MASTERARBEIT / MASTER'S THESIS

Titel der Masterarbeit / Title of the Master's Thesis

„Production Process Improvements by Utilizing Generic Close-To-Production Measurement Method"

verfasst von / submitted by

Nataliia Klievtsova

angestrebter akademischer Grad / in partial fulfilment of the requirements for the degree of

Master of Science (MSc)

Wien, 2022 / Vienna, 2022

Studienkennzahl lt. Studienblatt / degree programme code as it appears on the student record sheet:	UA 066926
Studienrichtung lt. Studienblatt / degree programme as it appears on the student record sheet:	Masterstudium Wirtschaftsinformatik
Betreut von / Supervisor:	Univ.-Prof. Dipl.-Math.oec. Dr. Stefanie Rinderle-Ma

Declaration of Authorship

I, Nataliia Klievtsova, declare that this thesis titled, ”Production Process Improvements by Utilizing Generic Close-To-Production Measurement Method” and the work presented in it are my own. I confirm that:

- This work was done wholly or mainly while in candidature for a research degree at this University.
- Where any part of this thesis has previously been submitted for a degree or any other qualification at this University or any other institution, this has been clearly stated.
- Where I have consulted the published work of others, this is always clearly attributed.
- Where I have quoted from the work of others, the source is always given. With the exception of such quotations, this thesis is entirely my own work.
- I have acknowledged all main sources of help.
- Where the thesis is based on work done by myself jointly with others, I tried to be as clear as possible to describe what was done by others and what I have contributed myself.
- Where the thesis is based on work done by myself jointly with others, the concepts and ideas always are my contribution. Implementation details are sometimes contributed by others, if so, this fact is clearly stated and their contribution is acknowledged.

Signed: ________________________

Date: ________________________

Acknowledgements

I would like to thank Prof. Dr. Stefanie Rinderle-Ma and Jürgen not only for their time and extensive guidance, but for their patience and best lectures, I have ever visited. I want to thank Dr. Christoph Pollak, Marcel Fuschlberger and the CDP (Center of Digital Production) for their help, cooperation and access to resources, data and research network.

Additionally, I want to thank my first professor Oleksandr Maksymov from Odessa National University for his motivation, support and inspiration.

I would like to express my special thanks of gratitude to my husband and my dog, who were listening to my complains and monologues about giving up the university at least once a week.

Also, I would like to mention the support of Elena Vladimirova, on whose kitchen I spent nights doing our semester projects, and the rest of my student colleagues, with whom we spent many evenings together trying to understand why something is not working (or working).

At last but not in least, I would like to thank my grandmother, my parents and my friends, who are now in Ukraine and not only, for their love and support to avoid procrastination during last two years and, especially, last four months.

Abstract

Production processes for individual products often consist of executable BPMN process models that contain hundreds or thousands of tasks. The more production steps that have to be conducted to produce a product, the more machines and people are involved, the more things can go wrong. This leads to a high number of parameters (physical properties such as temperature, machine properties such as the degree of wear of tools, ...) that can influence the quality of manufactured parts. This in turn makes it hard to establish deterministic models of how these parameters influence the quality of work pieces. With sufficient effort and data collection (in reality complexity is often to high) it might be possible to create such deterministic models for mass-produced work pieces. For lot-size one (i.e., small lot sized, or highly customized products) it is never feasible (or possible). Checking the tolerances after the manufacturing of work pieces is very time consuming, often taking over three times as long as the manufacturing itself. Also, measurement machines might have a resolution of sometimes more than 12000 measurements per second, which leads to huge chunks of data even for the measurement of very small pieces. The aim of this master thesis is to propose a concept for creating a generic close-to-production measurement method, that can (1) define end-of-line quality state of produced pieces, and (2) provide in-line prevention and correction of defects, while (3) reducing the data size from hundreds of gigabytes to almost zero. In other words: close-to-production measurement results with compressed data.

Kurzfassung

Produktionsprozesse für einzelne Produkte bestehen oft aus ausführbaren BPMN-Prozessmodellen, die Hunderte oder Tausende von Tasks enthalten. Je mehr Produktionsschritte zur Herstellung eines Produkts durchgeführt werden müssen, je mehr Maschinen und Menschen beteiligt sind, desto mehr Dinge können schief gehen. Dies führt zu einer großen Anzahl von Parametern (physikalische Eigenschaften wie Temperatur, Maschineneigenschaften wie der Abnutzungsgrad der Werkzeuge, ...), die die Qualität der gefertigten Teile beeinflussen können. Dies wiederum macht es schwierig, deterministische Modelle zu erstellen, wie diese Parameter die Qualität der Werkstücke beeinflussen. Mit ausreichendem Aufwand und Datenerfassung (in der Realität ist die Komplexität oft zu hoch) könnte es möglich sein, solche deterministischen Modelle für in Serie gefertigte Werkstücke zu erstellen. Jedoch für Losgröße Eins (in anderen Worten: sehr kleine Losgrößen oder hochgradig kundenindividuelle Produkte) ist dies niemals machbar (oder möglich). Die Überprüfung der Toleranzen nach der Fertigung der Werkstücke ist sehr zeitaufwendig und dauert oft mehr als dreimal so lange wie die Fertigung selbst. Außerdem haben Messmaschinen eine Auflösung von manchmal mehr als 12000 Messungen pro Sekunde, was selbst bei der Messung von sehr kleinen Teilen zu riesigen Datenmengen führt. Ziel dieser Masterarbeit ist es, ein Konzept für die Entwicklung einer generischen produktionsnahen Messmethode vorzuschlagen, mit der (1) der Qualitätszustand der produzierten Teile am Ende der Fertigungslinie bestimmt werden kann und (2) eine Inline-Prävention und -Korrektur von Fehlern möglich ist, während (3) die Datenmenge von Hunderten von Gigabytes auf nahezu Null reduziert wird. Mit anderen Worten: produktionsnahe Messergebnisse mit komprimierten Daten.

Contents

List of Figures

List of Tables

List of Algorithms

1. Introduction

Manufacturing has traditionally played a key role in the economic development and, despite argues that its importance is diminishing in the last recent years, achievement of economic development without industrialization is not possible. Manufacturing is still remaining the driver of sustainable economic development, as manufacturing goods are important for trade and service industries, etc. and it is an important source of innovation [16], as advanced manufacturing provides an important institutional foundation for learning and developing skills and capabilities, connected with core R&D, in the most important for the economic future industries [58].

However, most of this researches leads to increase of complexity of manufacturing systems, process monitoring and control, quality assurance, data amount and complexity and new challenges in 'industry 4.0', as they are concentrated on mass production's productivity, leaving small and medium-sized companies (SMEs) with single and small batch production behind. These companies encounter completely different challenges due to their specific characteristics.

1.1. Motivation

Manufacturing, in its simplest form, supposes the conversion of raw materials (typically supplied in simple or shapeless forms) into finished products with defined shape, structure, and properties by controlled application of energy [52, p.1]. Products are assembled from various parts, subsequently called work pieces. As only good work pieces (i.e., work pieces with their shapes, structure and properties being within defined tolerance ranges) are to be used in the final product, the fine-tuned production process is one of the core topics in the manufacturing.

Usually manufacturing entails the sequencing of the product-forms through a number of different processes [52, p.1]. In this work the manufacturing of a work piece is simplified to two generic activities/processes: (a) a series of operations to manufacture the actual work piece, and (b) determining the quality of the result.

If a work piece is good or not can be determined according to rules set during the design phase of the work piece: if the dimensions or surface properties adhere to defined tolerances, the part is good, else it is bad. Tolerances can exist for external and internal dimensions, concentricity, cylindricity and parallelism, etc.

Whether a finished work piece adheres to the tolerances, is influence by many parameters and events. If the machine itself is predetermined, the source of the fault can be:

(1) tools, which can be sharp or blunt and which are moved by the controlled machine axes according to

(2) software, which controls how (rotation speed, trajectory, amount of material that is removed per contact with the work pieces) the tools are used on the raw material.

Further parameter of influence is temperature, which changes with (1), (2) and the number of subsequently produced parts and leads to fluctuations: if more parts are produced in a row, the temperature is higher, than if we have short breaks in between the production of the individual work pieces. As all machines and tools are affected by environmental and usage influence additional parameters such vibration, contamination, collisions and humidity can affect its performance.

Avoidable	Unavoidable
tool wear	tool breakage
tool fracture	process interruption
flank wear	extra power consumption
tool rubbing	consequent vibratory effect
chip formation	surface roughness
chip breakage	surface derogation
crater wear	
notch wear	
nose wear	
chip removal	

Table 1.1.: List of various events occurred during machining

Numerous events (see Tab. 1.1), that can affect the state of the tool and manufacturing process during machining could be generally categorized into two types, i.e., avoidable and unavoidable. Despite proper monitoring and possibility to avoid such events as the tool breakage and process interruption, though their common occurrences in machining, there are a lot of events that influence the effectiveness of a tool as tool wear, tool fracture, chip formation, chip breakage, chip removal, tool breakdown, etc. which occurrence can't be controlled. Additionally to swarf formation, work pieces might be covered by dust or coolant emulsion[2].

Considering the fact, that we are talking about low - volume production, some manufacturing operations (e.g., machine loading and unloading, part inspection, part cleaning, bin picking, kitting) are largely done manually, that could carry in some side effects in the production process [1].

All these parameters make it hard to establish a deterministic model of how to produce good parts. While with sufficient effort and data collection it might (in reality complexity is often to high) be possible for mass-produced work pieces, for lot-size one (i.e., small lot sized, or highly customized products), it is not always possible to create such deterministic models. Regarding to [17] the distinction between low-volume and mass production is based on the number of products per batch and the frequency of repetition of orders per year. The average number is about 50 products per lot, which are produced less than 12 times a year on average. This tangles data acquisition and makes it problematic to create deterministic models and to use all specter of available data analysis tools. For example

Statistical Process Control (SPC), that is widely used in production industry, limited to series manufacturing. The minimum number of of samples to estimate the parameters of a set are usually not reached in this case [63].

Current local solutions (Zero Defect Manufacturing/ZDM) are therefor, focused on single production stages. They are also static and sequential, in the sense that when a problem is analyzed and solved at a specific stage, the company considers the process as 'frozen' and moves the attention to a new critical stage. This sequential strategy prevents the company from quickly adapting its production operations to changing production targets, thus undermining its competitiveness on the global market [9, p.1].

In both cases the end-of-line quality determination is considered, that excludes in-line prevention and correction of defects.However, checking the tolerances after the manufacturing of work pieces is very time consuming, often taking over two times as long as the manufacturing itself. Therewith end-of-line control concerns parts that are shipped directly afterwards,allowing defect parts to slip through and to be sent to the customer, what is unreliable in the scope of high quality requirements [20].

Also, end-of-line inspection could lead to waste of time and costs, as low-volume production is pretty unstable itself, because of machinery that could be re-calibrated from batch to batch. The more instable the production is, the higher scrap rate would be achieved. For some types of manufacturing scrap rate can be 15-20% or even higher, however if raw materials are cheap, scrap reduction is consider as an optional action. But scrap cost the company much more than simply the price of lost raw materials, as company should spend more on inspections and overall costs of the manufacturing process.

Moreover, equipment for end-of-line measurement technologies should be places in laboratories or should be designed with special additional characteristics (resistance to temperature changes, vibrations, moisture or dust) for harsh environment on the shop floor [13]. Exploitation and operation of these equipment could be complicates, what requires experts and additional training and special programs for each working piece should be created.

1.2. Research Questions

In traditional batch manufacturing quality attributes of products are tested at the end of each batch manufacturing step, following the so-called Quality-by-Testing (QbT) (see Fig. 1.2) [44]. This approach is preceded after manufacturing of the items. That fact is the reason, why sometimes production should be repeated multiple times before adequate quality is reached [21].

As more industry starts to implement smart manufacturing and "Industry 4.0" approaches, it is crucial for small batch manufacturing to use these practices as well. Based on process automation standards it is possible to integrate process equipment, sensors and monitoring systems for effective implementation of real-time data analytic to improve quality assurance [45].

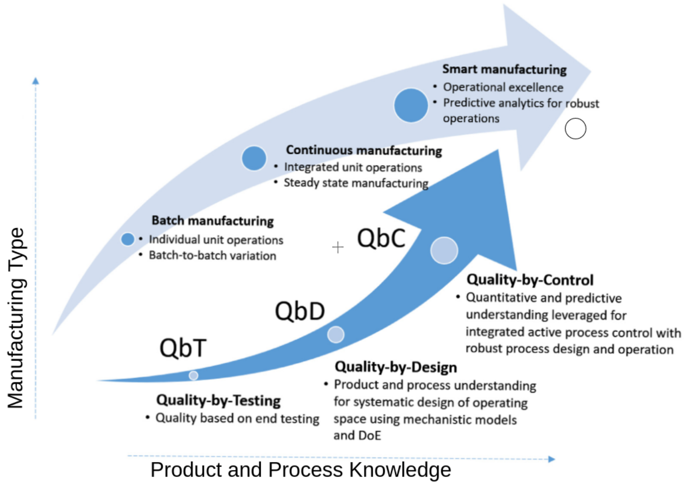


Figure 1.1.: The systematic progression in quality assurance [44]

When we are talking about integration of the production process quality control systems we emphasize 3 major steps [13, p.404]:

1. Preparation for in-line quality control
2. Implementation of measurement technology
3. Implementation of quality control cycles

As the main goal of the master theses is to improve production processes, we concentrate on the improvement of the second one. In order to improve current measurement technology we need to answer following questions:

RQ1. How can the software of close-to-production measurement machines work in a generic manner? I.e., the same program should be used to determine the properties of arbitrary work pieces.

RQ2. How can the data that such a generic program yields be structured, so that subsequent part specific logic does not have to deal with large point clouds, but with a set of human understandable values, such as dimensions and acclivities?

RQ3. What are feasible methods to automatically determine from a list of tolerances, which measured dimensions and acclivities are corresponding to (a subset of) these tolerances?

RQ4. Is it feasible, for a real-world data set, to reliably determine bad work pieces, without any further data (i.e., in-production, no access to historic final QA data)?

RQ5. How can tolerances and how individual work pieces adhere to them, be used to produce recommendations for changing the parameters of a manufacturing machine?

1.3. Contribution

The contribution of this master thesis is three-fold:

(a) The design and implementation of a measurement prototype that allows for the measurement of arbitrary parts instead of being based on a specific measurement program suitable for only one type of parts.

(b) A method to compress the data provided by the measurement machine, without loosing the information contained in alternative multi-gigabyte in size point-clouds. Under loosing of information we understand preservation of working piece contour in general and its the most important characteristics to be useful for further analysis.

(c) A method to analyse compressed data, which yields at-least as good results than analysis of the traditional point-cloud data for considered tolerances and dimensions.

1.4. Structure

The remainder of this thesis is structured as follows. In the Section 2 we will introduce concepts, commonly used or referenced in this work and related work, advantages and disadvantages of various measurement technologies. Section 3 and Section 4 introduce experimental set up and which solutions and suitable analysis methods for work piece quality control will be considered for the prototypical implementation and realization of implementation itself. In Section 5 evaluation of each supposed method is presented and discussed. The work finishes with obtained results and conclusion in Section 6.

2. Concepts and Related Work

Beside analysis if the related work to define current state of the art in the filed of research, all necessary concepts in terms that are used and references in the master thesis will be introduced in this section.

2.1. Low-Volume Production

Production systems (methods in industry to produce goods from raw materials) can be generally divided into two types (1) intermittent and (2) continuous based on type of production, company size and production line (see Fig. 2.1).

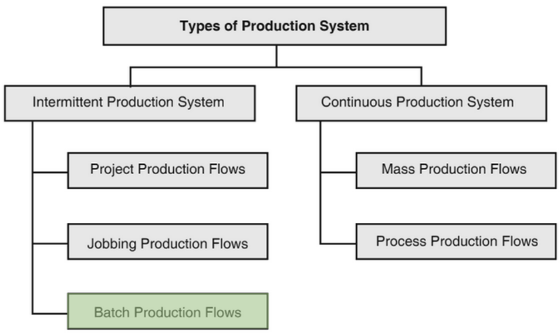


Figure 2.1.: Types of production system [47, p.281]

Continuous production system means that goods are produced persistently based on demand forecasting. Production process, its steps, parameters, conditions, raw materials and produced goods are standardised, in other words one production line is used for one product. This type of production system represents large-scale manufacturing.

On the other hand, intermittent production means that products are manufactured based on customer's orders. One production line is used for producing more types of goods (i.e., no standardization), machines are often reconfigured and there is no single production process, broadly speaking production is flexible, as product design is changed regularly [26, p.256].

Due to enormous difference of production system types as time requirements for production run, production scale, nature of products, flexibility etc. it is clear that approaches, methods and techniques that are appropriate for one type are not always applicable or effective for the other one.

Intermittent production systems is typical for low-volume manufacturing. The main difference between project, jobbing and batch production flows are:

1. in project PF company accepts a single complex order, cost and time limitations are discussed and defined before. Equipment, materials and tools are are brought to the precast location.

2. job PF is common for unique products (single unit production). This type of production is usually expensive, as product should answer all specifications, required by the customer. despite high costs products have perfect quality.

3. during batch PF goods are produced stage by stage in certain amount (lots). Each lot can vary from each other, but during one batch only one type of item can be produced. Each batch should be completely finished in order to start the next one, since the same production line is used. Machinery should be reconfigured and part of equipment or tools could be substitute [30].

In the scope of these master thesis we are concentrated on the small batch production because it is appropriate for a wide variety of businesses (from food to pharmaceuticals) in comparison with the project and job production, which covers too specific cases. Such big companies as IKEA use batch production, since this type of production has a lot of advantages over mass production, as flexibility, waste reduction, multi-use capabilities, lower costs, reusability etc.

However, batch production has following limitations:

- material handling is complex
- downtime between individual batches
- more planning, scheduling and control over the process
- higher set up costs
- lower supply of techniques and tools in comparison with mass production

2.2. Types of Quality Assurance

Because of high requirements to product quality in small batch manufacturing it is essential to understand which types of measurements exist and how they differ form each other. We differentiate between two types of measurements by how they are integrated into the production system(see Fig. 2.2):

(A) close-to-production measurements (in-line)

(B) final quality assurance (QA) measurements (off-line/at-line)

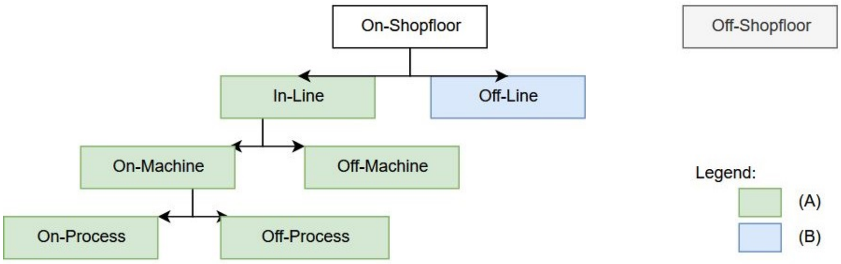


Figure 2.2.: Measurement technology in production system [13, p.403]

Final QA measurements are usually realized by Coordinate Measuring Machine (CMM) or Computed Tomography (CT). These measurements provide high accuracy and are pretty flexible, enable to perform more complex measurement processes, have less environmental impact on measuring results. However, the product is usually removed from the production line for inspection (see Fig. 2.3), have long duration (long time between manufacturing and measurement result means that production problems can be recognized too late), high costs, part size and density limitation or some parts of the work pieces can't be reachable, data collected in discrete intervals [62].

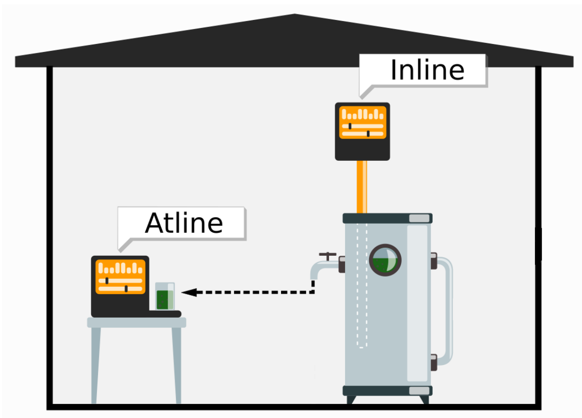


Figure 2.3.: Measurement variant in process analytics [57]

Close-to-production measurements are performed directly in the process (see Fig. 2.3). In-line measurements reduce the manual sampling required and can increase the number of data collected over the course of a batch [40]. They are very fast but yields only rough estimations, which often do not allow for determining of tolerances, and are often hard to implement given the requirement for high speed. There are some additional counter arguments against these measurements:

- measurements with the tool (axes, spindle, and frame) used for machining at the same time is distrusted

- these measurements are often considered as non-value-added as they detain machining operations and could lead to productivity reduction
- no confidence in accuracy of obtained measurements as manufacturing environment can be unstable and unpredictable because of impact of external factors (i.e., humidity, temperature, sticky cutting fluid, swarf, etc.) [18, p.2]

Both types of measurements provide real-time and near real-time measurements but require specialized programs to be run, in order to do their measurements, which is a big burden when manufacturing small lot-sizes. In other words: in addition to the NC-program that manufactures the part, two additional programs are needed to measure the part. All of these programs are typically required to be created by specialists and are difficult and time consuming.

2.3. Overview of Measuring Techniques

2.3.1. Hand Tools

Hand tools for quality assurance the most simple method to obtain the information about produced working piece. These tools are cheap, require no special knowledge and preparation for its use, no setup and professional treatment is necessary. They are very universal and flexible and can be used for different goods and for various dimensional inspections (micrometers, slide calipers, squares and gages etc) [60].

Usage of these tools however is always associated with high labor costs, relative low level of assurance. Additionally realisation of data collection is simplified and non automated, that not allows to perform adequate and meaningful data analysis.

2.3.2. Pneumatic Gauging

Pneumatic Gauging (aka air gauges) utilizes the restriction of air, and the resulting change in flow or pressure to measure a dimension like an outside diameter or hole, as it is a non-contact inspection method [53]. Pneumatic gauging has following advantages: it is fast and has the ability of the escaping air jet to clear fluid and grit from the working piece, absence of damage to delicate work surfaces and freedom from the vibration sometimes found in direct contact.

Air gauges are used to measure working pieces, produced by mass production and are applicable for parts, which are both smooth and flat, and static with respect to the gauging head [53].

2.3.3. Coordinate Measuring Machines

According to the ISO 10360-1 (2000) standard, a "coordinate measuring machine" (CMM) is a "*measuring system with the means to move a probing system and capability to* *determine spatial coordinates on a working piece surface*". CMM is the most popular tool to measure items with complex surface and structure. Term 'CMM' covers the wide range of measuring systems, as lightweight portable arms or heavy machines with special foundations, CMMs with contacting or non contacting probes. Each type of CMM has its advantages and disadvantage, so the configurations should be selected based on the measurement task and working piece characteristics. However, all CMMs are usually expensive and require trained and experienced staff, as all CMMs are controlled with a a computerized numerical control (CNC) system [4].

