A Modular Multimodal Pipeline for Scratch Detection, Segmentation, and Coordinate Extraction on Polymer Ski Surfaces

Pavel Kostarev
Neu-Ulm University of Applied Sciences
Neu-Ulm, Germany
pavel.kostarev@hnu.de

Alexander Bartel
Neu-Ulm University of Applied Sciences
Neu-Ulm, Germany
alexander.bartel@hnu.de

***Abstract*—We present a system that can automatically locate and extract scratches from plastic ski surfaces using artificial intelligence (AI). Using a custom dataset consisting of 180 pairs of 2D grayscale and 3D depth profile images, we designed and compared three scratch detection and segmentation pipelines. The resulting segmented defects were mapped to X and Y coordinates for use in subsequent production processes. The most effective strategy fused computer vision (CV) algorithms with a multimodal machine learning architecture comprising a You Only Look Once (YOLO) module for scratch detection, a UNet++ module for scratch segmentation, and a region-growing algorithm to enhance segmentation. Our best approach achieved mean Average Precision (mAP) of 91% computed at Intersection over Union (IoU) threshold of 0.5 over all defect classes when running on the test set. This approach outperformed the other two in terms of time, performance, and accuracy, while also requiring minimal hardware. The proposed methodology is also parameterised to provide flexibility and allow adjustments for various industry applications. The modular nature of the approach allows new algorithms to be integrated and further pre- and post-processing to be carried out.**

***Index Terms*—Anomaly detection, Scratch detection, Segmentation models**

I. Introduction

Following recent advances in AI, and especially in Deep Learning, many companies are increasing their interest in using AI to enhance well-established business processes. This is also true when it comes to detection of anomalies in industrial applications [1]–[4]. When working with anomaly detection in images, many industrial systems still rely on classical CV methods [4], [5]. While these offer a wide range of algorithms that can often be efficiently combined for solving various use cases, they still require the configuration of multiple threshold parameters that often should be additionally adjusted manually by an operator. This complicates the design of the entire anomaly detection pipeline unnecessarily, making it vulnerable to small changes in future input data. While Deep Learning architectures, such as Convolutional Neural Networks (CNNs), have demonstrated robust capabilities in capturing anomaly patterns, they have also significantly improved the precision of anomaly detection [2], [4]. In our paper we would like to focus on a particular case, where a smart fusion of classical CV methods with deep learning models has significantly improved the accuracy of detected anomalies.

For repairing of scratches on the damaged polymer ski surfaces, an operator needs to precisely understand the depth of the scratched surface, where traditional 2D images are not sufficient as they often can be affected by factors like *illumination*, *reflectivity*, and *contrast*. Optical profilometers are very useful in such cases, as they can effectively capture 3D surface topology, producing height maps and 3D point clouds [6]. The latter can be very large heavy files when point precision is high, hence the height maps are often preferred over point clouds due to faster preprocessing time especially on a low capabilities hardware.

The extraction of anomalies in the traditional CV pipeline often includes multiple image preprocessing steps, such as particle filtering and usage of various thresholding methods (e.g., Adaptive, Otsu, or local thresholding methods). Another common strategy is transforming the image into frequency domain and working with the frequency patterns, applying Fast Fourier Transform (FFT), Hough Transformation [7], Gabor Filtering or Wavelet Decomposition [8]. To extract the X and Y coordinates for each segmented area, different skeletonization techniques are often preferred, which can be further preprocessed using graph structure (neighbourhood analysis and distance metrics) or smart heuristics.

While there are many papers which solve the particular problems of anomaly detection, segmentation, and geometric extraction tasks separately, our solution fills the gap in research by implementing a unique, low-latency, end-to-end framework, which step by step classifies, extracts, and maps scratches onto 2D coordinates, handling the optical artifacts and background noise with low latency. Our modular solution was optimized for both GPU and CPU-based hardware to meet different production use cases. Non-uniform scratches, background noise, logos and critical time and robustness requirements state a non-trivial problem, which required several approaches to test. Analyzing and comparing distinct methods, we found the best combination for our dataset. We believe that the practical lessons learned from our project can bring a valuable contribution into designing production-oriented AI systems for

other researchers, where finding right tools for input data is still a big problem.

