



Aluminum Casting Inspection using Deep Object Detection Methods and Simulated Ellipsoidal Defects

Domingo Mery¹

Received: 27 August 2020 / Revised: 12 January 2021 / Accepted: 12 March 2021 / Published online: 16 April 2021
© The Author(s), under exclusive licence to Springer-Verlag GmbH Germany, part of Springer Nature 2021

Abstract

In the automotive industry, light-alloy aluminum castings are an important element for determining roadworthiness. X-ray testing with computer vision is used during automated inspections of aluminum castings to identify defects inside of the test object that are not visible to the naked eye. In this article, we evaluate eight state-of-the-art deep object detection methods (based on YOLO, RetinaNet, and EfficientDet) that are used to detect aluminum casting defects. We propose a training strategy that uses a low number of defect-free X-ray images of castings with superimposition of simulated defects (avoiding manual annotations). The proposed solution is simple, effective, and fast. In our experiments, the YOLOv5s object detector was trained in just 2.5 h, and the performance achieved on the testing dataset (with only real defects) was very high (average precision was 0.90 and the F_1 factor was 0.91). This method can process 90 X-ray images per second, i.e. this solution can be used to help human operators conduct real-time inspections. The code and datasets used in this paper have been uploaded to a public repository for future studies. It is clear that deep learning-based methods will be used more by the aluminum castings industry in the coming years due to their high level of effectiveness. This paper offers an academic contribution to such efforts.

Keywords Object detection · Aluminum inspection · X-ray testing · Deep learning

1 Introduction

In the automotive industry, light alloy aluminum castings (e.g. wheels, knuckles, gear boxes, etc.) are an important element for determining roadworthiness. During the production process, heterogeneous parts can be formed inside the workpiece. This manifests itself, for example, as cracks, bubbles, slags, or inclusions. In the quality control of aluminum castings, every detail must be thoroughly checked using X-rays, and 100% of the parts must be reviewed. The goal of X-ray testing of castings is to identify discontinuities (defects) located inside the test object that are not visible to the naked eye (see, for example, Fig. 1).

Over the past 35 years, several computer vision methods for conducting automated inspections of castings have been described with many promising results. Automated X-ray systems have improved quality through multiple objective inspections and improved processes and increased productivity and consistency by reducing labor costs [41].

The trend today in computer vision -in general- is to use methods based on deep learning. Deep learning has been established as the state-of-the-art in many areas of computer vision. The key idea of deep learning is to replace *handcrafted* features with features that are *learned* efficiently using a hierarchical feature extraction approach [2,22,23,41]. Usually, the learned features are so discriminative that no sophisticated classifiers are required. In recent years, we have witnessed tremendous improvements in many fields of computer vision that involve using complex deep neural network architectures trained with thousands or millions of images (e.g. facial recognition [7], object recognition and detection [31,63], diagnosis of prostate cancer [44], classification of skin cancer [9], among others). Methods based on deep learning have become fundamental in these fields. Nevertheless,

This work was supported in part by Fondecyt Grant 1191131 from CONICYT-Chile.

✉ Domingo Mery
domingo.mery@uc.cl
http://domingomery.ing.uc.cl

¹ Department of Computer Science, Pontificia Universidad Católica de Chile, Santiago, Chile

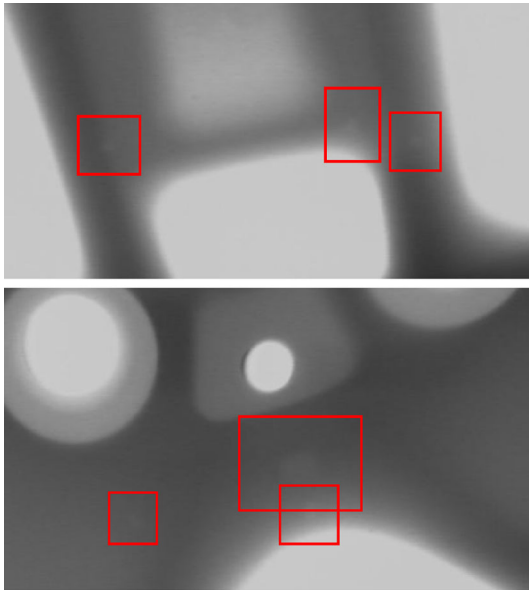


Fig. 1 X-ray images of a casting with real defects (images of series C0001 of GDXray dataset [42])

the use of deep learning is still limited in aluminum casting inspection.

In comparison with other computer vision applications, the introduction of techniques based on deep learning in computer vision for X-ray testing in industrial applications has been rather slow. We believe that there are three reasons for this. (i)The first has to do with the availability of public databases that can be used for these purposes. While in some areas of computer vision (e.g. facial recognition), hundreds of databases have been created since the 1990s, there is only one public database for X-ray testing for castings inspection, GDXray [42]. This database was created five years ago and contains around 2700 X-ray images. The rest of the datasets used in the experiments reported by the industry and academia are private. (ii)The second reason is related to the number of experts working in this field. While almost anyone can be an expert in other areas of computer vision (such as object recognition), in non-destructive testing of castings, the relative number of people working on these subjects is rather low and their work is usually expensive. In this kind of computer vision application, experts must be hired to label the data (make annotations, define bounding boxes, etc.). It is very simple to find people who can detect cats and dogs in photographs, but not so easy to find human operators who can identify the discontinuities in a casting by inspecting an X-ray image. (iii)The last reason is that in other applications of computer vision, color photographs can be acquired using inexpensive equipment (often a cell phone), whereas in X-ray testing we need expensive equipment. It is likely that fewer people are working on X-ray images than color images for these reasons.

This research simplifies the use of deep learning in the inspection of castings. It is our hope that these techniques can be used easily and effectively in the quality control of die castings in the near future. Our contributions are fourfold:

1. We developed a simple, effective, and fast deep learning strategy that can be used in the inspection of aluminum castings. The training stage requires a relatively small number of X-ray images and no manual annotations because a simulation model is used to superimpose defects onto the X-ray images. In our experiments, the training for the model was completed in just 2.5 h; in the testing dataset (with real defects), performance was very high (average precision was around 0.90 and the F_1 factor was 0.91), and the computational time is very low (only 11 ms per image i.e. it can be used in real-time inspection to aid human operators).
2. We propose a training/testing strategy in which defects that are used in the training are not used in testing. Due to the low number of real defects, in many cases, a defect can be captured in different X-ray images from different points of view. Thus, it is very common to use some captures of the defect in the training dataset and other captures of the same defect in the testing dataset. This practice, which could lead to overfitting, is avoided in our work.
3. We use well-established deep object detection methods with a good level of maturity in computer vision. That means that there are many examples available in public repositories that have already been successfully tested and evaluated. The use of these methods in aluminum inspection does not pose practical problems at the training or testing stages. Moreover, training and testing can be executed in Python in any browser with no intricate configuration and free access to GPUs using Google Colab¹.
4. The code and datasets used in this paper are available in a public repository². We believe that this practice, which is very common in other fields of computer vision, should be more common in X-ray testing. Thus, anyone can (i)reproduce the results reported in this paper and (ii)re-use the code in other inspection tasks.

In the field of computer vision, modern object-detection methods (like YOLO [50] and RetinaNet [28]) have been developed over the past five years with very promising results, but it has been very difficult to use them to identify defects in aluminum castings because there is not enough data for training. In order to overcome this problem, synthetic data were used in our work. Deep learning models trained with

¹ <https://colab.research.google.com>.

² <https://github.com/ Domingomery/defect-detection> → will be public after publication.

synthetic data is nothing new (see, for example, [53,57,58]). However, to the best of our knowledge, this is the first time that modern object-detection methods have been applied to the inspection of aluminum castings using simulated defects as training data. We tested ten object-detection models: eight modern object-detection models because they are the best performing and most representative deep learning-based object detection models in computer vision, as stated in [63], and two baseline classic models based on handcrafted features [34] and convolutional neural networks [37] in conjunction with a sliding windows strategy for comparison proposes. As we show in our experiments, it is clear that the modern strategies significantly outperform the classic ones.

2 Related work

In this Section, we cover the most important progress that has been made in the automated detection of defects in aluminum castings. The review is divided into two sections: the first (Sect. 2.1) is dedicated to the specific methods used in the inspection of aluminum castings. The second (Sect. 2.2) is related to computer vision methods that deal with object detection.

2.1 Defect detection in castings

Over the past 35 years, the literature has described various computer vision methods applied to the automated detection of casting defects. The first contributions were likely made in the 1980s [4,12]. Today, we can identify four different families of methods that are being used by industry or academia: (i) classic methods, (ii) methods based on multiple views, (iii) methods based on computed tomography, and (iv) methods based on deep learning. They are summarized in Table 1.

• **Classic Methods:** These methods correspond to approaches based on classic image processing and pattern recognition techniques. In these approaches, *handcrafted features* are used for the automated inspection of castings. This family of methods consists of two main groups [35]:

- Reference methods: In reference methods, still images must be taken from select inspection positions. A test image is then compared to the reference image. If a significant difference is identified, the test piece is classified as defective.
- Methods without a priori knowledge of the structure: These approaches use pattern recognition, expert systems, artificial neural networks, or general filters to make them independent of the position and structure of the test piece. An example of these methods is given in our experiments.

The fundamental disadvantages of the first group of methods include the complexity of their configuration and inflexibility to changes in the design of the workpiece. However, they are much more effective than the second group because the automated adaptive processes used to accommodate design modifications are far from perfect.