2.3.4. Optical Systems

Optical system is the system that utilizes light and optics to determine dimensional tolerances of manufactured item. Systems can vary by size, accuracy etc. Optical instruments and variations on these instruments are listed in ISO 25178 part 6 (2010). Common tools for this type of measurement systems are optical profilometer and optical 3D microscope. They are typically non-contact and obtain the surface profile of the produced work piece with the help of optical probe [23].

Most of the tools are oriented on definition of surface finish assessment (profilometer) or support of visual inspection (microscopes), that's why in order to define geometric and dimensional tolerances of the work piece requires additional instruments and software [4].

2.3.5. Light-Based Systems

This systems use structured light (LED) or laser line triangulation to make measurements on the working piece. Items are are measured non-contact with the help of the laser beam divergence technique [5]. Laser scanners are widely use if measuring range of several meters, have a very high data acquisition rate (up to 2,500 data points per second), and good resolution. High density profiles can easily be scanned over large free-form surfaces [13, p.311]. However, laser scanners could be sensible to the environment: vibrations, water, dust, chips etc. can provide wrong results, despite high accuracy of the later itself.

2.3.6. Vision-Based Measuring Systems

Vision-Based Measuring System is non-contact measuring systems involving software to produce and evaluate high-resolution images. Typical example of these systems is photogrammetry. Photogrammetry provides methods to give quantitative data from the photo. From a two-dimensional plane it is possible to get only two-dimensional coordinates [25].

Therefore, if three-dimensional coordinates are required, some protective transformation of different photos of the same item should be made (at least 2 photos). Photogrammetry can be used to implement measurement of free-form parts, but measurement speed is usually low, as some parts have no adequate homogeneous surface or too complex structure, a lot of photos from different angels, high computational performance and additional information is required [13, p.310].

2.4. Data Acquisition and Processing in Production Metrology

2.4.1. Data Acquisition

Since manufacturing faces a lot of requirements to stay competitive, digitization became one of the most important topics in this branch. Production data acquisition (PDA) provides an opportunity to collect important information about company processes and it steps in real-time or near real-time (i.e., energy, process, machine and manual data, operating cycles and times).

Quality assurance and production metrology data are part of the PDA as well, but as there is no specific standardisation for measurement industry approaches concerning fully automated data acquisition systems are not widely spread among SMEs [51], exactly this kind of data could be ignored, acquired data could be incomplete for exploitation or could be huge and difficult to understand and manage [48]. Nevertheless, production metrology (and collection of supported data) is important component to improve quality assurance and facilitator for higher productivity in total.

Fully automated solutions for data acquisition and data processing are widespread mostly in mass production manufacturing and are either too customized or too complex and require advanced data processing systems, what makes them inapplicable for flexible batch production. From batch to batch requirements for measurable quality criteria of a product can vary. Essentially, there are 3 different criteria of a product as material attributes, geometry and product functions (see Fig. 2.4).

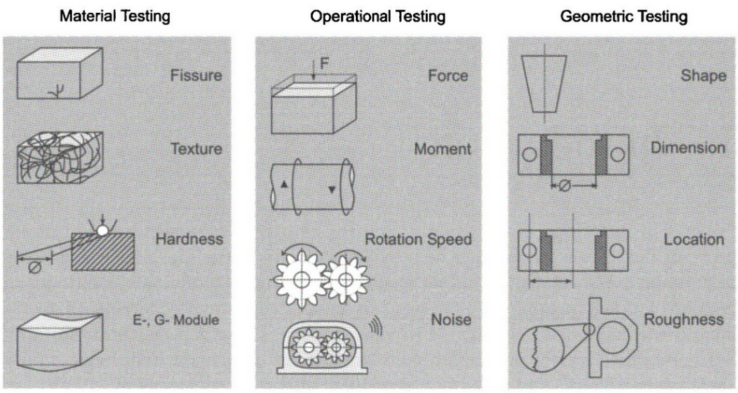


Figure 2.4.: Testing and measurement functions within production metrology [33, p.16]

Based on this criteria and techniques that are used to gather the data (see Sec. 2.3) distinct types of data would be obtained (single values, point clouds or volumetric data), i.e., raw data that should be reprocessed and transferred in appropriate models and representations or necessary information should be extracted from it [3].

2.4.2. Data Processing

For both in-line and off-line measurements process of data preparation and data analysis is the same. After obtainment of raw data from respective sources following steps take place:

1. Data Preparation
2. Feature Engineering
3. Data Analysis
4. Evaluation

Data preparation is the first and one of the most important phases of the whole data processing pipeline. On this step such manipulations as data filtering and cleaning (due to unstable manufacturing environment gathered data could include invalid or missing data, noise or measurement data of supported objects as grippers), data integration (most of the measurement techniques based on requirements capture more than one dimension/view of the working piece that should be integrated and adjusted according to the axes and coordinate systems), surface reconstruction (used for prepossessing of point clouds of unknown working pieces for future comparison with the models of provided items), data compression and simplification (allows to minimize data redundancy and to increase data analysis) are performed.

Feature engineering is the process of defining the most appropriate numerical representations of raw data (i.e., features). There are 4 types of techniques in FE as feature creation, transformation, extraction and selection. First two involve human interaction (identifying of important variables and data modification for simplified data representation respectively). Feature extraction is the process of information derivation from provided data with to create new subset of new features (often unsupervised), unlike the feature selection analyse already existing features in the data and determine which of these features are more influential and relevant [7].

After step 1 and 2 prepossessed data or obtained features are used for final analysis to answer the questions as whether the working piece is good or not, if swarf is present or not, which working piece was produced (if the information is not provided), which tolerances were violated etc. There are a lot of techniques starting from regression analysis to Neuronal Networks and advanced machine learning tools. For high accuracy usually supervised learning (data should be labeled) is used, but in the content of small batch manufacturing this approach is not optimal, as there is not enough data to train the model. According to the goals of quality assurance preference should be given to unsupervised learning, regression analysis or time series analysis [3].

Analysis itself can't be reliable without evaluation of results. Evaluation is essential to understand if the chosen model is useful, is the number of included features and features themselves are selected right, is the amount of data sufficient or another analysis techniques should be found. One of the most significant metric of evaluation is accuracy, however it has serious limitations by stand-alone evaluation.

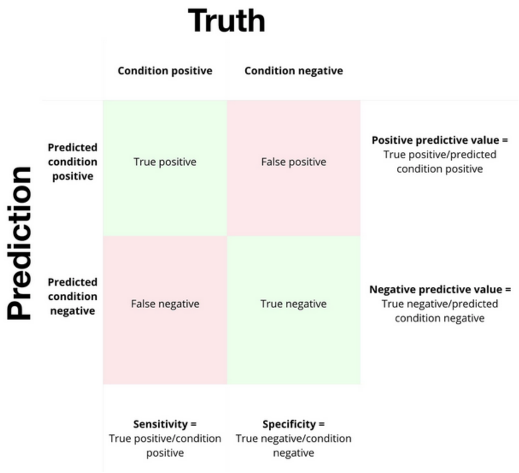


Figure 2.5.: Confusion Matrix

Based on accuracy it is not possible to define if obtained results are balanced (i.e., misclassification cost is not considered) [19]. In order to really evaluate the performance of selected method confusion matrix should be used (see Fig. 2.5).

Confusion matrix is a simple table representation of achieved performance of selected classification model. It is a two-dimensional matrix, which in one dimension represents the true class of an object of interest and in the second one - the class that was predicted by the classifier [49]. As most of questions in quality control can be answered with yes/no, classical case of confusion matrix is with two classes (positive and negative). In this context, the four outcomes occur: true positives, false positives, true negatives and false negatives [49]. With the help of these values additional parameters as precision, recall and specificity could be found.

Evaluation of these parameters helps to define real state of selected model and improve it. It is important to mention, that evaluation is not the end itself. Based on evaluation results not only improvement of the model can be made, but sufficient conclusions about limitations and strengths of chosen methods could be found and estimation of the model in current domain could be performed [19].

3. Solution Design

3.1. Improvement of measurement technology

On the first view close-to-production measurements have far more disadvantages in comparison with the final QA. However, some circumstances argues in favor to use of in-line production measurements. Production of faulty parts increases the costs exponentially during the manufacturing process (see Fig. 3.1) [13].

Therefore, in-line-production measurements may improve product quality, customer satisfaction, and profitability, when the close-to-production measurements are accurate enough and the cost of in-line-production measurements performance is less than the costs of raw materials and production of faulty parts [18, p.3]. Due to short control loops, changes in the production process can be implemented immediately.

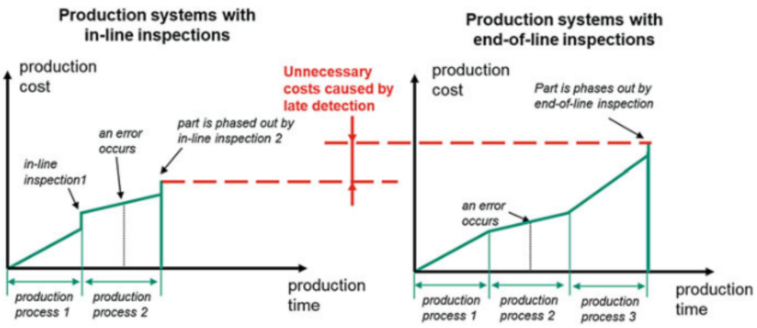


Figure 3.1.: Cost increase during in-line and end-of-line inspections [13]

By combining modern machine tools with reasonable measurement techniques very precise performance tolerances could be reached, as accurate measurements could be performed. In-line production measurements reduce number of scrap parts, operational and wasted machining time. They avoid the “second fixation problem of the work piece”, in other words the necessity of moving produced part from production line to a CMM [66].

Another benefit of in-line-production measurements is reduced human interaction which improves measurement accuracy, e.g., less manual ’cut and measure’ errors and overtime bills. This leads to improvement of conformance, consistency and lowers costs, as amount of rework time and number of defects is reduced. This type of measurement expands system capabilities by fulfillment of more complicated work, customer demand satisfaction for control and monitoring of manufacturing processes [61]. One more advantage of close-to-production measurements is an indirect opportunity to find out which parameters are really relevant for quality assurance and possibility to extend or to reduce current parameters.

To satisfy modern requirements for product quality in manufacturing it is necessary to achieve the improvement of dimensional inspection system (i.e., make it more faster and more accurate) . This claim forces CMM to inspect parts faster and generate their programs for inspection automatically. To fulfill this arisen demand deployment of CMMs in a computer integrated manufacturing environment is necessary. Nevertheless, for efficient realisation of such deployment inspection systems must be used in coordination with CAD systems [67].

The interactive CAD environment is specific software, that brings additional problems for lot-size one work pieces as the cost of ownership is too high and it's being criticised by employees with experience in the CAD software [56].

3.2. Experimental Set Up

Currently the production of the working pieces (parts for a gas-turbine GV12, see Fig. 3.2) consists of 3 main stages:

(1) definition of planned batch size and orchestration of all machines and robots involved in the manufacturing and measuring
(2) manufacturing of the working piece itself by using EMCO MaxxTrun 45 (milling machines for CNC turning and milling) -> 4 minutes on average per part
(3) measurement of a part with MicroVu Vertex 261 (automated precision measurement systems). Measuring is running parallel to the machining -> 8 minutes on average per part

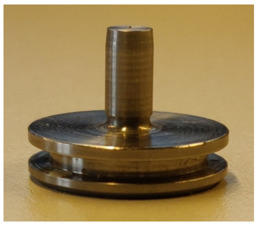


Figure 3.2.: GV12

As we can see, off-line-production measurement of the working piece takes 2 times longer as manufacturing itself. Even though this sub-process is running parallel to the manufacturing it leads to increase of production time in general. Additionally it is necessary to create a new measurement program for each part individually and sheer volume of data points from the obtained results is quite time-consuming to process and analyse.

With tactile probing it is not possible to inspect all hard-to-reach spaces and, as not the whole surface profile is depicted (see Fig. 3.4(a) where green lines are reached areas and red represent spaces, that can't be reached), what could lead to wrong classification of the produced items at the end. Chips formations, dust, oil etc. can't be detected before measurement procedure, that generates wrong measuring results and cause unnecessary run of such a time-consuming sub process.

In the scope of these master theses under close-to-production measurement tool we consider high precision micrometer system, which allows to correct vibrations and mis-alignment (however, only until particular degree) automatically (see Fig. 3.3). The system illuminates working pieces with a Green-LED and a telecentric lens. Green-LED light forms a uniform collimated beam. When working piece breaks this beam it creates a 2D shadow on a CMOS and different features (e.g., size and angle) can be calculated by measuring this shadow [59].

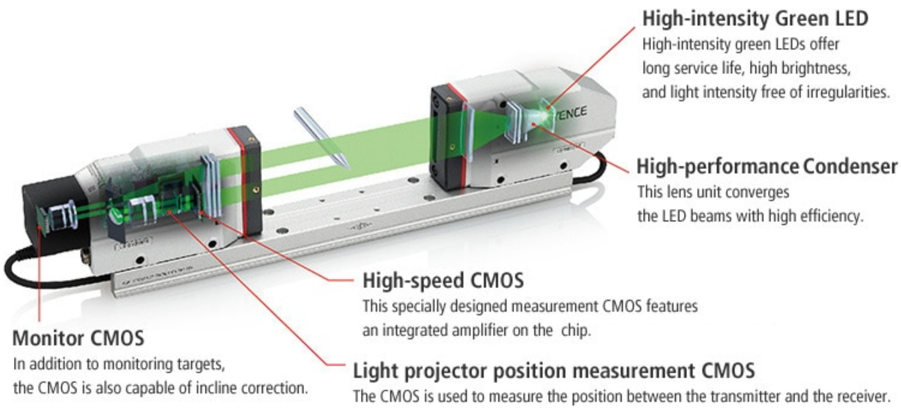
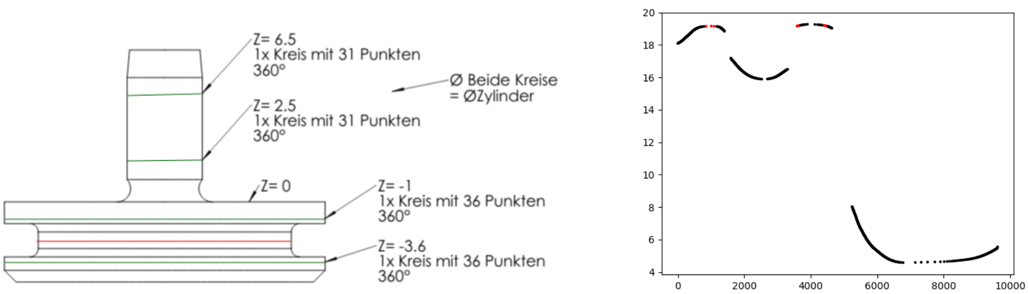


Figure 3.3.: 3-CMOS×Green-LED system [59]

This method utilises the unique advantages of the thrubeam model, which is not affected by lighting, to offer high-precision measurements. With other words the CMOS detects the position of the edge between light and dark edges of the received light and calculates measured value [59].

This approach allows us to collect the data about each work piece in a generic manner and avoid writing programs for different types of the work pieces. Whole surface is depicted by saving values during the part movement,that can be used for further analysis (see Fig. 3.4(b)). Unlike MicroVu measurements, we can't gather all necessary information (parameters as flatness, parallelism, cylindricity etc.), that is required for determining final quality state of produced working piece.



(a) MicroVu measurements of GV12

(b) Micrometer measurements of GV12

Figure 3.4.: Real work piece VS obtained surface profile

Further improvement for current measurement procedure is implementation of the lift in the middle of laser grid of Keyence machine (see Fig. 3.5, A.2) for swift and steady measuring of parts, to avoid misalignment of items by holding them with the gripper (or other tools involved in production process), to provide constant speed during working piece movement through the laser and to give the working piece higher freedom degree, i.e., as part is staying on the platform and is not holding by gripper each its segment can be measured. Lift is easy to implement into existing system with the Raspberry Pi, that makes it available for SMEs and low-volume production.

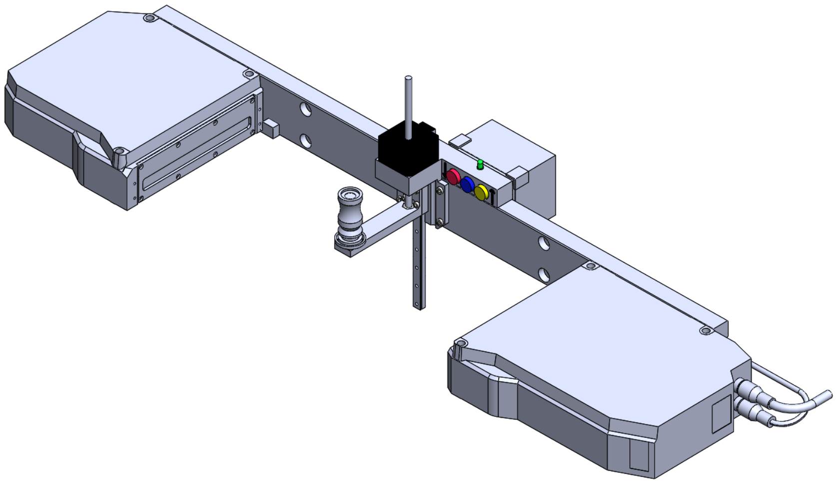


Figure 3.5.: CDP Lift

Experimental set up is currently restricted to work pieces with cylindrical geometry only, one surface view would be considered and influences of the environment would not be excluded or avoided. According to limitations of present experimental set up it is clear that complete removal of final QA measurements make no sense, but production process of the work piece can be changed by adding further steps with short control loops after to reduce the usage of off-line measurements.

Improved production process of the working piece contains following stages:

(1) definition of planned batch size and orchestration of all machines and robots involved in the manufacturing and measuring
(2) extract relevant tolerances (i.e., diameter) from technical drawing of desired working piece
(3) manufacturing of the working piece itself
(4) data compression of obtained surface
(5) data alignment
(6) analysis of the surface on the swarf presence (if there is no swarf go to the next step, otherwise break the process)
(7) extraction of necessary values representing tolerances from gathered surface

(8) comparison of values from step (2) and (7) to define the quality of the working piece (if part is good go to the next step, otherwise break the process)
(9) performance of final QA measurements to define final status of a working piece

In presented scenario all the data is collected with the help of process execution engine CPEE[1]. Whole information is stored in so called process execution logs in XES format[2]. Each log contains data about all steps of production process, its relevant attributes and sensor data.

3.3. Close-to-production measurement procedure

Collected data represents a time series with one variable - diameter of the working piece. In order to define whether the working piece is covered with the chips formation or not, the comparison of two time series is required.

When a comparison of two streams of data with implicit or explicit time information associated is executed, there is the need for a measurement function that provides information on the similarity of the two data streams. There is a huge variety of available distance measurement functions to perform time series comparison. However, traditional metrics, e.g., Euclidean distance, assume that time series values are aligned within the time axis and are synchronized relative to each other [12].

In manufacturing domain, because of different speeds and duration during measurements, similar regions may appear in different instants in time, leading to different degrees of time distortion, or time warping, among several sequences, since they are not aligned in the temporal domain. Under such conditions, traditional distances are not able to measure this distortion since they are very sensitive to time delays (when time series are out of phases) and typically unable to directly handle unequal length time series without some sort of prepossessing [12].

Furthermore most of the algorithms, that can handle unstable time axis, still suffer from the drawback that they may prevent the "correct" warping from being found and can produce pathological results, as algorithms try to explain variability in the Y-axis by warping the X-axis. As a result of misalignment and phases unsynchronization between time series a single point if the first one could map onto a wrong subsection of another one [27].

These limitations can be crucial obstacle by part's classification since during the measurement of working pieces some of created time series can obtain from 1.000 to 80.000 points per part depending on its size.

3.3.1. Data Compression

Despite using of micrometer systems allows us to get comprehensive information about the surface profile, we get a big cloud of repetitive data that are to large and not more human understandable. The problem with storage capacity, data operation time, increase

[1]Cloud Process Engine: https://cpee.org/
[2]http://www.xes-standard.org/

costs for storage hardware and network bandwidth should be considered as well [55]. Furthermore, large data volumes is an issue for cloud manufacturing, interest on which has been increased significantly recently.

In order to minimize data redundancy and save generic manner allowing to measure different work pieces we should use data compression approach which represents data in a less redundant fashion and remove the redundancy in data containing the main features of each work piece respectively [35].

"Techniques for lossy compression of geometry data often imply an ordering of the detail coefficients according to their significance" [11, p.359]. Under detailed coefficients we understand the combination of points of data cloud model. To obtain desired compression amount we can simply remove a certain percentage of data from cloud model starting with the least significant ones. To keep tolerances, we can remove the least significant detail coefficients as long as we do not violate or remove this information during coefficients reduction [11, p.359].

Not to violate the tolerances and carry out appropriate data compression it is necessary to define, which parameters are significant for quality control. In the process of machining, high product quality is inseparable from the very fundamental rules of size, dimension, geometrical tolerance and geometrical properties of the surface [8].

In the context of this work we concentrate our attention on geometrical tolerances, i.e., tolerances that have to be considered for surfaces, which contact with other parts. These tolerances are used to cover and pass all geometrical requirements, defined based on engineering drawings, in a brief and precise way [42].

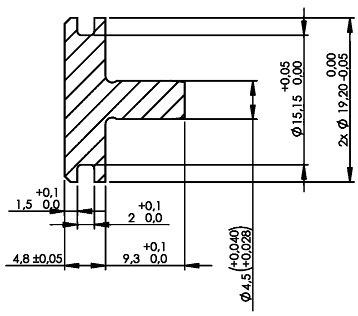


Figure 3.6.: Technical Drawing of the GV12 working piece

Since we consider only work pieces with cylinder form all the tolerances that bear on diameter are crucial(see Fig. 3.6). To keep these tolerances after data compression and still achieve good compression rate more approaches are suggested.

Approach 1:

> *Define all edge points of the surface's profile during measurement procedure by comparing the difference between each point with the previous one based on defined threshold.*

> *Extract all edge points and save them to the new array (length of the array is always even) and for each 2 points of the received array reproduce the whole profile's surface with the help of linear regression.*

We assume that such approach allows us to keep the information about the shape of the work piece and it's cylindrical parameters to check the tolerances with notable data reduction.

However, detection of the edge points can be difficult procedure as more/less candidates for edge points as their real amount can be found due to defined threshold, high resolution of the laser and working piece misalignment or its structure (i.e., flatness).

The threshold definition for edge point detection for each type of the working piece should be done by expert (time costly or not without human factor) or automatically, but only in case if there are enough training data and with predefinition of specific sensitivity.

Approach 2:

In order to extend *Approach 1* and to avoid edge point detection problem, another method is suggested:

> *Split up surface profile into equal areas: each N values or each N milliseconds. Save first and last point of the area to the new array (length of the array is always even) and like int the previous approach for each 2 points in received array reproduce the whole profile's surface with the help of linear regression.*

This method is simple in its implementation, but the size of the areas (N) should be defined by expert or should be counted automatically based on some criteria, as by small N value the data are still staying in point cloud format and by large N value a lot of important information would be lost and real surface profile could be distorted.

Additionally, we assume that due to the online environment, even if measurement procedure is constant, each working piece differentiates from the other one in some way and the same N value for two working pieces of the same type could cover their various segments.

