sensors

MDPI

Article

Segmentation of Drilled Holes in Texture Wooden Furniture Panels Using Deep Neural Network

Rytis Augustauskas [1,*], Arūnas Lipnickas [1] and Tadas Surgailis [2]

[1] Department of Automation, Kaunas University of Technology, 51367 Kaunas, Lithuania; arunas.lipnickas@ktu.lt
[2] MB Prorega, 48212 Kaunas, Lithuania; tadas@prorega.lt
* Correspondence: rytis.augustauskas@ktu.lt; Tel.: +370-61916583

Abstract: Drilling operations are an essential part of furniture from MDF laminated boards required for product assembly. Faults in the process might introduce adverse effects to the furniture. Inspection of the drilling quality can be challenging due to a big variety of board surface textures, dust, or woodchips in the manufacturing process, milling cutouts, and other kinds of defects. Intelligent computer vision methods can be engaged for global contextual analysis with local information attention for automated object detection and segmentation. In this paper, we propose blind and through drilled holes segmentation on textured wooden furniture panel images using the UNet encoder-decoder modifications enhanced with residual connections, atrous spatial pyramid pooling, squeeze and excitation module, and *CoordConv* layers for better segmentation performance. We show that even a lightweight architecture is capable to perform on a range of complex textures and is able to distinguish the holes drilling operations' semantical information from the rest of the furniture board and conveyor context. The proposed model configurations yield better results in more complex cases with a not significant or small bump in processing time. Experimental results demonstrate that our best-proposed solution achieves a Dice score of up to 97.89% compared to the baseline U-Net model's Dice score of 94.50%. Statistical, visual, and computational properties of each convolutional neural network architecture are addressed.

Keywords: CNN (convolutional neural networks); deep learning; image processing; hole detection; drilling; furniture manufacturing; quality inspection; industry 4.0

Citation: Augustauskas, R.; Lipnickas, A.; Surgailis, T. Segmentation of Drilled Holes in Texture Wooden Furniture Panels Using Deep Neural Network. *Sensors* **2021**, *21*, 3633. https://doi.org/10.3390/s21113633

Academic Editors: Janos Abonyi and Chen Chen

Received: 9 April 2021
Accepted: 21 May 2021
Published: 23 May 2021

1. Introduction

Furniture manufacturing of laminated MDF (Medium-density fiberboard) panels is a multistage process that consists of many manual or automated steps. It starts with the production of the chipboard and its lamination. When the designed shape furniture panel is cut out, the milling and drilling process starts, which is the most crucial in furniture manufacturing. The arrangement of drilled holes is critical for successful final product assembly. Deviation from template corrupts the final product. The faults might appear due to various reasons: failures or misalignments in drilling machinery template set-up, wear-off or lose parts, dull or broken drill, and others. Moreover, the manual inspection itself requires a lot of time due to measurement evaluation and knowledge about the individual part template. The situation can get even more complicated due to different sizes of drillings, multiple holes (blind and through), different parts, scobs, and dust, and other defects. Therefore, this process needs to be automated.

Nowadays inspection of manufacturing processes is an essential part of industry 4.0 standards. Investigating the quality in each step of production might lead to detecting the flaws in early fabrication stages and reducing materials usage and operations time needed. In the end, manufacturing costs can be cut down. Besides the obvious results, the risk of defects appeared at sold production can be decreased as well. A non-invasive check-up, such as one that is computer vision (CV)-based, might be used in the most observable

cases. As it is mentioned in the review [1], the visual-based approach for defect detection is one of the most common in the industry. It is complicated to take into consideration a variety of furniture board processing operations, possible defects, and complicated surface pattern cases. Moreover, the production quality evaluation factors can be disturbed by aggressive manufacturing conditions. However, from visual information, a broader context might be perceived. As mentioned before, the irrelevant parts, such as defects or milling, might appear along with drilled holes. Therefore, the algorithm should distinguish only the information that is pertinent for the task. Computer-vision-based methods need to act as an intelligent sensor for drilling localization.

In this paper, we present a novel, data-driven approach for contextual pixel-level drilled hole segmentation approach in textured wooden furniture panels from the images. We use a small architecture U-Net convolutional encoder-decoder network as a baseline and we are proposing the architectural modifications in a neural network with residual connections [2], atrous spatial pyramid pooling module [3], squeeze and excitation blocks [4], and *CoordConv* layers [5] that improves the standard architectures in Dice score for a pixel-level segmentation task with a slight computational performance increase. Besides the modification in models, we address segmentation precision and computational performance. We compare convolutional neural network results with conventional image processing methods to show the advantage of higher-level information representation and the ability to adapt to the context in a wide spectrum of cases. Our neural network implementation, conventional image processing comparison methods code, and more rendered results can be found in the GitHub repository [6].

2. Related Work

There can already be found computer-vision-aided approaches for through-hole inspection. In researches conducted by Hernandez et al. [7] and Caggiano et al. [8], the carbon fiber boards are being investigated. In papers, authors proposing segmentation based on Otsu threshold [9] and segmentation from HSV colormap respectively. Drilled hole contours can be separated, and the color/texture of boards is always constant. Another drilling inspection approach was described by Yu et al. [10]. Researchers have used multiple image preprocessing techniques and Canny edge [11] to extract holes and a flush for rivets in aircraft panels. More complicated hole segmentation in textured composites parts is presented in [12]. Authors were utilizing local binary patterns algorithm [13] in combination with deep learning segmentation with a lightweight U-Net convolutional neural network (CNN). Overall, the practical implementation of the proposed research on the drilled holes segmentation would be very limited. Most of the review articles are utilizing classical computer vision methods, such as thresholding (Otsu or from HSV colormap) or edge detection (Canny). Only one [12] of mentioned articles employs a deep neural network for more complicated hole image data.

Image processing algorithms can be a satisfying solution in a defined number of cases, however more dynamic inspection conditions or complex manufacturing processes or production require more advanced and higher-capability solutions. Representing a problem by strictly formed rules might be a narrow solution or it can get complicated to cover up states or situations in an extensive dataset. However, this problem can be overcome by utilizing data-driven solutions, such as deep learning (DL) approaches. Labeled samples can provide essential information for the chosen algorithm on how to cope with a particular task. Expert data knowledge and representation can be transferred to the model during its training process. Even extra-large-scale datasets, for example ImageNet classification [14], Microsoft COCO [15], and Open Images Dataset [16], are proven to be solved by engaging deep learning methods [17–19].

There can be found multiple application of artificial-intelligence-aided computer vision in a variety of automated manufacturing inspection cases, such as steel [20–22], wood [23–26], and resin/plastic [27–29]. The mentioned investigations utilize deep neural networks as an algorithm to distinguish defects. Taking into consideration segmentation

of drilled holes in furniture panels problem complexity, data diversity and industrial environment working conditions there can be made relations with approaches used for defects detection and inspection methods used in manufacturing and severe environments.

In most cases, convolutional neural networks (CNN) are employed for image processing. Popular architecture-based solutions can be found in several papers. Gao et al. [23] proposes ResNet-34 [2] for wood knot detection and classification. Moreover, the authors are proposing transfer learning (using a pre-trained model backbone) to overcome a limited number of samples in the used dataset. Yin et al. [30] describe sewer pipes defects detection techniques from CCTV footage using YOLOv3 [31] object detection model. Another single-shot technique for printed circuit board abnormalities search is described by Adibhatla et al. [32]. Authors of solar panels manufacturing defects detection investigation [33] use Fast RCNN [34] with VGG-16 [35] backbone for defective regions search. Researchers also propose the Complementary attention network module for features extracted (from the backbone) feature refinement. Roberts et al. [36] applied U-Net [37] encoder-decoder with additional dense connection for crystallographic defects in steel images semantic segmentation. Additionally, from the following papers, it can be seen that even small architectures can perform well in defect identification tasks. In [21], researchers uses MobileNetV2 [38] backbone for welding classification. Modified versions with dense connections of mentioned architecture are proposed in [39] for DAGM [40] defective patterns classification. As authors suggest, introduced adjustment allows coping better with the multiscale problem. Even smaller convolutional design networks are described in steel wire defects detection investigation [22]. The solution utilizes three convolutional layers of neural network for a 3-class classification task. A similar approach (regarding small neural network architecture) can be found in wood defects detection and classification investigation [24]. A minimalistic convolutional neural network can be seen in resin defection research [27], where the LeNet-5-like model is being utilized. Moreover, light-weight segmentation approaches are investigated by Huang et al. [41], where only one step of upscaling is employed and another enhancement, atrous spatial pyramid pooling (ASPP) [3], is utilized. The proposed architecture solution has shown good results on DAGM [40], Wood defects [42], and NEU [43] datasets. It can be summed up that deep-learning-based computer vision can give a solution in complicated situations.

3. Methods

3.1. Convention Image Processing

There are multiple computer vision algorithms for feature extraction from visual information. Most of the statistical methods rely on local intensity differences in the data without contextual analysis. A classical method such as intensity threshold is more suitable for static data, which does not alter that much. However, "real-world" scenarios usually are not fixed in a particular way. Especially manufacturing environments tend to be more diverse in conditions and production visual complexity might vary. For the mentioned case there can be found more robust methods that are more adaptive to dynamic. For example, Sobel [44] or Laplace [45] filters signify changes in visual information intensities. Kernels of Sobel filter along x and y axes are given in Equations (1) and (2) and the kernel of the Laplace filter is given in Equation (3).

$$G_x = \begin{bmatrix} 1 & 0 & -1 \\ 2 & 0 & -2 \\ 1 & 0 & -1 \end{bmatrix}, \tag{1}$$

$$G_y = \begin{bmatrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix}, \tag{2}$$

where G_x and G_y are Sobel filters kernel along x- and y-axes, respectively.

$$D^2_{xy} = \begin{bmatrix} 0 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & 0 \end{bmatrix}, \tag{3}$$

where D^2_{xy} is the Laplace filter kernel.

Another popular and powerful method for edge segmentation is Canny edge detector [11]. Firstly, the algorithm utilizes Gaussian filter to reduce the noise in the image, after, abrupt intensity changes (possible edges) are extracted using Sobel filter along x and y axes. Subsequently, non-maximum suppression is applied to remove spurious edges and thresholding engaged to remove weak results. After that, edges are processed with hysteresis and small artifacts that are not connected to "strong" edges are removed. However, even with these methods, it can be hard to capture specific details when the context is sophisticated: a variety of possible color combinations, object surface with patterns, similarities between a significant (desired to extract), and a minor (background) information.

3.2. Baseline U-Net

A more advanced segmentation approach—convolutional encoder-decoder (U-Net)—might be engaged. A data-driven model can represent features while taking into consideration not only the simple local intensity differences but also the relations between details and other semantical information. The knowledge presented in labels can be encoded into a high dimensional feature space and generalized making the U-Net a powerful tool for information extraction (segmentation) in a complicated context. In this work, as a baseline segmentation model, we employ a lightweight U-Net [37] convolutional neural network (Figure 1). The architecture consists of two main parts: encoder and decoder. The first extracts image features, and the second reconstructs the segmentation map. Opposite layers in the encoder and decoder are associated with skipped connections that allow transferring higher-level features from larger dimension layers. In this research, we utilized quite a small architectural design with three downscales. At the first stage (first layer) 16 feature maps are employed. After each width and height downscale by two, the number of feature maps is doubled. In the decoder, reversed operations are performed—dimensions upscale and feature maps count reduction by two. The output of convolutional encoder–decoder is 1×1 convolution with sigmoid activation which performs as binary classifier between two classes: drilled hole and background.

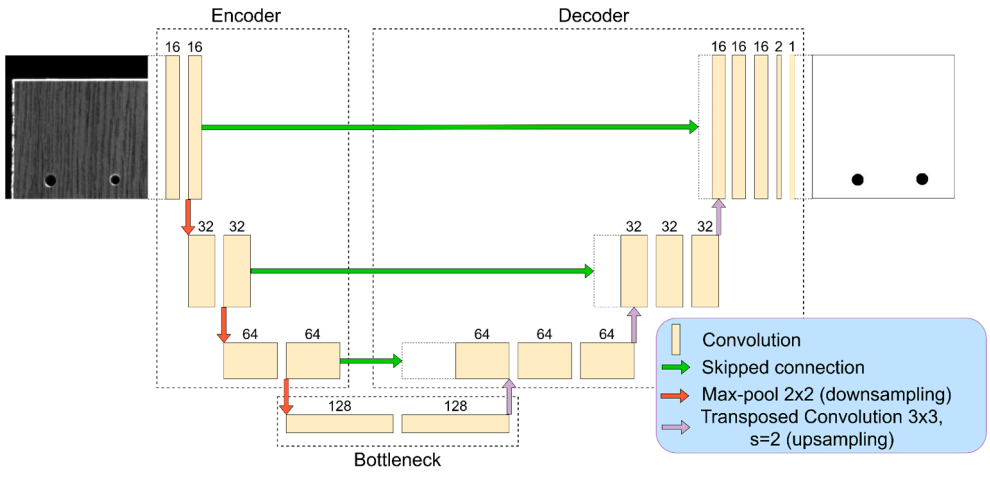


Figure 1. Baseline U-Net principal structure. The input is 320 × 320 px greyscale image and the output 320 × 320 px classified image (black is drilling segmentation, white is the background).

