

Data-Driven Assistance Functions for Industrial Automation Systems

Stefan Windmann, Oliver Niggemann

Fraunhofer Application Center Industrial Automation (IOSB-INA), Lemgo

E-mail: {stefan.windmann, oliver.niggemann}@iosb-ina.fraunhofer.de

Abstract. The increasing amount of data in industrial automation systems overburdens the user in process control and diagnosis tasks. One possibility to cope with these challenges consists of using smart assistance systems that automatically monitor and optimize processes. This article deals with aspects of data-driven assistance systems such as assistance functions, process models and data acquisition. The paper describes novel approaches for self-diagnosis and self-optimization, and shows how these assistance functions can be integrated in different industrial environments. The considered assistance functions are based on process models that are automatically learned from process data. Fault detection and isolation is based on the comparison of observations of the real system with predictions obtained by application of the process models. The process models are further employed for energy efficiency optimization of industrial processes. Experimental results are presented for fault detection and energy efficiency optimization of a drive system.

1. Introduction

The growing capacity of cost-effective equipment and the requirements of production technology are resulting in increasingly complex distributed automation systems. A consequence of this trend is an ever-increasing data volume in industrial processes [1], which must be analyzed and taken into account in process control. This leads to ever growing demands on data acquisition, process monitoring, fault recognition and analysis of failure causes. A possibility to cope with these challenges is the application of assistance systems. As a matter of course, application and configuration of such assistance systems is required to be as simple as possible. A step into this direction is the usage of process models that are automatically learned from process data. Self-learning of process models from data leads to considerable simplification of model creation and configuration. Process models can be applied to several model-based assistance functions such as visualization, diagnosis or optimization. An example is the energy efficiency optimization and fault detection for a conveying system. The purpose of the assistance system is to detect deviations in system behavior from the normal state (for example too much or not enough energy consumption of a drive). In addition, it diagnoses the causes for such anomalies (such as wear or wrong load). Furthermore, automatic optimization of system behavior regarding a given cost function can be accomplished. An example is the calculation of energy efficient movements of the conveying system.

This article deals with aspects of smart assistance systems such as process models, assistance functions and system interfaces. Novel research results with respect to self-diagnosis and self-optimization of industrial processes are presented. In doing so, the focus is set on components



of industrial automation systems such as conveying systems or processing modules. The paper highlights how to integrate approaches to data-driven model learning, fault detection and energy efficiency optimization in an assistance system for such application cases.

The remaining part of this paper is structured as follows. First, the state of the art for industrial assistance system is resumed (section 2). In the main part, assistance functions, such as model learning (section 3.1), self-diagnosis (section 3.2) and self-optimization (section 3.3) are investigated. The interaction between the assistance functions is outlined in section 3.4. In section 4, architecture and integration of smart assistance systems are considered. Furthermore, experimental results for the proposed assistance system are presented (section 5).

2. State of the art

Today's assistance systems can be categorized as follows:

- Assistance systems based on universally usable statistics or data mining tools
- Complete solutions for data acquisition and analysis
- Sector-specific special-purpose solutions.

Examples of data mining tools are WEKA (Waikato Environment for Knowledge Analysis, [2]) and RapidMiner [3]. These tools offer numerous classification possibilities in addition to preprocessing methods and methods for cluster and association analysis. Examples of statistics packages are R [4] and the Matlab Statistics Toolbox. Both of these packages offer many types of statistical process analysis, such as variance analyses (ANOVA), regression methods and statistical time series analyses. Common data acquisition and analysis solutions in an industrial context are SAS Quality Lifecycle Analysis and SPSS from IBM. With their powerful prediction methods, these solutions allow an early detection and user notification of quality defects and arising problems in production processes. They also provide comprehensive means of merging and correcting large data volumes. The described approaches offer a wide range of functions, but are not always easy to operate for inexperienced users, since they require many parameters such as threshold values for automatic warning functions to be configured manually. Also available are special-purpose solutions, tailored to specific fields of application (e.g. Siemens Sipplus CMS for condition monitoring of motors, generators, pumps, etc., CM4810 from B&R for condition monitoring at control system level or PUMon [5] for process monitoring in the chemical industry). Regarding the algorithms used in PUMon, which have been developed at Fraunhofer IOSB as part of the ProDaMi suite, there are already approaches for a generalization to other fields of application, such as monitoring of wind turbines.

In the described assistance systems, a conflict between the bandwidth of application possibilities and the required configuration effort can be observed. The price for a high level of flexibility in possible application areas is usually costly and time-consuming manual engineering. An approach to close this gap is the development of self-learning assistance functions, which, because of their learning capability, can be integrated into existing automation systems without extensive configuration.

3. Assistance functions

In the following section, particular assistance functions and the interaction between these assistance functions are described.