Approach 3:

To achieve some data standardization and similarity between working pieces inside one batch, another method is suggested:

> *Split up surface profile into some N areas based on the technical drawing and structure of the working piece. For each segment extract extreme values. If there are segments where only some area is of interest, define the size of this area and extract extreme values for this region as well. For each area perform linear regression and save first and last regression data into final compressed data.*

This method should be very efficient, however it requires specification, what makes this method not generic. Specification varies based on the type of the working piece and should be defined by expert and can't be counted automatically.

This method potentially has a lot of advantages as based on the number of obtained segments and expected number of segments chips could be easily detected. All working pieces that were aligned correctly would have the same N segments and structure. Extreme values could be used for further tolerance analysis.

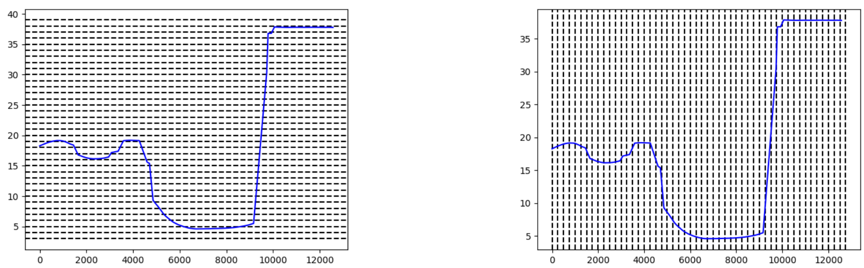
Saving regression coefficients (slope and intercept) evaluation of working piece alignment during measurement can be performed and judgment about correctness of measured values could be done.

Approach 4:

Based on assumptions about approaches 1 and 2 we expect not only edge point detection problem, but the segmentation problem as well. Previous approach (3) allows to avoid these problems, but has such limitation as specification. To bypass all of these issues new method is proposed:

> *Create horizontal and vertical grid with specific step through the whole profile surface. Find all points of the surface, that cross the grid in both directions (see Fig. 3.7). Combine cross points arrays for both directions, i.e., overlay and sort them by both axes and remove duplicates, if they exist.*

Using this approach good data compression can still be achieved, but we are not expected the significant percentage as in the previous approaches. No specification is needed, segmentation and edge point detection problems should be avoided, nonetheless some critical values could be lost, after reconstruction some points can overlap, and step size should be defined for each type of working piece by expert or additional script and methodology for step size definition should be created.



(a) horizontal (b) vertical

Figure 3.7.: Horizontal and vertical grid through profile surface

Approach 5:

To improve the level of data compression reached in previous approach (4) the new method is introduced:

> *Create only horizontal grid with specific step through the whole profile surface and find all points of the surface, that cross the grid (see Fig. 3.7 (a)). For each two pairs from the previous step calculate the difference between timestamps and measurement values. Based on the time and measurement delta both pairs would be left, additional local extreme values for this segment would be found, or pairs would be deleted. Obtained list of pairs is the compressed profile surface.*

This algorithm should provide better data compression, that previous one, help to avoid overlapping and save extreme values that are needed for further analysis. Nevertheless, the definition of step and threshold is necessary.

After consultation with the expert and analysis of different types of working pieces some simple method to define the step for the horizontal grid is suggested. Find the minimum and maximum of measurement values of the whole profile surface and count the difference between them. With the help of this difference we can suppose how uneven and bumpy the surface is and how frequent the grid lines should come to gather the most important points from the cloud point.

- if difference is larger than 1 cm, the step is equal to 1 mm
- if difference is larger than 1 mm but smaller than 1 cm, the step is equal to 0.5 mm
- if difference is smaller than 1 mm, the step is equal to 0.1 mm

To define step size for the vertical grid we decided to split the whole time length into 50 equal areas regardless of duration, as in the scope of our prototype we are working with relative small working pieces (i.e., not longer as 10 cm).

It should be pointed out that all advantages or disadvantages that are described in this section are only logical assumptions. After implementation and evaluation new arguments for and against, side effects or limitations could be found. Additionally, it could turned out, that steps and thresholds are insufficient and should be adjusted or redefined.

3.3.2. Time Series Alignment

As we're talking about laser measurements in manufacturing environment time series usually have unequal length even if it's handled about working pieces of the same type. Data compression helps to overcome some limitations of the time series measurement functions, as long computational time. However such problems as sensitivity to distortion in time axis, not constant time steps and the fact that some algorithms require two sequences being compared are of the same length are still remaining.

For proper comparison of two time series their alignment is necessary (i.e., way of arranging the time series to identify regions of their similarity to identify functional or structural relationships). Due to problems mentioned above some previous synchronisation/rearrangement should be performed as normalisation, resampling or interpolation of data.

Normalization is a rescaling of the data from the original range so that all values are within the range of 0 and 1. Interpolation used to fill the gaps in the time series [24].

Resampling is changing the frequency of the time series for imbalanced data sets to adjust the number of common and rare observations. Resampling is achieved by two strategies as deleting some common data from time series or generating and adding some rare values to it (undersampling and oversampling respectively) [29].

As we consider time series as surface profile (i.e., geometrical data) just resampling is not always effective, as we are interested not only to obtain time series of the same length but different segments of the working pieces should be synchronized (for suitable alignment segmentation should be done).

In other words data cloud should be split to the meaningful areas that correspond some parts of the real working piece. Most of the data segmentation algorithms can be grouped into 3 classes: edge-detection, region-growing and hybrid methods. Edge-detection methods extract edge corners of the working piece (i.e., discontinuities in the surfaces). Unlike previous ones, region-growing methods detect continuous regions with homogeneous geometrical properties. Hybrid methods combine the properties of the two methods mentioned before to overcome the limitations of both of them [65].

Suggested approach for ***time series segmentation*** to simplify time series comparison is following:

> *Find the value to split up surface profile into some N areas based on the technical drawing and structure of the working piece. To define this value create a horizontal grid through the whole profile surface based on the particular step (could be defined by an expert or automatically). For each line in the grid count, how often does the line crosses the profile surface and select the maximum value from the list of candidates.*

The number of how often do the grid lines cross the profile surface is the number of potential segments of the working piece. We decided to use maximum value as selection criteria, as we want to achieve better detailing and the greatest level of synchronization. However, it is possible, that after evaluation another criteria should be picked up.

To achieve a good alignment and synchronization between the original time series and the current one following procedure should be done:

> *For both time series call the segmentation function described above. Determine if the number of assumed segments segments and the split line value are the same. Split both time series into N segments and for each segment compare the length of these areas. If they are not the same resample the shorter area to the length of the larger one and update this area with the resampled one.*

3.3.3. Detection of chips formations

As was mentioned above, traditional time series similarity metrics are not capable enough for unstable manufacturing domain because of time distortion degree and misalignment in temporal domain. According to [22], in order to select the right metric such properties as time complexity, normalization, capability to handle difference in time series length and robustness in warping and scaling is important.

Results, got in this paper, are consistent with the other works in this domain and show that from 12 different time series similarity measures DTW and Edit distance perform best than the other ones. DTW however has additional modifications and extensions that make it the optimal choice (see Fig. A.1, Tab. A.1).

Dynamic time warping belongs to a class of algorithms that calculate dissimilarity (i.e., distance) between two selected sequences, mapping each value in one sequence to one or more values in the second one [32].

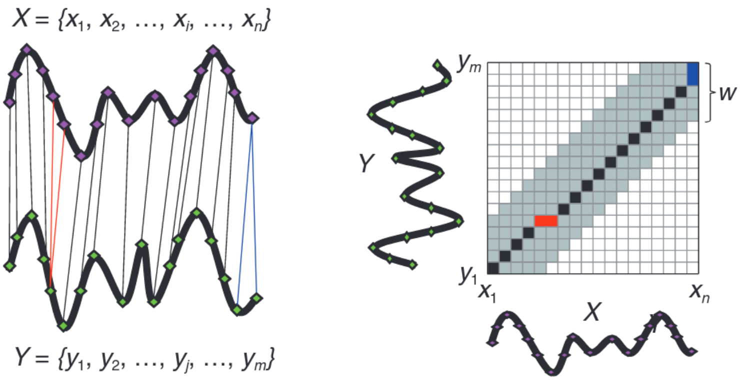
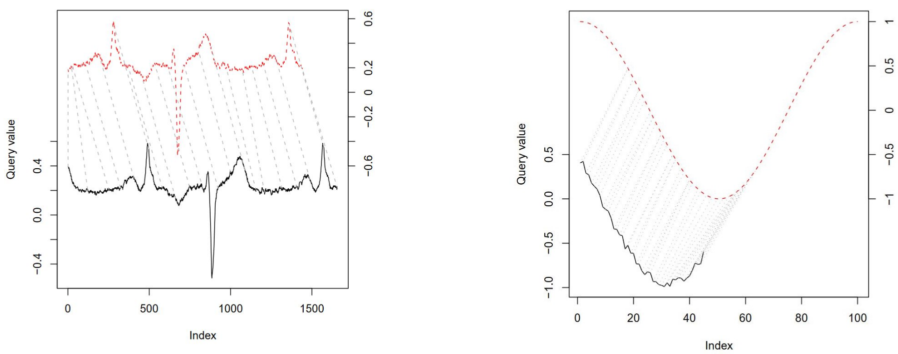


Figure 3.8.: Illustration of DTW [50]

Two time series are represented by sequences of values $X = (x_1, x_2, ...x_n)$ of length $N \in \mathbb{N}$ and $Y = (y_1, y_2, ...y_m)$ of length $M \in \mathbb{N}$. All sequences have values from some feature space. To compare two features in the scope of this space some *distance measure*, which is represented by a function, is required. If the cost of function of x and y is low, they are similar to each other, else-ways when the cost is high x and y vary to each other.

To obtain the *cost matrix* $C \in \mathbb{R}^{N \times M}$ evaluation for each pair of features of two sequences local distance measure should be done [28]. The aim is to define alignment path between two sequences with the lowest overall cost (see Fig. 3.8).



(a) DTW (b) Open Ended DTW

Figure 3.9.: DTW vs Open-ended DTW [14]

However, if the work piece was misaligned, obtained values about the surface have some deviations from the real state, which leads to the wrong labeling by classification. Misaligned time series could be mostly defined and excluded during time series alignment (see Sec. 3.3.2), but if the number of misaligned working pieces is too high inside the batch additional procedures in manufacturing should be made in order to provide stable environment for measuring process as CDP Lift (see Sec. 3.2). In case when gripper (or other tool) holds the working piece and it can't be measured completely only free segments of the working piece should be considered and open ended DTW should be used (see Fig. 3.9).

Despite the advantage of DTW by reliable time alignment between reference and test patterns, DTW has a disadvantage of heavy computational burden required to find the optimal time alignment path ($O(N^2)$ time and space complexity). In order to reduce the time complexity of DTW the FastDTW (an approximation of DTW with time complexity O(N)) cab be used, however it provides lower performance in comparison with DTW (see Fig. 3.11, 3.10).

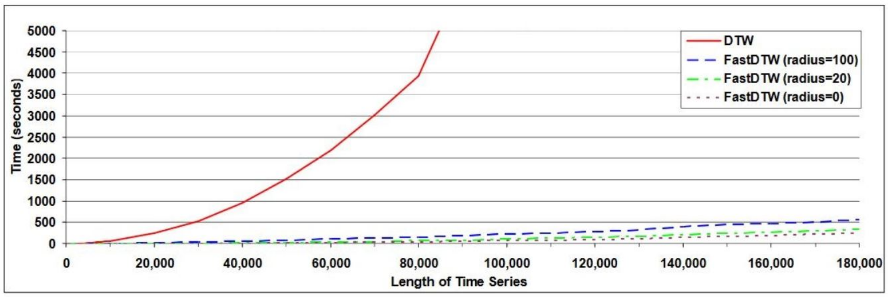


Figure 3.10.: Execution time of FastDTW and DTW [38]

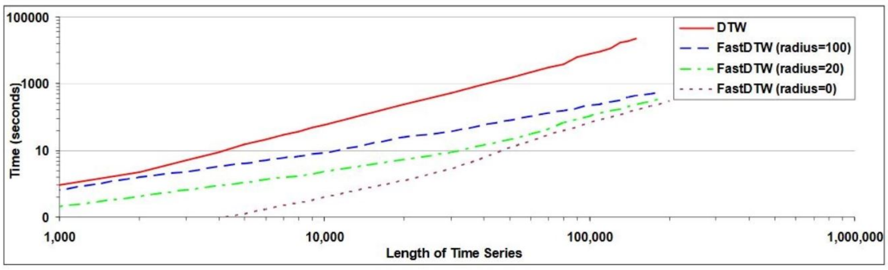


Figure 3.11.: The efficiency of FastDTW and DTW [38]

According to [43], DTW and comparison of the time series can be combined with online concept drift detection based on the *process event stream*. The state of the working piece is defined not only based on the similarity of the time series but based on drift detection in the process history.

Despite good results, obtained in the paper, this method is too sensible for general settings and has some limitations and disadvantages, as:

– if all working pieces in the batch or its majority are bad, the algorithm will still

classify them as good
- if alignment of the working pieces during measurement procedure is wrong, high rate of misclassification is present or concept drift is found, where no drift was actually
- because of wrong alignment threshold should be adjusted from batch to batch, and by good alignment is is almost impossible to define the drift, as distance in this case often equals zero
- for each type of the working piece new threshold should be defined
- if some error occurs during manufacturing and production data would be lost, it is impossible to evaluate the quality of the working piece, as this method can be used only if the process execution events are continuously collected during run-time
- no information about particular tolerances and dimensions could be done

Considering these limitations we will not use this method in the scope of this work.

3.3.4. Tolerance analysis

Data compression makes it possible to lead work piece classification faster. Nevertheless it is important to define not only the state of the part (chips/no chips), but the parameters, where the tolerances deviate from the desired output, since batch production brings variability to the process: executing the same process with the same parameters the quality of the final product can vary from batch to batch due to unobserved phenomena.

Different runs of the same recipe usually have different duration; sometimes, a batch can take a multiple of the time it took for another batch. This fact poses challenges in the production line and requires detailed information about work piece quality [15, p.18].

The tolerance analysis and the exact knowledge of the single point uncertainty of a measurement series can greatly influence the quality of the subsequent data evaluations [31, p.3505].

Based on the tolerance analysis work piece's deformations can be determined. Tolerances discrepancy and deformations are typically caused by machine, used tools and set up errors. These deviations from the original working piece form is usually affected by uncertainty, which depends on number of points measured on the profile surface and their location [6].

During each production process systematic pattern is left on the surface of the working piece (i.e., manufacturing signature). This pattern can be used to characterize how the elements of the working piece were machined along manufacturing [34]. Obtaining this information during the production allows to adjust the production process parameters amid batch running immediately or for the next batches without delays.

Even though manual measurements are clearly faster, than final QA measurements, they are usually not so precise and susceptible to human errors and leads to inconsistencies. Automatic detection whether the tolerances are kept by close-to-production measurement can optimize this operation, despite the fact, that limitations of the measurement procedure don't permit to obtain the measures for all key parameters of the work piece.

To extract tolerances from the cloud point model (i.e., even the reduced one) the segmentation of the surface profile should be performed (see Sec. 3.3.2). After segmentation necessary information that represent desired tolerance values is extracted and comparison with the original tolerances is performed. Dimensional and geometric tolerances may be verified in accordance with ISO/TR 14638[3] or other relevant standard as Y14.5M-1994[4] [54]. Working with only one view of working piece representation some simple statistical strategy for tolerance extraction as Extreme Points Selection (EPS) can be chosen, as only few points hold all needed information (i.e., essential subset) [6].

Due to high sensibility of micrometer and unstable uninsulated environment EPS could be dangerous, as event dust, dirt, grime and grease particles could influence obtained extreme values and lead to the wrong evaluations. As alternative to the EPS, considering the fact that we are evaluating flat surfaces (i.e., straight lines), the most frequently occurred value in the segment would be considered.

To be able to combine obtained values from the point cloud model with the original predefined tolerances, they should be introduces into the process. One of the opportunities to make it avoiding the manual interaction is to extract data from the engineering drawing of the working piece. Engineering drawings are 2D depictions of a work piece that include geometric as well as textual information such as measurements, tolerances, and applicable norms, which are essential for work piece production and its quality control.

As there is still no solution of integrating the textual information that is found in technical drawings into a continuous automated production process, the DigiEDraw algorithm, introduced in [39] would be used. This algorithm extracts textual elements from engineering drawings by its preprocessing, iterative clustering with DBSCAN and postprocessing.

In order automatically to determine from a list of tolerances, which measured dimensions and acclivities are corresponding to a subset of these tolerances the following approach would be suggested:

> *Extract all necessary information (boundaries) from the technical drawing of the working piece. Extract features from the surface profile, obtained after data compression, that correspond these boundaries. Based on mathematical representation of GD&T standards (cite appendix) look if extracted values are in the boundaries of tolerances from engineering drawing and each parameter is in the segment like in the original meta-model to define the status of each parameter and the state of the whole working piece. If at least one tolerance is violated the working piece is bad.*

For each segment average value of parameter could be found to define the manufacturing signature of the batch. Based on these values basic similarity between different batches, raw tools quality, adjustment and working piece alignment estimation could be fulfilled. If most of the time manufacturing signature shows the same pattern with equal deviations not only the wrong grasping, but more deep problems are the cause of this shift as gripper design, tool conditions, wrong raw material selection or even engineering error in design.

[3] Geometrical product specification (GPS) – Masterplan

[4] 1994. ANSI Y14.5M—1994. Dimensioning and Tolerancing. The American Society of Mechanical Engineers.

4. Implementation

In this chapter the implementation details for goals and solutions described in previous section are provided. As we are using process engine all services would be implemented as REST services. For this purpose we are using Python and Flask (web application framework).

Current process for working piece production, mentioned in Section 3.2, consists of 3 main stages, which include 4 common process steps (see Fig. 4.1). "MicroVu Measurement" is a sub process, which is running in parallel with the main one.

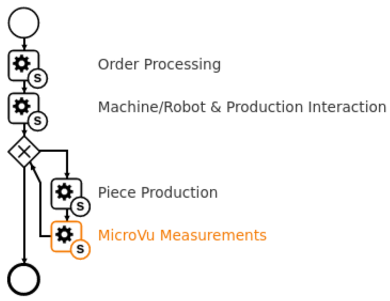
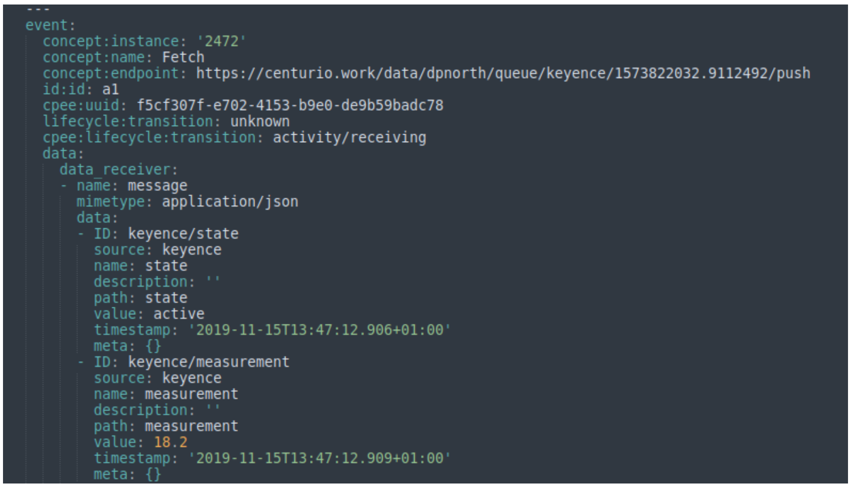


Figure 4.1.: Working piece production process

We introduce implementation of different data compression methods, chips and tolerance analysis, as well as user interface and describe how each block is connected with the other one. All these methods and practices would be added to the main process as additional steps. Nonetheless conclusions about effectiveness of the whole implemented methods would not be discussed in this section.

4.1. Data Preparation

Sensor data is stored in process execution logs, which are produced by process execution engine. Logs are represented as YAML files in the XES standard format [10] and contain list of events, where each event has production data, collected during machining and measurement procedure (i.e., instance number, event name and its id, timestamp and additional attributes relevant to the event (see Fig. 4.2)). In one log the information for one working piece is represented.



```yaml
---
event:
  concept:instance: '2472'
  concept:name: Fetch
  concept:endpoint: https://centurio.work/data/dpnorth/queue/keyence/1573822032.9112492/push
  id:id: a1
  cpee:uuid: f5cf307f-e702-4153-b9e0-de9b59badc78
  lifecycle:transition: unknown
  cpee:lifecycle:transition: activity/receiving
  data:
    data_receiver:
    - name: message
      mimetype: application/json
      data:
      - ID: keyence/state
        source: keyence
        name: state
        description: ''
        path: state
        value: active
        timestamp: '2019-11-15T13:47:12.906+01:00'
        meta: {}
      - ID: keyence/measurement
        source: keyence
        name: measurement
        description: ''
        path: measurement
        value: 18.2
        timestamp: '2019-11-15T13:47:12.909+01:00'
        meta: {}
```


Figure 4.2.: Example of process execution log with sensor data

In order to extract and simplify further analysis of sensor data i.e., measurements obtained from high precision micrometer system, only two attributes (value and timestamp) should be extracted from the log.

There are a lot of libraries, that allow to parse YAML data, however as data is too large and parse process takes to much time, a short python script is created to parse data as a text. For each line in text data we are searching for string "value:" and extract float parameter. In the next line we extract an timestamp parameter and save the pair value/timestamp to the dictionary.

As each working piece is produced in its own time frame, production speed could varies from batch to batch (or from piece to piece), pauses in production cant be excluded, different systems can have its own local time etc. the standardization of time domain is necessary.

Timestamps from the dictionary are converted to the Unix timestamps (i.e., numbers) and multiplied by 1000 as frequency of the entries is to high and difference between these entries is too low. From each entry we subtract the first converted timestamp value and transfer the time array to the array of integers from 0 to N. As we are talking about online environment during data obtaining some values can be saved not in the chronological order. All pairs value/timestamp that are not complete or have missing values should be deleted and sorted by the timestamp before further steps.

Additional filtering and cleaning is avoided to save generic manner of the measurement procedure.

4.2. Data Compression

Data compression is simplification of cloud point data to reduce the redundant information. In Section 3.3.1 we describe the importance of this step and suggest five different approaches for data compression and their possible advantages and disadvantages or limitations.

However, primitive data reduction and edge point detection is implemented already during measurement procedure, that influences the structure of the logs, data in it and its amount.

Each two points, obtained by micrometer, are compared with each other based on the defined threshold. If difference between two points is equal to zero new point won´t be saved, else if the difference is not equal zero but is less than the threshold the new point is added to the log. In other case, when the difference is equal or larger than the threshold not only the new point is added, but additional value "999.99" is added. These values symbolised the potential edge corners of the working piece.

