance, we flip each training stack along the x-axis to double the training set size.

Illumination-preserving rotations We use the illumination preserving rotations of Sect. 3.1 to create new training stacks, as in Eq. 2. We take the rotation angle to a random integer multiple of $\alpha = (360/n_a)°$ when using n_a different azimuth values. The segmentation masks are rotated accordingly. This effectively increases the training set size $360/\alpha$ times. We take $n_a = 12$, which results in 30° rotations and effectively

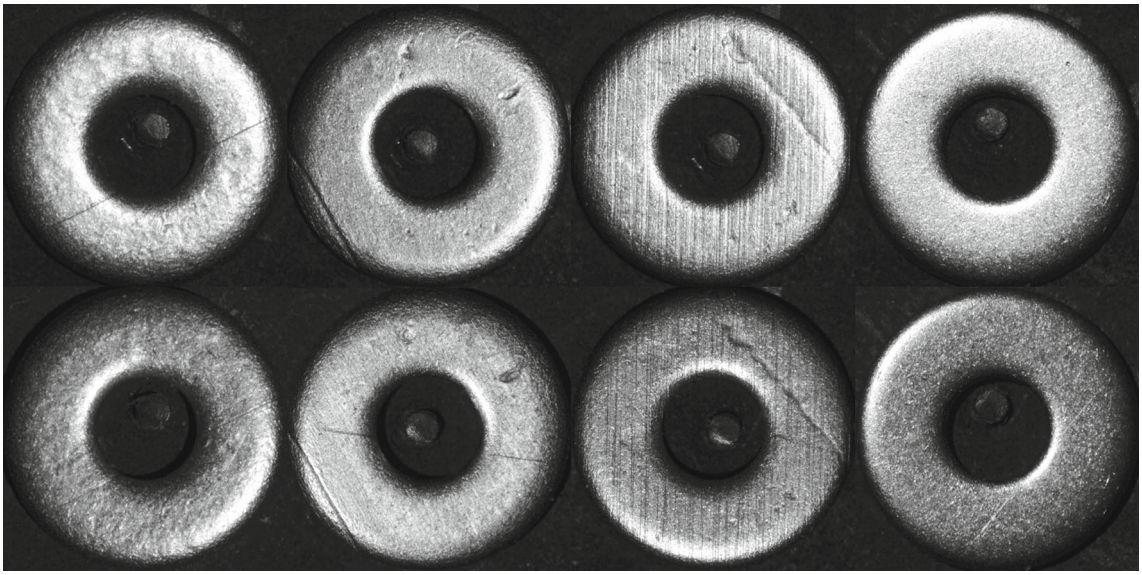


Fig. 8 Samples from the washers dataset. Each column depicts a single washer illuminated from two different angles. The variability between samples is higher than in the other datasets. For example, the rough surface in the first column and the striations in the third column are not defects, while the scratch and the notch are

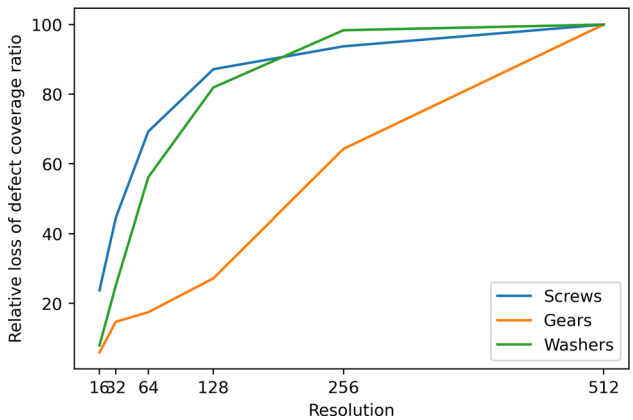


Fig. 9 Defect percentages. For each one of our three datasets, we plot the average percentage of pixels that belong to defects as a function of the image resolution we use, which ranges from 16×16 to 512×512. The percentages are reported with respect to the defect coverage ratio at the original resolution of 512×512

increases the number of training samples from 32 to 384 image stacks.

Random 30° rotations For a complete comparison, we also train the networks with the standard random rotation augmentation but where we limit the set of allowed angles to the multiples of 30° as in the case of the illumination-preserving ones.

In all four cases, we train the networks to minimize the cross-entropy loss between the predicted and the ground-truth segmentation masks using the Adam [10] optimizer with a learning rate of 0.001 and batch size of two. We train the networks for 5000 epochs, and we ramp down the learning rate during the last 30% of training using a cosine schedule. Similarly, we ramp up the learning rate during the first 10% of training.

Even though the environment is controlled, there can be slight variations in the brightness and position of the samples. Therefore, during training, we always augment the training data by adding random brightness changes by up to 2% and small shifts by up to 5 pixels. The shifts are effective because, although we use networks whose convolutional layers are shift-invariant, their pooling layers may not be.

