Available online at www.sciencedirect.com

ScienceDirect

Procedia CIRP 94 (2020) 469–473

www.elsevier.com/locate/procedia

11th CIRP Conference on Photonic Technologies [LANE 2020] on September 7-10, 2020

Image-based roughness estimation of laser cut edges with a convolutional neural network

Leonie Tatzel[a,b,*] , Fernando Puente León[a]

[a]*Karlsruhe Institute of Technology: Institute of Industrial Information Technology, Hertzstr. 16, Bldg. 06.35, 76187 Karlsruhe, Germany*
[b]*TRUMPF Werkzeugmaschinen GmbH + Co. KG, Johann-Maus-Str. 2, 71254 Ditzingen, Germany*

* Corresponding author. Tel.: +49 7156 303-33775. *E-mail address:* Leonie.Tatzel@kit.edu

Abstract

Laser cutting of metals is a complex process with many influencing factors. As some of them are subject to change, the cut quality needs to be checked regularly. This paper aims to estimate the roughness of cut edges based on RGB images instead of surface topography measurements.

We trained a convolutional neural network (CNN) on a broad database of images and corresponding roughness values. The CNN estimates the roughness well with a mean error of 3.6 µm. Sometimes it is more reliable than the surface measuring device because the RGB images are less prone to reflectivity problems than the measurements.

Keywords: laser cutting; cut edge quality; roughness, convolutional neural network

1. Introduction

Laser cutting is a complex process that depends on numerous parameters: properties of the laser beam (e.g. focus position), transport properties (e.g. feed rate), properties of the assist gas (e.g. gas pressure) and material properties (e.g. quality of the sheet) [1]. Even the change of a single parameter can cause a deterioration of the process quality, which therefore needs to be closely monitored. Despite decades of industrial application laser cutting is still the subject of current research. Arntz-Schroeder and Petring [2] and Pocorni et al. [3] recently presented analyses of the melt flow dynamics at the cut front. These dynamics influence the roughness of the resulting cut edge.

According to the ISO standard 9013 [4] roughness (*Rz*) and perpendicularity tolerance (*u*) define the quality of a thermal cut. Both are derived from the height profile of the cut edge. Consequently, a surface measuring device is required to determine them. This is usually not available in production. Instead, the machine operator carries out the quality inspection manually by simply looking at the edge. This method is subjective and difficult to automate. It would be better to complement or replace this procedure with an objective quality assessment.

This paper presents a contact-less approach to deduce the roughness of a cut edge from an RGB image (image with three additive colour channels, R: red, G: green, B: blue) without a measuring device. For this purpose, we generated a broad database containing cut edges of very different qualities: for each edge an RGB image was taken and *Rz* was determined. This data was used to train a convolutional neural network (CNN). In section 4 we evaluate the performance of the model in detail. The CNN sometimes estimates the roughness better than the actual measurement because the RGB images are less prone to reflectivity problems than the surface topographies.

Sun et al. [5] describe a CNN-based approach to evaluate the surface of milled metals. On milled surfaces grooves are clearly visible and can be easily identified. This is not the case

10.1016/j.procir.2020.09.166

for parts produced with a laser. Stahl and Jauch [6] present a CNN to estimate the roughness of a cut edge. They used *Ra* instead of *Rz* to describe the roughness and they labelled each edge with the mean value of all measuring lines. The average roughness, however, says little about the maximum roughness, which can occur very locally. Since they did not ensure that the images of one edge (that look very similar and have the same label) are either in the training or in the test split, they might overestimate the performance of the model.

2. Database

2.1. Generating edges of very different qualities

The database consists of 3336 stainless steel (1.4301) cut edges. All of them are 3 mm thick and were cut on a state-of-the-art laser cutting machine (TruLaser 5030 fiber with TruDisk 12001). To generate different cut edges four dominant process parameters were varied: the distance between the nozzle and the focus of the laser beam (−3.5 to −0.5 mm), the distance between the nozzle and the sheet metal (0.5 to 3 mm), the feed rate (13 to 29 m/min) and the pressure of the assist gas (9 to 21 bar). The fully factorial combination resulted in 1050 different parameter combinations and (less the miscuts) in 834 unique samples. Each sample is a square with a side length of 10 cm. In the following only the middle 5 cm of each edge were considered as the process parameters are constant only there.

2.2. Roughness measurement

The roughness was determined with an optical measurement system (3D profilometer VR-3200, Keyence Corporation) using light section. Light section is based on triangulation [7]. A thin line is projected onto the object to be measured and the projection is observed by a camera. Displacements of the line can be converted into 3D point clouds. The accuracy of the height measurement is ±3 µm.

