Computer Vision for Fault Detection in Aluminum Castings

Domingo Mery

Abstract—Castings produced for the automotive industry are considered important components for overall roadworthiness. To ensure the safety of construction, it is necessary to check every part thoroughly using non-destructive testing. X-ray testing rapidly became the accepted way for controlling the quality of die cast pieces. In this paper the fundamental principles of the automated detection of casting discontinuities are explained. A general computer vision inspection schema is presented, and several methods that have appeared in the literature in the past thirty years were explained showing the development of this sector in the areas of industry and academia.

Index Terms—X-ray testing, computer vision, image processing, image analysis, pattern recognition, aluminum castings, quality control.

1 Introduction

To ensure safety in the construction of important metallic components for roadworthiness, it is necessary to check every component thoroughly using non-destructive testing. In last decades, X-ray testing has been adopted as the principal non-destructive testing method to identify defects within a component which are undetectable to the naked eye. Nowadays, modern computer vision techniques, such as deep learning [37] and sparse representations [75], are opening new avenues in automatic object recognition in optical images. These techniques have been broadly used in object and texture recognition by the computer vision community with promising results in optical images. In this work, we present a general overview of image processing approaches that have been used in X-ray testing for the inspection of aluminum castings.

Light-alloy castings produced for the automotive industry, such as wheel rims, steering knuckles and steering gear boxes are considered important components for overall roadworthiness. Non-homogeneous regions can be formed within the work piece in the production process. These are manifested, for example, by bubble-shaped voids, fractures, inclusions or slag formation. To ensure the safety of construction, it is necessary to check every part thoroughly using X-ray testing. In casting inspection, automated X-ray systems have not only raised quality, through repeated objective inspections and improved processes, but have also increased productivity and consistency by reducing labor costs. Some examples are illustrated in Fig. 1.

X-Ray testing is one of the more accepted ways for examining an object without destroying it [49]. The purpose of this non-destructive method is to detect or recognize certain parts of interest that are located inside a test object and are thus not detectable to the naked eye. The material defects occurring in the casting process such as cavity, gas, inclusion and sponge must be detected to satisfy the security requirements; consequently, it is necessary to check 100% of the parts.

The principle aspects of an X-ray testing system is illustrated in Fig. 2. Typically, it comprises the following steps:

- The test object is located in the desired position using a manipulator.
- The X-ray source generates X-rays which pass through the test object.
- The X-rays are detected and converted by a flat panel in order to obtain a digital X-ray image.
- Computer vision algorithms are used to evaluate the X-ray image.

• *D. Mery is with the Department of Computer Science Department (DCC), Pontificia Universidad Católica, Vicuña Mackenna 4860, Santiago Chile. E-mail: dmery@ing.puc.cl URL: http://dmery.ing.puc.cl*

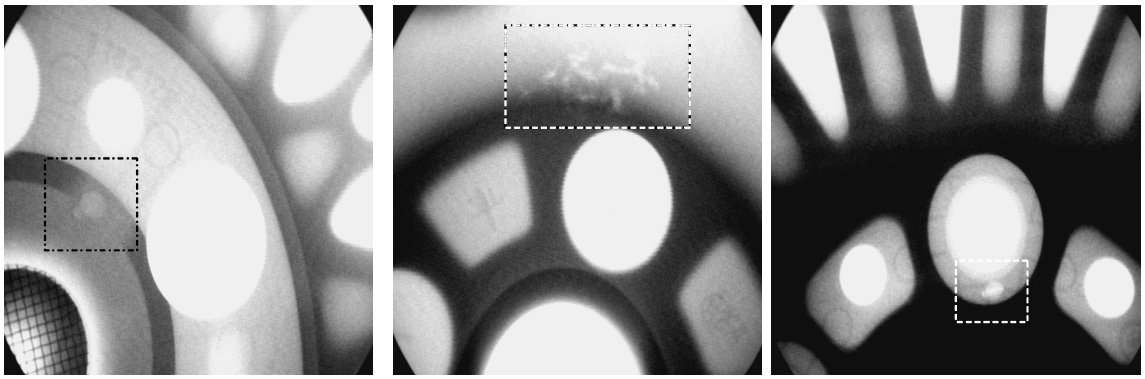

Fig. 1: Real defects in X-ray images of wheels.

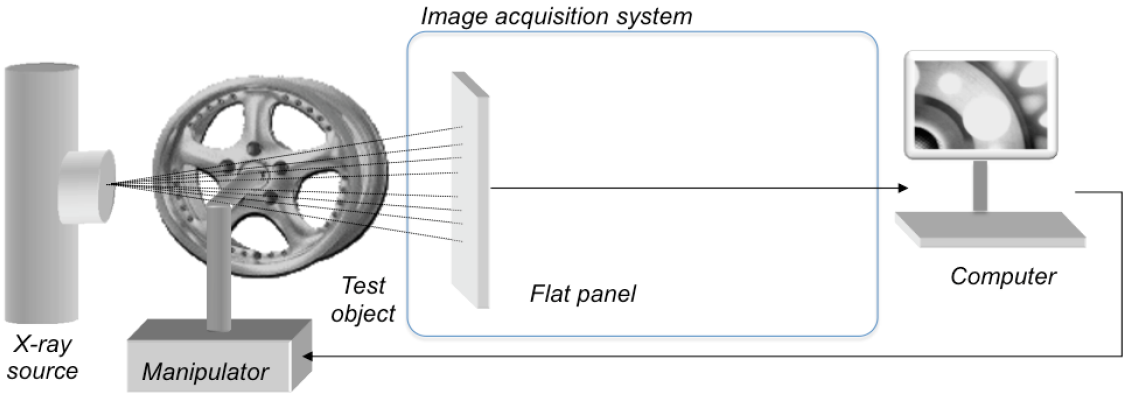

Fig. 2: X-ray testing system. The image acquisition system is based on flat panels. In this example, an aluminum wheel is inspected using a manipulator.

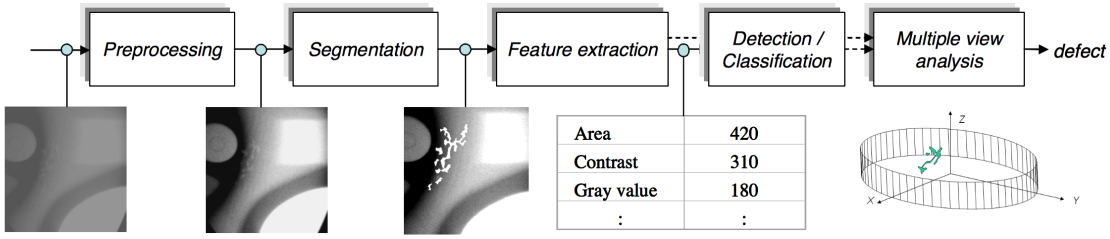

Fig. 3: Schematic diagram of an automated X-ray testing stand.

In recent years, flat detectors made of amorphous silicon have been widely used as image sensors in some industrial inspection systems [20]. In these detectors, the energy from the X-ray is converted directly into an electrical signal by a semi-conductor (without an image intensifier).

In the computer-aided inspection of castings, our aim is to identify discontinuities automatically using computer vision techniques. The general automated inspection process, presented in Fig. 3, consists of image formation, pre-processing, segmentation, feature extraction, detection/classification and multiple view analysis. Some approaches use pattern recognition and multiple view techniques. A general schema follows the mentioned six steps:

- *Image formation*: An X-ray image of the casting under test is taken and stored in the computer. The eye is only capable of resolving around 40 grey levels [8], however for the detection of discontinuities in aluminum castings, grey level resolution must be a minimum of 2^8 levels. In some applications, 2^{16} grey levels are

used, which allows one to evaluate both very dark and very bright regions in the same image.
- *Image pre-processing*: The quality of the X-ray image is improved in order to enhance the details of the X-ray image. Usually, the pre-processing techniques are used to remove noise, enhance contrast, correct the shading effect and restore blur deformation.
- *Image segmentation*: The digital images is divided into disjoint regions with the purpose of separating the parts of interest from the rest of the scene. The idea is to segment those regions that correspond to the defects of the specimen.
- *Feature extraction*: Since some structural parts of the object could be erroneously segmented as defectively regions in previous step, they are denoted as hypothetical defects. Subsequently, additional steps are required to eliminate the false alarms of the hypothetical defects. The first of these steps is feature extraction, which is centered principally around the measurement of geometric properties and on the intensity characteristics of regions. It is important to know which features provide information about discontinuities. With this end, a feature selection is carried out to find those features that best describe discontinuities, eliminating for example features that are correlated or provide no information whatsoever.
- *Detection/classification*: The extracted (and selected) features of each region are analyzed in order to detect or classify the existing defects. We will differentiate between the detection of discontinuities and the classification of discontinuities. Detection corresponds to a binary classification, because in the detection problem, the classes that exist are only two: 'discontinuities' (defects) or 'regular structures' (no defects), whereas the recognition of the type of discontinuity (*e.g.*, voids, cracks, bubbles, inclusions and slags) is known as classification of discontinuity types.
- *Multiple view analysis*: Some methods use an additional step based on multiple view geometry. The key idea of the multiple view analysis is to gain more information about a test object by analyzing multiple views taken at different viewpoints. It is a useful and powerful alternative for examining complex objects were uncertainty can lead to misinterpretation, because two or more views of the same object taken from different viewpoints can be used to confirm and improve the diagnostic done by analyzing only one image.

In this work, we will discuss the use computer vision as a tool in the automated radioscopic inspection of aluminum die castings. In Section 2 we present the state of the art. In Section 3, we discuss image processing techniques used in this field. In Section 4, we explain different methods that can be used to represent the image information of the castings. In Section 5, we give an overview of the multiple view analysis. Finally, in Section 6 we present concluding remarks[1]

2 State of the art

Different methods for the automated detection of casting discontinuities using computer vision have been described in the literature over the past thirty years [49]. One can see that the different approaches for this task can be divided into three groups [56], [47]:

- Reference methods: In reference methods it is necessary to take still images at selected programmed inspection positions. A test image is then compared with 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 using pattern recognition, expert systems, artificial neural networks, general filters or multiple view analyzes to make them independent of the position and structure of the test piece.
- Computed tomography: These approaches use computed tomography to make a reconstruction of the cast piece and thereby detect discontinuities.

A selection of recent approaches are summarized in Table 1. In general, they use classical image processing and pattern recognition techniques with a single view or multiples views, where decision is

1. We developed Matlab Toolbox called 𝕏vis Toolbox (available on-line on http://dmery.ing.puc.cl/index.php/book/xvis/), with around 150 functions for computer vision in X-ray testing. Moreover, there are around 20,000 X-ray images on the 𝔾𝔻𝕏ray database [59] (available on-line on http://dmery.ing.puc.cl/index.php/material/gdxray/) that can be used to test different algorithms and codes. The X-ray images of this work belong to 𝔾𝔻𝕏ray.

TABLE 1: Defect recognition on castings

Reference	Description
Carrasco [7]	multiple view correspondence with non-calibrated model
Cogranne [11]	statistical hypothesis testing using nonparametric tests
Li [40]	wavelet technique
Li [39]	peak location algorithm combined with neural networks
Mery [52]	multiple view model with calibrated model
Mery [48]	multiple views using an un-calibrated tracking approach
Pieringer [67]	3D model from 2D images
Pizarro [68]	multiple views based on affine transformation
Ramirez [69]	generative and discriminative approaches
Tang [73]	segmentation using fuzzy model
Zhao [80]	statistical feature based on grayscale arranging pairs
Zhao[81]	sparse representations

made by analyzing the correspondence among the views.

Nowadays, modern computer vision techniques, such as deep learning, *e.g.*, [37], [3] and [71], and sparse representations, *e.g.*, [75] and [78], are opening new avenues in automatic object recognition in optical images. These techniques have been broadly used in object and texture recognition by the computer vision community with promising results in optical images.

3 X-ray Image Processing

Image processing manipulates a digital image in order to obtain a *new* digital image, *i.e.*, in this process the input is an image and the output is another image. A typical example is *segmentation* as shown in Fig. 4, where the input is a grayscale image that contains a defect and the output is a binary image where the pixels that belong to the defect are detected. We distinguish image processing from image analysis, in which the output is rather an interpretation, a recognition or a measurement of the input image.

In this Section, we cover the following image processing techniques that are used in X-ray testing.

- Image pre-processing: The quality of the X-ray image is improved in order to enhance its details.
- Image Filtering: Mainly used to remove noise and detect high frequency details of the X-ray image.
- Edge detection: The details of the images can be highlighted by detecting the boundaries of the objects of the X-ray image.
- Image segmentation: Regions of interest of the X-ray image are identified and isolated from their surroundings.
- Image restoration: This involves recovering details in blurred images.

In this Section we provide an overview of these five techniques. Methodologies and principles will also be outlined, and some application examples followed by limitations to the applicability of the used methodologies will be presented.

In image processing methodology we have a continuous image f defined in a coordinate system (x, y). Image f is digitalized. The obtained image is a digital image which is stored in matrix $\mathbf{X}$ of size $M \times N$ pixels. The gray value of pixel (i, j) of image $\mathbf{X}$ is $X(i, j)$. Image $\mathbf{X}$ is processed digitally. The output image of this process is image $\mathbf{Y}$, usually a matrix of the same size of $\mathbf{X}$. In this example, the output is a binary image, that means $Y(i, j)$ is '1' (white) and '0' black. Image $\mathbf{Y}$ corresponds to the segmentation of a defect in an aluminum casting (that is the *object of interest* in this example).

3.1 Image pre-processing

The X-ray image taken must be pre-processed to improve the quality of the image before it is analyzed. In this Section, we will discuss pre-processing techniques that can remove noise, enhance contrast, correct the shading effect and restore blur deformation in X-ray images.

3.1.1 Noise removal

Noise in an X-ray image can prove a significant source of image degradation and must be taken

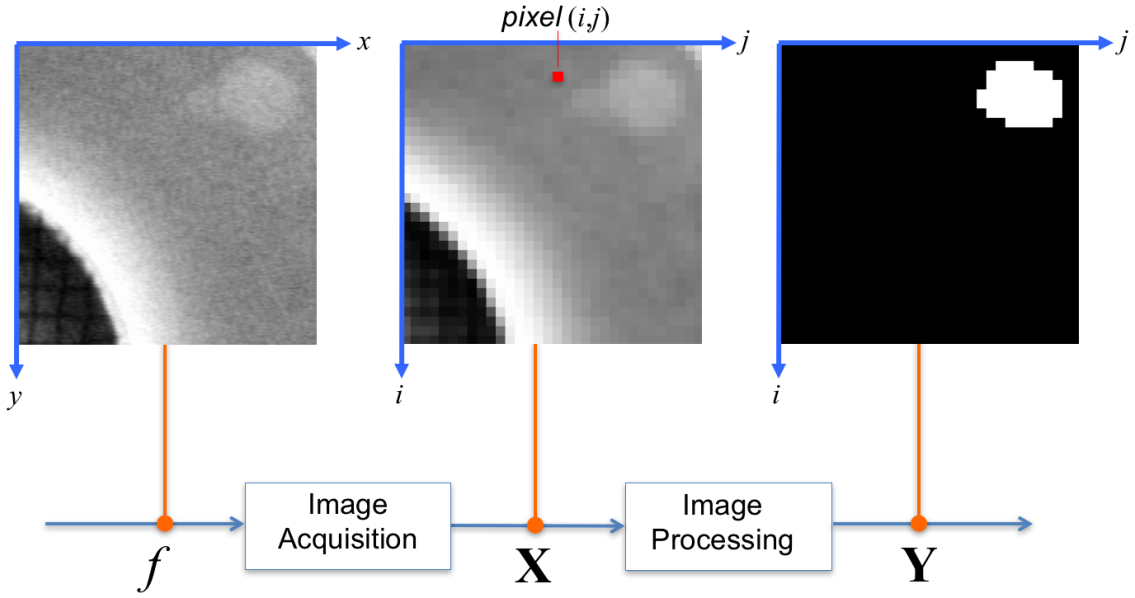

Fig. 4: Image processing: input is digital image $\mathbf{X}$, output is digital image $\mathbf{Y}$.

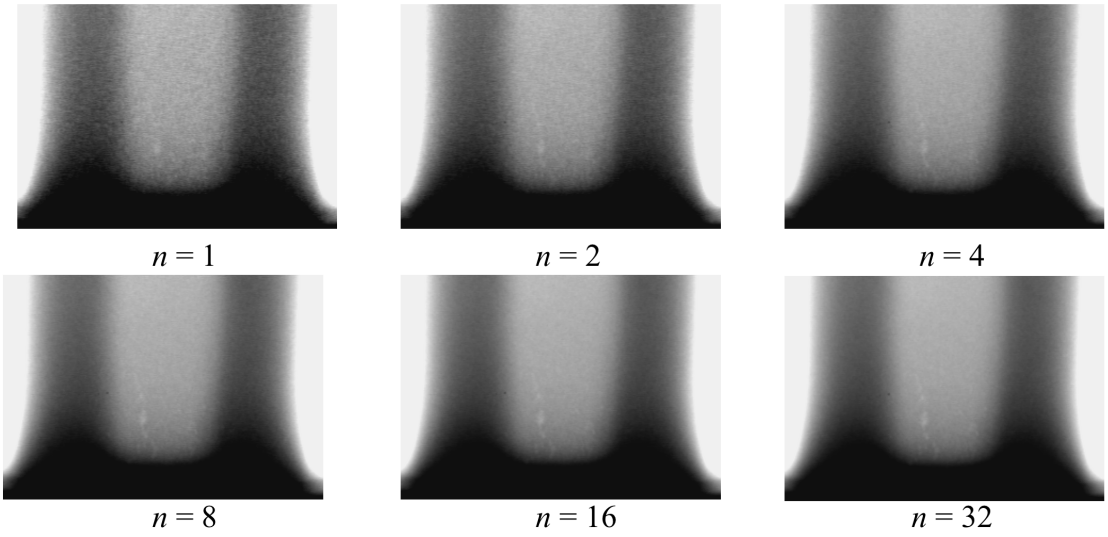

Fig. 5: Noise removal after an averaging of n frames. The noise is reduced by factor $\sqrt{n}$.

into account during image processing and analysis. In an X-ray imaging system, *photon noise* occurs given the quantum nature of X-rays. If we have a system that receives μ photons per pixel in a time ΔT on average, the number of photons striking any particular pixel in any time ΔT will be random. At low levels, however, the noise follows a Poisson law, characterized by the probability:

$$p(x|\mu) = \frac{e^{-\mu}}{\mu^x x!} \tag{1}$$

to obtain a value x of photons given its average μ photons in a time ΔT. The standard deviation of this distribution is equal to the square root of the mean[2]. This means that the photon noise amplitude

2. At high levels, the Poisson distribution approaches the Gaussian with a standard deviation equal to the square root of the mean: $\sigma = \sqrt{\mu}$.

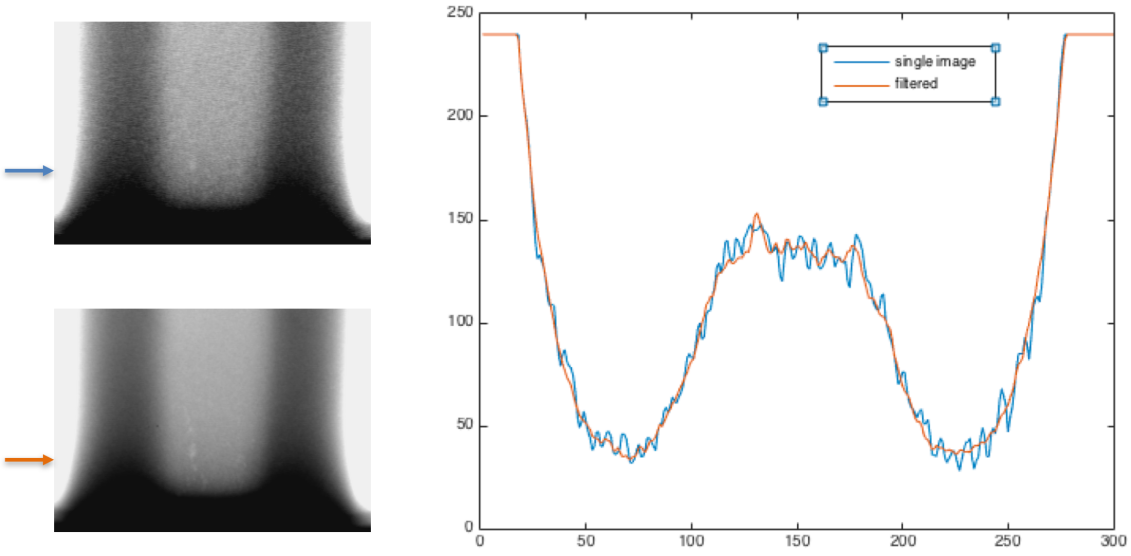

Fig. 6: Noise removal of an X-ray image of a casting after an averaging of 32 frames. Left) one of the 32 images (top) and filtered image (bottom). Right) row 140 of each image where two defects are present. The reader can see how difficult is to distinguish the defects in the original image (without filtering).

is signal-dependent.

Integration (or averaging) is used to remove X-ray image noise. This technique requires n stationary X-ray images. It computes the filtered image as follows:

$$Y(i,j) = \frac{1}{n}\sum_{k=1}^{n} X_k(i,j) \tag{2}$$

where $X_k(i,j)$ is pixel (i,j) of k-th stationary image, and $Y(i,j)$ is the corresponding pixel of the filtered image.

In this technique, the X-ray image noise is modeled using two components: the stationary component (that is constant throughout the n images) and the noise component (that varies from one image to the next). If the noise component has zero mean, by averaging the n images the stationary component is unchanged, while the noise pattern decreases by increasing n. Integrating n stationary X-ray images improves the signal-to-noise ratio by a factor of $\sqrt{n}$ [8], [5].

The effect of image integration is illustrated in Fig. 5 and 6 that uses n stationary images of an aluminum casting and shows the improvement in the quality of the X-ray image. The larger the number of stationary images n, the better the improvement. Normally, between 10 and 16 stationary images are taken ($10 \leq n \leq 16$).

3.1.2 Contrast enhancement

The gray values in some X-ray images lie in a relatively narrow range of the gray scale. In this case, enhancing the contrast will amplify the differences in the gray values of the image.

We compute the gray value histogram to investigate how an X-ray image uses the grayscale. The function summarizes the gray value information of an X-ray image. The histogram is a function $h(x)$ that denotes the number of pixels in the X-ray image that have a gray value equal to x. Fig. 7 shows how each histogram represents the distribution of gray values in the X-ray images.

A transformation can be applied to modify the distribution of gray value in an X-ray image. Simple contrast enhancement can be achieved if we use a linear transformation which sets the minimal and maximal gray values of the X-ray image to the minimal and maximal gray value of the grayscale respectively. Thus, the histogram is expanded to occupy the full range of the grayscale. Mathematically, for a scale between 0 and 255, this transformation is expressed as $\mathbf{Y} = 255\mathbf{X}_0$, with:

$$X_0(i,j) = \frac{X(i,j) - x_{\min}}{x_{\max} - x_{\min}} \tag{3}$$

where $x_{\min}$ and $x_{\max}$ denote the minimal and maximal gray value of the input X-ray image $\mathbf{X}$. The

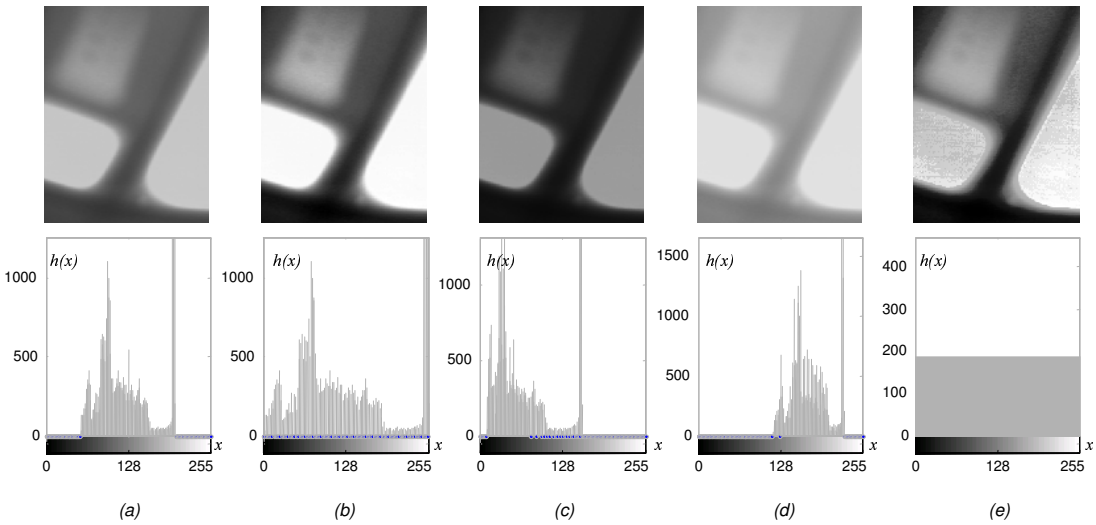

Fig. 7: Contrast enhancement: a) original image, b) linear transformation ($\gamma = 1$), c) non-linear transformation ($\gamma = 2$), d) non-linear transformation ($\gamma = 1/2$), e) gray values uniformly distributed.

output image is stored in matrix $\mathbf{Y}$. Fig. 7b shows the result of the transformation applied to the X-ray image in Fig. 7a. We observe in the histogram of the enhanced X-ray image, how the gray values expand from '0' to '255'. The mapping is linear, and means that a gray value equal to $\frac{1}{2}(x_{\max} - x_{\min})$ will be mapped to $255/2$. This linear transformation is illustrated in Fig. 8a, where the abscissa is the input gray value and the ordinate is the output gray value.

In a similar fashion, gray input image values can be mapped using a non-linear transformation $y = f(x)$, as illustrated in Fig. 8b and 8c, the results of which are shown in Fig. 7c and 7d, respectively. Here, x and y are the gray values of the input and output images respectively. The non-linear transformation is usually performed with a γ *correction* [43]. In these examples, if $\gamma > 1$ the mapping is weighted toward darker output values, and if $\gamma < 1$ the mapping is weighted toward brighter output values. Gamma transformation can be expressed as $\mathbf{Y} = 255\mathbf{Y}_0$, with

$$Y_0(i,j) = \begin{cases} 0 & \text{for } X(i,j) < x_{\min} \\ (X_0(i,j))^{\gamma} & x_{\min} \leq X(i,j) \leq x_{\max} \\ 1 & X(i,j) > x_{\max} \end{cases} \tag{4}$$

where $X_0(i,j)$ is defined in (3).

Finally, we present a contrast enhancement equalizing the histogram. Here, we can alter the gray value distribution in order to obtain a desired histogram. A typical equalization corresponds to the uniform histogram as shown in Fig. 7e. We see that the number of pixels in the X-ray image for each gray value is constant. In the example of Fig. 9, we have zones of an X-ray image of a casting with poor contrast. We see how the defects can be distinguishable using the mentioned approach.

3.1.3 *Shading correction*

A decrease in the angular intensity in the projection of the X-rays causes low spatial frequency variations in X-ray images [26], [5]. An example is illustrated in Fig. 11a, which shows an X-ray image of an aluminum plate with holes in it. Since the plate is of a constant thickness, we would expect to see a constant gray value for the aluminum part and another constant gray value for the holes. In fact, the X-ray image is darker at the corners. This deficiency can be overcome by using a linear shading correction.

In this technique, we take two reference images as shown in Fig. 12. The first one, $\mathbf{r}_1$, of a thin plate, and the second one, $\mathbf{r}_2$, of a thick plate. We define i_1 and i_2 as the ideal gray values for the first and second image, respectively. From $\mathbf{r}_1$, $\mathbf{r}_2$, i_1 and i_2, offset and gain, correction matrices $\mathbf{a}$ and $\mathbf{b}$ are calculated assuming a linear transformation between the original X-ray image $\mathbf{x}$ and corrected X-ray image $\mathbf{y}$:

$$Y(i,j) = a(i,j) \cdot X(i,j) + b(i,j), \tag{5}$$

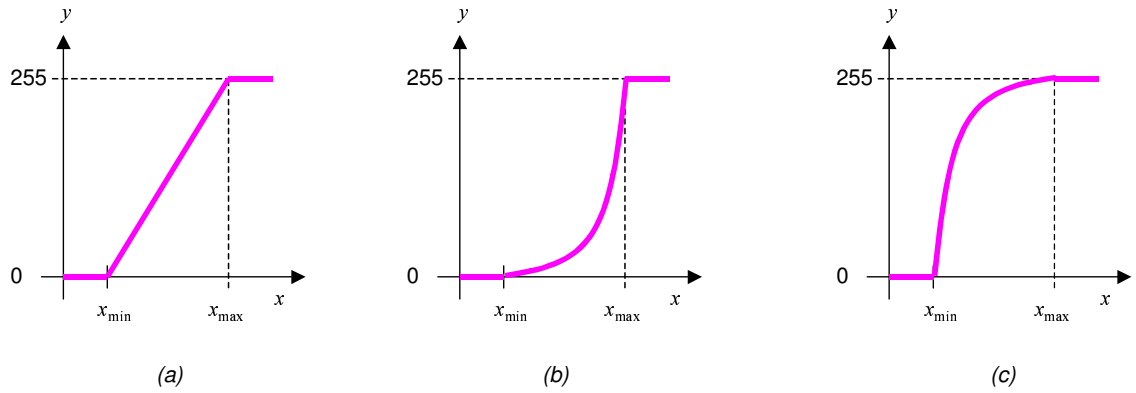

Fig. 8: Plots showing different transformations of the gray values: a) linear transformation ($\gamma = 1$), b) non-linear transformation with $\gamma > 1$, c) non-linear transformation with $\gamma < 1$.

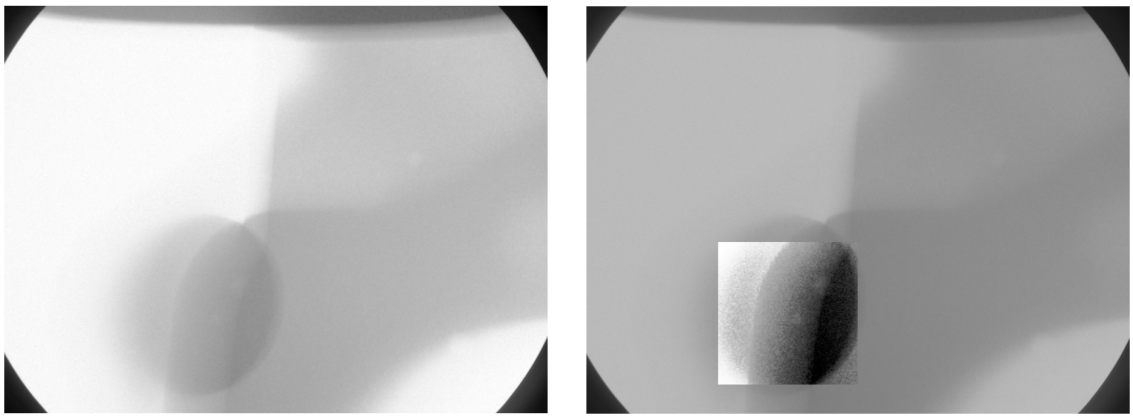

Fig. 9: Contrast enhancement by uniforming a histogram of the selected area.