3.1. Data-driven process modelling

Data-driven process modelling allows the flexible use of smart assistance functions in complex automation systems. With this approach, process models are learned from process data and are used for diagnosis (section 3.2) and process optimization (section 3.3). With increasing

complexity of industrial processes and the resulting data, manual generation of such process models is no longer practical and, furthermore, very time-consuming and prone to errors. The development of robust methods for automatic model learning from process data is becoming increasingly important. In these approaches, the interaction between continuous and discrete process variables must be considered. In many applications, distinct system states and modes can be observed, which alter as discrete process variables (such as switching signals) change. Especially the automatic learning of process models for discrete system elements has not, to date, been researched in great detail. Approaches to this task are ALERGIA [6], MDI [7], RTI+ [8], HyBUTLA [9] and OTALA [10]. In general, temporal dependencies must be considered in addition to the static dependencies between process variables. A further challenge consists in automatic learning of the structure of process models in addition to the model parameters.

Furthermore, continuous process behaviour such as energy consumption has to be taken into account. The continuous process behaviour is modeled with, among others, differential and difference equations, equivalent state space models, neural networks [11], self-organizing maps [12] or stochastic models such as Kalman filters (e.g. [13], [14]), particle filters (e.g. [15]) or Hidden Markov Models (e.g. [16]). In order to describe hybrid processes, such models are learned for each system state (the states of the machine shown in Fig. 2) from continuous process data.

The learning of energy consumption in the modes of a hybrid automaton is considered in section 3.1.1. The MoSDA algorithm [17] allows automatic learning of separated component models of hybrid industrial processes that can be used for both anomaly detection and the identification of causes for anomalies (see section 3.1.2). MapReduce technology is applied to learn the proposed process models from large amounts of data (see section 3.1.3).

3.1.1. Learning the energy consumption in the modes of hybrid automata: The learning of energy consumption in the modes of a hybrid system has been considered in [14]. The underlying system behavior is modelled in this approach with an automaton. The automaton consists of discrete system states which determine the system behavior. Transitions between system states are triggered by discrete events, e.g. control signals such as switch on/off or open/close valve. Model learning algorithms such as OTALA [10] can be applied to learn the automaton. The principle of the OTALA algorithm consists in creating a new state for each combination of discrete signals which occurs in the data. Transitions are inserted between states which correspond to subsequent combinations of discrete signals in the input data. The energy consumption in each particular mode of a hybrid automaton can be modelled as a sequence of linear state space models. Switches between the linear state space models account for non-linearities or switches in the energy consumption. The learning of state space models M_{E1}, M_{E2}, \dots consists of two steps (see Fig. 1):

In step 1, splitting points are detected (vertical dotted lines in Fig. 1) where the state space models are assumed to switch. In this work, a time series analysis based on a moving average (MA) method [18] is employed to obtain predictions $\hat{y}(k)$ for the measured energy consumption $y(k)$. Splitting point candidates are detected for $|\hat{y}(k) - y(k)| > y_S$ with the given threshold y_S . In order to reduce the number of splitting points, adjacent splitting point candidates are removed. More precisely, splitting point candidates with a distance less than a given parameter δ_S are merged. In doing so, the first and the last splitting point in a time series of adjacent candidates is preserved in each case.

In step 2, linear state space models are learned for each time segment between two splitting points. The energy consumption in each time segment s is described with a linear state space model M_{Es} (a standard random walk model). The linear models are learned from a sequence of training data $y(k)$ which is observed in the respective time segments. Expectation-maximization (EM) is employed for parameter estimation (see [19]).

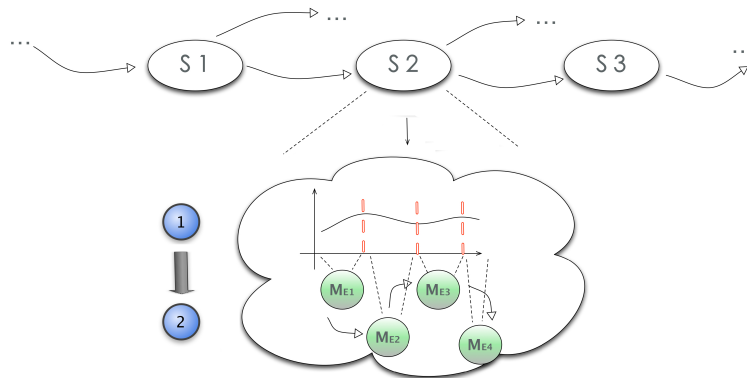


Figure 1. Model-based fault detection

3.1.2. Component-based process models: The algorithm MoSDA [20] exploits a typical situation for cyber-physical production systems: Anomalies often refer to anomalous observations on the system-level, e.g., an anomalous energy consumption of the production system as a whole. MoSDA leverages on the fact that these observations create an over-determined system and MoSDA can therefore break down the overall system model to component models. Fig. 2