The mean height of the profile *Rz* (here: roughness) indicates the absolute vertical distance between the highest profile peak and the deepest profile valley along the sampling length. Due to the dependency on the extreme values of the profile *Rz* is strongly influenced by local outliers.

The roughness values were calculated as follows: the 5 cm of the edge were divided into five areas of 10 mm length. In each area nine measurement lines were placed at different depths of the sheet with 0.3 mm spacing. This is shown in Fig. 1.

The lines are named after their distance (in mm) from the upper edge. For each of the measurement lines, the roughness was determined following the ISO standard [4] (with one minor change: we applied a fixed cut-off wavelength of 2.5 mm).

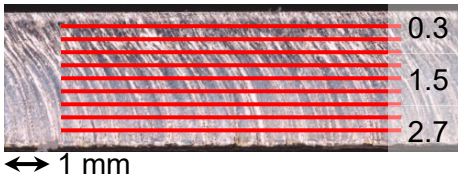

Fig. 1. RGB image of 10 mm (one area) of a laser cut edge with roughness measurement lines at different depths of the sheet.

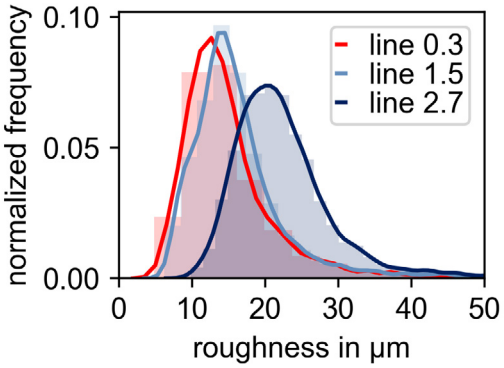

Fig. 2. Distribution of the roughness values for the lines 0.3, 1.5 and 2.7. The x-axis was limited to 50 µm. Less than 2 % of the edges have higher roughness values.

The *Rz* values of the five areas were then averaged. This resulted in nine regression targets (labels) for each edge: one value for line 0.3, one value for line 0.6, etc.

In Fig. 2 the roughness histograms of line 0.3, 1.5 and 2.7 are shown. The corresponding expected values are 15.8, 17.0 and 22.6 µm and the standard deviations are 10.4, 9.1 and 7.3 µm respectively.

According to the ISO standard [4] roughness and perpendicularity tolerance define the quality of a laser cut edge. The perpendicularity could probably be estimated analogously to the presented approach, though images taken from a different perspective would be needed.

2.3. RGB images of the cut edges

The measurement device was also used to take the RGB images. It is equipped with a double telecentric lens, a monochromatic CMOS Sensor and red, green and blue LEDs. The raw images have a size of about 3400 x 530 pixels and a resolution of 15 µm x 15 µm per pixel.

3. Model and methods

3.1. Convolutional neural networks

Convolutional neural networks (CNNs) are artificial neural networks that are mainly used in the field of computer vision. They are designed to extract local (image) features invariant to their location by using shared parameters and local receptive fields. Shared parameters lead to a reduced total number of parameters, which allows to increase the number of layers compared to a neural network with full connectivity [8].

The core building block of a CNN is the convolutional layer. Each of these layers applies $K \in \mathbb{N}$ different filter kernels resulting in K feature maps. Most modern CNNs (e.g. ResNet [9]) reduce the spatial size of the layers and increase the depth dimension (number of kernels) of the feature maps.

The architecture of the CNN used in this paper is shown in Fig. 3 and explained in more detail in Table 1. It was inspired by the VGG16 network [10], but with only 1,974,441 trainable parameters it is significantly smaller.

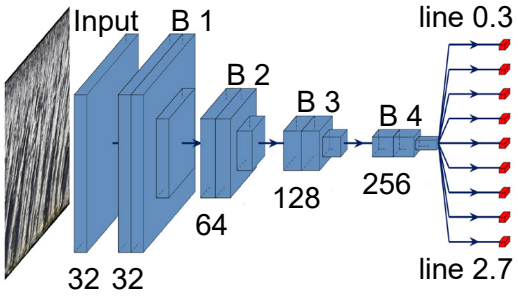

Fig. 3. CNN architecture: one input layer and four blocks (B 1, B 2, B 3 and B 4) each containing three convolutional layers. Input: square image of the cut edge. Output: nine *Rz* values, one for each measurement line.

