XI. Identification and Quantification of Wear on Gear Cutting Tools Using Computer Vision Methods –

Emil-Elias Breuer, M.Eng. (WZL)

Gears are typically soft-machined using manufacturing processes with a geometrically defined cutting edge, with the objective of balancing workpiece quality and manufacturing costs by controlling tool wear. Tool wear can be assessed either directly, using optical sensors, or indirectly by analyzing acceleration or acoustic signals. While optical measurements offer higher accuracy, there is also uncertainty due to human interpretation of the tool wear images. As a result, recent years have seen efforts to use computer vision to detect and quantify tool wear.

In this report, an algorithm is developed to detect and quantify tool wear on gear hobbing and bevel gear cutting tools using computer vision methods. Using U-Net with EfficientNet as the backbone, the wear was detected with an Intersection over Union (IoU) score of 0.65 and a loss of 0.05 on the training data and an IoU score of 0.64 and a loss of 0.06 on the validation data. The algorithm post-processes the identified wear areas using traditional computer vision methods such as k-means clustering, convex hull, binary thresholding, and dilation to obtain a closed contour of the wear area. Finally, to ensure the performance of the algorithm, a manually measured wear curve and an algorithm-generated wear curve from bevel gear cutting and hobbing wear trials are compared. In general, good agreement is achieved between the algorithm-generated and manually measured wear curves with an average absolute difference of ΔVB_{max} = 7.99 µm for bevel cutting and ΔVB_{max} = 19.95 µm for hobbing.

Identification and Quantification of Wear on Gear Cutting Tools Using Computer Vision Methods

Speaker: Emil-Elias Breuer M.Eng.
Author: Melina Kamratowski M.Sc.

WZL of RWTH Aachen
Prof. Dr.-Ing. Christian Brecher
MTI of RWTH Aachen
Prof. Dr.-Ing. Thomas Bergs
Dr.-Ing. Mareike Davidovic

The Brazilian Gear Conference ITA-WZL 2025, August 13th/ 14th 2025

Outline

Motivation and Objective
Challenges in Wear Measurement

Measurement Time

Wear Parameters

Resource Efficiency

Subjectivity

Motivation and Objective
Identification and Quantification of Tool Wear with Computer Vision Methods

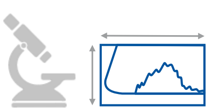
Objective

Identification and Quantification of Wear on Gear Cutting Tools Using Computer Vision Methods

Approach

Data Preparation

- Capturing images of worn tools
- Transformation, normalization and scaling of images

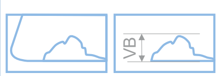


Comparison of Methods

- Implementation of traditional and deep learning based methods
- Comparison of the suitability of the methods

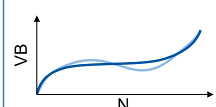
Implementation

- Algorithm for identifying tool wear
- Algorithm for quantifying tool wear



Application

- Generation of wear curves for gear hobbing and bevel gear cutting
- Comparison of manually measured and algorithm generated wear curves



Outline

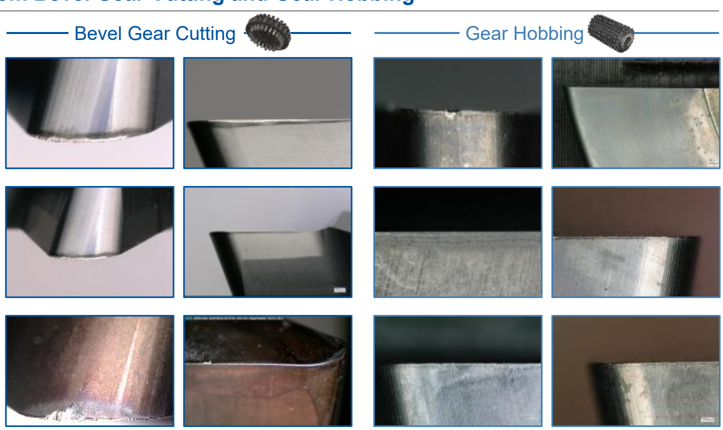
Data Set and Data Preparation

Images of Worn Tools from Bevel Gear Cutting and Gear Hobbing

Bevel Gear Cutting
Process
– Single Blade Group Trial
– Face Milling and Hobbing
– Plunging
Tool
– Substrate: K30
– Coating: AlCrN, AlTiN