4.3 Metric

We evaluate the performance of different techniques in terms of the area under the precision-recall curve (AuPRC), also known as the average precision. PR curve plots the trade-off between precision and recall over different decision thresholds. The area under this curve, therefore, does not depend on a specific threshold value, which is especially beneficial for defect segmentation, where a cumulation of uncertain defects still might have large importance. This metric is particularly useful for heavily imbalanced datasets, which is usually the case in defect segmentation since defective pixels occupy only a small fraction of the image area.

More specifically, let the TP_n, FP_n, and FN_n be the counts of true positives, false positives, and false negatives after thresholding the predictions with the threshold n. An area under the precision-recall curve can be then estimated from the precision $P_n = TP_n/(TP_n + FP_n)$ and recall $R_n = TP_n/(TP_n + FN_n)$ at every threshold $n \in [0, 1]$ as

$$\text{AuPRC} = \sum_{n\in[0,1]} (R_n - R_{n-1})\, P_n \ . \tag{3}$$

In practice, we use 200 thresholds uniformly distributed between 0 and 1. For a good estimate from a discrete set of thresholds, we interpolate between the precision-recall points as in [3], which is the default to the TensorFlow library.

4.4 Results

To reduce the effect of random initialization on performance, all AuPRC values reported in this section are computed as the mean over two independent runs. To keep computational requirements under control, we used only the 24 illuminations with the highest elevation angles for these experiments. In the following section, we will present an ablation study to show that this is enough to deliver close to the best possible performance on our datasets.

4.4.1 Varying approaches to regularization

When training a network, care must be taken to avoid overfitting to the training set. This is typically done by regularizing. To show that the influence of our illumination-preserving rotations is largely independent of the specific approach to regularization being used, we experimented with several.

No regularization As the U-Net is a very large network for a small dataset consisting of 70 annotated stacks, half defective and half intact, regularization is expected to be beneficial irrespective of the augmentation method used. Nevertheless, we show results without it for completeness sake.

L1 weight decay One of the most standard regularization techniques is to use an L1 weight decay, that is, adding to the loss term the λ multiple of L1 norm of all the convolutional kernels and biases. In our experiments, we set $\lambda = 1e-5$.

Spatial dropout During training, after each convolutional layer, we randomly drop out an entire feature channel with the probability p [27]. As the datasets are small in the number of samples, we use a relatively large p-value of 0.6 for **washers** and **gears** and 0.7 for **screws**.

Early stopping Overfitting can be prevented by stopping the training when the best performance is reached on the validation set. To fairly evaluate this approach, we put aside four defective and four intact samples from the training set to form the validation set. We allowed the network to optimize for 1000 epochs, instead of 5000, and selected the network weights that yield the best evaluation score on the validation set. We only evaluated the model on the validation set every 20 epochs.

In Table 1, we report AuPRC numbers as a function of the regularization strategy used to train the modified U-Net of Sect. 3.2. For **gears** and **washers**, our illumination preserving rotations consistently and markedly improve performance, no matter what regularization strategy is used. By contrast, random rotations often do not improve or even degrade performance, presumably due to interpolation-related distortion. This is also true for **screws** except for early stopping and spatial dropout, where 30° random rotations do well. However, in this specific case, the difference between random rotations and our approach does not appear to be statistically significant. Overall, the illumination preserving rotations used in conjunction with L1 decay yield the best result.

4.4.2 Varying network architectures

The U-Net is a well-known architecture but by no means the only one that can be used for defect detection. We, therefore, tested additional ones listed in the following:

Small network In Sect. 4.4.1, we prevented overfitting using different regularization strategies. Another approach is to substantially reduce the capacity of the network. We use a *small* fully convolutional network consisting of 3 convolutional layers with 18 feature channels in between them, instance normalization, and ReLU activation functions.

IterNet The U-Net architecture has been consistently improved for various tasks. One such successful modification is IterNet [13] that was designed for retinal vessel segmentation with just tens of annotated samples necessary for training. It iteratively refines by passing them through a set of 3 mini-U-Nets that all share the same weights. The original U-Net has 4 blocks, while the mini-U-Nets have just two.

FCN The Fully Convolutional Network (FCN) [14] was introduced the same year as U-Net and demonstrated outstanding performance on natural image segmentation tasks such as PASCAL VOC [6]. It uses a VGG16 backbone network pre-trained on the ImageNet dataset [25]. To use it on our multi-illumination image stacks, we changed the size of the kernel of the first convolutions and replicated the pre-trained weights along the channel dimension.

DeepLabv3+ The FCN network has been gradually refined over the years. According to the PASCAL VOC benchmark, one of the most successful follow-ups is DeepLabV3+ [1] that uses a modified Xception [2] backbone network, also pre-trained on ImageNet. We modified it in the same way as FCN to use it on our multi-illumination stacks.