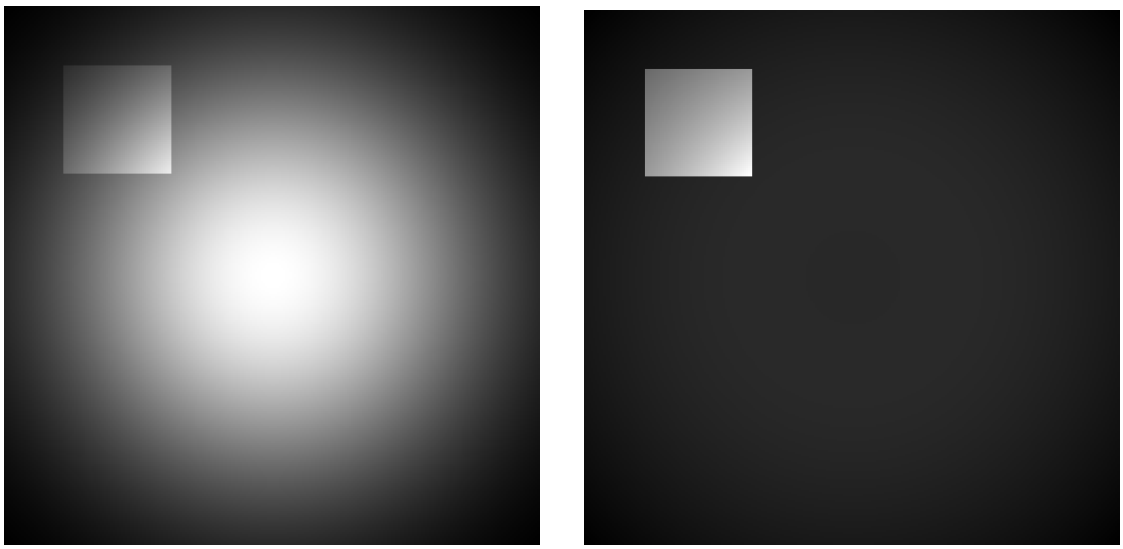

Fig. 10: Simulation of shading correction: Left) X-ray image for a plate with a square cavity ($\mathbf{X}$), Right) corrected image ($\mathbf{Y}$).

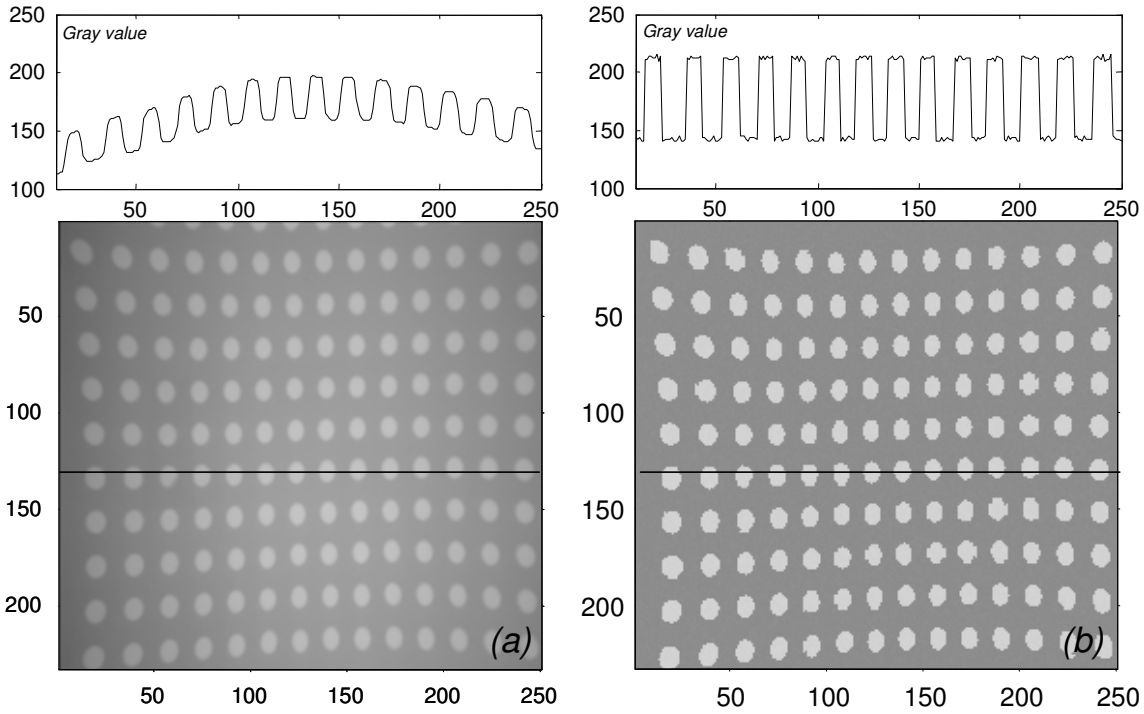

Fig. 11: Shading correction: a) original image, b) image after shading correction. The corresponding gray values profiles of row number 130 are shown above the images.

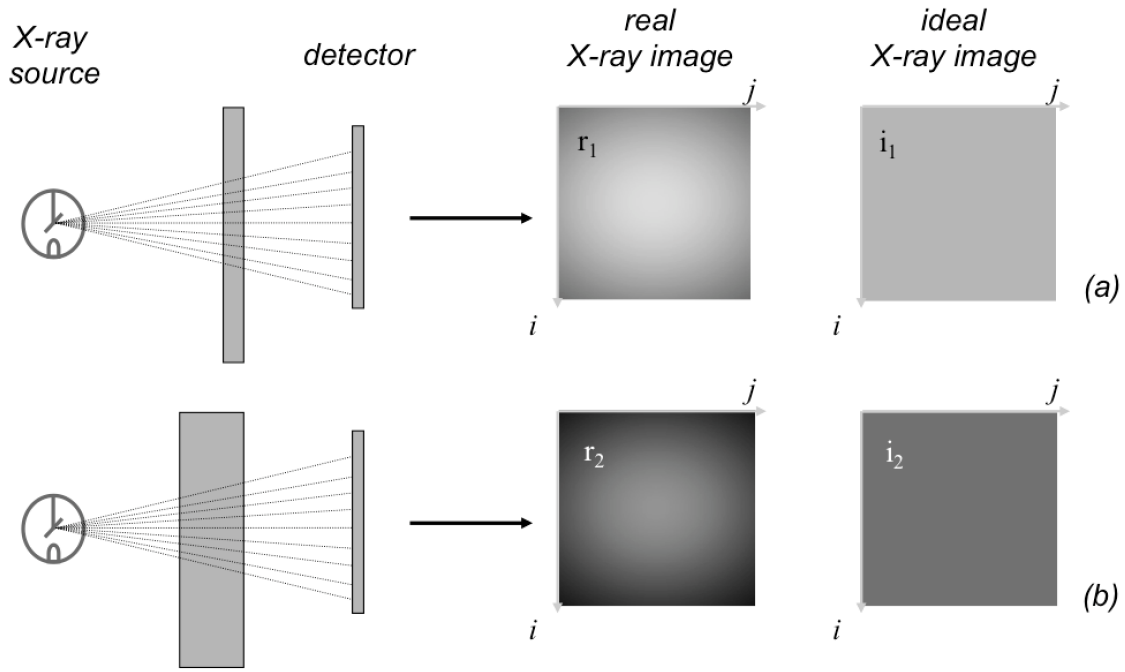

Fig. 12: Shading correction: a) X-ray image for a thin plate, b) X-ray image for a thick plate. Ideal X-ray images have a constant gray value.

where the offset and gain matrices are computed as

follows:

$$a(i,j) = \frac{i_2 - i_1}{r_2(i,j) - r_1(i,j)}$$

$$b(i,j) = i_1 - r_1(i,j) \cdot a(i,j). \quad (6)$$

An example of this technique is illustrated in Fig. 11b. In this case we obtain only two gray values (with noise) one for the aluminum part and another for the holes of the plate. In example of Fig. 10, we simulate image $\mathbf{X}$ (a plate with a square cavity). In addition, we simulate X-ray images $\mathbf{r}_1$ (a thin plate) and $\mathbf{r}_2$ (a thick plate) as illustrated in Fig. 12. The correction is evident: the appearance of the background is homogenous whereas the square is more distinguishable.

3.2 Image filtering

2D image filtering is performed in digital image processing using a small neighborhood of a pixel $X(i,j)$ in an input image to produce a new gray value $Y(i,j)$ in the output image, as shown in Fig. 13. A *filter mask* defines the input pixels to be processed by an *operator* f. The resulting value is the output pixel. The output for the entire image is obtained by shifting the mask over the input image. Mathematically, the image filtering is expressed as:

$$Y(i,j) = f[\underbrace{X(i-p, j-p), \ldots, X(i,j), \ldots, X(i+p, j+p)}_{\text{input pixels}}], \quad (7)$$

for $i = p{+}1 \ldots M{-}p$ and $j = p{+}1 \ldots N{-}p$, where M and N are the number of rows and columns of the input and output images. The size of the filter mask is, in this case, $(2p{+}1){\times}(2p{+}1)$. The operator f can be linear or non-linear. In this Section, the most important linear and non-linear filters for X-ray testing are outlined.

3.2.1 Linear filtering

The operator f is linear, if the resulting value $Y(i,j)$ is calculated as a linear combination of the input pixels:

$$Y(i,j) = \sum_{m=-p}^{p} \sum_{n=-p}^{p} h(m,n) \cdot X(i-m, j-n), \quad (8)$$

where $\mathbf{h}$ is called the *convolution mask*. The elements of $\mathbf{h}$ weight the input pixels. The convolution of an image $\mathbf{X}$ with a mask $\mathbf{h}$ can be written as:

$$\mathbf{Y} = \mathbf{X} * \mathbf{h}, \quad (9)$$

Averaging is a simple example of linear filtering. For a 3×3 neighborhood, the convolution mask is

$$\mathbf{h} = \frac{1}{9} \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$$

Gaussian mask can be used as well

$$h(m,n) = \frac{1}{2\pi\sigma^2} \cdot e^{-\frac{m^2+n^2}{2\sigma^2}} \quad (10)$$

scale factor $1/(2\pi\sigma^2)$ ensures $\sum_{m,n} h(m,n) = 1$ over all elements of $\mathbf{h}$.

A common application of filtering in X-ray testing is defect detection. Filtering out defects detected in an X-ray image will provide a reference *defect-free* image. The defects are detected by finding deviations in the original image from the reference image. The problem is how one can generate a defect-free image from the original X-ray image. Assuming that the defects will be smaller than the regular structure of the test piece, one can use a low-pass filter that does not consider the high frequency components of the image. However, if a linear filter is used for this task, the edges of the regular structure of the specimen are not necessarily preserved and many false alarms are raised at the edges of regular structures. Consequently, a non-linear filter is commonly used.

3.2.2 Non-linear filtering

In order to avoid the mentioned problems of linear filters, non-linear filters are used. Defect discrimination can be performed with a *median filter*. The median filter is a ranking operator (and thus non-linear) where the output value is the middle value of the input values ordered in a rising sequence [18]. For an even number of input numbers the median value is the arithmetic mean of the two middle values.

The application of a median filter is useful for generating the reference image because it smoothes noise yet preserves sharp edges, whereas other linear low-pass filters blur such edges (see a comparison with linear filters in Fig. 14). Hence, it follows that small defects can be suppressed while

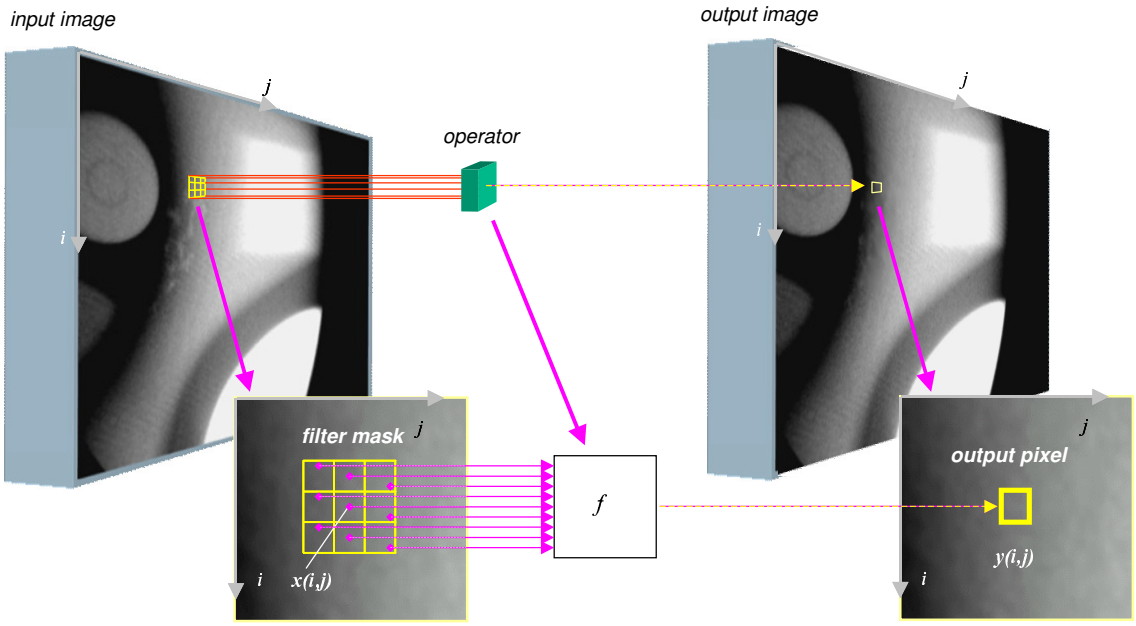

Fig. 13: Image filtering.

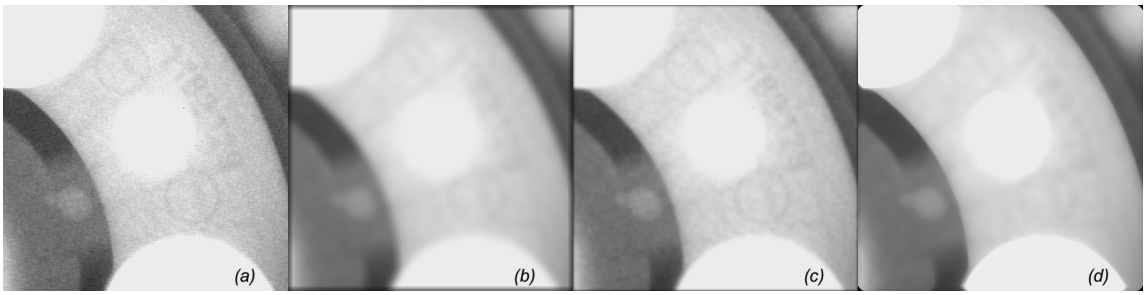

Fig. 14: Example of filtering of (a) an X-ray image of 400 $\times$ 400 pixels using (b) arithmetic, (b) Gaussian and (c) median filters with a mask of 15 $\times$ 15 pixels.

the regular structures are preserved. Fig. 15 shows this phenomenon for a 1D example. The input signal x is filtered using a median filter with 9 input elements, and the resulting signal is y. We can see that structures of length n greater than four cannot be eliminated. The third column shows the detection $x - y$. Large structures of $n \geq 5$ are not detected, as presented in the last two cases.

If the background captured by the median filter is constant, foreground structures could be suppressed if the number of values belonging to the structure is less than one half of the input value to the filter. This characteristic is utilized to suppress the defect structures and to preserve the design features of the test piece in the image.

An example for the application of a median filter on 2D signals (images) is shown in Fig. 16 and includes different structures and mask sizes compared to the effects of two linear low-pass filters. One can appreciate that only the median filter manages to suppress the relatively small structures completely, whereas the large patterns retain their gray values and sharp edges.

The goal of the background image function, therefore, is to create a defect-free image from the test image. A real example is shown in Fig. 17. In this example, from an original X-ray image $\mathbf{X}$ we generate a filtered image $\mathbf{Y}$ and a difference image $|\mathbf{X} - \mathbf{Y}|$. By setting a threshold, we obtain a binary image whose pixels are '1' (or white), where the gray values in the difference image are greater than the selected threshold. Finally, we eliminate very small regions. The remaining pixels correspond to the detected flaws.

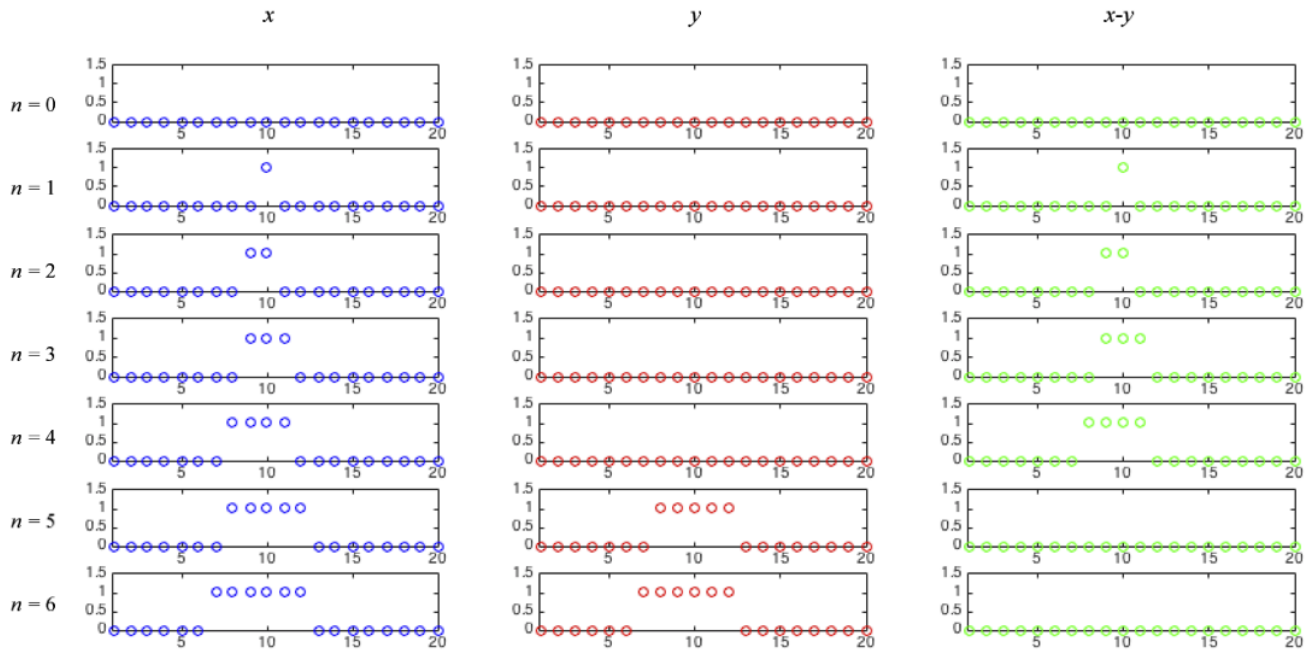

Fig. 15: Median filter application on a 1D signal x. The filtered signal is y (the size of the median mask is 9). Structures of length n less than 9/2 are eliminated in y. This filter can be used to detect small structures ($n \leq 4$).

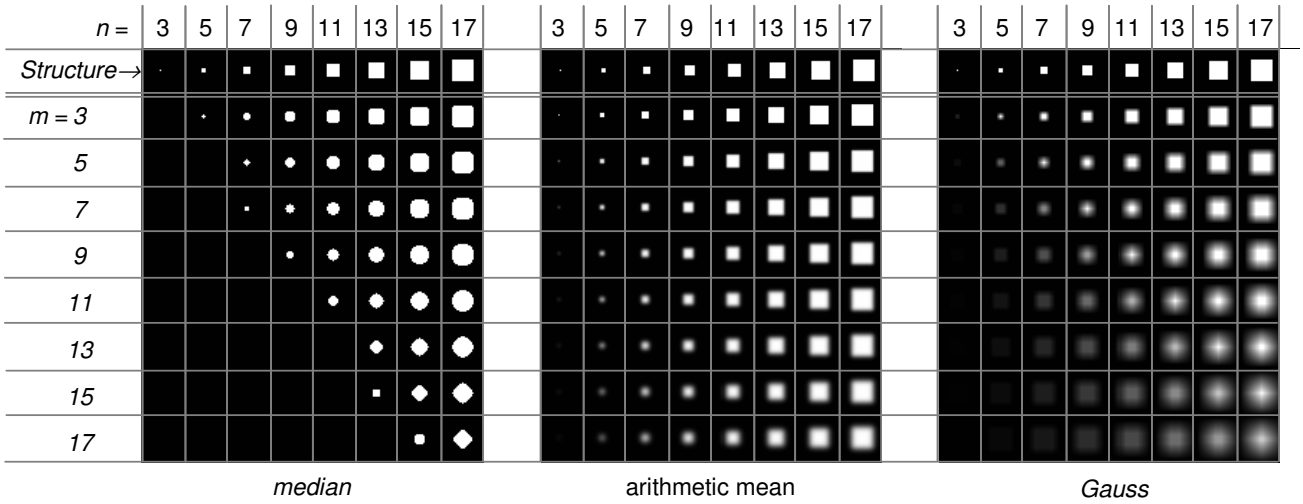

Fig. 16: Median filter application on an $n \times n$ structure using an $m \times m$ quadratic mask compared to average and Gauss low-pass filter application.

In example of Fig. 17 we detect small defects of an aluminum wheel. First, a reference defect-free image is estimated from original image itself using median filtering. Second, the difference between original and reference image is computed. Finally, defects are detected when the difference in gray values is high enough and the size of the detected region is large enough.

3.3 Edge detection

In this Section we will study how the *edges* of an X-ray image can be detected. The edges correspond to

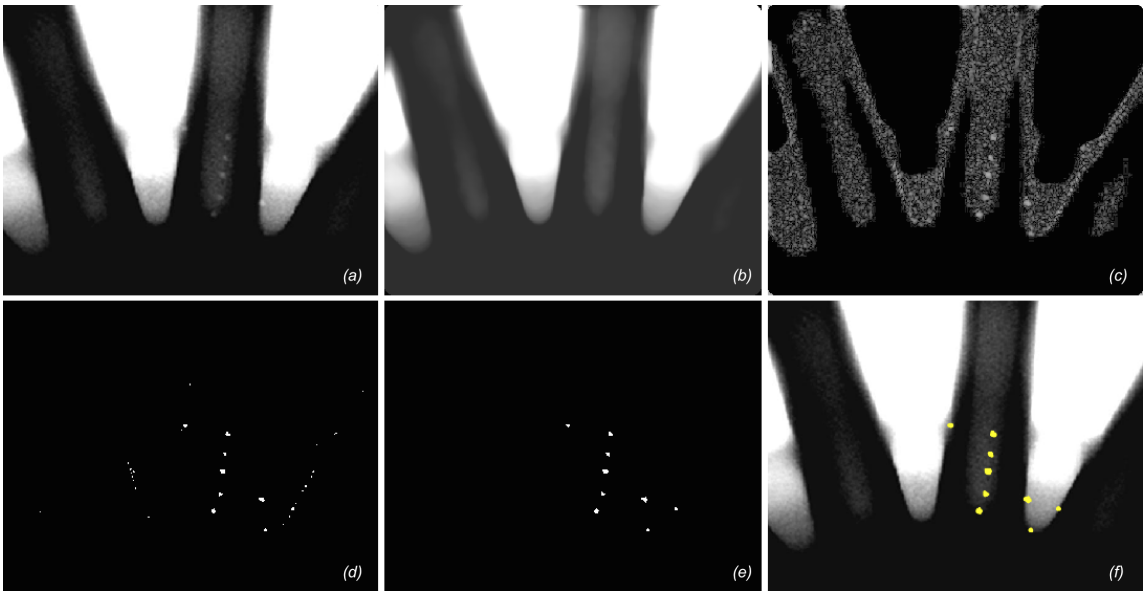

Fig. 17: Defect detection using median filtering: a) original X-ray image of an aluminum wheel with small defects, b) filtered X-ray image, c) difference image, d) binary image using a threshold, e) elimination of small regions, f) detection superimposed onto original image.

pixels of the image in which the gray value changes significantly over a short distance [8]. Since edges are discontinuities in the intensity of the X-ray image, they are normally estimated by maximizing the gradient of the image. Edge detection image corresponds to a binary image (of the same size of the X-ray image), where a pixel is '1' if it belongs to an edge, otherwise it is '0', as shown in Fig. 18. Before we begin a more detailed description of edge detection, it is worthwhile to highlight some aspects of its relevance in the analysis of X-ray images.

The edges of an X-ray image should show the boundary of objects, *e.g.*, boundaries of defects in control quality of aluminum castings, boundaries of the weld in welding inspection and boundaries of objects in baggage screening (Fig. 18). Thus, the input X-ray image is transformed into a binary image which shows structural properties of the X-ray image. The key idea is to detect objects of interest, such as defects in case of quality control or threatening objects in case of baggage screening, based on the information provided by edge detection.

In this Section, we will review some basic edge detection techniques that have been used in X-ray testing: gradient estimation (Section 3.3.1), Laplacian-of-Gaussian (Section 3.3.2) and Canny (Section 3.3.3). Segmentation techniques based on edge detection will be outlined in Section 3.4.

3.3.1 Gradient estimation

The gradient for a 1D function $f(x)$ is defined by:

$$f'(x) = \frac{\partial f}{\partial x} = \lim_{\Delta x \to 0} \frac{f(x + \Delta x) - f(x)}{\Delta x} \tag{11}$$

and for a 2D function $f(x, y)$ is defined by a vector of two elements, one in x direction and the another one in y direction:

$$\nabla f(x, y) = \left[\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y} \right]. \tag{12}$$

In digital images, after digitalization of $f(x, y)$, however, corresponding Δx or Δy values cannot be less than one pixel. A simple way to compute the gradient of image $\mathbf{X}$ in i and j direction can be respectively:

$$\begin{aligned} G_i(i, j) &= X(i + 1, j) - X(i, j) \\ G_j(i, j) &= X(i, j + 1) - X(i, j). \end{aligned} \tag{13}$$

Thus, the magnitude of the gradient can be computed as:

$$G(i, j) = \sqrt{(G_i(i, j))^2 + (G_j(i, j))^2} \tag{14}$$

and the direction of the gradient as:

$$A(i, j) = \arctan \frac{G_j(i, j)}{G_i(i, j)}. \tag{15}$$

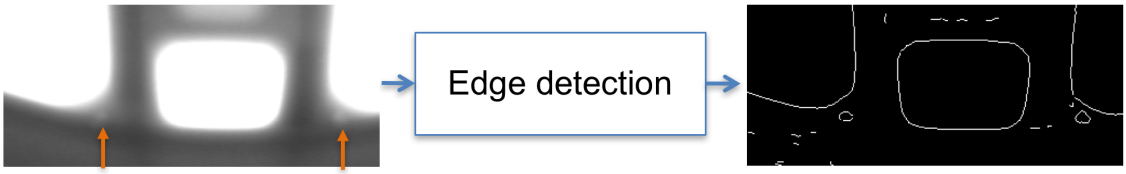

Fig. 18: Edge detection of an X-ray image of an aluminum wheel. The edges correspond to the boundaries of the image. The casting has two small defects (see arrows). The edges can be used to detect them.

In this formulation, gradient images $\mathbf{G}_i$ and $\mathbf{G}_j$ can be easily calculated by convolution (9). Thus,

$$\mathbf{G}_i = \mathbf{X} * \mathbf{h}^\mathsf{T} \quad \text{and} \quad \mathbf{G}_j = \mathbf{X} * \mathbf{h}. \tag{16}$$

where $\mathbf{h}$ is the mask used to compute the gradient in horizontal direction. For instance, if we compute the gradient using the simple way (13), we can use $\mathbf{h} = [-1 \quad +1]$ in (16). Nevertheless, for noisy images, larger masks are suggested for (16). Sobel and Prewitt masks are commonly used in image processing [18]. They are defined as follows:

$$\mathbf{h}_{\text{Sobel}} = \begin{bmatrix} -1 & 0 & +1 \\ -2 & 0 & +2 \\ -1 & 0 & +1 \end{bmatrix} \tag{17}$$

$$\mathbf{h}_{\text{Prewitt}} = \begin{bmatrix} -1 & 0 & +1 \\ -1 & 0 & +1 \\ -1 & 0 & +1 \end{bmatrix} \tag{18}$$

For severe noise, it is recommended to use Gaussian filtering before applying gradient operators. Since Gaussian and gradient operations are linear, the Gaussian gradient operator can be defined by taking the derivative of the Gaussian (10):

$$h_{\text{Gauss}}(m,n) = m \cdot e^{-\frac{m^2+n^2}{2\sigma^2}}. \tag{19}$$

It should be noted that edges are detected when the magnitude of the gradient is maximal. That means, the location of edge pixels will not be modified if a mask $\mathbf{h}$ is replaced by $\lambda\mathbf{h}$ with $\lambda \neq 0$. Moreover, the direction of the gradient does not become modified either. For this reason, the elements of $\mathbf{h}$ are usually shown in its simplest way.

An example of estimation of gradient using the explained masks is illustrated in Fig. 19. After the gradient image is calculated, the edges are detected by thresholding. Thus, if the magnitude of the gradient is greater than a certain threshold, then the pixel of the output image is set as an edge pixel. The output for the mentioned example is illustrated in Fig. 20. We can see how the boundaries are detected, especially for those objects that are very dark in comparison with their background.

In Fig. 19 and 20, we show the edge detection of an X-ray image of an aluminum casting using the gradient operators according to the method explained in this Section

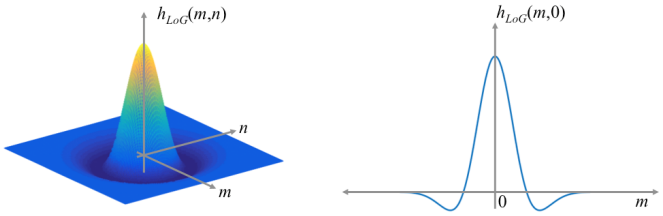

Fig. 22: LoG mask: Left) representation of (20), Right) profile for $n = 0$.

3.3.2 *Laplacian-of-Gaussian*

In the previous Section, we learned that the edges of a function can be located by detecting local maxima of the magnitudes of gradients. We know that the location of the maximal values of the gradient coincides with zero-crossing of the second derivative. In order to eliminate noisy zero-crossings, which do not correspond to high gradient values, this method uses a Gaussian low-pass filter (see Fig. 21). The method, known as Laplacian-of-Gaussian (LoG), is based on a kernel and a zero-crossing algorithm [14]. LoG-kernel involves a Gaussian low-pass filter (10), which is suitable for the pre-smoothing of the noisy X-ray images. LoG-kernel is defined as the Laplacian of a 2D-Gaussian function:

$$h_{LoG}(m,n) = \frac{1}{2\pi\sigma^4} \cdot (2 - \frac{m^2+n^2}{\sigma^2}) \cdot e^{-\frac{m^2+n^2}{2\sigma^2}}. \tag{20}$$

LoG-kernel is shown in Fig. 22. The parameter σ defines the width of the Gaussian function and, thus,

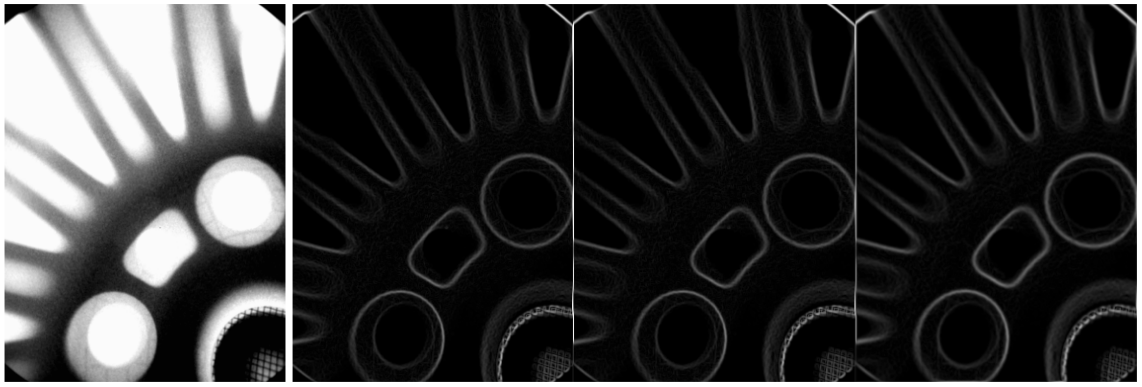

Fig. 19: Gradient of an X-ray of a pen case using different masks (Sobel, Prewitt, and Gaussian). See edge detection in Fig. 20.