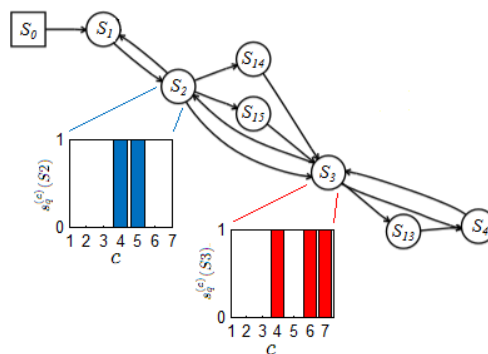


Figure 2. Automaton of the storage system with separation variables $s_{q,c}(S2)$ and $s_{q,c}(S3)$

illustrates the functional principle. The underlying system behavior is modelled with an automaton that can be learned from data as described in section 3.1.1. It is assumed that state $q = S1, S2, \dots$ determines whether the output $y_c(k)$ of component c contributes at time instance k to the overall output signal

$$y(k) = \sum_{c=0}^{n_c-1} s_c(k) y_c(k). \quad (1)$$

This is expressed with a separation variable $s_{q,c}(q)$ for each component c in state q which can be either 0 or 1. In many practical cases, $s_{q,c}(q)$ is known: System mode q determines the active control signals, the dependency between control signals and components is in many cases given. However, if $s_{q,c}(q)$ is not known a-priori, $s_{q,c}(q)$ can be learnt from training data, e.g. by exploitation of correlation as accomplished in [21]. In this paper, we consider components c where the output signal $y_c(k)$ of a component is a function of two input signals $\mathbf{x}_c(k) = (x_{1,c}, x_{2,c})$ at

each time instance k . This approach can be extended straightforwardly to an arbitrary amount of input and output signals. Output $y_c(k)$ is approximated by the quadratic function

$$y_c(k) = \theta_0^c + \theta_1^c x_{1,c}(k) + \theta_2^c x_{2,c}(k) + \theta_{1,1}^c x_{1,c}(k)^2 + \theta_{2,2}^c x_{2,c}(k)^2 + \theta_{1,2}^c x_{1,c}(k)x_{2,c}(k) \quad (2)$$

with coefficients θ_0^c , θ_1^c , θ_2^c , $\theta_{1,1}^c$, $\theta_{2,2}^c$ and $\theta_{1,2}^c$ which can be efficiently computed by solution of a linear over-determined optimization problem (see [17]).

3.1.3. Efficient model learning with MapReduce technology: MapReduce technology allows efficient model learning from large amounts of historical data (see e.g. [22]). The principle is illustrated in Fig. 3. Data is stored in a distributed filesystem in order to process splits of the input data on several host machines. A user-defined map function processes the records of each split and assigns the output to a key. Data is shuffled according to the keys and processed with a user-defined reduce function. Application of MapReduce technology to model learning for the OTALA algorithm described above is straightforward. Pairs of subsequent observations are processed in the map function in order to extract information for each transition (e.g. transition times) which is assigned to unique transition keys. The keys are computed from the discrete signals which change between the two observations. The transitions are shuffled according to the transition keys and are processed in the reduce function (e.g. distributions of the transition times are calculated) in order to build the final automaton. A great deal of model learning algorithms allow for parallelization in a similar fashion.

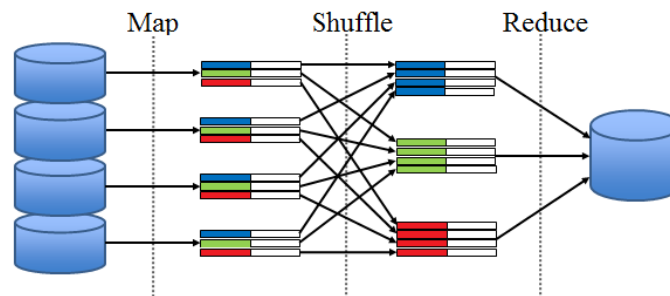


Figure 3. MapReduce technology

3.2. Fault Detection

Reliable fault detection allows costs and risks to be reduced through the early detection of faults and problems in the process and by preventing component failures and, in extreme cases, a production stop of the entire plant.

The principle of model-based fault detection in the operating phase (see Fig. 4) consists of comparing the plants actual behavior with its behavior as predicted from the process models. The process models can be learned from process data collected about the plant and its components in normal, fault-free operation, as described in section 3.1. This allows the detection of, on one hand, discrete events that are incompatible with the learned process models [23], and, on the other, deviations of the continuous behavior from the predicted process behavior.