The rest of the paper is organized as follows. In Section II we present a brief overview of literature related to the scope of the project. Section III covers the chosen methodology. We discuss and compare the results in Section IV. Finally, Section V concludes the paper, giving an overview of current limitations and potential further improvements.

II. Related work

Analyzing several systematic reviews on industrial surface defect detection we can draw the following conclusions:

- Encoder-decoder CNNs demonstrate the state-of-the-art performance in industrial use cases (especially using pyramid networks and attention-based architectures) [1];
- Gao et al. emphasize the improvements in anomaly detection accuracy when combining multimodal data (depth images with traditional images) [4];
- Qiao et al., in their review on metal surface defect detection, pay attention at the problem of capturing fine anomalies on metal surfaces by deep learning networks emphasizing the potential of combining multimodal data sources and classical computer vision methods [3].

Despite the traditional CV methods, there are established models that have proven their effectiveness. For example, for anomaly detection on wood [9]–[13], plastic [14], [15] metal steel surfaces [16]. YOLO is a very popular choice and has proven its effectiveness in the detection of anomalies.

The YOLO family of models provides a wide variety of models of different sizes. It is a convolutional object detector, which processes an image in one forward pass, splits it into a grid, and predicts, for each grid unit, a fixed number of bounding boxes. Each bounding box has a confidence score and class label [17]. Some variants of YOLO extend the object detection with Mask-Prediction Module, trained on Crack Segmentation Datasets [18].

Pyramid Convolution and Convolutional Attention Module can be additionally included in the network architecture to further enhance YOLO performance in an industrial setup, especially when detecting small defects [19], [20]. Pixel-wise segmentation of industrial defects is also commonly addressed by encoder-decoder architectures derived from U-Net. Architectures based on a U-Net backbone clearly outperform the classical CV threshold-based methods [21], [22]. The traditional U-Net architectures often tend to be extended via multi-scale feature fusion for better recall [23]. Mask R-CNN [24], [25] and DeepLabV3 [26], [27] networks also have been used for effective segmentation of surface defects in industry.

III. Methodology

In order to design an end-to-end framework for solving our problem case we have subdivided the entire process into 4 main steps, where within each step different tools were tested and compared: (1) Data preparation; (2) Logo detection and mask cleaning; (3) Scratch localization and segmentation; (4) Extraction of coordinates.

Our main focus of evaluation was, though, on the step (3), which was the most critical.

A. Input data & data preprocessing

The custom dataset consists out of (N=180) pairs of 8-bit grayscale and 32-bit height profile images of plastic ski surfaces. All images have the same size (1600x5600 pixels) and spatial resolution of 0.2 µm per pixel. Each image was manually assessed and classified as a "scratch" defect and annotated with bounding boxes using open source software tools like Labelme [28]. To ensure annotation quality, a peer review process involving several experts was conducted. For pixel-wise annotation of scratches (binary masks), we used a semi-supervised approach combining Labelme with Segment Anything Model (SAM ViT-H with $\approx$ 632M parameters) [29], which could accurately label deep scratches with high contrast. These steps established the strong ground truth (GT), both for bounding boxes and segmentation areas, which was essential for further model training.

Large images are very difficult to use for training without splitting them into smaller slices. Therefore, we have designed a 'puzzle-block'-inspired method, which tends to extract equal blocks of images, with exception for cases when tiling operation divides one scratch into several parts. To preserve the entire scratch in one image time, unequal slicing was applied (e.g., pixel blocks of 256x512, 256x1024). Consequently, on each image tile, we applied the following augmentation techniques: (1) Random *(p=0.5)* horizontal and vertical flips of the image; (2) Random hue, saturation or value channel shift; (3) Random change of image brightness and contrast; (4) Random change of gamma curve; (5) Adaptive histogram equalization for local contrast; (6) Gaussian and motion blur; (7) Gaussian noise.

The resulting dataset after all applied augmentations reached the amount of (N=33480) image-label pairs The dataset was split into 70 % training, 15 % validation, and 15% test sets.