A more detailed illustration of layers structure on opposite sides (encoder and decoder) is given in Figure 2. Each stage in the encoder consists of two convolutional operations with 3×3 size kernels, with a stride of 1 pixel. Only kernel size exception is applied in the first layer, where 5×5 is engaged. In decoder transposed convolution with 3×3 kernel and stride of 2 pixels. It increases the input dimensions by two. Further, it is a "learnable" approach for enlarged pixels interpolation. After upscale, feature maps from the previous layer are concatenated with opposite feature maps from the encoder (skipped connection). Every convolution and transposed convolution operation is followed by batch normalization [46]. It has trainable mean and variance parameters that help to keep output from convolution operation normalized. Moreover, it stabilizes the neural network model and increases training speed. As activation function, parametrized rectified linear unit or Leaky ReLU is operated. It is shown in the following equation:

$$f(x) = \begin{cases} x, \; if \; x > 0 \\ 0.1x, \; otherwise \end{cases}, \tag{4}$$

where x is activation function input.

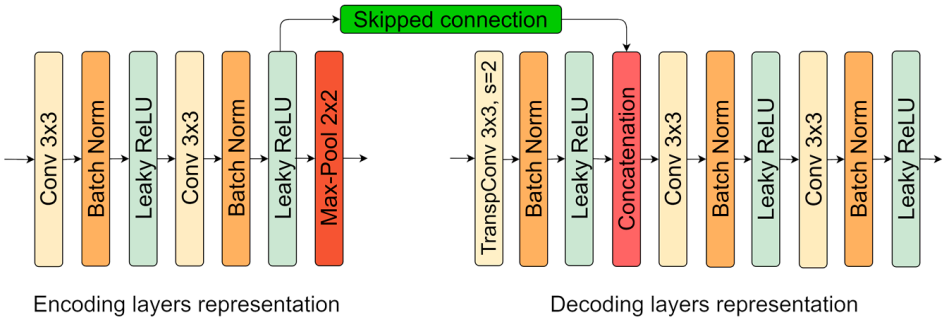


Figure 2. U-Net encoding and decoding layers representations.

Additionally, we investigate the modified versions of U-Net. While an increased number of feature kernels in convolutional operation might end up in better segmentation results, it also loads a model with significantly more computational operations and prolongs execution time. We propose tricks and lightweight enhancements to improve segmentation efficiency while the impact on computational performance is not significant. Architectural changes are described more briefly in the following subsections.

3.3. Residual Connections

Residual layers are proposed in ResNet [2]. The branch connected in parallel skips convolutional operation. Residual connections help to maintain information flow through the whole network, without a possible degradation in series of operations conducted in a neural network. Moreover, this block increases model accuracy and might cope with the vanishing gradient problem. Residual layers are used in popular architectures, such as SqueezeNext [47], DeepLab [48], and Inception [49]. The implementation used in this research is shown in Figure 3. We utilize 1×1 convolution to make the number of feature maps the same before the addition operation.

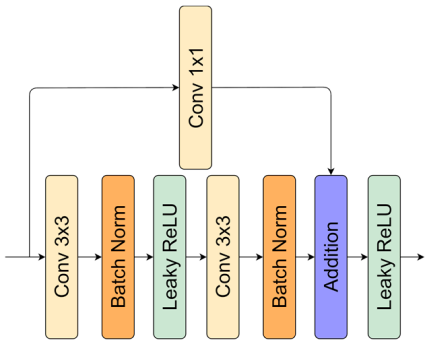


Figure 3. Residual layer representation.

3.4. Squeeze-and-Excitation

A light-weight solution proposed by Hu et al. [4] adaptively adjusts individual feature map weight. Squeeze and excitation (SE) block average each feature map to trainable fully connected neuron layers (Figure 4). After the second layer, sigmoid activation is applied that outputs values in the range [0.0,1.0]. Each value is a scalar for each feature map matrix. They recalibrate the significance channel-wise, taking into consideration dependencies between feature maps. In the mentioned research [4], squeeze and excitation enhanced convolutional neural network shown image classification accuracy boost on ImageNet [14], while not adding a lot computations to model (ResNet-50–top-1 error 24.8% (3.86 GFLOPs), ResNet-50-SE–top-1 error 23.29% (3.87 GFLOPs)).

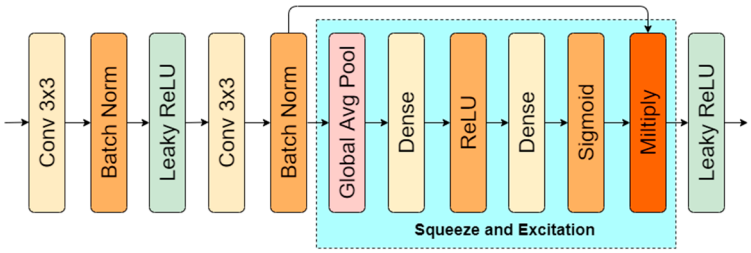


Figure 4. Squeeze and excitation block representation.

3.5. Atrous Spatial Pyramid Pooling

Convolutional operations with different dilation rates might extract multi-scale contextual information better than regular convolutions (with a dilated rate equal to 1). Atrous or dilated convolutions in the parallel idea was proposed by Chen et al. [3]. An expanded convolutional kernel can better respond to different resolution features. In our research, we used three parallel branches with three different dilation rates: 1, 2, and 4 (Figure 5). However, some papers utilize bigger rates. Even in the previously mentioned research, the authors used 6, 12, and 18 dilation rates in convolutional kernels. In another research [50], the authors conducted multiple experiments with various rates, which yielded different results. Our dilation rates were chosen with the motivation of not severe changes in the data view scale. Additionally, we added another branch in parallel with the average pooling of individual feature maps and upscaling to capture global information in the particular feature channel. This idea is inspired by ParseNet [51] approach.

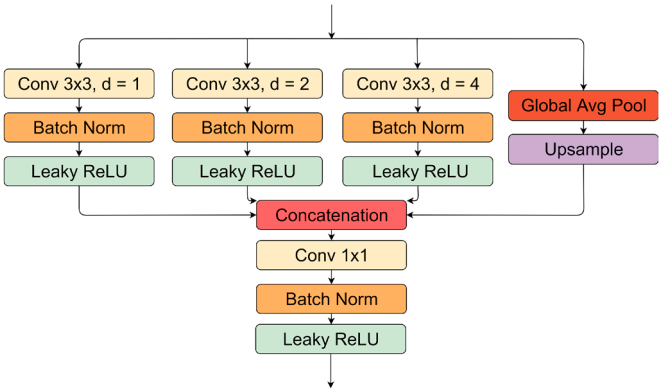


Figure 5. Atrous spatial pyramid pooling module.

3.6. CoordConv

An interesting approach by encoding position coordinates to cope with the data transition invariance problem was proposed by Liu et al. [5]. The authors suggested an idea to boost the prediction performance by introducing additional information in feature maps. *CoordConv* practices in convolutional neural networks have shown improvements in prediction [52–54]. For two-dimensional information, the authors propose two additional channels with a row index along the *y*-axis and a column index along the *x*-axis (Figure 6a). In this research, *CoordConv* operation is joined with other feature maps before convolutional operations (Figure 6b).

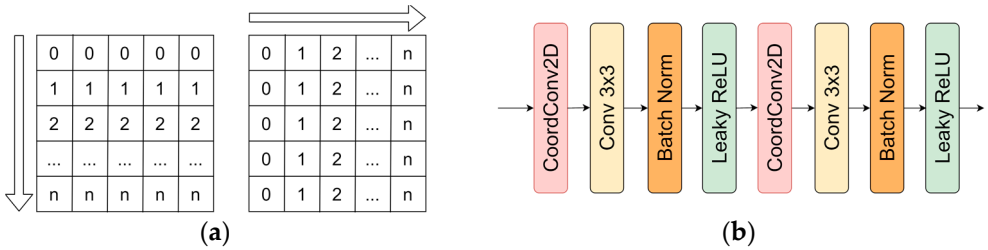


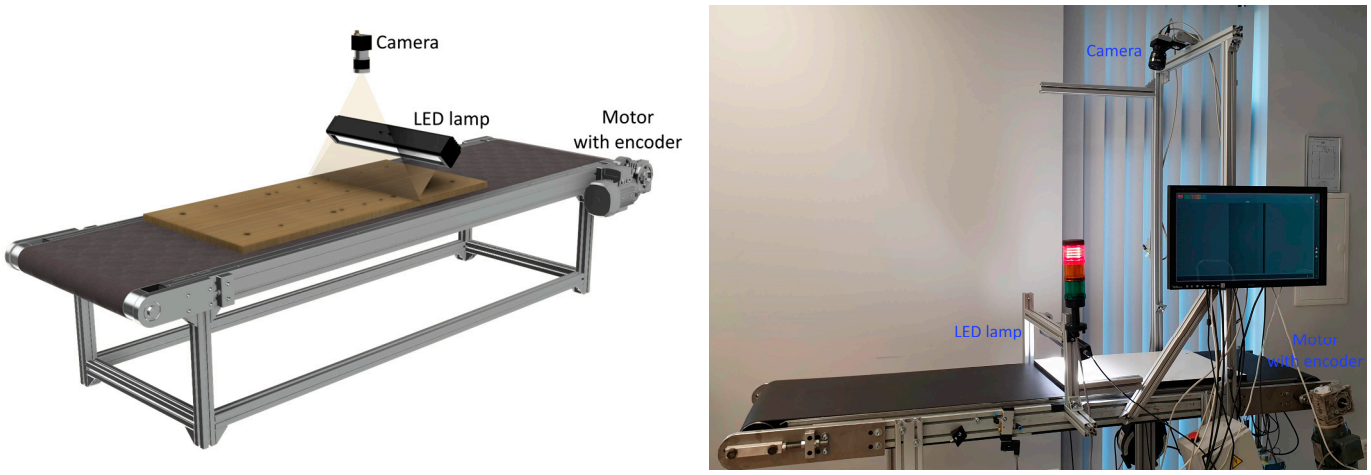
Figure 6. *CoordConv2D* representation: (**a**) 2 feature maps along *x*- and *y*-axis, indicating the position of the pixel at each axis; (**b**) *CoordConv2D* implementation before convolutional operation (adds 2 features maps—y-axis rows and x-axis columns indices).

4. Data

4.1. Image Capture Setup

Wooden furniture panels are scanned with a linear camera from an industrial conveyor. The image data acquisition stand (laboratory) is shown in Figure 7. The main parts of the visual inspection setup are a linear monochromatic camera with a scan width of up to 6144 pixels, an industrial LED light source, and a conveyor. The camera is attached 1.1 m above the conveyor belt. Its capture area (line) collides with an industrial LED light directional normal at the same line. Only the area around the camera scanning line is illuminated at a particular moment. Furniture panels are moved by a conveyor belt driven by the electrical motor. This motor is equipped with an encoder that triggers a scan of the linear camera. The start of capturing is invoked by a separate laser sensor that gives a high output signal when the furniture panel approaches the scanning area. The image capturing continues until the laser sensor signal is high or until the set image height is scanned. The mentioned links result in the system synchronization-camera scanning is triggered according to conveyor rotation (start and continues line scan). The equipment used for data grabbing is given in Table 1. The image capture setup is separate from the whole production line. Before furniture panels reach the visual inspection conveyor, they are

directed by correcting the alignment. The object on the conveyor is always perpendicular to the scanning line. The physical orientation error does not exceed the 2° angle.



(**a**) (**b**)

Figure 7. The image capturing set: (**a**) principle scheme, (**b**) laboratory setup. Linear camera pointing into the same line (area) as an industrial LED light source. The camera is triggered by an encoder on the motor.