• **Methods based on multiple views:** Over the past two decades, approaches based on multiple views have been proposed because they can be very effective when examining complex objects where the uncertainty of only one view can lead to misinterpretation. These approaches typically have two main steps:

- First, potential defects are identified in each view. Handcrafted or learned features can be used in this effort.
- Second, the potential defects are matched and tracked across multiple views.

The key idea of this approach is to consider potential defects which cannot be tracked to be false alarms.

• **Methods based on computed tomography:** In contrast to the rest of the methods, computed tomography produces a volumetric reconstruction of the test object: a 3D volume, i.e. ,a set of 2D images of slices of the object under test is estimated using reconstruction approaches. Computed tomography can be a very time intensive process requiring a minimum measurement time for adequate signal to noise ratios as well as a minimum number of projections for the desired local resolution.

• **Methods based on deep learning:** The key idea of deep learning is to replace *handcrafted* features with features that are *learned* efficiently using a hierarchical approach. This family of methods has made contributions to defect detection in terms of object classification and object detection, as seen in Table 1.

Multiple views and computed tomography are rarely used to inspect castings. It is clear that in the coming years, deep learning-based methods will be used more frequently by the industry due to their high level of effectiveness. This paper uses state-of-the-art object detection methods with deep learning to offer an academic contribution to such efforts.

2.2 Computer vision for object detection

In computer vision, we distinguish between *image classification* and *image detection*. The purpose of image classification is to assign an X-ray image to one class. Image classification is typically used when there is only one object per image to be recognized. In the inspection of castings, we use this approach when a small sub-image of an X-ray image, i.e. ,a patch of 32 × 32 pixels as reported in [38], is to be classified as ‘defective’ or ‘not defective.’ In image classification

Table 1 Published methods of defect recognition on castings

References	Multi-views	Deep learning	Description
<i>Classic methods</i>			
Cogranne and Reira [6]	□	□	Statistical hypothesis testing using nonparametric tests
Hernández et al. [17]	□	□	Methods based on neuro-fuzzy approaches
Jin et al. [20]	□	□	HOG and LBP features classified using Adaboost-SVM
Kamalakaran et al. [21]	□	□	Image processing based on spatial based segmentation
Li et al. [26]	□	□	Wavelet technique
Li et al. [25]	□	□	Peak location algorithm combined with neural networks
Mery [34]	□	□	Features based on crossing line profiles
Mery and Arteta [38]	□	□	Comparative evaluation of 24 computer vision methods
Ramirez and Allende [47]	□	□	Generative and discriminative approaches
Tang et al. [55]	□	□	Segmentation using fuzzy model
Zhang et al. [60]	□	□	Image processing based on adaptive thresholding
Zhao et al. [61]	□	□	Statistical feature based on grayscale arranging pairs
Zhao et al. [62]	□	□	Sparse representations
<i>Methods based on multiple views</i>			
Carrasco and Mery [5]	⊗	□	Multiple view correspondence with non-calibrated model
Mery and Filbert [39]	⊗	□	Multiple view model with calibrated model
Mery et al. [43]	⊗	□	Multiple view approach using 3D features
Mery [36]	⊗	□	Multiple views using an un-calibrated tracking approach
Pieringer and Mery [45]	⊗	□	3D model from 2D images
Pizarro et al. [46]	⊗	□	Multiple views based on affine transformation
<i>Methods based on computed tomography (CT)</i>			
Bandara et al. [1]	⊗	□	Inspection using dual-energy
Hangai et al. [15]	⊗	□	Detection of shrinkage pores using microfocus CT
Li et al. [24]	⊗	□	Inspection of Thixomolding Mg castings
<i>Methods based on deep learning</i>			
Du et al. [8]	□	⊗	Object detection using deep learning (FPN and R-CNN)
Ferguson et al. [10]	□	⊗	Deep learning: Faster R-CNN architecture
Ferguson et al. [11]	□	⊗	Instance segmentation using transfer learning
Hu and Wang [18]	□	⊗	CNN with attention mechanism
Lin et al. [27]	□	⊗	Inter-frame deep convolution neural network strategy
Mery and Arteta [38]	□	⊗	Comparative evaluation of 24 computer vision methods
Mery [37]	□	⊗	CNN with sliding windows
Ren et al. [51]	□	⊗	Deep learning method based on auto-encoders
Tang et al. [56]	□	⊗	Deep learning method including spatial attention
Yong et al. [59]	□	⊗	Classification based on deep learning features

using deep learning, the input image is fed into a convolutional neural network (CNN) that classifies the input image. In image detection, more than one object can be recognized in an X-ray image and the location of each recognized object is identified using a bounding box that encloses the detected object (see an example in Fig. 2).

A simple strategy that uses the sliding-window methodology has been utilized for image detection based on image classification. In this approach, a detection window is placed over an input image in both horizontal and

vertical directions. For each localization of the detection window, a classifier (e.g. a CNN) decides which class the corresponding portion of the image belongs to based on its representation. An example of this approach for defect detection in aluminum castings is given in [37] and in our experiments. It is worthwhile to mention that this approach requires the classification of huge number of patches. In addition, if the size of the objects to be detected varies, the sliding-windows approach must be performed for different patch-sizes, which may make the computational time

prohibitive. New approaches that avoid this problem have been developed over the past few years. We address them in this Section. They can be subdivided into two groups [19]: (i)detection in two stages, and (ii)detection in one stage. In many object detection experiments, the methods of the second group outperform those of the first in terms of both accuracy and speed [50,63]. As such, we will only provide an overview of the first methods in this paper.

2.2.1 Detection in two stages

The detection methods that use two stages are called *region-based methods*. The first stage is the region proposal and the second is the final classification (of the proposed regions).

In the region proposal, a method is used to identify regions of the input image that may contain an object. In the sliding-windows method explained above, this step involves an exhaustive search. However, there are other methods, e.g. R-CNN [14] that propose regions instead of analyzing all possible matches for the input image.

In the final classification, a CNN is used to classify the regions that have been proposed through the first step.

The most relevant methods that use two-stage detection are R-CNN [14], Fast R-CNN [13], and Faster R-CNN [52]. For a more detailed explanation of the use of these methods in X-ray testing, see [41].

2.3 Detection in one stage

In these approaches, a single CNN is trained on both location and classification, i.e. prediction of bounding boxes and estimation of the class probabilities of the detected bounding boxes. This group of approaches is the state-of-the-art in detection methods because they are very effective and very fast. They are the best performing and most representative deep learning-based object detection models, as stated in [63]. In this Section, we address the most representative methods, namely, YOLO [3,48–50]³, EfficientDet [54], and RetinaNet [28]. We offer a brief description of these detection models and their principal differences. We pay special attention to YOLO because it performed best in our work.

• **YOLO:** In regard to region-based approaches, as explained in Sect. 2.2.1, object detection is performed in two stages: region proposal and final classification. This means that the classification is not performed by *looking* at the complete image, but by viewing selected regions of the image. In order to overcome this disadvantage, a new method called YOLO, *You-Only-Look-Once* was proposed [48]. YOLO is a single (and powerful) convolutional neural network that *looks* the image once, i.e. the input image is fed into a single CNN and

the output is the simultaneous prediction of both the bounding boxes (localization) and the category probabilities (classification) of the detected objects. It is very fast because the input image is processed in a single pass by the CNN. Over the past few years, many versions of YOLO have been developed: YOLOv1 [48], YOLOv2 [49], YOLOv3 [50], YOLOv4 [3] and YOLOv5³. They use different approaches in terms of subdivision, scales, anchors, and architectures to improve performance. The main strategy of YOLO is described below.

The main idea behind YOLO is very simple: The input image is divided into a grid of $S \times S$ cells, and YOLO can detect B objects for each cell. For each detected bounding box, YOLO computes:

- (x, y, w, h) : variables that define the detected bounding box, i.e. location (x, y) and dimension (width, height),
- p : confidence score that gives the probability that the bounding box contains an object ($P(\text{Object})$),
- p_i : for $i = 1 \cdots K$: probability distribution over all K possible classes, i.e. p_i is a conditional class probability ($P(\text{Class}_i|\text{Object})$).

This means that for each bounding box, YOLO provides an array of $R = 4 + 1 + K$ elements: $(x, y, w, h, p, p_1 \cdots p_K)$, as illustrated in Fig. 2⁴. At the testing stage, an object of class i is detected if $P(\text{Object}) \times P(\text{Class}_i|\text{Object})$ is greater than a confidence threshold. Since B bounding boxes can be detected in a grid cell, an array of $Q = B \times R$ elements is computed for each cell. The simplicity of YOLO (see Fig. 2) is based on the fact that (i)the architecture has only standard convolution layers with 3×3 kernels and max-pooling layers with 2×2 kernels, and (ii)the output of the CNN is a tensor of $S \times S \times Q$. This means that we have $5 + K$ elements per bounding box for each grid cell that give us information about the localization of the bounding box and the category probability.

• **RetinaNet:** RetinaNet architecture [28] is another new object detection model. It combines the pyramidal feature extraction structure [29] with a residual architecture (ResNet) [16] that has yielded promising results for image classification. The pyramidal approach consists of decreasing the size of the image several times and making predictions for each of those sizes. Another novelty of this architecture is the shift from cross-entropy to a ‘focal loss’-based objective that reduces the penalty for well classified classes while punishing mis-classifications more aggressively.

