Detecting Out-Of-Control Sensor Signals in Sheet Metal Forming using In-Network Computing

Ike Kunze*, Philipp Niemietz†, Liam Tirpitz*, René Glebke*, Daniel Trauth†, Thomas Bergs†, Klaus Wehrle*
*Communication and Distributed Systems · {kunze, tirpitz, glebke, wehrle}@comsys.rwth-aachen.de
†Laboratory for Machine Tools and Production Engineering (WZL) · {p.niemietz, d.trauth, t.bergs}@wzl.rwth-aachen.de
All authors are affiliated with *RWTH Aachen University*, Aachen, Germany

***Abstract*—The ongoing digitization of production enables the collection of increasing volumes of data. These, in turn, allow for data-driven analysis that has the potential for deepening the process understanding by discovering previously unknown connections between process components and parameters. With these opportunities, however, come substantial challenges as current industrial settings are inadequately equipped for handling these large amounts of data. While setting up a local processing infrastructure is challenging, the limited bandwidth within many shop floors as well as their network access also make an upload of all data to external compute capacities infeasible. What is needed are local, process-aware filters that allow for significant data reduction while retaining data of value that can be used for the subsequent analysis. In this paper, we thus propose to leverage In-Network Computing to dynamically detect different states of the physical processes and then filter the sensor values on the data path. Our presented architecture maps the state detection to the switch-local controlplane while fast filtering decisions are performed at line-rate in the dataplane, thus enabling flexible and quick adjustments of the chosen sensor filtering. At the example of a fine-blanking line, we consequently demonstrate that In-Network Computing can sensibly support previously infeasible data analysis techniques in the industrial production landscape.**

***Index Terms*—Industrial communication, Communication systems, Edge computing, Software defined networking, Pattern recognition, Condition monitoring, Clustering methods, Process monitoring**

I. Introduction

Modern production systems are increasingly equipped with sensor telemetry that allows for closely monitoring the industrial processes. Concepts such as the Industrial Internet of Things (IIoT) [1] and Industry 4.0 [2] advocate for increasingly interconnecting these systems locally, utilizing principles of the Internet of Things (IoT). Going a step further, the Internet of Production (IoP) [3] proposes to form a global network of industrial systems that also crosses company borders. The main benefit of these concepts is that the available abundance of data can be connected and then used to create knowledge — an unachievable improvement if data stays in isolated data silos. The derived knowledge can then be fed back to, e.g., improve the production processes or the product quality.

At the same time, handling these enormous amounts of data raises fundamental challenges [4]. Most imminently, existing data analysis frameworks assume a logical centralization of the data which thus has to be moved from its sources to central locations [5]. This moving of data can severely strain or even overload existing communication infrastructures, especially if off-premise compute capacities are only reachable via constrained network connections. Alternatively, companies have to make significant investments to either establish on-premise compute resources or upgrade the communication infrastructure. Consequently, relying solely on either option is infeasible. Thus, companies are in need of a middle ground that resorts to external resources when necessary, but simultaneously tries to keep the load on the infrastructure low by sensibly reducing the transmitted data volume.

We tackle this goal by leveraging the novel In-Network Computing (INC) paradigm, i.e., utilizing compute resources in networking devices as well as their privileged position on the data path, to achieve meaningful data reduction that is precisely oriented at the application needs. For this, we concretize our previous proposal of integrating data processing into the data flow [6] and devise well-defined INC tasks to enhance the operation of a fine-blanking manufacturing system. More specifically, this paper contributes the following:

- We analyze the seasonal patterns of force signals captured at our fine-blanking system and identify varying required information levels for different process intervals.
- We design an INC pipeline that can detect these phases based on the seasonal patterns and subsequently adjusts the resolution of the forwarded information.
- We identify intriguing aspects for future research that are relevant beyond our own work.

Structure. In Sec. II, we introduce the fine-blanking system as a representative for sheet metal forming processes and identify and analyze the seasonal patterns of sensor values. We then discuss how related work handles such periodic data in Sec. III before presenting the INC paradigm in more detail in Sec. IV. Based on our findings on the fine-blanking system, we then present our approach that is capable of filtering data exactly to the needs of the application. After laying out our future research directions in Sec. VI, we finally conclude the paper.

