A PLC Variable Identification Method by Manual Declaration of Time-Stamped Events

Aurélien Cadiou, Sébastien Henry, Vincent Cheutet, Carole Eyssautier

A PLC Variable Identification Method by Manual Declaration of Time-stamped Events

Aurélien Cadiou, Sébastien Henry, Vincent Cheutet and Carole Eyssautier

Abstract The fourth industrial era, the digitisation of production and the emergence of digital twin involve the hyper-connection of all industrial equipment and especially Automated Production Systems (APSs). Nevertheless, the acquisition of data within these machines remains complicated. APS is an expensive invest, composed with many heterogeneous equipments, they are made for having a long lifespan. Hence, industries are composed with old equipment and it is not possible to wait new industry 4.0-compatible APSs to follow the digital evolution. A solution is to retrieve data from the APSs through its Programmable Logic Controller (PLC), with industrial network, like the Modbus. The PLC knows the APS state with sensors and pilots the APS with actuators, all this information is stocked into variables in memory of PLC, referenced with addresses. Due to several factors, such as heterogeneous data, lack of PLC program documentation or no automation specialist, the data collection is a complicated and time-consuming task. The difficulties are to link desired information and the memory address where it is stored. The aim of this contribution is to propose a method for identifying address, through photos of APS, an historical of memories values and an a posteriori declaration of viewed events with photos. This method must be suitable for non-specialists.

Aurélien Cadiou
Université de Lyon, INSA Lyon, DISP (EA4570), Villeurbanne, France and Infologic R&D, Bourg-Lès-Valence, France, e-mail: aurelien.cadiou@insa-lyon.fr

Sébastien Henry
Université de Lyon, UCBL, DISP (EA4570), Villeurbanne, France e-mail: sebastien.henry@univ-lyon1.fr

Vincent Cheutet
Université de Lyon, INSA Lyon, DISP (EA4570), Villeurbanne, France e-mail: vincent.cheutet@insa-lyon.fr

Carole Eyssautier
Infologic R&D, Bourg-Lès-Valence, France e-mail: cey-externe@infologic.fr

1 Introduction

The Industry 4.0 is a new industrial revolution, with the aim to improve industrial competitiveness, efficiency and flexibility. This important revolution has several aspects, such as smart factory, software/systems engineering, human-machine interaction, embedded systems, interconnection between equipment, communication technologies, big data, artificial intelligence or Digital Twin (DT) [13, 15]. All of these require a large amount of data, sometimes collected in real-time, and data exchanges to drive production and achieve objectives [11].

The objective of industrial plants is to produce goods and this is done with Automated Production System (APS). An APS could be composed with many equipment, heterogeneous in role, functionalities, age, machine builder, Programmable Logic Controller (PLC), etc. Production systems are often composed by special machines, they are expensive and made to last, with the necessity to amortise them. They can be up to 10 or 15 years old. It is not possible for industrial companies to change all APSs for being "industry 4.0" and it is not possible to wait the depreciation of each equipment to change them individually. The smart-industry transformation would take then many and many years.

The APS are piloted with the PLCs, sensors are connected to it inputs and actuators are connected to it outputs. The PLCs pilotes its outputs depending on its inputs and its program, thus it changes the state of the APS.PLC are programmed with proprietary software and code downloaded into PLC often have not memories name nor comments. Furthermore, due to the long lifespan of APSs, the PLC programs are not well documented, the program evolves and documentation is not updated, or, sometimes, documentation is lost.

These difficulties can be overcome by adding Industrial Internet of Things (IIoT) sensors on APSs, in order to retrieve state. Even if this solution gains interest these last years, it is more expensive and requires an IIoT infrastructure, mechanical interventions on APSs for installation and long-term maintenance, involving production shutdowns.

Moreover, the installation of IIoT sensors is not possible in all industries and the PLC still the best supplier of APS state. Retrieving data from PLCs is applicable for industries such food-processing, where health standards are mandatory, production pace is high involving production stops are complex to plan and equipment is ageing and unique with little knowledge capitalised with time.