For the latter, stochastic models can be used (see section 3.1.1). Measurements $y(k)$ are predicted in this approach according to a stochastic process model with input $u(k)$. A common approach to fault detection is the evaluation of the residuals $r(k) = y(k) - \hat{y}(k)$ between measurements $y(k)$ and predictions $\hat{y}(k)$. Methods such as log-likelihood ratios, generalized log-likelihood ratios, sequential probability ratio tests, or cumulative sum test statistics are used

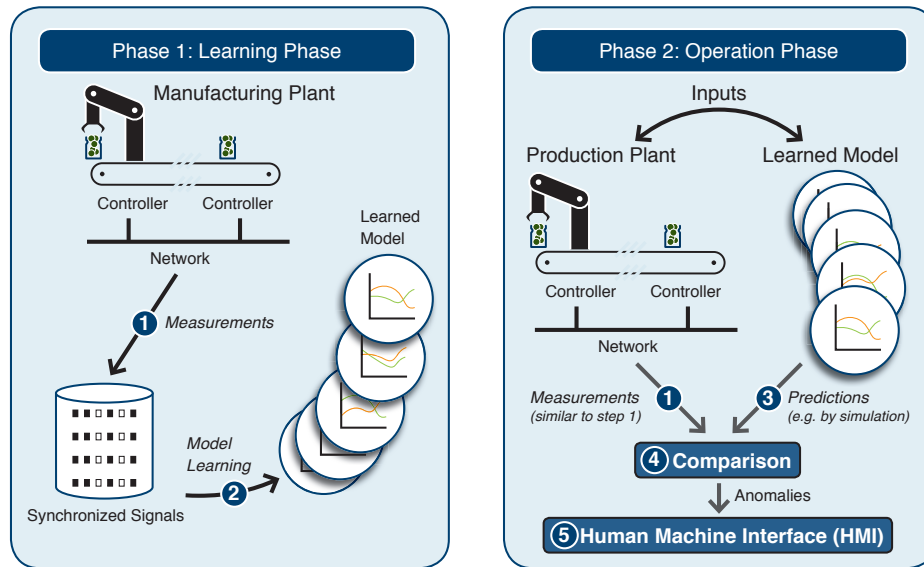


Figure 4. Model-based fault detection

in literature to detect changes in the residuums (see [24]) for an overview). A related approach is the evaluation of $y(k)$ with respect to the conditional probability density $p(y(k)|y(0) \dots y(k-1))$ (see e.g. [14], [17]). Observations are assumed to be improbable once measurement $y(k)$ at discrete time k drops out of a given confidence interval. This is the case for [14]

$$F(y(k)) < \frac{1}{2} - \frac{C}{2} \text{ or } F(y(k)) > \frac{1}{2} + \frac{C}{2} \quad (3)$$

$$\text{where } F(y(k)) = \int_{-\infty}^{y(k)} p(\tilde{y}(k)|y(0) \dots y(k-1)) d\tilde{y}(k)$$

denotes the distribution function of $p(y(k)|y(0) \dots y(k-1))$ and C the width of the confidence interval ($C \in [0, 1]$). Measurements beyond a confidence interval of width $C = C_1$ are detected as potential faults. In this case, anomaly $f_1(k)$ is reported for time instance k . Furthermore, anomalies with a duration of more than one time instance are taken into account as suggested in [8]. Measurements beyond a confidence interval of width $C = C_2$ are considered as candidates for such anomalies. If at least a given portion p_d of the last n_d measurements are beyond the confidence interval of C_2 , anomaly $f_2(k)$ is reported. Evaluation of (3) requires an adequate model for the process behaviour that essentially determines in which way the probability density $p(y(k)|y(0) \dots y(k-1))$ is obtained. A general approach is the application of discrete time nonlinear state space models (see e.g. [14], [15], [25]) that are specific for particular states or components of the hybrid automaton:

$$\begin{aligned} y(k) &= h(\mathbf{x}(k), u(k)) + v(k) \\ \mathbf{x}(k) &= f(\mathbf{x}(k-1), u(k)) + w(k) \end{aligned} \quad (4)$$

where f and h are the state and measurement dynamic functions, respectively. The two equations consider uncertainties in the form of additive terms for measurement noise $v(k)$ and state transition noise $w(k)$. The continuous system behaviour is modelled with a time-variant state vector $\mathbf{x}(k)$. The random walk model introduced in section 3.1.1 and the system

model introduced in (1) are special cases of this general state space model. The component models which are learnt by the MoSDA algorithm (see section 3.1) enable beside fault detection the location of faults by relating the faults to particular components (see section 3.1.2). The application of component models for model-based fault detection is evaluated in section 5.1.

3.3. Energy Efficiency Optimization

Another field of application for smart assistance systems is the self-optimization of industrial processes. The goal here is to analyze and improve the performance and efficiency of a plant to optimize its operation. In particular the energy efficiency optimization of industrial automation systems is becoming increasingly relevant, driven by rising energy prices and political targets [1]. In existing plants, interventions in the process usually consist of manual stopping, energy-efficient time planning of the production steps in the MES system, or control actions at 15-minute intervals [26]. To date, little research has been carried out regarding active methods to implement real-time optimization at second or millisecond intervals, even though, due to their process dynamics, many applications require process parameters to be rapidly adapted to changing operating conditions. First approaches have been developed for stopping the plant based on energy data [27], for reducing reactive power and for controlling energy in Smart Grids [28]. Another approach is based on the use of predictive process models that can be learned with the methods described in section 3.1. The process models allow the future process behavior to be predicted for various parameter combinations in order to determine the optimum process parameters. Optimization is accomplished with respect to the energy and load management of electrical drive systems, which are the most voracious energy consumers, especially in transportation and logistics applications.