• **EfficientDet:** In EfficientDet [54], improvements to the architecture design are performed and analyzed: (i)BIFPN uses a weighted bidirectional feature pyramid network to fuse fast multi-scale features. (ii)A scaling method is used

³ YOLOv5 was released in June 2020. See <https://github.com/ultralytics/yolov5>.

⁴ In our experiments, $K = 1$ because there is only one class to detect: ‘defects.’

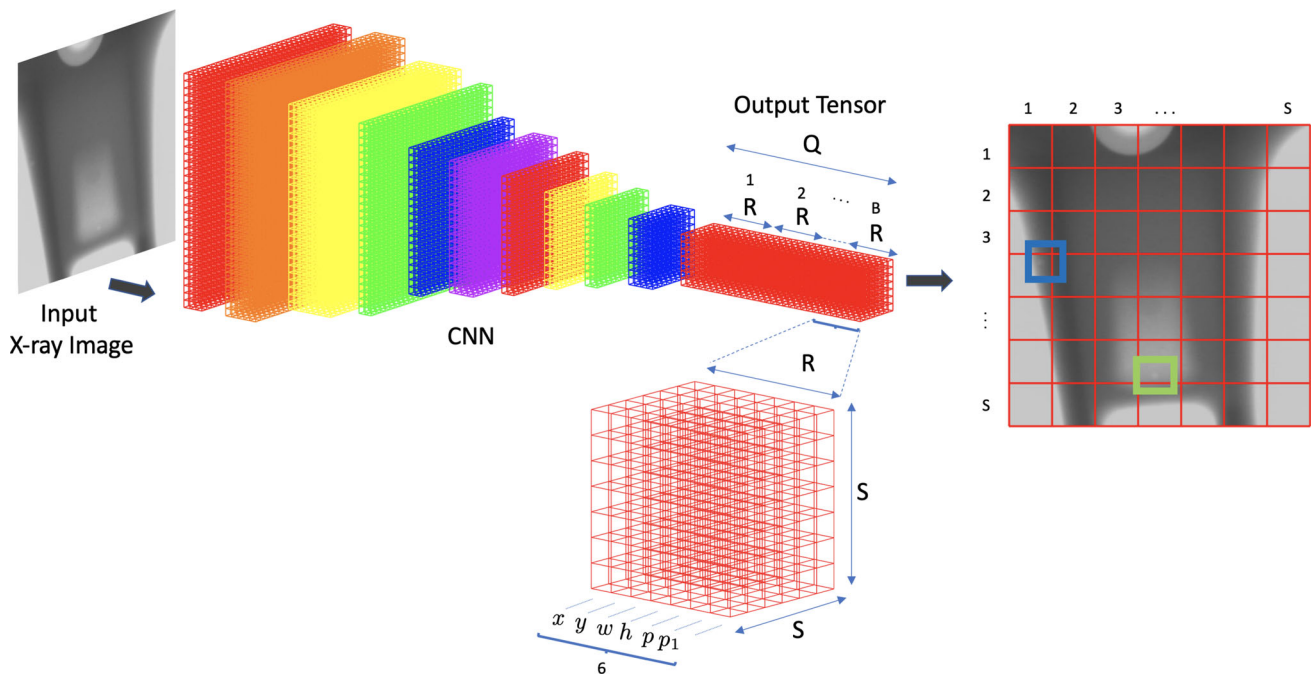


Fig. 2 YOLO strategy for object detection

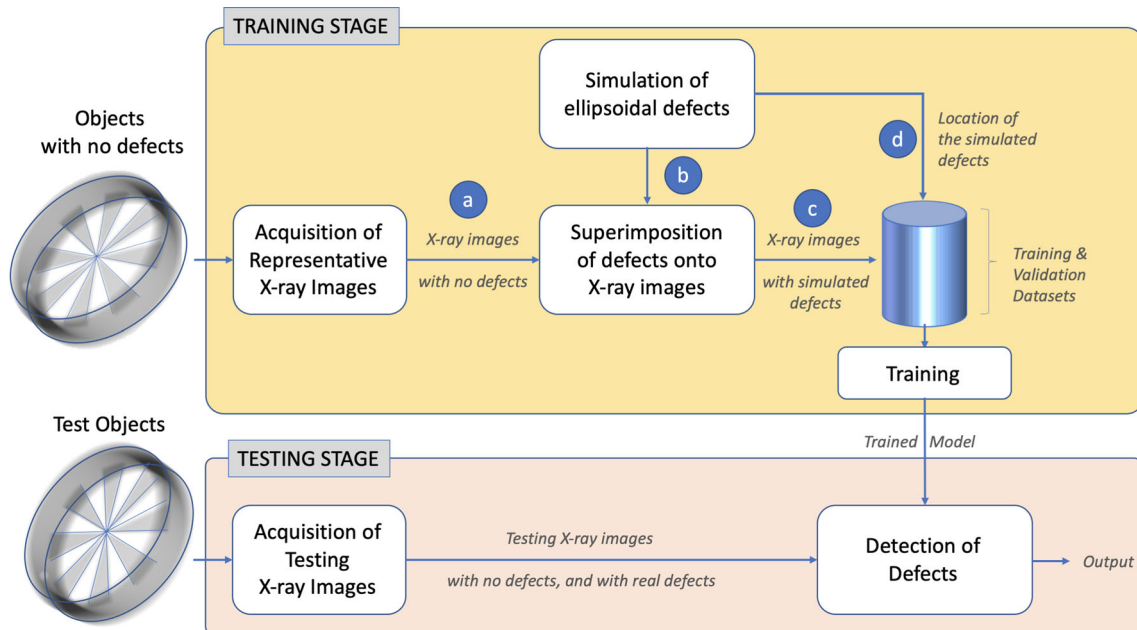


Fig. 3 Block diagram of the proposed method. See examples of images a, b, c, and d in Fig. 4

to scale depth, width, resolution and prediction networks simultaneously. This method achieves better performance than RetinaNet and YOLOv3 in terms of accuracy and speed in the COCO dataset⁵.

⁵ The COCO dataset is a well-known object recognition dataset that contains complex images of common objects (in a natural context) [30].

2.4 Overview

An overview of our proposed method is presented in Fig. 3. The recognition approach has two stages: training and testing. In our method, training is performed using real X-ray images of aluminum castings with simulated defects only (see Fig. 4). Thus, no real defect is used for training purposes because the number of real defects is very low. Testing is

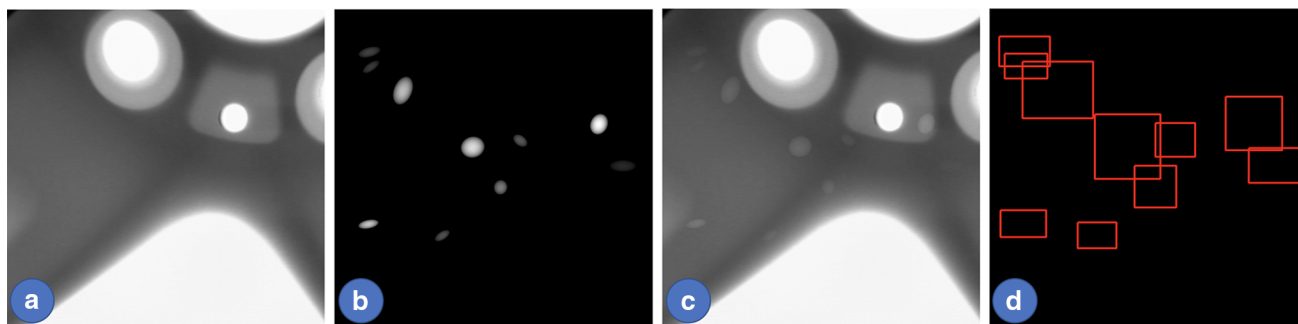


Fig. 4 Simulation of defects: **a** Original X-ray image with no defect. **b** Simulated projections of ellipsoids. **c** X-ray image with simulated defects (superimposition of simulated ellipsoids onto the original X-ray image). **d** Bounding boxes of the simulated defects. Using this method,

it is very simple to generate training data, i.e. ,X-ray images with (simulated) defects and location of the defects, where the bounding boxes are obtained with no manual annotation

achieved using real X-ray images of aluminum castings with real defects. Thus, the reported performance on the testing dataset corresponds to a real scenario.

2.5 Simulation of ellipsoidal defects

We use simulated ellipsoidal defects to train the detection model⁶. In this Section, we summarize the simulation approach. For further details and more examples, see [33,37, 40].