The presented difficulties are not always there. It is possible to classify three cases for retrieving APS state. The first is the most simple, it is when the PLC program is totally known, and in this case, the OPC-UA server could facilitate the collection of data. The OPC-UA is a machine-to-machine protocol and it specifies the structure of semantic information models and could be used by a constructor for open data exchange (like PackML [1] or AutomationML [6]) with their equipment [5].

The two other cases are appropriate when too little information about the PLC program are known, in such a way PLC is considered like a black-box. The PLC data collection is here complex. Several factors come into play, such as heterogeneity of data, multitude of industrial protocols, no knowledge of the PLC program, lack of

documentation or up-to-date documentation [18, 1, 7], low degree of automation in enterprises [17] as well as the seniority of many industrial equipment and their PLCs.

In this context, this contribution proposes a solution which purpose is to facilitate the data collection in PLC, in particular the identification of memory addresses of variables. Compared to the easiest possibility of adding IIoT sensors to obtain the desired information, the proposed solution aims at offering a simple and quick protocol, performed in two successive and punctual steps. Once these steps have been completed, variables are identified without the extra use of artifice. It consists of reverse-engineering of PLC programs. This contribution is focused on PLC with Modbus communication network.

This paper is structured as follows. After a brief analysis of the scientific literature on variable identification, section 3 describes the general scenario about usage and precise the entire procedure. Finally, an industrial case study is presented in section 5 with results detailed, analysed and discussed.

2 Related works

Reverse engineering of PLC programs is often necessary. This is mainly due to a lack of documentation: never created initially, lost or not updated once the modifications have been made. Information about PLC is used to make modifications to meet new production demands, re-implementations when new controller hardware is necessary [19], establish connections for collecting data or Machine To Machine (M2M) connections [14].

In the literature, reverse engineering could be classified in two categories, depending on knowledge about PLC.

For the first category, the PLC program is available, with or without a name or comments. In this situation, a static analysis of the program is performed. The code is converted and parsed to find addresses, labels, routines, contacts, coils, statements, etc. The result of this treatment is converted into XML files [19] or used for automatic generation of OPC server tags [9]. It is only possible when the PLC program is written in structured text or instruction list [18, 3].

For the second category, the PLC program is not known or readable, then the PLC is considered as a black-box. Some research investigating the reverse engineering of PLC programs with the observation of inputs and outputs [4, 2]. The main goal is to obtain a model of the PLC program, without any consideration of inputs and outputs names. The internals signals of PLC are not used with these methods [16].

The event sequence similarity can be helpful. Works about the representation of temporal information propose methods to compare and find similarities between several event sequences [10, 8]. They need a lot of data in order to compute and they use, for instance, data mining approaches [12].

The latest seems pertinent in out context, but no current implementation is applicable and new perspectives can be explored.

3 General scenario

The outline of our method is based on recording the physical state of APS and the digital state of the machine, reflect on physical state, in order to compare and link them. When links are established, the user has the memories linked to different states of APS, such as "machine in fault", "conveyor running", "part produced", "temperature of furnace", etc.

The physical state is saved through photographs, and the digital state of APS, via data collection from PLC, both are saved at the same time. The data collection is possible using Modbus protocol, having a 4 normalised addresses ranges :

- coil numbers start with 0 and span from 00001 to 09999,
- discrete input numbers start with 1 and span from 10001 to 19999,
- input register numbers start with 3 and span from 30001 to 39999,
- holding register numbers start with 4 and span from 40001 to 49999.

The proposed method consists into finding the right, or the best, memory address for a state. The method will compare the physical evolution of desired state with evolution of memories for finding the most convenient. The user describes the physical evolution by using photographs. A program was developed for taking photographs, retrieve values of memories and execute a scoring algorithm for determine the most relevant memories. Based of this classification, the final user can retrieve the most consistent addresses.