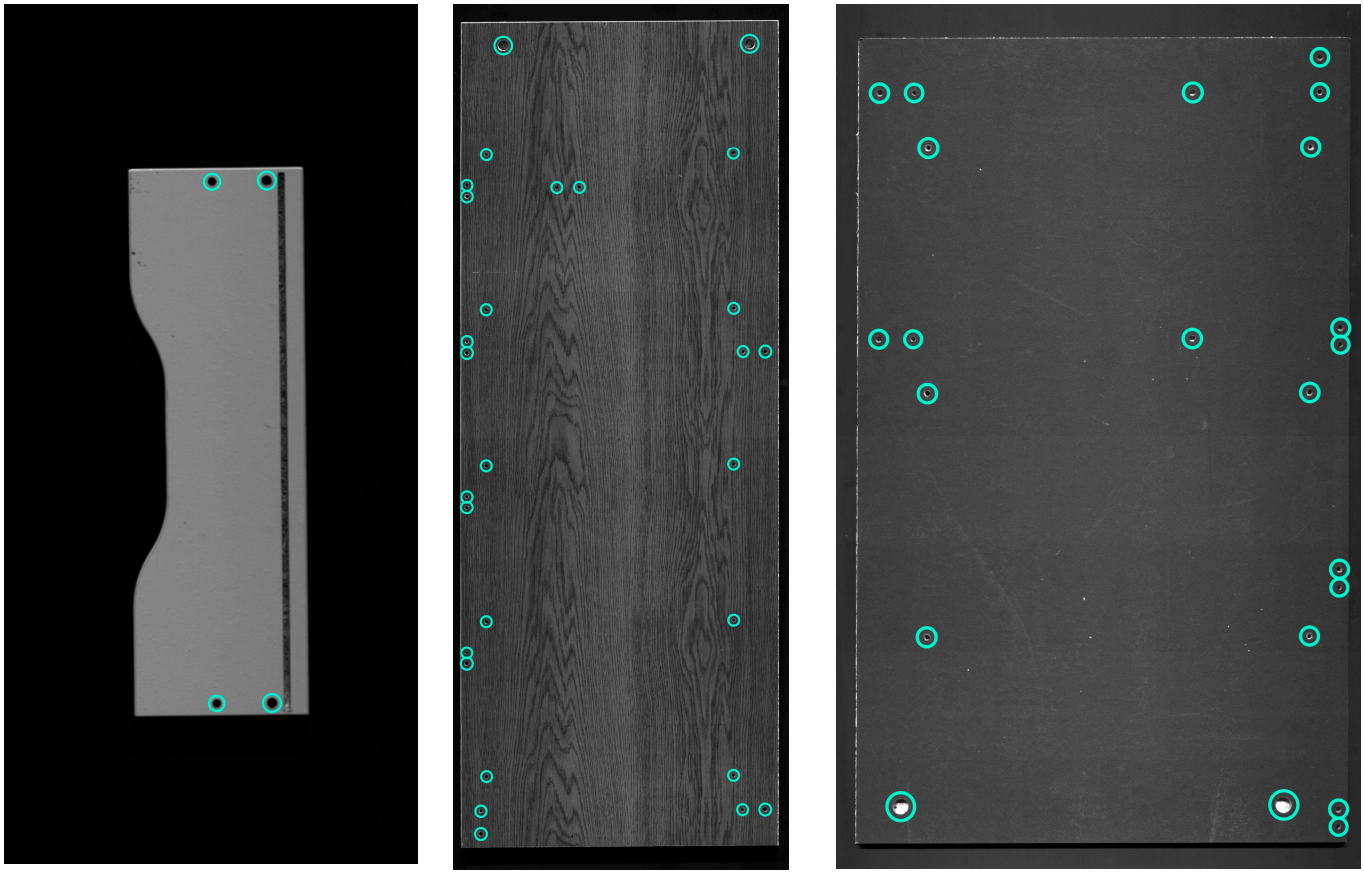
Table 1. Capturing set components.

Component	Model
Linear camera	raL6144-16 gm-Basler racer [55]
Camera optics	NIKON AF Nikkor 24 mm f/2.8D [56]
Encoder (on the motor)	Autonics E40S6-1500-3-T-24 [57]
Industrial LED lamp	EBAR-1125-WHI-7 TPL-Vision [58]

4.2. Wooden Furniture Panels Image Data

The size of furniture panels varies significantly. Depending on the manufactured product, the dimensions of the part can be as small as 0.13 m × 0.4 m (front panel of a table drawer) and as big as 0.9 m × 2.0 m (side of a cupboard) (Figure 8a–c). While there is a big diversity between panels size, there is no need to constrain the image size to be the same for all parts. Smaller furniture panels do not occupy the whole scanning area and it is pointless to analyze the rest of the conveyor context (outside the furniture part boundaries). As the information is not relevant for the analysis, the scanning range in width (as well as height) is adjusted. The image dimensions used in this research vary in width from 1000 to 6144 and height from 900 to 12,384 pixels. The biggest (consisting of two joined frames) image is 6144 × 12,384 pixels.

Additionally, to the furniture panels' dimension and image size variety, there are big alterations in production exterior texture and colors. A few samples can be seen in Figure 8a—white, Figure 8b—wood pattern imitation, Figure 8c—black. For better details extraction and enhancement, different exposure rates are set for image capture. It ranges from 100 to 500 nanoseconds. In the case of white laminate on the furniture panel (Figure 8a), a smaller value can be applied. Hole and cutouts made by drilling or milling are easier to distinguish from the rest of a board context. Nonetheless, it gets complicated on the other samples (Figure 8b,c). In the darker color furniture panels, it is harder to extract details with a lower exposure rate. However, increasing this parameter strengthens other non-desirable details, such as the visible bottom of the drilling (light and dark wooden pattern), dust, prints on the furniture panel. In addition, manufacturing defects might appear, the drilled hole might be covered with woodchips, or surface laminate might be ripped up. Moreover, one side of the drilled hole might get more illuminated than the other (Figure 8b,c), and also cutouts might be made in particular parts.



(**a**) (**b**) (**c**)

Figure 8. Data example. (**a**–**c**) different furniture panels in production: (**a**) part with cutout, (**b**) panel with wood pattern imitation, (**c**) black panel.

Overall, there is a great change in conditions: furniture panels dimensions, appearance, visual defects, and cutouts. These factors are taken into consideration for the unified drilled hole segmentation solution.

4.3. Data Preparation

In this research, we utilized a variety of images. As is mentioned in the previous subsection, the dimensions of data samples are changing severely. Further, most of the pictures are extremely large—exceeding 72 megapixels. Moreover, only the board context is useful for possible hole drilling segmentation. Taking into consideration the hardware limitation with model resources in video memory on the graphical processing unit (GPU), the dimensions of the data sample fed to the convolutional neural network cannot be relatively large. We utilize the tiles technique when the whole image is cropped into desired size regions with overlap. In this investigation, we divided the picture into 320 pixels width and 320 pixels height regions with 80 pixels overlap. A few samples (image and label) are shown in Figure 9. In this research, we also used not positioned data (not perpendicular to camera scan line as it is mentioned in the previous subsection) because parts of training data are grabbed by placing furniture panels on the conveyor by hand while skipping the orientation adjusting step (Figure 9b).

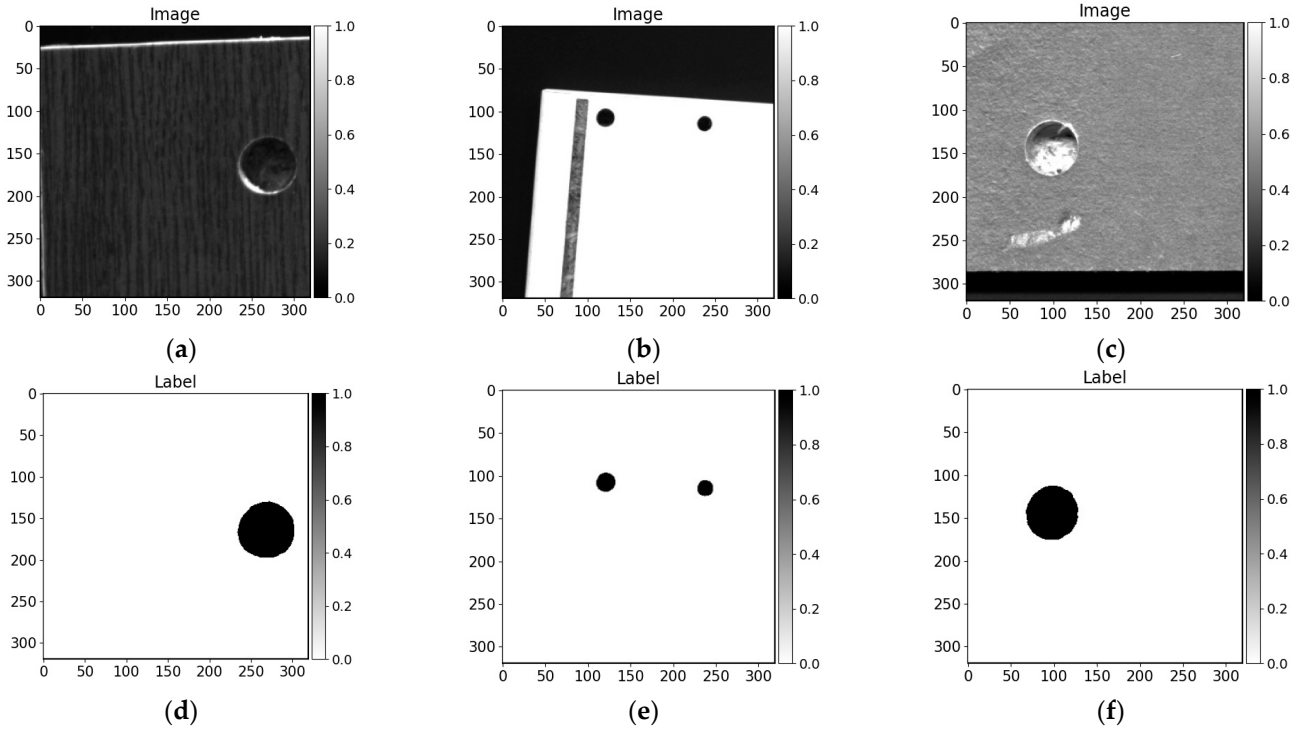


Figure 9. Prepared (cropped) data samples of 320 × 320 px size. All images are scaled to the range [0,1]. (**a**–**c**) cropped images, (**d**–**f**) drilled holed annotations.

Before image crop to tiles, we augmented the picture by rotating by 90° four times, resizing to 90% and 110% of original sample size, random brightness correction in the range [−10;10] (considering image intensity range [0;255]), random Gaussian noise and gamma correction. In this research, we used 189 images divided into 151 for training and 38 for testing. The drilled hole area is relatively small compared with the rest of the background. As the result, there might be not a lot of positive samples. We considered it randomly (with a 50% possibility) taking out region tiles that do not contain marked hole pixels. Moreover, regions with an average intensity of 5 and less there added only with a 10% possibility. These regions are conveyor belt regions, that occupied a lot of area in the picture with small furniture panel. By reducing negative samples, we increase the size of more contextually essential data—regions with drilled holes. The augmented training dataset contained 86,180 grayscale 320 × 320 pixels images and annotations. Labeled holes' pixels distribution through the image can be seen in Figure 10. Every place in the prepared data is covered at least in 0.46% of samples and the maximum covered area is in 1.09% of samples. More signified places of hole labels are given in Figure 10b,c. The most annotated regions in the augmented dataset are near corners and along vertical and horizontal center lines.

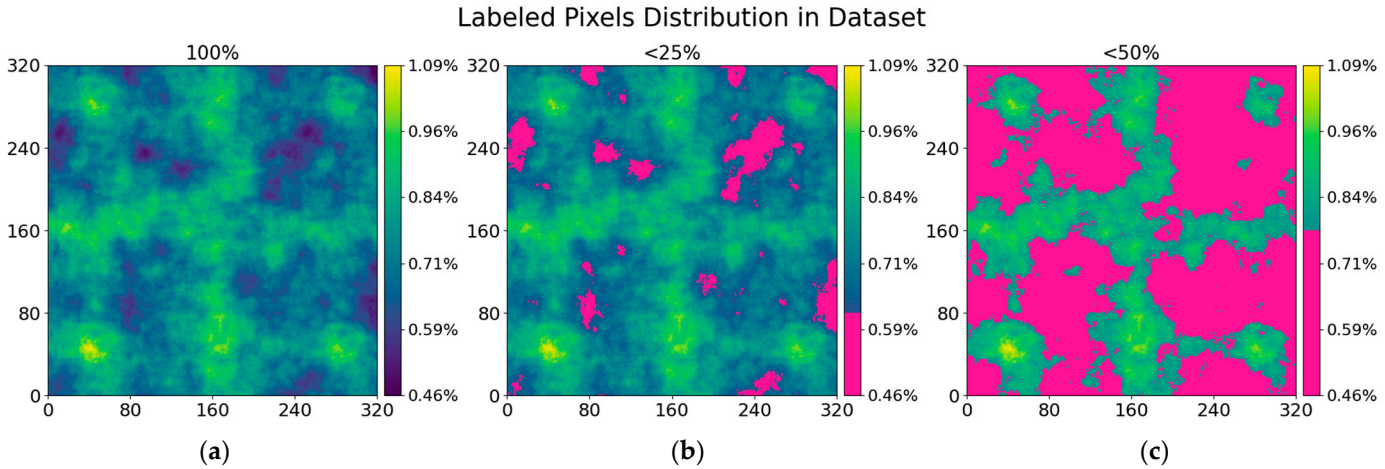


Figure 10. Individual pixels overlap by the label in the augmented dataset. (**a**) Full coverage of individual pixel covered by hole label pixel. (**b**,**c**) Areas where pixels are covered less: **b** < 25%, **c** < 50% in comparison with the maximum covered area (1.09% of data samples).

5. Experiments and Evaluation

The convolutional neural network architectures were written in Python (v3.7.9) using Keras abstraction layer [59] on Tensorflow 2.4.0 [60] backend. Experiments were conducted on desktop and laptop computers with parameters given in Table 2. Model training and testing were done in Windows 10 environment. Models trained on the desktop computer.

Table 2. Computer parameters.

Computer	CPU	RAM	GPU	OS
Desktop	AMD Ryzen 5 3600	16 GB	Nvidia 2070S	Windows 10
Laptop	Intel i5 8300H	16 GB	Nvidia 1050Ti	Windows 10

In this paper, specific modification's influence on prediction precision and computational performance are investigated. We trained and analyzed eight different convolutional encoder–decoder architectures:

- UNet;
- UNet with a squeeze and excitation (*UNet + SE*);
- UNet with CoordConv (*UNet + CoordConv*);
- UNet with a squeeze and excitation and CoordConv (*UNet + SE + CoordConv*);
- UNet with residual connections and atrous spatial pyramid pooling (*UNet + Res + ASPP*);
- UNet with residual connections, atrous spatial pyramid pooling, and squeeze and excitation (*UNet + Res + ASPP + SE*);
- UNet with residual connections, atrous spatial pyramid pooling, and CoordConv (*UNet + Res + ASPP + CoordConv*);
- UNet with residual connection, atrous spatial pyramid pooling, squeeze and excitation, and CoordConv (*UNet + Res + ASPP + SE + CoordConv*).