X-ray imaging can typically be modeled using the absorption law, which characterizes the intensity distribution of X-rays through matter [32] and a linear model [33]:

$$I = I(z) = A\varphi_0 e^{-\mu z} + B, \tag{1}$$

where A and B are the constant parameters of the linear model; and μ the absorption coefficient, z the thickness of the irradiated matter, φ_0 the incident energy flux density. In this case, $\varphi_0 \exp(-\mu z)$ corresponds to the energy flux density after passage through matter with the thickness of z . If the material has a bubble of thickness d (with null absorption coefficient), the output energy flux density can be written as $\varphi_0 \exp(-\mu(z - d))$ and the new X-ray image can be modeled from (1) by:

$$I' = I(z - d) = A\varphi_0 e^{-\mu(z-d)} + B \tag{2}$$

where $I(z - d)$ is the new gray-value of the X-ray image with the simulated defect, which can be rewritten as:

$$I' = I(z - d) = (I(z) - B)e^{\mu d} + B. \tag{3}$$

Thus, it is possible to model the X-ray image of a casting with a simulated defect ‘ $I(z - d)$ ’ from the X-ray image of

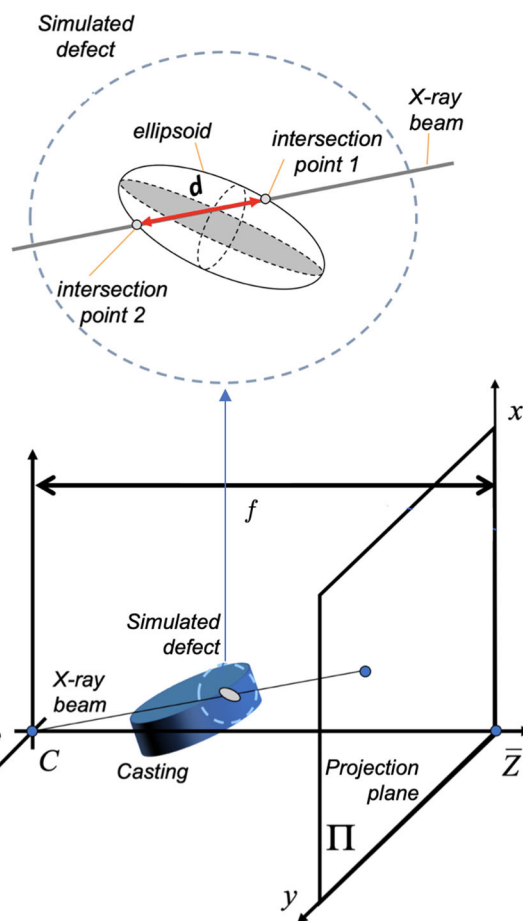


Fig. 5 Simulation of an ellipsoidal defect: for every pixel of the X-ray image, $(x, y) \in \Pi$, the corresponding X-ray beam is estimated. The two intersection points of the X-ray beam with the modeled ellipsoid surface are computed, and the length of the X-ray beam in the bubble d is calculated as the distance between the intersection points

casting with no defect ‘ $I(z)$ ’ and a 3D model of the defect. In our approach, a 3D defect is modeled as an ellipsoidal cavity, which is projected and superimposed onto real X-ray images

⁶ In the simulation, we chose to use ellipsoidal models instead of GAN models because [37] reports that the former perform better.

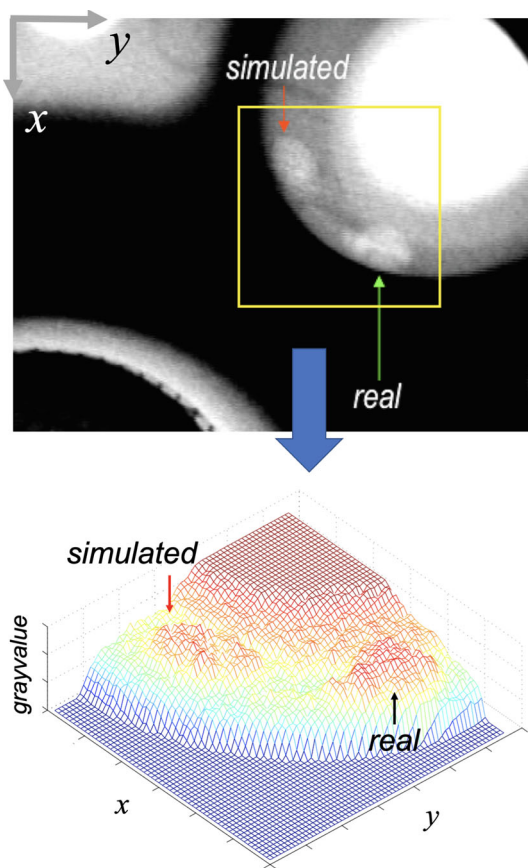


Fig. 6 Similarity between real and simulated defects: grayscale (top) and 3D (bottom) representations

of a homogeneous casting with absorption coefficient μ as shown in Fig. 5.

The synthetic image simulation process is described below:

- Step 1: A real X-ray \mathbf{I} image of an aluminum casting is acquired. The size of the image is $N \times M$ pixels. Image \mathbf{I} typically has no defect.
- Step 2: An ellipsoid is defined in 3D space. The size (dimensions of each axis), location in 3D space (3D coordinates of the center of the ellipsoid) and orientation (3 rotations of each axis) are determined.
- Step 3: For each pixel (x, y) of image \mathbf{I} , the two intersections of the X-ray beam with the ellipsoid surface are calculated, and the distance between them (d) is computed as illustrated in Fig. 5. If there is no intersection, d is set to zero. As output, we obtain a matrix \mathbf{d} of $N \times M$ elements, in which element $d(x, y)$ is the corresponding length d for pixel (x, y) .
- Step 4: A new image \mathbf{I}' of $N \times M$ elements is defined. For each pixel (x, y) of \mathbf{I}' , the corresponding gray-value is computed according to (3) as:

$$I'(x, y) = (I(x, y) - B)e^{\mu d(x, y)} + B. \quad (4)$$

The approach simulates only the defect and not the whole X-ray image of the casting, because for $d = 0$, $I'(x, y) = I(x, y)$.

The new gray-value of a pixel, where the 3D defect is projected, depends on just four parameters:

1. The original gray-value ' $I(x, y)$ ',
2. The linear absorption coefficient of the examined material ' μ ',
3. The calibration parameter ' B ', and
4. The length ' d ' of the intersection of the 3D flaw with the modeled X-ray beam, which is projected into the pixel (see Fig. 5).

A simulation of an ellipsoidal defect of any size and orientation can be performed in any position of the casting. In the simulation, we must consider the fact that the size of the ellipsoid should not be larger than the thickness of the casting where the simulated defect is projected. Some examples are illustrated in Fig. 4.

In our work, we use ellipsoidal defects because it is a very simple model with known geometry. Furthermore, the similarity between real and simulated defects is good enough, as we can see in Fig. 6.

2.6 Training

The detection model is trained using real X-ray images with simulated ellipsoidal defects only as follows (see Figs. 3, 4):

1. Representative X-ray images of the casting object with no defects are selected. The idea is to have X-ray images of every part of the object being tested.
2. In each representative X-ray, random ellipsoidal defects are simulated. The idea is to superimpose many ellipsoidal defects onto the defect-free X-ray images. Here, the length of each axis of the ellipsoid, the orientation, and the 3D location are set randomly. It is worthwhile to mention that the simulated defects must be located in the object, i.e., no simulated defect may be located in the holes of the regular structure of the casting. As we can see in Fig. 4c, all simulated defects are located on the object, and none are located outside of it (in the white areas).
3. For each simulated defect, a bounding box is defined as a rectangle that encloses the projected ellipsoid. Using these three steps, it is very simple to generate training data. Now, we have X-ray images with many (simulated) defects with their locations and the bounding boxes are obtained with no manual annotation.
4. We split the X-ray images (with simulated defects) into a set for training purposes and a set for validation purposes.

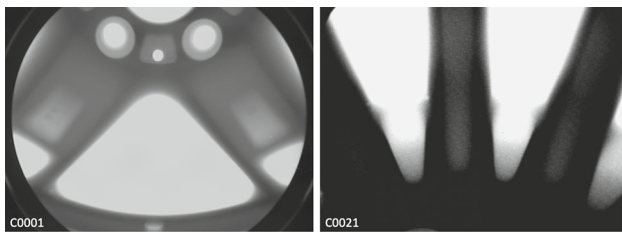


Fig. 7 An example of X-ray images from series C0001 and series C0021 of GD \times ray. Each series contains X-ray images of a specific wheel type

5. The detection model is trained using training and validation sets.

Details of numbers of images and simulated defects per image are provided in Sect. 3.

2.7 Testing

The trained model is tested on X-ray images with no defects and with real defects. The idea is to estimate the performance in a real scenario. Thus, no simulated defects are used in the testing dataset.

To build the testing dataset, we need X-ray images of the same casting type with real defects that are manually annotated by human operators.

Performance and computational time must also be measured in the testing stage.

3 Experimental results

In this Section, we present the experiments and results obtained using the proposed method. We used eight modern algorithms for object detection: YOLOv3 [50] (versions SPP and Tiny), YOLOv5³ (versions ‘s’, ‘l’, ‘m’ and ‘x’), RetinaNet [28] and EfficientDet [54] (see details of implementation in Section 3.3). YOLOv5 and EfficientDet were released in 2020 and the others were released over the past three years. As a baseline, we included two additional methods that were developed earlier: (i) Xnet based on a convolutional neural network [37] and (ii) CLP-SVM [34] based on handcrafted features and SVM classifier. Both methods use the sliding windows strategy. According to Table 1, Xnet is a deep learning method and CLP-SVM is a classic method.

The Section is subdivided into four main parts. The first, Sect. 3.1, focuses on the construction of the datasets using real and simulated defects. The second, Sect. 3.2, shows the results obtained. The third, Sect. 3.3, describes the implementation in detail. Finally, Sect. 3.4 offers a discussion of our results.