B. Logo detection and mask filtering

We selected image tiles covering the background logo and applied Gaussian blur to these regions to prevent false positive detections (e.g., high-contrast scratches, misclassified as logo components). The tiles were then annotated using a SAM ViT-H model, resulting in a dataset for training a compact U-Net-based network. The network was trained on 512×512 RGB tiles to segment the logo from the image background, thereby preventing its detection by the scratch detector in the subsequent processing step. The network uses depthwise separable convolutions to reduce complexity. For a faster inference on CPU, the model was quantized. The pixels where the logo was detected are excluded from the following step, handling false positives.

C. Scratch Detection and segmentation

Using our custom dataset to effectively detect and extract scratches, we have compared the following approaches: 1.) Fully YOLO-based approach; 2.) Combination of DeepLabV3;

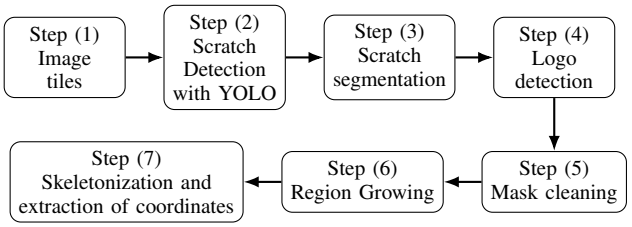

Fig. 1. Overview of the final proposed scratch detection pipeline.

Compact Watershed and SAM; 3.) UNet++ and region growing. More details can be found in Subsection IV-E.

D. Coordinates extraction

The predicted masks were further preprocessed by a block, which calculates the centerline for every mask. The X and Y coordinates of the centerlines produced for each polygon mask are finally saved in a .json file, which can be read by a 3D printer.

The overview of the final step-by-step approach using the trained models can be found in the Figure 1. To support the reader we deliver the overview of data preparation for training of models in the Figure 2. In Subsection IV-E we cover in detail the final chosen method.

IV. Results and Discussion

Before annotating our custom dataset of images entirely, we have also tried the following two transfer learning methods:

A. Zero-shot transfer learning

Prior to training a completely new model, we have tested if zero-shot transfer learning based on pre-trained models, such as ***YOLOv8-seg*** and ***YOLO11-seg***, could produce acceptable results. In both cases, we chose a medium size model, balancing precision and inference speed, with $\approx$ 25.9M parameters for YOLOv8-seg and $\approx$ 22.4M parameters for YOLO11-seg, respectively. For evaluating the models on our dataset, we used mean values for IoU and Dice score metrics, which are commonly accepted metrics for measuring the overlap between the GT and masks predicted by the model. Inference on both models indicated a pretty poor domain fit (*IoU = 0.193* and *Dice Score = 0.280* for YOLOv8-seg and *IoU = 0.464* and *Dice Score = 0.551* for YOLO11-seg, respectively). Only the biggest scratches with highest contrast relative to background (which covers less than a quarter of use cases) were correctly segmented.

Therefore, in the next step, we have decided to evaluate the domain adaptation by fine-tuning YOLOv8-seg and YOLO11-seg on our dataset.

B. Transfer learning via domain adaptation

Due to the limited amount of data, we decided to try the knowledge transfer approach, in which we fine tuned the YOLOv8-seg model with the open-source available datasets commonly used for scratch segmentation [30]. The fine-tuning and all further trainings mentioned in this paper were run on NVIDIA A40 GPU (46.07 GB VRAM). After 200 epochs with a batch size of 64, the average metrics were: *IoU=0.409*, *Dice Score=0.521*, *Precision=0.445*, *Recall=0.342*, *F1=0.387*). The same fine-tuning for YOLO11-seg showed slightly better results still indicating the poor overall performance (*IoU=0.420*, *Dice Score=0.598*, *Precision=0.571*, *Recall=0.483*, *F1=0.523*). This approach was unfeasible due to poor metric performance. Only scratches with a high visible contrast compared to the image background were identified.

The next three methods were evaluated only on our own, custom, annotated dataset.