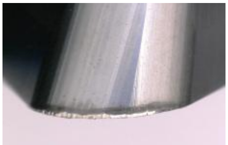
Gear Hobbing
Process
– Fly-Cutting Trial
– Dry
Tool
– Substrate: K30, S390, MC90
– Coating: AlCrN

Number of Images
Bevel Gear Cutting: 594 Images
Gear Hobbing: 646 Images



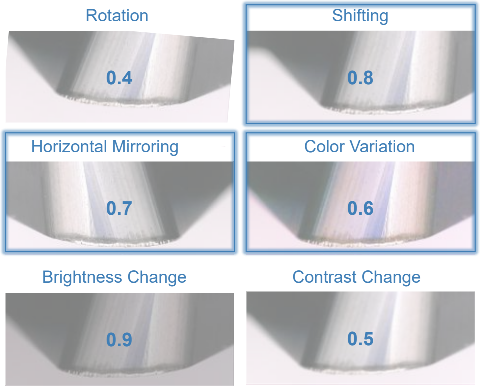
Transformation of Images of Worn Tools

Original Image



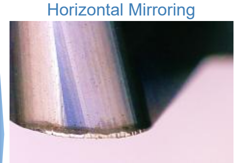
Number of Images
Bevel Gear Cutting:
594 Images
Gear Hobbing:
646 Images

Transformations



Transformed Image

Color Variation,
Shifting,
Horizontal Mirroring



Number of Images
Bevel Gear Cutting:
5940 Images
Gear Hobbing:
6460 Images

Outline

1	Motivation and Objective
2	Data Set and Data Preparation
3	**Comparison of Traditional and Deep Learning Based Computer Vision Methods**
4	Algorithm for Identifcation and Quantification of Tool Wear
5	Comparison of Manually Measured and Algorithm Generated Wear Curves
6	Summary and Outlook

Comparison of Traditional and Deep Learning Based Computer Vision Methods
Traditional Methods

Use Case
Gear Hobbing
Substrate: K30
Coating: AlCrN

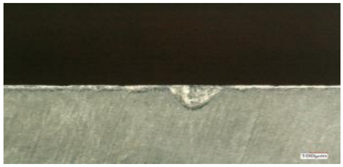
Binary Thresholding / Edge Detection
Lower Threshold = 160
Upper Threshold = 235

Contour Detection
Line Thickness = 2

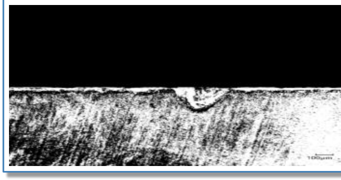
Computer
Intel® Core™ i9-10900K
Processor: 3.70 GHz; 10 Cores
Memory: 64 GB

Software
Python 3.11.4

Reference



Binary Thresholding



Edge Detection

Comparison of Traditional and Deep Learning Based Computer Vision Methods
Traditional Methods – Mode of Operation of Edge Detection

Input – Binary Thresholding

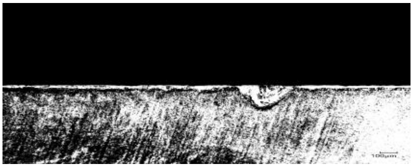
Mode of Operation

55	84	68	31	71	112	141	5	111	128
124	145	234	211	12	220	193	212	187	238
33	238	20	155	25	146	71	181	140	86
140	187	5	101	147	56	174	165	156	90
192	165	98	238	211	48	168	222	189	51
25	245	10	44	197	125	42	176	66	65
72	35	215	217	126	114	31	245	148	158
140	37	137	9	4	165	128	16	65	121

Lower Threshold 160
Upper Threshold 210

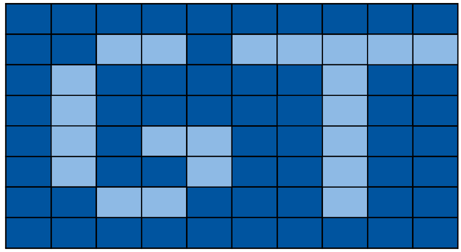
Traditional Methods – Mode of Operation of Edge Detection