II. Seasonal Patterns in Fine-blanking

Sheet metal forming processes are used for mass production in automotive and aerospace industry, e.g., for large car body parts or small safety critical components for breaks or engines [7]. Such processes are, above all, characterized by their stability: once the tool and its process parameters are set up correctly, the process can produce parts within the tolerance range for thousands of strokes. However, process forces as

 DOI: 10.1109/ISIE45552.2021.9576221.

well as the resulting workpiece quality vary substantially over a series of strokes, and these variations, often referred to as process noise, are not yet completely understood by engineers [8]. The complex interplay of tool design, wear progression of its active components, material properties, and the fact that the process is not easily accessible for direct measures of important system parameters hitherto hinders the understanding of such variations. Yet, a better understanding in this domain can support engineers to further push the tolerance limits of the process and thus strengthen economic efficiency.

Fine-blanking. In this study, we focus on the fine-blanking process which is used for parts with strict quality requirements to the shearing surface that are typically important for functionality of the corresponding workpieces. Fine-blanking is characterized by the *flow shear process* that results in a clean-cut and takes place in a compressive stress-dominated process area [7], leading to a shearing surface with up to no tear-off. This high quality of the shearing surface makes subsequent finishing processes unnecessary and cuts down production time and costs. In contrast to conventional blanking, fine-blanking presses are triple-acting machines: the V-ring force (*blank holder*) and *counter punch* forces are generated hydraulically, and the main blanking (*punch*) force can be either mechanical or hydraulic. Fig. 1 illustrates the interaction of these forces.

Process Monitoring. The complex interaction of the three process forces makes monitoring them particularly interesting and research has shown that such sensor signals contain valuable information about the working conditions of the process. These include the wear state of the tool [9], quality features of the resulting workpiece [10], machine faults, misfeed, and thickness variations [11]. Our fine-blanking tool is enhanced with substantial sensor equipment that can, among others, measure the force signals, acoustic emission, and acceleration [12]. The multitude of sensors combined with their high frequencies and high sample rates produces very large amounts of data [6] which makes the application of sensible data reduction techniques an intriguing option. For ease of illustration, we focus on force signals for the rest of this paper.

Seasonal Sensor Patterns. Due to the repeating nature of fine-blanking of an almost identical process, the occurring seasonal patterns in the signals are the starting points for any analysis and often referred to as sensor profiles [13]. As an example, Fig. 2 shows the sensor profiles of the three active process forces of fine-blanking which will, to a high degree, repeat stroke-by-stroke. Consequently, even minor anomalies, i.e., deviations from or changes of these characteristic patterns can represent changes in the underlying physical conditions of the process and are important to track and monitor in detail.

Process Phases. Anomalies are more likely to occur when the process is not in the ideal operating state as illustrated in Fig. 3. After a cold-start, sheet metal forming processes first require a *ramp-up time* to heat up the machine to its operating temperature and corresponding changes in temperature directly affect important material and lubricant parameters. In Fig. 3, these impacts can be noticed, e.g., in the form of fluctuating and constantly increasing maximum punch forces and it is

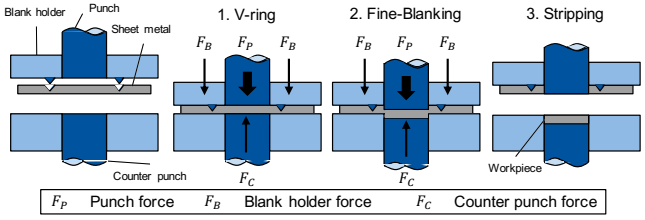

Fig. 1. The fine-blanking operation in detail: First, the material is clamped by applying the *blank holder* force. Second, the *punch* and *counter punch* forces are applied and the actual blanking process takes place. Lastly, the sheet metal is stripped of the punch.