C. Fine-tuning of YOLO on our dataset

Before starting the training process for maximization of fitness metric and segmentation accuracy, we performed a fine-tuning of hyperparameters (e.g., epochs, batch size, regularization, and image size). Our model training parameters for both YOLOv8-seg and YOLO11-seg were set as follows: 100 epochs, batch size = 64, image size = 512x512, and learning rate = 0.00031. The mAP@0.5 for a mask prediction and mAP@0.5 for a box prediction on a test set were 0.486 and 0.74 (for YOLOv8-seg) and 0.509 and 0.765 (for YOLO11-seg), respectively.

In this step, we have abandoned the idea of using YOLO for the segmentation step and use it only for prior inspection for the Region of Interest (ROI), returning bounding boxes with high confidence of localizing a scratch. This is supported by better metric performance on box prediction on our dataset.

D. DeepLabV3 combined with Segment Anything Model (SAM) and Compact Watershed

Using the YOLO model from previous trials to detect bounding boxes localizing the scratch, we have trained a DeepLabV3 model with a ResNet-101 architecture (batch size = 32, number of epochs = 100, image size = 512x512, and learning rate = $1e{-}4$) on our dataset. After evaluation on new, unseen data, we have noticed that the model fails at capturing continuous, fine, and thin scratches with low-contrast breaking them into partially detected segments. This might be due to the very strong downsampling nature of model's architecture, which employs pretty coarse feature maps.

To mitigate it and fill in the missing parts of scratches, we decided to extend the segmentation step for this approach with an additional step. Following the idea given in [31], where authors have effectively combined the results produced by a CNN-based neural network with superpixel segmentation by simple, linear, iterative clustering (SLIC) to effectively extract the scratch pixels, we have decided to extend our segmentation step with a SAM-guided soft prior using compact watershed for spatial regularization for superpixel level. We chose compact watershed due to better stability and higher speed performance in real-time applications compared to SLIC, as stated by authors [32]. Next, we additionally performed fine-tuning of SAM base model (SAM ViT-B with $\approx$ 91M parameters) to prevent detecting logos in the background as scratch regions.