The CNN consists of 13 convolutional layers, which are organized in blocks. The input block contains one convolutional layer, the subsequent four blocks contain three convolutional layers each, where the last layer of the block serves as spatial downsampling operation (convolution with stride 2, [11]). The number of filters increases from 32 to 256 while the dimension decreases from 112 x 112 to 7 x 7 pixels. Each convolutional layer uses a kernel size of 3 x 3 pixels (except for the input block which applies 7 x 7 kernels). After each convolutional layer batch normalization is performed [12] and the ReLu activation function [13] is applied. To connect the last block with the nine regression targets, we use global average pooling (GAP) [14] in combination with one dense layer.

Table 1. CNN architecture: The model consists of an input block and four subsequent blocks each containing three convolutional layers; the regression targets are connected with global average pooling and one dense layer.

layer	number of kernels	output spatial resolution
input block, conv 1	32	112 x 112
block 1, conv 1	32	112 x 112
block 1, conv 2	32	112 x 112
block 1, conv 3	32	56 x 56
block 2, conv 1	64	56 x 56
block 2, conv 2	64	56 x 56
block 2, conv 3	64	28 x 28
block 3, conv 1	128	28 x 28
block 3, conv 2	128	28 x 28
block 3, conv 3	128	14 x 14
block 4, conv 1	256	14 x 14
block 4, conv 2	256	14 x 14
block 4, conv 3	256	7 x 7
GAP	-	256 x 1
dropout	-	256 x 1
dense	-	9 x 1

3.2. Image preprocessing

The preprocessing of the raw images is displayed in Fig. 4. Since they contain a lot of black background, they were first cropped. This resulted in an image of approximately 215 pixels height (~3 mm) and 3,400 pixels length (~50 mm), which was then cut into 15 square images. Each of them was slightly rescaled to 224 x 224 pixels. The 15 images of one edge were labeled with the same roughness values.

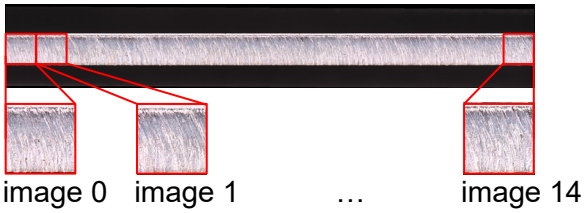

Fig. 4. The raw image was cropped and cut into 15 square images.

3.3. Train, validation and test split

The database consists of 834 samples, which correspond to 3,336 edges and 50,040 images. It was split into three parts: training set (72 %), validation set (13 %) and test set (15 %). The training set was used to learn the model parameters, the validation set was necessary for the hyperparameter optimization and with the test set the performance of the model is evaluated on unseen data.

When splitting the data, it must be guaranteed that a particular parameter combination (all images of one sample) is only contained in one of the three sets instead of splitting the images randomly. Otherwise we might overestimate the performance of the CNN: instead of generalizing well, it might only memorize the images.

3.4. Training

Since pre-trained architectures (e.g. ResNet [9] and VGG16 [10], pretrained on the ImageNet database) did not extract relevant features, we trained the CNN from scratch. The weights of the network were initialized with the Xavier uniform initializer [15] and updated with the Adam algorithm (β_1=0.9, β_2=0.999) [16]. The learning rate was set to 0.001. The batch size was 60.

To prevent overfitting during training we used 50 % dropout, data augmentation (vertically and horizontally flipping and rotating the images) and validation-based early stopping: when the validation loss does not improve for 20 epochs the training is aborted.

3.5. Evaluation

The mean absolute error (MAE) was used as loss function during training and to evaluate the quality of the regression on the test set. It was calculated for each of the nine targets separately:

$$MAE = \frac{1}{N}\sum_{i=1}^{N}\left|\hat{Y}_i - Y_i\right|$$

with N: number of test images, $\hat{Y}_i$: predicted value, Y_i: true value (label) for the image i.

In order to obtain a more stable statement about the model performance, the data was divided into training, validation and test set three times with a random permutation cross-validator

(in compliance with the boundary conditions described in 3.3). These splits are referred to as run 1, 2 and 3.

To the best of our knowledge there is no analytical method (for the calculation of *Rz* based on RGB images) to which the CNN could be compared.

3.5.1. Python libraries and hardware

The CNN was implemented in Python using the libraries TensorFlow 1.13.1 [17] and Keras 2.2.4 [18]. It was trained on the GPU GeForce RTX 2080 Ti graphical processing unit (NVIDIA corporation). The training time was below eight hours and is not considered further.

