# Prediction of the tool change point in a polishing process using a modular software framework

Meier, Nicolas; Papadoudis, Jan; Georgiadis, Anthimos

*Published in:*
Procedia CIRP

*DOI:*
10.1016/j.procir.2020.05.059

*Publication date:*
2020

*Document Version*
Publisher's PDF, also known as Version of record

Available online at www.sciencedirect.com

ScienceDirect

Procedia CIRP 88 (2020) 341–345

www.elsevier.com/locate/procedia

13th CIRP Conference on Intelligent Computation in Manufacturing Engineering, CIRP ICME '19

# Prediction of the tool change point in a polishing process using a modular software framework

Nicolas Meier[a,*], Jan Papadoudis[a], Anthimos Georgiadis[a]

[a]Leuphana University of Lueneburg, Institute of Product and Process Innovation (PPI), Volgershall 1, 21339 Lueneburg, Germany

* Corresponding author. Tel.: +49-4131-677-5419; fax: +49-4131-677-5300. E-mail address: nicolas.meier@leuphana.de

**Abstract**

A modular framework ensuring zero defect manufacturing developed for applications in different production processes was successfully tested on a polishing machine. Algorithms, predicting the optimal time to change the tools, were deployed to the framework and showed reproducible results under similar polishing conditions. Due to the modular software framework the algorithms, can be easily adjusted and changed, enabling the system to self-improve over time with minimal effort. The algorithms have been implemented into an embedded system.


Peer review under the responsibility of the scientific committee of the 13th CIRP Conference on Intelligent Computation in Manufacturing Engineering, 17-19 July 2019, Gulf of Naples, Italy.

*Keywords:* Prediction; Tool change point; Polishing; Modular software.

## 1. Introduction

Organizations that invest in advanced manufacturing technology will have improved competitive capabilities and better performance than firms that do not [1]. In response to that a number of automation solutions have been developed and successfully implemented over the years. Completely unattended manufacturing hasn't been widely achieved until now; current systems are decision support systems which relay on a human-based interaction [2]. Quality–related dimensions and other parameters are still checked manually, so that the human can adjust the machine parameter to ensure the quality [3]. An improvement would be the real-time sensing and control of quality parameters [4].

To achieve this, several sensors to monitor the machine status are needed as well as software for signal processing with an integrated optimization algorithm which can interact autonomously with the machine. Currently the software must be specifically written for each machine and for each application. In this paper, the integration of a novel framework, offering the possibility for adaptable modularized software [5,6,7], described in [8], into a polishing machine, is presented. The software was easily adapted to the used polishing machine and an algorithm to evaluate the state of the polishing process was integrated into to framework, to directly control the polishing robot, realizing a real-time control of the process. The software was used to determine a time frame for a tool change and to detect breakage of the polishing stone.

## 2. Concept of the framework

The software framework was developed to be adapted to different use cases. Thus a variety of applications were studied to identify main tasks. Three main tasks were identified, which were integrated into the developed framework. These tasks correspond with different control loops. These loops are:

- The real time control loop.
- The medium term optimization loop.
- The long term optimization loop.


Peer review under the responsibility of the scientific committee of the 13th CIRP Conference on Intelligent Computation in Manufacturing Engineering, 17-19 July 2019, Gulf of Naples, Italy.
10.1016/j.procir.2020.05.059

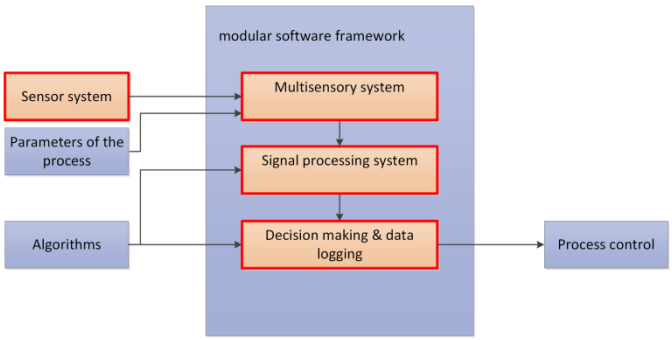


Fig. 1. Components of the modular framework (red). Algorithms and the parameters of the process are input to the system.