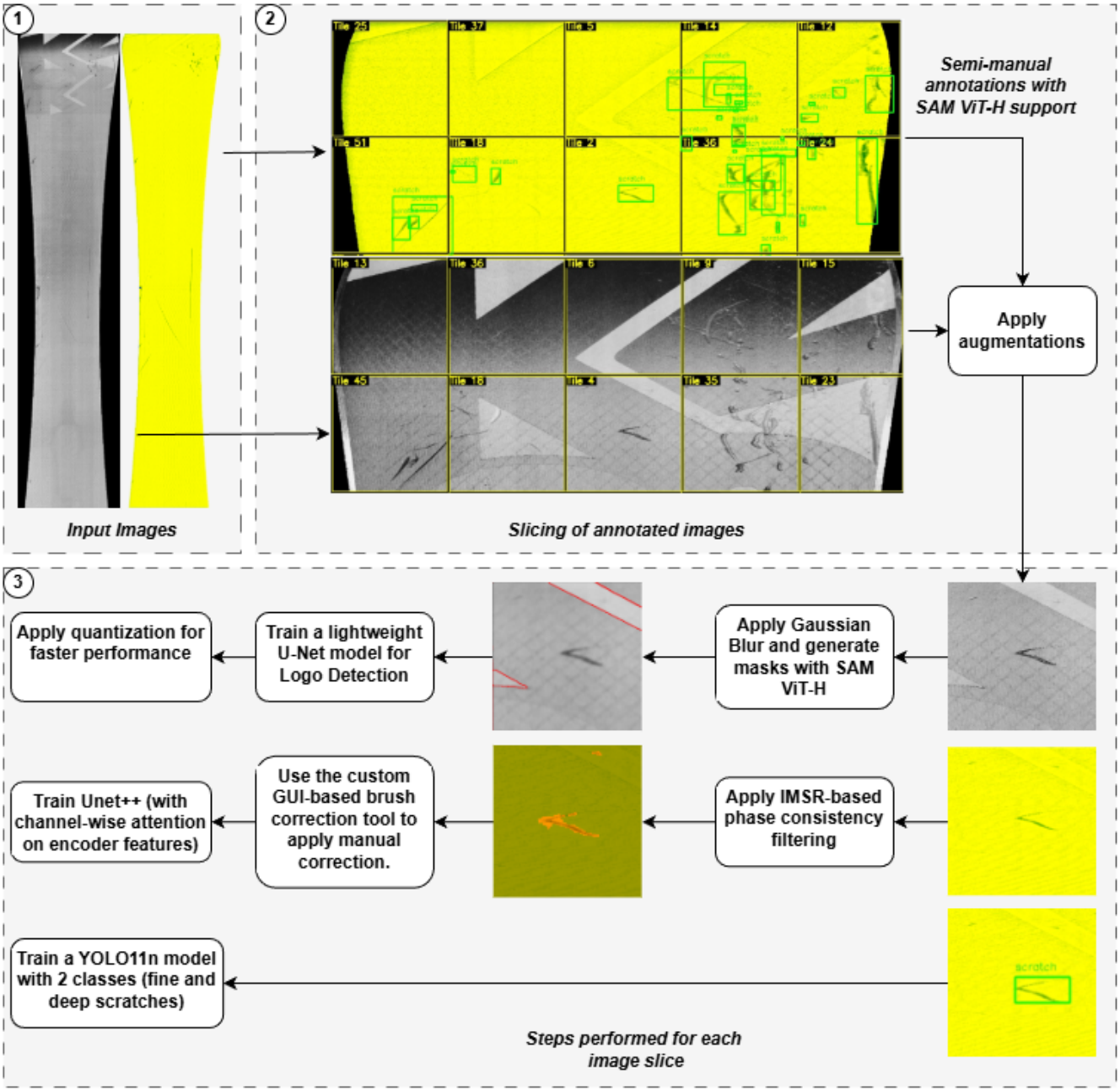

Fig. 2. Data preprocessing and model training used in the final (best fit) approach.

Bringing all steps together, after predicting the scratches with DeepLabV3, we use our fine-tuned SAM model's scratch probability map as a guidance for extracting neighboring clusters of scratch pixels (compact superpixels). We extract only those watershed regions, which are above the meaningful threshold from SAM-based mask ($\tau = 0.02$, which was obtained based on multiple tests with different threshold parameters). The final segmentation mask is a fusion of results obtained from both models.

This approach demonstrated a great performance on new, unseen data. However, the model struggled with performance running without GPU, making it difficult to deploy the model in a time-critical setup with limited hardware capabilities (a CPU-only computer equipped with an AMD Ryzen 5 processor with 6 physical cores and 8 GB of RAM). In addition, after validating the approach on more images, we have noticed that the threshold τ should be continuously adjusted in order to prevent introducing background noise into segmented scratches. In favor of keeping the model available for computation on CPU, we decided to replace the DeepLabV3 with a lighter model (for example, U-Net architecture with a light encoder).

E. Revised YOLO and UNet++ combined with Region Growing

Learning from the previous approaches, we identified the key problems which we wanted to solve in the final approach: (1) Non-uniform scratch structure (varying brightness levels) complicates the accurate and continuous extraction of fine scratches; (2) Industrial setup requires a very light model architecture, which cannot rely solely on GPU.

To address the first problem, we have reviewed our annotated image dataset and YOLO scratch detection model again, and, analyzing per-scratch image contrast values, decided to adjust the annotated image labels, splitting one *scratch* class into two classes of scratches: *fine* and *deep* scratches. Using a min-max normalization, we scaled scratch-background contrast for $s_i \in [0, 1]$ range, labeling scratches $s_i < 0.6$ as fine, while $s_i >= 0.6$ were marked as deep scratches. The newly trained model reached the mAP@0.5 of 0.91 for bounding box prediction for all classes.