3.1 Datasets

In our experiments, we used GD \times ray dataset [42]⁷. From GD \times ray, we used series C0001 for the main experiments with 72 X-ray images, and series C0021 for an additional experiment with 37 X-ray images. Each series contains X-ray images of a specific casting type as illustrated in Fig. 7. Both series have an annotated ground truth that includes real defects. In each series, there is a unique casting piece that is radiographed from different points of view. The following procedure must be performed for each casting type. The explanation and main experiments are provided below for series C0001. In the discussion, we include experiments on series C0021 to validate the proposed method.

Series C0001 belongs to an aluminum wheel commonly used for testing purposes (see, for example, [5,37–39,47]). These castings present two types of defects. The first is a group of blow hole defects (with $\varnothing = 2.0 - 7.5$ mm) which were already present in the castings. They were initially detected during (human) visual inspection (see, for example, Fig. 1). The remaining defects in these castings were produced by drilling small holes (with $\varnothing = 2.0 - 4.0$ mm) in parts of the casting that were known to be difficult to detect (for example, on the edges of regular structures).

3.1.1 Training and validation subsets

We used the following procedure to build the training and validation subsets:

- **Pre-processing:** The size of the images of series C0001 is 572×768 pixels. We resized them by a factor of two to $1144 \times 1,536$ pixels.
- **Selection:** Series C0001 has 72 X-ray images. For each resized X-ray image, we randomly select 100 windows of 640×640 pixels in locations where there is no real defect. An example is given in Fig. 4-a. Of the 100 images, 90 are selected for the training subset and 10 for the validation subset.
- **Simulation:** In each window selected for the previous step, we simulate defects using the ellipsoidal model explained in Sect. 2.5. We can use a Python function given in our repository². The input variables of this function are the size of the three axes of the ellipsoid, the location, and orientation in 3D space of the ellipsoid, the linear absorption coefficient μ of the casting, and a parameter called x_{\max} defined as the maximum thickness where gray values are minimal (it is used to compute parameter B in (3)). We used the same configuration reported in [40]. In our experiments, we set the number of simulated defects per image randomly (from 2 to 20) along with the size and orientation of the three axes of the ellipsoid

⁷ GD \times ray can be used free of charge for research and educational purposes only.

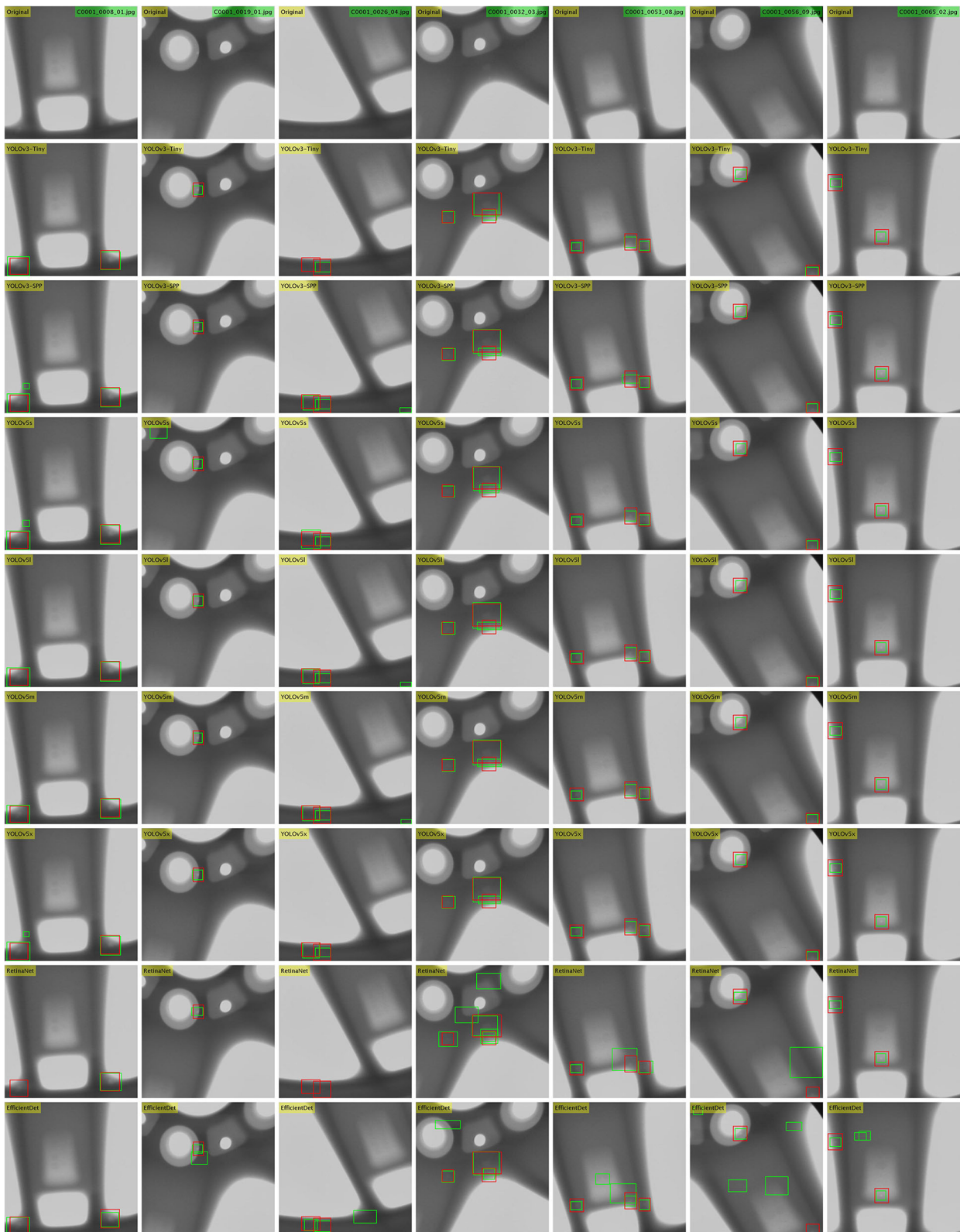


Fig. 8 Results for seven testing images of C0001 (one per column). The first row is the original testing image. The following eight rows are the results obtained using YOLOv3-Tiny, YOLOv3-SPP, YOLOv5s, YOLOv5l, YOLOv5m, YOLOv5x, RetinaNet and EfficientDet, respec-

tively (ground truth (GT), in red, and detection (DT) in green). Baseline methods are not shown due to low performance and due to the space limitations of the article format

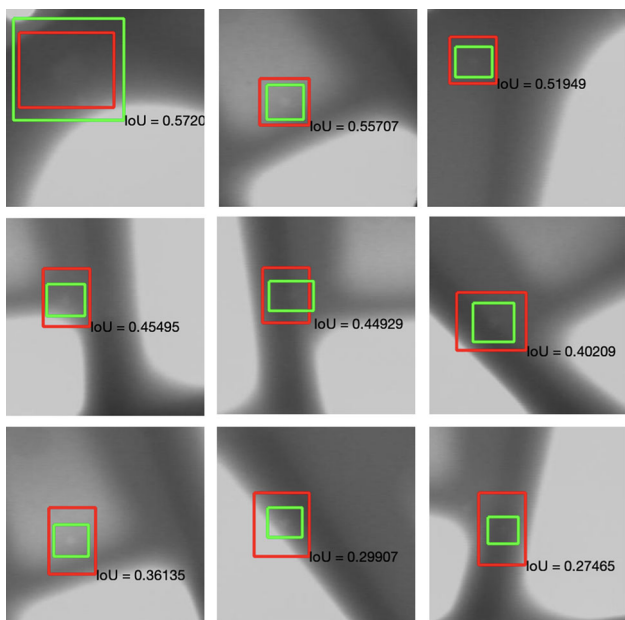


Fig. 9 Different good detections (ground truth (GT) in red, and detection (DT) in green). The size of the cropped images is 140×140 pixels. The almost perfect detection of the first row (the difference is a couple of pixels) yields an IoU score of $0.5 \sim 0.6$. IoU criterion in small defects depends on very accurate ground truth definition. Clearly, in these examples, the detection is more accurate than the ground truth. As such, we set the IoU-threshold in our work at $\alpha = 0.25$

(from 1 to 9mm and from 0 to 2π respectively). Examples are shown in Fig. 4. The defects have an elliptical shape of different sizes, and they are located and orientated in different ways. The size of the axes varies randomly from 1 to 9 mm. In these examples, we show the entire image and the simulated defects. We store the coordinates of the bounding box that enclosed each simulated defect.

In summary, for series C0001, we have 7,200 X-ray images of 640×640 pixels with around 80,000 simulated defects.

3.1.2 Testing subset

We use the following steps to define the testing subset.

- **Pre-processing:** The size of the images of series C0001 is 572×768 pixels. We resized them by a factor of two to $1144 \times 1,536$ pixels.
- **Selection:** Series C0001 has 72 X-ray images. For each resized X-ray image, we randomly selected 10 windows of 640×640 pixels which may contain real defects. An example is given in Fig. 8-first row.

In short, for series C0001, we have 720 X-ray images of 640×640 pixels with around 650 real defects. It is worth mentioning that there is no simulated defect in the testing dataset.