Table 1 AuPRC evaluation for various approaches to regularizing U-Nets

Dataset Network architecture	Rotations			
	None	Random	30°	Illum. pres. 30°
Gears				
No regularization	0.23 *(0.02)*	0.19 *(0.00)*	0.27 *(0.01)*	**0.32** *(0.01)*
Early-stopping	0.38 *(0.01)*	0.24 *(0.07)*	0.26 *(0.15)*	***0.46*** *(0.02)*
L1 weight decay	0.28 *(0.03)*	0.20 *(0.02)*	0.27 *(0.03)*	**0.40** *(0.05)*
Spatial dropout	0.33 *(0.01)*	0.14 *(0.01)*	0.26 *(0.01)*	**0.40** *(0.00)*
Washers				
No regularization	0.24 *(0.01)*	0.30 *(0.01)*	0.36 *(0.00)*	**0.37** *(0.01)*
Early-stopping	0.32 *(0.07)*	0.32 *(0.01)*	0.48 *(0.05)*	**0.50** *(0.00)*
L1 weight decay	0.29 *(0.01)*	0.39 *(0.02)*	0.50 *(0.01)*	***0.54*** *(0.03)*
Spatial dropout	0.26 *(0.01)*	0.37 *(0.01)*	0.36 *(0.01)*	**0.50** *(0.01)*
Screws				
No regularization	0.30 *(0.03)*	0.26 *(0.03)*	0.40 *(0.02)*	**0.45** *(0.01)*
Early stopping	0.44 *(0.01)*	0.48 *(0.02)*	**0.59** *(0.05)*	0.54 *(0.14)*
L1 weight decay	0.30 *(0.01)*	0.25 *(0.07)*	0.54 *(0.00)*	***0.65*** *(0.01)*
Spatial dropout	0.30 *(0.01)*	0.62 *(0.01)*	**0.63** *(0.00)*	0.62 *(0.00)*

We provide the mean and standard deviation (in brackets). The best result corresponding to any given regularization approach is boldfaced. The overall best result for each dataset is shown in bolditalic

We report our AuPRC results in Table 2. As before, our approach to data augmentation delivers the highest boost for all the architectures, except for FCN on **gears** for which all methods fail. This is presumably because **gears** mostly features thin elongated scratches, for which FCN is not well suited due to the lack of shortcut connections from high-resolution activations of the first few layers to the last few ones in the architecture. As a result, it is relatively ineffective at modeling high-frequency image content.

4.5 Ablation study

We now justify the various algorithmic choices we made in the above experiments.

4.5.1 Number of illuminations

Table 1 was built using only 24 of the 108 available illuminations to keep down the computational requirements. We now use the L1 decay approach with the illumination-preserving rotations, that is the one that yields the best performance, to show that using 24 illumination angles is a reasonable choice for our datasets.

As shown in Fig. 10, going from 2 to 6 to 24 images yields substantial improvements as additional illuminations naturally reveal defects that are only visible under a more narrow range of illumination angles. In that respect, we have a distinct advantage over the earlier methods discussed in the related work section that only use between 2 to 6 images. Nevertheless, increasing the number of images over 24 has a much more limited impact. Therefore, for our datasets, 24 represents a good compromise between speed and performance.

4.5.2 Dimensionality reduction

As discussed earlier, one of the main reasons why data augmentation is needed is that the image stacks exhibit a large degree of intra-channel redundancy. A natural approach to this problem is to eliminate this redundancy by dimensionality reduction. In this section, we show that this is *not* as effective as data augmentation.

One of the simplest ways to reduce the dimensionality of the stacks is to use per-pixel statistics over the images in it, such as mean, standard deviation, or skewness. This makes it easy to use *all* 108 illuminations instead of only 24. In Table 3, we, therefore, report results using 24 as in Sect. 4.4 and all 108 illuminations. We report results for the L1 regularized U-Net and for all the described types of rotation-based augmentations. We report the random 30° rotations and illumination preserving rotations jointly as the image ordering does not play a role if a dimensionality reduction is applied.

For **screws**, the results are close to those we obtained in Sect. 4.4, but they are significantly worse for **gears** and **washers**. This can be attributed to the fact that these statistical measures are not able to capture many defects, as illustrated by Fig. 11. For all datasets, we also observe significantly better results with random rotations. This stems from the fact that the per-pixel statistics used contain less light orientation-related information, and hence, do not depend on the rotation