We chose a combined loss function consisting of cross-entropy (Equation (5)) and *Dice* loss (*Dice* score—Equation (6), and *Dice* loss—Equation (7)). The first part, cross-entropy, is quite often used loss function that describes the likelihood or probability distribution between two sets. By default, it can be found in popular machine learning frameworks. Cross-entropy loss is $\mathbf{X}$ value related to $\dot{\mathbf{X}}$ value in the following expression:

$$L_{CE} = -\frac{\sum_{i=1}^{N} x_i \cdot \log(\dot{x}_i)}{N}, \quad (5)$$

where L_{CE}—cross-entropy loss, x_i—i pixel value in label matrix $\mathbf{X}$, $\dot{x}_i$—i pixel value in neural network prediction matrix $\dot{\mathbf{X}}$, and N—a total number of pixels.

The second loss function is *Dice* [61] loss. *Dice* loss evaluates the similarity of two datasets by overlap that is measured in the range from 0.0 to 1.0. In image segmentation, *Dice* score describes the overlap of sets—label and prediction.

$$D_{score} = \frac{2 \cdot \left| \mathbf{X} \cap \dot{\mathbf{X}} \right|}{|\mathbf{X}| + \left| \dot{\mathbf{X}} \right|}, \tag{6}$$

$$L_D = 1 - D_{score}, \tag{7}$$

where D_{score}—*Dice* score, $\mathbf{X}$—label matrix, $\dot{\mathbf{X}}$—predicted matrix, L_D—*Dice* loss.

The loss function used in this research is expressed in the following Equation (8):

$$L = 0.5L_D + 0.5L_{CE}, \tag{8}$$

where L—loss function, L_D—*Dice* loss, L_{CE}—cross-entropy loss.

Each convolutional neural network architecture trained for 15 epochs, by reducing the learning rate by half every 3 epochs (scheduled reduction). Starting rate was set to be 0.001. Adam optimizer [62] was employed in the training process. The data mini-batch was set to eight samples. The whole dataset (86,180 augmented regions images) is covered by 10,770 steps/iterations in every epoch. The model is tested at the end of every epoch. The evaluation was conducted on 38 test images dividing them into 320 × 320 pixel regions (same as training data) with 160 pixels overlap. The best performing solution (according to *Dice* score) from every training has been evaluated with the *Accuracy*, *Recall*, *Precision*, *Dice* score (same Formula (12) can be expressed as in Equation (6)) and *IoU* measures:

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN}, \tag{9}$$

$$Recall = \frac{TP}{TP + FN}, \tag{10}$$

$$Precision = \frac{TP}{TP + FP}, \tag{11}$$

$$D_{score} = \frac{2 * Precision * Recall}{Precision + Recall}, \tag{12}$$

$$IoU = \frac{GroundTruth \cap Prediction}{GroundTruth \cup Prediction}, \tag{13}$$

where *TP* is true positive (correspond to correct detection of pixels that belong to the labeled defect area), *TN* is true negative (are the non-defective "background" pixels that are correctly recognized by the detector), *FP* is false positive (are wrongly detected defect pixels), *FN* is false negative (are the defect pixels that have been undetected by the detector), *GroundTruth* are labeled image pixels. The *Precision* measure indicates the proportion of false alarms; the *Recall* refers to the proportion of not detected defect pixels; and D_{score} is *Dice* score or harmonic mean of *Precision* and *Recall*.

6. Results

6.1. Conventional Image Processing Methods

In contrast to data-driven approaches using a convolutional encoder–decoder, we also compared traditional image processing methods results on drilled furniture data. We tested furniture board images with different surface patterns with Sobel filter (3 × 3 along X and Y axes), Laplace filter, and Canny edge detector. Visual results are given in Figure 11. It can be seen that filtering by local intensity tends to extract the edges

(higher difference in neighbor pixel values). Sobel filter (Figure 11a2–d2) segments the boundaries the best among all compared conventional methods although it gets harder to distinguish the changes when the surface is complicated (Figure 11b2). Further, the Sobel filter is prone to reacting to the surface patterns even when these are insignificantly changing in surface colors. Moreover, in the case where drilled hole (blind) bottom is illuminated (Figure 11a2,d2), the transitions between wood chips are signified even more. The images processed with the Laplace filter (Figure 11a4–d4) give weaker features of edges after drilling and milling. It gets hard to distinguish the boundaries in the images shown in Figure 11a4,d4. Canny edge detector produces visually defined hole drilling edges, although some of them are not entirely closed or inside the drilling method tends to extract the pattern differences in wood chips of fiberboard (Figure 11a6,d6). Sobel, Laplace, and Canny edge filter segmented the milling gap shown in Figure 11c0. All conventional methods signify the differences in any pixel intensity changes. They do not carry out the ability to represent higher-level information or needs an additional step to perform data filtering. Moreover, methods tend to react to the pattern and require post-processing to finalize output prediction. We thresholded 50% of max processed (with filter) image intensity and clustered [63] edges points with a max distance of 5 pixels between neighbor pixels. Each cluster was closed with a convex hull [64], because edges tend to be open. Moreover, too small (5 pixels area) and too big (more than 20% of image 320 $\times$ 320 size) were filtered out. Post-processed results of Sobel filter are shown in Figure 11a3–d3, Laplace filter in Figure 11a5–d5, and Canny edge detector in Figure 11a7–d7. Even after edges contours clusterization, and additional filters, it is hard to define the drilling boundaries. The drilled hole shown in Figure 11b0 is being not fully extracted by all image processing algorithms and Laplace filter and Canny edge detector algorithm is tends to react to surface noise in Figure 11a5,a7 respectively. All methods extracted cutout from Figure 11c0 and board edge from Figure 11b0,c0. Due to the maximum size contour, filter edges from Figure 11a0,d0 are filtered out. The performance results of discussed image processing algorithms are given in Table 3. Considering a small hole area in the image (as shown in data overlay maps in the image in Figure 10), algorithms yielded high accuracy because the most of background predicted correctly. However, *Precision*, *IoU*, and *Dice* scores reveal that the performance of drilling segmentation is not as high. Along with investigated image processing algorithms, it can be seen that the Canny edge detected performs best.

The edge cases where conventional methods fail to deliver satisfactory results can be seen in cases with a darker board surface pattern (Figure 12d–i filtered and clustered). Comparing with U-Net convolutional neural networks produced results (Figure 12c), it can be seen the differences between the data-driven and traditional methods capabilities in data dynamics. Even the shallow baseline U-Net model architecture captures the context with lightning variations (taking into consideration the precision around drilling edges).

Table 3. Image processing methods performance results.

Method	Accuracy	Recall	Precision	IoU	Dice
Sobel filter	0.996943	0.919077	0.637585	0.580435	0.590472
Laplace filter	0.959769	0.931860	0.651680	0.607032	0.614552
Canny edge detector	0.934371	0.978507	0.693433	0.677103	0.685342

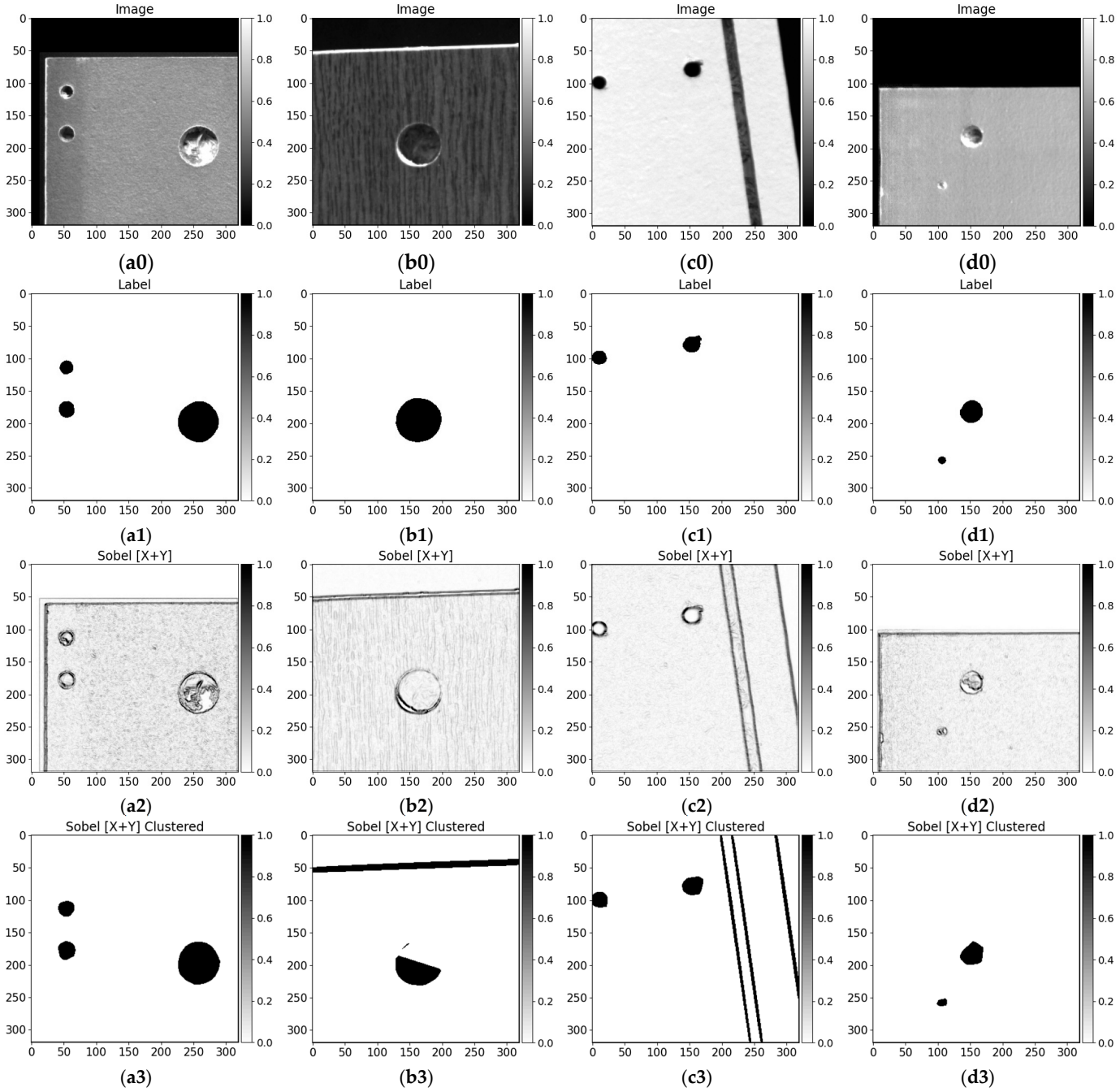


Figure 11. *Cont.*

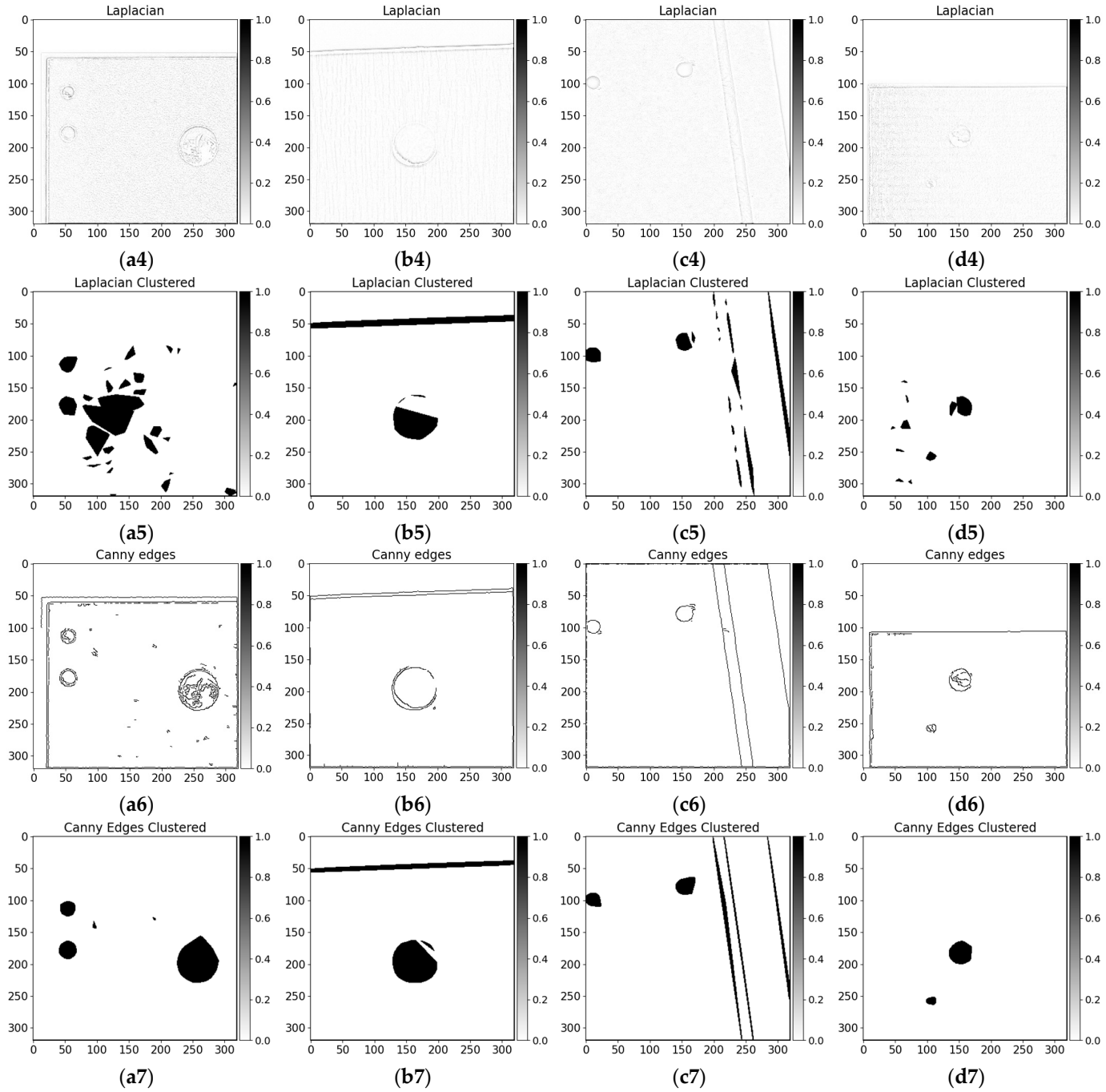


Figure 11. Sobel (3 × 3 filter, along X and Y axes), Laplace filter, and Canny edge detector results. Images—**a0**–**d0**, labels—**a1**–**d1**, Sobel filter results—**a2**–**d2,** Sobel filter clustered results—**a3**–**d3**, Laplace filter results—**a4**–**d4**, Laplace filter clustered results**—a5**–**d5,** Canny edge detector results—**a6**–**d6,** Canny edge detector clustered results—**a7**–**d7**.