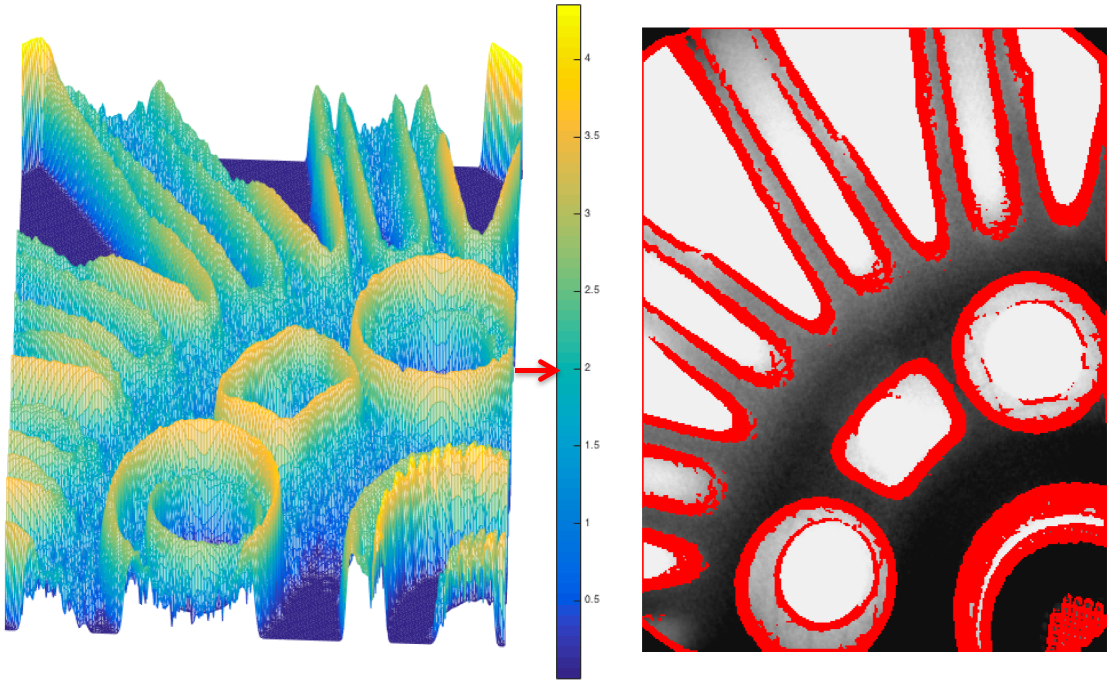

Fig. 20: Edge detection by thresholding a Gaussian gradient image of Fig. 19. The edges are detected for gradients greater than 3. In this representation a logarithmical scale for the gray values was used.

the amount of smoothing and the edges detected (see Fig. 23). Using (8) we can calculate an image $\mathbf{Y}$ in which the edges of the original image are located by their zero-crossing. After zero-crossing, the detected edges $\mathbf{Z}$ correspond to the maximal (or minimal) values of the gradient image. In order to eliminate weak edges, a threshold θ is typically used. Thus, all edge pixels in $\mathbf{Z}$ that are not strong enough are ignored. The higher the threshold, the less edges will be detected. On the other hand, if $\theta = 0$, *i.e.*, all zero-crossings are included, the edge image has closed and connected contours. As we will see in Section 3.4.2, this property is required when segmenting a region of the image.

In Fig. 23, we show the edge detection of the object of a part of a casting according to LoG algorithm explained in this Section.

3.3.3 *Canny edge detector*

Canny proposes a 2D linear mask for edge detection based on an optimization approach [6], in which the

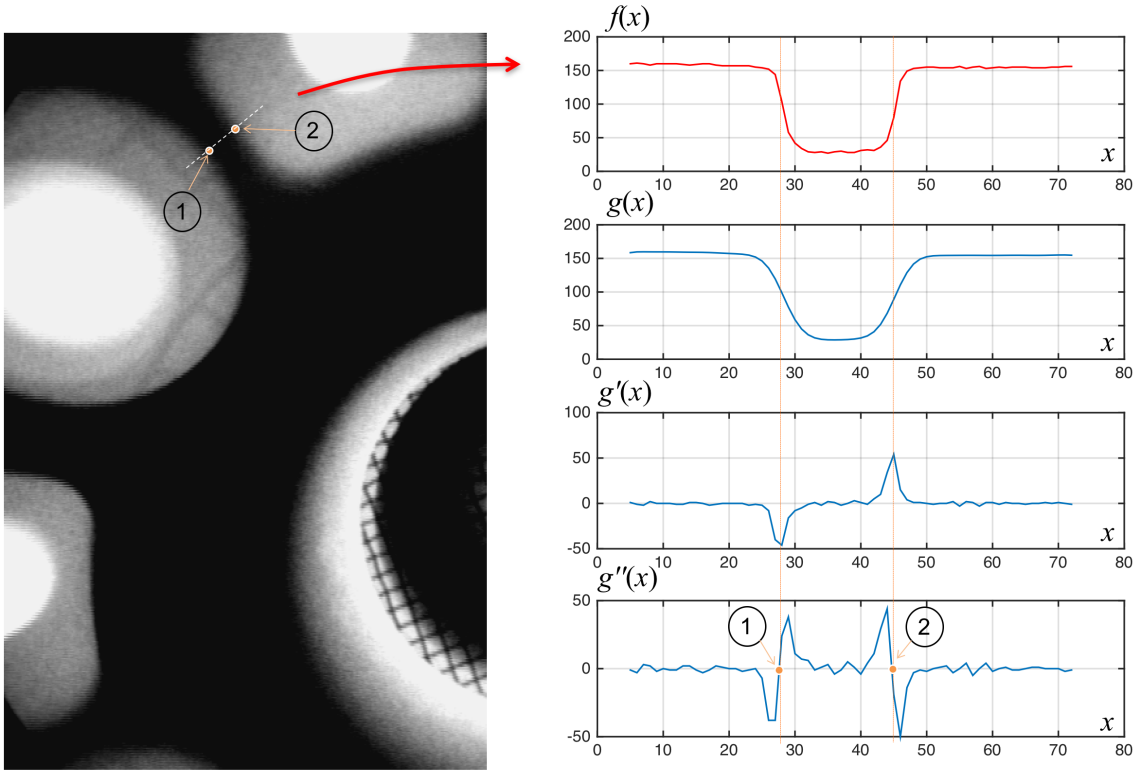

Fig. 21: Example of edge detection in 1D using LoG: The profile of the red line in an X-ray image is shown as $f(x)$. This function is filtered by a Gaussian low-pass filter obtaining $g(x)$. The gradient of $g(x)$, represented as $g'(x)$ shows the location of the maximal value (see dashed orange lines), that corresponds to the zero-crossing of the second derivative of $g(x)$. The edges '1' and '2' are then detected.

following criteria are met:

- Good detection: The detection should respond to an edge (and not to noise).
- Good localization: The detected edge should be near the true edge.
- Single response: It should be one detected edge per true edge.

The optimal mask is similar to a derivative of a Gaussian. Thus, the idea is to use this mask to find the local maxima of the gradient of the image. The practical implementation uses adaptive thresholding of the gradient (to detect strong and weak edges) with hysteresis (weak edges are detected only if they are connected to strong edges).

3.4 Segmentation

Image segmentation is defined as the process of subdividing an image into disjointed regions [8]. A region is defined as a set of connected pixels that correspond to a certain *object of interest*. Obviously,

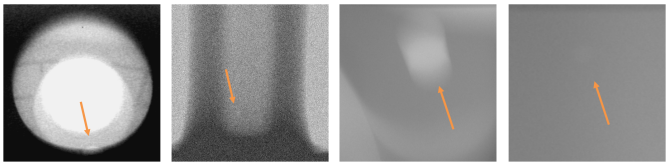

Fig. 25: Problems when detecting a gun. Detection of defects (see arrows) can be a very complex task due to a) occlusion, b) noise, c) blur, d) low contrast.

these regions of interest depend on the application. For instance, in the inspection of aluminum castings with X-ray images, the idea of segmentation is to find regions with defects. Here, the object of interest is the defects. An example is shown 24, where the segmentation are the small spots that indicate defective areas.

Segmentation is one of the most difficult processes in image processing. Clearly, there are some simple applications in which certain segmentation

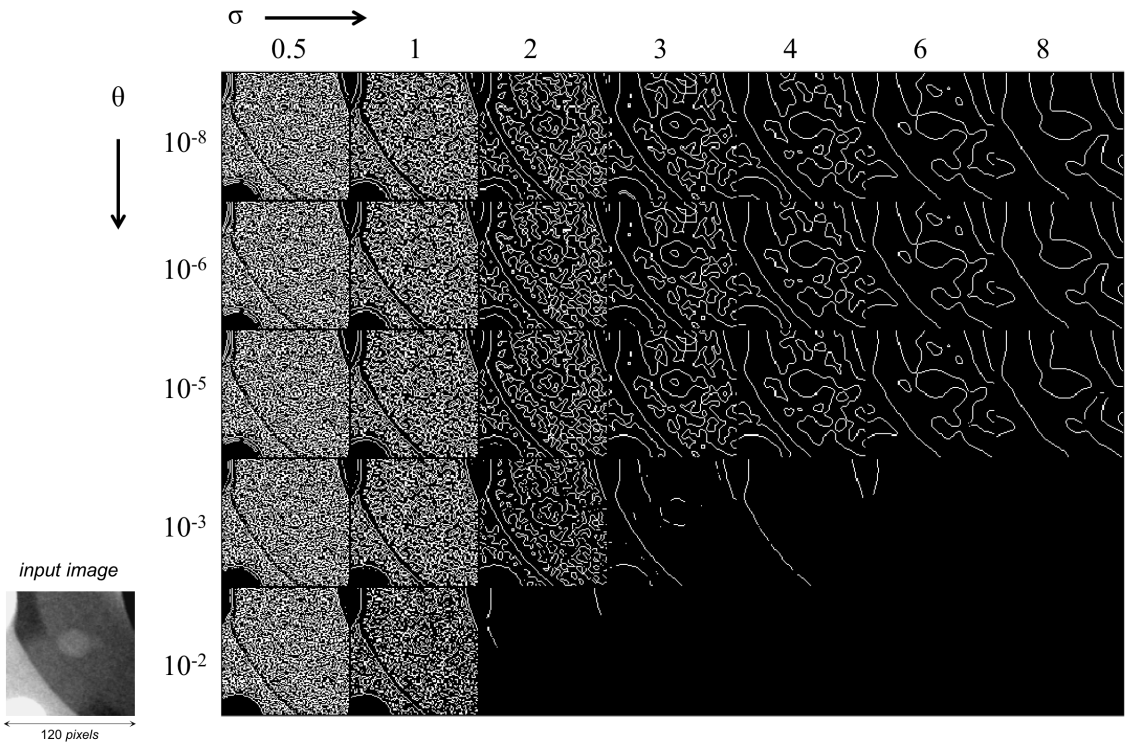

Fig. 23: Example of LoG edge detection of a defect. Several values for σ and θ are presented. The smoothness of the edges is controlled by increasing σ. The reduction of noisy edges is controlled by increasing θ.

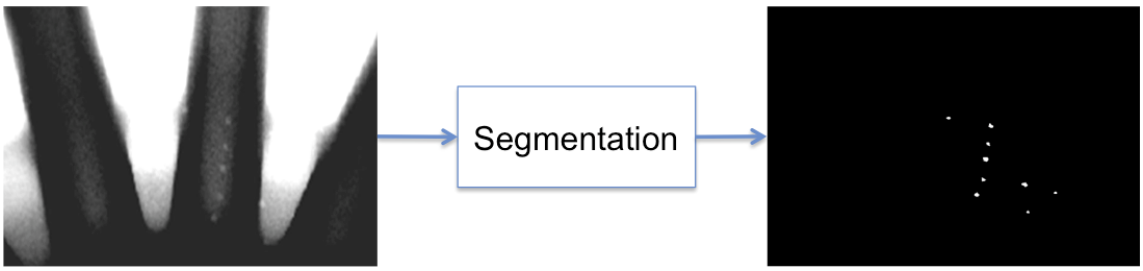

Fig. 24: Example of segmentation: detection of defects in an aluminum wheel (see details in Fig. 17).

techniques are very effective (*e.g.*, separation between a casting and its background as shown in Fig. 28), however, in many other applications segmentation is far from being solved as the appearance of the object of interest can become very intricate. This is the case of defect detection, where the segmentation of discontinuities can be extremely difficult due to problems of occlusion, noise, blur and low contrast (see Fig. 25).

In image processing for fault detection, segmentation is used to detect (potential) regions that can be the objects of interest that we are looking for. As mentioned in previous examples, segmentation divides the X-ray image into two areas: foreground and background. Foreground means the pixels of the object(s) of interest. Background means the remaining pixels of the image. Usually, a *binary image* is the output of the segmentation process as we can see in Fig. 24 and 28: where a pixel equals to '1' (white) is foreground, whereas '0' (black) means background. We use the term 'potential' throughout to make it clear that a segmented region is not necessarily the final detected region. In many applications, the segmentation is just the first step of the whole detection process. In such cases, an additional step that analyzes the segmented region is required. This additional step can include multiple view analysis or a pattern recognition technique.

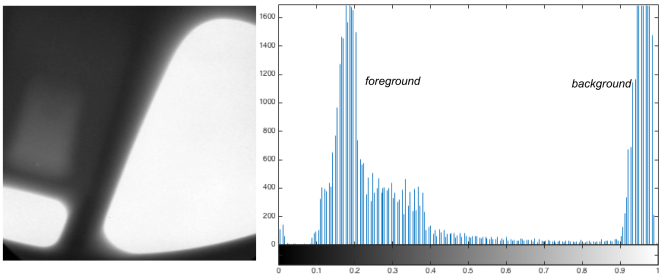

Fig. 26: X-ray image of a wheel and its histogram.

The later extracts and classifies features of the segmented region in order to verify whether it corresponds to the object that we are detecting or it is a *false detection*.

Thus, segmentation basically acts as a focus of attention mechanism that filters the information that is fed to the following steps, as such a failure in the segmentation is catastrophic for the final performance. In this Section, we will review some basic segmentation techniques that have been used in X-ray testing: thresholding (Section 3.4.1), region growing (Section 3.4.2) and maximally stable extremal regions (Section 3.4.3).

3.4.1 Thresholding

In some X-ray images, we can observe that the foreground is significantly darker than the foreground. This is the case of an X-ray image of a casting placed with holes as illustrated in Fig. 26. It is clear that the object of interest can be segmented using a very simple approach based on *thresholding*. In this Section, we will explain a methodology based on a estimation of a global threshold using a statistical approach. This method was originally presented for color food images [57], however, it can be easily adapted for X-ray images.

The X-ray image to be segmented is stored in matrix $\mathbf{I}$. In order to enhance the contrast of the image, a linear transformation can be performed (see Section 3.1.2). Additionally, a linear or non-linear filter can be used for noise removal (see Section 3.2). Here, after image enhancement and filtering, we obtain a new image $\mathbf{J}$ where $J_{\max} = 1$ and $J_{\max} = 0$. Image $\mathbf{J}$ has a bimodal histogram as shown in Fig. 26, where the left distribution corresponds to the casting and the right to the background. In this image, a first separation between foreground and background can be performed estimating a global threshold t. Thus, we define a binary image

$$K(i,j) = \begin{cases} 1 & \text{if} \quad J(i,j) < t \\ 0 & \text{else} \end{cases} \tag{21}$$

where '1' means foreground and '0' background, that define two classes of pixels in the image. Fig 27 illustrates different outputs depending on t. The problem is to determine the 'best' threshold t that separates the two modes of the histogram from each other. A good separation of the classes is obtained by ensuring *i)* a small variation of the gray values in each class, and *ii)* a large variation of the gray values in the image [22]. The first criterion is obtained by minimizing a weighted sum of the within-class variances (called *intraclass* variance $\sigma^2_W(t)$):

$$\sigma^2_W(t) = p_b(t)\sigma^2_b(t) + p_f(t)\sigma^2_f(t) \tag{22}$$

where the indices 'b' and 'f' denote respectively background and foreground classes, and p and σ^2 are respectively the probability and the variance for the indicated class. These values can be computed from the histogram.

The second criterion is obtained by maximizing the between-class variance (called *interclass* variance $\sigma^2_B(t)$):

$$\sigma^2_B(t) = p_b(\mu_b(t) - \mu)^2 + p_f(\mu_f(t) - \mu)^2 \tag{23}$$

where μ_b, μ_f and μ indicate the mean value of the background, foreground and the whole image respectively.

The best threshold t can be estimated by a sequential search through all possible values of t that minimizes $\sigma^2_W(t)$ (or maximizes $\sigma^2_B(t)$). Both criteria, however, lead to the same result because the sum $\sigma^2_W + \sigma^2_B$ is a constant and corresponds to the variance of the whole image [22]. Thus, the threshold can be computed by minimizing the intraclass variance $\sigma^2_W(t)$. In our example, the obtained threshold is $t = 0.5882$, that is approximately 0.6 (see Fig. 27).

In some cases, we can observe that the segmentation suffers from inaccuracy because there are many dark (bright) regions belonging to the foreground (background) that are below (above) the chosen threshold and therefore misclassified. For this reason, additional morphological processing such as *i)* remove small objects and *ii)* close the binary image can be performed.

In the first step, we remove from binary image $\mathbf{K}$ obtained from (21) all connected regions that have

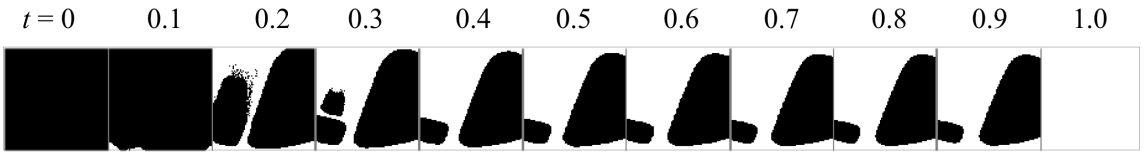

Fig. 27: Segmentation using threshold $t = 0.1, 0.2, \ldots 1.0$.

fewer than n pixels. This operation is necessary to eliminate those isolated pixels of the background that have a gray value greater than the selected threshold. Empirically we set $n = NM/100$, where $N \times M$ is the number of pixels of the image.

The second step *closes* the image, *i.e.*, the image is *dilated* and then *eroded*. The dilation is the process that incorporates into the foreground the background pixels that touch it. On the other hand, erosion is the process that eliminates all the boundary pixels of the foreground. The closing process (dilation followed by erosion) fills small holes and thins holes in the foreground, connecting nearby regions, and smoothing the boundaries of the foreground without changing the area significantly [8]. This operation is very useful in objects that have spots in the boundary. The result is shown in Fig. 28.

3.4.2 Region growing

In region growing, we segment a region using an iterative approach. We start by choosing a seed pixel, as shown in Fig. 29. At this moment, our region is initialized and its size is one pixel only. We extract some feature of the region, *e.g.*, the gray value. We extract the same feature of each neighboring pixel. In our example, there are four neighbors (up, down, right and left), as we can see in third image of Fig. 29. We increase our region by adding similar neighboring pixels, *i.e.*, those neighboring pixels that have a similar feature to the region. The whole process is continued, each added pixel is a new seed for the next iteration, until no more neighboring pixels can be added.

In Fig. 29, we have a binary edge image. The feature that we use to establish the similarity is the value of the pixel. In our example, there are only two pixel values: '0' for the edge pixels, and '1' for the remaining pixels. That means, that the value of the pixel of the seed is '1' and in each iteration, we can add only those neighboring pixels the value of which are '1'. As we can see, the red region grows up from 1 pixel to 5, 12, 16, 22, and finally 24 pixels. The output is the red region of the last step.

Region growing can be used directly in X-ray images as illustrated in Fig. 30. We start with a seed pixel, and neighboring pixels are added if they are similar enough.

In the following example, we show the performance of region growing in the segmentation of a part of a casting (see Fig. 30). The seed is chosen at pixel (100,60). The seed grows by adding neighboring pixels with similar gray values. In this example, the similarity between region and neighboring pixels is established if $|\bar{R} - r_n| \leq \theta$, where $\bar{R}$ is the average of the gray values of the region, r_n is the gray value of the neighboring pixel, and θ is a threshold. In this example, $\theta = 20$. The output of this example is shown in Fig. 30.

Region growing can be used in X-ray testing in defect detection (see for example interesting approaches in aluminum castings [45] and welds [58]). The method is illustrated in Fig. 33. The method uses an edge detection algorithm to obtain an edge image with closed and connected contours around the real defects. Thus, we use region growing to isolate each region enclosed by edges. The idea is to extract features from this isolated region (*e.g.*, area, average of gray value, contrast, etc.) that can be used in a classification strategy. In our example, a region is segmented using a very simple classifier (the features of a segmented region must be in certain ranges, *e.g.*, $A_{\min} \leq \text{Area} \leq A_{\max}$). Obviously, more sophisticated features and classifiers can be used to improve the segmentation performance in more complex scenarios as we will see in the following chapters.

In the following example, we show how to segment defects in aluminum castings using binary images of potential defects and some simple features that can be extracted from each potential region. In this example, we segment all those regions the area of which is between 200 and 2000 pixels, the average of the gray value is less than 150, and

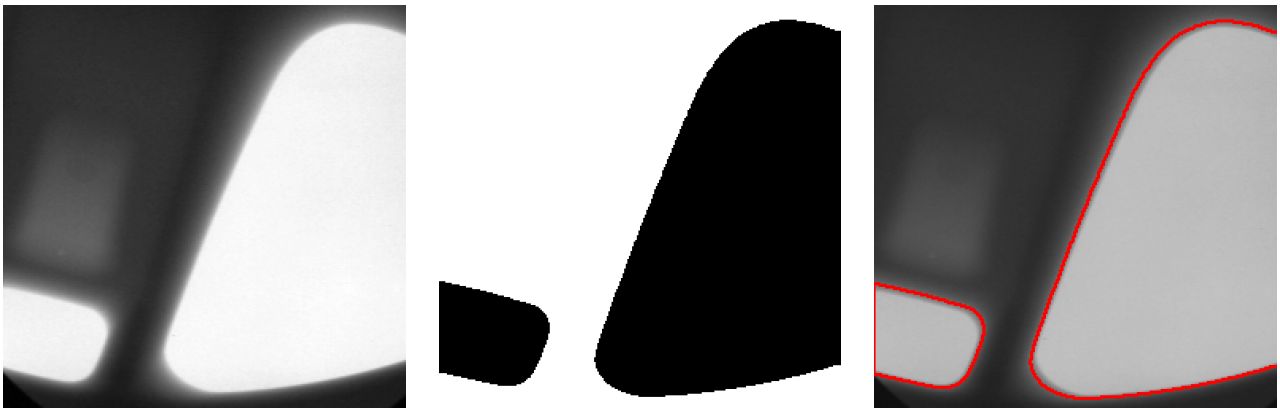

Fig. 28: Segmentation of a wheel using a global threshold.

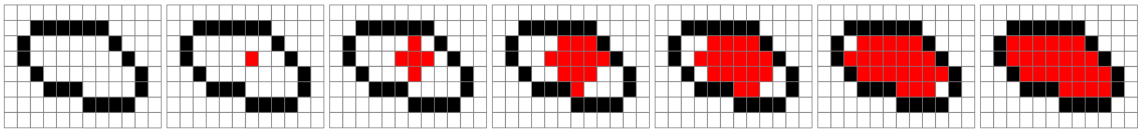

Fig. 29: Region growing: we start with a seed pixel that grows in each iteration in four directions until a boundary is found. The directions in this example are four: up, down, right and left.

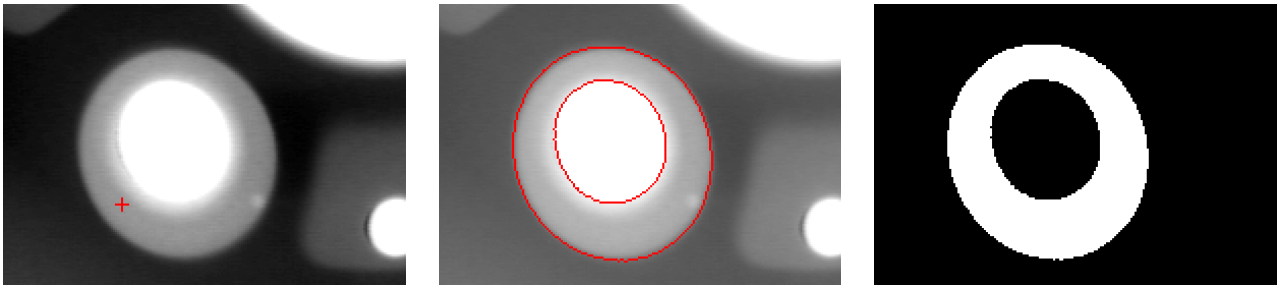

Fig. 30: Region growing in an X-ray image using a seed pixel in the object of interest (a regular structure). The region is well segmented as we can see in the binary image and in boundaries.

the contrast is greater than 1.1. The output of this example is shown in Fig. 33.

This method is very effective for regions of interest that have gray values significantly different from the background.

Nevertheless, the method may fail if the boundaries do not close a region of interest. This is the case in some defects of aluminum castings that are at an edge of a regular structure as illustrated in Fig. 31[3]. In this problem, we can see that the edges of LoG algorithm (and other edge detection algorithms like Sobel or Canny as well) cannot correctly find the defect's edge. Contrarily, it finds the regular structure's edge. To overcome this problem, we have to complete the remaining edges of these defects. A simple approach was suggested in [52] by thickening of the edges of the regular structure after LoG-edge detection: *i)* The gradient of the original image is calculated. The gradient image is computed by taking the square root of the sum of the squares of the gradient in horizontal and in vertical directions (see Section 3.3.1). These are calculated by the convolution of the radioscopic image with the first derivative (in the corresponding direction) of the Gaussian low-pass filter used in the LoG filter. *ii)* High gradient pixels are detected by thresholding. *iii)* The resulting image is added to the LoG-edge detection image. Afterwards, each closed region is segmented as a potential flaw. As can be observed the effectiveness of this method in Fig. 31, the

3. A video of this small defect can be watched at http://youtu.be/e3wDJhq2Tqg.

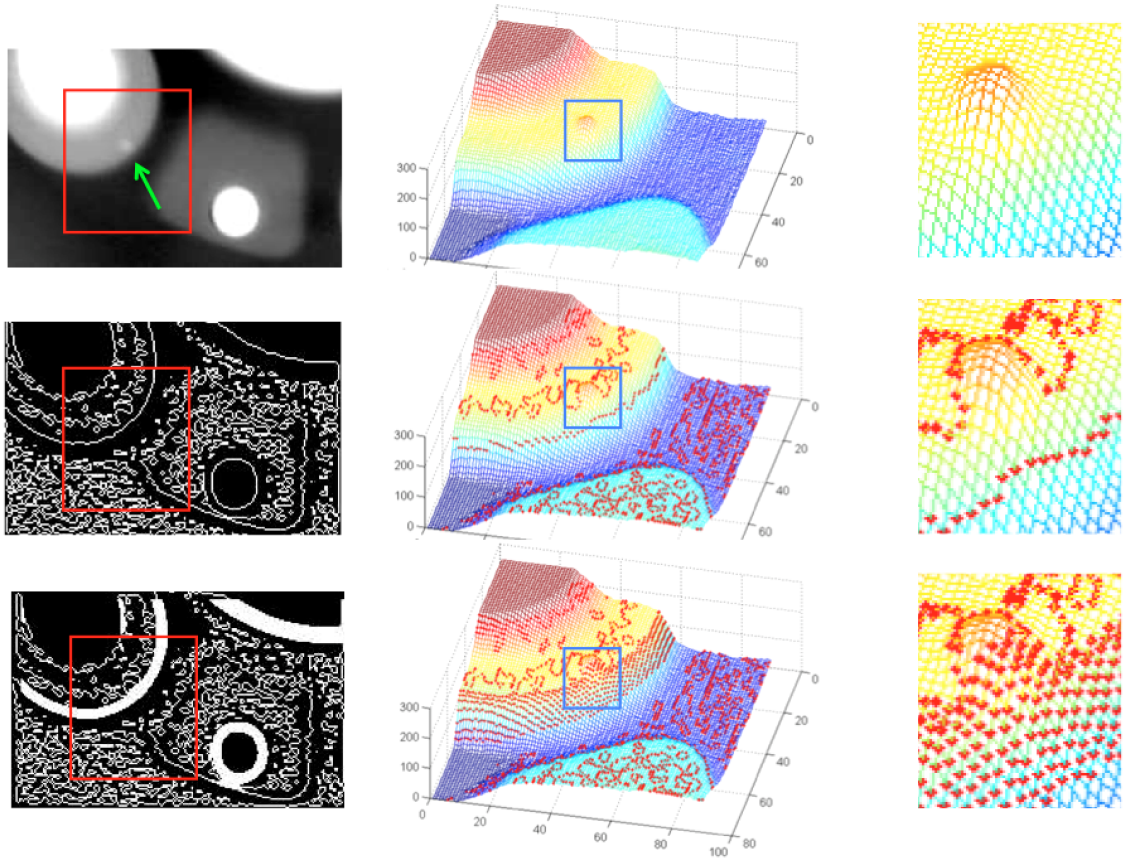

Fig. 31: X-ray image of an aluminum casting with a small defect at an edge (see defect pointed by green arrow). First row) original image. Second row) LoG. Third row) LoG and high gradient pixels. First column) image representation. Second column) 3D representation of red square. Third column) zoom of blue square. In this representation, the edge pixels are represented as red points superimposed onto the 3D surface. The output of this method is a binary image in which the real defects are closed by edges.

defect on an edge of a regular structure could be satisfactory closed. Thus, the method of Fig. 33 can be used.

3.4.3 *Maximally stable extremal regions*

In order to understand the MSER approach [42], the reader can imagine a simple video as follows. The video will have 256 frames. Frame t is defined as the binary image $\mathbf{I} < t$, where $\mathbf{I}$ is the input image we want to segment. If the binary image is black for '0' and white for '1', at the beginning our video will be very dark and at the end very bright. In the middle, we will have some regions depending on the threshold[4]. Thus, each region has an area $A(t)$, that depends on t. If the gray value of the region is very different from its background, the area of this region will be stable for some thresholds $t, t+1 \dots t+p$, *i.e.*, $A(t) \approx A(t+1) \cdots \approx A(t+p)$. The key idea of MSER, is to segment those regions which fulfill:

$$\frac{\Delta A}{\Delta t} < \theta, \tag{24}$$

where θ is a threshold. That means, those regions whose sizes remain approximately stable by varying the segmentation threshold t are to be detected.

In the following example we show the segmentation of an X-ray image of a small defect according to MSER approach. The output of this example is shown in Fig. 32.