We also have reviewed the binary masks for scratches used for segmentation training and followed the practical implications and methodology from [16], where authors applied an Improved Multi-Scale Retinex (IMSR) together with Log-Gabor phase consistency for effective automated extraction of scratches on steel surfaces. Using this classical signal processing approach, we were able to increase information entropy while reducing illumination influence and enhancing the fine, thin structures, producing more accurate binary masks compared to SAM. To ensure the data quality of the GT labels, this time we built a custom GUI-based subjective annotation quality assessment tool, which supported the review process for each label by introducing a brush tool, which enabled the finest and faster corrections compared to LabelMe, which was used previously.

As a model, we preferred UNet++ over U-Net due to its better horizontal and vertical feature aggregation, which provides a wider context for local edge information and is very beneficial in cases of thin and elongated edges. UNet++ also has a lower semantic gap between encoder and decoder layers [33]. For keeping the model light for CPU usage, we use a very lightweight EfficientNet-B0 encoder. We also used channel-wise attention mechanisms, which learn to emphasize scratches with higher weights, suppressing the background noise.

The segmentation model UNet++ was trained on 450 epochs for approximately 4 hours, and the training configuration was as follows: batch size = 64, image size = 256x256, and learning rate = $1e{-}3$. We also applied L2 weight decay as explicit regularization for preventing the overfitting. The newly trained model demonstrated an excellent performance.

Because the current method is very limited to bounding boxes detected in the first place, to capture those parts of scratches which are outside of initial bounding boxes, and ensure prolonged continuity of each scratch, an additional step for improving the precision of the segmentation model was introduced. We applied region growing algorithm (please refer to Fig. 3) to each segmented mask to extend the region, where the average pixel neighborhood for edges had a similar landscape within a certain threshold obtained from local background contrast [34].

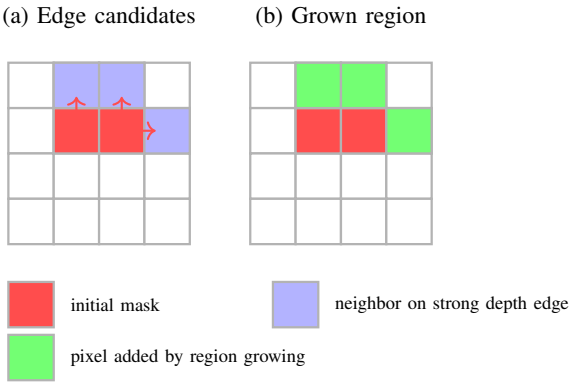

Fig. 3. This is a 4x4 pixel example illustrating the Region Growing algorithm. (a) Scratch mask, detected by UNet++ (shown in red color), and neighboring pixels on depth edges (shown in blue color) are growth candidates. (b) Pixels that satisfy both depth and border constraints are added to the scratch region (shown in green color).

It should also be emphasised that all approaches were evaluated using both 2D grayscale optical images and depth profile images. However, the metallic steel edges on the ski surface produced strong specular reflections in the 2D grayscale images under the acquisition lighting. The models consistently perceived these reflections as surface anomalies, substantially increasing the false positive rate. Therefore, three-dimensional data was used as the primary input modality for the final approaches since it captures geometric surface variation independently of optical reflectance.

Finally, in the Table I we present the comparison of the three main approaches that we have evaluated.

TABLE I
IoU AND DICE SCORES FOR THREE SEGMENTATION METHODS.

Method	IoU (mean)	Dice Score (mean)
Fine-tuned YOLO-based segmentation	0.78	0.87
DeepLabV3 + SAM and Compact Watershed	0.82	0.90
Revised YOLO and UNet++ combined with Region Growing	0.85	0.919

V. Conclusion and Future Work

In our framework, we developed a modular pipeline, which demonstrates that a fusion of classical CV image preprocessing together with deep-learning based defect detection and segmentation can deliver industrial scratch detection without the need for heavy GPU resources for inference. While the proposed methodology and the experience gained may provide a foundation for future research on defect detection in other industrial sectors, the results presented here are specific to the types of polymer structures studied. Further testing must be conducted on various materials and under various industrial

conditions before drawing more general conclusions about the universality of this methodology.

As future work plans, we define:

1) Better adaptation to other types of defects in polymer ski surfaces;
2) Implementing continual learning from new data, avoiding catastrophic forgetting;
3) Development of a separate model for path planning when extracting the centerline for geometrical post-processing.