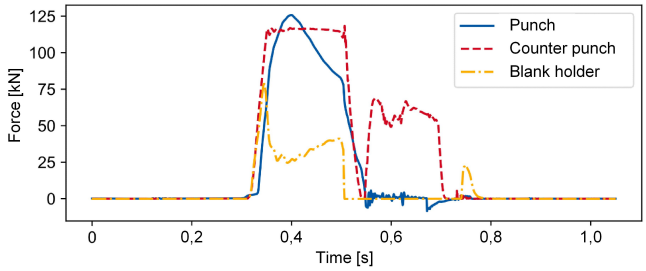

Fig. 2. The three active process forces (*punch*, *counter punch*, and *blank holder*) show seasonal patterns during fine-blanking of 3 mm thick steel.

important to monitor these fluctuations and their impact on the resulting workpiece quality.

Once the machine is in operating temperature (*in-control*), there are still variations and fluctuations, but they are less severe. Consequently, while they should still be monitored, an in-depth analysis for each and every blanking operation may not be required. However, as soon as process conditions change significantly, e.g., caused by lubrication errors or wear increase, force signals will change their characteristics accordingly (*out-of-control*). These changes are again of specific interest in the analysis and understanding of the process.

Takeaway. The different phases of the fine-blanking process contain different levels of information for the process analysis. Data reduction techniques that account for these phases, e.g., by identifying the *ramp-up* phase or when the process leaves the *in-control* phase, and correspondingly filter or preprocess the sensor signals thus have the potential of substantially reducing the amount of data that is transmitted and processed, while still keeping the data of interest available.

In the following, we discuss approaches by related work to cover these requirements.

III. Related Work in Industrial Data Processing

The periodic structure of the measured force signals as well the general underlying streaming characteristics of our fine-blanking setup are perfect candidates for the application of stream data processing techniques. Well-known frameworks, such as Apache Storm or Apache Flink, however, assume a logical centralisation of the processed data, i.e., it first needs to be collected, before it can be analyzed. In the context of the IIoT and the IoP, however, ever increasing data volumes

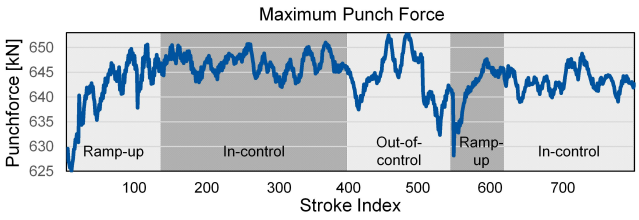

Fig. 3. Based on the fluctuations of the maximum punch force during fine-blanking of 3 mm thick steel, three execution phases can be identified: *ramp-up*, *in-control*, and *out-of-control*. Note that the machine was paused and restarted after roughly 550 strokes.

pose novel challenges for the data analysis frameworks [4]. In particular, the application of generic data stream processing techniques is no longer seen as a universally feasible option [5]. Among other reasons, uploading all data to powerful data centers that are capable of (i) handling the large volumes, and (ii) performing the required analytical tasks, is inhibited by upload bandwidth limitations of the companies. Thus, while data analysis in the data center is possible, collecting the data and bringing it to the data center is challenging. What is consequently needed are ways to cope with the large amount of data in a local context while reducing the amount of required processing capacities.

One solution to the large data volumes is tuning the granularity of the recorded data to the exact level that is needed for analysis. Lipp et al. [14], e.g., investigate the possibility to tune sensor reading intervals of an OPC-UA based system to inherent process phases of the controlled process. Using well-chosen reading intervals for the different phases, this approach allows for a significant data reduction while simultaneosuly keeping the exact amount of data necessary for performing the required analytical tasks. However, it builds upon statically defined phases with fixed phase transitions and its focus on data extraction does not allow for data preprocessing.

As an alternative solution, Qi et al. [15] propose combining edge-based analysis and filtering steps with a more complex analysis leveraging powerful cloud environments. While their approach addresses the problem, the use of edge resources for filtering the data only shifts the bottleneck, because the edge devices still need to handle significant data volumes [6] and the company infrastructure is, in this case, still flooded with sensor data. To address this issue, the emerging research area of INC focuses on substantially reducing the network load even before the data reaches the edge devices.