3.5 Image restoration

Image restoration involves recovering detail in severely blurred images. This process is more efficient when the causes of the imperfections are

4. The video can be found in http://youtu.be/tWdJ-NFE6vY.

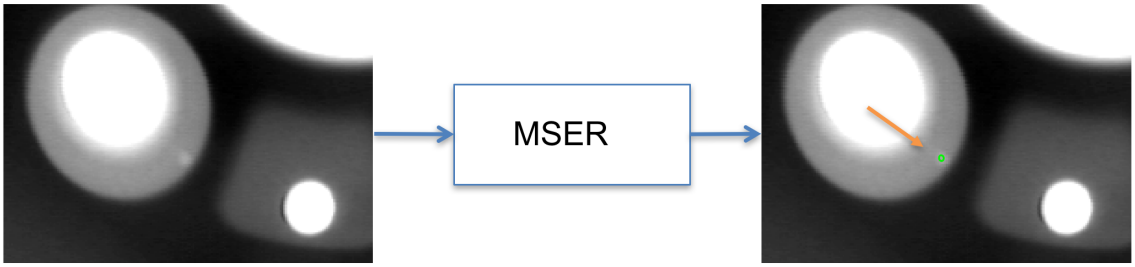

Fig. 32: Example of MSER: detection of a small defect at an edge of a regular structure of a casting.

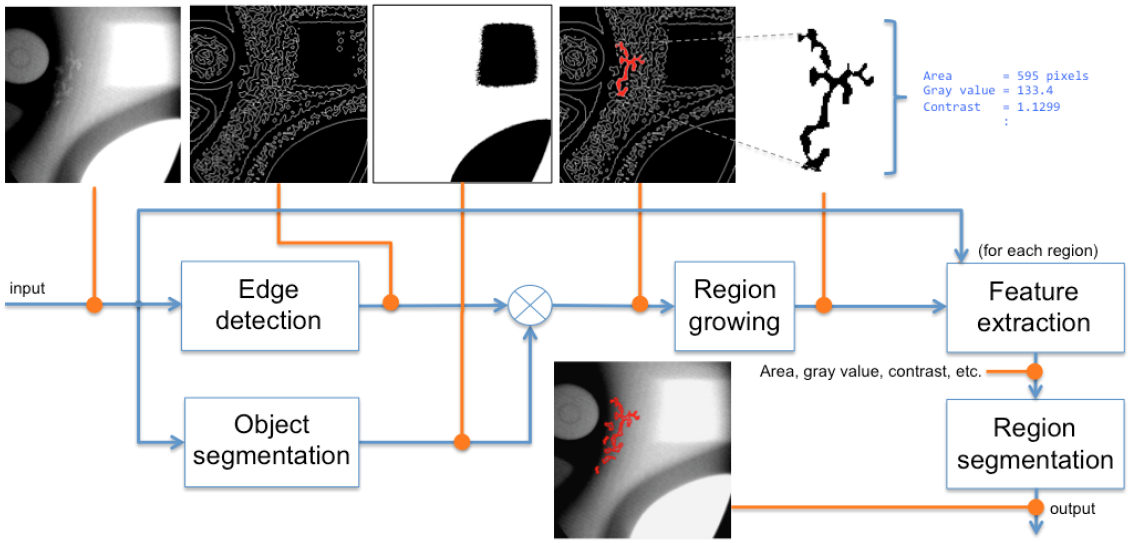

Fig. 33: Segmentation of defects in aluminum castings using region growing, edge detection and some features. The size of the image in this example is 286 $\times$ 286 pixels.

known a-priori [18]. This knowledge may exist as an analytical model, or as a-priori information in conjunction with knowledge (or assumptions) of the physical system that provided the imaging process in the first place. The purpose of restoration then is to estimate the best source image, given the blurred example and some a-priori knowledge.

In this Section, we concentrate on the particular case of blur caused by uniform linear motion, which may be introduced by relative motion between detector and object. Early work on restoring an image degraded by blurring calculated the deblurring function as an inverse filtering. The inverse filtering evaluation of the blurring function h (or point spread function PSF) in the frequency domain tends to be very sensitive to noise [18]. The cause of this sensitivity is the lowpass nature of the PSF: its frequency response $H(\omega)$ contains very small values, and small noise in the frequency regions where $1/H(\omega)$ is very large, may be greatly emphasized. Sondhi [18], proposed a non-iterative algorithm to find a solution to the uniform-blurring case, but the computational load is extremely high in small motions. Another two non-iterative approaches are presented in [43]. In the first one, the matrix left division calculates the restored signal as a signal that has the fewest possible nonzero components. This solution differs strongly from the original signal because the original signal must not have necessarily many zero components. The second one, the Moore-Penrose pseudo-inverse of a matrix, finds a restored signal whose norm is smaller than any other solution. This solution is very good, but the estimation is based on Singular-Value Decomposition (SVD), whose computation load is very high. In this Section, we address the above problems and reduce the computational times significantly using a new technique that minimize the norm between blurred and original.

A blurred X-ray image $g(x, y)$ that has been degraded by a motion in the vertical direction x and

the horizontal direction y can be modeled by:

$$g(x,y) = \frac{1}{T}\int_0^T f(x - x_t(t), y - y_t(t))dt, \quad (25)$$

where f, T, $x_t(t)$ and $y_t(t)$ represent, respectively, the deterministic original X-ray image, the duration of the exposure and the time-varying component of motion in the x and y directions. In this case the total exposure is obtained by integrating the instantaneous exposure over the time interval during which the shutter is open. By rotation of the camera or by using a transformation that rotates the blurred image, a new system of coordinates is chosen in which $x_t(t)$ is zero. Considering that the original image $f(x,y)$ undergoes uniform linear motion in the horizontal direction y only, at a rate given by $y_t(t) = ct/T$, let us write (25), with $u = y - ct/T$, as:

$$g(y) = \frac{1}{T}\int_0^T f(y - ct/T)dt = \frac{1}{c}\int_{y-c}^{y} f(u)dt, \quad (26)$$

or as a digital that has been discretized in spatial coordinates by taking N samples $\Delta y = Y/N$ units apart:

$$g_k = \frac{1}{n}\sum_{i=0}^{n-1} f_{k+i}, \quad (27)$$

where

$$g_k = g\left(y_0 + (k-1)\frac{c}{n}\right),$$

$$f_k = f\left(y_0 + (k-1)\frac{c}{n} - c\right). \quad (28)$$

with $n = c/\Delta y$. Fig. 34 shows a row $\mathbf{f} = [f_1 \ ... \ f_M]^\mathsf{T}$ of an original image and its corresponding row $\mathbf{g} = [g_1 \ ... \ g_N]^\mathsf{T}$ of the blurred image for $n = 3$ pixels. Equation (27) describes an underdetermined system of N simultaneous equations (one for each element of vector $\mathbf{g}$) and $M = N + n - 1$ unknowns (one for each element of vector $\mathbf{f}$) with $M > N$. This process is carried out for each row of the image. The degradation of $\mathbf{f}$ can be modeled using a convolution of $\mathbf{f}$ with $\mathbf{h}$, where $\mathbf{h}$ is the PSF, a n-element vector defined as the impulse response of this linear system [18]. Thus, element g_i of vector $\mathbf{g}$ is calculated as a weighted sum of n elements of $\mathbf{f}$, *i.e.*, $g_i = h_1 f_i + h_2 f_{i+1} + ... + h_n f_{i+n-1}$, for $i = 1, ..., N$. Using a circulant matrix, the convolution can be written as $\mathbf{Hf} = \mathbf{g}$:

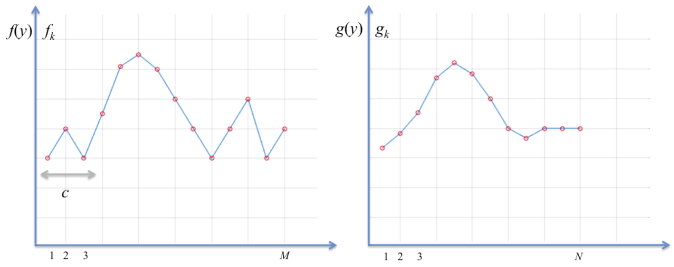

Fig. 34: Blurring process: Left) original row f. Right) Blurred row.

$$\mathbf{g} = \mathbf{f} * \mathbf{h} = \quad (29)$$

$$\begin{bmatrix} h_1 & ... & h_n & 0 & 0 & 0 & 0 \\ 0 & h_1 & ... & h_n & 0 & 0 & 0 \\ & \vdots & & & \vdots & & \\ 0 & 0 & ... & 0 & h_1 & ... & h_n \end{bmatrix} \begin{bmatrix} f_1 \\ f_2 \\ \vdots \\ f_M \end{bmatrix} = \begin{bmatrix} g_1 \\ g_2 \\ \vdots \\ g_N \end{bmatrix}$$

An example of a degradation process is shown in Fig. 35. In this example we can see how the vertical lines of the casting cannot be recognized when the degradation is severe.

If the PSF is not exactly known, but if we know that it corresponds to a uniform linear motion, the parameter n can be estimated from the spectrum of the blurred image. An example is shown in Fig. 36. The 2D-Fourier Transformation of a blurred test is represented in Fig. 36a, in this case a horizontal degradation took place with $n = 25$. The mean of its rows is illustrated in Fig. 36b. We can observe that the period of this function is inversely proportional to the length of the blurring process in pixels.

The problem of restoring an X-ray that has been blurred by uniform linear motion consists of solving the underdetermined system (29). The objective is to estimate an original row per row ($\mathbf{f}$), given each row of a blurred ($\mathbf{g}$) and a priori knowledge of the degradation phenomenon ($\mathbf{H}$). Since there is an infinite number of exact solutions for $\mathbf{f}$ in the sense that $\mathbf{Hf} - \mathbf{g} = \mathbf{0}$, an additional criterion that finds a sharp restored is required.

We observed that most solutions for $\mathbf{f}$ strongly oscillate. Fig. 37 shows an example in which four different solutions for $\mathbf{f}$ are estimated, all solutions satisfy equation (29): $\mathbf{Hf} = \mathbf{g}$. Although these solutions are mathematically right, they do not correspond to the original signal. By the assumption that the components of the higher frequencies of

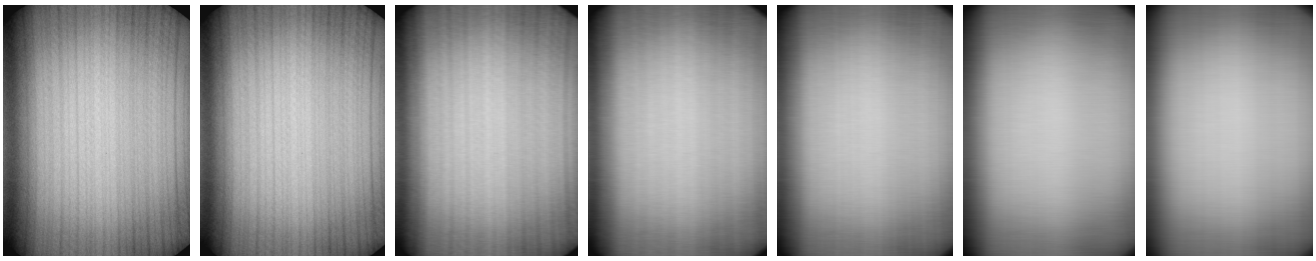

Fig. 35: Degradation of an X-ray image of of an aluminum wheel (768 $\times$ 572 pixels): Original image and degraded images with n = 5, 15, 25, 25, 45 and 60.

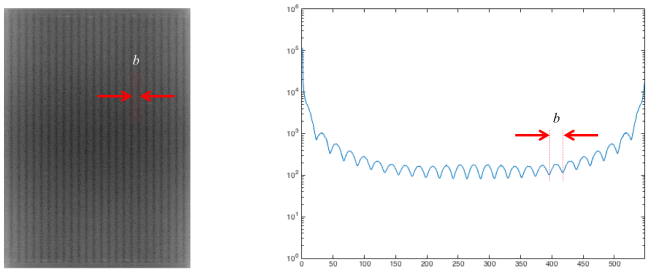

Fig. 36: Spectrum of a blurred image which was degraded by uniform linear motion with $n = 25$ pixels. Left) 2D-Fourier Transformation of the original image of Fig. 35. Right) Mean of the rows of the Fourier transformation. The size of the degraded image is 768 $\times$ 548 pixels. It can be demonstrated that b is inversely proportional to n.

$\mathbf{f}$ are not so significant in the wanted solution, these oscillations can be reduced by minimization of the distance between f_k and g_k, *i.e.*, we take a vector as a sharp solution of $\mathbf{Hf} = \mathbf{g}$ so, that this presents the smallest distance between original

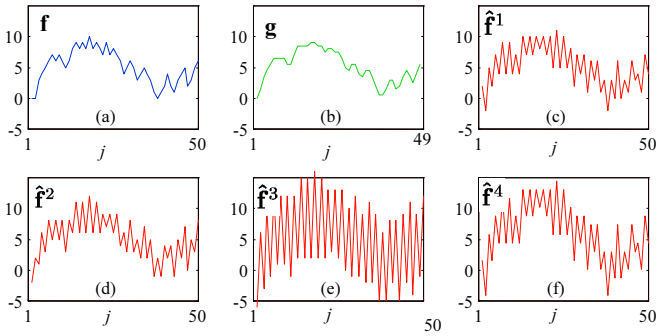

Fig. 37: Restoration of row $\mathbf{f}$: a) original row, b) degraded row with $n = 2$, c), d) e) and f) four possible solution that satisfy $\mathbf{H}\hat{\mathbf{f}} = \mathbf{g}$.

signal and blurred signal: we seek then to minimize the objective function

$$J(\mathbf{f}, \mathbf{g}) = \sum_{k=1}^{N} (f_k - g_k)^2 \rightarrow \min . \qquad (30)$$

The application of criteria of the minimization of the norm between input and output (MINIO) does not mean that $\mathbf{f}$ is equal to $\mathbf{g}$, because this solution does not satisfy the system of equations (29) and the size of $\mathbf{f}$ and $\mathbf{g}$ are different. The solution also is defined as the vector in the solution space of the underdetermined system $\mathbf{Hf} = \mathbf{g}$ whose first N components has the minimum distance to the measured data, *i.e.*, where the first N elements are of $\mathbf{f}$. We can express vector $\hat{\mathbf{f}} = \mathbf{Pf}$, with $\mathbf{f}$ a $N \times M$ matrix which projects the vector $\mathbf{f}$ on the support of $\mathbf{g}$:

$$\mathbf{P} = \begin{bmatrix} 1 & 0 & \dots & 0 & 0 & \dots & 0 \\ 0 & 1 & \dots & 0 & 0 & \dots & 0 \\ & & \vdots & & & \vdots & 0 \\ 0 & 0 & \dots & 1 & 0 & \dots & 0 \end{bmatrix} . \qquad (31)$$

The original optimization problem is now:

$$\hat{\mathbf{f}} = \underset{\mathbf{f}}{\operatorname{argmin}} \parallel \mathbf{Pf} - \mathbf{g} \parallel^2 \qquad (32)$$

subject to the constraint $\parallel \mathbf{Pf} - \mathbf{g} \parallel^2 = \mathbf{0}$. Applying the technique of *Lagrange multipliers* this problem can be alternatively formulated as an optimization problem without constraints:

$$V(\mathbf{f}) = \lambda \parallel \mathbf{Hf} - \mathbf{g} \parallel^2 + \parallel \mathbf{Pf} - \mathbf{g} \parallel^2 \rightarrow \min, \qquad (33)$$

if λ is large enough. The solution of this problem can be easily obtained by computing the partial derivative of criterion V with respect to the unknown $\mathbf{f}$:

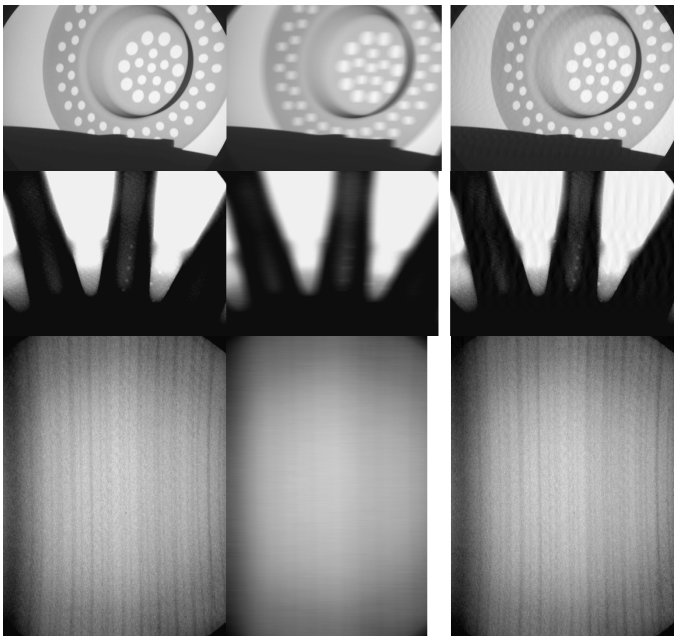

Fig. 38: Restoration in simulated degraded X-ray images. Each column shows the original, the degraded with n pixels and the restored images. The size of the images are respectively: 574×768, with $n = 30$; 574×768, with $n = 40$; and 768×574, with $n = 60$.

$$\frac{\partial}{\partial \mathbf{f}} V(\mathbf{f}) = 2\lambda \mathbf{H}^{\mathsf{T}}(\mathbf{H}\mathbf{f} - \mathbf{g}) + 2\mathbf{P}^{\mathsf{T}}(\mathbf{P}\mathbf{f} - \mathbf{g}) = \mathbf{0}, \tag{34}$$

then is

$$\hat{\mathbf{f}} = \left[\lambda \mathbf{H}^{\mathsf{T}}\mathbf{H} + \mathbf{P}^{\mathsf{T}}\mathbf{P}\right]^{-1} \left[\lambda \mathbf{H} + \mathbf{P}\right] \mathbf{g}. \tag{35}$$

This solution for the example of Fig. 37b is almost identical to the original sharp input signal of Fig. 37a. Fig. 38 shows three different restoration examples.

In the following example we simulate an X-ray image that has been degraded by a horizontal motion. The image is restored using MINIO algorithm (35). The output of this example is shown in the last row of Fig. 38. Details of the baggage are not discernible in the degraded image, but are recovered in the restored image.

The restoration quality is equally as good as the classical methods (see for example [18]), while the computation load is decreased considerably (see comparisons in [51]).

4 X-ray Image Representation

As we learned in the previous Section, in image processing for X-ray testing, segmentation is used to detect (potential) regions that can be the objects of interest that we are looking for (see Section 3.4). As segmented potential regions frequently set off false detections, an analysis of the segmented regions can significantly improve the effectiveness of detection. Measuring certain characteristics of the segmented regions (*feature extraction*) can help us to distinguish the false detection, although some of the features extracted are either irrelevant or are not correlated. Therefore, a *feature selection* must be performed. Depending on the values returned for the selected features, we can try to classify each segmented potential region in one of the following two classes: *background* or *object of interest*.

In this Section, we will explain several features that are normally used in image analysis and computer vision for X-ray testing. In our description, features will be divided into two groups: *geometric* and *intensity* features. Furthermore, we will cover some local descriptors and sparse representations that can be used in many X-ray testing applications. In this Chapter we shall concentrate on the extraction and selection of features, whereas in the following Chapter we will discuss the classification problem itself.

We will use Fig. 39 as our example in the description of features. In our example, we use an X-ray image of a circular defect. The segmentation is a binary image that gives information about the pixels that belongs to our object of interest (the defect). Geometric features are extracted from this binary image. Moreover, intensity features are extracted from the intensity image considering the pixels of the segmentation. Some intensity features consider only the gray values inside the segmented region, other ones take into account both gray values inside and outside the region (*e.g.*, contrast).

4.1 Geometric features

These provide information on the location, size and shape of the segmented region. Location and size features, such as center of mass, perimeter, height and width, are given in pixels. Shape features are usually coefficients without units. It is worth mentioning that we distinguish three different zones in the segmented image (see Fig. 39b), the segmented

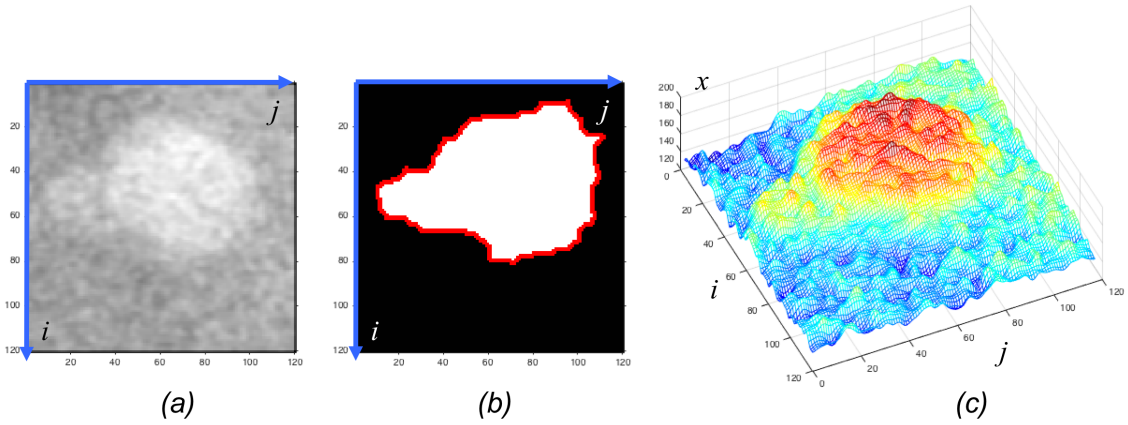

Fig. 39: Example of a region: a) X-ray image, b) segmented region (white pixels), c) 3D representation of the gray values.

region (white zone $\Re$), the boundary (red edge pixels ℓ) and the background (black zone).

4.1.1 Basic geometric features

In this Section we will summarize basic geometric features that can be easily extracted.

Height and width

The eight and width of a region can be defined as:

$$h = i_{\max} - i_{\min} + 1$$
$$w = j_{\max} - j_{\min} + 1 \qquad (36)$$

where $i_{\max}$ and $i_{\min}$ is the maximal and minimal value that takes coordinate i in the region. The same is valid for $j_{\max}$ and $j_{\min}$ in j-direction.

Area and perimeter

We define the area A of a region as the number of pixels that belong to the region. On the other hand, the perimeter L is the number of pixels that belong to the boundary. In the region of Fig. 39, the area and the perimeter are the number of the white and the number of the red pixels respectively. More accurate measurements for area and perimeter can also be estimated [10]: for instance, the boundary of the region can be fitted to a curve with known area and length, however, the computational time of such approaches can be extremely long if there are thousands of regions to be measured. Moreover, the shape of the region can be much more complex than a simple circle as shown in Fig. 33. We should remember, therefore, that the goal of feature extraction is not the accurate measurement, rather it is simply the extraction of features that can be used in a classification approach to separate our classes (objects of interest from background).

Center of mass

This provides information about the location of the region. It is computed as the average of coordinate i and coordinate j in pixels that belong to region $\Re$:

$$\bar{i} = \frac{1}{A}\sum_{i\in\Re} i \qquad \bar{j} = \frac{1}{A}\sum_{j\in\Re} j \qquad (37)$$

where A is the area of the region, *i.e.*, the number of pixels of the region.

Roundness

Shape features are usually attributed coefficients without units. An example is roundness that is defined as:

$$R = \frac{4 \cdot A \cdot \pi}{L^2} \qquad (38)$$

The roundness R is a value between 1 and 0. $R = 1$ means a circle, and $R = 0$ corresponds to a region without an area.

Other basic features

There are some useful features that can be extracted as follows [43]:

- *Euler Number*: The number of objects in the region minus the number of holes in those objects.
- *Equivalent Diameter*: The diameter of a circle with the same area as the region.

- *Major Axis Length* and *Minor Axis Length*: The length (in pixels) of the major and minor axis of the ellipse that has the same normalized second central moments as the region.
- *Orientation*: The angle (in degrees ranging from -90 to 90 degrees) between the x-axis and the major axis of the ellipse that has the same second-moments as the region.
- *Solidity*: The proportion of the pixels in the convex hull that are also in the region.
- *Extent*: The ratio of pixels in the region to pixels in the total bounding box.
- *Eccentricity*: The eccentricity of the ellipse that has the same second-moments as the region.

4.1.2 Elliptical features

Elliptical features can be used to extract information about location, size and shape of a region. They are extracted from a fitted ellipse to the boundary of the region [16]. From this ellipse we can extract the center, the length of the axes, the orientation and the eccentricity.

The pixels of the boundary are defined as (x_i, y_i) for $i = 1 \dots L$. It is well known that an ellipse is defined as:

$$ax^2 + bxy + cy^2 + dx + ey + f = 0, \qquad (39)$$

that can be written as $\mathbf{a}^\mathsf{T}\mathbf{x} = 0$, where $\mathbf{a} = [a\ b\ c\ d\ e\ f]^\mathsf{T}$ is a vector that includes the parameters of the ellipse and $\mathbf{x} = [x^2\ xy\ y^2\ x\ y\ 1]^\mathsf{T}$ is a vector that includes the coordinates of a point (x, y) that lies on the ellipse.

If our region is elliptical, then for each point (x_i, y_i) we have $\mathbf{a}^\mathsf{T}\mathbf{x}_i = 0$ with $\mathbf{x}_i = [x_i^2\ x_iy_i\ y_i^2\ x_i\ y_i\ 1]^\mathsf{T}$. Nevertheless, in practice the regions are not perfectly elliptical, not only because real regions have different shapes, but also there is a discretization error when forming a digital image. For this reason, we look for a vector $\mathbf{a}$ so that $\mathbf{a}^\mathsf{T}\mathbf{x}_i \to \min$ for every point $i = 1 \dots L$. That is, we can formulate the estimation of the parameters of the ellipse as an optimization problem as follows:

$$\| \mathbf{Xa} \| \to \min \qquad (40)$$

where $\mathbf{X}$ is matrix with L rows whose i-th row is $\mathbf{x}_i^\mathsf{T}$. Usually, a solution can be found by minimizing (40) subject to $\| \mathbf{a} \| = 1$. In this case, $\mathbf{a}$ is the last column of matrix $\mathbf{V}$, where $\mathbf{X} = \mathbf{USV}^\mathsf{T}$ is the singular-value-decomposition (SVD) of $\mathbf{X}$ [24].

The elliptical features can be extracted by writing (39) as follows:

$$\left(\frac{x - x_0}{a_e}\right)^2 + \left(\frac{y - y_0}{b_e}\right)^2 = 1 \qquad (41)$$

where

$$a_e = \frac{1}{\sqrt{s\ a_p}}\ ,\ b_e = \frac{1}{\sqrt{s\ b_p}} \qquad (42)$$

with

$$s = \frac{1}{v - f} \qquad v = \mathbf{t}^\mathsf{T}\mathbf{T}\mathbf{t}$$

$$\mathbf{T} = \begin{bmatrix} a & b/2 \\ b/2 & c \end{bmatrix}$$

$$\mathbf{t} = \begin{bmatrix} x_0 \\ y_0 \end{bmatrix} = \frac{1}{2}\mathbf{T}^{-1}\begin{bmatrix} d \\ e \end{bmatrix}$$

$$\begin{aligned} a_p &= a\cos^2(\alpha) + b\cos(\alpha)\sin(\alpha) + c\sin^2(\alpha) \\ b_p &= a\sin^2(\alpha) - b\cos(\alpha)\sin(\alpha) + c\cos^2(\alpha) \end{aligned}$$

and

$$\alpha = \frac{1}{2}\arctan\left(\frac{b}{a - c}\right) \qquad (43)$$

The axes of the ellipse are defined by a_e and b_e, the center of the ellipse is located on (x_0, y_0) and the orientation is α. Thus, the eccentricity is defined by

$$e_x = \frac{\min(a_e, b_e)}{\max(a_e, b_e)} \qquad (44)$$

For circular shapes, the eccentricity as the roundness (38), takes values between 0 and 1, where 1 means a perfect circle.

4.1.3 Fourier descriptors

Shape information –invariant to scale, orientation and position– can be measured using *Fourier descriptors* [79], [9], [66]. The coordinates of the pixels of the boundary are arranged as a complex number $i_k + j \cdot j_k$, with $j = \sqrt{-1}$ and $k = 0, ..., L - 1$, where L is the perimeter of the region, and pixel k and $k + 1$ are connected. The complex boundary function can be considered as a periodical signal of period L. The Discrete Fourier Transformation [8] gives a characterization of the

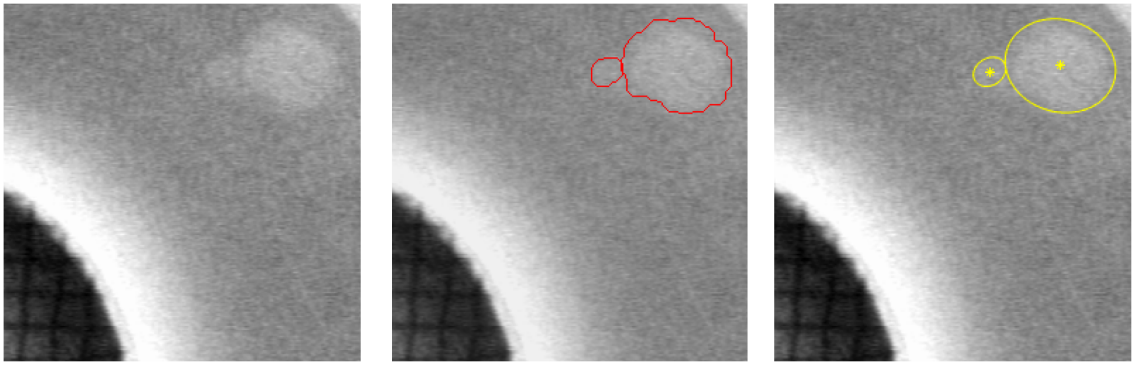

Fig. 40: Elliptical features of two defects (the original image, the boundaries of the segmentation and the estimated ellipse are shown). The size of the image is 256 $\times$ 256 pixels. In this example, the coordinates of the center of each ellipse correspond to the yellow cross. The estimated length of each axis are 9.6 and 12.1 pixels for the small ellipse and 32.7 and 39.7 for the large ellipse. The orientation (with respect to vertical axis in a counterclockwise direction) is 59.6^0 and -74.6^0 respectively.

shape of the region. The Fourier coefficients are defined by:

$$F_n = \sum_{k=0}^{L-1}(i_k + j \cdot j_k)e^{-j\frac{2\pi kn}{L}} \quad \text{for } n = 0, ..., L-1. \tag{45}$$

The Fourier descriptors correspond to the coefficients F_n for $n > 0$. The Fourier coefficient F_0 is not used because it gives information about the location of the region. The magnitude and phase of Fourier descriptors give information about orientation and symmetry of the region. In general, only the magnitude $|F_n|$ is used. Fourier descriptors are invariant under rotation.