Input – Binary Thresholding



Output – Edge Detection

Mode of Operation



Lower Threshold 160
Upper Threshold 210

Traditional Methods

Use Case
Gear Hobbing
Substrate: K30
Coating: AlCrN

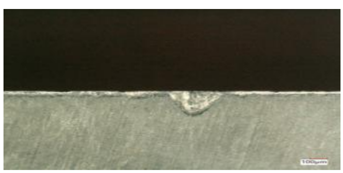
Binary Thresholding / Edge Detection
Lower Threshold = 160
Upper Threshold = 235

Contour Detection
Line Thickness = 2

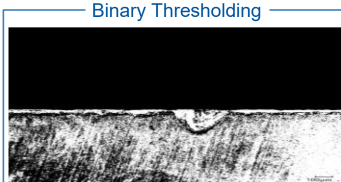
Computer
Intel® Core™ i9-10900K
Processor: 3.70 GHz; 10 Cores
Memory: 64 GB

Software
Python 3.11.4

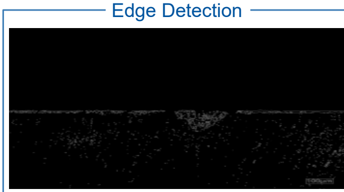
Reference



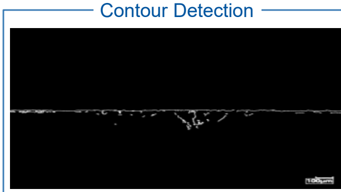
Binary Thresholding



Edge Detection



Contour Detection



Comparison of Traditional and Deep Learning Based Computer Vision Methods
Deep Learning Based Methods

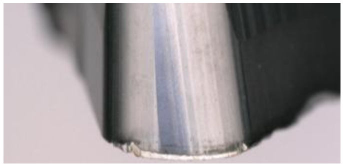
Use Case
Bevel Gear Cutting
Substrate: K30
Coating: AlCrN

Training
Batch Size = 8
Learning Rate = 0.01
Epochs = 10
Training Data = 992 Images
Test Data = 124 Images
Validation Data = 124 Images