IV. Recent Advances In In-Network Computing

In-Network Computing (INC) advocates for deploying sensible functionality on networking devices which are capable of processing very high data rates and already lie on the data path. Data thus does not have to be redirected to edge devices for filtering, making INC a prime candidate for our use case.

In recent years and, especially, following the introduction of the P4 programming language [16], networking devices have become increasingly programmable and there is a growing market of P4 programmable devices. These range from purely software-based solutions, such as the P4 behavioral model (bmv2) [17], to smart network interface cards, such as the Netronome Agilio series [18], that map the bmv2 to special packet processors. There are even first programmable switching ASICs, such as the Intel Tofino [19].

P4 Principles. The core design choice of these platforms is a separation of concerns between a high-speed *dataplane* and slower *controlplane*. While the former can process packets at line-rate, i.e., up to several hundreds of Gbit/s in the case of Tofino for example, only simple operations are possible, e.g., multiplications are often already too complex [20]. In contrast, the latter typically runs on a standard CPU co-located with the dataplane and can thus perform arbitrary computations albeit at much slower and less predictable speeds. Finding a clever division of labor between the dataplane and the controlplane by leveraging the individual strengths of the two components can thus be very beneficial for application performance.

Research Focus. Most research investigates the applicability of INC to core networking tasks, such as active queue management [20], load balancing [21], or heavy-hitter detection [22]. However, there is also substantial work that broadens the scope of INC to application-level functionality, including the deployment in industrial systems [23]–[25].

Of particular interest for our work is the finding that INC aggregation tasks co-designed with the overall analysis function can provide significant benefits [26]. These findings nicely fit into our previously proposed general architecture [6] in which we argue that integrating processing steps into the data path can be an effective measure for addressing the data volume challenges laid out before. In this work, we thus concretize our general architecture by matching requirements and analysis steps of the fine-blanking tool (see Sec. II) to the capabilities of programmable networking devices. In the following, we present our concrete approach for enhancing a fine-blanking tool monitoring system using INC.

V. Detecting Sensor Signals In The Network

Our fine-blanking tool produces large amounts of data that follow a seasonal pattern (see Sec. II). Additionally, we have identified different process phases with different requirements regarding the forwarded data volumes. Based on these findings, we now aim to shape a filtering and preprocessing architecture that is capable of acquiring the right volume of data at the right moment. More specifically, we aim to detect the different process phases of our fine-blanking tool based on a stroke-by-stroke level (see Fig. 3) and enhance the ensuing analysis by leveraging the actual seasonal pattern of the force signals (see Fig. 2). Before we present the architecture in more detail, we first condense our previous findings on the different phases and their required information levels.

A. Process Phases And Information Levels

While our fundamental approach is also applicable to other systems, its effectiveness is highly dependent on a precise