4.1.4 Invariant moments

The statistical moments are defined by:

$$m_{rs} = \sum_{i,j\in\Re} i^r j^s \qquad \text{for } r, s \in \mathbb{N} \tag{46}$$

where $\Re$ is the set of pixels that belong to the region (see white pixels in Fig. 39b). The parameter $r + s$ corresponds to the order of the moment. The reader can demonstrate that the zero-th moments m_{00} is equal to the area A of the region. Moreover, the center of mass of the region is easily defined by:

$$\bar{i} = \frac{m_{10}}{m_{00}} \qquad \bar{j} = \frac{m_{01}}{m_{00}} \tag{47}$$

The center of mass and statistical moments of higher order, however, are not invariant to the location of the region. This can be useful for detecting objects that must be in certain locations. Nevertheless, when objects of interest may be everywhere in the image we must use features that are invariant to the position. Using the center of mass, the central moments are defined. They are invariant to the position:

$$\mu_{rs} = \sum_{i,j\in\Re} (i - \bar{i})^r (j - \bar{j})^s \qquad \text{for } r, s \in \mathbb{N}. \tag{48}$$

Other known moments that can be used are the well–known Hu-moments [28], [74]. These were developed using the central moments as follows:

$$\begin{aligned}
\phi_1 &= \eta_{20} + \eta_{02} \\
\phi_2 &= (\eta_{20} - \eta_{02})^2 + 4\eta_{11}^2 \\
\phi_3 &= (\eta_{30} - 3\eta_{12})^2 + (3\eta_{21} - \eta_{03})^2 \\
\phi_4 &= (\eta_{30} + \eta_{12})^2 + (\eta_{21} + \eta_{03})^2 \\
\phi_5 &= (\eta_{30} - 3\eta_{12})(\eta_{30} + \eta_{12})[(\eta_{30} + \eta_{12})^2 - 3(\eta_{21} + \eta_{03})^2] + \\
&\quad (3\eta_{21} - \eta_{03})(\eta_{21} + \eta_{03})[3(\eta_{30} + \eta_{12})^2 - (\eta_{21} + \eta_{03})^2] \\
\phi_6 &= (\eta_{20} - \eta_{02})[(\eta_{30} + \eta_{12})^2 - (\eta_{21} + \eta_{03})^2] + \\
&\quad 4\eta_{11}(\eta_{30} + \eta_{12})(\eta_{21} + \eta_{03}) \\
\phi_7 &= (3\eta_{21} - \eta_{03})(\eta_{30} + \eta_{12})[(\eta_{30} + \eta_{12})^2 - 3(\eta_{21} + \eta_{03})^2] - \\
&\quad (\eta_{30} - 3\eta_{12})(\eta_{21} + \eta_{03})[3(\eta_{30} + \eta_{12})^2 - (\eta_{21} + \eta_{03})^2]
\end{aligned} \tag{49}$$

with

$$\eta_{rs} = \frac{\mu_{rs}}{\mu_{00}^t} \qquad t = \frac{r+s}{2} + 1.$$

Hu–moments are invariant to translation, rotation and scale. That means that regions that have the same shape, but have a different size, location and orientation, will have similar Hu-moments.

In addition, there are similar invariant features, called Gupta moments, that are derived from the

pixels of the boundary (instead of the region) [19]. They are invariant to translation, rotation and scale.

Sometimes, it is necessary to have features that are invariant to affine transformation. For this reason Flusser moments, *i.e.*, features invariant to translation, rotation, scale and affine transformation were derived from second and third order central moments [72], [17]:

$$I_1 = \frac{\mu_{20}\mu_{02} - \mu_{11}^2}{\mu_{00}^4}$$

$$I_2 = \frac{\mu_{30}^2\mu_{03}^2 - 6\mu_{30}\mu_{21}\mu_{12}\mu_{03} + 4\mu_{30}\mu_{12}^3 + 4\mu_{21}^3\mu_{03} - 3\mu_{21}^2\mu_{12}^2}{\mu_{00}^{10}}$$

$$I_3 = \frac{\mu_{20}(\mu_{21}\mu_{03} - \mu_{12}^2) - \mu_{11}(\mu_{30}\mu_{03} - \mu_{21}\mu_{12}) + \mu_{02}(\mu_{30}\mu_{12} - \mu_{21}^2)}{\mu_{00}^7} \tag{50}$$

$$\begin{aligned} I_4 = \; & (\mu_{20}^3\mu_{03}^2 - 6\mu_{20}^2\mu_{11}\mu_{12}\mu_{03} - 6\mu_{20}^2\mu_{02}\mu_{21}\mu_{03} + 9\mu_{20}^2\mu_{02}\mu_{12}^2 \\ & +12\mu_{20}\mu_{11}^2\mu_{21}\mu_{03} + 6\mu_{20}\mu_{11}\mu_{02}\mu_{30}\mu_{03} - 18\mu_{20}\mu_{11}\mu_{02}\mu_{21}\mu_{12} \\ & -8\mu_{11}^3\mu_{30}\mu_{03} - 6\mu_{20}\mu_{02}^2\mu_{30}\mu_{12} + 9\mu_{20}\mu_{02}^2\mu_{21} \\ & +12\mu_{11}^2\mu_{02}\mu_{30}\mu_{12} - 6\mu_{11}\mu_{02}^2\mu_{30}\mu_{21} + \mu_{02}^3\mu_{30}^2)/\mu_{00}^{11} \end{aligned}$$

4.2 Intensity features

These provide information about the intensity of a region. For gray value images, *e.g.*, , X-ray images, there is only one intensity channel. The following features are computed using the gray values in the image, where $x(i,j)$ denotes the gray value of pixel (i,j).

4.2.1 Basic intensity features

In this Section we summarize basic intensity features that can be easily extracted.

Mean gray value
The mean gray value of the region is computed as:

$$G = \frac{1}{A}\sum_{i,j\in\Re} x(i,j) \tag{51}$$

where $\Re$ is the set of pixels of the region and A the area. A 3D representation of the gray values of the region and its neighborhood of our example is shown in Fig. 39. In this example $G = 142.70$ ($G = 0$ means 100% black and $G = 255$ corresponds to 100% white).

Mean gradient in the boundary
This feature gives information about the change of the gray values in the boundary of the region. It is computed as:

$$C = \frac{1}{L}\sum_{i,j\in\ell} x'(i,j) \tag{52}$$

where $x'(i,j)$ means the gradient of the gray value function in pixel (i,j) (see Section 3.3.1) and ℓ the set of pixels that belong to the boundary of the region. The number of pixels of this set corresponds to L , the perimeter of the region.

Mean second derivative
This feature is computed as:

$$D = \frac{1}{A}\sum_{i,j\in\Re} x''(i,j) \tag{53}$$

where $x''(i,j)$ denotes the second derivate of the gray value function in pixel (i,j). The Laplacian-of-Gauss (LoG) operator can be used to calculate the second derivate of the image. If $D > 0$ we have a region that is darker than its neighborhood as shown in Fig. 21.

4.2.2 Contrast

The contrast gives a measure of the difference in the gray value between region and its neighborhood. The smaller the gray value difference, the smaller the contrast. In this work, region and neighborhood

define a zone. The zone is considered as a window of the image:

$$g(i,j) = x(i + i_r, j + j_r) \tag{54}$$

for $i = 1, ..., 2h + 1$ and $j = 1, ..., 2w + 1$, where h and w are the height and width as expressed in (36). The offsets i_r and j_r are defined as $i_r = \bar{i} - h - 1$ y $j_r = \bar{j} - b - 1$, where $(\bar{i}, \bar{j})$ denotes the center of mass of the region as computed in (47).

Contrast is a very important feature in fault detection, as the differences in the gray values are good for distinguishing a region from its neighborhood. The smaller the gray value difference, the smaller the contrast. In order to visualize the contrast we can use a 3D representation with three coordinates (x, y, z), where (x, y) are used to represent the location of a pixel (i, j), and z is used for the representation of the gray value. An example is illustrated in Fig. 39c that shows the 3D representation of Fig. 39a. The reader can observe in this example a high contrast region.

There are many definitions of contrast. A common definition of contrast is given using texture features (see for example Section 4.3.1). Other simple definitions of contrast are given in [30], [72]:

$$\begin{aligned} K_1 &= \frac{G - G_e}{G_e}, \\ K_2 &= \frac{G - G_e}{G + G_e}, \\ K_3 &= \ln(G/G_e). \end{aligned} \tag{55}$$

where G an G_e denote the mean gray value in the region and in the neighborhood respectively.

Two further definitions of contrast are given in [53] where new contrast features are suggested. According to Fig. 41 these new features can be calculated in four steps: *i)* we take a profile in i direction and in j direction centered in the mass center of the region (see P_1 and P_2 respectively); *ii)* we calculate the ramps R_1 and R_2 that are estimated as a first order function that contains the first and last point of P_1 and P_2; *iii)* new profiles without background are computed as $Q_1 = P_1 - R_1$ and $Q_2 = P_2 - R_2$ (they are stored together as $Q = [Q_1\ Q_2]$); *iv)* the new contrast features are given by:

$$K_\sigma = \sigma_Q \quad \text{and} \quad K = \ln(Q_{max} - Q_{min}). \tag{56}$$

Another definition of contrast can be found in [32], where the contrast is given by the mean of absolute differences between pixel values and mean of adjacent (*e.g.*, 8-adjacent pixels):

$$K_c = \frac{1}{A_T} \sum_{(i,j)\in\mathbb{T}} |g(i,j) - \mu_{A(i,j)}|. \tag{57}$$

where A_T is the area of the region and its neighborhood and $\mu_{A(i,j)}$ is the mean value of pixels locations adjacent of pixel (i, j).

Other basic features

A simple texture feature is the local variance [29]. This is given by:

$$\sigma_g^2 = \frac{1}{4hb + 2h + 2b} \sum_{i=1}^{2h+1} \sum_{j=1}^{2b+1} (g(i,j) - \bar{g})^2 \tag{58}$$

where $\bar{g}$ denotes the mean gray value in the zone.

Other basic intensity features such as kurtosis and skewness can be computed as (58).

4.2.3 *Crossing line profiles*

An approach based on *crossing line profiles* (CLP) was originally developed to detect aluminum casting defects [45], however, it can be used to detect spots in general, or regions that have some gray value difference with their neighborhood. As the contrast between a defect and a defect-free neighborhood is distinctive, the detection is usually performed by thresholding this feature (as we already learned in Section 4.2.2). Nevertheless, this measurement suffers from accuracy error when the neighborhood is not homogeneous, for example when a defect is at an edge of a regular structure of the test object (see Fig. 31). For this reason, many approaches use a-priori information about the location of regular structures of the test piece. CLP is able to detect those defects without a-priori knowledge using *crossing line profiles*, *i.e.*, the gray level profiles along straight lines crossing each segmented potential region in the middle. The profile that contains the most similar gray levels in the extremes is selected. Hence, the homogeneity of the neighborhood is ensured. Features from the selected profile are extracted.

In this approach, we follow a simple automated segmentation approach based on Fig. 33 and Fig. 31. The steps of detection based on CLP are shown

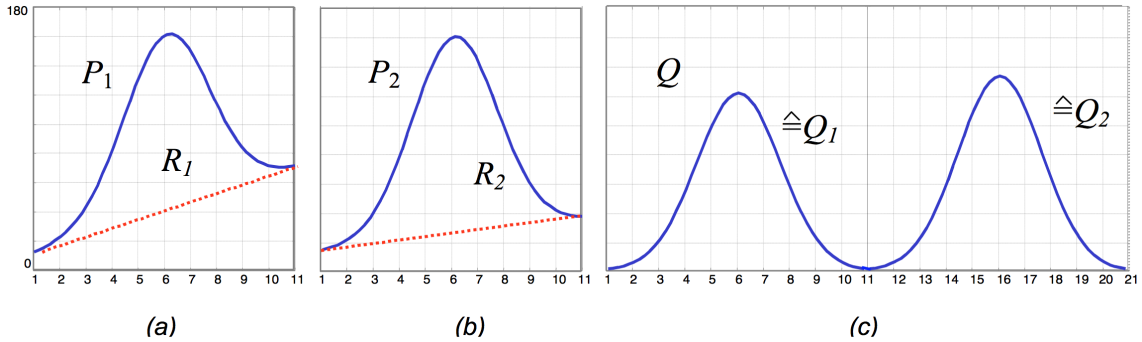

Fig. 41: Computation of Q for contrast features for region of Fig. 39: a) Profile in i direction, b) profile in j direction, c) fusion of profiles: $Q = [Q_1\ Q_2]$.

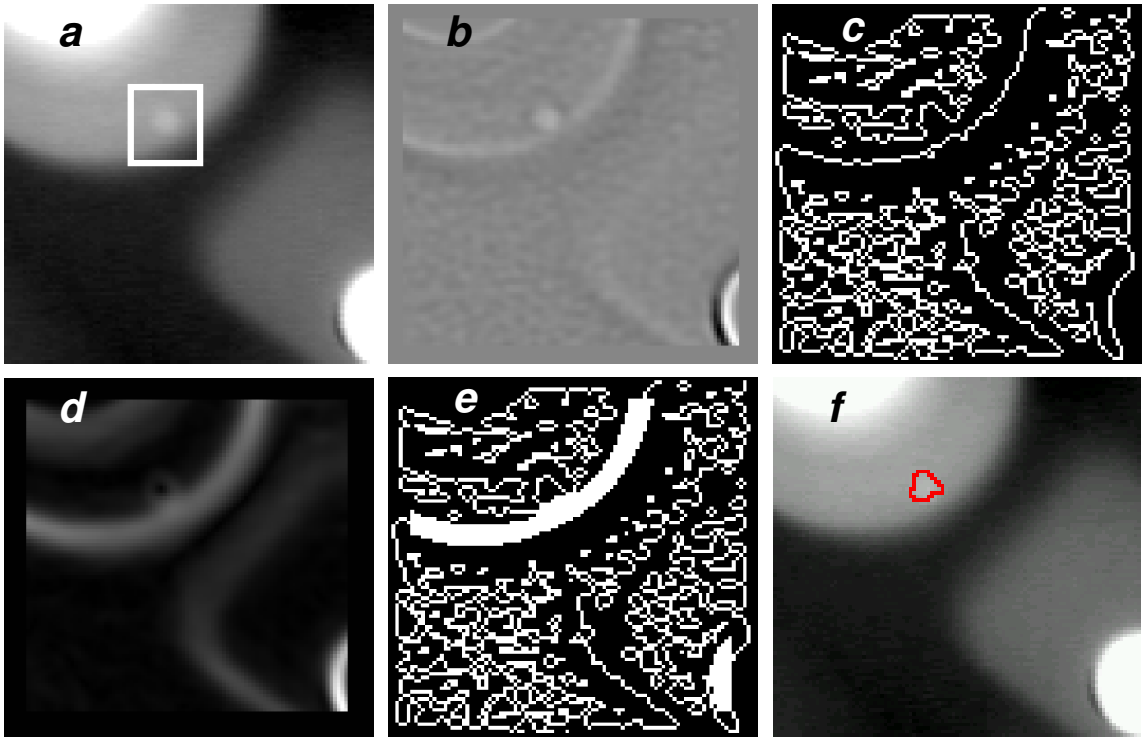

Fig. 42: Detection of flaws: a) radioscopic image with a small flaw at an edge of a regular structure, b) Laplacian-filtered image with σ = 1.25 pixels (kernel size = 11×11), c) zero crossing image, d) gradient image, e) edge detection after adding high gradient pixels, and f) detected flaw using feature F_1 extracted from a crossing line profile.

in Fig. 42. First, a Laplacian-of-Gaussian (LoG) kernel and a zero crossing algorithm are used to detect the edges of the X-ray images. The LoG-operator involves a Gaussian low–pass filter which is a good choice for the pre-smoothing of our noisy images that are obtained without frame averaging. The resulting binary edge image should produce at real defects closed and connected contours which demarcate *regions*. However, a region of interest may not be perfectly enclosed if it is located at an edge of a regular structure as shown in Fig. 42c. In order to complete the remaining edges of these defects, a thickening of the edges of the regular structure is performed as follows: a) the gradient of the original image is calculated (see Fig. 42d); b) by thresholding the gradient image at a high gray level a new binary image is obtained; and c) the resulting image is added to the zero-crossing image (see Fig. 42e). Afterwards, each closed region is segmented as a potential flaw. For details see a description of

the method in [52].

This is a very simple detector of potential regions with a large number of false detections flagged erroneously. However, the advantages are as follows: *i)* it is a single detector (it is the same detector for each image), *ii)* it is able to identify potential defects independent of the placement and the structure of the specimen, *i.e.*, without a-priori information of the design structure of the test piece, and *iii)* the detection rate of real flaws is very high (approximately 90%). In order to reduce the number of the false positives, the segmented regions must be measured and classified.

A segmented potential region is defined as a region enclosed by edges of the binary image obtained in the edge detection (see connected black pixels in Fig. 42e). For each segmented region, a window g is defined from the X-ray image x as: $g(i,j) = x(i+i_r, j+j_r)$ for $i = 1 \dots 2h+1$, and $j = 1 \dots 2w+1$, where h and w are the height and width of the region as defined in (36). The offsets i_r and j_r are defined as $i_r = \bar{i} - h - 1$ and $j_r = \bar{j} - w - 1$ where $(\bar{i}, \bar{j})$ denotes the coordinates of the center of mass of the region (47), rounded to the nearest integers. Hence, g is a window of size $(2h+1) \times (2w+1)$, in which the middle pixel corresponds to the center of mass of the segmented potential flaw, *i.e.*, $g(h+1, w+1) = x(\bar{i}, \bar{j})$.

Now, we define the crossing line profile P_θ as the gray level function along a straight line of window g through the middle pixel $(h+1, w+1)$ forming an angle θ with i-axis. In Section 4.2.2, P_0 and $P_{\pi/2}$ were analyzed together in order to obtain two features, K and K_σ, that give a measurement of the difference between maximum and minimum, and the standard deviation of both crossing line profiles. However, the analysis does not take into account that the profiles could include a non-homogeneous area. For example, if a non-defect region is segmented at an edge of a regular structure, it could be that P_0 (or $P_{\pi/2}$) includes a significant gray level change of the regular structure. In this case, the variation of the profile will be large and therefore the region will be erroneously classified as defect.

In order to avoid this problem, we suggest an individual analysis of eight crossing line profiles P_θ, at $\theta = k\pi/8$, for $k = 0, ..., 7$, as illustrated in Fig. 43. In this analysis, the crossing line profile that contains the most similar gray levels in the extremes is selected. Hence, the attempt is made to ensure the homogeneity of the neighborhood filtering out those profiles that present a high gray level change in the edge of the regular structure. In the example of Fig. 43, the selected profile is obtained for $k = 5$ where the gray values of the extremes are both approximately equal to 150. We can observe that the selected crossing line is approximately perpendicular to the direction of the gradient of the X-ray image without defect. This coincides with one of the criteria used by approaches with a-priori knowledge: the selected pixels of the defect-free area are located perpendicular to the direction of the gradient of the piece's contour [54].

Before the features are extracted, a pre-processing of the selected crossing line profile is performed as follows: 1) The selected profile is resized to size $n = 32$ using a nearest neighbor interpolation. The resized profile will be denoted by P. 2) In order to obtain a defect profile without the background of the regular structure, P is linearly transformed by $Q_i = mP_i + b$, for $i = 1, ..., n$, where m and b are so chosen that $Q_1 = Q_n = 0$.

Finally, the proposed features are extracted from the normalized profile Q. They are defined as follows:

$$\begin{aligned} \bar{Q} &= \text{mean}(Q) \\ \sigma_Q &= \text{std}(Q) \\ \Delta_Q &= \max(Q) - \min(Q) \\ F_i &= \textstyle\sum_{k=0}^{n-1} Q_{k+1} e^{-j\frac{2\pi ki}{n}} \quad \text{for } i = 1, ..., 4. \end{aligned} \tag{59}$$

That is $\bar{Q}$: mean of Q; σ_Q: standard deviation of Q; Δ_Q: difference between maximum and minimum of Q; and F_i: magnitude of the i-th harmonic of the Discrete Fourier Transform of Q for $i = 1, ..4$.

In the following example, we show how to detect a very small casting defect that is located at the edge of a regular structure as illustrated in Fig. 42 using area and CLP features. We follow the general block-diagram of Fig. 33. That is area and contrast features are extracted for each region defined by enclosed edges. The detection is performed if the size of the region is between some thresholds and a CLP feature is high enough. The output of this example is shown in Fig. 42 and 43.

CLP features were tested on detecting casting defects. In this experiment, 50 X-ray images of aluminum wheels were analyzed. In the segmentation, approximately 23,000 potential flaws were obtained, in which there were 60 real defects. Some

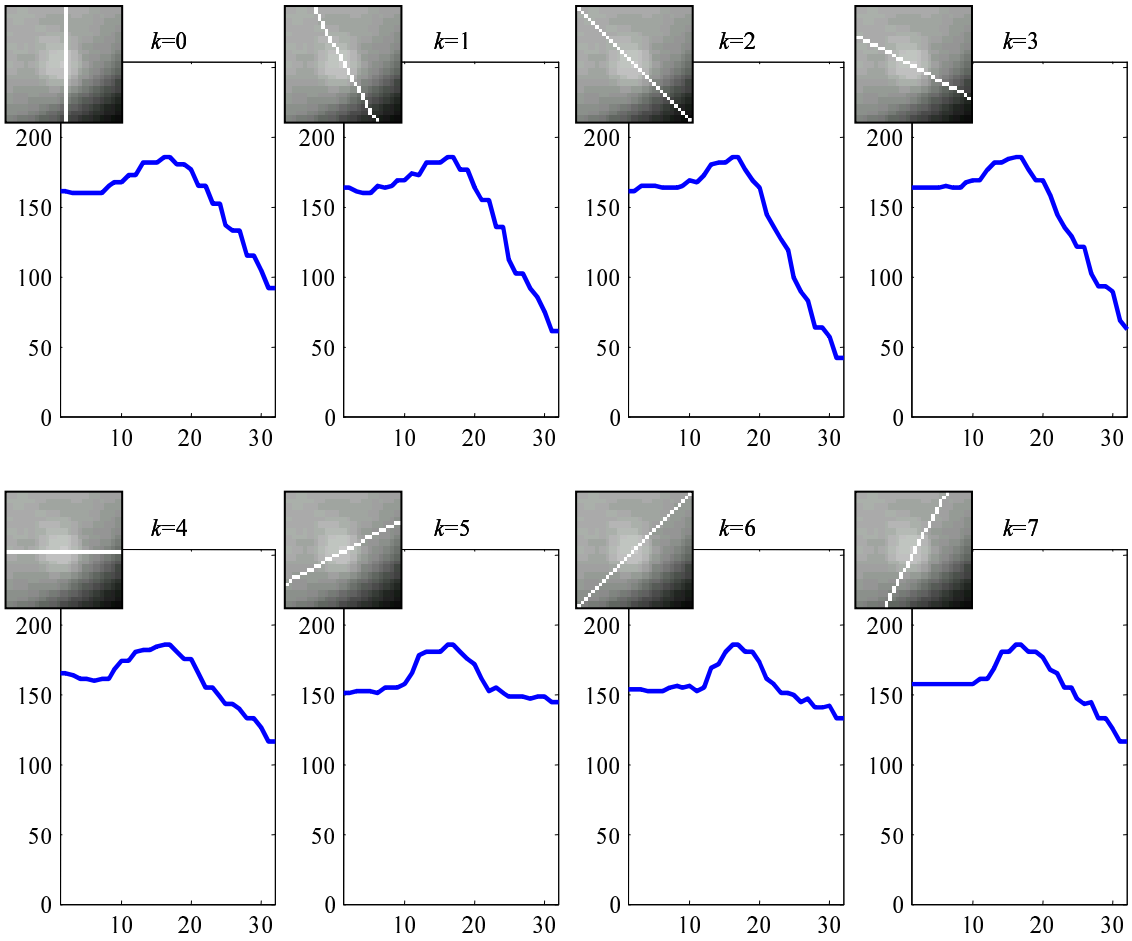

Fig. 43: Crossing line profiles for the window shown in Fig. 42a.

of these were existing blow holes. The other defects were produced by drilling small holes in positions of the casting which were known to be difficult to detect. In the performance analysis, the best result was achieved by our feature F_1 (59). The class distribution between class 'defect' and 'non-defect' (or regular structure) is illustrated in Fig. 44. The reader can observe the effectiveness of the separation clearly. For more details see [45].

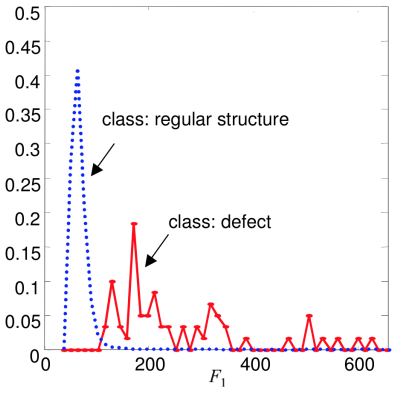

Fig. 44: Class distribution of CLP feature F_1 in detection of casting defects.

4.3 Texture representation

These features provide information about the distribution of the gray values in the image. In this work however, we restrict the computation of the texture features for a zone only defined as region and neighborhood (see equation 54).

4.3.1 Statistical textures - Haralick

Statistical texture features can be computed using the co-occurrence matrix $\mathbf{P}_{kl}$ [21]. The element $P_{kl}(i,j)$ of this matrix for a zone is the number of times, divided by N_T, that gray-levels i and j occur in two pixels separated by that distance and direction given by the vector (k,l), where N_T is the number of pixels pairs contributing to build matrix

$\mathbf{P}_{kl}$. In order to decrease the size $N_x \times N_x$ of the co-occurrence matrix the gray scale is often reduced to 8 gray levels. From the co-occurrence matrix several texture features can be computed. Haralick in [21] proposes (here $p(i,j) := P_{kl}(i,j)$):

$$
\begin{array}{lrcl}
\text{Angular second moment:} & f_1 & = & \sum_{i=1}^{N_x}\sum_{j=1}^{N_x}[p(i,j)]^2 \\
\text{Contrast:} & f_2 & = & \sum_{n=0}^{N_x-1} n^2 \sum_{i=1}^{N_x}\sum_{j=1}^{N_x} p(i,j) \text{for } |i-j|=n \\
\text{Correlation:} & f_3 & = & \frac{1}{\sigma_x\sigma_y}\sum_{i=1}^{N_x}\sum_{j=1}^{N_x}\left[ij\cdot p(i,j)-\mu_x\mu_y\right]^2 \\
\text{Sum of squares:} & f_4 & = & \sum_{i=1}^{N_x}\sum_{j=1}^{N_x}(i-j)^2 p(i,j) \\
\text{Inverse difference moment:} & f_5 & = & \sum_{i=1}^{N_x}\sum_{j=1}^{N_x}\frac{p(i,j)}{1+(i-j)^2} \\
\text{Sum average:} & f_6 & = & \sum_{i=2}^{2N_x} i\cdot p_{x+y}(i) \\
\text{Sum variance:} & f_7 & = & \sum_{i=2}^{2N_x}(i-f_8)\cdot p_{x+y}(i) \\
\text{Sum entropy:} & f_8 & = & -\sum_{i=2}^{2N_x} p_{x+y}(i)\cdot\log(p_{x+y}(i)) \\
\text{Entropy:} & f_9 & = & -\sum_{i=1}^{N_x}\sum_{j=1}^{N_x} p(i,j)\log(p(i,j)) \\
\text{Difference variance:} & f_{10} & = & \text{var}(\mathbf{p}_{x+y}) \\
\text{Difference entropy:} & f_{11} & = & -\sum_{i=0}^{N_x-1} p_{x-y}(i)\cdot\log(p_{x-y}(i)) \\
\text{Information measures of correlation 1:} & f_{12} & = & \frac{f_9-HXY1}{\max(HX,HY)} \\
\text{Information measures of correlation 2:} & f_{13} & = & \sqrt{1-\exp\left(-2(HXY2-HXY)\right)} \\
\text{Maximal correlation coefficient:} & f_{14} & = & \sqrt{\lambda_2}
\end{array}
\tag{60}
$$

where μ_x, μ_y, σ_x and σ_y are the means and standard deviations of p_x and p_y respectively with

$$
\begin{array}{rcl}
p_x & = & \sum_{j=1}^{N_x} p(i,j) \\
p_y & = & \sum_{i=1}^{N_x} p(i,j) \\
p_{x+y}(k) & = & \sum_{i=1}^{N_x}\sum_{j=1 i+j=k}^{N_x} p(i,j) \quad \text{for } k=2,3,\ldots 2N_x \\
p_{x-y}(k) & = & \sum_{i=1}^{N_x}\sum_{j=1 |i-j|=k}^{N_x} p(i,j) \quad \text{for } k=0,1,\ldots N_x-1
\end{array}
$$

and

$$
\begin{array}{rcl}
HX & = & -\sum_{i=1}^{N_x} p_x(i)\log\left(p_x(i)\right) \\
HY & = & -\sum_{j=1}^{N_x} p_y(j)\log\left(p_y(j)\right) \\
HXY1 & = & -\sum_{i=1}^{N_x}\sum_{j=1}^{N_x} p(i,j)\log\left(p_x(i)p_y(j)\right) \\
HXY2 & = & -\sum_{i=1}^{N_x}\sum_{j=1}^{N_x} p_x(i)p_y(j)\log\left(p_x(i)p_y(j)\right)
\end{array}
\quad .
$$

In f_14, λ_2 is the second largest eigenvalue of Q defined by

$$
Q(i,j) \;=\; \sum_{k=1}^{N_x}\frac{p(i,k)p(j,k)}{p_x(i)p_y(k)}
$$

The texture features are extracted for four directions (0^0-180^0, 45^0-225^0, 90^0-270^0 and 135^0-315^0) in different distances $d = \max(k,l)$. That is, for a given distance d we have four possible co-occurrence matrices: P_{0d}, P_{dd},P_{d0} andP_{-dd}. For example, for $d = 1$, we have (k,l) = (0,1); (1,1); (1,0); and (-1,1). After Haralick, fourteen texture features using each co-occurrence matrix are computed (60), and the mean and range for each feature are calculated, *i.e.*, we obtain 14 × 2 = 28 texture features for each distance d. The features will be denoted as $\bar{f}_i$ for the mean and f_i^{Δ} for the range, for $i = 1 \ldots 14$.

4.3.2 Local binary patterns

LBP, *Local Binary Patterns* was proposed as a texture feature [64]. The idea is to extract texture information from occurrence histogram of local binary patterns computed from the relationship between each pixel intensity value with its eight neighbors. The LBP features are the frequencies of each one of the histogram bins. LBP is computed in three steps: *i)* coding, *ii)* mapping and *iii)* histogram.

Coding

Each pixel (i,j) of the input image has a set of neighbors. Typically, the set of eight neighbors defined by the 8-connected pixels is used. However, more neighbors for different distances can be defined as well. For 8 connected pixels, the locations

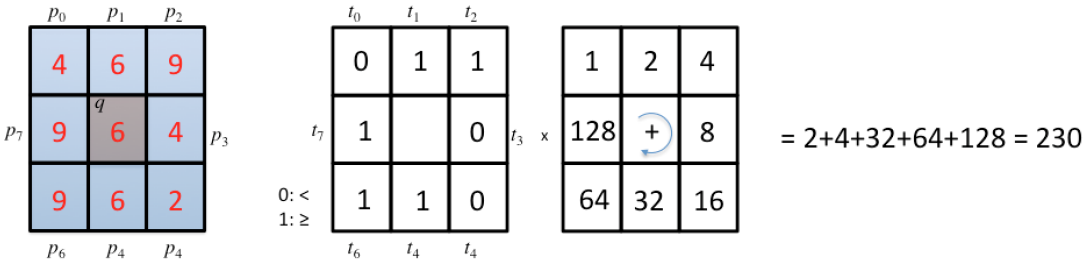

Fig. 45: LBP coding: a central pixel q the gray value of which is 6 has 8 neighbors with gray values $p_0 = 4$, $p_1 = 6$, $p_2 = 9$, $p_3 = 4$, $p_4 = 2$, $p_5 = 6$, $p_6 = 9$, and $p_7 = 9$. A new mask with 8 bits is built where $t_i = 1$ if $p_i \geq q$ otherwise $t_i = 0$. The LBP code is computed as $\sum_i t_i 2^i$, in this example the code is 230.

are $(i-1, j-1)$; $(i-1, j)$; $(i-1, j+1)$; $(i, j+1)$; $(i+1, j+1)$; $(i+1, j+1)$; $(i+1, j)$ and $(i+1, j-1)$ respectively as shown in Fig. 45. The central pixel has a gray value q, and the neighbors have gray values p_i, for $i = 0 \ldots 7$. The code is computed by:

$$y = \sum_{i=0}^{7} t_i 2^i, \tag{61}$$

where $t_i = 1$ if $p_i \geq q$ otherwise $t_i = 0$. That means, a pixel q with its neighbors can be coded as a number $y \in \{0 \ldots 255\}$. The code can be represented as a string of bits as shown in Fig. 45.