Computer
Intel® Core™ i9-10900K
Processor: 3.70 GHz; 10 Cores
Memory: 64 GB

Software
Python 3.11.4
LabelMe 5.4.0

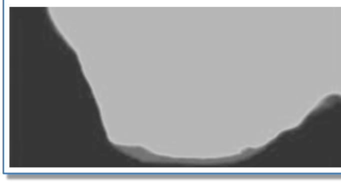
Reference



Labelled Reference

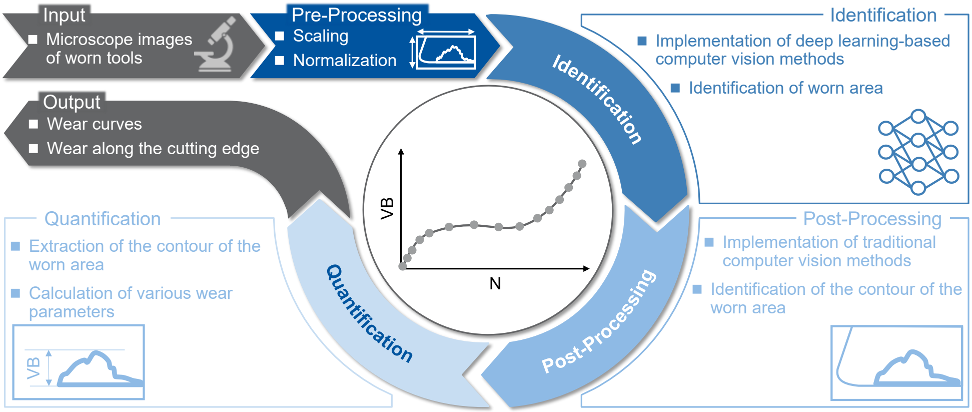
ResNet

U-Net and EfficientNet

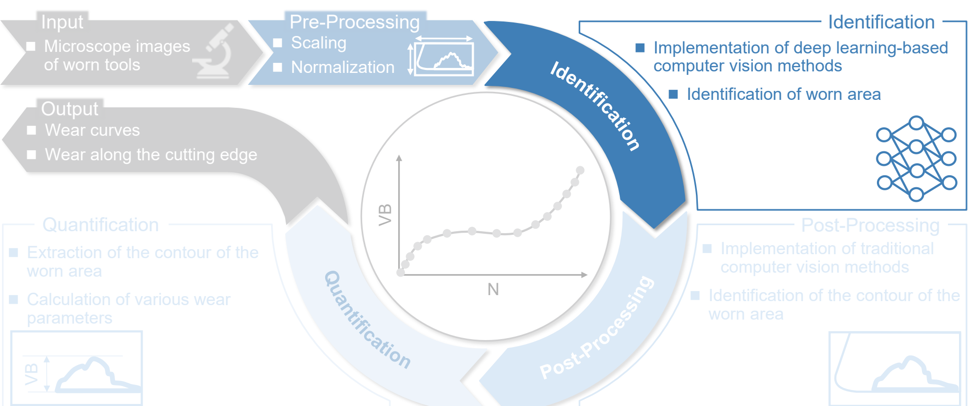


Outline

Algorithm for Identification and Quantification of Tool Wear
Procedure of the Algorithm



Algorithm for Identification and Quantification of Tool Wear
Procedure of the Algorithm – Identification



Algorithm for Identification and Quantification of Tool Wear
Principle of Training the Neural Network

Data
Training = 9,920 Images
Test = 1,240 Images
Validation = 1,240 Images

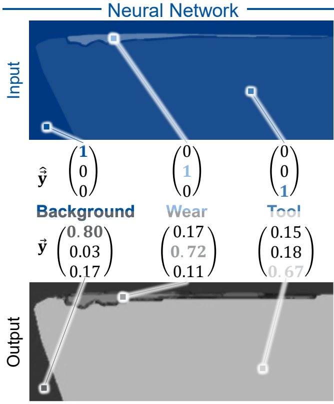
Neural Network
U-NET with EFFICIENTNET

Hyperparameter
Batchsize = 8
Learning Rate = 0.001
Epochs = 60

Computer
RWTH High Performance Cluster
Intel ® HNS2600BPB
Memory: 32 GB

Software
Python 3.11.4
LabelMe 5.4.0

Neural Network



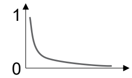
$$\hat{\vec{y}} \quad \begin{pmatrix}1\\0\\0\end{pmatrix} \quad \begin{pmatrix}0\\1\\0\end{pmatrix} \quad \begin{pmatrix}0\\0\\1\end{pmatrix}$$

$$\vec{y} \quad \begin{pmatrix}0.80\\0.03\\0.17\end{pmatrix} \quad \begin{pmatrix}0.17\\0.72\\0.11\end{pmatrix} \quad \begin{pmatrix}0.15\\0.18\\0.67\end{pmatrix}$$

Loss

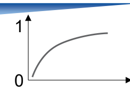
Input: $\hat{\vec{y}}_i = \begin{pmatrix}0\\1\\0\end{pmatrix}$ Output: $\vec{y}_i = \begin{pmatrix}0.17\\0.72\\0.11\end{pmatrix}$

$$\text{Loss} = -\sum_{i=0}^{n} \hat{\vec{y}}_i \cdot \log(\vec{y}_i)$$



Intersection over Union (IoU)

$$\text{IoU} = \frac{\textbf{Input} \cap \textbf{Output}}{\textbf{Input} \cup \textbf{Output}}$$



Algorithm for Identification and Quantification of Tool Wear
Result of Training the Neural Network

Data
Training = 9,920 Images
Test = 1,240 Images
Validation = 1,240 Images

Neural Network
U-NET with EFFICIENTNET

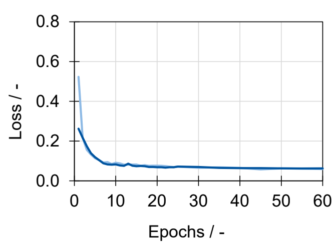
Hyperparameter
Batchsize = 8
Learning Rate = 0.001
Epochs = 60