4.2.1. Approach 1

For obtained YAML log proceed at first data preparation script described in the previous subsection. For each pair of values, check if it is not the edge point or not the last pair in the log (first and last pairs in the log are edge points as well, but without additional "999.99" marking). Save all the pairs in the temporal list. As long as the pair belongs to the exceptions (edge point/last pair) perform linear regression and get predicted values. Save the first and the last values of the timestamps and predicted values.

```
Input  : log : YAML log
Output : compressed_data : array with compressed data
pairs = data_preprocessing(log);
length = length(pairs) - 1;
regression_list = [ ];
compressed_data = [ ];
for pair, indexinpairs do
    if pair.value! = 999.99andindex! = pair_length then
        regression_list.append([pair.timestamp, pair.value]);
    else
        temporalDF = convert_to_dataframe(regression_list);
        x = temporalDF["timestamp"];
        y = temporalDF["value"];
        linear_regression = fit(x, y);
        y_predicted = linear_regression.predict(x);
        compressed_data.append([x.first, y_predicted.first]);
        compressed_data.append([x.last, y_predicted.last]);
        regression_list.clear();
    end
end
return compressed_data;
```

Algorithm 1: Data Compression. Approach 1

Under linear regression we understand the simple linear model with two variables, where explanatory variable is a timestamp and response variable is measured value. For this implementation python library `scikit-learn`[1] is used.

Multidimensional array with compressed data always have the even number of pairs. Each 2 pairs represent the segment of the working piece between two edge points. Compressed data allows to reestablish the original profile surface and could be used for further purposes as chips or tolerance analysis or visual reports (see Fig. 4.3).

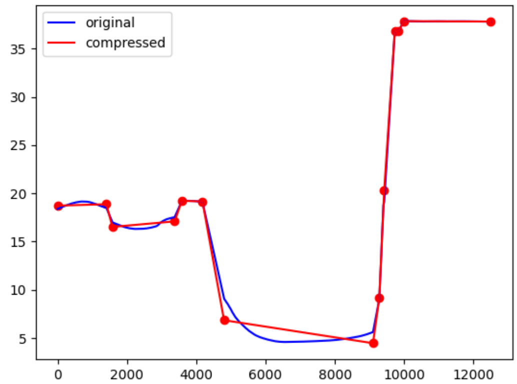


Figure 4.3.: Data Compression. Approach 1

4.2.2. Approach 2

Before the call of the method the N parameter and its type should be define. N parameter is an integer, which represents the size of the area and type is a string which defines whether we split up the working piece by measured values or by timestamps ("val","ts").

For obtained YAML log proceed at first data preparation script described in the previous subsection. As we are not interested in the potential edge points before further operations we delete all "999.99" values.

For each pair check the list of conditions (see line 8,10) and append current pair to the temporal list. As far as some of the conditions can't be completed increment N value for the next iteration, perform linear regression and get predicted values. Save the first and the last values of the timestamps and predicted values to the final multidimensional array, which would have the even number of pairs.

As in the previous approach, linear regression is linear model with two variables (independent: timestamp and dependent:value) and the same python library is used.

We should notice, that despite split of surface to the equal areas with predefined length, last segment is always smaller then defined N value and from piece to piece has different

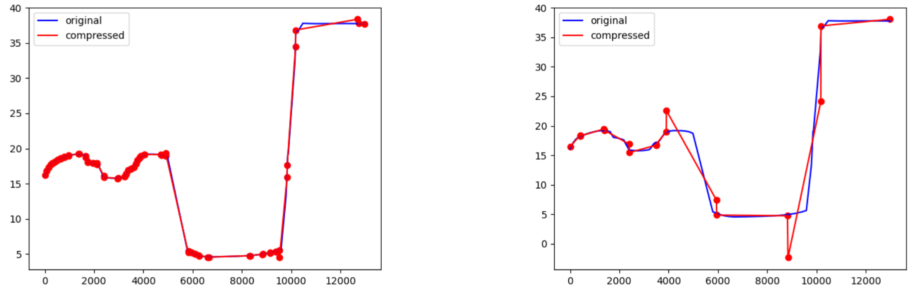
[1] https://scikit-learn.org/stable/

length. Equal length of other segments is nominal, i.e., the same amount of points in each segment covers different time spans and vice versa. Also, segmentation will vary from working piece to working piece, as each part is measured with approximate, but not identical time, start point of measurement varies, chips, misalignment and unstable environment influence quality and duration of measurement procedure and obtained data.

```
Input  : log : YAML log,
         N : size of the segment,
         type : type of split up variable
Output : compressed_data : array with compressed data
pairs = data_preprocessing(log);
pairs = delete_999(pairs);
pair_length = length(pairs) - 1;
regression_list = [ ];
compressed_data = [ ];
count = N;
for pair, index in pairs do
    if type == "val" and index != count and index != pair_length then
        regression_list.append([pair.timestamp, pair.value]);
    else if type == "ts" and pair.timestamp < count and index != pair_length
     then
        regression_list.append([pair.timestamp, pair.value]);
    else
        count = count + N;
        temporalDF = convert_to_dataframe(regression_list);
        x = temporalDF["timestamp"];
        y = temporalDF["value"];
        linear_regression = fit(x, y);
        y_predicted = linear_regression.predict(x);
        compressed_data.append([x.first, y_predicted.first]);
        compressed_data.append([x.last, y_predicted.last]);
        regression_list.clear();
    end
end
return compressed_data;
```

Algorithm 2: Data Compression. Approach 2

This approach is very sensible to its thresholds. If threshold is too small, data compression rate is not significant, and by large value of the threshold a lot of surface details could be lost (see Fig. 4.4).



(a) N=20, value axis

(b) N=100, value axis

Figure 4.4.: Data Compression. Approach 2. Threshold influence

4.2.3. Approach 3

As was mentioned earlier, for this method working piece specification is required (see Fig. 4.5). To divide the surface profile we find the value (horizontal line), which has the most crosses with current surface. For each area of interest define, which extreme values should be extracted. Quartiles are used when only some area of the segment is of interest and not the whole one.

```
{
    "split": 19.1,
    "extreme": {
        "1": "max",
        "2": "min",
        "3": "max",
        "4": "min"
    },
    "quartile": {
        "2": [
            37,
            75
        ],
        "4": [
            37,
            75
        ]
    }
}
```

Figure 4.5.: Example of working piece specification

To convert log data into desired format we proceed data preparation script and delete all "999.99" values. Additional preparation step is to extract the information from specification.

After data preparation call the function that will find all pairs, where profile surface cross the border line from specification (see line 6 A. 12). Based on these pairs split the whole profile surface on some number of segments.

After successful segmentation find the number of resulting segments. If quartiles are defined in the specification, for each mentioned segment count the values of both quartiles based on the time axes and append the new segment with values that are between two quartiles to already existing list (see line 11 A. 13).

For each segment create linear regression model and append first and last pair of obtained data to the final array (see Fig. 4.6).

Input : *log* : YAML log,
spath : location of the specification data
Output : *compressed_data* : array with compressed data

```
1  regression_lists = [ ];
2  compressed_data = [ ];
3  pairs = data_preprocessing(log);
4  pairs = delete_999(pairs);
5  specification = extract_info(spath);
6  indexes = cross_points(pairs,specification.boarder);
7  for a in range(0,length(indexes),2) do
8      regression_lists.append(pairs[indexes[a]:indexes[a+1]]);
9  end
10 number_segments = length(regression_lists);
11 get_quartiles(regression_lists, specification);
12 for list in regression_lists do
13     x = list["timestamp"];
14     y = list["value"];
15     linear_regression = fit(x, y);
16     y_predicted = linear_regression.predict(x);
17     compressed_data.append([x.first, y_predicted.first]);
18     compressed_data.append([x.last, y_predicted.last]);
19 end
20 compressed_data = sort(compressed_data);
21 return compressed_data;
```

Algorithm 3: Data Compression. Approach 3

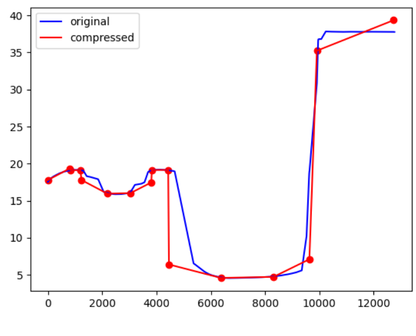


Figure 4.6.: Data Compression. Approach 3

As all three approaches use linear regression, they could be extended and additional data could be collected for further analysis. We can calculate slope, intercept and the number of defined segments. Slope shows us how much dependent variable (measured value) will change as independent variable (timestamp) increases and despite there are no clear dependency between timestamps and measured values, as each working piece type is different, we can use this parameter to define how straight (i.e., parallel to the x axis) surface profile segment is laying.

Intercept is the mean value of the dependent variable at the moment when independent variables are equal to zero, i.e., starting point of measured value. This data could be used to define whether the part is misaligned, or chips formations are present.

The number of defined segments could be used, to check if good and current time series have equal structure.

4.2.4. Approach 4

Before the call of the method the intervals (steps) for the grid frequency should be defined.

For obtained YAML log proceed at first data preparation script described in the previous subsection. As we are not interested in the potential edge points before further operations we delete all "999.99" values.

Next step is to get all cross points between profile surface and our grid in both directions. For this purpose separate function for one particular direction with 3 parameters is created (see A. 14). Basically, we find the minimum and maximum values of profile surface and based on our predefined step count the planed number of lines in the grid. Split variable is the value of the first grid line, which would be incremented later.

For each line we check whether there are some cross points or not by comparing split value with the current and the next value of the surface profile and as both points could be pretty similar to avoid redundancy we keep both points only if their difference is considerable.

After that, cross points from both directions should be overlaid and all duplicates, that could occur, should be removed (see Fig. 4.7).

Despite simplicity of this approach , we expect that the data would still look like data point cloud, however with the significant volume reduction. How successful this data could be applied for further analysis, whether some additional operations are necessary for adequate results, would be discussed in the next chapter.

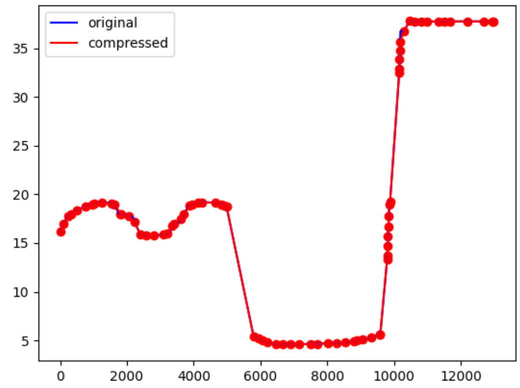


Figure 4.7.: Data Compression. Approach 4

Input : *log* : YAML log,
hor_step : step for horizontal grid,
vert_step : step for vertical grid
Output : *compressed_data* : array with compressed data

$pairs$ = data_preprocessing(log);
$pairs$ = delete_999($pairs$);
$compressed\_data = [\,]$;
$time\_h, value\_h$ = create_grid($pairs, hor\_step$, "$value$");
$time\_v, value\_v$ = create_grid($pairs, vert\_step$, "$timestamp$");
x = concat($time\_h, time\_v$);
y = concat($value\_h, value\_v$);
$sorted$ = sort($[x, y]$);
$compressed\_data$ = remove_duplicates($sorted$);
return $compressed\_data$;

Algorithm 4: Data Compression. Approach 4

4.2.5. Approach 5

Before the call of the method the intervals for the grid and threshold for the time delta should be defined and data preparation should be done.

In comparison with the previous approach, only horizontal grid is used and not the pairs themselves, but their indexes are collected. To find all pairs, which cross the horizontal grid and their indexes we are using similar to `create_grid` function with 2 parameters (see line 4 A. 15).

Next step is a while loop, that would be executed until last index is less than the number of pairs. During each iteration calculate the absolute difference between current and next measured value and timestamp.

If value delta is less than the grid step call function that starts the process of point selection from the cloud model (see line 15 A. 16).

```
Input   : log : YAML log,
          hor_step : step for horizontal grid,
          ttx : threshold for time axis
Output  : compressed_data : array with compressed data
compressed_data = [ ];
pairs = data_preprocessing(log);
pairs = delete_999(pairs);
(glob) indexes = index_grid(pairs, hor_step);
data_length = length(pairs.value) - 1;
(glob) a = 0;
(glob) extreme = [ ];
if ttx==0 then
    min_value = int(min(pairs.value)) -1 ;                    // extension
    max_value = int(max(pairs.value)) +1;
    difference = max_value - min_value;
    ttx = find_threshold(difference,"timestamp");
end
while indexes[a] < data_length do
    ci = indexes[a];
    ni = indexes[a+1];
    delta_value = abs(pairs.value[ci]-pairs.value[ni]);
    delta_time = abs(pairs.timestamp[ci]-pairs.timestamp[ni]);
    area = pairs.value[ci:ni];
    if delta_value < step then
        select_points(area, ttx, pairs, delta_time);
    else
        select_points(area, ttx/10, pairs, delta_time);
    end
    a = a + 1;
end
indexes = sort(concat(indexes,extreme));
for i in indexes do
    compressed_data.append(pairs.timestamp[i],pairs.value[i]);
end
return compressed_data;
```

Algorithm 5: Data Compression. Approach 5

If time delta is equal or larger than the defined threshold find extreme values of this area and add it to the temporal array, otherwise call recursive function that will check time delta between next two points and remove all pairs, which are too close to each other. Else point selection is called as well, however with the ten times smaller threshold (see line 17 A. 17).

In the first case we can get rid of redundant points which on the flat and smooth areas of the working piece, in the second one we can avoid unnecessary points on vertical areas of the working piece.

After while loop condition becomes false we concatenate remains indexes with obtained extreme values, sort and append all the data to the final compressed array (see Fig. 4.8).

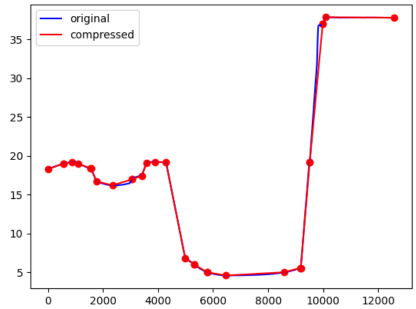


Figure 4.8.: Data Compression. Approach 5

Approach 4 and 5 could be extended with automatic threshold definition. For this purpose we create a simple function with 2 parameters (distance and direction). Direction is an optional parameter that is set for horizontal grid per default.

To include this function to the approach 4 (see Alg. 4) input parameters will not be changed, but such parameters as horizontal and vertical step will be optional with default value 0. Inside of `create_grid()` function if step is not defined by user the function `find_threshold()` would be called (see line 7 A. 14).

```
Function find_threshold(difference, direction="value"):
    if direction == "value" then
        if difference > 10 then
            step = 1;
        else if difference <= 10 and difference > 1 then
            step = 0.5;
        else
            step = 0.1;
        end
    else
        step = int(difference/50);
    end
    return step;
```

Algorithm 6: Function to define thresholds automatically

To extend approach 5 step and threshold parameters would be changed to the optional as well. As `index_grid()` is a function which creates only horizontal grid, function `find_threshold()` with one parameter (difference) would be added to, if user didn't set the step himself (see line 7 A. 15).To calculate `ttx` threshold we add `find_threshold()` for vertical direction before while loop (see line 8 Alg. 5).

Advantage of such extension is that user don't need to predefine the thresholds obligatory, but at the same time can still tune the function with own values or combine predefined thresholds with automatically calculated to achieve better results.

All approaches have log as an input, some approaches require additional thresholds. Output values are always the same, as the idea is to have an opportunity to select the most effective one from the list of alternatives. It is also possible, that different approaches are not always equally competent based on the type and shape of the working piece, that's why it is important to be able to interchange the approaches without changing the structure of the process. It could be detected that some approaches are not reasonable or not effective at all or they provide a good data compression, however this data can't be used for further purposes.

4.3. Time Series Synchronization and Chips Formation

In order to define whether there are chips formations or none two time series need to be compared. Under two time series we understand the basic one: time series, that represents the good working piece without chips formation as a foundation for future comparison and the current one: profile surface obtained during production step.

Most of the time series have similar duration but different length, especially after data compression. There are a lot of libraries dedicated to time series analysis. Based on the systematic review [41] of these packages we selected the `tslearn`[2] library, as this package support preprocessing, similarity measures, clustering and classification together and has adequate documentation.

We will use 3 methods for time series synchronization, two of them are provided in `tslearn.preprocessing`. One of them is resampling (TimeSeriesResampler), where variable-length time series are transformed into the time series with the same length. Between two compared time series we select the current one and resample it to the size of the constant one.

```
resampled_ts = TimeSeriesResampler(sz=new_length).fit_transform(time_series)
```

After resampling we can scale time series such that each output time series has zero mean and unit variance, as we suppose that the range of a given time series is uninformative and we want to compare only shapes (i.e., surface profile) of the working piece [46].

```
scaled_ts = TimeSeriesScalerMeanVariance(mu=0,std=1).fit_transform(time_series)
```

[2]https://tslearn.readthedocs.io/en/stable/

In the library documentation it is mentioned, that both these methods don't solve the problem with distortion over time axis and can bring some additional desynchronisation.

Also during production the same working pieces could be measured differently: not only duration and length of the procedure can vary or environmental conditions can influence the structure of obtained profile surface, but the start point of the measurement procedure could be mismatched.

The principal of working piece segmentation is taken from Sec. 3.3.1. However, to avoid the necessity of specification we implement the segmentation method, described in the Sec. 3.3.2.

```
Function find_sline(data, direction="value",step = 0):
    segment_candidates = create_segmentation_grid(data, direction, step=0);
    segment_candidates = segment_candidates.sort(by=count);
    split_line, segment_number, areas = max(segment_candidates);
    return split_line, segment_number, areas;
```

Algorithm 7: Function to find segmentation line

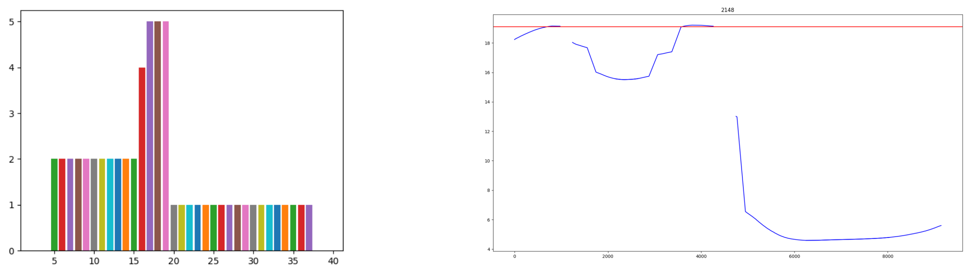
At the beginning we create a horizontal grid and instead of saving the cross points count the number of cross points for each line in the grid (see Fig. 4.9(a)). Save the line value and how often the line would be crossed by the profile surface. Then select that line, which has the maximum number of crosses and the largest value of line (see Fig. 4.9(b), A. 18).

Segmentation function will be used to achieve maximum synchronisation between two time series to reduce time distortion degree. As we can see in Alg. 8 we take two time series (the original one and the current one) as an input and get two new time series and the status as an output. Than we call our segmentation function and check if we can perform the synchronization at all. If the value of the segmentation line varies or the number of segments for both time series is different that means that chip formation or an error is occurred during the measurement procedure. The status variable will be set to "false" and we can't do any further analysis.

Otherwise for each area in both time series calculate their length. If length is same in both cases just append these areas to the output variables, else refactor the shortest area based on the length of the largest one and rechange this area.

Evaluation of these methods will be done by Time Alignment Measurement and during use of Dynamical Time Warping and based on these results we will determine if synchronization is useful in this case and if yes, the best method or the combination of them would be found.

After time series alignment we need to determine how similar two time series are to each other. To calculate the optimal path and the distance between two sequences we will use `tslearn` library as well. In the `metrics` package we have an implementation of dynamical time warping with squared Euclidean distance as the base metric in the optimization problem.



(a) Number of cross points for each horizontal line

(b) Found segmentation line

Figure 4.9.: Segmentation procedure

```
Input  : good_ts :good time series,
         cur_ts : current time series
Output : resampled_good : resampled good time series,
         resampled_cur : resampled current time series,
         status : bool value
resampled_good = [ ];
resampled_cur = [ ];
status = False;
cur_split, cur_segment, cur_areas = find_sline(cur_ts);
good_split, good_segment, good_areas = find_sline(good_ts);
if cur_split == good_split and cur_segment == good_segment then
    for a in range(0,cur_segment+1) do
        len_cur = length(cur_areas[a]);
        len_good = length(good_areas[a]);
        if len_cur > len_good then
            good_areas[a] =
             TimeSeriesResampler(sz=len_cur).fit_transform(good_areas[a]);
        end
        if len_cur < len_good then
            cur_areas[a] =
             TimeSeriesResampler(sz=len_good).fit_transform(cur_areas[a]);
        end
        resampled_good.append(good_areas[a]);
        resampled_cur.append(cur_areas[a]);
    end
    status = True;
end
return resampled_good, resampled_cur, status;
```

Algorithm 8: Time series synchronization

```
dtw_score = dtw(x, y);
optimal_path, dtw_score = dtw_path(x, y);
```

However, the approximation of the DTW - FastDTW is not supported by this library. In order to apply this algorithm we will use `fastdtw`[3] library.

```
optimal_path, dtw_score = fastdtw(x, y, dist=euclidean);
```

To decide if there are some chips formations the DTW distance should be evaluated and some threshold should be defined by expert.

```
Input  : good_ts :good time series,
         cur_ts : current time series,
         dtwth : threshold for DTW,
         mode : "dtw" or "fastdtw"
Output : status : bool value
status = False;
if mode == "dtw" then
    optimal_path, dtw_score = dtw_path(good_ts, cur_ts);
else
    optimal_path, dtw_score = fastdtw(good_ts,cur_ts, dist=euclidean);
end
if dtw_score<=dtwth then
    status = True;
end
return status;
```

Algorithm 9: Dynamic Time Warping

Interesting is, whether for both methods different threshold is necessary or only one could be defined and if there are some advantages or disadvantages between DTW and FastDTW.

4.4. Tolerance Analysis

To determine whether produced working piece is good or not, tolerances mentioned in technical drawings should be provided. These tolerances are reference points for evaluation of the current produced parts and their extraction should be done after definition of planned batch size and orchestration of all machines and robots involved in the manufacturing and measuring.

In order to use `DigiEDraw` algorithm we download the github repository[4] and call main

[3] https://pypi.org/project/fastdtw/

[4] https://github.com/DigiEDraw/extraction

function with 4 parameters:

- uuid = unique identifier consisting of up to 4 numbers (set randomly)
- dwpath = path to the PDF file
- db = redis parameters e.g., "localhost"
- eps = (optional) a custom EPS value (by default set to 1)

```
td_tolerances = digiedraw.main(uuid,dwpath,db,eps);
```

After tolerances extraction we get a json body with the list of resulting key value set, where keys, as "0.00 19.20 - 0.05 ", represent dimensions of the working pieces and values, as [451.8948, 226.069084, 470.8308, 266.389049], represent associated coordinates of the bound box used for clustering purposes (see Fig. 4.10) [39].