The real time control loop adapts machine parameters within milliseconds. Thus it must be fully automatic in terms of signal sampling, signal processing and machine communication. This will ensure a proper reaction to an unwanted state of the machine, based on measurements of the vital process parameters.

The medium term optimization loop predicts the process behavior based on simulation or extrapolation approaches, which are taking knowledge from previous measurements into account. This will ensure the part quality across the entire process chain. This approach provides a global optimization of the entire process chain and also is adapted to small lot sizes.

The long term optimization loop will also take human knowledge (e.g. of the operator) into account to ensure that also tacit knowledge of the process is included. This knowledge is used in machine learning algorithms to optimize machine parameters for multiple batches.

The ability to combine these different loops generates new possibilities for in-line process control in order to achieve zero defect manufacturing. The concept of the framework is shown in Fig. 1. The modules in red are part of the modular framework, with the sensor system as an optional addition. The framework needs as input the algorithms that should be used, as well as the description of the parameters of the process to control the process.

## 3. Integration

The integration into the polishing machine utilizes the concept of the real-time control loop and medium time optimization loop. Algorithms were developed and integrated to detect tool breakage as well as the optimal time for a tool change to improve the polishing process.

The aim of the software is to stop the polishing robot immediately if the polishing stone breaks and to predict an optimal time to change the tool, e.g. to use another grain size.

The general sequence of the software operations in the Software framework is the following:

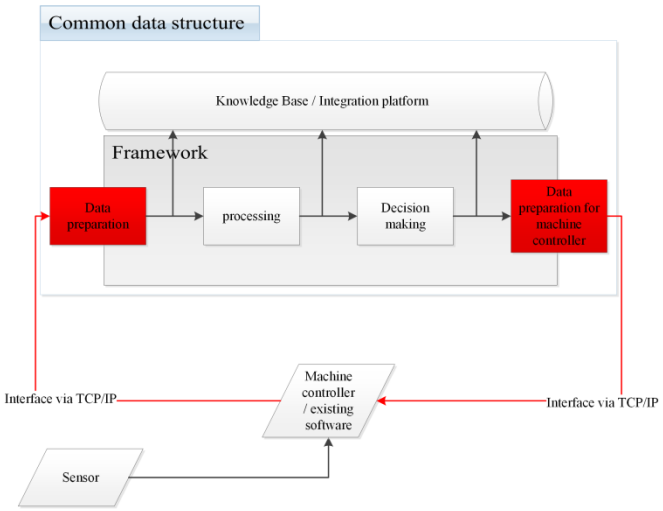


Fig. 2. Used modules of the framework to control the polishing machine. The red functions are for the interface to the ABB robot and needed initial configuration.

1. Listen to UDP broadcast for machines in the network.
2. Establish TCP connection.
3. Receive list of variables.
4. Register to variables that should be observed.
5. Monitor sensor data.
6. Evaluate algorithms.
7. Send decision to robot.

The scheme of the integration is shown in Fig. 2 including the modules necessary for the integration into the production environment (red).

### *3.1. Experimental Setup*

The machine monitoring was realized using an acoustic emission sensor (Fuji Ceramics Corporation R-CAST M304A) and a force sensor (using strain gauges) which were directly mounted on the polishing arm of the robot. The sensor data was sampled using a National Instrument DAQ connected to the ABB polishing robot.

The experimental setup was used to test the system regarding following 4 properties:

1. Long term reliability: The system should run in long term during various polishing processes. Long term is considered as ~10000 measurement values roughly corresponding to >15minutes of polishing
2. Interfacing: Different information are send from the RAP machine to the system to test if the system parses the protocol correctly
3. Integration of algorithm into the framework: The algorithms which were developed and tested in Matlab are (automatically) converted to C++. This test evaluates if the algorithm in Matlab and C++ gives the same results.