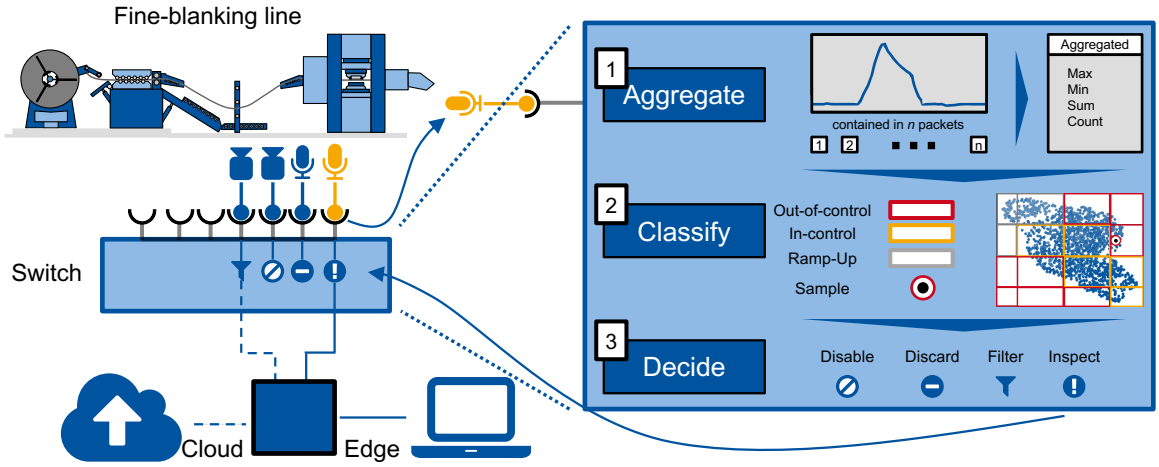

Fig. 4. Analyzing the data already in the network allows to quickly react to different machine states. (1) Our approach first *aggregates* all sensor readings belonging to a single stroke into one aggregated packet. (2) Based on this packet, the subsequent *classification* step uses grid-based density clustering to detect the current phase of the fine-blanking line. (3) Depending on the current process phase, the switch can then *decide* on the dataplane how to process individual sensor readings.

tailoring to the concrete setting. In the case of our fine-blanking line, we set the following objectives:

Identify *Ramp-Up* Phase. As presented in Sec. II and visualized in Fig. 3, the operation of our fine-blanking tool begins with a *ramp-up* phase in which the process heats up to its operating temperature. In this phase, the changes in temperature, e.g., cause the force signals to steadily increase and fluctuate significantly, leading to a higher number of expected anomalies. This warrants a closer inspection of the produced signals to enable a fine-grained analysis, e.g., regarding the quality of the produced product.

***In-Control* Phase Sampling.** After the end of the *ramp-up*, the machine enters a steadier *in-control* phase. In this phase, there are only minor fluctuations and the produced products have a high quality. Here, we do not have to closely inspect each stroke in detail and can thus reduce the signal resolution, e.g., by sampling at lower frequencies, only inspecting every other stroke, or even completely discarding sensor signals.

***Out-Of-Control* Phase Detection.** During the *in-control* phase, the physical characteristics of the process can change over time, e.g., due to malfunctioning components or steady wear increase of tool components. It is important to quickly detect these *out-of-control* phases and subsequently increase the resolution of the signals again to allow for closer inspection and complex data analysis.

Combined Goals. To summarize, we aim to (i) identify the different process phases by comparing force signals on a stroke-by-stroke level, and (ii) adjust the resolution of the forwarded sensor information accordingly. In the following, we present our design that leverages both dataplane and controlplane resources to allow for more sophisticated data analyses of a variety of sensors.

B. Architecture

The central component of our architecture is a programmable networking switch that is capable of first identifying the current state of the fine-blanking operation and then performing appropriate tasks to address the situation. We illustrate a high-level look on this structure on the left side of Fig. 4: several sensors monitor the fine-blanking tool and transmit their readings to the switch which, in turn, decides whether to discard, filter, or forward the readings. Additionally, it can be either manually or automatically decided to discard or activate specific sensors, e.g., if a sensor is only interesting during specific events. The decision process is (i) distributed among the dataplane and the controlplane, and (ii) internally divided into the three steps shown on the right side of Fig. 4. In the following, we discuss the individual steps in more detail.

1) Aggregate: A single stroke operation of the fine-blanking process consists of more than 10 000 individual sensor readings. Performing a complete stroke-by-stroke comparison thus requires holding these readings in memory which is not feasible on current P4-based platforms. Consequently, we opt to first detect individual strokes based on their sensor profile. We then aggregate characteristic values from all force values of a stroke to derive an abstract representation for subsequent pipeline steps. As this procedure needs to process all sensor readings, it is realized entirely in the dataplane.

Stroke identification. Once a fine-blanking process has been set up, operation durations and sensor profiles remain similar. We can thus simply identify the start of a stroke by detecting a change from a value close to zero to a larger value, accounting for the steep force increase at the start of the stroke (see Fig. 2). Similarly, we assume the end of the stroke if the force values stay close to zero for a number of consecutive sensor readings.