Mapping

We can observe that the code generated by the previous step can be categorized according to number of changes (from '1' to '0', or from '0' to '1') in a cycle. For instance, in the example of Fig. 45, where the code is 01100111, we define a cycle with eight transitions as: $0 \rightarrow 1 \rightarrow 1 \rightarrow 0 \rightarrow 0 \rightarrow 1 \rightarrow 1 \rightarrow 1 \rightarrow 0$ (the last bit is the repetition of the first one because it is a cycle). The number of changes is $U = 4$. Thus, we can have codes with $U = 0, 2, 4, 6$ and 8 as illustrated in Fig. 46. After the authors, there are *uniform* and *non-uniform* patterns. The first ones ($U = 0$ and 2) correspond to textures with a low number of changes, the last ones ($U > 2$) can be interpreted as noise because there are many changes in the gray values. There are 58 uniform patterns and 198 non-uniform patterns. Each uniform code is mapped as a number from 0 to 57 as illustrated in Fig. 46, whereas all non-uniform codes are mapped as number 58. This descriptor is known as LBP-u2. LBP-u2 mapping correspond to a mapping that varies with the orientation of the image, *i.e.*, it is not rotation-invariant. In order to build a rotation-invariant LBP descriptor, all patterns that have the same structure but with different rotations are mapped as an unique number. For instance, all patterns of the second row of Fig. 46 are mapped with the same number. The same is valid for the third row. In this mapping, we have 36 different numbers. This descriptor is known as LBP-ri.

Histogram

The process of coding and mapping is performed at each pixel of the input image. Thus, each pixel is converted into a number from 0 to $M-1$, with a mapping of M numbers. Afterwards, a histogram of M bins of this image is computed. The LBP descriptor of the image is this histogram.

LBP is very robust in terms of gray-scale and rotation variations [64]. An example is shown in Fig. 47. Other LBP features like *semantic* LBP (sLBP) [61] can be used in order to bring together similar bins.

The reader can find a descriptor with similar properties in [65], where LPQ (from *local phase quantization*) is proposed.

4.3.3 Binarized statistical image features

This is a texture description based on a binary code that is computed via linear projection using independent component analysis of natural images [31]. We tested all filters proposed by the authors, the best performance was achieved by a filter of 7×7 pixels, with a binarization of 11 bits and a normalized histogram. The size of the descriptor is 2.048. In our experiments, we call these features BSIF.

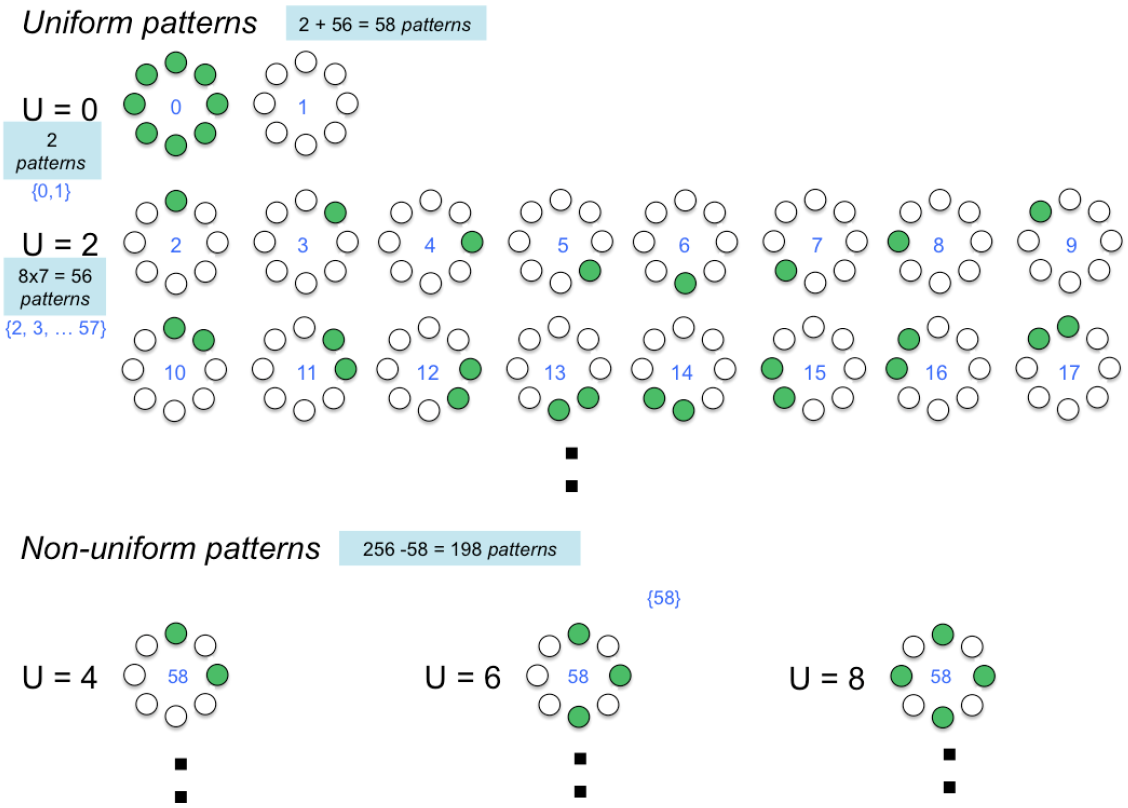

Fig. 46: LBP mapping for 8 neighbors. Each small circle represents a bit t_i of the code, green means '1', white means '0'. U is the number of changes from '1' to '0', or from '0' to '1' in one cycle. For a small number of changes, *i.e.*, $U = 0$ and 2, the codes represent *uniform patterns*, for $U > 2$ the patterns are *non-uniform*.

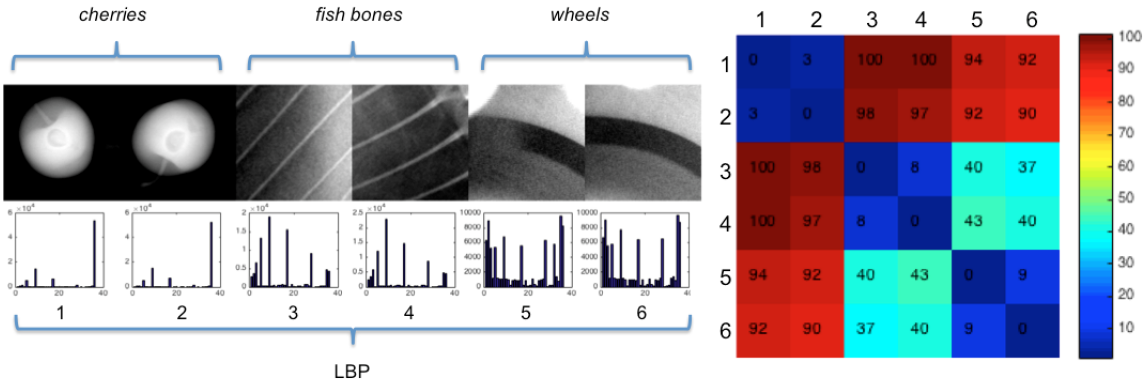

Fig. 47: Comparison of six textures using LBP-ri descriptor. It is clear that descriptors of the same texture are very similar, and descriptors from different textures are very different. A measurement of the Euclidean distance between all six descriptors is shown in the right color matrix.

4.4 Filter banks

Filter banks can be used to represent visual information using a transformation of the original image. In this Section we present some filters used in the detection of defects.

4.4.1 Fourier and Cosine Transforms

They are used in image transformations like Discrete Fourier Transform (DFT) (magnitude and phase), Discrete Cosine Transform (DCT) [18], and Wavelets as Gabor features based on 2D Gabor functions (see Section 4.4.2).

For an image $\mathbf{X}$ of $N \times N$ pixels, the Discrete Fourier Transformation in 2D is defined as follows:

$$F(m,n) = \sum_{i=1}^{M}\sum_{k=1}^{N} X(i,k) e^{-2\pi j\left(\frac{(m-1)(i-1)}{N} + \frac{(n-1)(k-1)}{N}\right)} \tag{62}$$

where $j = \sqrt{-1}$. $F(m,n)$ is a complex number. That means magnitude and phase can be used as features.

Discrete Cosine Transform in 2D is defined as:

$$D(m,n) = \alpha_m \alpha_n \sum_{i=1}^{N}\sum_{k=1}^{N} X(i,k) \cos\left(\frac{\pi(2i-1)(m-1)}{2N}\right) \cos\left(\frac{\pi(2k-1)(n-1)}{2N}\right) \tag{63}$$

where $\alpha_1 = 1/\sqrt{N}$ and $\alpha_m = \sqrt{2/N}$, for $m = 2 \ldots N$. DCT features are real numbers instead of complex number such as Fourier features.

It is worth mentioning that these features are not rotation invariant, however, we can extract rotation invariant features if we use maximum, minimum and a range of them.

4.4.2 Gabor

The Gabor functions are Gaussian shaped band-pass filters, with dyadic treatment of the radial spatial frequency range and multiple orientations, which represent an appropriate choice for tasks requiring simultaneous measurement in both space and frequency domains. The Gabor functions are a complete (but a nonorthogonal) basis set given by:

$$f(x,y) = \frac{1}{2\pi\sigma_x\sigma_y} \exp\left(-\frac{1}{2}\left(\frac{x^2}{\sigma_x^2} + \frac{y^2}{\sigma_y^2}\right)\right) \tag{64}$$

where σ_x and σ_y denote the Gaussian envelope along the x and y-axes, and u_0 defines the radial frequency of the Gabor function. Examples of Gabor functions are illustrated in Fig. 48. In this case a class of self-similar functions are generated by rotation and dilation of $f(x,y)$.

Each Gabor filter has a real and an imaginary component that are stored in $M \times M$ masks, called $\mathbf{R}_{pq}$ and $\mathbf{I}_{pq}$ respectively, where $p = 1 \ldots S$, denotes the scale, and $q = 1 \ldots L$, denotes the orientation (for details see [35]). Usually, $S = 8$ scales, and $L = 8$ orientations as shown in Fig. 48, with $M = 27$.

The Gabor filters are applied to each segmented window $\mathbf{W}$, that contains the segmented region and its surrounding (see Fig. 39). The filtered windows $\mathbf{G}_{pq}$ are computed using the 2D convolution (9) of the window $\mathbf{W}$ of the X-ray image with the Gabor masks as follows:

$$\mathbf{G}_{pq} = \left[(\mathbf{W} * \mathbf{R}_{pq})^2 + (\mathbf{W} * \mathbf{I}_{pq})^2\right]^{1/2} \tag{65}$$

The Gabor features, denoted by g_{pq}, are defined as the average output of $\mathbf{G}_{pq}$, *i.e.*, it yields $S \times L$ Gabor features for each segmented window:

$$g_{pq} = \frac{1}{n_w n_w} \sum_{i=1}^{n_w}\sum_{j=1}^{m_w} G_{pq}(i,j) \tag{66}$$

where the size of the filtered windows $\mathbf{G}_{pq}$ is $n_w \times m_w$.

Three additional Gabor features can be extracted: *i)* maximum of all Gabor features: $g_{\max} = \max(\mathbf{g})$, *ii)* minimu of all Gabor features: $g_{\min} = \min(\mathbf{g})$, and *iii)* range of all Gabor features: $g_\Delta = g_{\max} - g_{\min}$. These features are very useful because they are rotation invariant.

4.5 Local descriptors

Descriptors have been very relevant on computer vision applications [60]. This is because they are able to provide highly distinctive features, and can be used in applications such as multiple view analysis, in object recognition, texture recognition, and others. In this Section we provide some descriptors that are very useful in X-ray testing.

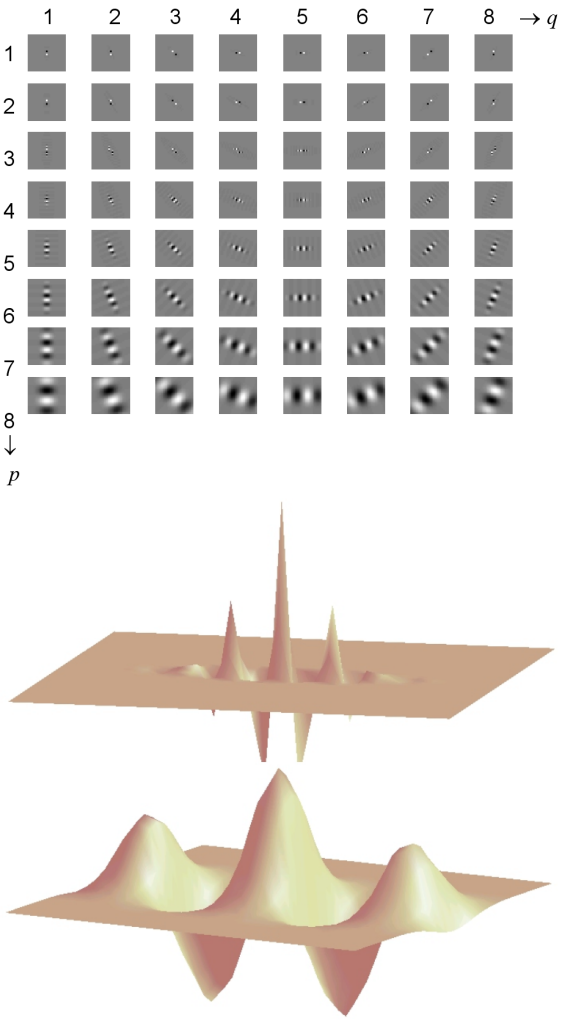

Fig. 48: Example of Gabor functions in spatial domain: Top) imaginary components of self-similar filter bank by using $p = 1 \dots 8$ scales and $q = 1 \dots 8$ orientations, Bottom) 3D representations of two Gabor functions of a).

4.5.1 SIFT

The SIFT-descriptors (Scale Invariant Feature Transform) are local features based on the appearance of the object. The descriptors are invariant to scale, rotation, lighting, noise and minor changes in viewpoint [41]. In addition, they are highly distinctive and relatively easy to extract and match against a (large) database of local features. The SIFT-descriptor consists of a 128-element vector, which corresponds to a set of 16 gradient oriented histograms of 8 bins distributed in a 4×4 grid.

4.5.2 SURF

The Speeded Up Robust Feature (SURF) is based on the sum of the Haar Wavelet response in 4×4 square sub-regions of the patch [2]. Similar to SIFT, SURF–descriptor is invariant to scale and rotation. The descriptor has 64 elements.

4.5.3 HOG

HOG is a descriptor that contains histograms of oriented gradients in cells distributed in a grid manner across the image [13].

4.5.4 BRISK

The Binary Robust Invariant Scalable Keypoint (BRISK) is a descriptor composed as a binary string by concatenating the results of simple brightness comparison tests using a circular sampling pattern [38]. The descriptor has 64 elements.

4.6 Sparse representation

In the sparse representation approach, a dictionary is built from the learning images, and testing is done by reconstructing the testing image using a sparse linear combination of the atoms of the dictionary. The testing image is assigned to the class with the minimal reconstruction error [75]. We tested two well known sparse representation approaches. In the first one, the dictionary is the gallery of training images itself [78]. In the second one, the dictionary is computed from the training images using KSVD algorithm [1].

4.7 Deep models

In recent years, deep learning has been successfully used in image and video recognition (see for example [37], [3] and [71]), and it 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. There are several deep architectures such as deep neural networks, convolutional neural networks, energy based models, Boltzmann machines, and deep belief networks, among others [3]. Convolutional neural networks (CNN), originally inspired by a biological model [36], is a very powerful class of methods for image recognition [34] which replaces feature extraction and classification with a single neural network. A CNN maps an input image $\mathbf{X}$ onto an output vector $\mathbf{y} = \mathcal{G}_L(\mathbf{X})$, where the function $\mathcal{G}_L$ can be viewed as a feed-forward network with L layers which are typically

linear operations followed by non-linearities (*e.g.*, convolutions followed by rectified linear units [62]). These layers f_l, for $l = 1 \dots L$ contain parameters $\mathbf{w} = (\mathbf{w}_1, \cdots \mathbf{w}_L)$ that can be discriminatively learned from training data: a set of input images $\mathbf{X}_i$ and their corresponding labels $\mathbf{z}_i$, for $i = 1, \cdots n$, so that $\sum_i \ell(\mathcal{G}_L(\mathbf{X}_i), \mathbf{z}_i)/n \rightarrow \min$, where ℓ is a loss function. This optimization problem can be solved using the back-propagation algorithm [70].

4.8 Example

For the detection task, two general approaches can be used: a traditional image segmentation (as explained in Section 3.4) or a *sliding–window* approach. In the first case, image processing algorithms are used (*e.g.*, histograms, edge detection, morphological operations, filtering, etc. [18]). Nevertheless, inherent limitations of traditional segmentation algorithms for complex tasks and increasing computational power have fostered the emergence of an alternative approach based on the so-called *sliding–window* paradigm. Sliding–window approaches have established themselves as state–of–art in computer vision problems where a visually complex object must be separated from the background (see, for example, successful applications in face detection [77] and human detection [12]). In the sliding–window approach, a detection window is moved over an input image in both horizontal and vertical directions, and for each localization of the detection window, a classifier decides to which class the corresponding portion of the image belongs according to its features. Here, a set of candidate image areas are selected and all of them are fed to the subsequent parts of the image analysis algorithm. This resembles a brute force approach where the algorithm explores a large set of possible segmentations, and at the end the most suitable is selected by the classification steps.

An example for casting inspection using sliding-windows is presented as follows. From a dataset containing around 47.500 cropped X-ray images of 32 $\times$ 32 pixels with defects and no-defects in aluminum wheels, we defined a training and testing dataset. We extracted LBP features from each cropped image, and we trained a Support Vector Machine (SVM). In order to illustrate the effectiveness of the use of this approach based on descriptors, we implemented a very simple sliding-window strategy for a whole X-ray image that was not used in the training set. Fig. 49 shows the obtained results. In this case, the size of the sliding-window is 32 $\times$ 32 pixels, *i.e.*, the size of the cropped images of the dataset. We observe that the small defect present at the edge of the regular structure (see arrow in left image) could be detected perfectly. In this implementation, the size of the image is 140 $\times$ 200 pixels, and the step of the sliding-window process was 1 pixel. The heat-map of this detection, obtained by superimposing Gaussian-masks when a detection window is classified as 'defect' is illustrated in the middle image. Evidently, a more robust sliding-windows strategy must be tested at different scales in which saliency maps can be used to pre-filter false detection and to reduce the search space of candidate sliding-windows. An evaluation on a broader data base is necessary.

5 Multiple View Analysis

In this Section, we present a method for the automated detection of flaws based on *tracking principle* in an X-ray image sequence, *i.e.*, first, it identifies potential defects in each image of the sequence, and second, it matches and tracks these from image to image. The key idea is to consider as false alarms those potential defects which cannot be tracked in the sequence [52]. The method for automated flaw detection presented here has basically two steps (see Fig. 50): *identification* and *tracking of potential flaws* (based on a geometric model). These will be described in this Section.

5.1 Geometric model

The X-ray image of a test object corresponds to a projection in perspective, where a 3D point of the test object is viewed as a pixel in the digital X-ray image, as illustrated in Fig. 53. A geometric model that describes this projection can be very useful for 3D reconstruction and for data association between different views of the same object. Thus, 3D features or multiple view 2D features can be used to improve the diagnosis performed by using a single view.

For the geometric model, four coordinate systems are used (see Fig. 53):

• OCS (X, Y, Z): Object Coordinate System, where a 3D point is defined using coordinates attached to the test object.

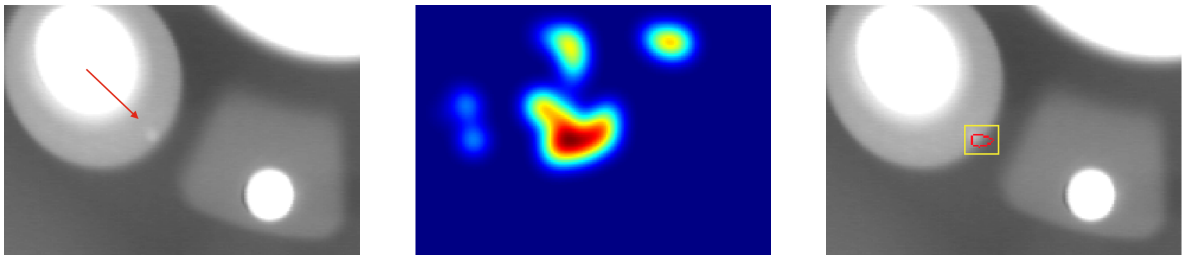

Fig. 49: Defect detection in an X-ray image using a sliding-window approach: left) original image with a small defect (see arrow), middle) heat map after sliding-window, c) detection by thresholding the heat map.

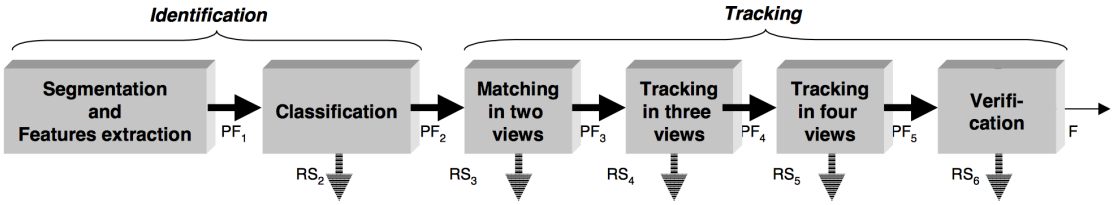

Fig. 50: Automated flaw detection in aluminum castings based on the tracking of potential defects in an X-ray image sequence: PF = potential flaws, RS = potential flaws classified as regular structures, F = detected flaws [52].

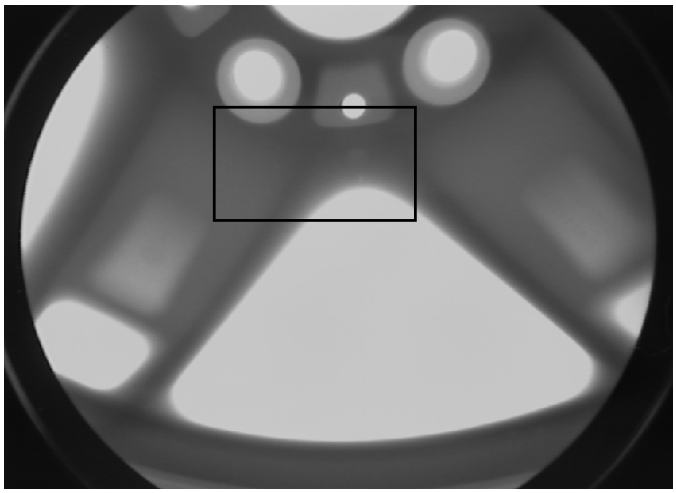

Fig. 51: X-ray image `C0001_0030` of an aluminum wheel (see zoom in Fig. 52).

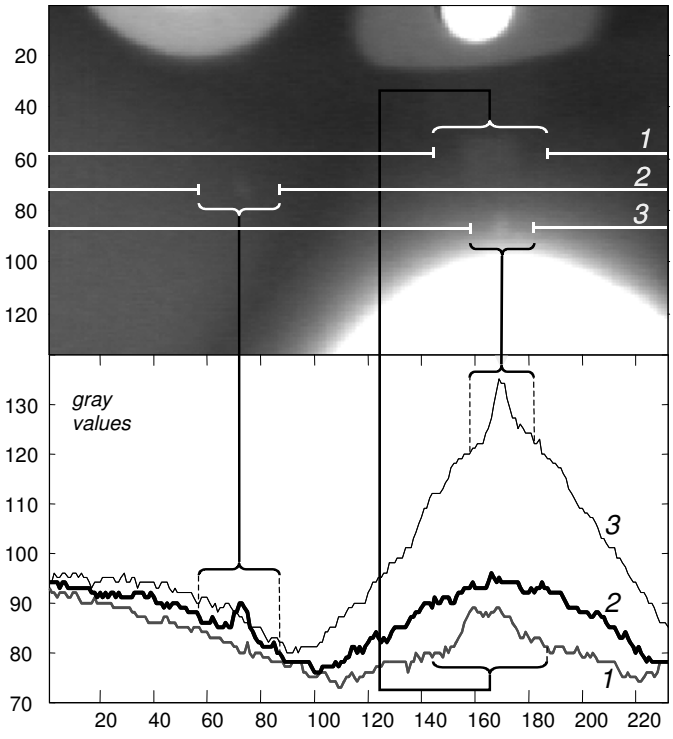

Fig. 52: Zoom of Fig. 51 and gray level profile along three rows crossing defects.

• WCS $(\bar{X}, \bar{Y}, \bar{Z})$: World Coordinate System, where the origin corresponds to the optical center (X-ray source) and the $\bar{Z}$ axis is perpendicular to the projection plane of the detector.

• PCS (x, y): Projection Coordinate System, where the 3D point is projected into projection plane $\bar{Z} = f$, and the origin is the intersection of this plane

with $\bar{Z}$ axis.
• ICS (u, v): Image Coordinate System, where a projected point is viewed in the image. In this case, (x, y)–axes are set to be parallel to (u, v)–axes.

The geometric model OCS $\rightarrow$ ICS, *i.e.*, transformation $\mathbf{P} : (X, Y, Z) \rightarrow (u, v)$, can be expressed in homogeneous coordinates as [46]:

$$\lambda \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \mathbf{P} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}, \tag{67}$$

where λ is a scale factor and $\mathbf{P}$ is a 3×4 matrix modeled as three transformations:
i) OCS $\rightarrow$ WCS, *i.e.*, transformation $\mathbf{T}_1 : (X, Y, Z) \rightarrow (\bar{X}, \bar{Y}, \bar{Z})$, using a 3D rotation matrix $\mathbf{R}$, and 3D translation vector $\mathbf{t}$;
ii) WCS $\rightarrow$ (PCS), *i.e.*, transformation $\mathbf{T}_2 : (\bar{X}, \bar{Y}, \bar{Z}) \rightarrow (x, y)$, using a perspective projection matrix that depends on focal distance f; and
iii) PCS $\rightarrow$ ICS, *i.e.*, transformation $\mathbf{T}_3 : (x, y) \rightarrow (u, v)$, using scales factor α_x and α_y, and 2D translation vector (u_0, v_0).
The three transformations OCS $\rightarrow$ WCS $\rightarrow$ PCS $\rightarrow$ ICS are expressed as:

$$\mathbf{P} = \underbrace{\begin{bmatrix} \alpha_x & 0 & u_0 \\ 0 & \alpha_x & v_0 \\ 0 & 0 & 1 \end{bmatrix}}_{\mathbf{T}_3} \underbrace{\begin{bmatrix} f & 0 & 0 & 0 \\ 0 & f & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}}_{\mathbf{T}_2} \underbrace{\begin{bmatrix} \mathbf{R} & \mathbf{t} \\ \mathbf{0}^\mathsf{T} & 1 \end{bmatrix}}_{\mathbf{T}_1} \tag{68}$$

The parameters included in matrix $\mathbf{P}$ can be estimated using a calibration approach as illustrated in Fig. 54 [24].

In order to obtain multiple views of the object, m different projections of the test object can be achieved by rotating and translating it (for this task a manipulator can be used as shown in Fig. 55). For the k-th projection, for $k = 1 \ldots m$, the geometric model $\mathbf{P}_k$ used in (67) is computed from (68) including 3D rotation matrix $\mathbf{R}_k$ and 3D translation $\mathbf{t}_k$.

Our strategy deals with detections in multiple views. In this problem of data association, the aim is to find the correct correspondence among different views. For this reason, we use multiple view geometric constraints to reduce the number of matching candidates between monocular detections. In our approach, the geometric constraints are established from bifocal (epipolar) and trifocal geometry [24]. Thus, for a point $\mathbf{x}_i$ in view $\mathbf{I}_i$, the corresponding point $\mathbf{x}_j$ in a second view $\mathbf{I}_j$ must lie on its *epipolar line* estimated by using the bifocal tensors (or fundamental matrix) $\mathcal{F}_{ij}$ of views i and j. On the other hand, for a point $\mathbf{x}_i$ in view $\mathbf{I}_i$ and its corresponding point $\mathbf{x}_j$ in view $\mathbf{I}_j$, the corresponding point $\mathbf{x}_k$ in a third view $\mathbf{I}_k$ is a point estimated by using the trifocal tensors $\mathcal{T}_{ijk}$ of views i, j and k. Multifocal tensors can be estimated from projection matrices $\mathbb{P} = \{\mathbf{P}_i\}_{i=1}^m$, where $\mathbf{P}_i$ is used to calculate the projection of a (3D) point $\mathbf{X}$ of the test object into a (2D) point $\mathbf{x}_i$ of image $\mathbf{I}_i$. The projection is computed as $\lambda \mathbf{x}_i = \mathbf{P}_i \mathbf{X}$ using homogeneous coordinates, where λ is a scale factor. The fundamental matrix $\mathcal{F}_{ij}$ is calculated from $\mathbf{P}_i$ and $\mathbf{P}_j$, and the trifocal tensors $\mathcal{T}_{ijk}$ from $\mathbf{P}_i$, $\mathbf{P}_j$ and $\mathbf{P}_k$ (see details in [24]).

The estimation of $\mathbb{P}$ can be performed by minimizing the error between real and modeled projection 3D $\rightarrow$ 2D. Two well known approaches can be used: *i) calibration*, in which known 3D points from a calibration object is used [24], [46] or *ii) bundle adjustment*, in which both 3D points and projection matrices are calculated from views of the test object itself using stable tracked keypoints across multiple views [76], [50].