3.1 Retrieve state of the APS

A camera takes photographs of all the APS or a part of the APS. If the user wants to retrieve address of memory reflecting the state of a cylinder (extended, retracted), the camera could be focused on this, and particularly on sensors if there are any. In fact, sensors are the reflect of state of the cylinder, and they are connected to PLC inputs. For global state of the APS, it is possible to focus the camera on Human-Machine Interface (HMI) or an indicator light.

Records are launched at regular interval and they are synchronised with each other. Records frequency must be defined according to the occurrence frequency of events to retrieve. The Shannon theorem could be used for defining this parameter: the frequency of recording must be twice more than the frequency of events.

The user has to analyse photos to add an event when an evolution of the desired state is observed. Thus, all events are saved and they describe the evolution of APS. With this work doing by the user, a list of events was created and will be used as a reference of the evolution of the state of APS.

3.2 Identify memory addresses

The scoring algorithm compares all the value changes of all memories, called signals, with the declared events, in order to suggest most concordant of them. Each declared event must involve a value variation, because an event describes a modification of the physical state, and this modification is most probably due to the PLC order and/or detected by a sensor connected to the PLC. In order to avoid a time lag between the physical occurrence and manual declaration, a temporal window is created on both sides of each event. The width of this window is an input parameter for the algorithm.

There is no limitation regarding the type of states identified, the algorithm compares only value modification of memories with declared events. Nevertheless, it is possible to classify signals in two categories and events in two other categories. This classification provides a better interpretation of signal scores obtained with the algorithm. The first category concerns the occurrence frequency: the desired state could be periodic or aperiodic. A periodic state could assimilate from machine rate-depending state: parts counting, actuator sensor, etc. An aperiodic state could be assimilated to an intermittent state or a transitory state, it could be a fault state, an operating mode state, etc. The second classification concerns the event type and this observation: the event could be related to an action of PLC, as an actuator movement, or indirectly related to an action of PLC, like a new produced part, for these the user does not know exactly when the PLC gets or produces this piece of information.

The algorithm assigns a score to each variable, with for data the signals recorded, declared events and a parameter that is the temporal window width around each event (figure 1). Depending of the classification of event, the wide of windows could be defined. When the event is related to an action of PLC, this wide could be small. In the other case, a higher value must be used.

The algorithm is based on generating data during recording: signals, pictures of the production equipment and declared events.

The algorithm starts by demultiplexing 16 bit integer signals to generate 16 new binary signals. The integer signal and its bits will be analysed by the algorithm. It is common that binary variables are regrouped into one or more integer variables, for data exchange with Supervisory Control And Data Acquisition (SCADA), for example.

Then, the algorithm defines temporal windows, linked to the events. Each temporal window is defined for an event with a wide parameter, defined by τ, while each window has a width equal to 2τ. Two or more overlapped windows will be combined in one single window, while keeping the information that it represents a serie of events.

For each window, the algorithm computes a score. It calculates the ratio, reduces it to zero, between the number of events included in the window and the number of changes in signal.

For each signal, the average of the obtained scores is calculated in order to have an overall score $scoreIn$. The highest score is equal to 0.