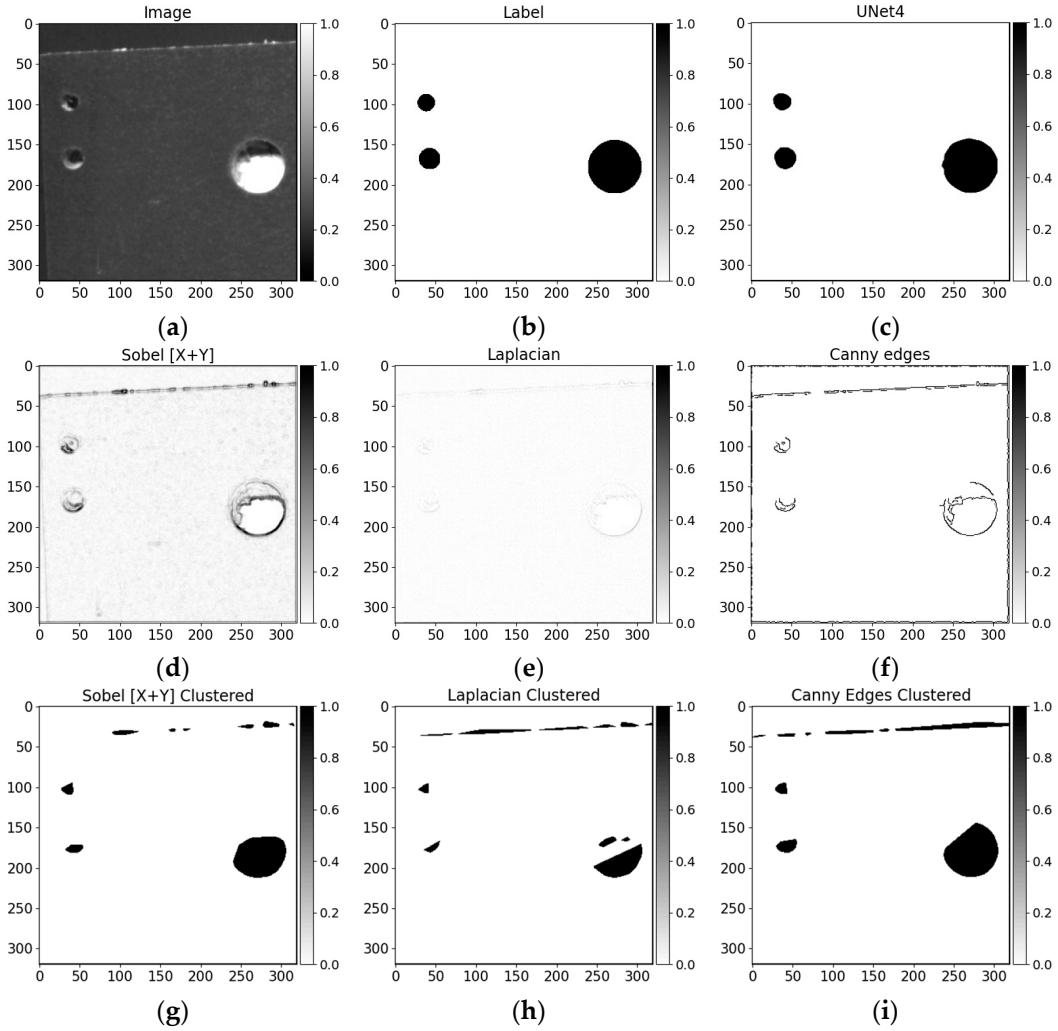


Figure 12. (**a**) Image, (**b**) label, (**c**) U-Net prediction, (**d**) Sobel filter, (**e**) Laplace filter, (**f**) Canny edges detector, (**g**) Sobel filter clustered, (**h**) Laplace filter clustered, (**i**) Canny edges detector clustered.

6.2. Convolutional Neural Network Results

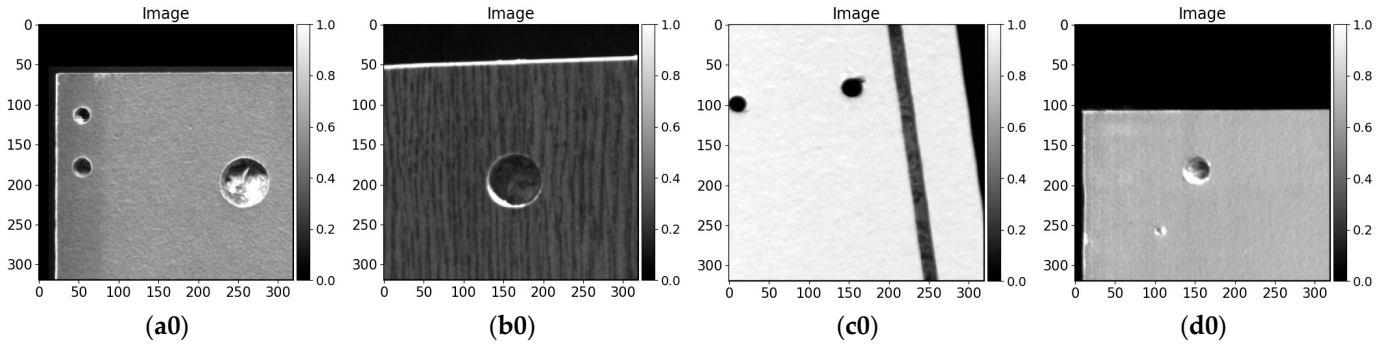
Each convolutional neural network architecture's best-performing weights are picked according to the best *Dice* score on the test dataset. The results are given in Table 4. Any additional block to "baseline" *UNet* increased most of the overall results. A minimal 0.8504% increase in the *Dice* score can be seen by only enhancing the model with the squeeze and excitation blocks (*UNet* + *SE*). A more noticeable score increase can be seen by any other (*CoordConv, Res* + *ASPP*, etc.) addition to the original *UNet* model. The biggest *Dice* score is produced by encoder-decoder architecture with residual connections, atrous spatial pyramid pooling module, squeeze and excitation blocks (*UNet* + *Res* + *ASPP* + *SE*). It surpasses "baseline" by 3.3905% (in *Dice* score). Moreover, the particular solution yielded the highest *Recall* score. The top result in precision is produced by *UNet* with squeeze and excitation and *CoordConv* (*UNet* + *SE* + *CoordConv2D*). The same solution gave the highest intersection over union (*IoU*) score. Comprehensively, the accuracy measurement in this data case is not relevant, because it does not reflect the actual prediction performance accurately properly label-wise. The hole annotation is small and it takes a relatively small area compared with the background. The true negatives (*TN* is the right prediction on the background) make the biggest influence area-wise on the overall *Accuracy*, while the true positive (*TP* is right-predicted drilled hole pixels) might not make a significant impact on the score. This can also be seen in Table 4, where the differences in *Accuracy* measurements

along models are indistinguishable and severely saturated due to precise predictions along most of the image context.

Table 4. Each model's best-performing weights results.

CNN Architecture	Accuracy	Recall	Precision	IoU	Dice
UNet	0.999485	0.959081	0.958613	0.955272	0.944966
UNet + SE	0.998978	0.936343	0.978481	0.979132	0.953470
UNet + CoordConv2D	0.999390	0.961770	0.975089	0.973536	0.965975
UNet + SE + CoordConv2D	0.999102	0.949620	0.983831	0.980100	0.962330
UNet + Res + ASPP	0.999475	0.959433	0.973082	0.970765	0.961194
UNet + Res + ASPP + SE	0.999681	0.982027	0.977736	0.975958	0.978871
UNet + Res + ASPP + CoordConv2D	0.999548	0.967609	0.977881	0.974820	0.969476
UNet + Res + ASPP + SE + CoordConv2D	0.999414	0.962808	0.977196	0.974946	0.968346

Each model's output on four different test set samples is given in Figure 13. We show results on the same data samples processed by conventional image processing methods (Figure 11). All architectures performing well on more common drilling samples, such as the left side of Figure 13a0 or the left side of 13c0. Moreover, all models are able to detect holes and separate them from another furniture panel processing, the milling cutout (Figure 13c0), despite the same wood chip pattern below the surface lamination. In drilling segmentation, even baseline *UNet* delivers visually appropriate results. Although, according to *Precision* (Table 4), the architecture yields more false-alarm predictions. The difference between convolutional neural networks might be more significant around the drilled hole edges and in more arduous samples. Figure 13a0,b0 have wider drilled holes. Additionally, there are drilled holes sides that are contrary illuminated—the lower part is more saturated. Sample in Figure 13a0 is handled better; however, *UNet + SE + CoordConv2D* is not as capable to segment the right side of the drilled hole (Figure 13a5). The same solution produces a small gap in Figure 13b5. Slight variations in prediction output can be seen between *UNet + SE* (Figure 13b3), *UNet + RES + ASPP* (Figure 13b6) and *UNet + RES + ASPP + CoordConv* (Figure 13b8) around the lower saturated hole edge—dilated or eroded edge. A rarer case with shallow drilling is given in Figure 13d0. A smaller diameter hole is entirely lit up and also the bottom part of the drilling might be similar to the top lamination (color- and texture-wise). Models enhanced with residual connections and atrous spatial pyramid pooling are able to capture the bigger context of the drilling. Interestingly enough, even "baseline" *UNet* segments a similar area of the hole. However, mentioned model's drawback can be highlighted on the same image (Figure 13d0) centered drilling. In Figure 13d2, the "visual roundness" of the extraction is not as good as from models with *RES* and *ASPP*. However, architecture configuration *CoordConv* and squeeze and excitation modules (Figure 13d5) yields even worse output.



(a0) **(b0)** **(c0)** **(d0)**

Figure 13. *Cont.*

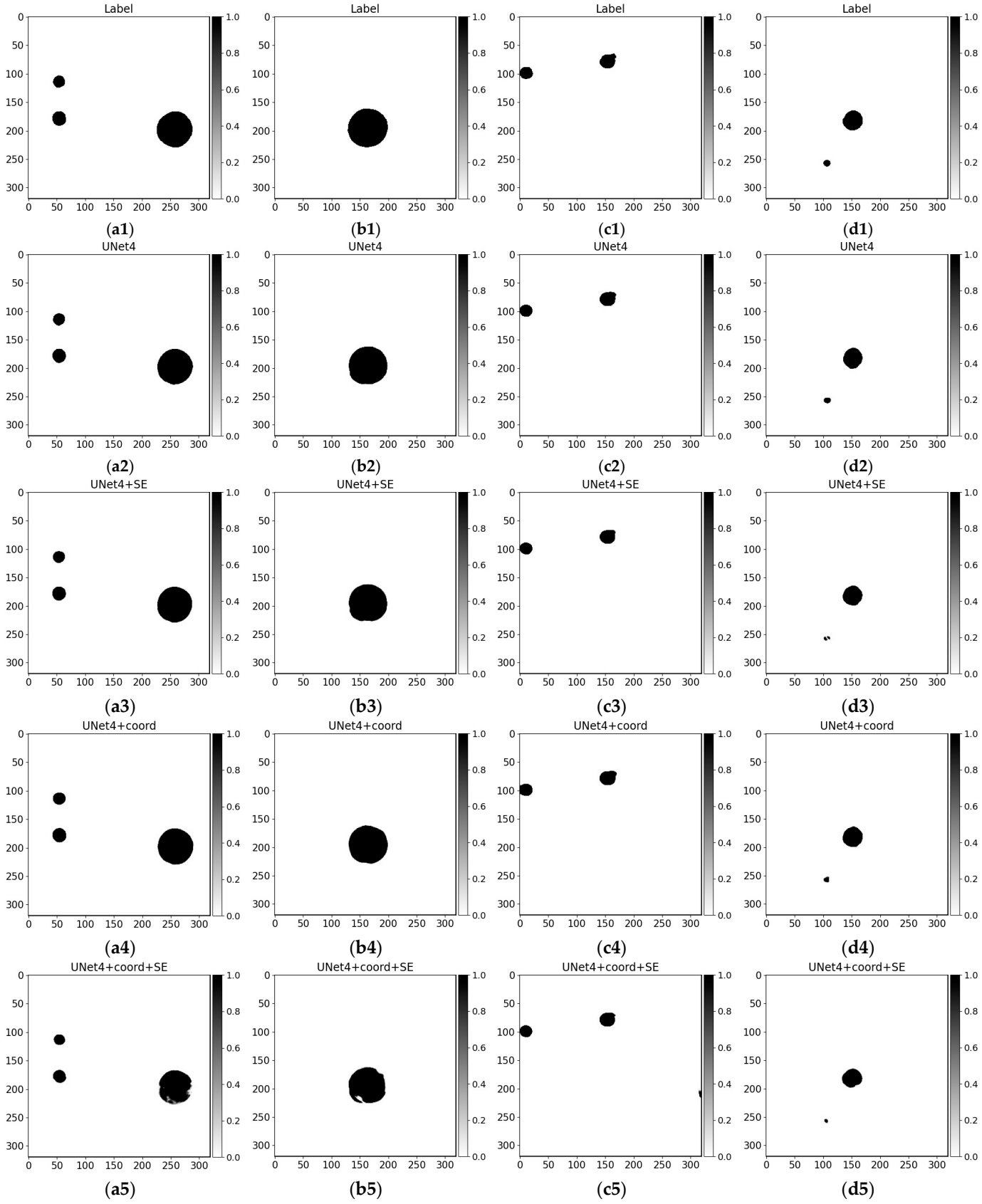


Figure 13. *Cont*.