Table 2 AuPRC evaluation for various architectures

Dataset Network architecture	Rotations			
	None	Random	30°	Illum. pres. 30°
Gears				
U-Net (L1 decay)	0.28 *(0.03)*	0.20 *(0.02)*	0.27 *(0.03)*	**0.40** *(0.05)*
Small	0.24 *(0.04)*	0.10 *(0.01)*	0.19 *(0.01)*	**0.33** *(0.01)*
IterNet	0.21 *(0.00)*	0.29 *(0.02)*	0.16 (***0.23***)	***0.44*** *(0.02)*
FCN	0.08 *(0.03)*	0.06 *(0.00)*	0.09 *(0.00)*	0.09 *(0.00)*
DeepLabV3+	0.17 *(0.04)*	0.29 *(0.01)*	0.32 *(0.01)*	**0.34** *(0.00)*
Washers				
U-Net (L1 decay)	0.29 *(0.01)*	0.39 *(0.02)*	0.50 *(0.01)*	***0.54*** *(0.03)*
Small	0.30 *(0.00)*	0.31 *(0.01)*	0.35 *(0.02)*	**0.38** *(0.00)*
IterNet	0.16 *(0.03)*	0.18 *(0.04)*	0.33 *(0.04)*	**0.37** *(0.00)*
FCN	0.18 *(0.02)*	0.21 *(0.05)*	0.34 *(0.01)*	**0.39** *(0.00)*
DeepLabV3+	0.17 *(0.00)*	0.37 *(0.01)*	0.40 *(0.02)*	**0.45** *(0.02)*
Screws				
U-Net (L1 decay)	0.30 *(0.01)*	0.25 *(0.07)*	0.54 *(0.00)*	***0.65*** *(0.01)*
Small	0.30 *(0.04)*	0.23 *(0.05)*	0.34 *(0.00)*	**0.52** *(0.02)*
IterNet	0.18 *(0.03)*	0.22 *(0.01)*	0.46 *(0.02)*	**0.55** *(0.00)*
FCN	0.34 *(0.01)*	0.34 *(0.06)*	0.56 *(0.03)*	**0.62** *(0.00)*
DeepLabV3+	0.17 *(0.00)*	0.45 *(0.02)*	**0.56** *(0.00)*	0.55 *(0.00)*

We provide the mean and standard deviation (in brackets). The best result corresponding to any given architecture is boldfaced. The overall best result is shown in bolditalic

of the physical sample in the light-dome. Random rotations are, however, more prone to interpolation-related problems.

For completeness, we tested another classic approach to dimensionality reduction that has been extensively used for defect detection [9,16,21]: We estimated the surface normals and albedo at each pixel in standard photometric stereo fashion [29] assuming a Lambertian shading model.

We also report results in the same manner as in the previous case. The only exception to this is that when using a rotation augmentation, we also rotate normal directions to simulate the physical rotation of the sample in the light-dome. At the bottom of Table 3, we show that for **gears** and **screws**, with or without the rotations, this photometric stereo approach performs worse than all the others. This is attributable to the Lambertian shading assumption, which is clearly not warranted as the defects are highly specular and mostly without shadows. For **screws**, nonetheless, we see that random rotations improve over the illumination-preserving ones as the surface properties are completely orientation independent. We do not observe this effect for **gears**, where the detection failed completely. This is likely to be due to the increased complexity of the task that was unsolvable for the heavily regularized U-Net. For **washers**, the Lambertian approach is the best technique when used with standard random rotations. This dataset has more matte surfaces and deeper defects, which makes them better visible in the estimated normal maps. Nevertheless, the performance gain achieved using illumination preserving rotations keeps the Lambertian approach behind.

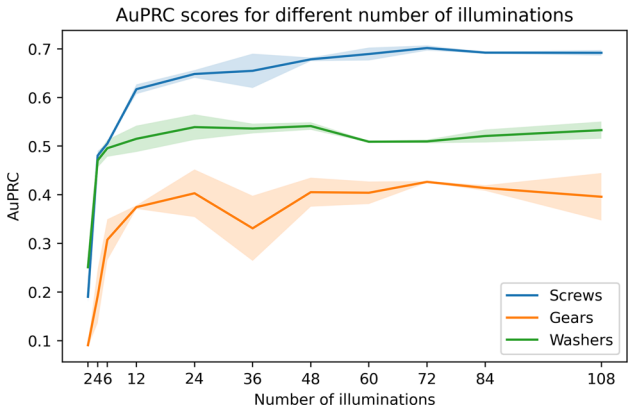


Fig. 10 Illumination versus performance. Effect of increasing the number of illuminations on the defect detection performance

4.5.3 Working in image patches

As stated in Sect. 3.2, we do not crop our images. Hence, our image stacks are large, and we have to use a relatively small batch size to avoid overwhelming the GPU memory. Here, we explore the alternative approach of using image crops and larger batch sizes.

Table 3 AuPRC evaluation for dimensionality reduction using either 24 or 108 illuminations

Input type	# illums.	Rotations		
		None	Random	Illum. pres. 30°/ 30°
Gears				
No reduction	24	0.28	0.20	***0.40***
No reduction	108	**0.33**	**0.22**	***0.40***
Mean + Std	24	0.13	0.20	0.23
Mean + Std	108	0.14	0.19	0.27
Mean + Std + Skew	108	0.16	0.13	0.21
Lambertian	108	0.21	0.02	0.20
Washers				
No reduction	24	0.29	0.39	***0.54***
No reduction	108	**0.32**	0.38	0.53
Mean + Std	24	0.17	0.35	0.33
Mean + Std	108	0.10	0.26	0.24
Mean + Std + Skew	108	0.10	0.24	0.26
Lambertian	108	0.29	**0.49**	0.50
Screws				
No reduction	24	0.30	0.25	0.65
No reduction	108	0.41	0.40	***0.69***
Mean + Std	24	0.40	0.60	0.60
Mean + Std	108	0.46	**0.65**	0.68
Mean + Std + Skew	108	**0.48**	0.63	0.68
Lambertian	108	0.38	0.52	0.33