4. Results and discussion

In Table 2 the MAEs for the different measurement lines are shown. They range between 2.7 µm and 5.7 µm. For the same line they vary by up to 1.4 µm for different runs. The fact that the performance depends on the partitioning of the data is probably caused by the small database. The MAEs vary for different measurement lines. The prediction is generally better for lines in the lower region of the cut edge. The MAEs of line 0.3 are by far the worst for all three runs.

Table 2. Mean absolute errors (rounded) of the nine regression targets (line 0.3 to line 2.7) on the respective test set for three random splits of the data.

line		0.3	0.6	0.9	1.2	1.5
MAE in µm	run 1	5.7	3.7	3.8	2.9	2.9
	run 2	4.5	4.2	4.3	4.0	3.2
	run 3	4.3	3.6	4.2	3.5	3.4
	Ø	4.8	3.9	4.1	3.4	3.2
line		1.8	2.1	2.4	2.7	Ø
MAE in µm	run 1	3.4	3.0	2.9	2.9	3.5
	run 2	2.7	2.8	2.8	3.4	3.5
	run 3	3.5	3.4	3.6	3.6	3.7
	Ø	3.2	3.1	3.1	3.3	3.6

In Fig. 5 the roughness values predicted by the CNN on the test set of run 1 are compared with the labels (measurement results) for the lines 0.3, 1.5 and 2.7. Prediction and label are mostly similar at the middle and bottom line. The model is slightly biased: for low roughness values the predicted value is generally a little too high, for high roughness values it is too low. This is probably caused by the distribution of the data (see Fig. 2).

Line 0.3 stands out negatively again in Fig. 5. For some samples the prediction of the CNN seems to be completely wrong. It estimates roughness values of less than 25 µm for samples with labels of more than 50 µm. In truth, the higher MAEs in Table 2 and the huge differences between prediction and label are due to the quality of the data. In rare cases, the optical measurement device cannot cope with the high reflectivity of the stainless steel surfaces. This results in faulty surface topography measurements with outliers (deep holes) and causes very large roughness values. The uppermost part of the cut edge (line 0.3 to 0.9) is most affected because the reflectivity is particularly high there.

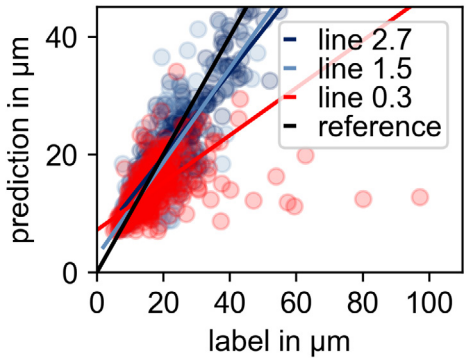

Fig. 5. Comparison of predicted value and label for the lines 0.3, 1.5 and 2.7, exemplarily for run 1.

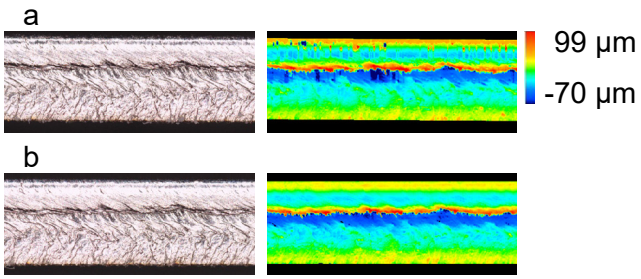

Fig. 6. RGB image and height topography of two measurements of edge 0700-4, a: with measurement errors (deep holes) caused by reflections, b: without errors.

For example, edge 0700-4 (see Fig. 5, rightmost red dot) is labelled with a value of 97 µm. The high *Rz* value is mainly caused by small, but very deep holes (see Fig. 6, a). The repetition of the measurement with a slightly deviating positioning of the edge in the sample holder, produces a surface topography without any holes (see Fig. 6, b). In this case *Rz* is 12 µm instead of 97 µm. The CNN estimates a value of 13 µm. It follows that the label is wrong. The CNN estimates the roughness better than the measurement system. The same applies to the other outliers in Fig. 5.

The neural network correctly maps the relationship between RGB image and roughness. Prediction and label only differ greatly, if the label is wrong. Although the outliers only affect few edges, they are not negligible due to their magnitude. They do not only increase the test error, but also the training error and influence the training process negatively. As the loss is averaged over the nine regression targets during training, the outliers affect the quality of the overall regression. A reliable detection and elimination of the outliers would surely improve the model, but not all outliers are as easy to detect as the one in the example above and repeating the measurement of each edge several times is too time-consuming.