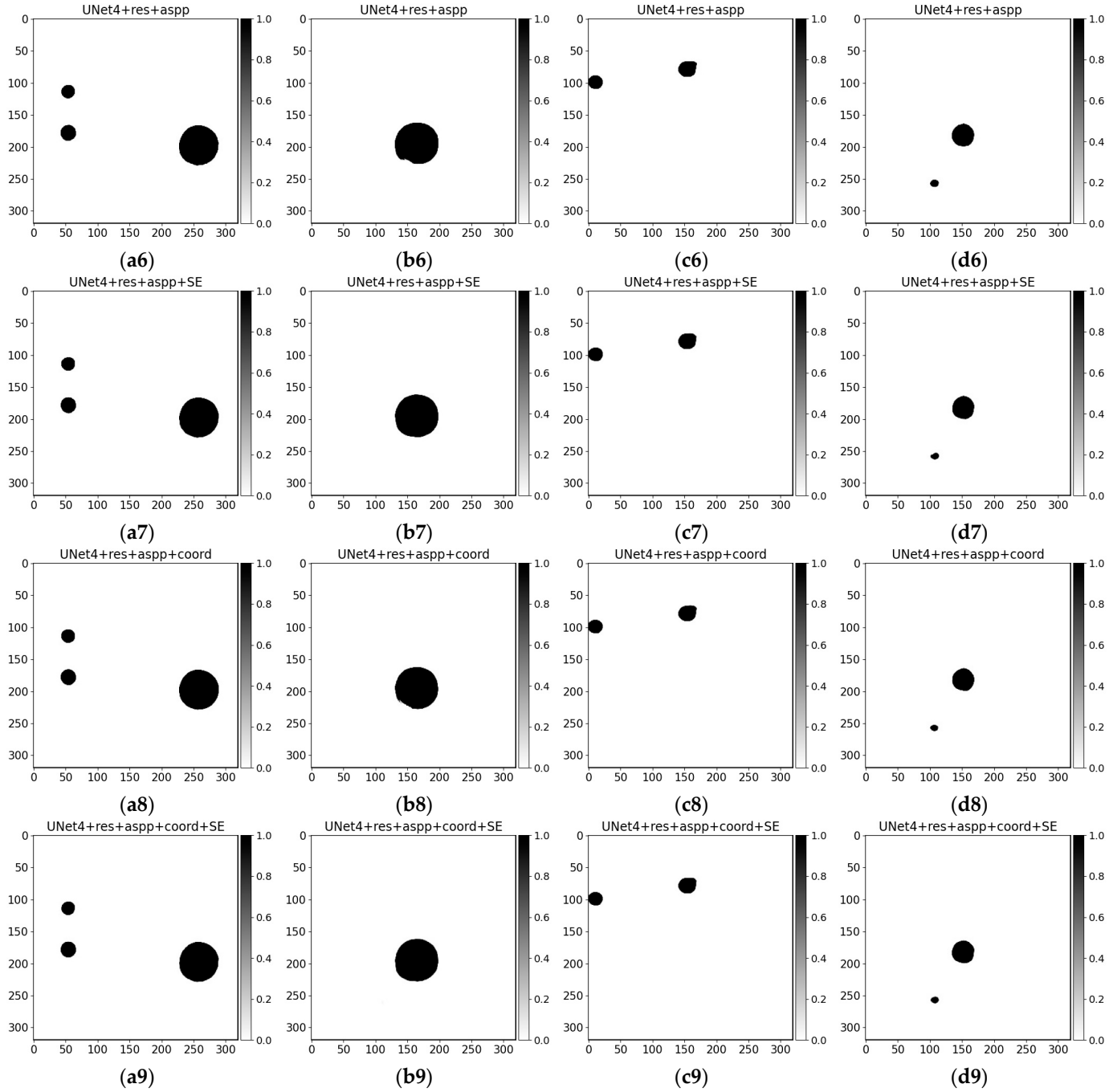


Figure 13. Each neural network architecture output on 4 cropped samples from the test set. Images (**a0**–**d0**), labels (**a1**–**d1**), network predictions (**a2**–**d9**).

Despite the models' output precision benchmarks and visual evaluation, the computational performance aspect needs to be taken into consideration. Prediction speed is also critical in the best solution selection because the drilling visual analysis time is limited. The tradeoff between the speed and precision needs to be taken into comparison. While there are a lot of enhancements to the "baseline" *UNet*, there can be a noticeable increase in parameters. As it is given in Figure 14, solutions with residual connections and atrous spatial pyramid pooling modules double the number of neural network parameters. The minimal difference can be seen in architectures with *CoordConv* and slightly bigger in modifications with the squeeze and excitation blocks.

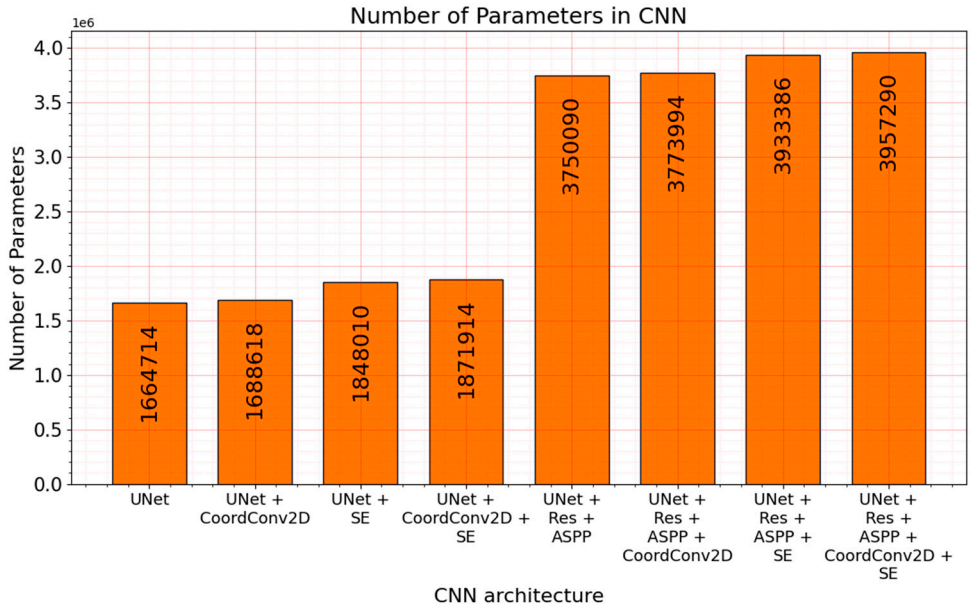


Figure 14. Number of parameters in each convolutional encoder–decoder.

However, the number of parameters does not directly correlate with computational speed. As it can be seen in Figure 15, the architectures with the biggest number of parameters (enhanced with residual connections and atrous spatial pyramid pooling) are not increasing prediction time significantly.

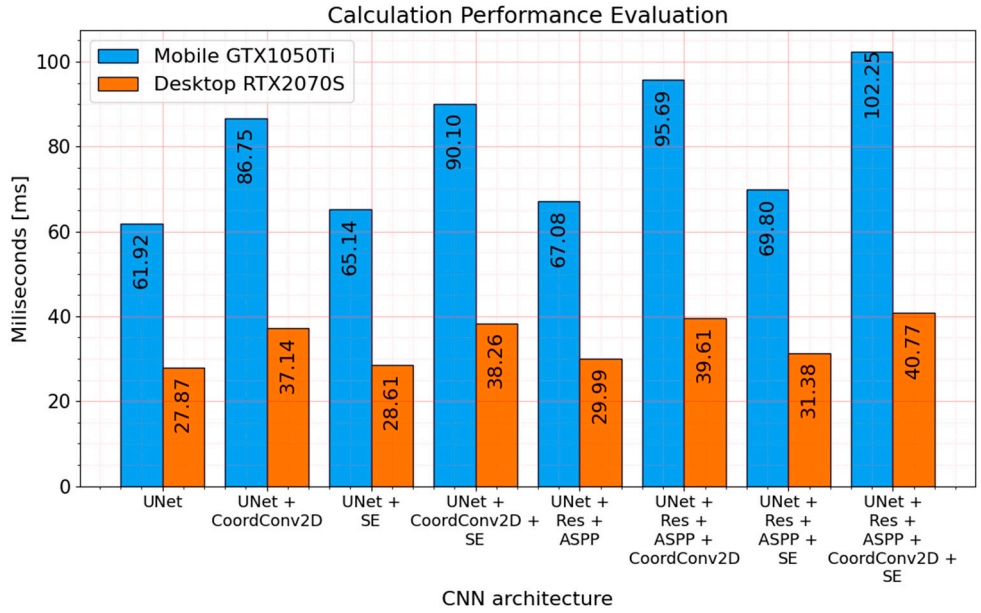


Figure 15. Each convolutional encoder–decoder performance prediction speed on Nvidia GTX1050Ti mobile (laptop) and Nvidia RTX2070 Super (desktop) GPUs. Tensorflow 2.4.0 prebuild from Python PIP package manager is used. Each time is averaged from 1000 forward image passes through the individual model.

Comparing *UNet* and *UNet + RES + ASPP* computational speed, there is only a 7.61% increase in the system with Nvidia RTX 2070 Super and 8.33% in Nvidia GTX 1050 Ti laptop GPU. Even smaller prediction time increases can be seen in solution with a squeeze and excitation blocks (*UNet + SE*)—2.66% and 5.20% in desktop and laptop GPUs, respectively. In this particular case, computational speed decrease is more noticeable in mobile GTX

1050Ti. The biggest prediction time bump is noticeable in every solution with *CoordConv*—33.26% and 40.10%, respectively, in desktop and laptop machines. However, it is not a native Tensorflow 2.4.0 library layer and the results might be improved. Further, the speed might vary on implementation. The best-performing solution according to *Dice* score (*UNet + RES + ASPP + SE*) takes 12.59% and 12.73% more time or 3.51 and 7.88 milliseconds longer, respectively, in investigated desktop and laptop computers. The time for multiple image processing can be reduced by passing multiple images at once. For example, processing of input consisting of 16 images (shape (16, 320, 320, 1)) took 148.11ms on RTX2070S with *UNet + Res + ASPP + SE* model.

7. Discussion

In this work, we proposed a computer-vision-based approach for drilled blind and through-hole segmentation in wood chip furniture panels using convolutional neural networks. We also conducted experiments with *Sobel*, *Laplace* filters and *Canny* edge detector for comparison. The conventional image processing methods tend to segment simple samples; however, even with post-processing and edges filtering it was hard to fully distinguish the edges of the drilling in complicated cases. Moreover, methods reacted to the intensity differences on the board edges and complicated board surfaces. The best performing solution with image processing—*Canny* edge detector produced a 0.685342 *Dice* score, which significantly fell behind the baseline UNet solution with 0.944966.