For each dataset, three groups of techniques are evaluated: from top to bottom, the technique of Sect. 4.4 without dimensionality reduction, dimensionality reduction using per-pixel statistics, and dimensionality reduction using surface normals under a Lambertian assumption. Best results corresponding to any given augmentation technique are boldfaced, and the best result overall is shown in bolditalic

During training, we crop the images to size 128×128, which allows us to increase the batch size to 32. We also switch from using instance normalization (IN) layers to batch normalization (BN) ones as the statistics of individual crops might differ significantly.

In Fig. 12, we show the test cross-entropy losses during training. Similar to the previous experiments, we use U-Net regularized by the L1 weight decay. Both approaches converge to similar results with a slight advantage for small patches but at the cost of significant instability of the training and longer optimization time. Therefore, we chose to proceed without patching as it is simpler and faster.

An explanation might stem from the fact that while the sample statistics do not differ significantly among the samples, they can differ significantly among different crops of samples. Although this effect is partially attenuated by using a larger batch size, it is impossible to completely avoid extreme cases where the batch statistics largely differ from the expected ones. Such batches are then poorly normalized, which may lead to more unstable training.

4.5.4 Training set size

Effective augmentation techniques improve classification performance even on small datasets. To demonstrate this for our technique, we performed an ablation study of the training set size with the best performing method (L1 regularization). For each experiment, we randomly sampled the same number of intact and defective samples out of the original training set to have a roughly balanced dataset. Furthermore, to show the consistency of the results across different random samplings, we run four experiments for train sizes of 4 and 8, and two experiments for train sizes of 16 and 32, each with a different training set.

Figure 13 gives the mean AuPRC results for the two datasets. For **gears**, reducing the train size has little influence on the defect detection performance with illumination-preserving rotations, whereas the performance drops rapidly with standard random rotations or no augmentations. For **screws**, on the other hand, all techniques are adversely affected by the reduced training set size. Nevertheless, augmentation by illumination-preserving-rotations is consistently superior for all the cases.

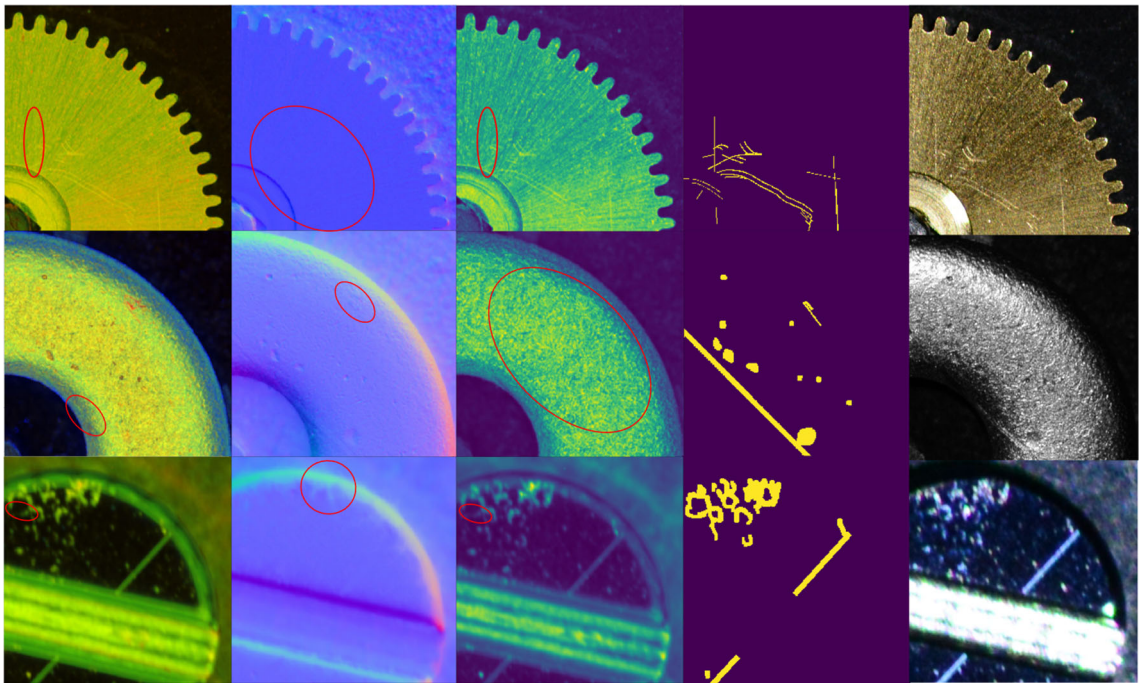


Fig. 11 Defect (in)visibility in features obtained by two dimensionality reduction techniques. From left to right: mean, std, and skew over images, Lambertian normals, Lambertian albedo, ground-truth mask, one of the source images. Problematic areas are circled red (colour figure online)