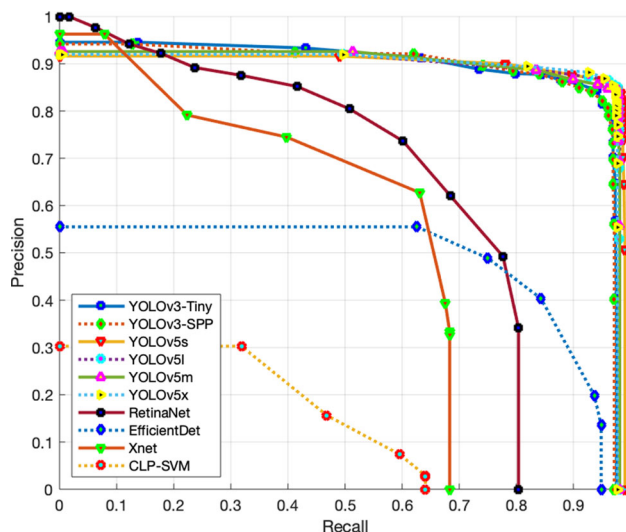


Fig. 10 Precision and recall curves for IoU-threshold $\alpha = 0.25$

Table 2 Evaluation metrics for $\alpha = 0.25$

Method	AP	Pr^*	Re^*	F_1^*
YOLOv3-Tiny	0.8905	0.8441	0.9414	0.8901
YOLOv3-SPP	0.8858	0.8401	0.9321	0.8837
YOLOv5s	0.8962	0.8512	0.9754	0.9091
YOLOv5l	0.8902	0.8654	0.9621	0.9112
YOLOv5m	0.8929	0.8445	0.9773	0.9061
YOLOv5x	0.8887	0.8692	0.9546	0.9099
RetinaNet	0.6482	0.7362	0.6018	0.6622
EfficientDet	0.5091	0.4887	0.7496	0.5917
Xnet	0.5222	0.6271	0.6309	0.6290
CLP-SVM	0.1478	0.3027	0.3187	0.3105

3.2 Results

After training, the models were tested on the testing X-ray images. Some of them are illustrated in Fig. 8. Some models performed well visually, especially the YOLO-based detectors. As we will see in the next experiments, the baseline methods (Xnet and CLP-SVM) did not perform well.

An evaluation metric based on the *Intersection over Union* (IoU) score is used [30] to evaluate the performance of the detectors. In this definition, an existing defect is considered to be detected if the overlap between defect and detection is high enough. Two bounding boxes are used to measure the overlap: one for the defect, called the ground truth GT (see red rectangles in Fig. 9), and one for the detection DT (see green rectangles in Fig. 9). Using these bounding boxes, the normalized overlap is defined as:

$$IoU = \frac{\text{area}(GT \cap DT)}{\text{area}(GT \cup DT)}. \tag{5}$$

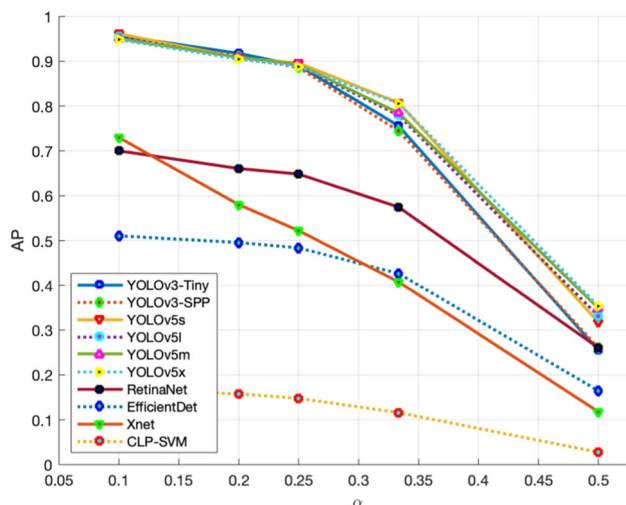


Fig. 11 Average precision (AP) depending on IoU-threshold (α)

The criterion establishes that a defect is detected if $IoU > \alpha$, where α is called the IoU-threshold. In general, the α is set at 0.5. Nevertheless, as we can see in Fig. 9, for very small defects, this IoU-threshold would be so tight that half of the detections would likely be wrong. A more reasonable IoU-threshold for our experiments should be lower, with $\alpha = \frac{1}{4}$ for example all defects of Fig. 9 would be correctly detected. In our experiments, we evaluate the performance for $\alpha = \frac{1}{10}, \frac{1}{5}, \frac{1}{4}, \frac{1}{3}, \frac{1}{2}$.

Using this IoU-criterion, the statistics of true positives (TP), false positives (FP) and false negatives (FN) can be computed, and with them, the precision, recall, and F_1 values are calculated as:

$$Pr = \frac{TP}{TP + FP}, Re = \frac{TP}{TP + FN}, F_1 = 2 \frac{Pr Re}{Pr + Re}. \quad (6)$$

By varying the confidence threshold of the detectors, we will obtain different (Pr, Re) values that can be plotted on the precision-recall curve. In Fig. 10, we show the precision-recall curve for $\alpha = 0.25$. In Table 2, we report the (Pr^*, Re^*) values at maximum F_1 value ($F_1^* = \max(F_1)$). Finally, in our experiments, we use the average precision (AP), computed as the area under the precision-recall curve as an evaluation metric of the detector's performance. Table 2 gives the AP value for $\alpha = 0.25$. In Fig. 11, we show the AP values for different IoU-thresholds (α).

3.3 Implementation

All modern methods were implemented in Python in Google Colab notebooks¹. The following well-known implementations were adapted to our task:

- YOLOv3: Ultralytics⁸
- YOLOv5: Ultralytics³
- RetinaNet: Fizyr⁹
- EfficientDet: Roboflow¹⁰

The use of these methods in aluminum inspection does not present any practical problems at the training or testing stages, such as configuration, versions, etc.

All object detection methods and the X-ray images used for training, validation and testing in this work for aluminum defect detection are available on our public repository². Details of computational time are given in Table 3.

3.4 Discussion

In our experiments, we implemented and tested eight modern object detectors based on YOLO, RetinaNet, and EfficientDet, and two baseline methods (Xnet and CLP-SVM) based on CNNs and handcrafted features, respectively. The main results are given in terms of performance (Figs. 10, 11, and Table 2) and computational time (Table 3). In terms of performance, we have identified three groups according to average precision (AP): (i)YOLO-based methods with $AP > 0.88$, (ii)RetinaNet, EfficientNet, and Xnet with $AP = 0.5\text{--}0.65$, and (iii)CLP-SVM with $AP < 0.15$. In addition, we identified two groups according the testing stage computational time: (i)modern object detection methods with only a few tens of milliseconds per image (ii)baseline methods with more than 1 sec/image.

In defect detection in aluminum castings, since the defects are so small (some of the defect diameters are only 16 pixels), and the manual definition of the bounding boxes of the real defects are very inaccurate, we relaxed the IoU-threshold to $\alpha = 0.25$, so we can consider those small detections as correct (see Fig. 9).

In the reported results (see, for example, Table 2), it is evident that baseline methods Xnet and CLP-SVM did not perform very well. On the other hand, the YOLO-based detectors did perform very well (and at very similar levels), and YOLOv5-methods performed best (better than YOLOv3-methods at IoU-threshold $\alpha > 0.25$). Moreover, RetinaNet and EfficientDet did not perform well, probably because they were designed for larger objects. To overcome this problem, we could increase the resolution of the training images, but that would increase the training time considerably.

In order to explain what the results mean, we will provide more details about YOLOv5s because it achieved one of the

⁸ <https://github.com/ultralytics/yolov3>.

⁹ <https://github.com/fizyr/keras-retinanet>.

¹⁰ <https://blog.roboflow.com/training-efficientdet-object-detection-model-with-a-custom-dataset/>.

Table 3 Computational time

Method	Number of Epochs	Training/epoch [sec]	Training total [hours]	Testing/image [msec]
YOLOv3-Tiny	75	87	1.82	6
YOLOv3-SPP	75	87	1.81	16
YOLOv5s	75	120	2.49	11
YOLOv5l	75	344	7.16	29
YOLOv5m	75	197	4.36	17
YOLOv5x	75	664	13.84	43
RetinaNet	25	107	0.74	56
EfficientDet	200	250	13.89	33
Xnet	100	64	1.78	1.2×10^3
CLP-SVM	–	–	0.25	15×10^3

highest evaluation metrics. As shown in Table 2, for $\alpha = 0.25$, 97.54% of all existing defects were correctly detected (recall), and 85.12% of all detections were true positives. These metrics correspond to $F_1 = 0.9091$. We observe in Fig. 11 that the performance decreases with the IoU score (α value): the larger α , the lower the average precision (AP). For instance, in YOLOv5s, for $\alpha = 0.25$, AP is 0.8962, however, for $\alpha = 0.33$ and 0.5, AP is 0.8075 and 0.3174 respectively. Finally, according to Table 3, YOLOv5s was trained in just 2.5 hours, and the computational time in testing stage was only 11 ms per testing image, i.e., 90 images per second.

For these reasons, we believe that the proposed methodology based on YOLO object detectors could satisfy the requirements of the industry according to the following attributes:

- **Simplicity:** The construction of the training dataset is very simple because we need a low number of defect-free X-ray images¹¹ and a simulation process for including simulated defects in the dataset. This means that no manual annotation is required. Furthermore, we used well-established deep object detection methods that have been easily adapted to our task. The codes are implemented in Python and executed in Google Colab with no intricate configuration and free access to GPUs.

- **Effectiveness:** The performance of the object detectors based on YOLOv5 was very high. The average precision was 0.89–0.90 and the F_1 factor was 0.91 as shown in Table 2.