5.2 Identification of potential flaws

A digital X-ray image sequence of the object test is acquired (see for example series C0001 of $\mathbb{GDX}$ray [59]). In order to ensure the tracking of flaws in the X-ray images, similar projections of the specimen must be achieved along the sequence. For this reason, the sequence consists of X-ray images taken by the rotation of the casting at small intervals (*e.g.*, 5^0). Since many images are captured, the time of the data acquisition is reduced by taking the images without frame averaging. The position of the casting, provided on-line by the manipulator is registered at each X-ray image to calculate the perspective projection matrix $\mathbf{P}_p$ of image p of the sequence (for details see Section 5.1). An X-ray image sequence is shown in Fig. 56.
The detection of potential flaws identifies regions in X-ray images that may correspond to real defects. This process takes place in each X-ray image of the sequence without considering information about the correspondence between them. Two general characteristics of the defects are used for identification

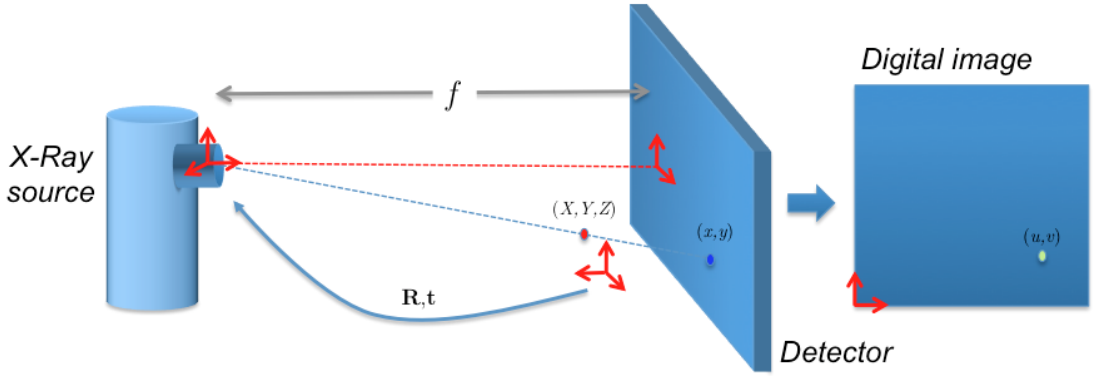

Fig. 53: Geometric model: a 3D point (X, Y, Z) is projected as a 2D point (u, v).

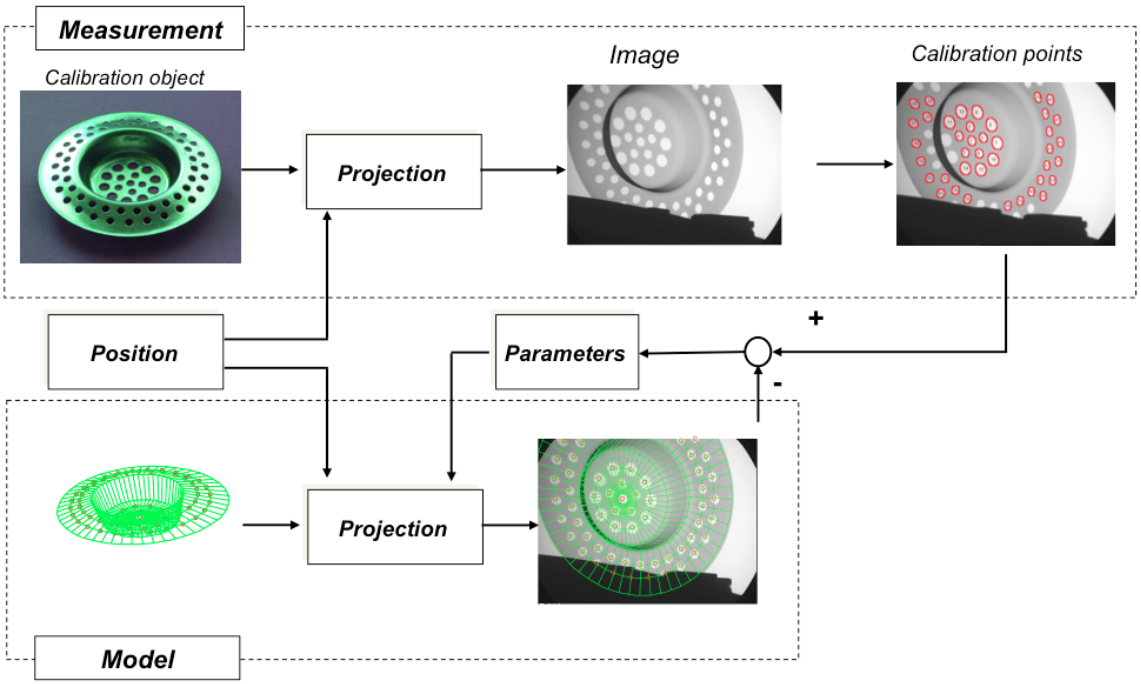

Fig. 54: Geometric calibration of a computer vision X-ray imaging system: the modeled projection of a 3D model coincides with real geometry of the calibration object.

purposes: *i)* a flaw can be considered as a connected subset of the image, and *ii)* the gray level difference between a flaw and its neighborhood is significant. However, as the signal-to-noise ratio in our X-ray images is low, the flaws signal is slightly greater than the background noise, as illustrated in Fig. 52. In our experiments, the mean gray level of the flaw signal (without background) was between 2.4 and 28.8 gray values with a standard deviation of 6.1. Analyzing a homogeneous background in different areas of interest of normal parts, we found that the noise signal was within ± 13 gray values with a standard deviation of 2.5. For this reason, the identification of real defects with poor contrast can also involve the detection of false alarms.

According to the mentioned characteristics of the real flaws, our method of identification has the following two steps (see Fig. 42): Edge detection and Segmentation and classification of potential flaws. In this case, the approach based on crossing line profiles can be used (see Section 4.2.3). A statistical study of the classification of potential flaws using more than 70 features can be found in [53].

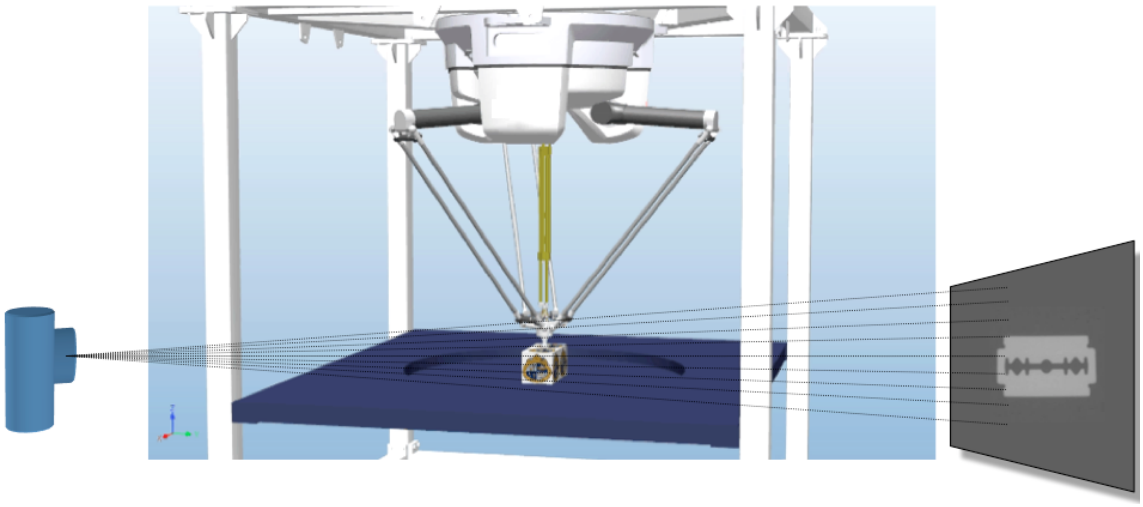

Fig. 55: Generally a manipulator is used to locate the test object in a desired position.

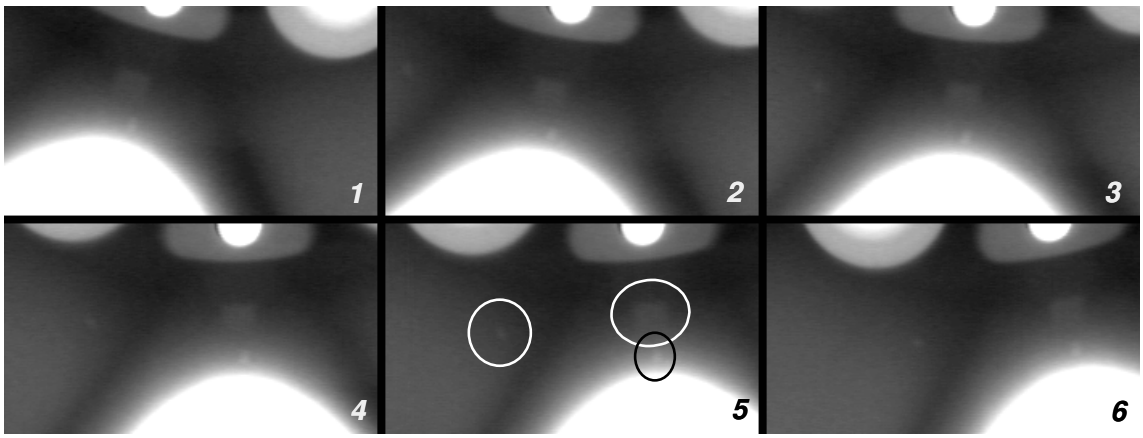

Fig. 56: X-ray image sequence with three flaws (image 5 is shown in Fig. 52).

This is a very simple detector of potential flaws. However, the advantages are as follows: a) it is a single detector (it is the same detector for each image), and b) it is able to identify potential defects independent of the placement and the structure of the specimen.

Using this method, some real defects cannot be identified in all X-ray images in which they appear if the contrast is very poor or the flaw is not enclosed by edges. For example, in Fig. 57 one can observe that the biggest real flaw was identified in images 1, 2, 3, 4 and 6, but not in image 5 where only two of the three real flaws were identified (compare with Fig. 52). Additionally, if a flaw is overlapped by edges of the structure of the casting, not all edges of the flaw can be detected. In this case, the flaw will not be enclosed and therefore not be segmented. Furthermore, a small flaw that moves in front (or behind) a thick cross section of the casting, in which the X-rays are highly absorbed, may cause an occlusion. In our experiments, this detector identified the real flaws in four or more (not necessarily consecutive) images of the sequence.

5.3 Multiple view detection

In the previous step, n_1 potential regions were segmented and described in the entire image sequence $\mathbb{I}$. Each segmented region is labeled with a unique number $r \in \mathbf{T}_1 = \{1, ..., n_1\}$. In view i, there are m_i segmented regions that are arranged in a subset $\mathbf{t}_i = \{r_{i,1}, r_{i,2}, ..., r_{i,m_i}\}$, *i.e.*, $\mathbf{T}_1 = \mathbf{t}_1 \cup \mathbf{t}_2 \cup ...\mathbf{t}_m$. The matching and tracking algorithms combine all regions to generate consistent tracks of the object's

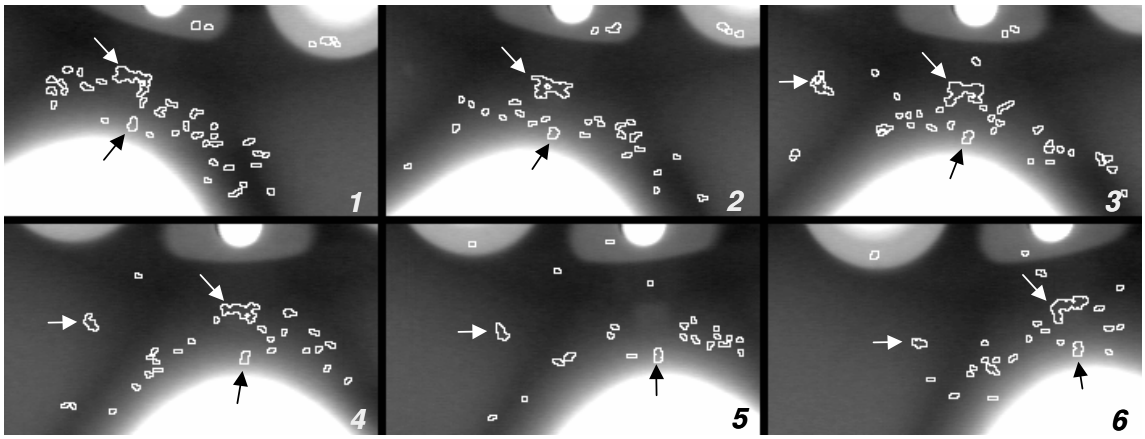

Fig. 57: Identification of potential flaws (the arrows indicate real flaws).

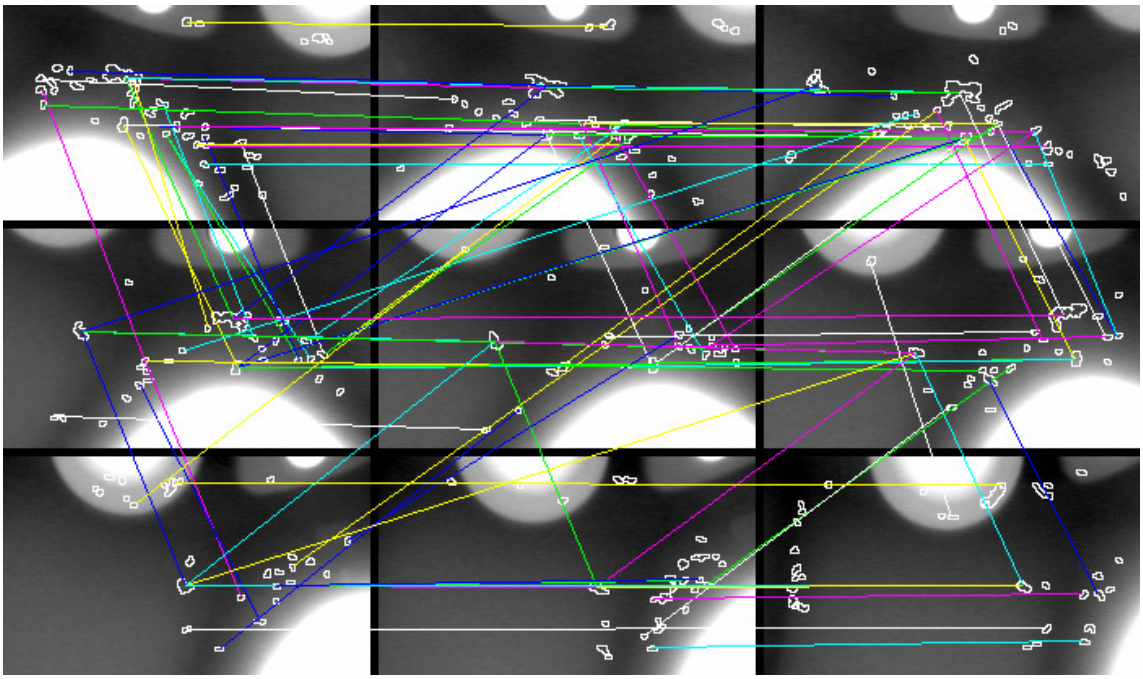

Fig. 58: Matching of potential flaws in two views.

parts of interest across the image sequence. The algorithm has the following steps:

Matching in two views All regions in view i that have corresponding regions in the next p views are searched, *i.e.*, regions $r_1 \in \mathbf{t}_i$ that have corresponding regions $r_2 \in \mathbf{t}_j$ for $i = 1, ..., m-1$ and $j = i+1, ..., \min(i+p, m)$. In our experiments, we use $p = 3$ to reduce the computational cost. The matched regions (r_1, r_2) are those that meet *similarity* and *location* constraints. The similarity constraint means that corresponding descriptors $\mathbf{y}_{r_1}$ and $\mathbf{y}_{r_2}$ must be similar enough such that:

$$||\mathbf{y}_{r_1} - \mathbf{y}_{r_2}|| < \varepsilon_1. \tag{69}$$

The location constraint means that the corresponding locations of the regions must meet the epipolar constraint. In this case, the Sampson distance between $\mathbf{x}_{r_1}$ and $\mathbf{x}_{r_2}$ is used, *i.e.*, the first-order geometric error of the epipolar constraint must be small enough such that:

$$|\mathbf{x}_{r_2}^{\mathsf{T}} \mathcal{F}_{ij} \mathbf{x}_{r_1}| \left(\frac{1}{\sqrt{a_1^2 + a_2^2}} + \frac{1}{\sqrt{b_1^2 + b_2^2}} \right) < \varepsilon_2, \tag{70}$$

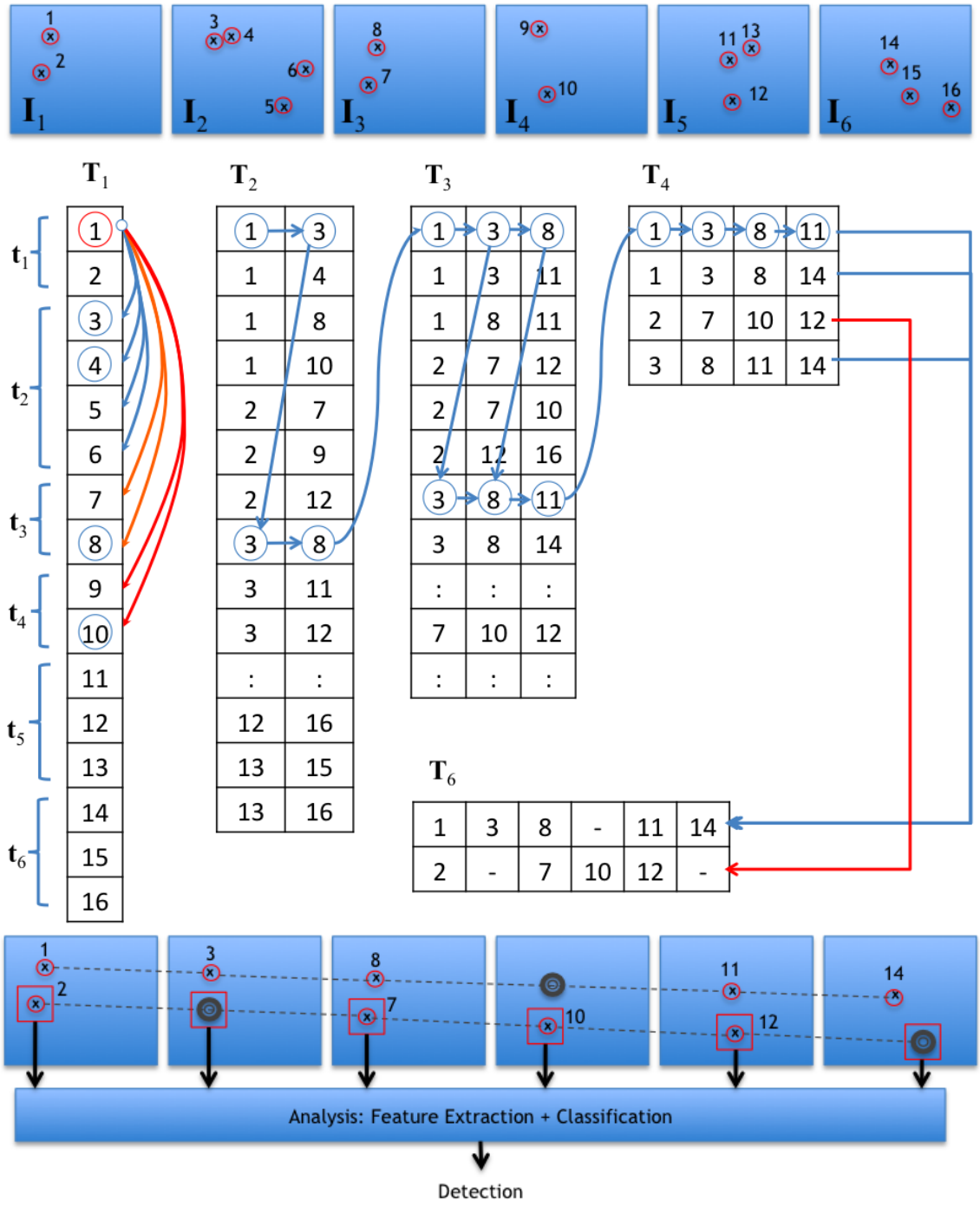

Fig. 59: Tracking example with $m = 6$ views. In each view there are 2, 4, 2, 2, 3 and 3 segmented regions, *i.e.*, there are $n_1 = 16$ regions in total. For each region we seek corresponding regions in the next $p = 3$ views (see matching arrows in $\mathbf{T}_1$: region 1 with regions (3,4,5,6) in view 2, regions (7,8) in view 3, and (9,10) in view 4). We observe that after tracking in 2, 3 and 4 views there are only two tracks in $\mathbf{T}_6$ that could be tracked in 5 and 4 views respectively. The regions that were not segmented can be recovered by reprojection (see gray circles in views 2, 4 and 6). Finally, each set of tracked regions are analyzed in order to take the final decision.

with $\mathcal{F}_{ij}\mathbf{x}_{r_1} = [a_1\ a_2\ a_3]^\mathsf{T}$ and $\mathcal{F}_{ij}^\mathsf{T}\mathbf{x}_{r_2} = [b_1\ b_2\ b_3]^\mathsf{T}$. In this case, $\mathcal{F}_{ij}$ is the fundamental matrix between views i and j calculated from projection matrices $\mathbf{P}_i$ and $\mathbf{P}_j$ [24] (see Section 5.1). In addition, the location constraint used is as follows:

$$||\mathbf{x}_{r_1} - \mathbf{x}_{r_2}|| < \rho(j-i), \tag{71}$$

because the translation of corresponding points in these sequences is smaller than ρ pixels in consecutive frames.

If we have 3D information about the space where our test object should be, it is worth to evaluating whether the 3D point reconstructed from the centers of mass of the regions must belong to the space occupied by the casting. From $\mathbf{m}_p^a$ and $\mathbf{m}_q^b$ the corresponding 3D point $\hat{\mathbf{M}}$ is estimated using the linear approach of Hartley in [23]. For two views this approach is faster than the least squares technique. It is necessary to examine if $\hat{\mathbf{M}}$ resides in the volume of the casting, the dimensions of which are usually known a priori (*e.g.*, a wheel is assumed to be a cylinder)[5].

Finally, a new matrix $\mathbf{T}_2$ sized $n_2 \times 2$ is obtained with all matched duplets (r_1, r_2), one per row. If a region is found to have no matches, it is eliminated. Multiple matching, *i.e.*, a region that is matched with more than one region, is allowed. Using this method, problems like non-segmented regions or occluded regions in the sequence can be solved by tracking if a region is not segmented in consecutive views.

Matching in 3 views Based on the matched regions stored in matrix $\mathbf{T}_2$, we look for triplets (r_1, r_2, r_3), with $r_1 \in \mathbf{t}_i$, $r_2 \in \mathbf{t}_j$, $r_3 \in \mathbf{t}_k$ for views i, j and k. We know that a row a in matrix $\mathbf{T}_2$ has a matched duplet $[T_2(a,1)\ T_2(a,2)] = [r_1\ r_2]$. We then look for rows b in $\mathbf{T}_2$ in which the first element is equal to r_2, *i.e.*, $[T_2(b,1)\ T_2(b,2)] = [r_2\ r_3]$. Thus, a matched triplet (r_1, r_2, r_3) is found if the regions r_1, r_2 and r_3 meet the trifocal constrain:

$$||\hat{\mathbf{x}}_{r_3} - \mathbf{x}_{r_3}|| < \varepsilon_3, \tag{72}$$

This means that $\mathbf{x}_{r_3}$ must be similar enough to the re-projected point $\hat{\mathbf{x}}_{r_3}$ computed from the points in views i and j ($\mathbf{x}_{r_1}$ and $\mathbf{x}_{r_2}$), and the trifocal tensors $\mathcal{T}_i^{jk}$ of views i, j, k calculated from projection matrices $\mathbf{P}_i$, $\mathbf{P}_j$ and $\mathbf{P}_k$ [24] (see Section (5.1)). A new matrix $\mathbf{T}_3$ sized $n_3 \times 3$ is built with all matched triplets (r_1, r_2, r_3), one per row. Regions in which the three views do not match are eliminated.

The results of our example are shown in Fig. 58.

Matching in more views For $v = 4, ..., q \leq m$ views, we can build the matrix recursively $\mathbf{T}_v$, sized $n_v \times v$, with all possible v-tuplets $(r_1, r_2, ..., r_v)$ that fulfill $[T_{v-1}(a,1)\ ...\ T_{v-1}(a,v-1)] = [r_1\ r_2\ ...\ r_{v-1}]$ and $[T_{v-1}(b,1)\ ...\ T_{v-1}(b,v-1)] = [r_2\ ...\ r_{l-1}\ r_v]$, for $j, k = 1, ..., n_{v-1}$. No more geometric constraints are required because it is redundant. The final result is stored in matrix $\mathbf{T}_q$. For example, for $q = 4$ we store in matrix $\mathbf{T}_4$ the matched quadruplets (r_1, r_2, r_3, r_4) with $r_1 \in \mathbf{t}_i$, $r_2 \in \mathbf{t}_j$, $r_3 \in \mathbf{t}_k$, $r_4 \in \mathbf{t}_l$ for views i, j, k and l.

Fig. 60 shows the tracked regions of our example that fulfill this criterion. Only two false trajectories are observed (see arrows).

As our detector cannot guarantee the identification of all real flaws in more than four views, a tracking in five views could lead to the elimination of those real flaws that were identified in only four views. However, if a potential flaw is identified in more than four views, more than one quadruplet can be detected. For this reason these corresponding quadruplets are joined in a trajectory that contains more than four potential flaws (see trajectory with arrows in Fig. 60).

The matching condition for building matrix $\mathbf{T}_i, i = 3, ..., q$, is efficiently evaluated (avoiding an exhaustive search) by using a k-d tree structure [4] to search the nearest neighbors for zero Euclidean distance between the first and last $i-2$ columns in $\mathbf{T}_{i-1}$.

Merging tracks Matrix $\mathbf{T}_q$ defines tracks of regions in q views. It can be observed that some of these tracks correspond to the same region. For this reason, it is possible to merge tracks that have $q-1$ common elements. In addition, if a new track has more than one region per view, we can select the region that shows the minimal reprojection error after computing the corresponding 3D location. In this case, a 3D reconstruction of $\hat{\mathbf{X}}$ is estimated from tracked points [24]. Finally, matrix $\mathbf{T}_m$ is obtained with all merged tracks in the m views. See an

5. It is possible to use a CAD model of the casting to evaluate this criterion more precisely. Using this model we could discriminate a small hole of the regular structure that is identified as a potential flaw. Additionally, the CAD model can be used to inspect the casting geometry, as shown in [63].

example of the whole tracking algorithm in Fig. 59.

Analysis The 3D reconstructed point $\hat{\mathbf{X}}$ from each set of tracked points of $\mathbf{T}_m$ can be reprojected in views where the segmentation may have failed to obtain the complete track in all views. The reprojected points of $\hat{\mathbf{X}}$ should correspond to the centroids of the non-segmented regions. It is then possible to calculate the size of the projected region as an average of the sizes of the identified regions in the track. In each view, a small window centered in the computed centroids is defined. These corresponding small windows, referred to as *tracked part*, will be denoted as $\mathbb{W} = \{\mathbf{W}_1, ..., \mathbf{W}_m\}$. In each view a small window is defined with the estimated size in the computed centers of gravities (see Fig. 61). Afterwards, the corresponding windows are averaged. Thus, the attempt is made to increase the signal-to-noise ratio by the factor $\sqrt{n}$, where n is the number of averaged windows. As flaws must appear as contrasted zones relating to their environment, we can verify if the contrast of each averaged window is greater than 2.5%. With this verification it is possible to eliminate all remaining false detections. Fig. 61 shows the detection in our sequence using this method. Our objective is then achieved: the real defects were separated from the false ones.

5.4 Experimental Results

In this Section, results of automatic inspection of cast aluminum wheels using the outlined approach are presented. These results have been achieved recently on synthetic flaws and real data. The parameters of our method have been manually tuned, giving $\sigma = 1.25$ pixels (for LoG-operator), ε_2 = 0.75 mm, ε_s = 0.7 and ε_3 = 0.9 mm. These parameters were not changed during these experiments. A wheel was considered to be a cylinder with the following dimensions: 470 mm diameter and 200 mm height. The focal length (distance between X-ray source and entrance screen of the image intensifier) was 884 mm. The bottom of a wheel was 510 mm from the X-ray source. Thus, a pattern of 1 mm in the middle of the wheel is projected in the X-ray projection coordinate system as a pattern of 1.73 mm, and in the image coordinate system as a pattern of 2.96 pixels. The sequences of X-ray images were taken by rotation of the casting at 5^0.

The detection performance will be evaluated by computing the number of true positives (TP) and false positives (FP). They are respectively defined as the number of flaws that are correctly classified and the number of misclassified regular structures. The TP and FP will be normalized by the number of existing flaws (E) and the number of identified potential flaws (I). Thus, we define the following percentages: TPP = TP / E $\times 100$ and FPP = FP / I $\times 100$. Ideally, TPP = 100% and FPP = 0%.

Synthetic Flaws To evaluate the performance of our method in critical cases, real data in which synthetic flaws have been added were examined. A simple 3D modeled flaw (a spherical bubble) was projected and superimposed on real X-ray images of an aluminum wheel according to the law of X-ray absorption [44]. In our experiment, a flaw is simulated in 10 X-ray images of a real casting, in an area that included an edge of the structure (see Fig. 62a). In this area the synthetic flaw was located in 24 different positions in a regular grid manner. At each position TPP and FPP were tabulated. This test was repeated for different sizes of the flaws ($\emptyset = 1.5 \sim 7.5$ mm) which are illustrated in Fig. 62b. The results are shown in Fig. 62c. It was observed that the FPP was always zero. The TPP was 100% for $\emptyset \geq 2.5$ mm, and greater than 95% for $\emptyset \geq 2.1$ mm. However, the identification of the flaw may fail (and therefore also its detection) if it is very small and is located at the edge of the structure of the casting. In this case one may choose a smaller value of the parameter σ in the LoG operator of the edge detection, which will unfortunately increment the FPP. Other non–critical experiments, where the area of the simulation does not include an edge of the structure, have led to perfect results (TPP = 100%, FPP = 0%) for $\emptyset \geq 1.5$ mm (≥ 4.4 pixels). Usually, the minimum detectable defect size according to inspection specifications is in the order of $\emptyset = 2$ mm. In X-ray testing, smaller flaws can be detected by decreasing the distance of the object test to the X-ray source.

Real Data Fourteen X-ray image sequences of aluminum wheels with twelve known flaws were inspected. Three of these defects were existing blow holes (with $\emptyset = 2.0 \sim 7.5$ mm). They were initially detected by a visual (human) inspection. The remaining nine flaws were produced by drilling small holes ($\emptyset = 2.0 \sim 4.0$ mm) in positions of the casting

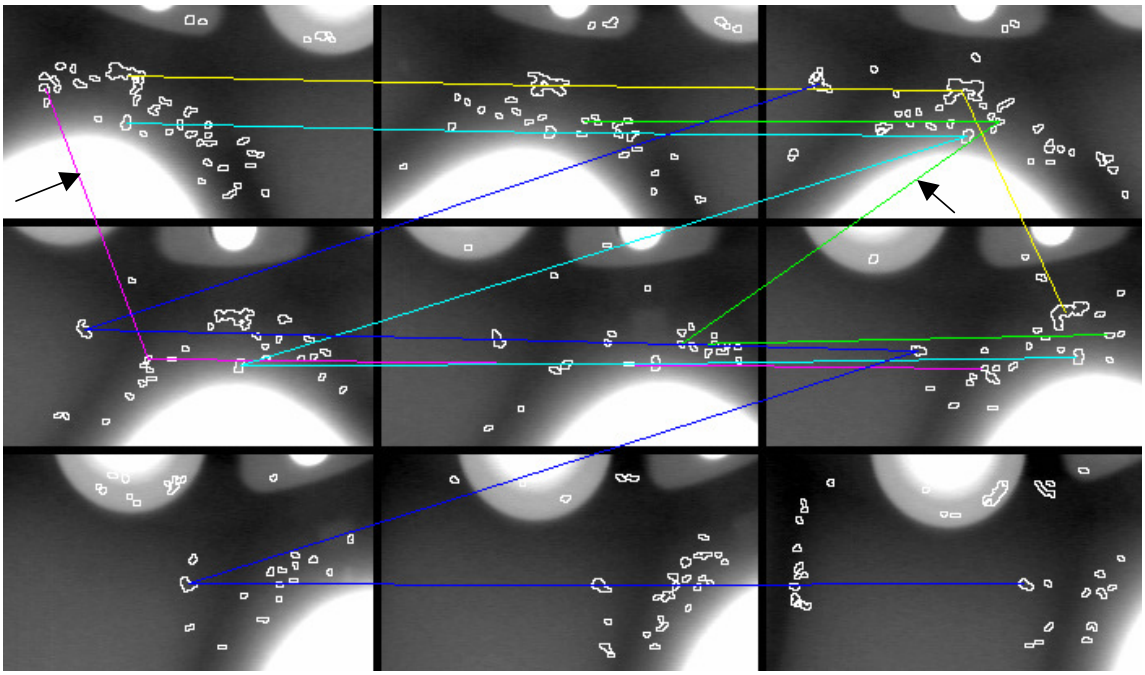

Fig. 60: Tracking in more views (the arrows indicate false detections).

which were known to be difficult to detect. Casting flaws are present only in the first seven sequences. The results are summarized in Table 2, Fig. 63, and Fig. 64. In the identification of potential flaws, it was observed that the FPP was 98% (4310/4381). Nevertheless, the TPP in this experiment was good, and it was possible to identify 85% (71/84) of all projected flaws in the sequences (13 of the existing 84 flaws were not identified because the contrast was poor or they were located at edges of regular structures). It was observed that in the next steps, the FPP was reduced to nil. The detection of the real flaws was successful in all cases. The first six images of sequence 3 and its results were already illustrated in Figures 56, 57, ... 61. The results on the other sequences with flaws are shown in Fig. 63.