REFERENCES

[1] R. Ameri, C.-C. Hsu, and S. S. Band, "A systematic review of deep learning approaches for surface defect detection in industrial applications," *Eng. Appl. Artif. Intell.*, vol. 130, no. C, Apr. 2024.

[2] M. Prunella, R. M. Scardigno, D. Buongiorno, A. Brunetti, N. Longo, R. Carli, M. Dotoli, and V. Bevilacqua, "Deep Learning for Automatic Vision-Based Recognition of Industrial Surface Defects: A Survey," *IEEE Access*, vol. 11, pp. 43 370–43 423, 2023.

[3] Q. Qiao, H. Hu, A. Ahmad, and K. Wang, "A Review of Metal Surface Defect Detection Technologies in Industrial Applications," *IEEE Access*, vol. 13, pp. 48 380–48 400, 2025.

[4] Y. Gao, X. Li, X. V. Wang, L. Wang, and L. Gao, "A Review on Recent Advances in Vision-based Defect Recognition towards Industrial Intelligence," *Journal of Manufacturing Systems*, vol. 62, pp. 753–766, Jan. 2022.

[5] L. Yang, F. Zhou, and L. Wang, "A Scratch Detection Method Based on Deep Learning and Image Segmentation," *IEEE Transactions on Instrumentation and Measurement*, vol. 71, pp. 1–12, 2022.

[6] J. Du, C. Tao, X. Cao, and F. Tsung, "3D vision-based anomaly detection in manufacturing: A survey," *Frontiers of Engineering Management*, vol. 12, no. 2, pp. 343–360, Jun. 2025.

[7] S. Ghosh and R. Saha, "A simple and robust algorithm for the detection of multidirectional scratch from digital images," in *2015 Eighth International Conference on Advances in Pattern Recognition (ICAPR)*, Jan. 2015, pp. 1–6.

[8] Q. Jin and L. Chen, "A Survey of Surface Defect Detection of Industrial Products Based on A Small Number of Labeled Data," Mar. 2022.

[9] Z. Xu, Y. Lin, D. Chen, M. Yuan, Y. Zhu, Z. Ai, and Y. Yuan, "Wood broken defect detection with laser profilometer based on Bi-LSTM network," *Expert Systems with Applications*, vol. 242, p. 122789, May 2024.

[10] Y. Yang, X. Zhou, Y. Liu, Z. Hu, and F. Ding, "Wood Defect Detection Based on Depth Extreme Learning Machine," *Applied Sciences*, vol. 10, no. 21, Oct. 2020.

[11] Y. Fang, X. Guo, K. Chen, Z. Zhou, and Q. Ye, "Accurate and automated detection of surface knots on sawn timbers using YOLO-V5 model," in *BioResources*, vol. 16, Jun. 2021, pp. 5390–5406.

[12] T. He, Y. Liu, Y. Yu, Q. Zhao, and Z. Hu, "Application of deep convolutional neural network on feature extraction and detection of wood defects," *Measurement*, vol. 152, p. 107357, Feb. 2020.

[13] B. Syla, S. Saleh, and W. Hardt, "Automated Identification of Wood Surface Defects Based on Deep Learning," *Embedded Selforganising Systems*, vol. 10, no. 7, pp. 55–61, Sep. 2023.

[14] M. T. Çelik, S. Arslankaya, and A. Yildiz, "Real-time detection of plastic part surface defects using deep learning- based object detection model," *Measurement*, vol. 235, p. 114975, Aug. 2024.

[15] M. Roslan, Z. Ibrahim, and Z. Abd Aziz, "Real-Time Plastic Surface Defect Detection Using Deep Learning," May 2022, pp. 111–116.

[16] L. Yang, X. Huang, Y. Ren, and Y. Zhang, "Study on steel plate scratch detection based on improved MSR and phase consistency," *Signal, Image and Video Processing*, vol. 17, no. 1, pp. 119–127, Feb. 2023.

[17] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You Only Look Once: Unified, Real-Time Object Detection," May 2016.