Aggregation. During a stroke, the dataplane maintains the maximum and minimum forces, the sum over all force values observed for a single stroke, which is equivalent to the total force applied on the sheet metal, as well as the number of data points considered. When the end of a stroke is detected, these characteristic values are transmitted to the next processing

step in the form of a single, aggregated packet. Each packet entering the second step, now represents a single stroke.

2) Classify: The second processing step applies a clustering algorithm to the aggregated, multidimensional data points to identify the current process phase. Since programmable switches lack support for loop-based programming [27] and have limited computational capabilities, most established algorithms for clustering, or even stream-based clustering cannot be implemented solely on the dataplane. However, as described in Sec. IV (P4 Principles), we can distribute compute tasks between the dataplane and the controlplane. The *grid-based density clustering (GBDC)* approach by Chen et al. [28] differentiates between *online* and *offline* processing and is thus a well-suited candidate for this purpose.

GBDC – Online. The *online* component of the algorithm is designed for streaming data; thus, it does not require loops and only uses simple calculations. In short, every observed dimension is divided into n sections, resulting in a grid of d-dimensional partitions. Instead of maintaining the position of every individual data point in this d-dimensional data space, the approach only maintains the so-called *density* for each partition which is an indicator for the number of data points that recently hit that partition. To focus on recent data points, the densities of unhit partitions are constantly reduced. We map this online component and the corresponding maintenance of partition densities to the dataplane.

GBDC – Offline. The *offline* component of GBDC uses the previously derived densities to compute clusters from neighboring partitions with high density. As this process does not need to work on individual data items, yet requires higher computational capabilities, we map it to the controlplane. In fact, using the controlplane generally allows for deploying algorithms of arbitrary complexity for these computations.

By regularly computing the current clustering, the controlplane can identify phase transitions and trends in the aggregated force data. For example, a large number of medium density partitions could indicate an out-of-control process while a small number of neighboring partitions with high density would indicate a stable process. Similarly, by comparing successive "snapshots", the controlplane can detect trends such as a rising force, indicating the ramp-up phase. If the controlplane detects a phase change, it will notify the dataplane of the currently detected phase which can then adjust the handling of the raw sensor data.

3) Decide: Following the classification by the controlplane, per-packet decisions are again solely performed by the dataplane to reduce the data volume in the network and maximize the utility of the transmitted data. In the *in-control* phase, e.g., detailed force data might not be required and the switch may (a) drop all packets with individual force values and only communicate the machine state via the aggregation packets generated in Sec. V-B1, or (b) sub-sample the raw data and only transmit a smaller number of individual readings. If an unstable process state is detected, the dataplane will redirect the unfiltered force values to an external system for further analysis, thus ensuring an in-depth analysis of critical data.

By combining the strengths of the fast dataplane and the more powerful controlplane, our pipeline can handle large amounts of raw data and react quickly to phase shifts in the process while limiting network communication to useful data.

C. Limitations

Since our approach has to conform to the capabilities of the dataplane in multiple regards, certain limitations apply. First, grid-based clustering is, by design, an approximation, because it does not preserve the individual data points. Therefore, the size of the individual partitions determines the accuracy of the resulting clustering. While smaller partitions increase accuracy, they also lead to a larger number of partitions that need to be maintained. The number of partitions is limited by the resources of the programmable switch. Furthermore, the density of the individual partitions can only be approximated due to the limited arithmetic capabilities on the dataplane. Despite these limitations, we expect the accuracy of our approach to be sufficient for our application.

VI. Towards Deployment in Industrial Production Lines

The basis of our work is a proof-of-concept implementation that is generally able to perform different types of preprocessing on artificial sensor data. In this context, we are currently working on adjusting this proof-of-concept for the overall architecture described in Sec. V. While we first focus on realizing a simple variant of our proposed architecture, we plan on fully investigating the intriguing potential for future research that is offered by this setting.

Phase Detection. In a first step, we implement and test the identification of different process phases which is already challenging due to fluctuations in the signals. Thus, we will investigate which state identification variants are possible and reasonable for deployment on networking hardware. Available options range from simple threshold-based variants to variants directly considering the fluctuations between consecutive strokes. We compare the different options using a data set containing over $40\ 000$ individual fine-blanking strokes.