5. Conclusion and outlook

We have shown that it is possible to estimate the roughness *Rz* of a laser cut edge with a CNN based on an RGB image. The roughness at different depths of the sheet could be determined with a mean error between 4.8 (for line 0.3) and 3.2 µm (for line 1.5). Consequently, a 3D measurement device is only

needed to generate training data for the model and obsolete in practice. Systems that automatically sort laser cut parts could for example be supplemented with cameras that take images of the cut edges or separate photo stations could be used by the machine operators when necessary.

The CNN performs only apparently worse for the uppermost measurement line. In truth, some labels are wrong due to measurement errors and in these cases the prediction of the CNN is better than the actual measurement. By comparing the errors of the different runs, it becomes clear that improvements could be made, if more or better data was available. Then the performance would be less dependent on the split of the data and the outliers would have less impact on the overall performance. The data quality could be improved by using a mechanical instead of an optical measuring method (e.g. stylus tip measuring device). However, this would make data collection even more time consuming.

Further interesting steps include the application of the network to other sheet thicknesses and materials. In addition, it would be desirable to estimate the roughness based on RGB images with a poorer resolution that are taken by simpler cameras without double telecentric lenses.

Acknowledgements

Part of the work has been performed within the SeHer research project, which is funded by the German Federal Ministry of Education and Research. The authors wish to express their sincere gratitude for the support.

References

[1] Steen WM, Mazumder J. Laser Material Processing, 4. ed. London: Springer, 2010.
[2] Arntz-Schroeder D, Petring D. Analyzing the Dynamics of the Laser Beam Cutting Process: An explanation of characteristic frequencies of melt film dynamics. PhotonicsViews 2020; 17: 43–7.
[3] Pocorni J, Powell J, Deichsel E, Frostevarg J, Kaplan AFH. Fibre laser cutting stainless steel: Fluid dynamics and cut front morphology. Optics & Laser Technology 2017; 87: 87–93.
[4] DIN ISO 9013. Thermal cutting - Classification of thermal cuts - Geometrical product specification and quality tolerances. 2017.
[5] Sun W, Yao B, Chen B et al. Noncontact Surface Roughness Estimation Using 2D Complex Wavelet Enhanced ResNet for Intelligent Evaluation of Milled Metal Surface Quality. Applied Sciences 2018; 8: 381.
[6] Stahl J, Jauch C. Quick roughness evaluation of cut edges using a convolutional neural network. In: Fourteenth International Conference on Quality Control by Artificial Vision. Munich, Germany, 2019.
[7] Leach R, ed. Optical Measurement of Surface Topography. Berlin: Springer, 2011.
[8] LeCun Y, Bengio Y, Hinton G. Deep learning. Nature 2015; 521: 436–44.
[9] He K, Zhang X, Ren S, Sun J. Deep Residual Learning for Image Recognition. In: 2016 IEEE Conference on Computer Vision and Pattern Recognition. Las Vegas, USA, 2016.
[10] Simonyan K, Zisserman A. Very Deep Convolutional Networks for Large-Scale Image Recognition. In: 3rd International Conference on Learning Representations. San Diego, USA, 2015.
[11] Springenberg JT, Dosovitskiy A, Brox T, Riedmiller M. Striving for Simplicity: The All Convolutional Net. In: 3rd International Conference on Learning Representations. San Diego, USA, 2015.
[12] Ioffe S, Szegedy C. Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift. In: Proceedings of the 32 Nd International Conference on Machine Learning. Lille, France, 2015.
[13] Nair V, Hinton GE. Rectified Linear Units Improve Restricted Boltzmann Machines. In: Proceedings of the 27th International Conference on International Conference on Machine Learning. Madison, USA, 2010: 807–814.
[14] Lin M, Chen Q, Yan S. Network In Network. In: 2nd International Conference on Learning Representations. Banff, Canada, 2014.
[15] Glorot X, Bengio Y. Understanding the difficulty of training deep feedforward neural networks. In: Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics. Sardinia, Italy, 2010.
[16] Ba J, Kingma D. Adam: A Method for Stocastic Optimization. In: 3rd International Conference for Learning Representations. San Diego, USA, 2015.
[17] Abadi M, Agarwal A, Barham P et al. TensorFlow: Large-Scale Machine Learning on Heterogeneous Distributed Systems. arXiv preprint, 2016.
[18] Chollet F. Keras. Keras Documentation, 2015. https://keras.io (accessed February 26, 2020).