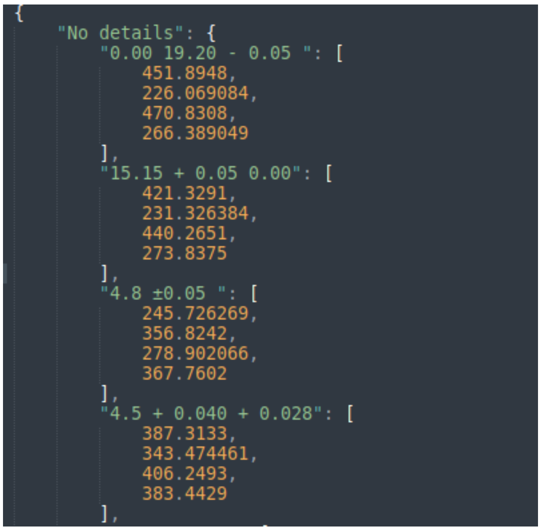


Figure 4.10.: Example of DigiEDraw Algorithm Output

As we are not interesting in the bound boxes we create an array, where only keys are present. However, the structure of each dimensional entry is not consistent, i.e., actual value and its tolerances occur on different places, there are spaces between mathematical signs and the value itself or there is nothing and etc. These inconsistencies make it complicated to automatically calculable final boundary dimensions for final tolerance analysis.

Also, we obtain only dimensioning and size tolerances, but we have no additional information , that is present on technical drawing (i.e., tolerance type or how often particular tolerance appears). As we are interested only in tolerances that represent diameter, user interface is provided, where user should select manually, which tolerances would be considered for further analysis.

After user select necessary dimensions, we should call the function that clean the data to avoid these inconsistencies and to get final boundaries.

In each key in the array extract all floats and if the list of floats is not empty find its maximum value. If the length of the list is 2, that means that we have one actual value and its upper and lower tolerance is the same but with the opposite sign like this: "4.8 ± 0.05". We find then the minimum value of the list of floats and calculate the upper and lower bound based on minimum and maximum (see line 9 Alg. 10).

```
1  Function get_boundaries(dimension_list):
2      boundaries = [ ];
3      for dimension in dimension_list do
4          floats= extract_float(dimension);
5          if floats nor empty then
6              maxv= max(floats);
7              if length(floats) == 2 then
8                  minv= min(floats);
9                  boundaries.append([maxv+minv,maxv-minv]);
10             else
11                 temp = [ ];
12                 for f in floats do
13                     if f != maxv then
14                         if f == 0 then
15                             temp.append(maxv);
16                         else
17                             sign = get_sign(dimension);
18                             if sign == "+" then
19                                 val = maxv + f;
20                             else
21                                 val = maxv - f;
22                             end
23                             temp.append(val);
24                         end
25                     end
26                 end
27                 boundaries.append([max(temp),min(temp)]);
28             end
29         end
30     end
31     boundaries.round(boundaries,3);
32     return boundaries;
```

Algorithm 10: get_boundaries function

If the length of the list is equal to 3, then for each element in the list of floats if they are not equal the maximum element we check if it is zero. If the element has zero value we add it to the temporal array, otherwise we extract from the key 2 characters before the tolerance to check the sign. Based on the sign add or extract the tolerance value from the maximum one and add it to the temporal array. For temporal array we find the minimum and maximum, which are our upper and lower bound (see line 27 Alg. 10).

As a result we get structured array with upper and lower boundaries for tolerances, which are relevant to the further analysis:

```
[[19.2 19.15],[15.2 15.15],[ 4.54 4.528]]
```

To find out if current working piece corresponds its dimensional requirements (i.e., boundaries) we need to extract extreme values from particular areas of the working piece. For this purpose we call the `find_sline()` function and split profile surface into the segments.

To extract extreme values for each segment find mean value and if mean value is lower than segmentation line calculate the minimum value of the segment, otherwise - maximum and save that to the array (see A. 19). To extract the most frequent value in the segment calculate the difference between timestamps and select the maximum time difference and corresponded measured value.

As we want to avoid the use of specifications, we need to get the list of the relevant segments that should be good. To reach this goal tolerance analysis algorithm have two modes. One is to define this ideal list of tolerances for our good working piece and the second one is to compare list of relevant tolerances of the current working piece with the ideal one (parameter "ideal" is optional).

By comparing of two lists (ideal and current one) we look not only if the length of two lists is equal, but additionally whether the keys in both lists are identical. Assuming that chips formations are present and were bypassed during previous steps wrong segments could be defined as relevant, this check is necessary to ensure that sequence of the keys and keys themselves are identical in relevant and ideal list (i.e., no chips, errors and some side effects are present).

Same dimensions could occur multiple time on different segments, that's why numeration of segments is also essential for manufacturing signature to define which of the dimensions are violated and which not.

```
Input  : extremes : current time series,
         boundaries : dimensions of tolerances,
         ideal : list of desired tolerances (optional)
Output : relevant : list of good tolerances,
         status : status of working piece(good/bad)
relevant = [ ];
status = true;
for b in boundaries do
    if extreme >= b.low and extreme <= b.up then
        relevant.append(extreme);
    end
end
if no ideal then
    return relevant;
else
    if length(ideal)!= length(relevant) then
        status = false;
    else
        for key in relevant do
            if key != ideal.key then
                status = false;
            end
        end
    end
    return relevant, status;
end
```

Algorithm 11: Tolerance analysis

4.5. Rest API and General Workflow

In order to combine all the steps, that were described above, in one process we extend the current process of working piece production (see Fig. 4.1). Each step in the process engine has defined endpoint, where resources for needed functionality are located.

All endpoints will be implemented with the help of the REST API in Python using Flask. This way of communication is selected, as it is easy interchangeable and extendable.

4.5.1. Rest Services

In this section all available endpoints and their functionality would be described. Each service would be presented as table (see Tab. 4.1).

Method	endpoint
Parameters	
Response body	
Success Response	**Error Response**

Table 4.1.: Documentation template

First row, left cell shows which method should be used, right cell shows the endpoint itself. Second row demonstrates if there are some parameters needed. Parameters can be passed as JSON body or as arguments in the url. Third row show if there is some response body. In our implementation response body is always in the JSON format. In fourth row standard status codes are listed, to show the results of a client's request.

POST	../keyence/initialization
Params: list of all process variables	
Response body: {"message":"Process is initialized"}	
Success Response: 200 OK	**Error Response: 404 NOT FOUND**

`keyence/initialization` endpoint is required for instantiation of a process instance in a process engine. User should pass the list of parameters, that should be defined before process can be started like batch size, type of data compression, alignment and dynamical time warping and additional information. If more than one process engine is used, than the type of process engine should be specified as well.

Inside this request another POST request is done that sends data to process engine. All data, that is needed for instantiation of process instance in selected process engine should be found in its documentation.

GET	../keyence/gettolerances
Params: {"file":"file name","techpath":"path to file"}	
Response body: {"tolerances":"list of all tolerances"}	
Success Response: 200 OK	**Error Response: 404 NOT FOUND**

`keyence/gettolerances` endpoint returns the list of all tolerances, extracted from technical drawing. Currently, it is not possible to select automatically relevant tolerances by their type. In the future, one more additional parameter should be considered as input (type of tolerance) to return the list of only relevant boundaries.

GET	../keyence/boundaries
Params: {"tolerances":"list of all tolerances"}	
Response body: {"boundaries":"list of all boundaries"}	
Success Response: 200 OK	**Error Response: 404 NOT FOUND**

`keyence/boundaries` endpoint returns the list of all boundaries for tolerances, extracted from technical drawing.

GET	../keyence/datapreparation
Params: {"log":"log name","path":"path to log"}	
Response body: {"pairs":"list of transformed pairs value/timestamp"}	
Success Response: 200 OK	**Error Response: 404 NOT FOUND**

`keyence/datapreparation` endpoint converts YAML log to JSON dictionary of pairs (timestamp,value) and some additional information as QR code of current working piece, instance number etc.

GET	../keyence/datacompression
Params: {'pairs':'list of pairs','approach':'selected approach'} *(optional):* for some approaches additional parameters are required	
Response body: {"compressed_data":"list of pairs after compression"}	
Success Response: 200 OK	**Error Response: 404 NOT FOUND**

`keyence/datacompression` endpoint perform data compression for the list of pairs obtained after data preparation. To select desired approach string "dcNumber" (i.e., "dc1") should be entered. After selection of compression approach additional parameters should be added to the list of parameters.

For `approach 2` two parameters should be added: "n" as integer (number of points in one segment) and "type" as string (to declare the axis of segmentation: "val" or "ts").

For `approach 3` one parameter should be added: "spath" as string (path to the specification).

`Approach 4,5` could be called without further parameters. In this case necessary thresholds would be calculated automatically. Otherwise two parameters are required: "hstep" and "vstep" as integers to define the steps for the grids to proceed the segmentation.

If user wants to pass only one of the parameters and other one wants to determine automatically, this parameter should be passed with value 0.

GET	../keyence/segmentdefinition
Params: {"boundaries":"list of relevant boundaries","good":"good time series after compression"}	
Response body: {"ideal":"list of good tolerances and their segments"}	
Success Response: 200 OK	**Error Response: 404 NOT FOUND**

`keyence/segmentdefinition` endpoints is used to obtain the list of all relevant segments and their tolerances for a good (basic) time series. As input two parameters are considered. These parameters are results of two previous endpoints: `keyence/boundaries` and `keyence/datacompression`.

GET	../keyence/alignment
Params: {"type":"type of alignment","current":"current time series after compression","good":'good time series after compression"}	
Response body: {"current":"current time series after alignment","good":"good time series after alignment","status":"status if further analysis possible"}	
Success Response: 200 OK	**Error Response: 404 NOT FOUND**

`keyence/alignment` endpoint is required to prepare data to the next step (dynamical time warping) and to align time series. There are 4 types of alignment:

- "dono" : prepare the data for DTW but make no alignment
- "res" : perform resampling and prepare the data for DTW
- "scal" : perform resampling, scaling (zero mean and unit variance) and prepare it for DTW
- "syn" : synchronise time series and prepare it for DTW

Status is a boolean variable, that in 3 first methods will be always "true". However, if synchronization is not possible according to algorithm restrictions, status will be "false" and that means that either chips formations are present or some error during data collection is happened.

GET	../keyence/dytiwa
Params: {"type":"type of time warping","current":"current time series after alignment","good":'good time series after alignment","dtwth":"distance threshold"}	
Response body: {"distance":"time series similarity value","status":"status for chips formations"}	
Success Response: 200 OK	**Error Response: 404 NOT FOUND**

`keyence/ditiwa` endpoint is necessary for Dynamical Time Warping. As input two time series are required (both should be prepared in previous step) and threshold. DTW distance would be compared with the threshold and the status of the working piece would be defined. If status is "true" than the current working piece has no chips formations and could be considered for further analysis, otherwise status is set to "false".

Currently 3 types of time warping are supported: `"dtw"`, `"fastdtw"` and `"ctw"`.

GET	../keyence/toleranceanalysis
Params: {"boundaries":"list of relevant boundaries","current":"current time series after compression","ideal":'list of good segments for control"}	
Response body: {"result":"list of good tolerances","status":"quality state of working piece"}	
Success Response: 200 OK	**Error Response: 404 NOT FOUND**

`keyence/toleranceanalysis` endpoint is used to define which dimensions and acclivities are corresponding to the list of relevant tolerances. As the output the list of corresponding segments and their tolerances is represented and the status of the working piece is defined. If at least one dimension violates the tolerances the status of the working piece is "false", otherwise is set to "true".

GET	../keyence/logging
Params: {"content":"list of values to be saved","logging":"path to logging file"}	
Response body: {"message":"Logging was successful"}	
Success Response: 200 OK	**Error Response: 404 NOT FOUND**

`keyence/logging` endpoint is used to save the information used or obtained during the process in desired location. Content is extended with current timestamp.

GET	../keyence/manufacsig
Params: {"logpath":"path to log files directory"}	
Response body: {"manufacsig":"Manufacturing signature for existing logs"}	
Success Response: 200 OK	**Error Response: 404 NOT FOUND**

`keyence/manufacsig` endpoint returns a dictionary with manufacturing signature values and number of their respective segments for existing in the folder log files.

4.5.2. General Workflow

Before start user initializes new process instance with Main Production Process with the help of `keyence/initialization` endpoint and passes all necessary parameters into the process instance.

All other rest services described above will be integrated into already existed production process. Each step in the process is is either subprocess or service call.

In comparison to the already existing production process (see Fig. 4.1) we add 5 steps for extracting tolerances from the technical drawing, converting them to the boundaries, data preparation and compression for good working piece, and defining relevant segments based on good working piece (see Fig. 4.11). Data preparation and Data compression are optional steps, only for the case if we already have necessary data in the old format.

Preparation of the meta model happens only once before actual manufacturing of the working pieces is started. Manufacturing runs without changes like in the previous process, however after it is finished instead of MicroVu measurements Keyence measurements are collected. After this step data obtained from Keyence, good time series in required format, type of compression, type of alignment etc. are sent to the "Quality Assurance" subprocess.

In "Quality Assurance" subprocess data preparation and compression of current time series and time series alignments are done. If returned status after time series alignment is false, current working piece should be thrown away, sub process is ended and production of the next piece should be started. The same sequence is relevant for status obtained after dynamical time warping and tolerance analysis (see Fig. 4.12).

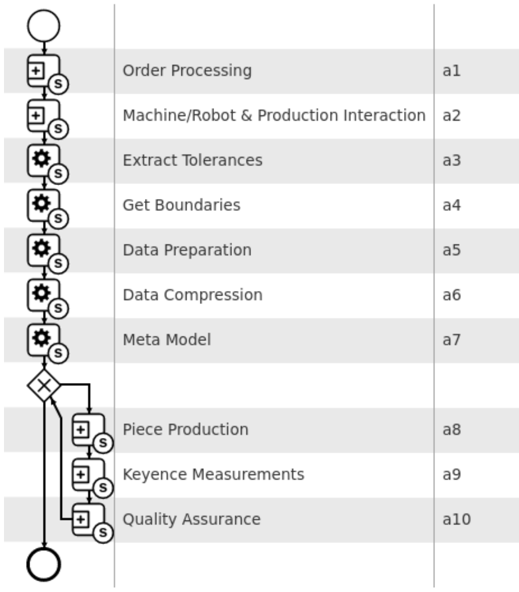


Figure 4.11.: Main Production Process

Independently of selected process engine list of input, output parameters and sequence of steps should stay the same.

After initialization and during process running user can see the results saved during logging for each working piece. Automatic updates are sent with the help of server push technology Server-Sent Events (SSE). Each time, when logging folder is updated, server sent a message to a subscribed stream(see Fig. 4.13).

On the right side the list of all logs, that are currently available in the logging directory, is shown. One log represents one working piece, data is stored in JSON format. On the left side whole information, that was stored during subprocess "Quality Assurance" about one particular working piece is displayed.

Information is separated into three blocks. First block is a table, which shows all segments, that are involved in the analysis and their status. Second one is a block with different visual charts and the third one includes table with all the parameters involved in the process.

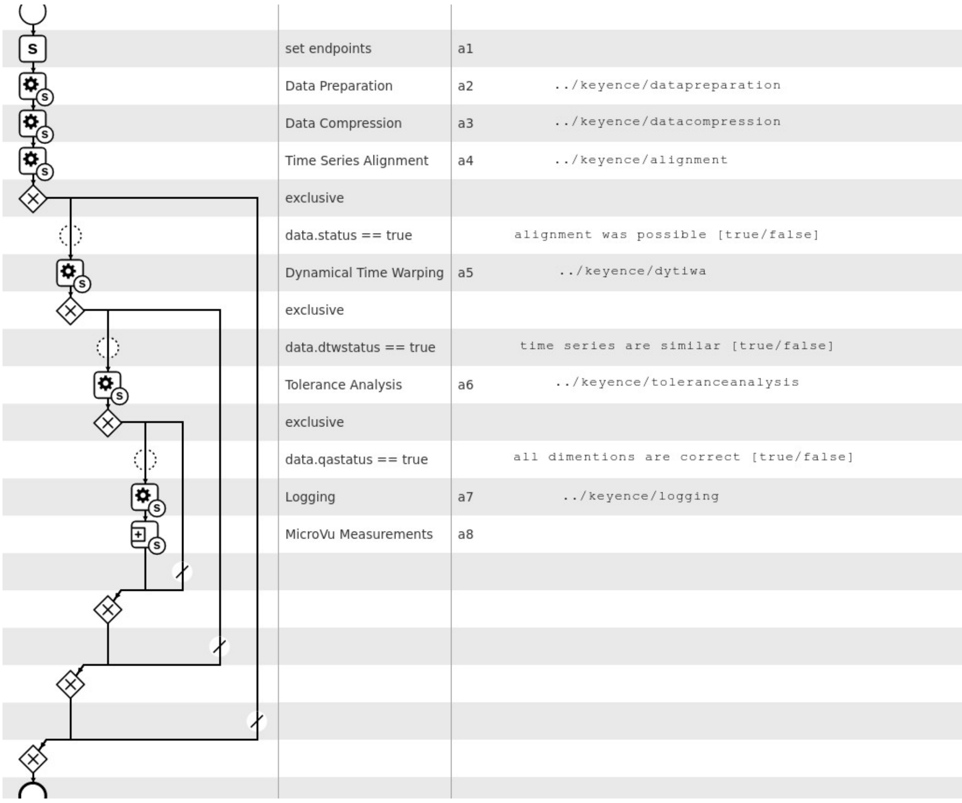


Figure 4.12.: Quality Assurance Subprocess

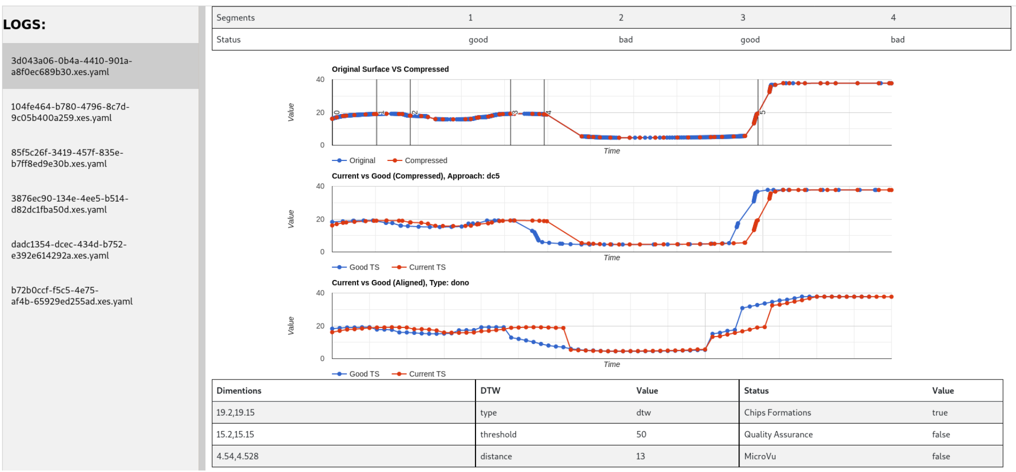


Figure 4.13.: Dashboard

In Figure 4.14 general workflow of the prototype is represented, where communication between 3 participants (user, rest api and process engine) is roughly depicted. "Get Meta Model" and "Quality Assuarance" are represented as sub-processes for simplification purpose. Separate call to the rest service and its response (callback to the process engine) should be understood, if more than one action in the task is listed.

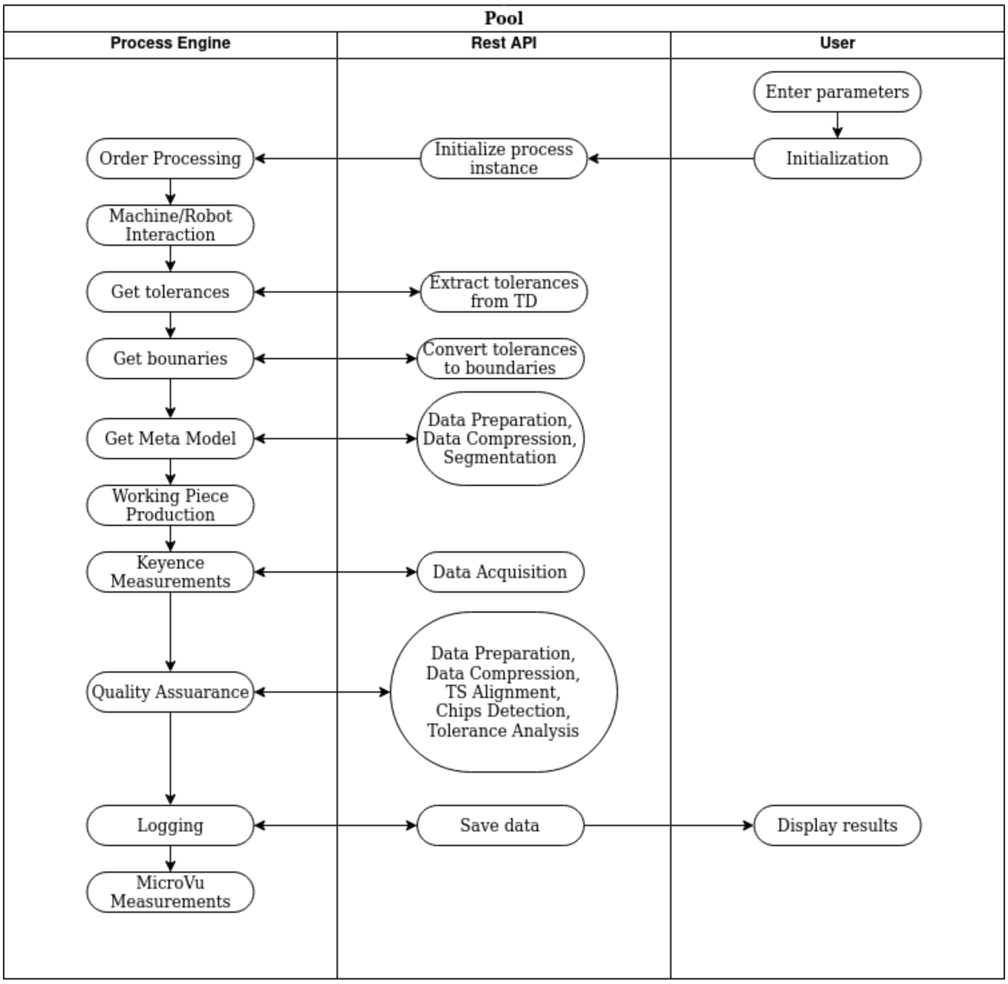


Figure 4.14.: Workflow

5. Evaluation

In order to evaluate how effective the quality assurance is we need to evaluate the efficiency of each step separately and select the best approaches and methodologies from suggested.

For evaluation not only mentioned earlier confusion matrix should be used, but appropriate evaluation method should be found for each step individually. his chapter describes not only the evaluation itself, however provide some elaborations and clarifications according each step, its necessity, possible limitations, side effects and improvement.

5.1. Data Compression

There are several criteria to evaluate compression techniques. The most common of them are the compression ratio, amount of compression, measuring the complexity of the technique, how fast the technique performs on a given machine, how closely the reconstruction resembles the original or how much memory is required for implemented technique [37].