Computer
RWTH High Performance Cluster
Intel ® HNS2600BPB
Memory: 32 GB

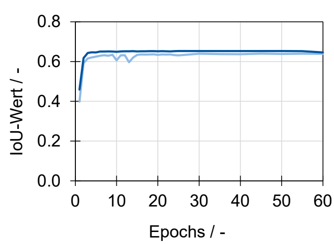
Software
Python 3.11.4

— Training Data
— Validation Data

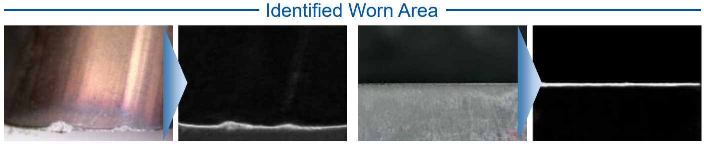
Loss



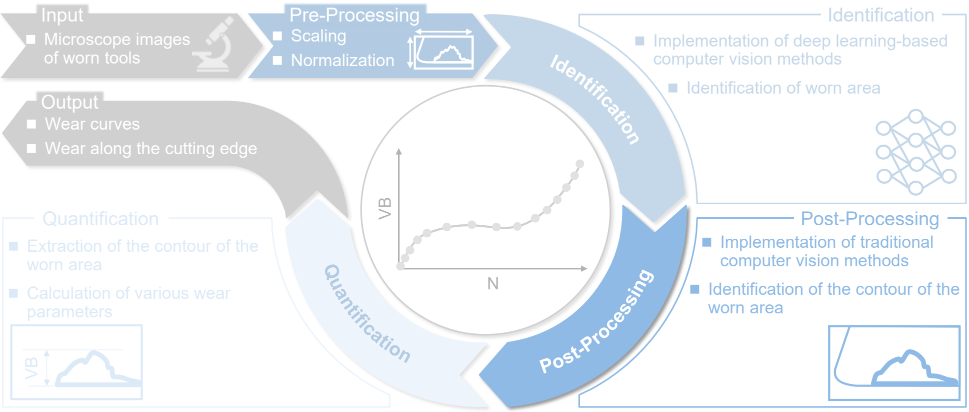
IoU-Wert



Identified Worn Area



Algorithm for Identification and Quantification of Tool Wear
Procedure of the Algorithm – Post-Processing



Algorithm for Identification and Quantification of Tool Wear
Post-Processing of the Identified Worn Area

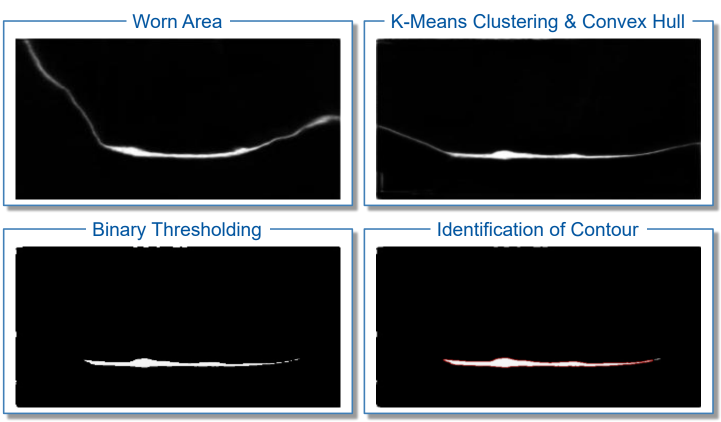
Use Case
Bevel Gear Cutting
Substrate: K30
Coating: AlCrN

K-Means Clustering
Number of Classes = 1

Binary Thresholding
Lower Threshold = 100
Upper Threshold = 200

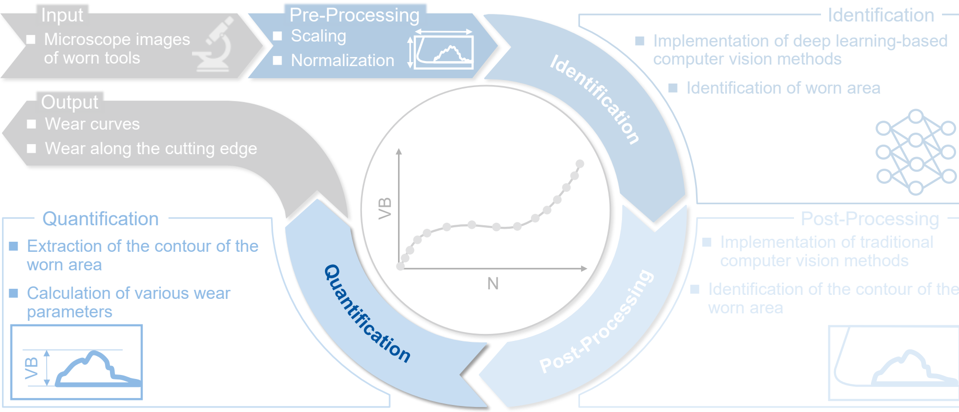
Computer
Intel® Core™ i9-10900K
Processor: 3.70 GHz; 10 Cores
Memory: 64 GB