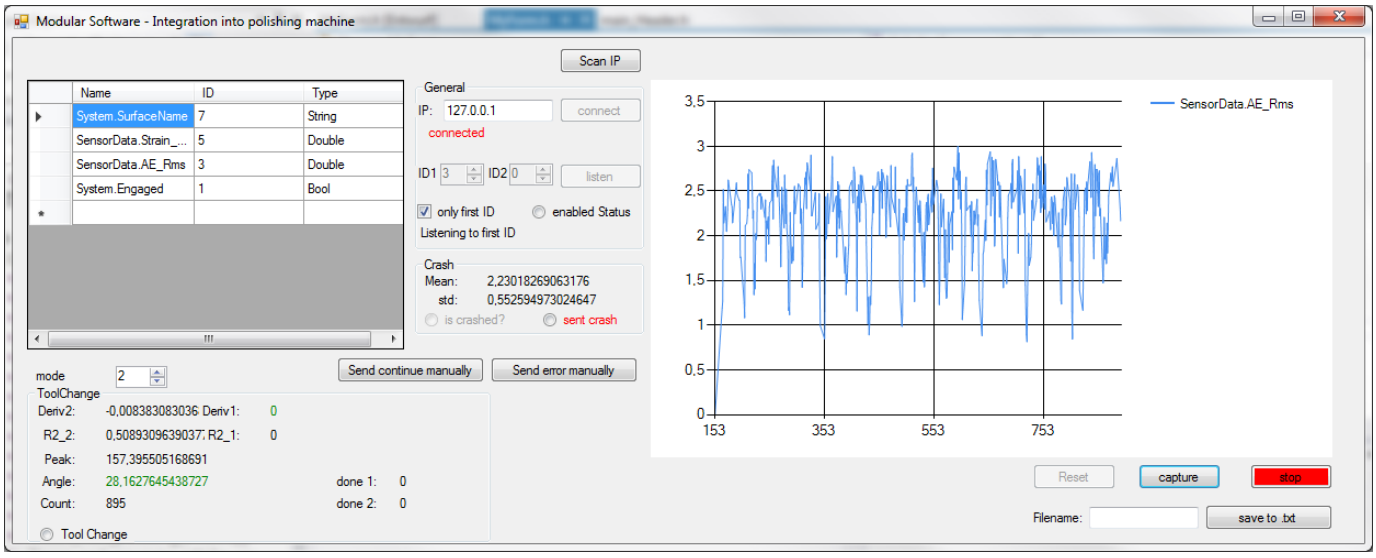


Fig. 3. Example GUI of the modular framework to control the polishing process. A list of variables can be monitored and the results of the used algorithms are shown.

4. Repeatability and performance of the algorithm: Testing how fast the system reacts after receiving an order from the machine, and how fast the machine reacts on stop signals. Additionally the output of the algorithm in different test scenarios is compared.

The developed software framework including the evaluation algorithms was connected via Ethernet with the polishing machine.

The setup of the polishing process consists of a steel roller "Vanadis 4E" with a diameter of 37,2 mm. The polishing stone is a #600 from Gesswein no 435-5601. Polishing was done with 800 g force, 1000 pulses per minute, 1 mm/s feed rate and rotating with 400 rpm.

## 4. Software

The software framework was developed in C++ with the ability to integrate directly algorithms which were converted to C++ from Matlab, without adjustments. The algorithms were developed in Matlab and tested in C++, assuring the ability to be used in production environment. The integration of the modular framework as well as the possible integration of signal processing algorithms is explained in detail in [8]. A GUI to visualize the monitored sensor data and the algorithms was designed for testing purpose, shown in Fig. 3. On startup the software only needs to be connected via Ethernet to the polishing machine.