As we are talking about lossy compression of geometrical surface we are interested to obtain compressed profile similar to the original one and the amount of compression. To evaluate the amount of compression we should count the number of pairs of the original surface profile and profile length after compression and calculate their ratio. Actual compression amount (CA) is the difference between absolute data size and this ratio converted into percentage:

$$\text{compression\_amount} = 100 * \left(1 - \frac{\text{profile\_length(after\_compression)}}{\text{profile\_length(before\_compression)}}\right)$$

To define how similar is the surface profile after compression to the original one Area under the Curve (AUC) for both profiles will be observed (see Fig. 5.1). We calculate how much original area is explained by compressed profile, how much is ignored and how much area, that is not a part of the original profile is included. Area covered by both surface profiles (SP):

$$\text{consolidated\_area} = \text{intersection}(\text{original SP}, \text{compressed SP})$$

Area which is not covered by compressed surface profiles, but is covered by original one, i.e., subtractions due to data compression:

$$\text{subtractions} = \text{original\_area} - \text{consolidated\_area}$$

Area which is covered by compressed surface profiles, but is not covered by original one, i.e., additions due to data compression:

$$\text{additions} = \text{compressed\_area} - \text{consolidated\_area}$$

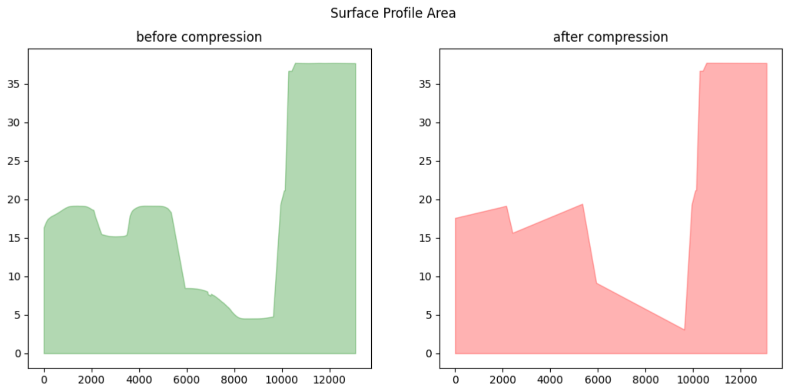


Figure 5.1.: AUC for Non-Compressed and Compressed Part

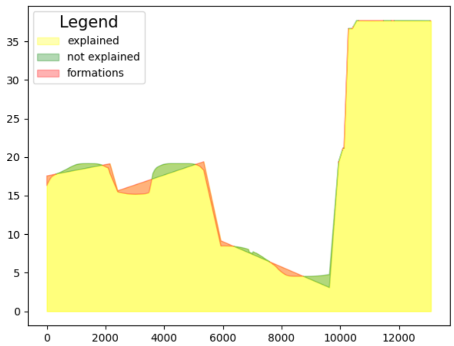


Figure 5.2.: Example of Surface Coverage Analysis

The third value is important for evaluation to avoid misclassification by chips formation analysis (see Fig. 5.2). Even if compressed area covers 100% of original area but percentage of additions is high, data compression approach can't be acknowledged as efficient.

For evaluation purposes we consider one batch with 37 working pieces. For each working piece compression amount (CA) and area parameters will be calculated and the average values for whole batch and for each compression approach will be found (see Tab. 5.1, 5.2).

Approach 1 shows good results in general, but this approach is good only in case when edge points are detected right (i.e., threshold for its detection corresponds smoothness of the working piece). The more edge points are detected, the higher compression amount and the area coverage are.

Approach	**Length of full PS**	**Length of compressed PS**	**Ratio**	**CA,%**
Approach 1	763	14	0.02	98
Approach 2	763	31	0.04	96
Approach 3	763	16	0.022	97.8
Approach 4	763	72	0.095	90.5
Approach 5	763	26	0.035	96.5

Table 5.1.: Evaluation of Compressed Amount

Approach	**Consolidated area,%**	**Subtractions,%**	**Additions,%**
Approach 1	98.8	1.2	2.3
Approach 2	95.9	4.1	3.8
Approach 3	97.6	2.3	1.7
Approach 4	99.96	0.04	0.03
Approach 5	99.6	0.4	2.2

Table 5.2.: Evaluation of Surface Coverage

On Figure 5.3 example of wrong edge point detection is shown, where compression amount is higher than 99%, consolidated area is 92.2, subtractions constitute 7.8% and additions percentage is equal to 6%. If less edge points as structure of the working piece suggests are found, sensible information about the working piece is lost. Percentage of false formations increases that could lead to misclassification in further analysis.

Approach 2 was evaluated with N = 50 by ”value” axis. If N value is too large the value of compression amount increases, but surface coverage quality decreases and vice versa. With N=100 compression amount is 98% and consolidated area, subtractions and additions values are 92.8 ,7.2 and 6.4% and with N = 200 these values are already 99, 85, 15 and 8.8% respectively.

Because of specification approach 3 always shows good results, with less or no additions, but there is always some area that is not covered by compressed surface profile, which can be important. Nonetheless if split value or other parameters in specification were defined wrong, results will be worth as compressed surface profile will deviate from original one.

Figure 5.3.: Approach 1. Wrong detection of edge points

By default in approaches 4,5 horizontal and vertical steps are equal to 1 and 250 respectively. As we can see in Table 5.1 and 5.2 the results of both approaches are pretty similar, but compression amount in approach 4 is always lower and its ratio between full and compressed surface length is 3 times higher as in approach 5. By increase of both steps compression amount increases as well and surface coverage decreases in both approaches.

In the Table 5.3 we can see how all values behave by changing of steps. Both approaches are stable and show similar results for consolidated area coverage. However in approach 5 the additions percentage grows significantly from 2 to 15%. In approach 4 by increase of steps values data compression doesn't loose in quality and compression amount increases.

	Approach 4				Approach 5			
thresholds	CSA	Sub	Add	CA	CSA	Sub	Add	CA
1,500	99.9	0.1	0.01	6.9	99.05	0.95	7.9	2.5
2,250	99.8	0.2	0.06	8.1	99.5	0.5	2.25	3.25
2,500	99.6	0.4	0.15	5.5	99	1	7.6	2.6
3,750	99.6	0.4	0.4	3.7	99.5	0.5	17.7	1.8
5,1000	99.4	0.6	0.5	3	97.6	2.4	15.5	1.5

Table 5.3.: Parameter analysis for approaches 4 and 5

Based on the evaluation results we can see, that despite significant data reduction the surface coverage is high. In almost all approaches compression amount and profile surface coverage is higher than 95%. However it is important to mention, that quality of surface coverage depends on parameters, that are required for data compression. Approach 4 is the most stable and the least sensible to set up thresholds, but with the lowest compression amount.

5.2. Time Series Alignment

In this section the evaluation of time series in temporal domain is performed. To decide which type of time series alignment is appropriate and whether it is necessary Time Alignment Measurement[1] (TAM) is selected.

The proposed methodology is able to describe the behaviour in time between two signals by measuring the fraction of time distortion between them (i.e., temporal periods of delay or advance). This method is based on DTW and optimal warping alignment path search with warp in time as extension.

To define whether two time series are in phase or out of it the optimal alignment path between them should be found. The principal is similar to the DTW, but this distance penalizes signals which are out of phase and benefits series that are in phase in the time domain. The larger the distance is, the larger the dissimilarity between both signals is [12].

The distance is bounded between 0 and 3 (i.e., both series are equal in temporal domain or completely out-of-phase respectively) [12].

Type of alignment	**Approach 1**	**Approach 2**	**Approach 3**	**Approach 4**	**Approach 5**
no alignment	0.69	0.69	0.15	0.82	0.32
resampling	0.52	0.87	0.16	0.9	0.48
scaling	0.58	1.00	0.24	0.99	0.77
synchronization	0.67	0.93	0.27	0.91	0.54

Table 5.4.: Evaluation of Surface Coverage

According to results provided in Table 5.4 best TAM values were achieved without any processing and the worst were achieved after scaling. Such effect proves that resampling and scaling don't solve problem with distortion in temporal domain and can even exacerbate it.

The reason for degradation during synchronisation can be sensitivity for chips formations and misalignment of the working piece as synchronization adjust surfaces based on the equal segments. But with time series alignment or without it average TAM distance for each case is less than 1 what means that time series are generally in phase (see Fig. 5.4).

After evaluation of time series alignment on the same batch but with measured with the help of CDP lift influence of misalignment is excluded. Gripper doesn't hold the working piece during measurement any more, the robotic arm moves to the lift's platform with special place holder, where working piece will be located. Such construction ensures that all parts are measured at the same slice under the same angle and with the same speed.

[1] https://github.com/dmfolgado/tam

Type of alignment	Approach 1	Approach 2	Approach 3	Approach 4	Approach 5
no alignment	0.05	0.33	0.12	0.39	0.1
resampling	0.1	0.37	0.16	0.45	0.2
scaling	0.13	0.44	0.24	0.52	0.29
synchronization	0.1	0.91	0.26	0.47	0.14

Table 5.5.: Evaluation of Surface Coverage

As we can see in Table 5.5 by good alignment of working pieces during measurement procedure level of synchrony in temporal domain increases. However time series can't be completely simultaneous as each working piece is of the same type but is not identical wit each other. If we exclude time series with chips formations TAM values in both cases will be reduced only on 1-4% on average. Nevertheless that fact doesn't show that chips formations have no impact on the time series alignment as percentage of time series with chips formations is too low (only 10% of the whole batch).

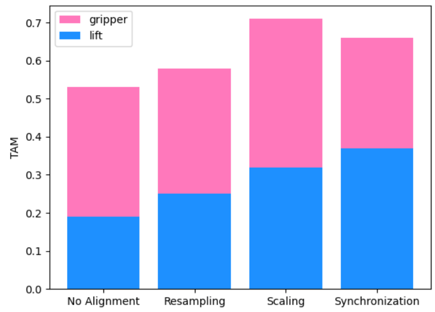


Figure 5.4.: Average TAM scores

In both cases the best result is achieved without any time series alignment and the more complex alignment method is chosen, the worst level of synchronization in temporal domain was reached(see Fig. 5.4).

But even if after time series alignment time series are not synchronous in temporal domain, alignment could be still important for Dynamical Time Warping as after alignment all time series have the same length and are more balanced in time axis, which is ignored in DTW.

5.3. Chips Formations Detection

In this section we evaluate not only efficiency of chips formation detection, but deviations of data compression and time series alignment from real state.

For evaluation batch with 37 working pieces will be used, where out of 37 working pieces only 4 have chips formations. As data set is unbalanced and it's easy to get high accuracy even if no chips were predicted confusion matrix will be used(see Sec. 2.4.2, Tab. 5.6).

batch size	**Predicted: CHIPS**	**Predicted: NO CHIPS**
Actual: CHIPS	TP: there are chips and we predict chips	FP: we predict chips, but there are no chips
Actual: NO CHIPS	FN: we predict no chips, but there are chips	TN: there are no chips and we predict no chips

Table 5.6.: Confusion matrix

All concomitant parameters are calculated for original and compressed profile surfaces and all compression and alignment methods will be included. Usually the most important parameter is false negative value. The more working pieces with chips are passed into the further production process, the large the possibly is that faulty working pieces could be sent to the customer and the more steps will be done further without its necessity increasing of operating time and general occupancy of machines.

However, in our case false positive value can't be ignored because unbalances batches with poor amount of time series with chips formations are normal practice in manufacturing as machines are setup to reach as low chips formations, errors and side effects during the process as possible. Constant misclassification of large amount of chips-free working pieces as bad once can be expensive in long perspective.

To get results presented in Table 5.7 data obtained during measurement procedure with gripper were used. The worst accuracy is achieved by use of original time series. Such side effect can be a reason of DTW limitations by handling large time series because of its sensitive to the mismatch of phases and segments.

The best accuracy and the lowest false negative value are achieved by time alignments methods, that shows that ability of DTW to handle sequences of different length is not a great advantage of this algorithm and provides no statistically significant difference in accuracy [36]. In almost all cases false positive values are between 10 and 20 pieces, that is around 40-60% of whole batch. Trying to tune threshold to reach zero false negative value accuracy falls to 1-5% and false positive quote grows up to 30-33 working pieces that is 85-90% of whole batch.

High rate of misclassification is caused not only by DTW constraints but by misalignment which occurs because of wrong grasping of the working piece by gripper. Misalignment or failures (i.e., working piece was not grasped at all) leads to significant errors [64].

In order to avoid an impact of misalignment and to obtain new data for adequate results estimation the same working pieces were measured with the help of CDP Lift (see

Tab. 5.8).

Stable environment provided by CDP Lift shows immediate affection to all types of data. 100% accuracy is reached by evaluation of full data by all kinds of alignment. The same results are obtained by approach 4, that approves the assumption about its efficiency. Approach 5 shows almost the same results with the small average accuracy reduction, however false negative value is still equals zero and only 1 working piece was misclassified. Such loss is not significant if we consider that compression amount in approach 5 is 3 times smaller as in approach 4.

Compression	Alignment	Accuracy	TP	TN	FP	FN	TH
—	—	45.9	2	15	18	2	63
	resampling	40.5	1	14	19	3	63
	scaling	37.8	1	13	20	3	16
	synchronization	48.6	2	16	17	2	63
approach 1	—	48.6	2	16	17	2	17
	resampling	75.6	0	28	5	4	17
	scaling	86.4	0	32	1	4	2
	synchronization	45.9	3	14	19	1	17
approach 2	—	78.3	1	28	5	3	32
	resampling	67.5	1	24	9	3	32
	scaling	59.4	1	21	12	3	4
	synchronization	37.8	3	11	22	1	32
approach 3	—	67.5	3	22	11	1	11
	resampling	75.6	2	26	7	2	14
	scaling	91.8	1	33	0	3	2
	synchronization	70.2	3	23	10	1	12
approach 4	—	56.7	1	20	13	3	26
	resampling	62.1	1	22	11	3	26
	scaling	37.8	2	12	21	2	2
	synchronization	59.4	2	20	13	2	26
approach 5	—	83.7	2	29	4	2	16
	resampling	89.1	2	31	2	2	16
	scaling	59.4	2	20	13	2	2
	synchronization	83.7	2	29	4	2	16

Table 5.7.: Chips Formations Detection Evaluation (Gripper)

Generally we can see the tendency that all outcome parameters were improved for all approaches after lift integration into measurement procedure (see Fig. 5.5). Interesting observation is, that by misalignment the worst results we achieved by no compression or by the use of approach with the largest compression ratio, and by removing of misalignment this trend is inverse. Simplification of profile surface and reduction of roughness caused by misalignment improve classification and the same reduction of details by normal conditions could overdo and remove important profile characteristics.

Compression	Alignment	Accuracy	TP	TN	FP	FN	TH
—	—	100	4	33	0	0	11
	resampling	100	4	33	0	0	11
	scaling	100	4	33	0	0	5
	synchronization	100	4	33	0	0	11
approach 1	—	91.8	3	31	2	1	11
	resampling	89.1	3	30	3	1	11
	scaling	83.7	2	29	4	2	1
	synchronization	91.8	3	31	2	1	11
approach 2	—	83.7	3	28	5	1	11
	resampling	86.4	3	29	4	1	11
	scaling	97.2	3	33	0	1	1
	synchronization	19.4	4	3	30	0	11
approach 3	—	91.8	3	31	2	1	11
	resampling	81.1	2	27	6	1	11
	scaling	94.5	2	33	0	2	2
	synchronization	86.4	3	29	4	1	13
approach 4	—	100	4	33	0	0	10
	resampling	100	4	33	0	0	10
	scaling	100	4	33	0	0	1
	synchronization	100	4	33	0	0	10
approach 5	—	100	4	33	0	0	10
	resampling	100	4	33	0	0	10
	scaling	97.2	4	32	1	0	1
	synchronization	100	4	33	0	0	10

Table 5.8.: Chips Formations Detection Evaluation (CDP Lift)

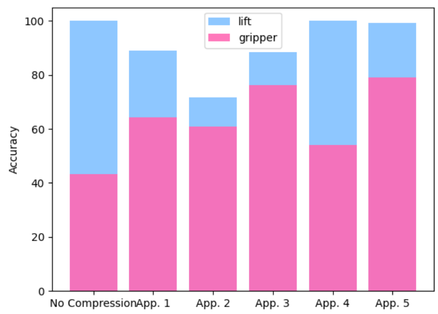


Figure 5.5.: Average Accuracy Scores for Chips Detection

5.4. Tolerance Analysis

To evaluate how efficient is selected tolerance analysis methodology the same batch with 37 working pieces would be used. Quality status of each working piece is predefined by the use of MicroVu. As 4 parts have chips formations they are excluded from the list of observations. We use 33 time series obtained during keyence measurements, where 20 parts are good and 13 are bad.

batch size	**Predicted: BAD**	**Predicted: GOOD**
Actual: BAD	TN: Part id bad and is predicted as bad	FP: Part is bad, but is predicted as good
Actual: GOOD	FN: Part is good, but is predicted as bad	TP: Part id good and is predicted as good

Table 5.9.: Confusion matrix

At first we will look at non-compressed data which we acquired with gripper and lift usage. For both cases we extract extreme or the most frequent values and create confusion matrix (see Tab. 5.9) for each of these methods (see Tab. 5.10). Additionally we calculate the average values for each of these methods to get manufacturing signature.

Measurement type	Features	Accuracy	TP	TN	FP	FN
gripper	extreme values	48	3	13	0	17
	frequent values	45	2	13	0	18
lift	extreme values	100	20	13	0	0
	frequent values	100	20	13	0	0

Table 5.10.: Tolerance Analysis Results for Non-Compressed Data

Due to construction of GV12 part (small notch after second large cylinder, see Fig. 5.6) when we are collecting extreme values, we obtain the wrong value for the fourth segment. Based on this peculiarity during tolerance analysis with extreme values we have only 3 values for evaluation instead of 4.

This is the reason, why the number of true positive values by the use of frequent values with gripper measurements decreases in comparison to the results obtained with extreme values.

In both cases we can see, that accuracy is pretty low but the false positive value is equal to zero, that means that precision is equals to 100%. That is good, as we can't deliver bad parts to customers, but at the same time recall is very low - 15/10%, that indicates that 85/90% of all good working pieces are misclassified (17/18 out of 33 parts, that is more than 50% of the whole batch).

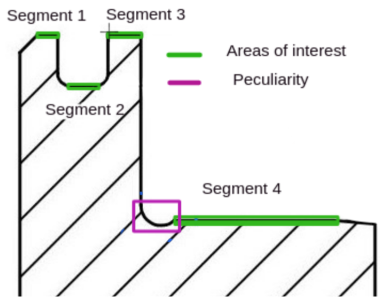


Figure 5.6.: Regions of Interest for GV12 Part

Using CDP lift the results in both cases are the same. Accuracy is 100%, recall and precision as well. Based on this outcome we can sum up, that correct alignment of the working piece has high influence on the data and its analysis.

The fact, that by extreme values we have one value less for quality status evaluation doesn't have any impact on the classification. We can suggest, that if tools are set up right and there are no errors in the production process, the whole part is good if the most of the segments are good. This can be useful information for the cases, where we have some segments that could not be measured, however it is still possible, that due to extreme values more parts are classified as good, than in the real state.

In the Table 5.11 tolerance analysis results for each compression approach are presented. For tolerance analysis no time series alignment is necessary, as interpolation and resampling, using during alignment, deform and distort original measurement values.

The average accuracy for all approaches with compressed data using gripper and lift is 42 and 65% respectively, what is worse than with non-compressed data.

In comparison to results obtained with non-compressed data using gripper drop of accuracy value is not significant (only 4%). However, it is important to mention, that not only accuracy value fell but recall and precision values became smaller as well - 8 and 65% instead of 13 and 87%. With the both types of data accuracy around 40-50% is reached only because almost all parts are predicted as bad. The average value of all bad predicted values is equal to 31 out of 33 parts, nevertheless not all parts that are predicted as bad, were bad, that shows specificity, which is equal only to 95.5%.

Despite good surface coverage by data compression (consolidation area is 98% on average), accuracy of tolerance analysis with compressed data even using gripper decreased by 35%. Precision, recall and specificity are 70,56 and 80% corresponding.

Unsatisfactory average outcomes could be explained by the fact, that during approaches 1-3 real values are substituted by values, received after apply of linear regression model. Basically, these methods should not be used for tolerance analysis or should be extended by including extreme or frequent values during compression.

Both approaches 4 and 5 show similar results as by using non-compressed data in cases with gripper and lift, but only by extraction of extreme values. Despite the fact, that only during approach 5 compressed profile surface is extended by extreme values, generally in all cases better results we reached by the use of them.

Frequent values method is more sensible to misalignment or data compression as extreme values. In order to reach good results in tolerance analysis data compression techniques should be extended by finding selected features from original data and adding them to the compressed surface profile.

Even adding features to the compressed data doesn't guaranty high efficiency by wrong threshold selection, as all approaches are fractious to their presetting. It implies that approach 4, after its extension, will be the most appropriated one, since this approach is the most stable and the less sensible to the threshold definition.

Compression	Measurement type	Features	Accuracy	TP	TN	FP	FN
approach 1	gripper	extreme values	42	2	12	1	18
		frequent values	39	1	12	1	19
	lift	extreme values	51	17	0	13	3
		frequent values	45	13	2	11	7
approach 2	gripper	extreme values	48	3	13	0	17
		frequent values	39	1	12	1	19
	lift	extreme values	39	0	13	0	20
		frequent values	39	0	13	0	20
approach 3	gripper	extreme values	42	1	13	0	19
		frequent values	39	0	13	0	20
	lift	extreme values	75	12	13	0	8
		frequent values	72	11	13	0	9
approach 4	gripper	extreme values	42	2	12	1	18
		frequent values	39	1	12	1	19
	lift	extreme values	100	20	13	0	0
		frequent values	69	11	12	1	9
approach 5	gripper	extreme values	48	3	13	0	17
		frequent values	42	2	12	1	18
	lift	extreme values	100	20	13	0	0
		frequent values	66	9	13	0	11

Table 5.11.: Tolerance Analysis Evaluation

Analysing the average values for each relevant segment, that represent manufacturing signature, we will ignore approaches 1-3, as they don't represent real measured values of the working piece. Exploring manufacturing signature the same tendency is present, as by tolerance analysis: better results are reached by the use of extreme values. Deviations in frequent values between compressed and original data explain decrease of accuracy in tolerance analysis.

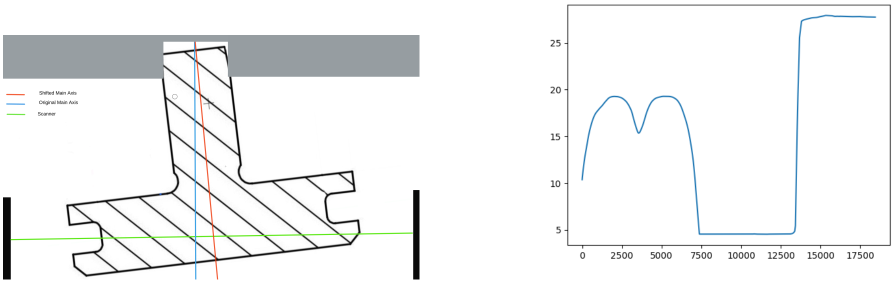
Looking at signatures of non-compressed data and other two approaches (see Tab. 5.12) we can see, that the first segment is always good, but the values for this segment gathered with gripper are smaller, than the other, gathered with lift, and further parameters in gripper signature are larger than the tolerance boundaries.