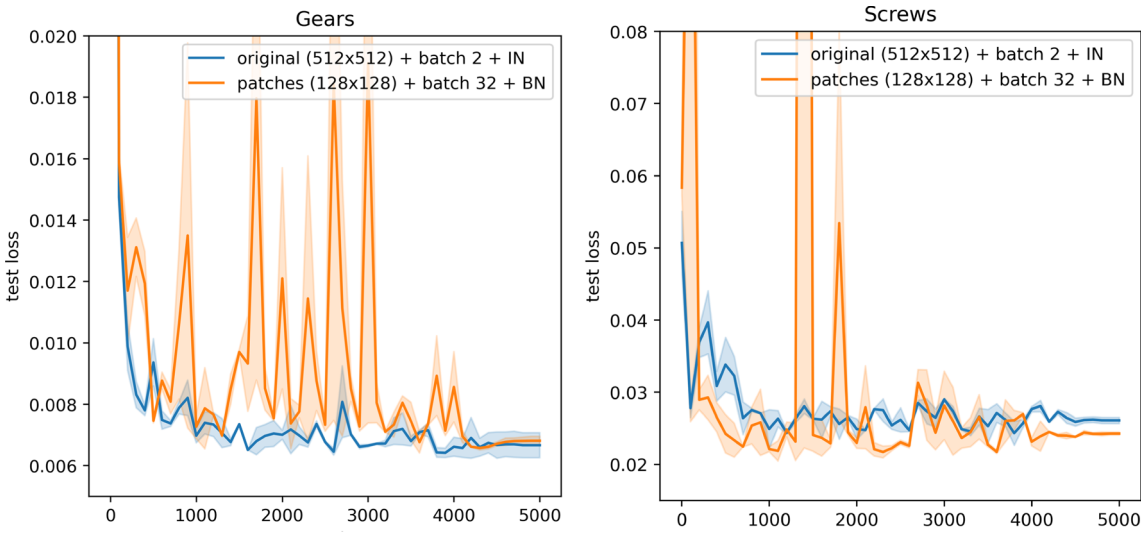


Fig. 12 Influence of patching. Cross-entropy test loss as a function of the number of epochs when using or not using patches

5 Conclusion

We presented an end-to-end deep learning approach to the defect segmentation problem that can make use of an arbitrary number of images of a sample captured under different illumination angles. Our main contribution is the introduction of a novel rotation-based augmentation technique, *illumination-preserving rotations*, which effectively copes with the high dimensionality and informational redundancy of such data. We have shown that, on datasets with a small number of annotated samples, our augmentation technique is, compared to the standard ones, more effective in reducing the overfitting and consistently improves the performance for all the tested network architectures and regularization strategies.

To this end, we have used a light-dome system capable of capturing a sample under 108 different illumination angles to acquire three real-world defect segmentation datasets. While similar setups are widely used in industry, no adapted dataset was publicly available. We hope that releasing one

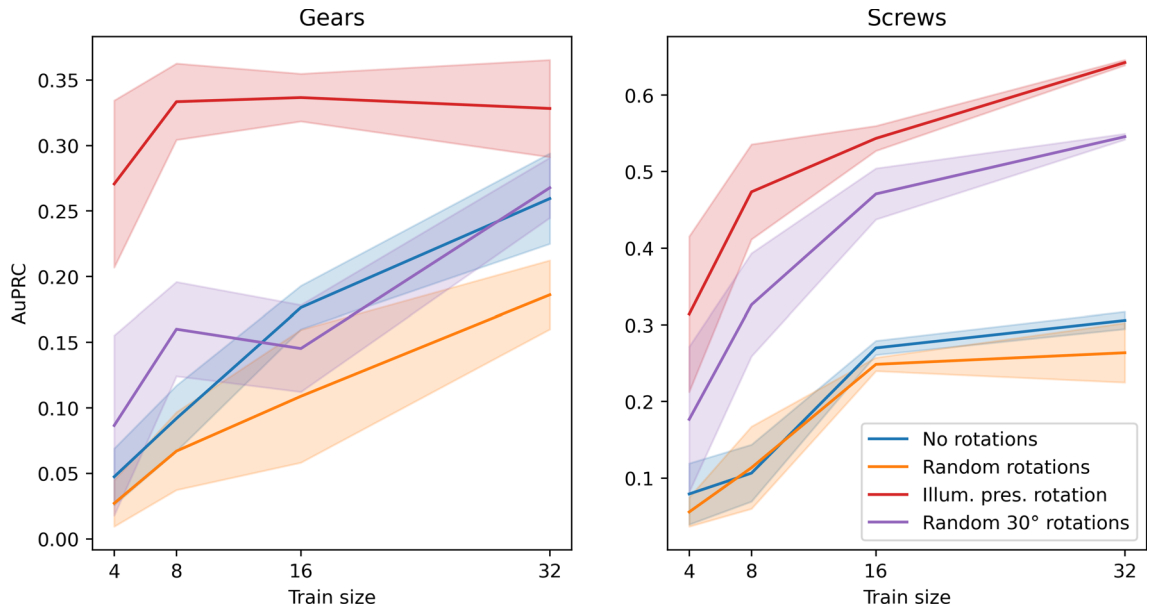


Fig. 13 Influence of the training set size. Effect of lowering the training set size on the defect detection performance measured as the AuPRC