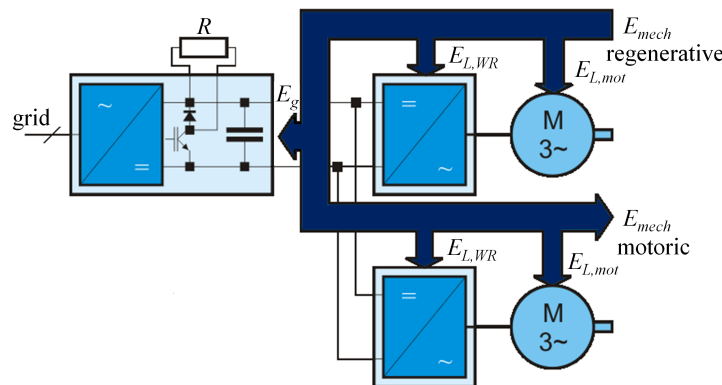


Figure 5. Energy flow between two drives [29]

The basic optimization problem is illustrated in Fig. 5 for two drives. The main optimization aim in this application is to minimize the overall energy consumption $y_0(k)$ and $y_1(k)$ of the two converters ($E_{L,WR}$) and motors ($E_{L,mot}$) for time instances $k = 0 \dots n - 1$. In addition, the energy fed back $E_g(k)$ into the internal DC link or the grid (i.e. the difference between regenerative and motive power) is to be minimized. To this end, energy consumption and energy fed back are weighted with factor λ according to the primary usage of power fed back (consumption at braking resistances, regenerative feedback to the mains, and temporary storage in capacitors). Altogether, optimal driving speeds \mathbf{v} of the conveying system can be obtained as solution of the following optimization problem [20]:

$$\mathbf{v} = \operatorname{argmin} \left\{ \sum_{k=0}^{n-1} \sum_{c=0}^{n_c-1} y_c(k) + \lambda \sum_{k=0}^{n-1} E_g(k) \right\} \quad (5)$$

where $y_c(k)$, $c = 0 \dots n_c - 1$, $n_c = 2$, denote the output variables of the component models of the particular drives (see (2)) which are functions of velocities $x_{1,c}(k)$ and accelerations $x_{2,c}(k)$ for the given application case. Optimization constraints regarding energy flows and the sequences of movements (starting position, end position, speed limits, and positioning time) are detailed in [20]. The overall optimization problem is a mixed integer quadratically constrained problem (MIQP) that can be efficiently solved with standard methods. Evaluation results for energy-efficiency optimization of movements in conveying systems by solution of the exposed optimization problem are given in section 5.2.

3.4. Interaction of the assistance functions

In this section, the interaction of the introduced assistance functions is outlined for the example of the drive system shown in Fig. 5. The overall system model (see (1)) with output variable

$$y(k) = \sum_{c=0}^{n_c-1} s_c(k) y_c(k) \quad (6)$$

is composed of n_c ($n_c = 2$ for the example) component models with output variables $y_c(k)$ according to (2). Switching variables $s_c(k)$ have been introduced in section 3.1.2. The system model is obtained from training data by application of the MoSDA algorithm. Energy efficient movements of the drives are calculated by solution of optimization problem (5) with respect to optimal velocities \mathbf{v} and constraints detailed in [20]. Furthermore, anomalous energy consumption of the drive system is detected by evaluation of

$$F(y(k)) = \int_{-\infty}^{y(k)} p(\tilde{y}(k)|y(0) \dots y(k-1)) d\tilde{y}(k) \quad (7)$$

in (3) for probability density

$$p(y(k)|y(0) \dots y(k-1)) = \mathcal{N}\left(y(k); \sum_{c=0}^{n_c-1} s_c(k) y_c(k), \sigma\right) \quad (8)$$

where standard deviation σ of the prediction error is obtained from training data.

4. Architecture and integration

The flexible use of an assistance system in different industrial environments requires the simple integration of the system in various automation systems. Furthermore, an uncomplicated connection of application-specific user interfaces is required. Besides, aspects such as configurability, modularization capability and scalability have to be taken into account. In proKNOWS - an assistance system developed at Fraunhofer IOSB-INA - these requirements are met through the software architecture shown in Fig. 6.

A unified system interface based on OPC UA allows an abstraction of the assistance functions from heterogeneous networks and data acquisition methods. This, in turn, allows the assistance system to be connected to a wide range of plants and automation systems. The assistance system interfaces with the particular automation systems through OPC UA, with OPC DA servers, or particular instruments and controls or data loggers being integrated through corresponding interfaces. Interfaces for CSV and SQL are also implemented.