This finding allows us to make an assumption that the central axis of the working piece is shifted to the left (see Fig. 5.7(a)). If such behaviour repeats in the other batches as well, the gripper should be inspected and probably rechanged and its design should be updated.

compression	segments	extreme 1	extreme 2	extreme 3	frequent 1	frequent 2	frequent 3	frequent 4
—	gripper	19.17	15.9	19.21	19.165	17.19	19.21	7.14
	lift	19.19	15.17	19.20	19.19	15.17	19.20	4.538
approach 4	gripper	19.15	15.9	19.21	19.15	16.0	19.20	7.16
	lift	19.19	15.17	19.20	19.17	15.16	19.21	4.541
approach 5	gripper	19.17	15.9	19.21	19.165	16.11	19.21	4.55
	lift	19.19	15.17	19.20	19.19	15.22	19.20	4.4

Table 5.12.: Manufacturing Signature Analysis

This shift explains high values for segment 2 in the gripper signature and its large difference in comparison with the lift signature. As we can see on Fig. 5.7(b), although the same working piece is measured, surface profile of this current part has massive deviations from the real working piece profile, as segment 2 roughly speaking disappeared.



(a) Shift (b) Obtained Measurements

Figure 5.7.: Shift of the working piece

5.5. Transferability

Based on evaluation results obtained above we can make an assumption, that with compressed data the same results as with the non-compressed data could be obtained. In order to prove this hypothesis we will try the approach on a different part, with different characteristics.

Right working piece alignment during measurement procedure is very important to get sophisticated results during chips formations detection and tolerance analysis.

The best compression approach, i.e., approach with the best consolidate area index is Approach 4. Approach 5 shows almost the same results as approach 4, but has better compression amount. Time series alignment has no significant impact on further analysis, so it could be excluded from the process. During tolerance analysis better result in each case were reached by use of extreme values, as frequent values are to sensible to misalignment and data compression. By use of these approaches we reached the same results as by use of MicroVu measurements, but with significant operational time reduction, from 8 minutes to less than 1 minute.

During control evaluation only 2 approaches (4,5) will be used for data compression, no time series alignment will be included, only extreme values will be considered. Data for evaluation will be gathered with the help of CDP lift. As we announced our methodology as generic one another type of the working piece is used. Due to existing restrictions of our procedure (only rotary components) rook chess figure is selected instead of original GV12 working piece. However, working pieces are very different by size (14,2mm vs 50mm long and 19,2mm vs 22mm bright) and their form (see Fig. 5.8).

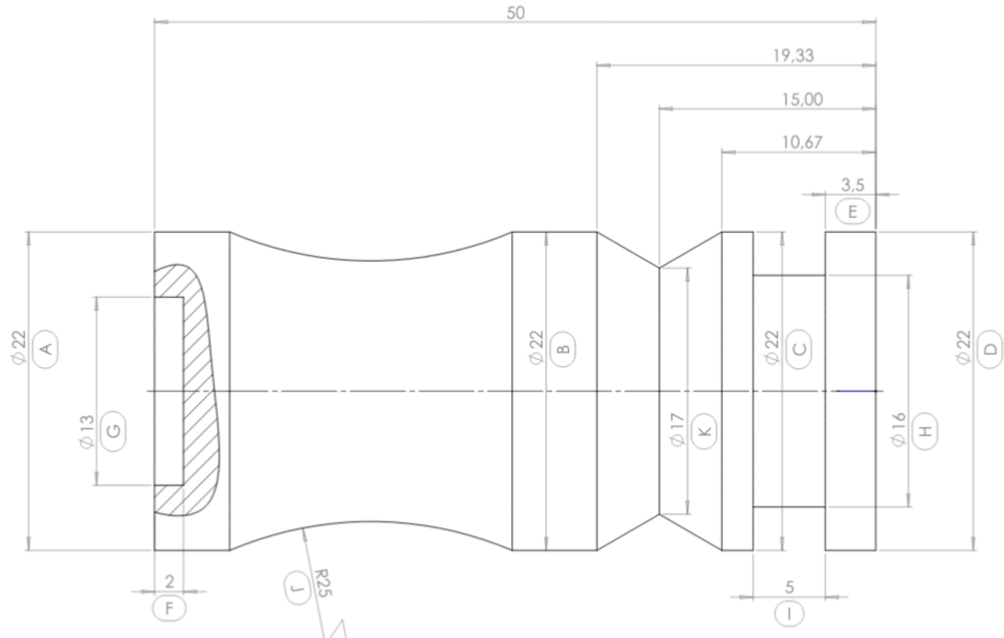


Figure 5.8.: Engineering Drawing for Rook Working Piece

For control evaluation we use one batch with 43 working pieces. As we can see in Table 5.13 compression amount is more than 90% in both cases. Compression amount by rook working pieces is even higher as by GV12. We can make an assumption, that the longer the working piece surface is, the larger compression amount could be achieved, however this difference is not significant. Coverage of the consolidated area is larger than 98% of the whole surface in both cases.

Nevertheless the size of subtractions and additions in approach 5 increase (see Tab. 5.14). Good results, achieved by both approaches, prove that not only approaches themselves are generic, but automatic threshold definition is effective as well and approach 4 is the most stable and insensitive to thresholds and conditions as working piece size and structure.

Approach	**Length of full PS**	**Length of compressed PS**	**Ratio**	**CA,%**
Approach 4	2575	165	0.064	93.6
Approach 5	2575	35	0.013	98.7

Table 5.13.: Evaluation of Compression Amount for Rook Batch

Approach	**Consolidated area,%**	**Subtractions,%**	**Additions,%**
Approach 4	99.93	0.06	0.06
Approach 5	98.3	1.6	6.1

Table 5.14.: Evaluation of Surface Coverage for Rook Batch

Out of 43 working pieces 6 have chips formations, so by chips formations detection 100% accuracy is achieved. Important is that precision and recall are equal to 100% as well, that means that all the time when we predict that part has chips formations, it is correct.

Important is, that for chips formations detection it is not necessary to have a good working piece, that simplified production process for the working piece, that is manufactured for the first time. We can use first manufactured part (bad as well) for further analysis, however this part should be measured correctly without chips, dust etc. The reason for that is, that difference between good and failed part is so insignificant, that DTW distance between good and bad working piece is too small or sometimes equals to zero. But for each type of the working piece threshold should be defined separately.

Compression	Alignment	Accuracy	TP	TN	FP	FN	TH
approach 4	—	100	6	37	0	0	30
approach 5	—	100	6	37	0	0	30

Table 5.15.: Chips Formations Detection for Rook Batch

After chips formations detection out of 43 parts only 37 remain. According to MicroVu measurements all remaining parts are good. Under good working piece we understand one, which dimensions don't violate tolerances in areas A,B,C,D,H and K (see Fig. 5.8).

As all parts are good and we can't have any true negative and false positive values, we exclude this values from our evaluation table (see Tab. 5.16). Based on results from

Table 5.16 the best accuracy for extreme vales was achieved by approach 4. For frequent values the best results were reached by approach 5 and by no data compression. By the use of the frequent values it was possible to get accuracy 100% by predicting all parts as good.

Compression	Features	Accuracy	TP	FN
—	extreme values	54	20	17
	frequent values	100	37	0
approach 4	extreme values	86	32	5
	frequent values	57	21	16
approach 5	extreme values	54	20	17
	frequent values	54	20	17

Table 5.16.: Tolerance Analysis Results for Rook Working Piece

At the first sight tolerance analysis results are very poor. Interesting is, that obtained results are very inconsistent as well. In order to explain this inconsistency we decided at first to look at manufacturing signatures. As meta model of the working piece was created automatically for each case based on the good time series, it is necessary to mention, that not all areas of interest were considered during tolerance analysis (see empty blanks in Tab. 5.17).

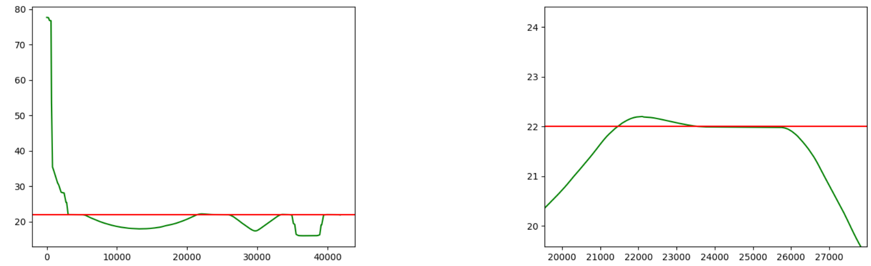
compression	features	dimensions					
		A	B	C	D	H	K
—	extreme	—	22.11	22.03	22.00	16.03	16.99
	frequent	22.00	21.99	22.02	21.99	16.04	17.01
approach 4	extreme	—	22.07	22.01	22.00	16.03	16.99
	frequent	22.00	21.99	21.63	22.00	16.04	16.99
approach 5	extreme	—	22.11	22.03	22.00	16.03	16.99
	frequent	22.00	22.09	21.96	22.00	—	17.02

Table 5.17.: Manufacturing Signature for Rook Working Piece

Low value in area C for approach 4 could be the reason for low performance (57% accuracy). As in all other cases, the value of area C is pretty similar and lies between its boundaries, we can assume, that such deviation is caused because of data compression.If this value would not be included in the meta model accuracy would increase.

All other areas in manufacturing signature table are in their boundaries, however area B is always too large in comparison with other circles and sometimes is out of its boundaries. It could be the side effect of data compression as well, however the same tendency is present by non-compressed data as well.

As such tolerances violation was present only in one area, we decided to inspect the batch more precisely. After short investigation, we found out that all parts in the batch have some convexity exactly in B segment (see Fig. 5.9).



(a) Rook Profile Surface

(b) Convexity in Segment B

Figure 5.9.: Inspection of Rook Working piece

Despite the fact, that this convexity appears by each part, it was not possible to detect it with MicroVu, as it measures each part only at certain position (see Fig. 5.10). MicroVu measures segment B exactly in its middle, and camper occurs on the left side of this segment. By the use of frequent values this camper is ignored as well, as it is small and took not more than 1/4 of the whole segment (i.e., can't be frequent value).

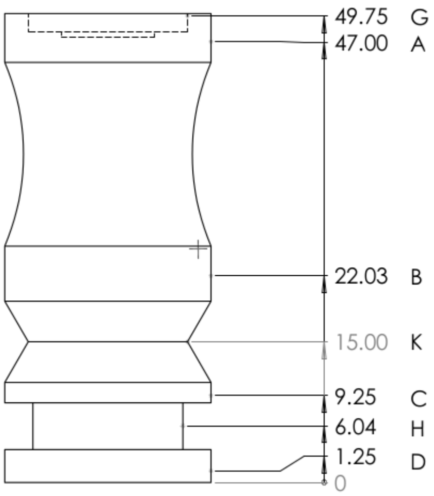


Figure 5.10.: Predefined positions for MicroVu measurements

Because of relatively big boundaries range (i.e., +/- 0.1mm from 21.9 to 22.1mm), by the use of extreme values some of the parts are still classified as good. However, in almost half of the cases (17 parts) this convexity is the reason of boundaries violation. Basically, by the use of extreme values we achieve an accuracy of 100% by analysing non-compressed data and data, compressed with approach 5 and by the use of approach 4 we get accuracy of 67%, as we predict 5 bad parts out of 17.

Like with the previous working piece type, we can make a conclusion, that extreme values are more appropriate for tolerance analysis. Compressed data could be used for tolerance analysis, however all approaches should be extended to add necessary features to compressed surface profile during compression.

6. Discussion, Conclusion and Future Work

In this work we show that software for close-to-production measurement could not only be set up in a generic manner, but could be effective and in some cases even more informative, as classic in-line measurement technologies and tools. To summarize the whole content of this thesis each research question, limitations and future work will be briefly discussed further in this chapter.

RQ1. *How can the software of close-to-production measurement machines work in a generic manner?*

To be able to use the same program to determine the properties of arbitrary work pieces appropriate measurement tools should be selected, as some tools are restricted according to their construction and functionality.

In proposed prototype high precision micrometer system is used to obtain surface profile cloud model of the working piece. As shown in Section 5.5 proposed method for data acquisition can be applied for working pieces with different size and contour structure.

Despite ability of the system to automatically correct misalignment and vibrations occurred in the unstable manufacturing environment additional factors as gripper, its design, pose estimation etc. should be considered. In this thesis (see Sec. 3) we suggest the implementation of CDP lift, which is optimal equipment for low-volume manufacturing due its maintenance, integration and price.

RQ2. *How can the data that such a generic program yields be structured, so that subsequent part specific logic does not have to deal with large point clouds, but with a set of human understandable values, such as dimensions and acclivities?*

In Section 3 and 4 we suggest 5 different data compression approaches for simplification of cloud point model of obtained surface profile of the working piece. All approaches have their pros and cons, but with each approach it is possible to reduce the data volume of measurement values by more than 90% and despite the fact, that we use lossy compression more than 95% of working piece area is covered by compressed contour.

Low percentage of subtractions and additions due to data compression (between 0 and 4%) confirms the fact, that data compression is effective and no accidental side effects for different types of working pieces was noticed.

Instead of point cloud we get a short list of points which could be interpreted without complicated algorithms. With the help of data compression, the reduction of analysis complexity and computational time is possible. Furthermore it solves problems with storage and processing capacities.

RQ3. *What are feasible methods to automatically determine from a list of tolerances, which measured dimensions and acclivities are corresponding to (a subset of) these tolerances?*

To get the set (subset) of tolerances which are relevant for quality assurance algorithm for extraction of textual elements from engineering drawings and convert them to boundaries. To define which dimensions and acclivities are corresponding or violating these tolerances segmentation of the working piece should be performed to extract the values of relevant dimensions. Obtained dimensions should be compared with the list of extracted boundaries.

In current prototype (see Sec. 3.3.4, 4) two types of values (extreme values and the most frequent values) were implemented and evaluated. Frequent values results correspond to the final QA measurements and are less sensible to unstable environment. Extreme values are more sensible but could be more informative and provide information, that is lost by use of final QA measurements.

Two of five data compression approaches (4 and 5) could be applied and used for further tolerance analysis. Other approaches should be extended with extreme or frequent value for efficient analysis.

RQ4. *Is it feasible, for a real-world data set, to reliably determine bad work pieces, without any further data (i.e., in-production, no access to historic final QA data)?*

In Section 5 it is shown on a real world data set that with extracted dimensions we can determine whether chips formations are present and quality status of the working piece could be determined, however at least one good sample is necessary for analysis.

However, results obtained in Section 5 show that high performance could be reached already with one good sample avoiding use of methods, where large historical data are required.

Unfortunately, it is not possible to completely avoid final QA measurements to provide extensive and sufficient quality control for other dimensions, which are ignored by micrometer.

At the same time, some segments, that is not possible to reach with final QA measurements could be evaluated by close-to-production measurement procedure far more effective and less resource-intensive.

RQ5. *How can tolerances and how individual work pieces adhere to them, be used to produce recommendations for changing the parameters of a manufacturing machine?*

The tolerances, i.e., how individual work pieces adhere to them, and manufacturing signature could be used to produce recommendations for changing the parameters of manufacturing machine or it tools, to estimate the working piece alignment and setups for measurement procedure in general, allows to adjust the production process parameters during the process run and improve batch production monitoring.

For example, finding unusual behaviour during manufacturing, as camper occurrence (see Sec. 5.5) could improve the whole process, prevent future production of faulty parts and help to select the right strategy for error fixing based on current resources. One of the reasons for such camper could be turning tool, which slides the profile of the working piece and because of the wear and tear can't finish step ordinary and escape previous area clean. To avoid it the tool should be simply changed or the whole profile should be abraded and polished after manufacturing.

It is shown that with compressed data obtaining the same analysis results as by usage of the complete data set is feasible. Moreover, by wrong working piece alignment simplification of the surface profile helps to improve the efficiency of performed analysis. Keyence measurements are precise and could be even more informative as MicroVu measurement, as we have an opportunity to analyse whole profile and not only some parts of it.

However working piece alignment is crucial for its effectiveness. For good working piece alignment it is not always necessary to integrate complicated and expensive sensors, cameras and some methods for pose estimation. Gripper design upgrade or process adjustment to avoid use of gripper during measurement procedure could improve the whole process a lot.

Furthermore, close-to-production measurements add benefit by reducing wasted machining time and scrap parts. Each time, when chips formations detection is successful and process is stopped, we save time avoiding final QA measurements.

Future research efforts will be focused on overcoming of the current limitations of the measurement method, e.g., considering working pieces with the form, differs from cylindrical one. For rotary parts implementation of additional view could be considered to extend the analysis opportunities and accuracy of predictions.

Data compression approaches could be extended by adding meaningful data points and features, that could be crucial for further analysis. Improvement of segmentation methodology or automatic definition of sub-segments is necessary for more complex working piece structures and forms.

Manufacturing signatures could be extended by considering machining data, gathered during manufacturing.

Bibliography

[1] Banerjee, A.G., Barnes, A., Kaipa, K.N., Liu, J., Shriyam, S., Shah, N., Gupta, S.K.: An ontology to enable optimized task partitioning in human-robot collaboration for warehouse kitting operations. In: Next-Generation Robotics II; and Machine Intelligence and Bio-inspired Computation: Theory and Applications IX. vol. 9494, p. 94940H. International Society for Optics and Photonics (2015)

[2] Bhuiyan M.S.H., C.I.: Comprehensive Materials Processing. Elsevier, 1 edn. (2014), https://www.sciencedirect.com/science/article/pii/B9780080965321013303

[3] Bi, Z., Wang, L.: Advances in 3d data acquisition and processing for industrial applications. Robotics and Computer-Integrated Manufacturing 26(5), 403–413 (2010), https://www.sciencedirect.com/science/article/pii/S073658451000013X

[4] Bianca M. Colosimo, N.S.: Geometric Tolerances. Impact on Product Design, Quality Inspection and Statistical Process Monitoring. Springer, London, 1 edn. (2011), https://link.springer.com/book/10.1007/978-1-84996-311-4

[5] Chursin, Y.A., Fedorov, E.M.: Methods of resolution enhancement of laser diameter measuring instruments. Optics & Laser Technology 67, 86–92 (2015), https://www.sciencedirect.com/science/article/pii/S0030399214002461

[6] Colosimo, B.M., Moya, E.G., Moroni, G., Petrò, S.: Statistical sampling strategies for geometric tolerance inspection by cmm. Stochastics and Quality Control 23(1), 109–121 (2008), https://doi.org/10.1515/EQC.2008.109

[7] Dorst, T., Ludwig, B., Eichstädt, S., Schneider, T., Schütze, A.: Metrology for the factory of the future: towards a case study in condition monitoring. In: 2019 IEEE International Instrumentation and Measurement Technology Conference (I2MTC). pp. 1–5 (2019)

[8] Durakbasa, N.: Geometrical Product Specifications and Verification For The Analytical Description Of Technical and Non-Technical Structures. TU, Abt. Austauschbau und Messtechnik (2014)

[9] Eger, F., Coupek, D., Caputo, D., Colledani, M., Penalva, M., Ortiz, J.A., Freiberger, H., Kollegger, G.: Zero defect manufacturing strategies for reduction of scrap and inspection effort in multi-stage production systems. Procedia CIRP 67, 368–373 (2018), https://www.sciencedirect.com/science/article/pii/S2212827

117311721, 11th CIRP Conference on Intelligent Computation in Manufacturing Engineering, 19-21 July 2017, Gulf of Naples, Italy

[10] Ehrendorfer, M., Fassmann, J.A., Mangler, J., Rinderle-Ma, S.: Combining conformance checking and classification of xes log data for the manufacturing domain. Technical report, arxiv (April 2019), http://eprints.cs.univie.ac.at/6049/

[11] Farin, G., Hoschek, J., Kim, M.S.: Handbook of computer aided geometric design. North Holland, 1st edn. (2005)

[12] Folgadoa, D., Barandasa, M., Matiasbc, R., Martinsb, R., Carvalhod, M., Gamboae, H.: Time alignment measurement for time series. Pattern Recognition 81, 268–279 (2018)

[13] Gao, W.: Metrology. Precision Manufacturing, Springer Singapore, 1st edn. (2019)

[14] Giorgino, T.: Computing and visualizing dynamic time warping alignments in r: The dtw package. Journal of Statistical Software 31 (2009)

[15] Graniero, P., Gärtler, M.: Prediction of Batch Processes Runtime Applying Dynamic Time Warping and Survival Analysis, pp. 53–61. Springer Vieweg, Berlin, Heidelberg (01 2021)

[16] Haraguchi, N., Cheng, C.F.C., Smeets, E.: The importance of manufacturing in economic development: Has this changed? World Development 93(C), 293–315 (2017), https://EconPapers.repec.org/RePEc:eee:wdevel:v:93:y:2017:i:c:p:293-315

[17] Hirsch, B.: CIM in der Unikatfertigung und -montage. Verlag TÜV (1992)

[18] Horst, J., Hedberg, T., Feeney, A.B.: On-machine measurement use cases and information for machining operations. NIST Advanced Manufacturing Series 400-1 (2019)

[19] Japkowicz, N., Shah, M.: Evaluating learning algorithms: a classification perspective. Cambridge University Press (2011)

[20] Kareta, W.N.: Automation of end of line quality control. https://www.spotlightmetal.com/automation-of-end-of-line-quality-control-a-916467/, accessed: 2020-03-25

[21] Kareta, W.N.: Quality-by-design (qbd). https://tbmed.eu/quality-by-design, accessed: 2020-03-25

[22] Kljun, M.: Measuring similarity of univariate time series. RUL. University of Ljubljana (2020), https://repozitorij.uni-lj.si/IzpisGradiva.php?lang=slv&id=119310

[23] Leach, R.: Optical Measurement of Surface Topography. Springer, Berlin, Heidelberg, 1 edn. (2011), https://link.springer.com/book/10.1007%2F978-3-642-12012-1

[24] Lepot, M., Aubin, J.B., Clemens, F.H.: Interpolation in time series: An introductive overview of existing methods, their performance criteria and uncertainty assessment. Water 9(10) (2017), https://www.mdpi.com/2073-4441/9/10/796

[25] Linder, W.: Digital Photogrammetry. A Practical Course. Springer, Berlin, Heidelberg, 3 edn. (2009), https://link.springer.com/book/10.1007%F978-3-540-92725-9

[26] Louis E. Boone, David L. Kurtz, S.B.: Contemporary Business. Wiley, 18 edn. (2019)

[27] M.A.Anusuya, S.: Classification techniques used in speech recognition applications: A review. International Journal of Circuit Theory and Applications,IJCTA p. 910–911 (2011)

[28] Meinard, M.: Dynamic Time Warping, pp. 69–84. Springer Berlin Heidelberg, Berlin, Heidelberg (2007), https://doi.org/10.1007/978-3-540-74048-3_4

[29] Moniz, N., Branco, P., Torgo, L.: Resampling strategies for imbalanced time series. In: 2016 IEEE International Conference on Data Science and Advanced Analytics (DSAA). pp. 282–291 (2016)

[30] Murthy, P.R.: Production And Operations Management. New Age International Pvt Ltd (2009)

[31] Müller, A.M., Oberleiter, T., Willner, K., Hausotte, T.: Implementation of parameterized work piece deviations and measurement uncertainties into performant meta-models for an improved tolerance specification. Proceedings of the Design Society: International Conference on Engineering Design 1(1), 3501–3510 (2019)

[32] Nawaz, S., Mascolo, C.: Mining users' significant driving routeswith low-power sensors. Proceedings of the 12th ACM Conference on Embedded Network Sensor Systems p. 236–250 (2014)