Comparison with other methods In this Section we present a comparison of our proposed algorithm with other methods that can be used to detect defects in aluminum castings. In this comparison we evaluate the same real fourteen sequences used in the previous Section. The results are summarized in Table 3.

Firstly, we compared the first step of our method (*identification of potential flaws*). The objective of this step is the use of a single filter, instead of a set of filters adapted to the regular structure of the specimen. We evaluated the well known Canny filter (see for example [14]). As this filter detects sparse edge pixels that not necessarily produce at real flaws closed and connected contours, the TPP of this detector was unacceptable, only 4% of the real flaws were identified ('Canny I' in Table 3). In order to increase the number of closed regions a dilation of the edges using a 3×3 mask was performed. Although the TPP is improved to 40% ('Canny II' in Table 3), many flaws were not detected in any of the images of the sequence. For this reason, only 17% of the real flaws were detected after the tracking and verification.

Another detection of potential flaws can be per-

TABLE 3: Comparison with other methods

Method	**Identification**		**Detection**	
	TPP	FPP	TPP	FPP
Canny I	4%	97%	0%	–
Canny II	40%	99%	17%	40%
Median I	55%	85%	33%	36%
Median II	88%	98%	92%	45%
Tracking in 3	85%	98%	100%	25%
Tracking in 5	85%	98%	83%	0%
PXV-5000	–	–	100%	0%
Proposed	85%	98%	100%	0%

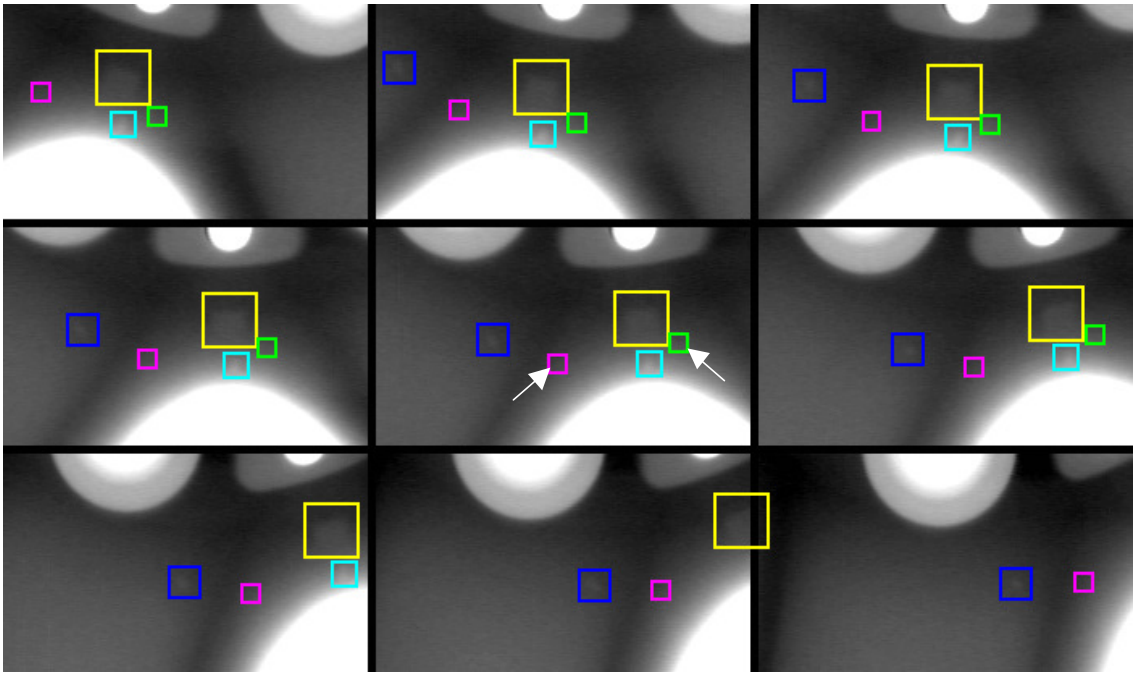

Fig. 61: Reconstruction and verification: the false detections (indicated by the arrows) are eliminated after the verification in all images of the sequence.

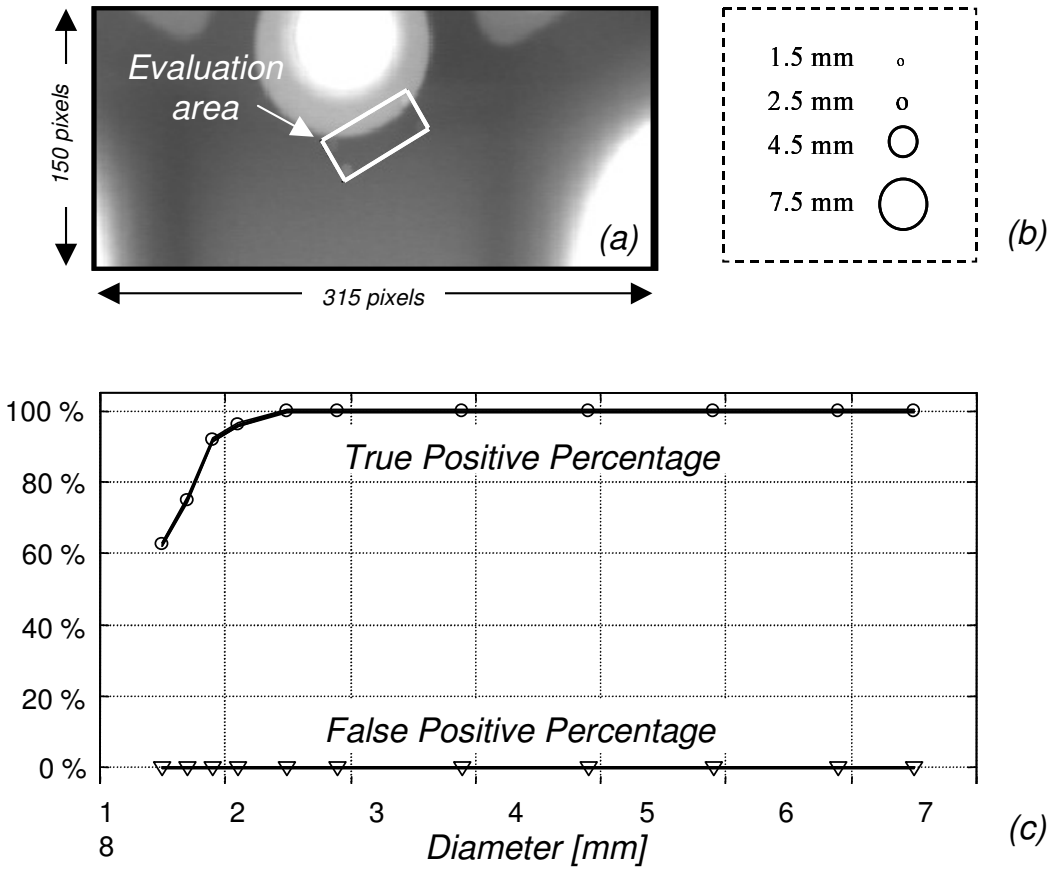

Fig. 62: Detection on synthetic flaws: a) X-ray image and evaluated area, b) flaw sizes, and c) TPP and FPP.

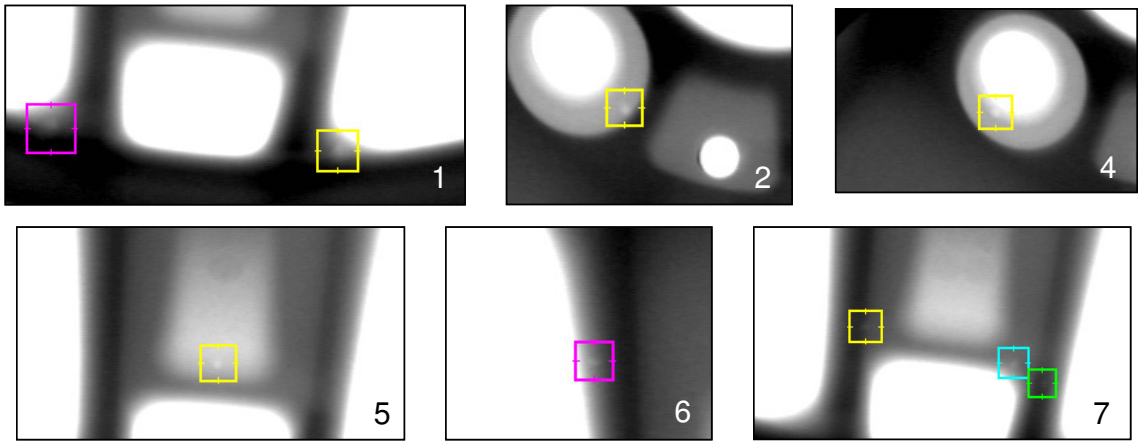

Fig. 63: Detected flaws in sequences 1, 2, 4, 5, 6 and 7 (sequence 3 is shown in Fig. 61).

TABLE 2: Detection of flaws on real data

Sequence	X-ray images	Flaws in the sequence	Flaws in the images (E)	**Identification** TP	FP	Total (I)	**Detection** TP	FP
1	10	2	12	12	249	261	2	0
2	9	1	9	8	238	246	1	0
3	9	3	23	19	253	272	3	0
4	8	1	8	4	413	417	1	0
5	6	1	6	6	554	560	1	0
6	8	1	8	8	196	204	1	0
7	6	3	18	14	445	459	3	0
8	6	0	0	0	178	178	0	0
9	9	0	0	0	256	256	0	0
10	8	0	0	0	150	150	0	0
11	8	0	0	0	345	345	0	0
12	6	0	0	0	355	355	0	0
13	6	0	0	0	365	365	0	0
14	9	0	0	0	313	313	0	0
Total	108	12	84	71	4310	4381	12	0
Percentage				85%	98%		100%	0%

formed using a region based segmentation. Median filtering is normally used to generate an error-free image, since defect structures are essentially eliminated, while design features of the test piece are normally preserved [54]. Once the error-free reference image is computed, an error difference image between original and error-free images is calculated. Casting defects are then identified when a sufficiently large gray level in the error difference image occurs. The best results were obtained using a median filter with a 11×11 mask. We evaluated two thresholds: $\theta = 6$ and $\theta = 2$ –by 256 gray levels– (see 'Median I' and 'Median II' in Table 3). In the first case the TPP was only 55%. By decreasing the threshold value we increased the TPP to 88%, that is slightly better than our detector (85%). However, systematic false alarms were detected at the corners of the regular structures. Since these false alarms satisfy the multifocal conditions, they can be tracked in the sequence. For this reason, this detector can only be used if the median filter is adapted to the regular structures of the specimen using a priori information. Normally, a set of median filters is used for each X-ray image [15], [27], [25].

In order to evaluate the second step of our method (*tracking of potential flaws*), we tested the method

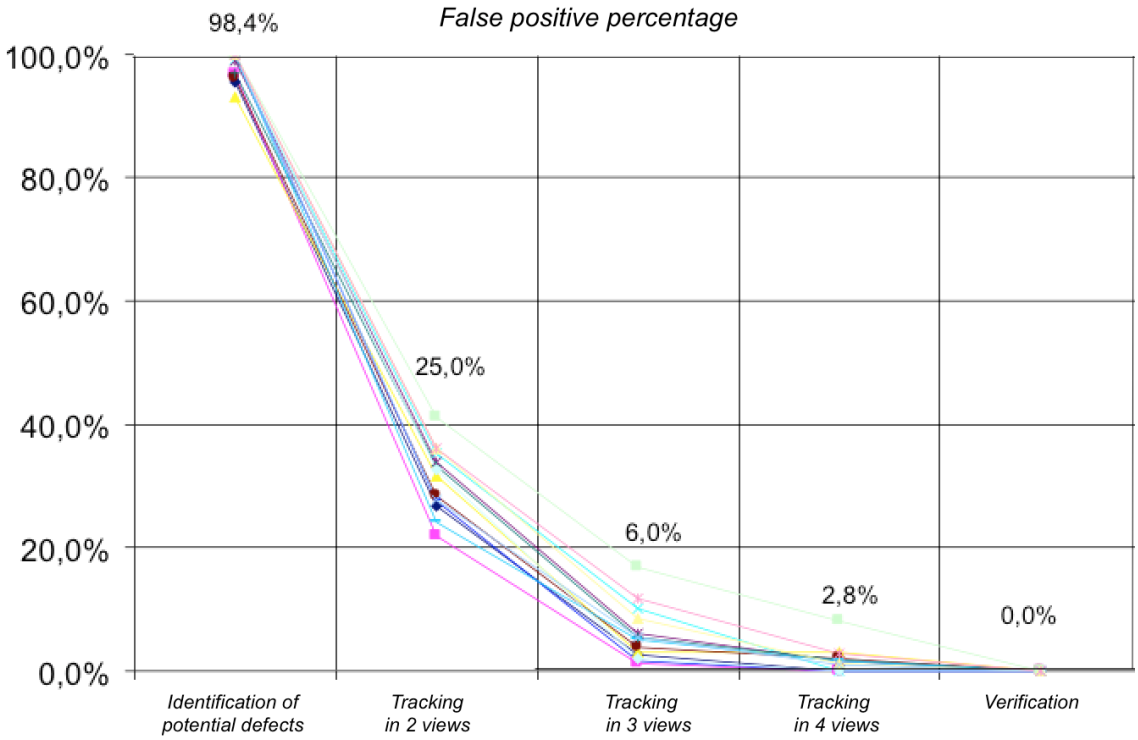

Fig. 64: False positive percentage on real data in the fourteen real sequences (the number of identified potential flaws corresponds to 100%). The mean of each step is given over the fourteen curves.

by tracking the potential flaws in 3 and in 5 views, instead of 4 views (see 'Tracking in 3', 'Tracking in 5' and 'Proposed' in Table 3). By considering only three views we obtained so many false alarms that the verification step detected 4 false alarms (25%). In the other case, by tracking the potential flaws in five views, real flaws that were segmented in only four views of the sequences were not tracked. For this reason, only 83% of the real flaws were detected.

Finally, we inspected the test castings using a classic image processing method. In our experiments, we used the industrial software PXV-5000 [33], [55]. The results were excellent: 100% of the real flaws were detected without false alarms. As a result of its peak detection performance, the classic image processing methods have become the most widely established in industrial applications. However, these method suffer from the complicated configuration of the filtering, which is tailored to the test piece. In our experiments, the configuration process has taken two weeks. Nevertheless, as our method requires only a few number of parameters, the configuration could be carried out in hours.

6 Conclusions

In this work the fundamental principles of the automated detection of die casting discontinuities have been explained. A general inspection schema was presented, and several methods that have appeared in the literature in the past thirty years were explained showing the development of this sector in the areas of industry and academia. It is worth to mention that automated defect detection in castings is very effective.

As a result of its peak detection performance, the reference inspection methods have become most widely established in industrial applications. These methods suffer from the complicated configuration of their filtering, which is tailored to the test piece. Typically, this optimization process takes one week, independently of whether it is performed manually or automatically.

On the other hand, the prerequisite for the use of a method with no a priori information of the piece's

structure, is the existence of common properties which define all casting discontinuities well and also differentiate them from design features of the test pieces. These prerequisites are often fulfilled only in special testing situations.

It is clear that the recent progress in computer technology allows the handling of various theoretical and experimental problems in science and technology which were inaccessible before. Currently, the processing of image sequences, the use of sophisticated filters in digital image processing and the design of new deep learning models –to cite a few– are possible. However, in order to asses the performance objectively, it will be necessary to analyze a broader databank.

ACKNOWLEDGMENTS

This work was supported in part by Fondecyt Grant No. 1161314 from CONICYT, Chile.

REFERENCES

[1] M. Aharon, M. Elad, and A. Bruckstein. K-SVD: An algorithm for designing overcomplete dictionaries for sparse representation. *IEEE Transactions on Signal Processing*, 54(11):4311–4322, 2006.

[2] Herbert Bay, Tinne Tuytelaars, and L. Van Gool. Surf: Speeded up robust features. In *9th European Conference on Computer Vision (ECCV2006)*, Graz Austria, May 2006.

[3] Yoshua Bengio, Aaron Courville, and Pierre Vincent. Representation learning: A review and new perspectives. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35(8):1798–1828, 2013.

[4] J.L. Bentley. Multidimensional binary search trees used for associative searching. *Communications of the ACM*, 18(9):509–517, 1975.

[5] H. Boerner and H. Strecker. Automated X-ray inspection of aluminum casting. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 10(1):79–91, 1988.

[6] J. Canny. A computational approach to edge detection. *IEEE Trans. Pattern Analysis and Machine Intelligence*, PAMI-8(6):679–698, 1986.

[7] Miguel Carrasco and Domingo Mery. Automatic multiple view inspection using geometrical tracking and feature analysis in aluminum wheels. *Machine Vision and Applications*, 22(1):157–170, 2011.

[8] K.R. Castleman. *Digital image processing*. Prentice-Hall, Englewood Cliffs, New Jersey, 1996.

[9] R. Chellappa and R. Bagdazian. Fourier coding of image boundaries. *IEEE Trans. Pattern Analysis and Machine Intelligence*, PAMI-6(1):102–105, 1984.

[10] David Coeurjolly and Reihnard Klette. A comparative evaluation of length estimators of digital curves. *IEEE transactions on pattern analysis and machine intelligence*, 26(2):252–258, 2004.

[11] Rémi Cogranne and Florent Retraint. Statistical detection of defects in radiographic images using an adaptive parametric model. *Signal Processing*, 96:173–189, 2014.

[12] N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. *Conference on Computer Vision and Pattern Recognition (CVPR2005)*, 1:886–893, 2005.

[13] Navneet Dalal and Bill Triggs. Histograms of oriented gradients for human detection. In *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, volume 1, pages 886–893. IEEE, 2005.

[14] O. Faugeras. *Three-Dimensional Computer Vision: A Geometric Viewpoint*. The MIT Press, Cambridge MA, London, 1993.

[15] D. Filbert, R. Klatte, W. Heinrich, and M. Purschke. Computer aided inspection of castings. In *IEEE-IAS Annual Meeting*, pages 1087–1095, Atlanta, USA, 1987.

[16] A. Fitzgibbon, M. Pilu, and R.B. Fisher. Direct least square fitting ellipses. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 21(5):476–480, 1999.

[17] J. Flusser and T. Suk. Pattern recognition by affine moment invariants. *Pattern Recognition*, 26(1):167–174, 1993.

[18] R.C. Gonzalez and R.E. Woods. *Digital Image Processing*. Pearson, Prentice Hall, third edition, 2008.

[19] L. Gupta and Mandyam D. Srinath. Contour sequence moments for the classification of closed planar shapes. *Pattern Recognition*, 20(3):267–272, 1987.

[20] Randolf Hanke, Theobald Fuchs, and Norman Uhlmann. X-ray based methods for non-destructive testing and material characterization. *Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment*, 591(1):14–18, 2008.

[21] R.M. Haralick, K. Shanmugam, and I. Dinstein. Textural features for image classification. *IEEE Trans. on Systems, Man, and Cybernetics*, SMC-3(6):610–621, 1973.

[22] R.M. Haralick and L.G. Shapiro. *Computer and robot vision*. Addison-Wesley Publishing Co., New York, 1992.

[23] R. I. Hartley and A. Zisserman. *Multiple view geometry in computer vision*. Cambridge University Press, 2000.

[24] R. I. Hartley and A. Zisserman. *Multiple view geometry in computer vision*. Cambridge University Press, second edition, 2003.

[25] H. Hecker. *Ein neues Verfahren zur robusten Röntgenbildauswertung in der automatischen Gußteilprüfung*. PhD thesis, vom Fachbereich Elektrotechnik, Technische Universität Berlin, 1995.

[26] W. Heinrich. *Automated inspection of castings using X-ray testing*. PhD thesis, Institute for Measurement and Automation, Faculty of Electrical Engineering, Technical University of Berlin, 1988. (in German).

[27] W. Heinrich. *Automatische Röntgenserienprüfung von Gußteilen*. PhD thesis, Institut für Allgemeine Elektrotechnik, Technische Universität Berlin, 1988.

[28] M.-K. Hu. Visual pattern recognition by moment invariants. *IRE Trans. Info. Theory*, IT(8):179–187, 1962.

[29] B. Jähne. *Digitale Bildverarbeitung*. Springer, Berlin, Heidelberg, 2 edition, 1995.

[30] K.-F Kamm. Grundlagen der Röntgenabbildung. In K. Ewen, editor, *Moderne Bildgebung: Physik, Gerätetechnik, Bildbearbeitung und -kommunikation, Strahlenschutz, Qualitätskontrolle*, pages 45–62, Stuttgart, New York, 1998. Georg Thieme Verlag.

[31] Juho Kannala and Esa Rahtu. BSIF: Binarized statistical image features. In *Pattern Recognition (ICPR), 2012 21st International Conference on*, pages 1363–1366. IEEE, 2012.

[32] Reinhard Klette. *Concise computer vision: an introduction into theory and algorithms*. Springer Science & Business Media, 2014.

[33] J.-M. Kosanetzky and H. Putzbach. Modern X-ray inspection in the automotive industry. In *Proceedings of the 14th World*

Conference on Non-Destructive Testing (WCNDT–1996), New Delhi, Dec. 8-13 1996.

[34] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. ImageNet classification with deep convolutional neural networks. *NIPS*, pages 1106–1114, 2012.

[35] A. Kumar and G.K.H. Pang. Defect detection in textured materials using gabor filters. *IEEE Trans. on Industry Applications*, 38(2):425–440, 2002.

[36] Y LeCun, L Bottou, and Y Bengio. Gradient-based learning applied to document recognition. In *Proceedings of the Third International Conference on Research in Air Transportation*, 1998.

[37] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. *Nature*, 521(7553):436–444, 2015.

[38] Stefan Leutenegger, Margarita Chli, and Roland Yves Siegwart. BRISK: Binary robust invariant scalable keypoints. In *Computer Vision (ICCV), 2011 IEEE International Conference on*, pages 2548–2555. IEEE, 2011.

[39] Wei Li, Kangshun Li, Ying Huang, and Xiaoyang Deng. A new trend peak algorithm with x-ray image for wheel hubs detection and recognition. In *Computational Intelligence and Intelligent Systems*, pages 23–31. Springer, 2015.

[40] Xiaoli Li, S. K. Tso, , Xin-Ping Guan, and Qian Huang. Improving automatic detection of defects in castings by applying wavelet technique. *IEEE Transactions on Industrial Electronics*, 53(6):1927–1934, 2006.

[41] D. Lowe. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, 60(2):91–110, 2004.

[42] J. Matas, O. Chum, M. Urban, and T. Pajdla. Robust wide-baseline stereo from maximally stable extremal regions. *Image and Vision Computing*, 22(10):761–767, 2004.

[43] MathWorks. *Image Processing Toolbox for Use with MATLAB: User's Guide*. The MathWorks Inc., 2014.

[44] D. Mery. Flaw simulation in castings inspection by radioscopy. *Insight*, 43(10):664–668, 2001.

[45] D. Mery. Crossing line profile: a new approach to detecting defects in aluminium castings. *Proceedings of the Scandinavian Conference on Image Analysis (SCIA 2003), Lecture Notes in Computer Science*, 2749:725–732, 2003.

[46] D. Mery. Explicit geometric model of a radioscopic imaging system. *NDT & E International*, 36(8):587–599, 2003.

[47] D. Mery. Automated radioscopic testing of aluminum die castings. *Materials Evaluation*, 64(2):135–143, 2006.

[48] D. Mery. Automated detection in complex objects using a tracking algorithm in multiple X-ray views. In *Proceedings of the 8th IEEE Workshop on Object Tracking and Classification Beyond the Visible Spectrum (OTCBVS 2011), in Conjunction with CVPR 2011, Colorado Springs*, pages 41–48, 2011.

[49] D Mery. *Computer Vision for X-Ray Testing*. Springer, 2015.

[50] D. Mery. Inspection of complex objects using multiple-X-ray views. *IEEE/ASME Transactions on Mechatronics*, 20(1):338–347, Feb 2015.

[51] D. Mery and D. Filbert. A fast non-iterative algorithm for the removal of blur caused by uniform linear motion in X-ray images. In *Proceedings of the 15th World Conference on Non-Destructive Testing (WCNDT–2000)*, Rome, Oct. 15-21 2000.

[52] D. Mery and D. Filbert. Automated flaw detection in aluminum castings based on the tracking of potential defects in a radioscopic image sequence. *IEEE Trans. Robotics and Automation*, 18(6):890–901, December 2002.

[53] D. Mery and D. Filbert. Classification of potential defects in automated inspection of aluminium castings using statistical pattern recognition. In *8^{th} European Conference on Non-Destructive Testing (ECNDT 2002)*, pages 1–10, Barcelona, 17-21 June 2002.

[54] D. Mery, D. Filbert, and Th. Jaeger. Image processing for fault detection in aluminum castings. In D.S. MacKenzie and G. Totten, editors, *Analytical Characterization of Aluminum and Its Alloys*, New York, 2003. Marcel Dekker.

[55] D. Mery, D. Filbert, and N. Parspour. Improvement in automated aluminum casting inspection by finding correspondence of potential flaws in multiple radioscopic images. In *Proceedings of the 15th World Conference on Non-Destructive Testing (WCNDT–2000)*, Rome, Oct. 15-21 2000.

[56] D. Mery, Th. Jaeger, and D. Filbert. A review of methods for automated recognition of casting defects. *Insight*, 44(7):428–436, 2002.

[57] D. Mery and F. Pedreschi. Segmentation of colour food images using a robust algorithm. *Journal of Food Engineering*, 66(3):353–360, 2004.

[58] Domingo Mery and Miguel Angel Berti. Automatic detection of welding defects using texture features. *Insight-Non-Destructive Testing and Condition Monitoring*, 45(10):676–681, 2003.

[59] Domingo Mery, Vladimir Riffo, Uwe Zscherpel, German Mondragón, Iván Lillo, Irene Zuccar, Hans Lobel, and Miguel Carrasco. Gdxray: The database of x-ray images for nondestructive testing. *Journal of Nondestructive Evaluation*, 34(4):1–12, 2015.

[60] Krystian Mikolajczyk and Cordelia Schmid. A performance evaluation of local descriptors. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27(10):1615–1630, 2005.

[61] Y. Mu, S. Yan, Y. Liu, T. Huang, and B. Zhou. Discriminative local binary patterns for human detection in personal album. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR 2008)*, pages 1–8, 2008.

[62] Vinod Nair and Geoffrey E Hinton. Rectified linear units improve restricted boltzmann machines. In *Proceedings of the 27th International Conference on Machine Learning (ICML-10)*, pages 807–814, 2010.

[63] A. Noble, R. Gupta, J. Mundy, A. Schmitz, and R. Hartley. High precision X-ray stereo for automated 3D CAD-based inspection. *IEEE Trans. Robotics and Automation*, 14(2):292–302, 1998.

[64] Timo Ojala, Matti Pietikainen, and Topi Maenpaa. Multiresolution gray-scale and rotation invariant texture classification with local binary patterns. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(7):971–987, 2002.

[65] Ville Ojansivu and Janne Heikkilä. Blur insensitive texture classification using local phase quantization. In *Image and signal processing*, pages 236–243. Springer, 2008.

[66] E. Persoon and K.S. Fu. Shape discrimination using Fourier descriptors. *IEEE Trans. Systems, Man, and Cybernetics*, SMC-7(3):170–179, 1977.

[67] C Pieringer and D Mery. Flaw detection in aluminium die castings using simultaneous combination of multiple views. *Insight*, 52(10):548–552, 2010.

[68] Luis Pizarro, Domingo Mery, Rafael Delpiano, and Miguel Carrasco. Robust automated multiple view inspection. *Pattern Analysis and Applications*, 11(1):21–32, 2008.

[69] F Ramírez and H Allende. Detection of flaws in aluminium castings: a comparative study between generative and discriminant approaches. *Insight-Non-Destructive Testing and Condition Monitoring*, 55(7):366–371, 2013.

[70] David E Rumelhart, Geoffrey E Hinton, and Ronald J Williams. Learning representations by back-propagating errors. *Cognitive modeling*, 5(3):1, 1998.

[71] Karen Simonyan and Andrew Zisserman. Very Deep Convolutional Networks for Large-Scale Image Recognition. *CoRR abs/1409.1556*, 2014.

[72] M. Sonka, V. Hlavac, and R. Boyle. *Image Processing, Analysis, and Machine Vision*. PWS Publishing, Pacific Grove, CA, 2 edition, 1998.

[73] Yinggan Tang, Xiumei Zhang, Xiaoli Li, and Xinping Guan. Application of a new image segmentation method to detection of defects in castings. *The International Journal of Advanced Manufacturing Technology*, 43(5-6):431–439, 2009.

[74] C.H. Teh and R.T. Chin. On digital approximation of moment invariants. *Computer Vision, Graphics and Image Processing*, 33(3):318–326, 1986.

[75] I. Tosic and P. Frossard. Dictionary learning. *Signal Processing Magazine, IEEE*, 28(2):27–38, 2011.

[76] B. Triggs, P. McLauchlan, R. Hartley, and A. Fitzgibbon. Bundle adjustment: a modern synthesis. *Vision algorithms: theory and practice*, pages 153–177, 2000.

[77] P. Viola and M. Jones. Robust real-time object detection. *International Journal of Computer Vision*, 57(2):137–154, 2004.

[78] John Wright, Allen Y Yang, Arvind Ganesh, Shankar S Sastry, and Yi Ma. Robust face recognition via sparse representation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 31(2):210–227, 2009.

[79] C.T. Zahn and R.Z. Roskies. Fourier descriptors for plane closed curves. *IEEE Trans. Computers*, C-21(3):269–281, 1971.

[80] Xinyue Zhao, Zaixing He, and Shuyou Zhang. Defect detection of castings in radiography images using a robust statistical feature. *JOSA A*, 31(1):196–205, 2014.

[81] Xinyue Zhao, Zaixing He, Shuyou Zhang, and Dong Liang. A sparse-representation-based robust inspection system for hidden defects classification in casting components. *Neurocomputing*, 153(0):1 – 10, 2015.