The assistance functions implemented in the modeling engine can be run on a central server in real time. In the modeling engine, the data-driven process modeling methods described in the previous sections (HyButla, OTALA, MoSDa, switching state-space models, and electrical drive models) are implemented for fault detection (see e.g. [23] and [14]), fault localisation [17])

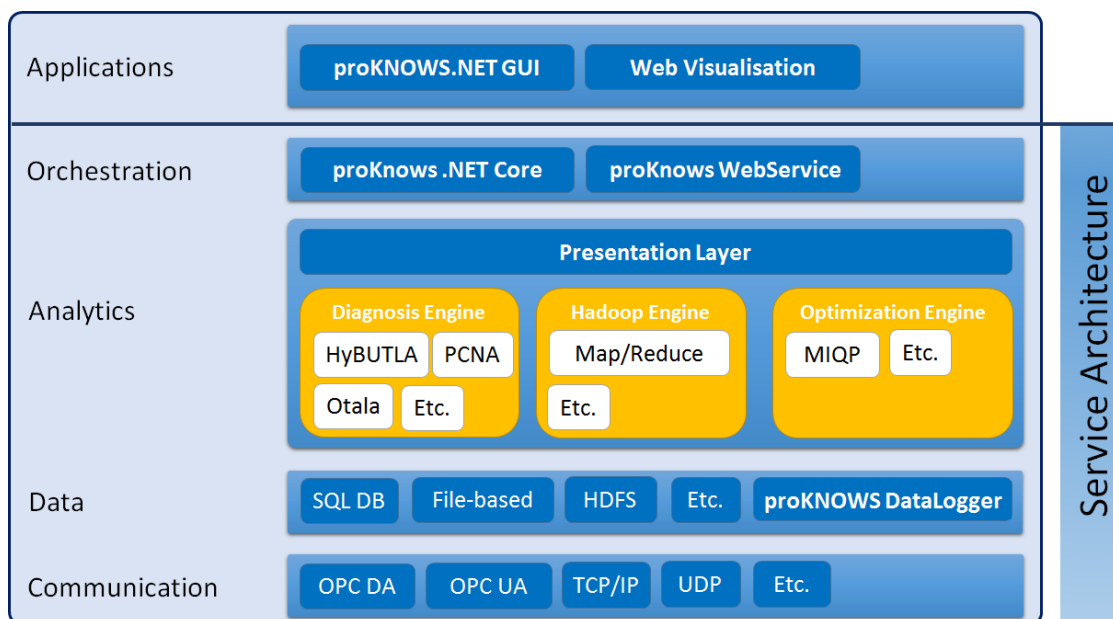


Figure 6. Architecture of proKNOWS

and for process visualization. The modeling engine uses a simple structure model of the plant, which can be configured manually in XML with little effort. The plant structure is split into separate modules. The behavior of these modules is described with process models, on which the represented assistance functions are executed. The process models can be learned automatically from process data, as described in section 3.1. These approaches allow the assistance system to be flexibly adapted to various processes.

The presentation layer converts representations of the process models and results of analysis into a simple exchange format which essentially consists of vector graphics and status information. This provides a convenient way of displaying the relevant information on various user interfaces. In addition to the control panel, these could include mobile apps for particular plant modules, which can be selected specifically for application cases such as commissioning or troubleshooting. A Hadoop engine allows for efficient model learning from large historical databases and provides a framework for efficient analysis of big data. The proKNOWS tool box also contains an optimization engine, in which the methods described in section 3.3 are implemented.

5. Evaluation Results

The introduced assistance functions have been evaluated in several research projects on demonstrators of the Lemgo Smart Factory (LSF), such as a bulk goods processing module (Fig. 7 a) and a conveying system (Fig. 7 b).

The structures of the learned process models for each of the demonstrators differ in terms of system states, transitions and characteristics of the continuous system variables. Typical faults in the represented application cases are errors in the discrete event system, for example delayed switching processes or signal failures, and deviations of continuous signals from their expected trend, such as unexpectedly high or low power consumption by drives or heater fans. Evaluation results for the assistance functions outlined in section 3.4 are presented in the subsequent sections.



Figure 7. a) Module for bulk goods processing b) Conveying system

5.1. Fault Detection

The separated component models, which have been learned in section 3.1.2, have been employed for both fault detection and diagnosis. The proposed methods have been evaluated in the research conveying system depicted in in Fig. 7 b). The system comprises five conveyor belts and seven drives, which are used to move objects between four storage positions on two levels (five drives for horizontal movements and two drives for vertical movements). A power meter continuously captures the overall power consumption. Further sensor information is collected via the programmable logic controller (PLC) of the storage system (Lenze Control 3200 C). For evaluation, process cycles have been recorded, in which an object has been moved between two positions of the conveying system. Faults have been artificially added to 10 process cycles of the conveying system. The created faults differ with respect to position (start time), duration (temporal length) and offset (signal jump or signal drop). The offset is assumed to be constant for each fault.

Fault detection has been accomplished with the method outlined in section 3.4. Results for 10 different durations are averaged with respect to the 10 test cycles, five different offsets and 20 randomly selected positions for each test cycle. The F-measure, i.e. the harmonic mean of sensitivity and precision, was calculated as performance metric.