On the samples containing a large variety of different surface lamination textures, milling cuts, and other faults appearing in the production, deep-learning-based models performed well. It was shown that despite the complexity in images, even a lightweight UNet model is able to generalize and segment drilled holes. This research revealed that more advanced modules and layers increased the model's segmentation accuracy. Differences might be more distinguishable in more complicated samples. As the main subject of the investigation, UNet architecture was enhanced with squeeze and excitation block, *CoordConv* layers, residual connections, and atrous spatial pyramid pooling modules and inspected in segmentation and computational performance. All proposed model architectures with modifications yield results with a higher *Dice* score, compared with "baseline" architecture. Neural network models induced with squeeze and excitation (*UNet + SE*) raised *Dice* results by the minimum 0.8504%, while significantly better composition with *CoordConv* (*UNet + CoordConv*) boosted by 2.1009%. However, the combination of these two mentioned modules with "baseline" UNet (*UNet + SE + CoordConv*) did not give a better solution. From the images, it can be seen that it outputs significantly worse results in more rare cases. The best-proposed neural network configuration employed in this research was UNet with residual connection, atrous spatial pyramid pooling, and squeeze and excitation blocks (*UNet + RES + ASPP + SE*). It increased *Dice* score by 3.3905% (comparing with "baseline" UNet), scoring 0.978871 on 320 $\times$ 320 pixel image in 31.38 and 69.8 milliseconds (taking 3.51 and 7.88 milliseconds more than "baseline" solution) on desktop RTX 2070S and laptop GTX 1050Ti. Enhancing this architecture with *CoordConv* resulted in poorer segmentation. Moreover, in all cases, the mentioned layer resulted in a significantly bigger computation time $-$33.26% and 40.10% in desktop and laptop systems, respectively, comparing to the base model. On the other hand, it was the custom *CoordConv* implementation that was not a part of the deep learning framework. The overall time of analysis can be reduced by passing bigger input formed from multiple images to the model. The input of 16 images (16, 320, 320, 1) took 148.11ms to process and the input of one image (1, 320, 320, 1) took 31.38ms on RTX2070S with *UNet + RES + ASPP + SE* model.

The proposed neural network model or modifications can be engaged in problems such as remote image segmentation [65,66], medicine [67,68], faults detection [69,70], and others.

8. Integration and Future Work

Drilled hole segmentation from the whole furniture panel can be a huge overhead for inspection timewise, taking into consideration huge image dimensions (the maximum size of the image is 6144 × 12384 pixels). Moreover, not all the panel's area needs to be drilled. Therefore, knowing the place in the panel where drilling should be, only certain regions might be fed into the drilled hole segmentation neural network. Identified reference point in all particular model furniture panels can be assigned as coordinates system. From this point, all the drilling, according to the furniture template, needs to be located in the same places. The top-left point of the panel can be taken as the reference for the coordinate system. By extracting the panel from the conveyor belt and calculating the intersection between the top and the left side (panel's edge) extrapolated lines, the coordinate system's start could be found. Moreover, the part rotation can be evaluated from found edge lines. The idea of hole region search is shown in Figure 16.

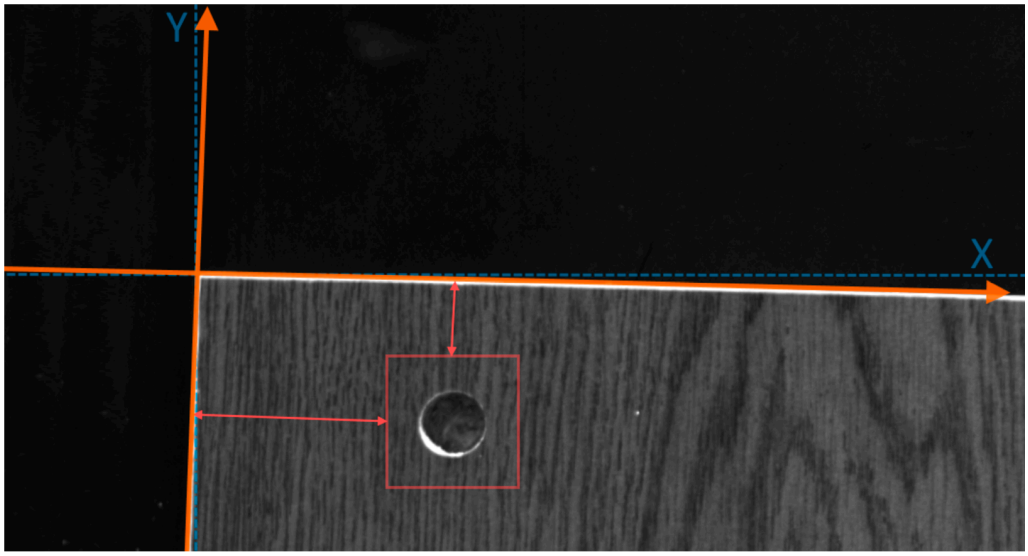
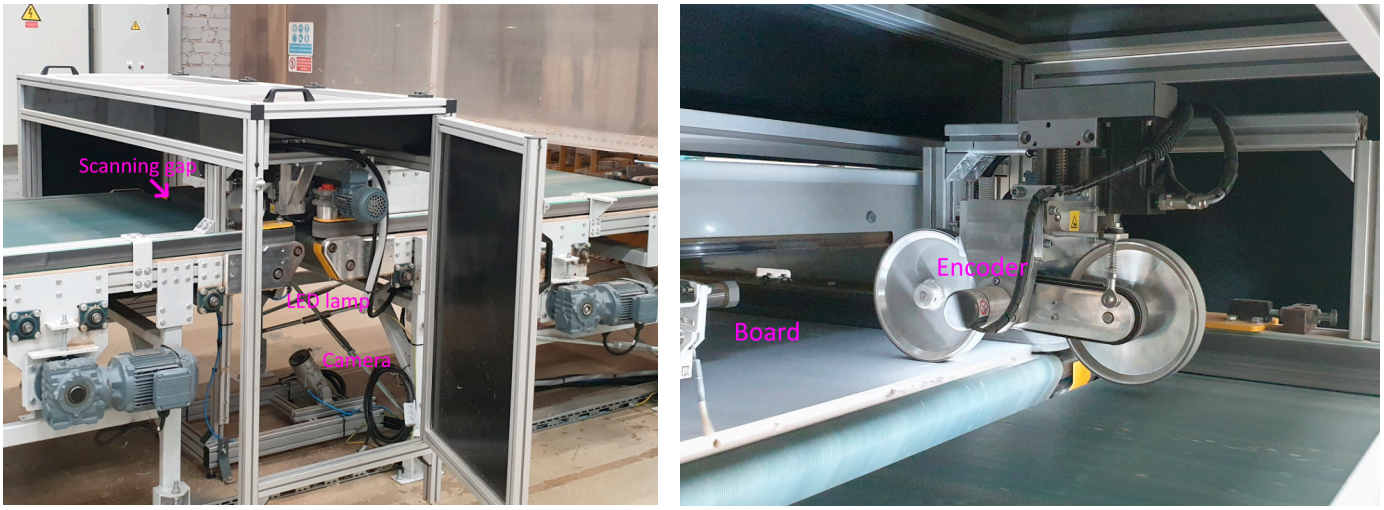


Figure 16. Furniture panel reference point and coordinate systems. The rotation of the object is evaluated and the coordinates system is turned accordingly. The hole drilling segmentation region is offset from the reference point by the given distance.

The quality of the segmented drilled hole can be determined based on the *Dice* score or area differences between the template board and processed board. Further, the drilled hole position, according to its mass center point, can be evaluated. The drilled hole center point distance from the reference system start should be the same or diverge with the allowed error.

Real inspection system implementation is given in Figure 17. A camera is placed near the ground and the LED light source is directed upwards (towards camera sensors direction). It is a different configuration than given in Figure 7. The camera (Figure 17a) is placed inside an additional metal safety cover with transparent windows that is blown by compressed air to remove the dust. Scanning is made through the gap between two conveyors. The camera is triggered by the encoder mounted on the roller that presses down the furniture board (Figure 17b), preventing it from shaking. Further, the rollers are covered with rubber to provide the grip with the board for precise movement detection (with encoder) that gives proper camera trigger. The whole image analysis system is covered to block outside light interference with separate analysis system lightning.



(**a**) (**b**)

Figure 17. Image capture setup: (**a**) vision inspection box, (**b**) board scanning place. The camera is triggered with an encoder that is rotated by a roller that pushes down the furniture board.

Visual wooden furniture panels surface inspection might take a different kind of algorithm than the proposed drilled holes segmentation method. However, the drilling regions should not be taken into consideration with regular (without drilling) areas in the furniture panel surface, or the drilled holes might be taken out from these regions and the rest of the region area could be considered as a regular surface and processed with surface defects detection algorithms.

In future work, we are considering utilizing a more advanced algorithm for surface defect inspection and edge inspection. The defects, such as faulty gluing and deficiency in paint coverage, appear in the lamination process. Moreover, surface damages might appear in any stage of manufacturing. Therefore, the inspection can be made from the same visual data.

Author Contributions: Conceptualization, R.A., A.L., and T.S.; methodology, R.A.; software, R.A.; validation, R.A., A.L., and T.S.; formal analysis, R.A. and T.S.; investigation, R.A., A.L., and T.S.; resources, T.S.; data curation, R.A. and T.S.; writing—original draft preparation, R.A.; writing—review and editing, R.A., A.L., and T.S.; visualization, R.A.; supervision, A.L. and T.S.; project administration, T.S. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Contact MB Prorega (email: tadas@prorega.lt) for dataset inquiries.

Acknowledgments: We would like to express our appreciation to the company MB Prorega for providing data grabbing equipment, furniture panels samples, help and tips in data acquisition, data labeling, and for the overall contribution to this research.

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

References

1. Czimmermann, T.; Ciuti, G.; Milazzo, M.; Chiurazzi, M.; Roccella, S.; Oddo, C.M.; Dario, P. Visual-Based Defect Detection and Classification Approaches for Industrial Applications—A SURVEY. *Sensors* **2020**, *20*, 1459. [CrossRef] [PubMed]
2. He, K.; Zhang, X.; Ren, S.; Sun, J. Deep residual learning for image recognition. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 27–30 June 2016; pp. 770–778.
3. Chen, L.-C.; Papandreou, G.; Schroff, F.; Adam, H. Rethinking atrous convolution for semantic image segmentation. *arXiv* **2017**, arXiv:1706.05587.

4. Hu, J.; Shen, L.; Sun, G. Squeeze-and-excitation networks. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 27–30 June 2018; pp. 7132–7141.
5. Liu, R.; Lehman, J.; Molino, P.; Such, F.P.; Frank, E.; Sergeev, A.; Yosinski, J. An Intriguing Failing of Convolutional Neural Networks and the CoordConv Solution. July 2018. Available online: https://proceedings.neurips.cc/paper/2018/file/60106888f8977b71e1f15db7bc9a88d1-Paper.pdf (accessed on 4 April 2021).
6. Augustauskas, R. Models Implementation Code. 2021. Available online: https://github.com/rytisss/PanelsDrillSegmentation (accessed on 5 April 2021).
7. Hernandez, A.; Maghami, A.; Khoshdarregi, M. A Machine Vision Framework for Autonomous Inspection of Drilled Holes in CFRP Panels. In Proceedings of the 2020 6th International Conference on Control, Automation and Robotics (ICCAR), Singapore, 20–23 April 2020; Institute of Electrical and Electronics Engineers (IEEE): Piscataway, NJ, USA, 2020; pp. 669–675.
8. Caggiano, A.; Angelone, R.; Teti, R. Image Analysis for CFRP Drilled Hole Quality Assessment. *Procedia CIRP* **2017**, *62*, 440–445. [CrossRef]
9. Otsu, N. A threshold selection method from gray-level histograms. *IEEE Trans. Syst. Man Cybern.* **1979**, *9*, 62–66. [CrossRef]
10. Yu, L.; Bi, Q.; Ji, Y.; Fan, Y.; Huang, N.; Wang, Y. Vision based in-process inspection for countersink in automated drilling and riveting. *Precis. Eng.* **2019**, *58*, 35–46. [CrossRef]
11. Canny, J. A Computational Approach to Edge Detection. *IEEE Trans. Pattern Anal. Mach. Intell.* **1986**, *PAMI-8*, 679–698. [CrossRef]
12. Li, G.; Yang, S.; Cao, S.; Zhu, W.; Ke, Y. A semi-supervised deep learning approach for circular hole detection on composite parts. *Vis. Comput.* **2021**, *37*, 433–445. [CrossRef]
13. He, D.-C.; Wang, L. Texture Unit, Texture Spectrum, and Texture Analysis. *IEEE Trans. Geosci. Remote Sens.* **1990**, *28*, 509–512. [CrossRef]
14. Deng, J.; Dong, W.; Socher, R.; Li, L.J.; Li, K.; Li, F.F. Imagenet: A Large-Scale Hierarchical Image Database. In Proceedings of the 2009 IEEE Conference on Computer Vision and Pattern Recognition, Miami, FL, USA, 20–25 June 2009; pp. 248–255.
15. Lin, T.; Maire, M.; Belongie, S.J.; Bourdev, L.D.; Girshick, R.B.; Hays, J.; Zitnick, C.L. Microsoft COCO: Common Objects in Context. In *Computer Vision—ECCV 2014. ECCV 2014. Lecture Notes in Computer Science*; Fleet, D., Pajdla, T., Schiele, B., Tuytelaars, T., Eds.; Springer: Cham, Switzerland, 2014; Volume 8693. [CrossRef]
16. Kuznetsova, A.; Rom, H.; Alldrin, N.; Uijlings, J.; Krasin, I.; Pont-Tuset, J.; Kamali, S.; Popov, S.; Malloci, M.; Kolesnikov, A.; et al. The Open Images Dataset V4. *Int. J. Comput. Vis.* **2020**, *128*, 1956–1981. [CrossRef]
17. Touvron, H.; Vedaldi, A.; Douze, M.; Jegou, H. Fixing the train-test resolution discrepancy. In *Advances in Neural Information Processing Systems*; Curran Associates, Inc.: Red Hook, NY, USA, 2019; Volume 32. Available online: https://proceedings.neurips.cc/paper/2019/file/d03a857a23b5285736c4d55e0bb067c8-Paper.pdf (accessed on 4 April 2021).
18. Du, X.; Lin, T.-Y.; Jin, P.; Ghiasi, G.; Tan, M.; Cui, Y.; Le, Q.V.; Song, X. SpineNet: Learning Scale-Permuted Backbone for Recognition and Localization. In Proceedings of the 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Seattle, WA, USA, 14–19 June 2020; Institute of Electrical and Electronics Engineers (IEEE): Piscataway, NJ, USA, 2020; pp. 11589–11598.
19. Kaggle Competition. Open Images 2019. Deep Neural Network ResNeXt152 Solution. Kaggle Competition. 2019. Available online: https://www.kaggle.com/c/open-images-2019-object-detection/discussion/110953 (accessed on 21 February 2021).
20. Qian, K. Automated Detection of Steel Defects via Machine Learning based on Real-Time Semantic Segmentation. *ACM Int. Conf. Proceeding Ser.* **2019**, 42–46. [CrossRef]
21. Xue, B.; Chang, B.; Du, D. Multi-Output Monitoring of High-Speed Laser Welding State Based on Deep Learning. *Sensors* **2021**, *21*, 1626. [CrossRef]
22. Huang, X.; Liu, Z.; Zhang, X.; Kang, J.; Zhang, M.; Guo, Y. Surface damage detection for steel wire ropes using deep learning and computer vision techniques. *Measurement* **2020**, *161*, 107843. [CrossRef]
23. Gao, M.; Chen, J.; Mu, H.; Qi, D. A Transfer Residual Neural Network Based on ResNet-34 for Detection of Wood Knot Defects. *Forests* **2021**, *12*, 212. [CrossRef]
24. Yang, Y.; Zhou, X.; Liu, Y.; Hu, Z.; Ding, F. Wood Defect Detection Based on Depth Extreme Learning Machine. *Appl. Sci.* **2020**, *10*, 7488. [CrossRef]
25. Urbonas, A.; Raudonis, V.; Maskeliūnas, R.; Damaševičius, R. Automated Identification of Wood Veneer Surface Defects Using Faster Region-Based Convolutional Neural Network with Data Augmentation and Transfer Learning. *Appl. Sci.* **2019**, *9*, 4898. [CrossRef]
26. Liu, S.; Jiang, W.; Wu, L.; Wen, H.; Liu, M.; Wang, Y. Real-Time Classification of Rubber Wood Boards Using an SSR-Based CNN. *IEEE Trans. Instrum. Meas.* **2020**, *69*, 8725–8734. [CrossRef]
27. Sheu, R.-K.; Teng, Y.-H.; Tseng, C.-H.; Chen, L.-C. Apparatus and Method of Defect Detection for Resin Films. *Appl. Sci.* **2020**, *10*, 1206. [CrossRef]
28. Muresan, M.P.; Cireap, D.G.; Giosan, I. Automatic Vision Inspection Solution for the Manufacturing Process of Automotive Components Through Plastic Injection Molding. In Proceedings of the 2020 IEEE 16th International Conference on Intelligent Computer Communication and Processing (ICCP), Cluj-Napoca, Romania, 3–5 September 2020; Institute of Electrical and Electronics Engineers (IEEE): Piscataway, NJ, USA, 2020; pp. 423–430.