### *4.1. Signal Processing*

An approach to determine the tool change time was integrated into the software, so a complete loop/process could be tested. The algorithm was developed with MATLAB and then automatically converted into C++ and integrated into the framework.

The algorithm itself calculates a polynomial regression function 2nd order of the logarithmic force data.

$$f(x) = ax^2 + bx + c \tag{1}$$

Where $x$ represents the sampling points and a,b,c are calculated using LSM. The main task is to find criteria on which the algorithms detect the tool change time without a static threshold, because the data vary a lot in different tests.

The algorithm calculates the minimum $m_r$ of the regression function. The position of the minimum corresponds to a sampling point. If the position of the minimum is after the current sampling point $s_c$ it means, that the trend of the force data is still decreasing and therefore the polishing process is not done. The easiest way to calculate this point is, to determine the angle $\varphi$ of the point$(m_r, s_c)^T$, where $x$ and the real sampling points form a coordinate system, so that:

$$\varphi = tan^{-1}\left(\frac{-b}{2as_c}\right) \tag{2}$$

If $\varphi > 45°$ $m_r$ and $s_c$ are not equal and the polishing process should continue.

The algorithm also calculates a regression 1st order and makes a decision based on the derivation of the regression function. If the derivation is close to zero, it indicates that the polishing process is done.

A combination of both approaches, taking into account the correlation coefficient of each regression, generates the final decision for a tool change.

To determine the stone breakage, the assumption was made, that in case of a breakage the acoustic emission respectively the force changes abruptly. The mean value and the standard deviation are constantly calculated using the last 300 data points. If the next data point differs more than two times the standard deviation from the mean it indicates a breakage.

## 5. Results

The described software framework was initially tested for long term monitoring. 3 tests of 15 minutes of polishing were carried out, which results in over 10 000 data points. To verify if the software captured all data points correctly, the sensors were simultaneously connected to the software of the sensor manufacturer. In all tests all data were captured successfully. Thus the software could be used to control the robot and further tests were done.

To assure real-time capabilities the reaction time has to be evaluated. To do so, the software sends a stop signal to the machine. After the robot stopped it sends a confirmation signal. The time was measured between the software was sending the signal to the robot and receiving the signal back. So in reality the robot stopped earlier due to the signal propagation time. Different test setups were used with different connection types and sampling frequencies.

Table 1. Result on the reaction time tests.

| Sampling rate \ connection type | Time [ms] | | | successful |
|---|---|---|---|---|
| | Same machine | cabled | Wireless | |
| 0 Hz | 20 | 20 | 100 | **Yes** |
| 10 Hz | 20 | 100 | 200 | **Yes** |
| >100 Hz | 100 | 200 | 400-700 | **Yes** |

It can be seen, that the connection type significantly impacts the duration. Depending on the needed reaction time and sampling frequency the software should be deployed at the appropriate place. However the tests showed that the software can easily be used on a remote device, e.g. a tablet, for process monitoring, with a reaction time within 400-700ms even higher sampling rates can be observed for remote monitoring or to make adjustments which are not time critical. With a reaction time of 20ms the software and the algorithms can be used for real time adjustments of the polishing process.

### *5.1. A. Algorithm*

The algorithm was developed in Matlab and integrated into the modular framework hence there are two possibilities to validate it: Directly in Matlab and afterwards in process.

Initial tests, which were carried out, should validate if the algorithm makes similar decisions under similar conditions. 12 evenly distributed stripes over the steel roller, 20 mm in size, were polished. Each stripe was polished with 80 passes, resulting in roughly 12000 data points per stripe. As all stripes had similar initial surface roughness, the algorithm should give related results for each stripe.