[18] University, "crack dataset," dec 2022, visited on 2024-01-23. [Online]. Available: https://universe.roboflow.com/university-bswxt/crack-bphdr

[19] S. Ruan, C. Zhan, B. Liu, Q. Wan, and K. Song, "A high precision YOLO model for surface defect detection based on PyConv and CISBA," *Scientific Reports*, vol. 15, no. 1, p. 15841, May 2025.

[20] L. Cui, X. Jiang, M. Xu, W. Li, P. Lv, and B. Zhou, "SDDNet: A Fast and Accurate Network for Surface Defect Detection," *IEEE Transactions on Instrumentation and Measurement*, vol. 70, pp. 1–13, 2021.

[21] O. Ronneberger, P. Fischer, and T. Brox, "U-Net: Convolutional Networks for Biomedical Image Segmentation," May 2015.

[22] R. P. Vengaloor and R. Muralidhar, "Deep Learning Based Semantic Segmentation Technique for Anomaly Detection on Metal Surfaces Using High Calibre U- Shaped Network," *Traitement du Signal*, vol. 39, no. 6, pp. 2023–2031, Dec. 2022.

[23] H. Chen and B.-W. Min, "A high-precision segmentation network for industrial surface defect detection," *AIP Advances*, vol. 15, no. 5, p. 055118, May 2025.

[24] K. He, G. Gkioxari, P. Dollár, and R. Girshick, "Mask R-CNN," Jan. 2018.

[25] F. Yang, J. Huo, Z. Cheng, H. Chen, and Y. Shi, "An Improved Mask R-CNN Micro-Crack Detection Model for the Surface of Metal Structural Parts," *Sensors (Basel, Switzerland)*, vol. 24, no. 1, p. 62, Dec. 2023.

[26] L.-C. Chen, Y. Zhu, G. Papandreou, F. Schroff, and H. Adam, "Encoder-Decoder with Atrous Separable Convolution for Semantic Image Segmentation," Aug. 2018.

[27] H. Zhang, Z. Zhao, Y. Liu, J. Liu, T. Ma, K. Wu, Z. Zhuang, and J. Wang, "Steel surface defect segmentation with SME-DeeplabV3+," *PLOS ONE*, vol. 20, no. 8, p. e0329628, Aug. 2025.

[28] B. C. Russell, A. Torralba, K. P. Murphy, and W. T. Freeman, "LabelMe: A Database and Web-Based Tool for Image Annotation," *International Journal of Computer Vision*, vol. 77, no. 1-3, pp. 157–173, May 2008.

[29] A. Kirillov, E. Mintun, N. Ravi, H. Mao, C. Rolland, L. Gustafson, T. Xiao, S. Whitehead, A. C. Berg, W.-Y. Lo, P. Dollár, and R. Girshick, "Segment Anything," in *2023 IEEE/CVF International Conference on Computer Vision (ICCV)*. Paris, France: IEEE, Oct. 2023, pp. 3992–4003.

[30] S. Kulkarni, S. Singh, D. Balakrishnan, S. Sharma, S. Devunuri, and S. C. R. Korlapati, "CrackSeg9k: A Collection and Benchmark for Crack Segmentation Datasets and Frameworks," Aug. 2022.

[31] C. Liu and B. Xu, "Weakly-supervised structural surface crack detection algorithm based on class activation map and superpixel segmentation," *Advances in Bridge Engineering*, vol. 4, no. 1, p. 27, Nov. 2023.

[32] P. Neubert and P. Protzel, "Compact Watershed and Preemptive SLIC: On Improving Trade-offs of Superpixel Segmentation Algorithms," in *2014 22nd International Conference on Pattern Recognition*. Stockholm, Sweden: IEEE, Aug. 2014, pp. 996–1001.

[33] Z. Zhou, M. M. R. Siddiquee, N. Tajbakhsh, and J. Liang, "UNet++: A Nested U-Net Architecture for Medical Image Segmentation," Jul. 2018.

[34] K. S. Fu and J. K. Mui, "A survey on image segmentation," *Pattern Recognition*, vol. 13, no. 1, pp. 3–16, Jan. 1981.