- **Speed:** The models were trained in a matter of hours. In addition, the computational time of one testing image is only a few tens of milliseconds, as we can see in Table 3, i.e., it can be used in real-time inspection to aid human operators. It is worth mentioning that the computational time of the baseline methods is extremely high because they use the sliding-windows strategy.

Although the YOLO methods performed very well, one disadvantage of the proposed method is that each casting type needs an ad-hoc trained model. That means that if we train the model on images of a specific wheel type (e.g., series C0001 of GDXray, as we did in our experiments), and we use this trained model on X-ray images of another wheel type (e.g., series C0021 of GDXray), it may not perform well (in this example $AP = 0$ because no real defect could be detected). The reader can see the differences between these two wheel types in Fig. 7. This result was expected because the model has learned about the details of the regular structures of one casting type that are not present in the another (and vice versa, the model has not learned the details of the other casting type). However, if we train a new model for this specific wheel type –using the methodology reported in this work–, the performance is increased to a very high level. In the aforementioned example, for wheel C0021 with 37 X-ray images, we obtain performance similar to the performance obtained for wheel C0001 using YOLOv5s, as shown in Table 4.

To overcome the disadvantage of having one trained model per casting type, we could train a single model with X-ray images from both casting types. In our example, if we train and test the model using images of both casting types, the object detector can recognize defects in both (the performance is, however, slightly lower, as shown in Table 4). However, with this solution, we are not avoiding a new training when we have a new casting type. Moreover, the computational time of this new training is higher (because we have more training images), and with the new training, the individual performances must be evaluated again to ensure the effectiveness of the model on all included casting types. For these reasons, we recommend that a specific model be learned for each casting type.

It could be interesting to analyze how many casting types can use a single model. We believe, however, that it is ambitious to have one model for all wheel types. It would be best

¹¹ In our experiment series C0001 has only 72 X-ray images.

Table 4 Evaluation of YOLOv5s on two wheel-types

Training with	Testing on	<i>AP</i>	<i>Pr*</i>	<i>Re*</i>	<i>F₁*</i>
A	A	0.8962	0.8512	0.9754	0.9091
A	B	0.0000	0.0000	0.0000	0.0000
B	B	0.8811	0.8764	0.9155	0.8955
A + B	A	0.8817	0.9754	0.8445	0.9053
A + B	B	0.8654	0.8436	0.8653	0.8543

A: wheel C0001, B: wheel C0021.

to have one model for each type. This should not be a problem for the industry, as the training process can be completed in just 2.5 hours.

4 Conclusions

In this article, we proposed a training strategy that uses defect-free X-ray images of a casting with the superimposition of simulated defects. No manual annotations are required because the locations of all simulated defects are known. In addition, tests are performed using real X-ray images of aluminum castings with real defects. Thus, the reported testing dataset corresponds to a real scenario.

We used well-established object detection methods (YOLO, RetinaNet, and EfficientDet) to detect defects in aluminum castings. All of them were developed in the past three years, and many examples are available in public repositories that could be adapted to our task. The strategies implemented are simple, effective, and fast. The training stage requires a relatively small number of X-ray images. In our experiments, YOLO-based detectors perform best. One of the models, YOLOv5s, was trained in just 2.5 hours. In addition, the testing dataset (with real defects) performed very well (average precision was 0.90 and the F_1 factor was 0.91), and the computational time is very low (the method is able to process 90 X-ray images per second, i.e., this solution can be used in real-time inspection to aid human operators).

The code and the datasets used in this paper have been uploaded to a public repository so that anyone can reproduce (and improve upon) the reported results or re-use the code in other inspection tasks.

In the coming years, deep learning-based methods will be used more frequently by the aluminum castings industry due to their high effectiveness. This paper offers an academic contribution to such efforts.

In the future, we will model random shapes that can be used to simulate other kind of defects such as cracks. This

feature can be very useful in the automated inspection of welds.

Compliance with ethical standards

Conflict of interest The author declares that he has no conflict of interest.

References

- Bandara, A., Kan, K., Morii, H., Koike, A., Aoki, T.: X-ray computed tomography to investigate industrial cast Al-alloys. *Prod. Eng. Res. Devel.* **14**(2), 147–156 (2020)
- Bengio, Y., Courville, A., Vincent, P.: Representation learning: a review and new perspectives. *IEEE Trans. Pattern Anal. Mach. Intell.* **35**(8), 1798–1828 (2013)
- Bochkovskiy, A., Wang C-Y., Liao, H-Y.M.: Yolov4: Optimal speed and accuracy of object detection. [arXiv:2004.10934](https://arxiv.org/abs/2004.10934) (2020)
- Boerner, H., Strecker, H.: Automated X-ray inspection of aluminum casting. *IEEE Trans. Pattern Anal. Mach. Intell.* **10**(1), 79–91 (1988)
- Carrasco, M., Mery, D.: Automatic multiple view inspection using geometrical tracking and feature analysis in aluminum wheels. *Mach. Vis. Appl.* **22**(1), 157–170 (2011)
- Cogranne, R., Retraint, F.: Statistical detection of defects in radiographic images using an adaptive parametric model. *Signal Process.* **96**, 173–189 (2014)
- Deng, J., Guo, J., Xue, N., Zafeiriou, S.: Arcface: additive angular margin loss for deep face recognition. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 4690–4699 (2019)
- Du, W., Shen, H., Fu, J., Zhang, G., He, Q.: Approaches for improvement of the X-ray image defect detection of automobile casting aluminum parts based on deep learning. *NDT & E Int.* **107**, 102144 (2019)
- Esteva, A., Kuprel, B., Novoa, R.A., Ko, J., Swetter, S.M., Blau, H.M., Thrun, S.: Dermatologist-level classification of skin cancer with deep neural networks. *Nature* **542**(7639), 115–118 (2017)
- Ferguson, M., Ak, R., Lee, Y.T.T., Law, K.H.: Automatic localization of casting defects with convolutional neural networks. In: *2017 IEEE International Conference on Big Data*, pp. 1726–1735. IEEE (2017)
- Ferguson, M.K., Ronay, A., Lee, Y.T.T., Law, K.H.: Detection and segmentation of manufacturing defects with convolutional neural networks and transfer learning. *Smart Sustain. Manuf. Syst.* **2** (2018)
- Filbert, D., Klatte, R., Heinrich, W., Purschke, M.: Computer aided inspection of castings. In: *IEEE-IAS Annual Meeting*, pp. 1087–1095. Atlanta, USA (1987)
- Girshick, R.: Fast R-CNN. In: *Proceedings of the IEEE International Conference on Computer Vision*, pp. 1440–1448 (2015)
- Girshick, R., Donahue, J., Darrell, T., Malik, J.: Rich feature hierarchies for accurate object detection and semantic segmentation. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 580–587 (2014)
- Hangai, Y., Kuwazuru, O., Yano, T., Utsunomiya, T., Murata, Y., Kitahara, S., Bidhar, S., Yoshikawa, N.: Clustered shrinkage pores in ILL-conditioned aluminum alloy die castings. *Mater. Trans.* (2010). <https://doi.org/10.1520/SSMS20180033>
- He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. *CoRR arXiv:1512.03385* (2015)