will boost interest in both supervised and unsupervised multi-illumination defect segmentation, and that the newly developed methods bring deep learning quality control to small and middle-size production lines.

We have shown that using many different illuminations, end-to-end training, and the proposed augmentation all contribute to effective defect detection. This is in contrast with most published works in the field that either use a small number of illuminations or rely on feature engineering to characterize the surface properties of the samples. However, the performance boost that multi-illumination imaging depends on the surface material of the samples. While highly effective for surfaces whose appearance change drastically under different illumination conditions, its benefits are less pronounced when dealing with predominantly matte surfaces.

Data-augmentation significantly reduces the number of annotated samples required for training. However, recent progress in unsupervised and self-supervised learning suggests that we can reduce the required amount of annotated data even further, which might increase the range of possible applications and reduce the cost of the solution. In future work, we, therefore, intend to disentangle the effects of illumination and surface properties in an unsupervised manner using a better regularized encoder-decoder architecture. This should not only allow us to leverage the task-specific information residing in a large number of unlabeled samples in our datasets but also reduce the dimensionality of our samples without the ill-effects we observed in our ablation study.

Acknowledgements The acknowledgements will be filled at the time of acceptance.

Funding The authors did not receive support from any organization for the submitted work.

Availability of data and materials The datasets will be made public at the time of acceptance.

Declarations

Conflict of interest The authors have no relevant financial or non-financial interests to disclose.

Code availability The source code is available at https://github.com/DawyD/illumination-preserving-rotations; The dataset is available at: https://doi.org/10.5281/zenodo.5513768

References

1. Chen, L.C., Zhu, Y., Papandreou, G., Schroff, F., Adam, H.: Encoder-decoder with atrous separable convolution for semantic image segmentation. In: Proceedings of the European Conference on Computer Vision (ECCV), pp. 801–818 (2018)