Software
Anaconda 23.7.4
Python 3.11.4
OpenCV 4.10.0-dev



Algorithm for Identification and Quantification of Tool Wear

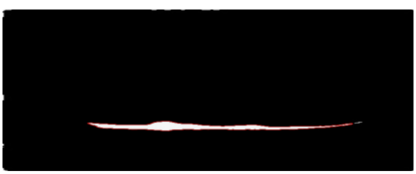
Procedure of the Algorithm – Quantification



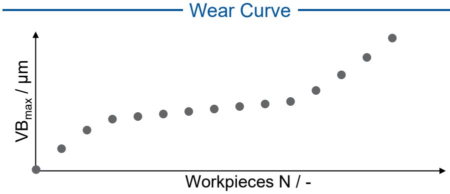
Algorithm for Identification and Quantification of Tool Wear

Quantification of the Tool Wear

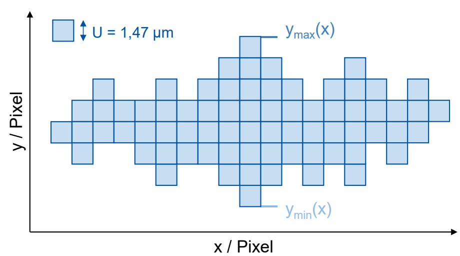
Identified Worn Area



Wear Curve



Quantification



$$VB(x) = U \cdot (y_{max}(x) - y_{min}(x)) \quad \begin{cases} VB_{max} = \max_x VB(x) \\ VB_m = \frac{\sum_{x=1}^{n} VB(x)}{\text{Anzahl}(VB(x))} \\ VB_{area} = \sum_{x=1}^{n} VB(x) \end{cases}$$

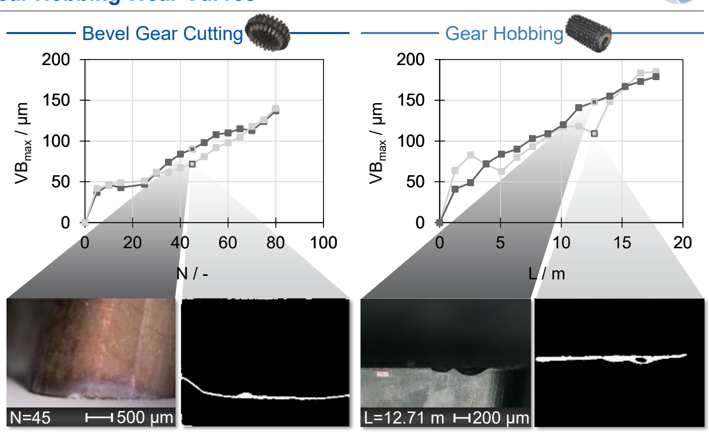
Outline

Comparison of Manually Measured and Algorithm Generated Wear Curves
Bevel Gear Cutting and Gear Hobbing Wear Curves

Bevel Gear Cutting
Process
– Single Blade Group Trial
– Face Hobbing Plunging
Tool
– Substrate: K30
– Coating: AlTiN
– Outside Blade, Tip

Gear Hobbing
Process
– Fly-Cutting Trial
– Dry
Tool
– Substrate: S390
– Coating: AlCrN
– Leading Flank, Section B

■ Manually Measured
■ Algorithm Generated



Outline

Summary and Outlook

Summary

- Comparison of traditional and deep learning based computer vision methods for wear detection on gear cutting tools
- Implementation of an algorithm to identify and quantify tool wear
- Comparison of manually measured and algorithmically generated wear curves

Outlook

- Adjustment of post-processing of identified wear for more accurate and robust quantification of wear
- Application of the algorithm to further gear cutting processes such as gear skiving
- Linking the algorithm to a machine-integrated measurement system for automated wear monitoring