[33] Pfeifer, T.: Production Metrology. Oldenbourg Wissenschaftsverlag (2015), https://doi.org/10.1515/9783486810424

[34] Polini, W., Moroni, G.: Manufacturing signature for tolerance analysis. Journal of Computing and Information Science in Engineering 15(2) (06 2015), https://doi.org/10.1115/1.4028937, 021005

[35] Pu, I.: Data compression. University of London (2005), https://london.ac.uk/sites/default/files/study-guides/data-compression.pdf

[36] Ratanamahatana, C., Keogh, E.: Three myths about dynamic time warping data mining. In: Proceedings of the 2005 SIAM International Conference on Data Mining. SDM 2005. pp. 506–510 (04 2005)

[37] Rushdy, E.: Performance evaluation of data compression techniques versus different types of data. International Journal of Computer Science and Information Securiy Vol. 11, No. 12, 73–78 (12 2013)

[38] Salvador, S., Chan, P.: Toward accurate dynamic time warping in linear time and space. In: KDD workshop on mining temporal and sequential data. vol. 11, pp. 561–580 (2007), http://content.iospress.com/articles/intelligent-data-analysis/ida00303

[39] Scheibel, B., Mangler, J., Rinderle-Ma, S.: Extraction of dimension requirements from engineering drawings for supporting quality control in production processes. Computers in Industry 129, 103442 (2021), https://www.sciencedirect.com/science/article/pii/S016636152100049X

[40] Shek, C.F., Betenbaugh, M.: Taking the pulse of bioprocesses: at-line and in-line monitoring of mammalian cell cultures. Current Opinion in Biotechnology 71, 191–197 (2021), https://www.sciencedirect.com/science/article/pii/S0958166921001531, analytical Biotechnology

[41] Siebert, J., Groß, J., Schroth, C.: A systematic review of python packages for time series analysis. Engineering Proceedings (04 2021)

[42] Simmons, C.H., Phelps, N., Maguire, D.E.: Manual of Engineering Drawing. Butterworth-Heinemann, 4 edn. (2012)

[43] Stertz, F., Rinderle-Ma, S., Mangler, J.: Analyzing Process Concept Drifts Based on Sensor Event Streams During Runtime, pp. 202–219. Springer International Publishing (09 2020)

[44] Su, Q., Ganesh, S., Le Vo, D.B., Nukala, A., Bommireddy, Y., Gonzalez, M., Reklaitis, G.V., Nagy, Z.K.: A quality-by-control approach in pharmaceutical continuous manufacturing of oral solid dosage via direct compaction. In: Kiss, A.A., Zondervan, E., Lakerveld, R., Özkan, L. (eds.) 29th European Symposium on Computer Aided Process Engineering, Computer Aided Chemical Engineering, vol. 46, pp. 1327–1332. Elsevier (2019), https://www.sciencedirect.com/science/article/pii/B9780128186343502228

[45] Su, Q., Ganesh, S., Moreno, M., Bommireddy, Y., Gonzalez, M., Reklaitis, G.V., Nagy, Z.K.: A perspective on quality-by-control (qbc) in pharmaceutical continuous manufacturing. Computers & Chemical Engineering 125, 216–231 (2019), https://www.sciencedirect.com/science/article/pii/S0098135418309116

[46] Tavenard, R., Faouzi, J., Vandewiele, G., Divo, F., Androz, G., Holtz, C., Payne, M., Yurchak, R., Rußwurm, M., Kolar, K., Woods, E.: Tslearn, a machine learning toolkit for time series data. Journal of Machine Learning Research 21(118), 1–6 (2020), http://jmlr.org/papers/v21/20-091.html

[47] Telsang, M.T.: Industrial Engineering and Production Management. S. Chand Publishing, 3 edn. (2018)

[48] Theys, P., Roque, T., Constable, M.V., Williams, J., Storey, M.: Current status of well logging data deliverables and a vision forward. In: SPWLA 55th Annual Logging Symposium. SPWLA Annual Logging Symposium, vol. All Days (05 2014), sPWLA-2014-DD

[49] Ting, K.M.: Confusion Matrix, pp. 209–209. Springer US, Boston, MA (2010), https://doi.org/10.1007/978-0-387-30164-8_157

[50] Toyoda, M., Sakurai, Y., Ishikawa, Y.: Pattern discovery in data streams under the time warping distance. The VLDB Journal 22, 295–318 (2012)

[51] Uhlemann, T.H.J., Schock, C., Lehmann, C., Freiberger, S., Steinhilper, R.: The digital twin: Demonstrating the potential of real time data acquisition in production systems. Procedia Manufacturing 9, 113–120 (2017), https://www.sciencedirect.com/science/article/pii/S2351978917301610, 7th Conference on Learning Factories, CLF 2017

[52] Unit Manufacturing Process Research Committee, C.o.E., Technical Systems, N.R.C.: Unit Manufacturing Processes: Issues and Opportunities in Research. THE NATIONAL ACADEMIES PRESS, Washington, D.C. (1995)

[53] Wager, J.: Surface effects in pneumatic gauging. International Journal of Machine Tool Design and Research 7(1), 1–14 (1967), https://www.sciencedirect.com/science/article/pii/0020735767900212

[54] Wang, F.K., Hubele, N.F.: Quality evaluation of geometric tolerance regions in form and location. Quality Engineering 14(2), 205–211 (2002), https://doi.org/10.1081/QEN-100108677

[55] Webpage: Data compression. https://searchstorage.techtarget.com/definition/compression, accessed: 2017-08-01

[56] Webpage: Four things users hate most about their cad systems. https://www.engineering.com/DesignSoftware/DesignSoftwareArticles/ArticleID/15581/4-Things-Users-Hate-Most-About-Their-CAD-Systems.aspx, accessed: 2017-09-28

[57] Webpage: Inline, online or atline - measurement approaches in process analytics. https://www.en.silicann.com/blog/post/inline-online-atline-process-analytics/, accessed: 2021-01-21

[58] Webpage: Just how important is manufacturing? https://hbr.org/2012/02/just-how-important-is-manufact, accessed: 2012-02-21

[59] Webpage: Keyence measurement library. https://www.keyence.eu/ss/products/measure/measurement_library/type/optical/, accessed: 2021-08-14

[60] Webpage: Precision hand tools. `https://www.starrett.com/category/11`

[61] Webpage: Survival of the fittest - the process control imperative. `http://resources.renishaw.com/en/details/white-paper-survival-of-the-fittest-the-process-control-imperative--35900`, accessed: 2017-07-13

[62] Webpage: X-ray ct emerging as cmm alternative for dimensional inspection. `https://metrology.news/x-ray-ct-emerging-as-cmm-alternative-for-dimensional-inspection/`, accessed: 2018-01-25

[63] Wiederhold, M., Greipel, J., Ottone, R., Schmit, R.: Clustering of similar processes for the application of statistical process control in small batch and job production. Int. J. Metrol. Qual. Eng. 7(4) (2016), `https://EconPapers.repec.org/RePEc:eee:wdevel:v:93:y:2017:i:c:p:293-315`

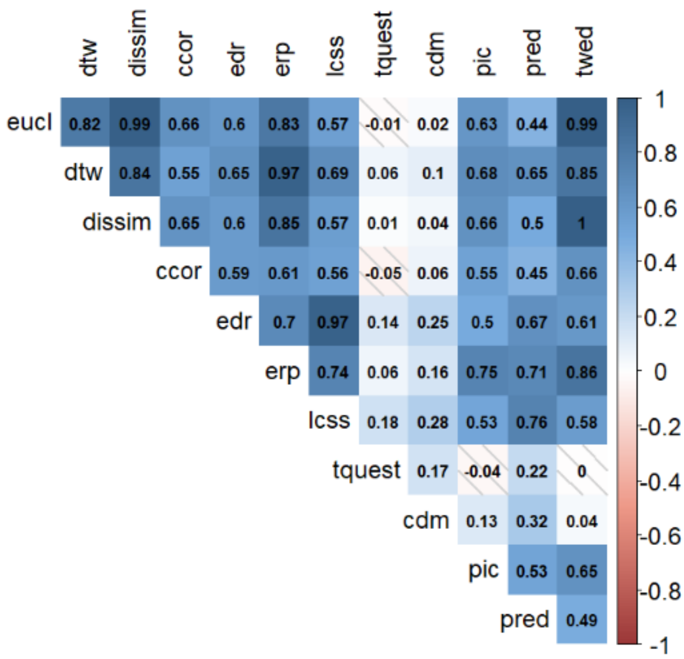
[64] Wolniakowski, A., Kramberger, A., Gams, A., Chrysostomou, D., Hagelskjær, F., Thulesen, T.N., Kiforenko, L., Buch, A.G., Bodenhagen, L., Petersen, H.G., Madsen, O., Ude, A., Krüger, N.: Optimizing grippers for compensating pose uncertainties by dynamic simulation. In: 2016 IEEE International Conference on Simulation, Modeling, and Programming for Autonomous Robots (SIMPAR). pp. 177–184 (2016)

[65] Woo, H., Kang, E., Wang, S., Lee, K.H.: A new segmentation method for point cloud data. International Journal of Machine Tools and Manufacture 42(2), 167–178 (2002), `https://www.sciencedirect.com/science/article/pii/S0890695501001201`

[66] Xia, R., Han, J., Lu, R., Xia, L.: Vision-based on-machine measurement for cnc machine tool. Ninth International Symposium on Precision Engineering Measurements and Instrumentation 9446 (03 2015)

[67] Yueh-Jaw Lin, R.M., Massina, E.: Cad-based cmm dimensional inspection path planning – ageneric algorithm. Robotica (2001) 19, 137–148 (2020)

A. Appendix



Note: DTW, Euclidean distance, DISSIM, EDR, ERP, LCSS, and RWS all give numerically similar results.
The following are particularly similar: Euclidean distance and DISSIM, DTW and ERP and RWS, and EDR and LCSS.
CCor, PIC, and PRED are somewhat similar to each other and other methods. TQuEST and CDM strongly differ from the other measures and each other [22].

Figure A.1.: Pearson correlation between similarity measures across all pairs of time series from all nine groups [22]

	warping	scaling	Comp.	normalization	Wang	Esling	Montero	Diff. lengths
L_p	strong	no	$O(n)$	$\sqrt[p]{n}$	lock-step	shape	model-free	no
DISSIM	no	no	$O(n+m)$	$n-1$	lock-step	shape	*model-free*	no
DTW	strong	no	$O(nm)$	$n, m, n+m$	elastic	shape	model-free	yes
ERP	weak†	no	$O(nm)$	$max(n, m)$	elastic	edit	*model-free*	yes
EDR	weak†	no	$O(nm)$	$max(n, m)$	elastic	edit	*model-free*	yes
LCSS	weak†	no	$O(nm)$	normalized	elastic	edit	*model-free*	yes
Tquest	weak†	no	$O(nm \log nm)$	normalized	threshold	feature	*model-free*	yes
CCor	strong	yes	$O(nm)$	$(\sqrt{ml})^{-1}$	*elastic*	*feature*	model-free	yes
PIC	strong	yes	$O(n+m)$	$max(k1, k2)$	/	structure	model-based	yes
CDM	weak†	no	$O(n+m)^*$	normalized	/	structure	complexity	yes
PRED	no	no	$O(n+m)^*$	normalized	/	*structure*	prediction	yes*
RWS	weak	no	$O(n)^*$	normalized	/	*structure*	*model-based*	no

Legend:
/ denotes that Wang et al. treated different representations separately from similarity measures;
* denotes the most typical implementation but may vary with choice of underlying models;
† denotes invariance based on the original definitions, where time is considered, but this vary with implementation [22].

Table A.1.: Overview of different time series measures [22]

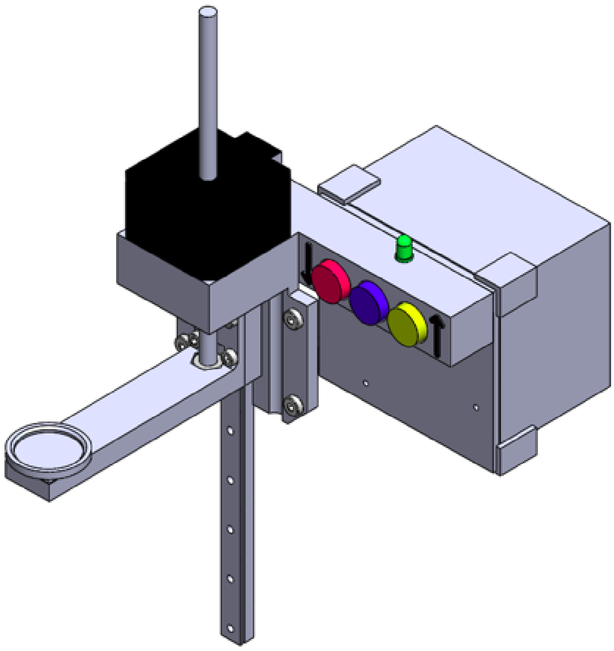


Figure A.2.: Lift Components

QUANTUM Lift was designed and constructed by CDP[1] employee Marcel Fuschlberger.

Components of QUANTUM Lift:

- LML9H 150mm linear rail
- L4118S1404-T6X1 non captive linear motor
- Arduino as a controlling unit
- Raspberry Pi for remote access and automation
- Mounted on Keyence LS-9120M optical micrometer
- 3D-printed components

Technical data of QUANTUM Lift:

- $\pm 8\mu$m measurement accuracy
- Up to 16.000 measurements per second
- Working piece diameters of 0.8mm up to 120mm
- Working piece maximum height of 110mm (can be extended)
- Working piece maximum weight of 3000g
- Maximum speed of 40mm/s

[1] Center for Digital Production: https://acdp.at

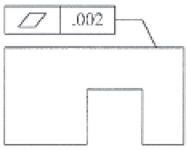
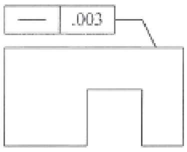
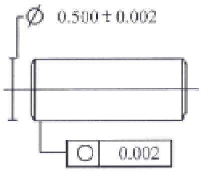
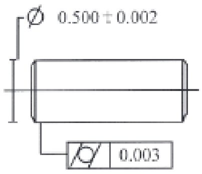
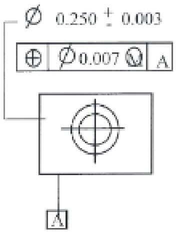
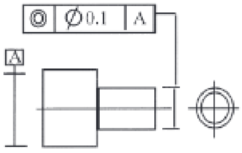
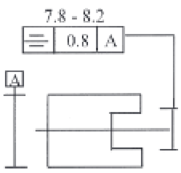
Feature	Symbolic Call-outs	Geometry of Tolerance Region
a. Flatness		0 .002 Allowable difference between two parallel planes containing all points along the inspection plane. Univariate.
b. Straightness		0 .003 Allowable difference between two lines containing all points along the inspection line. Univariate.
c. Circularity for a cylinder		0 .002 The two concentric circles containing each circular element of the surface in a plane perpendicular to an axis must be within 0.002. Univariate.
d. Cylindricity		0 .003 Allowable difference between two concentric cylinders must be within 0.003. Univariate.
e. Position		0 .007 Allowable difference when the hole has minimum diameter (MMC); its position must be within 0.007. Univariate.
f. Concentricity		0 0.1 Within the limits of size and regardless of feature size, all median points of diametrically opposed elements of the feature must lie within a diameter of 0.1 in cylindrical tolerance region. Univariate.
g. Symmetry		0 0.8 Within the limits of size and regardless of feature size, all median points of opposed elements of the slot must lie between two parallel planes 0.8 apart. Univariate.

Table A.2.: Example of Form and Location GD&T[2] Specifications [54]

[2]Geometric Dimensioning and Tolerancing

Feature	Tolerance Region	Mathematical Representation	value t
a. Flatness	an interval region between zero and the conform value t	0<x<t, where x is the uni-variate value of flatness	t=0.002 (Tab. A.2)
b. Straightness	an interval region between zero and the conform value t	0<x<t, where x is the uni-variate value of straightness	t=0.003 (Tab. A.2)
c. Circularity	an interval region between zero and the conform values t1 and t2	t1 <r < t2, where r is the uni-variate value of circularity	t1=0.0 and t2=0.002 (Tab. A.2)
d. Cylindricity	an interval region between zero and the conform values t1 and t2	t1<r<t2,where r is the uni-variate value of cylindricity	t1=0.0 and t2=0.003 (Tab. A.2)
e. Position	an interval region between zero and the conform value t	0<x<t, where x is the uni-variate value of position	t=0.007 (Tab. A.2)
f. Concentricity	an interval region between zero and the conform value t	0<r<t, where r is the univariate value of concentricity	t=0.1 (Tab. A.2)
g. Symmetry	an interval region between zero and the conform value t	0<x<t, where x is the univariate value of symmetry	t=0.8 (Tab. A.2)

Table A.3.: Mathematical Representation of the GD&T specification [54]

```
Function cross_points(pairs, split):
    data_length = length(pairs.value)-1;
    indexes = [ ];
    indexes.append(pairs.value.first);
    for a in range(0,data_length) do
        current = pairs.value[a];
        next = pairs.value[a + 1];
        if (next >= split and current < split) or (next < split and current >= split)
         then
            indexes.append(pairs.value[a]);
            indexes.append(pairs.value[a+1]);
        end
    end
    indexes.append(pairs.value.last);
    return indexes;
```

Algorithm 12: Cross_points function. Approach 3

```
Function get_quartiles(regression_lists, specification):
    quartiles = specification.quartile;
    extreme = specification.extreme;
    if quartiles != "NA" then
        for q in quartiles) do
            time = [];
            time.append(regression_lists[q].first);
            time.append(regression_lists[q].last);
            q1 = percentile(time, quartiles[q][0]);
            q2 = percentile(time, quartiles[q][1]);
            current_segment = regression_lists[q];
            reduced = [i for i in current_segment if i >= q1 and i <= q3];
            regression_lists.append(reduced);
            segment_len = length(regression_lists) - 1;
            extreme[segment_len] = extreme[q];
        end
    end
```

Algorithm 13: Get_quartiles function. Approach 3

```
Function create_grid(pairs, direction, step=0):
    min_val = int(min(pairs[direction]))-1;
    split = min_val;
    max_val = int(max(pairs[direction]))+1;
    data_length = length(newData[direction])-1;
    difference = max_val-min_val;
    if step == 0 then
        step = find_threshold(difference,direction);                // extension
    end
    lines = int(difference/step) + 1;
    cross_points = {};
    foreach line in lines do
        for a in range(0,data_length) do
            current = pairs[direction][a];
            next = pairs[direction][a + 1];
            if (next >= split and current < split) or (next < split and current >=
             split) then
                if (abs(next-current) < step) then
                    cross_points[pairs["timestamp"][a]] = pairs["value"][a];
                else
                    cross_points[pairs["timestamp"][a]] = pairs["value"][a];
                    cross_points[pairs["timestamp"][a+1]] = pairs["value"][a+1];
                end
            end
        end
        split = split + step;
    end
    cross_points[pairs["timestamp"][0]] = pairs["value"][0];
    cross_points[pairs["timestamp"][length]] = pairs["value"][length];
    time, value = dict_to_list(cross_points);
    return time,value;
```

Algorithm 14: Create_grid function. Approach 4

```
Function index_grid(pairs, step):
    min_val = int(min(pairs[direction]))-1;
    split = min_val;
    max_val = int(max(pairs[direction]))+1;
    data_length = length(newData[direction])-1;
    difference = max_val-min_val;
    if step == 0 then
        step = find_threshold(difference);                  // extension
    end
    lines = int( difference/step) + 1;
    indexes = [];
    indexes.append(first);
    foreach line in lines do
        for a in range(0,data_length) do
            current = pairs[direction][a];
            next = pairs[direction][a + 1];
            if (next >= split and current < split) or (next < split and current >=
              split) then
                if (abs(next-current) < step) then
                    indexes.append(a);
                else
                    indexes.append(a);
                    indexes.append(a+1);
                end
            end
        end
        split = split + step;
    end
    indexes.append(last);
    return indexes;
```

Algorithm 15: Index_grid function. Approach 5

```
Function recursive_function(a, indexes, pairs, ttx):
    ind_length = length(indexes) - 1;
    while a < ind_length do
        delta = abs(pairs.timestamp.current-pairs.timestamp.next);
        if delta < ttx then
            recursive_function(a + 1, indexes, pairs, ttx);
            indexes.delete(a);
        end
        a = a + 1;
    end
```

Algorithm 16: Recursive_function. Approach 5

```
Function select_points(area, ttx, pairs, delta_time):
    if delta_time >= ttx then
        if length(area) > 2: then
            extreme.append(min(area));
            extreme.append(max(area));
        end
    else
        recursive_function(a + 1, indexes, pairs, ttx);
    end
```

Algorithm 17: Select_points function. Approach 5

```
Function create_segmentation_grid(pairs, direction, step=0):
    min_val = int(min(pairs[direction]))-1;
    split = min_val;
    max_val = int(max(pairs[direction]))+1;
    data_length = length(newData[direction])-1;
    difference = max_val-min_val;
    if step == 0 then
        step = find_threshold(difference,direction)
    end
    lines = int(difference/step) + 1;
    segment_points = {};
    foreach line in lines do
        count = 0;
        for a in range(0,data_length) do
            current = pairs[direction][a];
            next = pairs[direction][a + 1];
            if (next >= split and current < split) or (next < split and current >=
             split) then
                if (abs(next-current) < step) then
                    count = count +1 ;
                end
            end
        end
        segment_points[split] = count;
        split = split + step;
    end
    return segment_points;
```

Algorithm 18: Create_segmentation_grid function

```
Function extreme_values(compressed_data):
    extremes = [ ];
    cur_split, cur_segment, cur_areas = find_sline(cur_ts);
    for area in cur_areas do
        meanval = mean(area);
        if meanval < cur_split then
            extreme = min(area);
        else
            extreme = max(area);
        end
        extremes.append(extreme);
    end
    return extremes;
```

Algorithm 19: Function to find extreme values

```
Function frequent_values(data):
    frequents = [ ];
    cur_split, cur_segment, cur_areas = find_sline(data['value']);
    for area in cur_areas do
        time_val_dif = [] for a in area do
            temp = a.time.next - a.time.current;
             time_val_dif.append([temp,a.value.current]);
        end
        frequent = max(time_val_dif,by=temp);
        frequents.append(frequent);
    end
    return frequents;
```

Algorithm 20: Function to find frequent values

B. Appendix

B.1. Source Code

The source code for prototype, describe in Section 4, is available as GitHub repository:

```
https://intra.acdp.at/gogs/nklievtsova/keyence
https://github.com/WSaccoun/keyence
```

To install all dependencies (i.e., software components required for this project) use requirements.txt file, provided in the repository:

```
python -m pip install -r requirements.txt
```

As most of the endpoints are called from the process engine and most of the parameters/arguments are already defined there, list of initial parameters should be passed into the process engine during `keyence/initialization` endpoint call.

Following parameters should be passed:

- batch size
- path to technical drawing
- compression type and respective parameters
- path to basic (good) time series
- type of time series alignment
- type of distance measurement and threshold for chips detection
- path to desired logging folder

This list could be always adjusted and extended, if additional tasks should be added to the current production process or if another framework for process workflows execution would be used.