2. Chollet, F.: Xception: deep learning with depthwise separable convolutions. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 1251–1258 (2017)
3. Davis, J., Goadrich, M.: The relationship between precision-recall and roc curves. In: Proceedings of the 23rd International Conference on Machine Learning, pp. 233–240 (2006)
4. Dulecha, T., Giachetti, A., Pintus, R., Ciortan, I.M., Jaspe Villanueva, A., Gobbetti, E.: Crack detection in single- and multi-light images of painted surfaces using convolutional neural networks. In: Eurographics Workshop on Graphics and Cultural Heritage (2019)
5. Eaton-Rosen, Z., Bragman, F., Ourselin, S., Cardoso, M.J.: Improving data augmentation for medical image segmentation (2018)
6. Everingham, M., Van Gool, L., Williams, C.K.I., Winn, J., Zisserman, A.: The pascal visual object classes (VOC) challenge. Int. J. Comput. Vis. **88**(2), 303–338 (2010)
7. Galan, U., Ahuett-Garza, H., Orta, P., Kurfess, T.: Shadow identification and height estimation of defects by direct processing of grayscale images. Int. J. Adv. Manuf. Technol. **101**(1–4), 317–328 (2019)
8. Hasler, D., Chebira, A., Basset, G., Beuchat, P.A.: Method and apparatus for detection of visible defects. European Patent, EP 2 887 055 B1 (2016)
9. Kang, D., Jang, Y.J., Won, S.: Development of an inspection system for planar steel surface using multispectral photometric stereo. Opt. Eng. **52**(3), 039701 (2013)
10. Kingma, D.P., Ba, J.: Adam: a method for stochastic optimization (2014). arXiv preprint arXiv:1412.6980
11. Landstrom, A., Thurley, M.J., Jonsson, H.: Sub-millimeter crack detection in casted steel using color photometric stereo. In: 2013 International Conference on Digital Image Computing: Techniques and Applications (DICTA), pp. 1–7. IEEE (2013)
12. Lee, J.H., Oh, H.M., Kim, M.Y.: Deep learning based 3D defect detection system using photometric stereo illumination. In: 2019 International Conference on Artificial Intelligence in Information and Communication (ICAIIC), pp. 484–487. IEEE (2019)
13. Li, L., Verma, M., Nakashima, Y., Nagahara, H., Kawasaki, R.: Iternet: Retinal image segmentation utilizing structural redundancy in vessel networks. In: Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision, pp. 3656–3665 (2020)
14. Long, J., Shelhamer, E., Darrell, T.: Fully convolutional networks for semantic segmentation. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 3431–3440 (2015)
15. Lumitrax^{TM} lighting. https://www.keyence.com/ss/products/vision/vj/lumitrax.jsp
16. Manfredi, M., Bearman, G., Williamson, G., Kronkright, D., Doehne, E., Jacobs, M., Marengo, E.: A new quantitative method for the non-invasive documentation of morphological damage in paintings using RTI surface normals. Sensors (Switzerland) **14**(7), 12271–12284 (2014). https://doi.org/10.3390/s140712271
17. Odena, A., Dumoulin, V., Olah, C.: Deconvolution and checkerboard artifacts. Distill (2016). https://doi.org/10.23915/distill.00003
18. Palfinger, W., Thumfart, S., Eitzinger, C.: Photometric stereo on carbon fiber surfaces. In: 35th Workshop of the Austrian Association for Pattern Recognition (2011)
19. Pang, G.K., Chu, M.H.: Automated optical inspection of solder paste based on 2.5 d visual images. In: 2009 International Conference on Mechatronics and Automation (ICMA), pp. 982–987. IEEE (2009). https://doi.org/10.1109/ICMA.2009.5246351
20. Redmon, J., Divvala, S., Girshick, R., Farhadi, A.: You only look once: unified, real-time object detection. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 779–788 (2016). https://doi.org/10.1109/CVPR.2016.91
21. Ren, M., Wang, X., Xiao, G., Chen, M., Fu, L.: Fast defect inspection based on data-driven photometric stereo. IEEE Trans. Instrum. Meas. **68**(4), 1148–1156 (2018)
22. Ronneberger, O., Fischer, P., Brox, T.: U-net: convolutional networks for biomedical image segmentation. In: International Conference on Medical Image Computing and Computer-Assisted Intervention, pp. 234–241. Springer (2015)
23. Shiina, T., Iwahori, Y., Kijsirikul, B.: Defect classification of electronic circuit board using multi-input convolutional neural network. Int. J. Comput. Softw. Eng. **3**(2), 2456–4451 (2018). https://doi.org/10.15344/2456-4451/2018/137
24. Shorten, C., Khoshgoftaar, T.M.: A survey on image data augmentation for deep learning. J. Big Data **6**(1), 1–48 (2019)
25. Simonyan, K., Zisserman, A.: Very deep convolutional networks for large-scale image recognition. In: Bengio, Y., LeCun, Y. (eds.) Proceedings of the 3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7–9, 2015 (2015)
26. Soukup, D., Huber-Mörk, R.: Convolutional neural networks for steel surface defect detection from photometric stereo images. In: International Symposium on Visual Computing, pp. 668–677. Springer (2014)
27. Tompson, J., Goroshin, R., Jain, A., LeCun, Y., Bregler, C.: Efficient object localization using convolutional networks. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 648–656 (2015)
28. Woodham, R.J.: Reflectance map techniques for analyzing surface defects in metal castings (1978)
29. Woodham, R.J.: Photometric method for determining surface orientation from multiple images. Opt. Eng. **19**(1), 191139 (1980). https://doi.org/10.1117/12.7972479

David Honzátko is a PhD student at Edge AI & Vision group at CSEM in Switzerland and Computer Vision Laboratory at EPFL. He received a M.Sc. degree in Computer Science at Charles University in Prague. His research interests include computer vision and deep learning with a focus on image processing and photometric stereo.

Dr. Engin Türetken received his PhD in Computer Science in 2013 from EPFL, Switzerland. His research interests include computer vision, deep learning, algorithm theory, embedded programming, and microscopic image analysis. He is currently a machine learning expert in the Edge AI & Vision group at CSEM and is leading various research projects in the fields of human-machine interaction, environmental understanding, and event detection using modern vision and artificial intelligence approaches.

Siavash A. Bigdeli is a data scientist at CSEM, Switzerland. He received a PhD in Computer Science from the University of Bern in 2018. Prior to joining CSEM in 2019, he was a postdoctoral fellow at EPFL. His interests are computer vision, deep learning, and particularly statistical signal processing.

L. Andrea Dunbar is head of the Edge AI & Vision group at CSEM in Switzerland and lectures at the EPFL on digitalization. She received her PhD in Physics at Trinity College, Dublin, Ireland, in non-linear optics. Her team currently works on machine learning, intelligent vision systems, hyperspectral imaging system, and ultra-low-power imagers and embedded intelligence for highly constrained environments such as for IoT applications.

Pascal Fua is a Professor of Computer Science at EPFL, Switzerland, and heads the Computer Vision Laboratory. His research interests include shape and motion reconstruction from images, analysis of microscopy images, and augmented reality. He has (co)authored over 300 publications in refereed journals and conferences. He is an IEEE Fellow and has been an Associate Editor of the IEEE journal Transactions for Pattern Analysis and Machine Intelligence.