17. Hernández, S., Sáez, D., Mery, D.: Neuro-fuzzy method for automated defect detection of aluminium castings. *Lect. Notes Comput. Sci.* **3212**, 826–833 (2004)
18. Hu, C., Wang, Y.: An efficient convolutional neural network model based on object-level attention mechanism for casting defect detection on radiography images. *IEEE Trans. Indus. Electron.* **67**(12), 10922–10930 (2020). <https://doi.org/10.1109/TIE.2019.2962437>
19. Jiang, X., Hou, Y., Zhang, D., Feng, X.: Deep learning in face recognition across variations in pose and illumination. In: *Deep Learning in Object Detection and Recognition*, pp. 59–90. Springer, Singapore (2019). https://link.springer.com/chapter/10.1007/978-981-10-5152-4_3
20. Jin, C., Kong, X., Chang, J., Cheng, H., Liu, X.: Internal crack detection of castings: a study based on relief algorithm and adaboost-svm. In: *The International Journal of Advanced Manufacturing Technology*, pp. 1–10 (2020)
21. Kamalakannan, A., Rajamanickam, G.: Spatial smoothing based segmentation method for internal defect detection in X-ray images of casting components. In: *2017 Trends in Industrial Measurement and Automation (TIMA)*, pp. 1–6. IEEE (2017)
22. Krizhevsky, A., Sutskever, I., Hinton, G.E.: ImageNet classification with deep convolutional neural networks. *NIPS*, pp. 1106–1114 (2012)
23. LeCun, Y., Bottou, L., Bengio, Y.: Gradient-based learning applied to document recognition. In: *Proceedings of the Third International Conference on Research in Air Transportation* (1998)
24. Li, J., Oberdorfer, B., Schumacher, P.: Determining casting defects in thixomolding mg casting part by computed tomography. In: *Shape Casting*, pp. 99–103. Springer, Cham (2019). https://link.springer.com/chapter/10.1007/978-3-030-06034-3_9
25. Li, W., Li, K., Huang, Y., Deng, X.: A new trend peak algorithm with X-ray image for wheel hubs detection and recognition. In: *International Symposium on Computational Intelligence and Intelligent Systems*, pp. 23–31. Springer, Singapore (2015). https://link.springer.com/chapter/10.1007/978-981-10-0356-1_3
26. Li, X., Tso, S.K., Guan, X.P., Huang, Q.: Improving automatic detection of defects in castings by applying wavelet technique. *IEEE Trans. Ind. Electron.* **53**(6), 1927–1934 (2006)
27. Lin, J., Yao, Y., Ma, L., Wang, Y.: Detection of a casting defect tracked by deep convolution neural network. *Int. J. Adv. Manuf. Technol.* **97**(1–4), 573–581 (2018)
28. Lin, T., Goyal, P., Girshick, R.B., He, K., Dollár, P.: Focal loss for dense object detection. *CoRR* **abs/1708.02002** (2017)
29. Lin, T.Y., Dollár, P., Girshick, R., He, K., Hariharan, B., Belongie, S.: Feature pyramid networks for object detection. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 2117–2125 (2017)
30. Lin, T.-Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., Dollár, P., Lawrence Zitnick, C.: Microsoft coco: common objects in context. In: *European Conference on Computer Vision*, pp. 740–755. Springer, Cham (2014). https://link.springer.com/chapter/10.1007/978-3-319-10602-1_48
31. Liu, L., Ouyang, W., Wang, X., Fieguth, P., Chen, J., Liu, X., Pietikäinen, M.: Deep learning for generic object detection: a survey. *Int. J. Comput. Vision* **128**(2), 261–318 (2020)
32. Martz, H.E., Logan, C.M., Schneberk, D.J., Shull, P.J.: *X-ray Imaging: Fundamentals, Industrial Techniques and Applications*. CRC Press, Boca Raton (2016)
33. Mery, D.: A new algorithm for flaw simulation in castings by superimposing projections of 3D models onto X-ray images. In: *Proceedings of the XXI International Conference of the Chilean Computer Science Society (SCCC-2001)*, pp. 193–202. IEEE Computer Society Press, Punta Arenas (2001)
34. Mery, D.: Crossing line profile: a new approach to detecting defects in aluminium castings. In: *Proceedings of the Scandinavian Conference on Image Analysis (SCIA 2003)*, *Lecture Notes in Computer Science*, vol. 2749, pp. 725–732 (2003)
35. Mery, D.: Automated radiosopic testing of aluminum die castings. *Mater. Eval.* **64**(2), 135–143 (2006)
36. Mery, D.: Inspection of complex objects using multiple-X-ray views. *IEEE/ASME Trans. Mechatron.* **20**(1), 338–347 (2015)
37. Mery, D.: Aluminum casting inspection using deep learning: a method based on convolutional neural networks. *J. Nondestr. Eval.* **39**(1), 12 (2020)
38. Mery, D., Arteta, C.: Automatic defect recognition in X-ray testing using computer vision. In: *2017 IEEE Winter Conference on Applications of Computer Vision (WACV)*, pp. 1026–1035. IEEE (2017)
39. Mery, D., Filbert, D.: Automated flaw detection in aluminum castings based on the tracking of potential defects in a radiosopic image sequence. *IEEE Trans. Robot. Autom.* **18**(6), 890–901 (2002)
40. Mery, D., Hahn, D., Hirschfeld, N.: Simulation of defects in aluminum castings using cad models of flaws and real X-ray images. *Insight* **47**(10), 618–624 (2005)
41. Mery, D., Pieringer, C.: *Computer Vision for X-ray testing*, 2nd edn. Springer, New York (2021)
42. Mery, D., Riffo, V., Zscherpel, U., Mondragón, G., Lillo, I., Zuccar, I., Lobel, H., Carrasco, M.: GDxray: the database of X-ray images for nondestructive testing. *J. Nondestr. Eval.* **34**(4), 1–12 (2015)
43. Mery, D., Riffo, V., Zuccar, I., Pieringer, C.: Automated X-ray object recognition using an efficient search algorithm in multiple views. In: *Proceedings of the 9th IEEE CVPR workshop on Perception Beyond the Visible Spectrum*, Portland (2013)
44. Nagpal, K., Foote, D., Liu, Y., Chen, P.H.C., Wulczyn, E., Tan, F., Olson, N., Smith, J.L., Mohtashamian, A., Wren, J.H., et al.: Development and validation of a deep learning algorithm for improving gleason scoring of prostate cancer. *NPJ Digit. Med.* **2**(1), 1–10 (2019)
45. Pieringer, C., Mery, D.: Flaw detection in aluminium die castings using simultaneous combination of multiple views. *Insight* **52**(10), 548–552 (2010)
46. Pizarro, L., Mery, D., Delpiano, R., Carrasco, M.: Robust automated multiple view inspection. *Pattern Anal. Appl.* **11**(1), 21–32 (2008)
47. Ramírez, F., Allende, H.: Detection of flaws in aluminium castings: a comparative study between generative and discriminant approaches. *Insight Non Destr. Test. Cond. Monit.* **55**(7), 366–371 (2013)
48. Redmon, J., Divvala, S.K., Girshick, R.B., Farhadi, A.: You only look once: Unified, real-time object detection. *CoRR* **arXiv:1506.02640** (2015)
49. Redmon, J., Farhadi, A.: YOLO9000: better, faster, stronger. *CoRR* **arXiv:1612.08242** (2016)
50. Redmon, J., Farhadi, A.: Yolov3: An incremental improvement. *CoRR* **arXiv:1804.02767** (2018)
51. Ren, J., Ren, R., Green, M., Huang, X.: Defect detection from X-ray images using a three-stage deep learning algorithm. In: *2019 IEEE Canadian Conference of Electrical and Computer Engineering (CCECE)*, pp. 1–4. IEEE (2019)
52. Ren, S., He, K., Girshick, R., Sun, J.: Faster R-CNN: Towards real-time object detection with region proposal networks. In: *Advances in Neural Information Processing Systems*, pp. 91–99 (2015)
53. Saavedra, D., Banerjee, S., Mery, D.: Detection of threat objects in baggage inspection with x-ray images using deep learning. *Neural Comput. Appl.* (2020). <https://doi.org/10.1007/s00521-020-05521-2>
54. Tan, M., Pang, R., Le, Q.V.: EfficientDet: scalable and efficient object detection. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 10781–10790 (2020)

55. Tang, Y., Zhang, X., Li, X., Guan, X.: Application of a new image segmentation method to detection of defects in castings. *Int. J. Adv. Manuf. Technol.* **43**(5–6), 431–439 (2009)
56. Tang, Z., Tian, E., Wang, Y., Wang, L., Yang, T.: Non-destructive defect detection in castings by using spatial attention bilinear convolutional neural network. *IEEE Trans. Ind. Inf.* (2020). <https://doi.org/10.1109/TII.2020.2985159>
57. Tremblay, J., Prakash, A., Acuna, D., Brophy, M., Jampani, V., Anil, C., To, T., Cameracci, E., Boochoon, S., Birchfield, S.: Training deep networks with synthetic data: Bridging the reality gap by domain randomization. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pp. 969–977 (2018)
58. Tripathi, S., Chandra, S., Agrawal, A., Tyagi, A., Rehg, J.M., Chari, V.: Learning to generate synthetic data via compositing. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 461–470 (2019)
59. Yongwei, Y., Liuqing, D., Cuilan, Z., Jianheng, Z.: Automatic localization method of small casting defect based on deep learning feature. *Chin. J. Sci. Instrum.* **2016**(6), 21 (2016)
60. Zhang, J., Guo, Z., Jiao, T., Wang, M.: Defect detection of aluminum alloy wheels in radiography images using adaptive threshold and morphological reconstruction. *Appl. Sci.* **8**(12), 2365 (2018)
61. Zhao, X., He, Z., Zhang, S.: Defect detection of castings in radiography images using a robust statistical feature. *JOSA A* **31**(1), 196–205 (2014)
62. Zhao, X., He, Z., Zhang, S., Liang, D.: A sparse-representation-based robust inspection system for hidden defects classification in casting components. *Neurocomputing* **153**, 1–10 (2015)
63. Zhao, Z., Zheng, P., Xu, S., Wu, X.: Object detection with deep learning: a review. *IEEE Trans. Neural Netw. Learn. Syst.* **30**(11), 3212–3232 (2019). <https://doi.org/10.1109/TNNLS.2018.2876865>



Domingo Mery is a Full Professor in the Department of Computer Science at UC. He received the Diploma (M.Sc.) degree in Electrical Engineering from the Technical University of Karlsruhe, Germany, in 1992, and the Ph.D. degree with distinction at the Technical University of Berlin, in 2000. He was a Research Scientist at the Institute for Measurement and Automation Technology at the Technical University of Berlin with the collaboration of YXLON X-Ray International. He

was a recipient of a Scholarship from the Konrad-Adenauer Foundation, and from a Scholarship from the German Academic Exchange Service (DAAD) for his Ph.D. work. He was Associate Research in 2001 at the Department of Computer Engineering at the University of Santiago, Chile. Now, he is a Full Professor at the Department of Computer Science at the Pontificia Universidad Católica de Chile (UC), Chile. He was Chair of the Computer Science Department in 2005–2009. He was an Associate Visiting Professor at the Computer Vision Research Lab of the University of Notre Dame in 2014–2015. He was Director of Research and Innovation of the School of Engineering at the UC in 2015–2018. He serves as Associate Editor of the *IEEE Transactions on Information, Forensics and Security*, and the *IEEE Transactions on Transactions on Biometrics, Behavior, and Identity Science*. His research interests include image processing for fault detection in aluminum castings, X-ray imaging, real time programming, and computer vision. He is the author of more than 80 technical SCI publications, and more than 90 conference papers.

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.