29. Lenty, B. Machine vision system for quality control of molded plastic packaging. In Proceedings of the Photonics Applications in Astronomy, Communications, Industry, and High-Energy Physics Experiments 2019, Wilga, Poland, 6 November 2019; p. 77. [CrossRef]
30. Yin, X.; Chen, Y.; Bouferguene, A.; Zaman, H.; Al-Hussein, M.; Kurach, L. A deep learning-based framework for an automated defect detection system for sewer pipes. *Autom. Constr.* **2020**, *109*, 102967. [CrossRef]
31. Redmon, J.; Farhadi, A. YOLO9000: Better, Faster, Stronger. In Proceedings of the 30th IEEE Conference on Computer Vision and Pattern Recognition, Honolulu, HI, USA, 21–26 July 2017; pp. 6517–6525.
32. Adibhatla, V.A.; Chih, H.-C.; Hsu, C.-C.; Cheng, J.; Abbod, M.F.; Shieh, J.-S. Defect Detection in Printed Circuit Boards Using You-Only-Look-Once Convolutional Neural Networks. *Electronics* **2020**, *9*, 1547. [CrossRef]
33. Su, B.; Chen, H.Y.; Chen, P.; Bian, G.-B.; Liu, K.; Liu, W. Deep Learning-Based Solar-Cell Manufacturing Defect Detection with Complementary Attention Network. *IEEE Trans. Ind. Inform.* **2021**, *17*, 4084–4095. [CrossRef]
34. Girshick, R. Fast R-CNN. In Proceedings of the 2015 IEEE International Conference on Computer Vision (ICCV), Santiago, Chile, 7–13 December 2015; pp. 1440–1448.
35. Simonyan, K.; Zisserman, A. Very deep convolutional networks for large-scale image recognition. *arXiv* **2014**, arXiv:1409.1556.
36. Roberts, G.; Haile, S.Y.; Sainju, R.; Edwards, D.J.; Hutchinson, B.; Zhu, Y. Deep Learning for Semantic Segmentation of Defects in Advanced STEM Images of Steels. *Sci. Rep.* **2019**, *9*, 1–12. [CrossRef]
37. Ronneberger, O.; Fischer, P.; Brox, T. U-net: Convolutional networks for biomedical image segmentation. In Proceedings of the International Conference on Medical Image Computing and Computer-Assisted Intervention, Cham, Switzedland, 5–9 October 2015; pp. 234–241.
38. Sandler, M.; Howard, A.; Zhu, M.; Zhmoginov, A.; Chen, L. Mobilenetv2: Inverted Residuals and Linear Bottlenecks. In Proceedings of the 2018 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Salt Lake City, UT, USA, 18–23 June 2018; pp. 4510–4520.
39. Lin, Z.; Ye, H.; Zhan, B.; Huang, X. An Efficient Network for Surface Defect Detection. *Appl. Sci.* **2020**, *10*, 6085. [CrossRef]
40. DAGM. Weakly Supervised Learning for Industrial Optical Inspection. DAGM Dataset. 2007. Available online: https://hci.iwr.uni-heidelberg.de/node/3616 (accessed on 4 April 2021).
41. Huang, Y.; Qiu, C.; Wang, X.; Wang, S.; Yuan, K. A Compact Convolutional Neural Network for Surface Defect Inspection. *Sensors* **2020**, *20*, 1974. [CrossRef]
42. Niskanen, M.; Kauppinen, H. Wood inspection with non-supervised clustering. *Mach. Vis. Appl.* **2003**, *13*, 275–285. [CrossRef]
43. Kechen, S.; Yunhui, Y. Northeastern University (NEU) Surface Defect Database. Available online: http://faculty.neu.edu.cn/yunhyan/NEU_surface_defect_database.html (accessed on 4 April 2021).
44. Danielsson, P.-E.; Seger, O. Generalized and Separable Sobel Operators. In *Machine Vision for Three-Dimensional Scenes*; Elsevier BV: Amsterdam, The Netherlands, 1990; pp. 347–379.
45. van Vliet, L.J.; Young, I.T.; Beckers, G.L. A nonlinear laplace operator as edge detector in noisy images. *Comput. Vis. Graph. Image Process.* **1989**, *45*, 167–195. [CrossRef]
46. Ioffe, S.; Szegedy, C. Batch normalization: Accelerating deep network training by reducing internal covariate shift. *arXiv* **2015**, arXiv:1502.03167.
47. Gholami, A.; Kwon, K.; Wu, B.; Tai, Z.; Yue, X.; Jin, P.; Zhao, S.; Keutzer, K. SqueezeNext: Hardware-Aware Neural Network Design. In Proceedings of the 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW), Salt Lake City, UT, USA, 18–22 June 2018; pp. 1719–171909.
48. Chen, L.-C.; Papandreou, G.; Kokkinos, I.; Murphy, K.; Yuille, A.L. DeepLab: Semantic Image Segmentation with Deep Convolutional Nets, Atrous Convolution, and Fully Connected CRFs. *IEEE Trans. Pattern Anal. Mach. Intell.* **2017**, *40*, 834–848. [CrossRef] [PubMed]
49. Szegedy, C.; Ioffe, S.; Vanhoucke, V.; Alemi, A.A. Inception-v4, Inception-ResNet and the Impact of Residual Connections on Learning. In Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence, San Francisco, CA, USA, 4–9 February 2017.
50. Chen, G.; Li, C.; Wei, W.; Jing, W.; Woźniak, M.; Blažauskas, T.; Damaševičius, R. Fully Convolutional Neural Network with Augmented Atrous Spatial Pyramid Pool and Fully Connected Fusion Path for High Resolution Remote Sensing Image Segmentation. *Appl. Sci.* **2019**, *9*, 1816. [CrossRef]
51. Liu, W.; Rabinovich, A.; Berg, A.C. ParseNet: Looking Wider to See Better. June 2015. Available online: http://arxiv.org/abs/1506.04579 (accessed on 4 April 2021).
52. el Jurdi, R.; Petitjean, C.; Honeine, P.; Abdallah, F. CoordConv-Unet: Investigating CoordConv for Organ Segmentation. *IRBM* **2021**. [CrossRef]
53. Zhao, H.; Jia, J.; Koltun, V. Exploring Self-Attention for Image Recognition. In Proceedings of the 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Seattle, WA, USA, 13–19 June 2020; pp. 10073–10082.
54. Uselis, A.; Lukoševičius, M.; Stasytis, L. Localized Convolutional Neural Networks for Geospatial Wind Forecasting. *Energies* **2020**, *13*, 3440. [CrossRef]
55. raL6144-16gm—Basler Racer Camera Website. Available online: https://www.baslerweb.com/en/products/cameras/line-scan-cameras/racer/ral6144-16gm/ (accessed on 4 April 2021).

56. AF Nikkor 24 mm f/2.8D Optics Website. Available online: https://www.nikon.lt/en_LT/product/nikkor-lenses/auto-focus-lenses/fx/single-focal-length/af-nikkor-24mm-f-2-8d (accessed on 4 April 2021).
57. Autonics E40S6-1500-3-T-24 Encoder Website. Available online: https://www.autonicsonline.com/product/product&product_id=14505 (accessed on 4 April 2021).
58. EBAR-1125-WHI-7 TPL-Vision LED Lamp Website. Available online: https://www.tpl-vision.fr/en/bar/ebar-plus/ (accessed on 4 April 2021).
59. Keras: The Python Deep Learning API. Available online: https://keras.io/ (accessed on 25 March 2021).
60. TensorFlow. An End-to-End Open Source Machine Learning Platform. Available online: https://www.tensorflow.org/ (accessed on 27 August 2020).
61. Dice, L.R. Measures of the Amount of Ecologic Association Between Species. *Ecology* **1945**, *26*, 297–302. [CrossRef]
62. Shindjalova, R.; Prodanova, K.; Svechtarov, V. Modeling data for tilted implants in grafted with bio-oss maxillary sinuses using logistic regression. *AIP Conf. Proc.* **2014**, *1631*, 58–62. [CrossRef]
63. Arthur, D.; Vassilvitskii, S. K-Means++: The Advantages of Careful Seeding. In Proceedings of the Eighteenth Annual ACM-SIAM Symposium on Discrete Algorithms, New Orleans, LA, USA, 7–9 January 2007; pp. 1027–1035.
64. Sklansky, J. Finding the convex hull of a simple polygon. *Pattern Recognit. Lett.* **1982**, *1*, 79–83. [CrossRef]
65. Dai, P.; Ji, S.; Zhang, Y. Gated Convolutional Networks for Cloud Removal from Bi-Temporal Remote Sensing Images. *Remote Sens.* **2020**, *12*, 3427. [CrossRef]
66. Zhang, M.; Jing, W.; Lin, J.; Fang, N.; Wei, W.; Woźniak, M.; Damaševičius, R. NAS-HRIS: Automatic Design and Architecture Search of Neural Network for Semantic Segmentation in Remote Sensing Images. *Sensors* **2020**, *20*, 5292. [CrossRef]
67. Raudonis, V.; Paulauskaite-Taraseviciene, A.; Sutiene, K. Fast Multi-Focus Fusion Based on Deep Learning for Early-Stage Embryo Image Enhancement. *Sensors* **2021**, *21*, 863. [CrossRef]
68. Khan, M.; Sharif, M.; Akram, T.; Damaševičius, R.; Maskeliūnas, R. Skin Lesion Segmentation and Multiclass Classification Using Deep Learning Features and Improved Moth Flame Optimization. *Diagnostics* **2021**, *11*, 811. [CrossRef]
69. Glowacz, A. Fault diagnosis of electric impact drills using thermal imaging. *Measurement* **2021**, *171*, 108815. [CrossRef]
70. Piekarski, M.; Jaworek-Korjakowska, J.; Wawrzyniak, A.I.; Gorgon, M. Convolutional neural network architecture for beam instabilities identification in Synchrotron Radiation Systems as an anomaly detection problem. *Measurement* **2020**, *165*, 108116. [CrossRef]