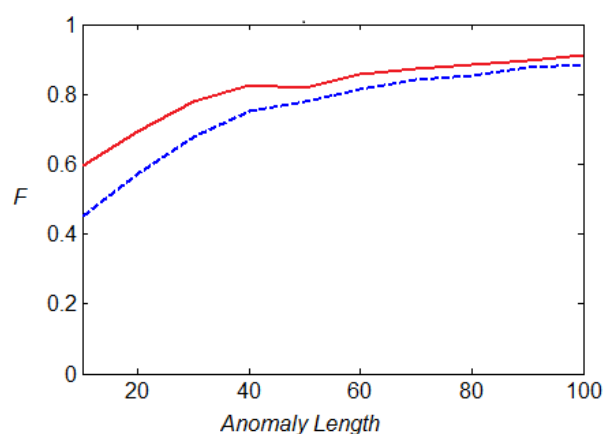


Figure 8. Fault detections results for different fault lengths

Fig. 8 shows the fault detection results for the novel separation approach (straight line). Further, the results for the baseline method without separation (see [11]) are depicted (dashed

line). In the test case, significantly improved F-measures are achieved with the separation method. In total, the F-measure improved from 0.75 to 0.81. The introduced component models further allow for diagnosis by relating faults to particular components.

5.2. Optimization

Initial experimental results for the optimization approach introduced in section 3.3, which is summarized in section 3.4, are shown in Fig. 9. The graphs show the speed $v(t)$ and position $x(t)$ for a belt conveyor with a rated power of $470W$ as a function of time t . The movement sequence of the belt conveyor was optimized for given workpiece start and end positions and for given constraints regarding speeds $v(t)$. As can be seen in Fig. 9, the optimization results

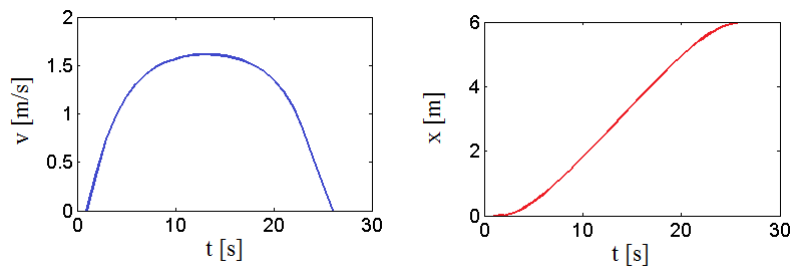


Figure 9. Optimization results

in a movement profile in which acceleration of the conveyor decreases with velocity. For the shown application case, energy consumption was reduced from $5988Ws$ to $5557Ws$ compared to a movement profile with maximum acceleration and deceleration, i.e. energy savings of 7.2% were obtained. Joint optimization of two drives results in a motion profile in which the first drive is decelerated exactly during the second drives acceleration phase. In this application case, the complete regenerative energy of the first drive could be recovered as input for the second drive. Hence, the automatic coordination of motor speeds leads to cancellation of undesired feedback of energy.

6. Conclusion

Because of the increasing complexity of industrial automation systems, process control and monitoring is becoming increasingly challenging. It is therefore desirable to support users in these tasks with smart assistance systems that can take over some of the key tasks. To date there are many partial, but not yet any integrated solutions available. To close this gap, various aspects of smart assistance systems have been integrated, including the system interface and the learning of process models as basis for various assistance functions, such as process monitoring and optimization. The described approaches have proven to make the complexity in automation better manageable, thereby sustainably increasing the productivity of industrial processes.

References

- [1] MANUFACTURE-EU, “Factories of the Future PPP Strategic Multi-annual Roadmap,” 2010.
- [2] I. Witten, E. Frank, and M. Hall, *Data Mining: Practical Machine Learning Tools and Techniques*. Morgan Kaufmann, 2011.
- [3] M. Hofmann and R. Klinkenberg, Eds., *RapidMiner: Data Mining Use Case and Business Analytics Applications*. Chapman & Hall/CRC Data Mining and Knowledge Discovery Series, 2013.
- [4] R. Ihaka and R. Gentleman, “R: A language for data analysis and graphics,” *Journal of Computational and Graphical Statistics*, vol. 5, no. 3, pp. 299–314, 1996.