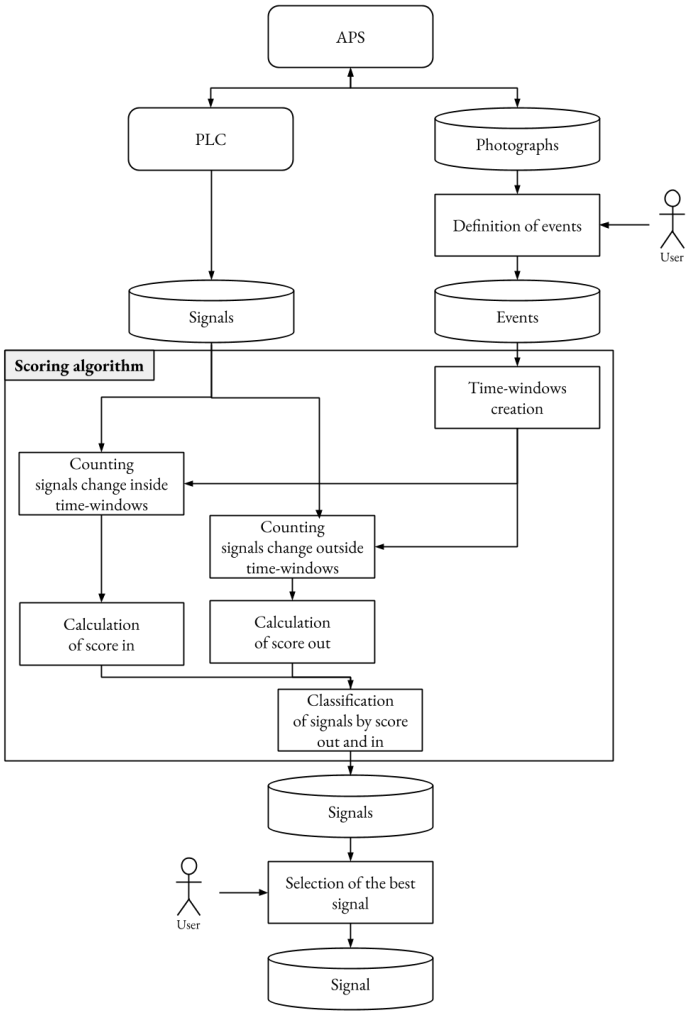

Fig. 1 Schema of signal scoring treatment

The algorithm then calculates the number of changes in signal value in out-of-windows areas, this value being weighted by the width of these areas. This value must be as low as possible to correspond to the desired signal. Indeed, this signal must not change when no events are declared. Each out-of-windows area is defined between each window. The highest possible score is then 0, it corresponds to a signal that does not change the value in out-of-windows areas.

Scores of the desired signals could be altered by several elements, in such a way that the signal has not a $scoreIn$ equal to 0 and a $scoreOut$ equal to 0. The $scoreIn$ and the $scoreOut$ could not be equal to 0, if there is an important offset between

the signal and the event, in case of event unrelated to a PLC action or in the case of sampling frequency being too low, implying that some physical events were no recorded by the camera. The $scoreIn$ could evolve if the user omits to report an event or reports more events than the system evolution.

In the case of an aperiodic signal and events unrelated to an action of the PLC, it could appear as an offset between event declarations and signal evolving, then it will impact $scoreIn$ and $scoreOut$.

In order to sort out and give to the user a selection of best signals, the algorithm removes every signal that did not evolve during the period of recording. Then, it selects, according to the Pareto principle, the most relevant signals depending on their $scoreIn$ and order them by $scoreOut$, from the best to the worst. With this ranking, the exploitation and analysis of results is simplified for the user, by focusing only on relevant signals, ordered by their score.

4 Case study

The aim of this case study is to validate the presented method with a specific case in terms of effectiveness (i.e. to validate the discriminating characterisation among address memories). It was realised on an APS filling bottles. This APS is supplied with empty bottles, caps and liquid and, it produces caped filled bottles at the outlet (figure 2). A degraded mode allows the production without bottle capping.

The goal is to retrieve two types of information: the number of parts produced and the state of conveyor. The first information is described with a periodic signal and events are unrelated to an action of PLC. The second information is described with an aperiodic signal and events are related to an action of the PLC.