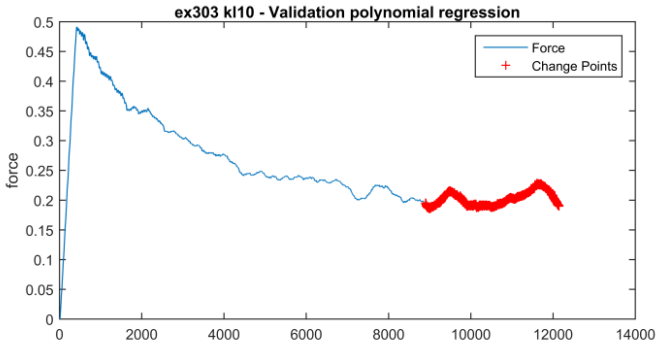


Fig. 4. Measured force data during polishing. The red dots show a steady state of the process indicating the tool change time.

The first sampling point which fulfills the requirements stated in (1) was taken as reference for the decision point. Tests over 12 rollers, each with 12 stripes, showed that the algorithm is stable and generate reproducible results. The mean of all decision were at data point 8800 with a standard deviation of 900 data points. As one pass equals 600 data point, the algorithm detects a tool change point within the accuracy of 1-2 polishing passes.

In Fig. 4 the force data of one experiment is shown. The red “change points” indicate positions that fulfill the requirements mentioned in the previous chapter. To eliminate the probability of errors, the command to change the tool was sent when 250 consecutive data points fulfill the requirements.

The algorithm for the tool breakage was tested by using polishing stones with predetermined breaking points. In this scenario the breakage was detected every time with the described algorithm and the robot stopped. However data where the polishing stone broke unintentionally could not be provided at this stage, so that this algorithm was not tested under real circumstances.

## 6. Conclusion

A modular framework, which was developed to be integrated in different production processes, was successfully integrated into a polishing machine. Algorithms developed in Matlab to detect the optimal tool change time were easily deployed to the framework and showed reproducible results under similar polishing conditions. Due to the modular software framework the algorithms, can be easily adjusted and changed, enabling the system to improve over time with minimal effort.

## 7. Acknowledgment

The presented framework was developed within the „IFaCOM“-Project of the 7th Framework Program from the European Union.

The software was applied and adapted to different demonstrators throughout a variety of manufacturing

processes. The framework was successfully verified. The presented algorithm was successfully tested and implemented on one of the end-users polishing robot.

## References


[1] Tracey M, Vonderembse MA, Lim J. Manufacturing technology and strategy formulation: keys to enhancing competitiveness and improving performance. In: Journal of Operations Management, Volume 17, Issue 4; 1999. p. 411-428.

[2] Monfared MAS., Yang JB. Design of integrated manufacturing planning, scheduling and control systems: a new framework for automation. In: The International Journal of Advanced Manufacturing Technology; 2007. p. 545-559.

[3] Sheridan TB, Parasuraman R. 2: Human-Automation Interaction. In: Nickerson RS, editor. Reviews of Human Factors and Ergonomics. Santa Monica, CA, USA: Human Factors and Ergonomics Society; 2006. p. 89–129.

[4] Ulsoy AG. Monitoring and Control of Machining. In: Condition Monitoring and Control for Intelligent Manufacturing, first edition. London: Springer.; 2006. p. 1-32.

[5] Liang SY, Hecker RL, et al. Machining Process Monitoring and Control: The State-of-the-Art. In: ASME 2002. Fairfield: International Mechanical Engineering Congress and Exposition; 2003.

[6] Lee J, Ni J, Djurdjanovic D, Qiu H., Liao H. Intelligent prognostics tools and maintenance. In: Computers, Industry, Vol. 57; 2006: p. 476-489.

[7] Nan C, Khan F, Iqbal MT. Real-time fault diagnosis using knowledge-based expert system. In: Process Safety and Environmental Protection, Vol. 86; 2008.

[8] Papadoudis J, Georgiadis A. Distributable Modular Software Framework for Manufacturing Systems. In: Procedia CIRP, Volume 41; 2016. p. 712-716.