Impact Of Preprocessing. In a second step, we investigate the implications of different filtering and preprocessing techniques on subsequent analysis methods that involve complex feature engineering and machine learning pipelines. To this end, we plan to specifically evaluate whether decisions to sub-sample or drop sensor data made by our platform negatively influence existing monitoring approaches. Improved understanding of the interaction between the preprocessing and the analysis will allow for devising better, custom-tailored solutions and carefully chosen trade-offs, e.g., accepting slightly worse analytical accuracy for a significant reduction of the data volume. In this context, incorporating additional signal processing steps might be beneficial, e.g., entirely filtering out the "idle" phases before and after each stroke or cleverly dividing one stroke in several sub-segments with different sensor resolutions.

Quality Prediction. Going beyond the presented scope, our design also allows us to incorporate previously unexplored

facets of the problem. For example, sensor information is often used to predict the quality of the resulting workpiece. Including simple quality prediction tools into our pipeline could enable an on-demand up- and down-scaling of information resolution depending on the expected product quality on a stroke-to-stroke basis so that products with potential issues could be inspected in higher detail even if they fall into an *in-control* phase where our current approach would scale down the information level.

Multi-Sensor Settings. Another property that can be utilized is the complex correlation structure of the process force in fine-blanking. While previously known higher correlation in signals might hint at redundancy between sensor signals, the auto-correlation of signals themselves can be used to remove redundancy within the data of a signal. A complex historical analysis of sensor signals and a deployment of this complex correlation structure may further advance the capabilities of switches to conduct detailed definitions on the basis of large streaming data within the dataplane.

Takeaway. Overall, we have a fixed roadmap at hand to turn our envisioned INC-based data processing of sensor readings into an approach that is ready for deployment in real-world production lines. Thus, we provide an important next step to improve industrial processes, such as fine-blanking lines.

VII. Conclusion

When turning the vision of an Internet of Production into reality, an important step is to leverage the vast amounts of available data, e.g., to identify issues in the production process or, more general, develop a better understanding of the process. The presented fine-blanking line is one use case that is well-suited for these benefits. Yet, handling the sheer amount of data can cause problems as network access and bandwidths of companies are often too limited. To still benefit from the available data and information, we thus propose to scale and preprocess the data based on the different process phases of our fine-blanking line. Using In-Network Computing, we can detect these phases on the data path and subsequently perform a timely preprocessing on a sub-signal level.

Our outlined roadmap to realize the introduced approach allows us to turn these envisioned benefits into reality. Using a rich dataset of real-world fine-blanking process data, we will evaluate its effectiveness in real-world settings. Specifically, we will investigate (i) different techniques to robustly detect the process phases, (ii) the implications of our filtering and preprocessing on subsequent analysis pipelines, (iii) additional facets of the problem, such as quality prediction, that might be addressable using our approach, and (iv) the extension of the pipeline to multiple synchronous and asynchronous sensors.

Overall, we demonstrate that deploying In-Network Computing in industrial settings for filtering and preprocessing tasks is feasible and that it has the potential of enabling previously infeasible data analysis techniques while protecting the local networks from overloading.

Acknowledgment

Funded by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) under Germany's Excellence Strategy – EXC-2023 Internet of Production – 390621612.

References

[1] E. Sisinni et al., "Industrial Internet of Things: Challenges, Opportunities, and Directions," *IEEE TII*, vol. 14, no. 11, pp. 4724–4734, 2018.

[2] H. Lasi et al., "Industry 4.0," *BISE*, vol. 6, no. 4, pp. 239–242, 2014.

[3] J. Pennekamp et al., "Towards an Infrastructure Enabling the Internet of Production," in *IEEE ICPS*, 2019, pp. 31–37.

[4] S. Verma et al., "A Survey on Network Methodologies for Real-Time Analytics of Massive IoT Data and Open Research Issues," *IEEE COMST*, vol. 19, no. 3, pp. 1457–1477, 2017.