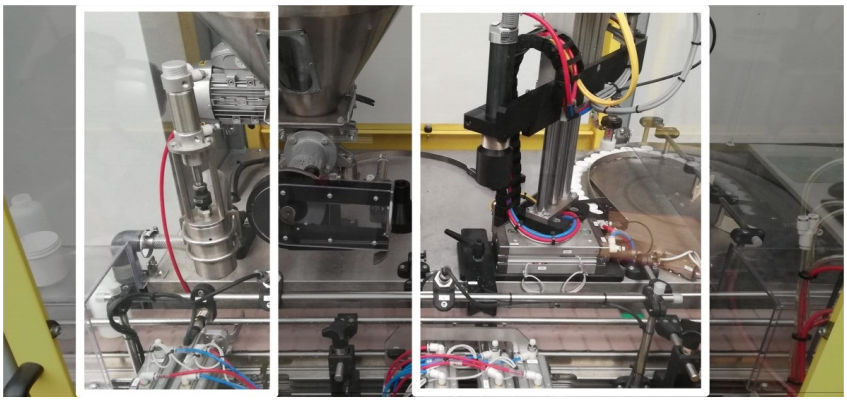

Fig. 2 Detailed view of the concerned two stations of the APS. At the left, the filling station. At the right, the capping station with cap pick-and-place and screwing.

The APS is controlled by a Schneider Electric M340 PLC. Inputs and outputs are connected to an AS-i industrial network. Actuators are electric motors (dosing pump, conveyor, rotary table) and pneumatic actuators (linear cylinders, rotary cylinders and Venturi system). The PLC is connected to the OPC server with Modbus TCP-IP protocol.

The data collection is performed by the KEPServerEX 6.5 OPC server. The history of variable values is recorded with DataLogger plugin. Data are saved in MySQL database. The values of variables are saved in a table of a database with the following format: `Unique ID;Date and time;Variable name;Value`. Each record creates 400 lines in this table. The time is stored with a precision of ten milliseconds.

4.1 Protocol

In this case study, the OPC server and the created program are installed on the same computer. The PLC is connected to the same network as the computer. The first 100 boolean variables (`%M`) and the first 300 variables (`%MW`) of PLC are added in the OPC server. The APS production rate is about 400 parts per hour, the duration between each record is set to 1 second.

The camera is positioned in order to observe the two stations and the conveyor. A first sequence of 3 minutes of recording is made in the normal production mode. This sequence is used to determine the counting parts variable. A second sequence of 3 minutes of recordings is made with normal production times and conveyor start-up times in manual mode. This last sequence is used to determine the conveyor state variable.

The searched events are then added through the interface of the tool (figure 3). For the part counting, an event is declared as soon as a bottle leave station 2 and before they disappear from the picture frame. This reference is arbitrary, it depends on the user's selection because it is not possible to know where and when the PLC counts the bottle. For the conveyor, an event is declared as soon as the conveyor starts or stops.

The state of the conveyor is an aperiodic signal and events are directly related to an action of the PLC, the selected value of τ is low.

4.2 Results and Discussions

4.2.1 Counting parts

The type of signal corresponding to counting parts is periodic and events are unrelated to a PLC action, the selected value of τ is high. The τ is set to 10. The algorithm returns 88 variables that evolve during the recording and it selects 18 variables (figure 4). Obtained results show us three variables with an identical $scoreIn$ and

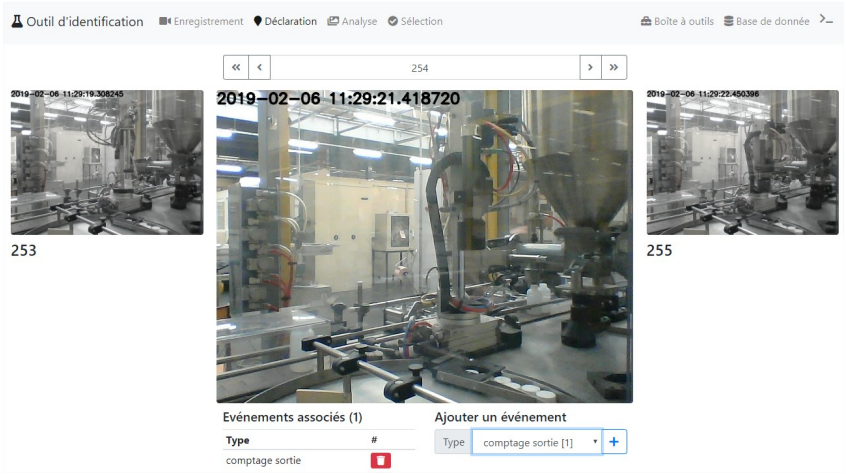

Fig. 3 The HMI for the identification of variables. The figure shows the page for adding events depending on photographs. The developed program assists the user from the recording to the identification.

scoreOut. The temporal analysis, with the graph, confirms that these three variables have identical values. This record does not allow to discriminate them. The next variables have a relatively distant score and can be eliminated by human. Temporal analysis confirms the found correspondence of variables (figure 5).