- [5] C. Frey, "Diagnosis and monitoring of complex industrial processes based on self-organizing maps and watershed transformations," in *Computational Intelligence for Measurement Systems and Applications, 2008. CIMSIA 2008. 2008 IEEE International Conference on*, July 2008, pp. 87–92.
- [6] R. C. Carrasco and J. Oncina, "Learning deterministic regular grammars from stochastic samples in polynomial time," in *RAIRO (Theoretical Informatics and Applications)*, vol. 33, 1999, pp. 1–20.
- [7] F. Thollard, P. Dupont, and C. de la Higuera, "Probabilistic DFA inference using Kullback-Leibler divergence and minimality," in *Proc. of the 17th International Conf. on Machine Learning*. Morgan Kaufmann, 2000, pp. 975–982.
- [8] S. Verwer, "Efficient identification of timed automata: Theory and practice," Ph.D. dissertation, Delft University of Technology, 2010.
- [9] O. Niggemann, B. Stein, A. Vodenčarević, A. Maier, and H. Kleine Buning, "Learning behavior models for hybrid timed systems," in *Twenty-Sixth Conference on Artificial Intelligence (AAAI-12)*, Jul 2012.
- [10] A. Maier, "Online passive learning of timed automata for cyber-physical production systems," in *The 12th IEEE International Conference on Industrial Informatics (INDIN 2014)*. Porto Alegre, Brazil, Jul 2014.
- [11] A. Vodenčarević, H. Kleine Büning, O. Niggemann, and A. Maier, "Using behavior models for anomaly detection in hybrid systems," in *Information, Communication and Automation Technologies (ICAT), 2011 XXIII International Symposium on*, 2011, pp. 1–8.
- [12] C. Frey and T. Ross, "Anwendung datengetriebener Modelle zur Zustandsanalyse bei verfahrenstechnischen Anlageneinheiten: Theoretische Grundlagen und Beispiele aus der Industriellen Umsetzung," *Chemie Ingenieur Technik*, vol. 84, no. 8, p. 1378, 2012.
- [13] S. Narasimhan and G. Biswas, "Model-based diagnosis of hybrid systems," in *Proceedings of the 18th international joint conference on Artificial intelligence*. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 2003, pp. 376–381.
- [14] S. Windmann, S. Jiao, O. Niggemann, and H. Borchertding, "A Stochastic Method for the Detection of Anomalous Energy Consumption in Hybrid Industrial Systems," in *INDIN*, 2013.
- [15] M. Wang and R. Dearden, "Detecting and Learning Unknown Fault States in Hybrid Diagnosis," in *Proceedings of the 20th International Workshop on Principles of Diagnosis, DX09*, Stockholm, Sweden, 2009, pp. 19–26.
- [16] M. Jager, C. Knoll, and F. Hamprecht, "Weakly supervised learning of a classifier for unusual event detection," *Image Processing, IEEE Transactions on*, vol. 17, pp. 1700–1708, 2008.
- [17] S. Windmann and O. Niggemann, "Automatic model separation and application to diagnosis in industrial automation systems," in *IEEE International Conference on Industrial Technology (ICIT 2015)*, 2015.
- [18] T. Mingyu and C. Jin, "The time series of the double moving average forecast," *China Academic Journal Electronic House*, vol. 1, pp. 33–37, 2010.
- [19] Z. Ghahramani and G. E. Hinton, "Parameter estimation for linear dynamical systems," Univ. Toronto, Dept. Comput. Sci., Tech. Rep., 1996.
- [20] S. Windmann, O. Niggemann, and H. Stichweh, "Energy efficiency optimization by automatic coordination of motor speeds in conveying systems," in *IEEE International Conference on Industrial Technology (ICIT 2015)*, 2015.
- [21] O. Niggemann, S. Windmann, S. Volgmann, A. Bunte, and B. Stein, "Using learned models for the root cause analysis of cyber-physical production systems," in *DX*, 2014.
- [22] T. White, *Hadoop: The Definitive Guide*, 2nd ed., M. Loukides, Ed. O'Reilly Media, Inc., 2011.
- [23] S. Faltinski, H. Flatt, F. Pethig, B. Kroll, A. Vodenčarević, A. Maier, and O. Niggemann, "Detecting anomalous energy consumptions in distributed manufacturing systems," in *Industrial Informatics (INDIN), 2012 9th IEEE International Conference on*, 2012, pp. 358–363.
- [24] W. Liu and I. Hwang, "Robust estimation and fault detection and isolation algorithms for stochastic linear hybrid systems with unknown fault input," *Control Theory Applications, IET*, vol. 5, no. 12, pp. 1353–1368, August 2011.
- [25] L. Christiansen, A. Fay, B. Opgenoorth, and J. Neidig, "Improved diagnosis by combining structural and process knowledge," in *Emerging Technologies Factory Automation (ETFA), 2011 IEEE 16th Conference on*, 2011, pp. 1–8.
- [26] P. Palensky and D. Dietrich, "Demand Side Management: Demand Response, Intelligent Energy Systems, and Smart Loads," in *IEEE Transactions on Industrial Informatics*, vol. 7, 2011.
- [27] A. Cannata, S. Karnoukos, and M. Taisch, "Energy efficient driven process analysis and optimization in discrete manufacturing," in *IECON*. IEEE, 2009, pp. 4449–4454.
- [28] G. Reed, "Sample Survey of Smart Grid Approaches and Technology Gap Analysis," in *Innovative Smart Grid Technologies Conference Europe*, 2010, pp. 1–10.
- [29] Lenze SE, "Internal Report," 2014.