[5] A. Feldmann et al., "Enabling Wide Area Data Analytics with Collaborative Distributed Processing Pipelines (CDPPs)," in *IEEE ICDCS*, 2017, pp. 1915–1918.

[6] R. Glebke et al., "A Case for Integrated Data Processing in Large-Scale Cyber-Physical Systems," in *HICSS*, 2019, pp. 7252–7261.

[7] F. Klocke, *Manufacturing Processes 4: Forming*. Springer Science & Business Media, 2014.

[8] T. Bergs et al., "Punch-to-Punch Variations in Stamping Processes," in *IEEE SAMI*, 2020, pp. 213–218.

[9] B. Voss et al., "Using stamping punch force variation for the identification of changes in lubrication and wear mechanism," in *J. Phys. Conf. Ser.*, vol. 896, no. 1, 2017, p. 012028.

[10] J. Havinga et al., "Estimating Product-to-product Variations in Metal Forming using Force Measurements," in *AIP Conference Proceeding*, vol. 1896, no. 1, 2017, p. 070002.

[11] A. Bassiuny et al., "Fault diagnosis of stamping process based on empirical mode decomposition and learning vector quantization," *Int. J. Mach. Tools Manuf.*, vol. 47, no. 15, pp. 2298–2306, 2007.

[12] P. Niemietz et al., "Stamping Process Modelling in an Internet of Production," *Procedia Manufacturing*, vol. 49, pp. 61–68, 2020.

[13] S. Zhou et al., "An SPC Monitoring System for Cycle-Based Waveform Signals Using Haar Transform," *IEEE T-ASE*, vol. 3, no. 1, pp. 60–72, 2006.

[14] J. Lipp et al., "When to Collect What? Optimizing Data Load via Process-driven Data Collection." in *ICEIS*, 2020, pp. 220–225.

[15] Q. Qi et al., "A Smart Manufacturing Service System Based on Edge Computing, Fog Computing, and Cloud Computing," *IEEE Access*, vol. 7, pp. 86 769–86 777, 2019.

[16] P. Bosshart et al., "P4: Programming Protocol-Independent Packet Processors," *ACM SIGCOMM CCR*, vol. 44, no. 3, pp. 87–95, 2014.

[17] p4language, "Behavioral model (bmv2)," 2021. [Online]. Available: https://github.com/p4lang/behavioral-model

[18] Netronome, "Agilio CX SmartNICs," 2021. [Online]. Available: https://www.netronome.com/products/agilio-cx

[19] Intel, "Intel® Tofino™," 2021. [Online]. Available: https://www.intel.com/content/www/us/en/products/network-io/programmable-ethernet-switch/tofino-series.html

[20] I. Kunze et al., "Tofino + P4: A Strong Compound for AQM on High-Speed Networks?" in *IFIP/IEEE IM*, 2021.

[21] N. K. Sharma et al., "Evaluating the Power of Flexible Packet Processing for Network Resource Allocation," in *USENIX NSDI*, 2017, pp. 67–82.

[22] R. Ben-Basat et al., "Efficient Measurement on Programmable Switches Using Probabilistic Recirculation," in *IEEE ICNP*, 2018, pp. 313–323.

[23] J. Rüth et al., "Towards In-Network Industrial Feedback Control," in *ACM NetCompute*, 2018, pp. 14–19.

[24] R. Glebke et al., "Towards Executing Computer Vision Functionality on Programmable Network Devices," in *ACM ENCP*, 2019, pp. 15–20.

[25] F. Cesen et al., "Towards Low Latency Industrial Robot Control in Programmable Data Planes," in *IEEE NetSoft*, 2020, pp. 165–169.

[26] A. Sapio et al., "In-Network Computation is a Dumb Idea Whose Time Has Come," in *ACM HotNets*, 2017, pp. 150–156.

[27] I. Kunze, "Investigating the Applicability of In-Network Computing to Industrial Scenarios," in *IEEE ICPS*, 2021.

[28] Y. Chen et al., "Density-Based Clustering for Real-Time Stream Data," in *ACM SIGKDD*, 2007, p. 133–142.