Part count can also be retrieved by the status of APS sensors. However, the declaration of events does not allow these variables to be collected. Indeed, when a sensor detects a part, the generated signal corresponds to a slot, meaning two changes in value: the rising edge and the falling edge. In this case, the user should have reported two consecutive events.

4.2.2 State of conveyor

The type of signal corresponding to the conveyor state being of the aperiodic type and related to an action of the PLC, the value of τ chosen is 4. The algorithm returns 49 potential variables and selects 9 of them (figure 6). The obtained results show a variable with a *scoreIn* and a *scoreOut* both at zero. The temporal analysis makes it possible to highlight the correlation between this variable and the events declared in the state of the conveyor (figure 7). During the recording, normal production mode and manual production mode are used. These modes are displayed with a different screen on HMI and are notified with several states of the green light indicator. This is why `MW74.7`, image of the green light indicator, and `MW144`, image of the screen number on HMI, obtain a good score. Nevertheless, the temporal analysis shows to the user values that do not correspond to the state of the conveyor.

Variables	Score in	Score out	Correspondence
Modbus TCP.M340.MW33	0,0000	0,0000	Production counter for HMI
Modbus TCP.M340.MW32	0,0000	0,0000	Container filled
Modbus TCP.M340.MW35	0,0000	0,0000	Daily production counter
Modbus TCP.M340.MW75.0	0,0139	0,0358	Rotation cylinder pick&place cap to place
Modbus TCP.M340.MW146	0,0833	0,0725	Display rules for HMI (show/hide)
Modbus TCP.M340.MW74.15	0,0833	0,0725	Cylinder filling shutter
Modbus TCP.M340.M98	0,0833	0,0725	Start order to fill container
Modbus TCP.M340.MW128	0,0972	0,0421	Products counter for rate calculation
Modbus TCP.M340.MW140.3	0,1111	0,0721	Sensor cylinder "in" before station 1
Modbus TCP.M340.MW140.1	0,1111	0,0725	Sensor "part on station 1"
Modbus TCP.M340.MW74.10	0,1111	0,0725	Out cylinder before station 1
Modbus TCP.M340.MW140.2	0,1944	0,0721	Sensor "part after station 1"
Modbus TCP.M340.MW140.5	0,3194	0,0002	Sensor "part on station 2"
Modbus TCP.M340.MW75.1	0,3889	0,0000	Rotation cylinder pick&place cap to pick
Modbus TCP.M340.MW140.4	0,3889	0,0721	Sensor cylinder "out" after station 1
Modbus TCP.M340.MW34.1	0,4028	0,0002	Message ID to display on HMI (bit 1)
Modbus TCP.M340.MW34.4	0,4028	0,0067	Message ID to display on HMI (bit 4)
Modbus TCP.M340.MW34.0	0,4028	0,0067	Message ID to display on HMI (bit 0)

Fig. 4 Results for counting parts and $\tau = 10$. "Correspondence" column is not know by the user. The green gradient defines the ranking of the highest scores. `MW33`, `MW32` and `MW35` have the highest score in and the highest score out.

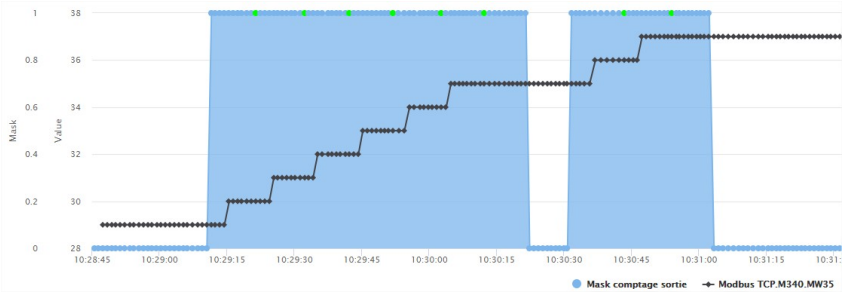

Fig. 5 Time graph of results for counting parts and $\tau = 10$. Windows are in blue. Green dots on windows are events. The black line is `MW35`

The obtained results make it possible to identify the corresponding variable (figure 6). The temporal analysis of potential variables allows the user to discriminate the best results thanks to their knowledge.

5 CONCLUSION AND FUTURE WORK

In this paper, a method to find variable addresses into PLC memory via industrial network is presented. The technique allows the identification of PLC variables, the representation of the physical state of the APS through the recording values of variables and photographs of the APS. The algorithms issued for the method is

Variables	Score in	Score out	Correspondence
Modbus TCP.M340.MW74.0	0,0000	0,0000	Start conveyor motor
Modbus TCP.M340.MW74.7	0,0657	0,1726	Running (green light)
Modbus TCP.M340.MW144	0,0657	0,0262	Display rules for buttons for HMI (show/hide)
Modbus TCP.M340.MW6	0,1010	0,0585	(No information)
Modbus TCP.M340.MW4	0,1313	0,0022	Screen ID to display on HMI
Modbus TCP.M340.MW78	0,2424	0,0262	Display rules for HMI (show/hide)
Modbus TCP.M340.MW34	0,2424	0,0336	Message ID to display on HMI
Modbus TCP.M340.M25	0,3737	0,0000	Start autorisation
Modbus TCP.M340.MW75.6	0,4394	0,0000	Start labelling motor (optional, not used)

Fig. 6 Results for conveyor and $\tau = 4$. "Correspondence" column is not know by the user. The green gradient defines the ranking of the highest scores. `MW74.0` has the highest score in and the highest score out.

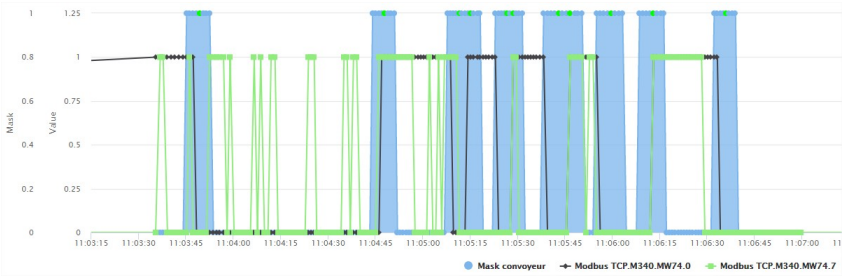

Fig. 7 Time graph of results for conveyor and $\tau = 4$. Windows are in blue. Green dots on windows are events. Black and green lines are `MW74.0` and `MW74.7` variables.

simple, with high efficiency and could be used to retrieve every state of system, as long as it is driven by the PLC.

This method addresses issues that are not covered by current scientific work. It could be used to complement the works identified in the state of the art to improve results by providing physical knowledge of the system that is generally ignored (physical correspondence between inputs and outputs).

The benefits of the proposed methods were experimented on a real industrial case, with tests performed on an automated production system. This approach brings progression for the industrial 4.0, it provides the ability to collect data from many devices that can be seen as a black-box.

The next step is the amelioration of the algorithm. Event sequences detection could improve the differentiation as it would present less results to the user. The detection of significant value corresponding to an event, in addition to variations of signals could also ameliorate the identification. An automatic declaration of events based on computer vision and movement detection will allow longer records without the necessity to check each picture by the user and, thus, obtain better results.

Moreover, this contribution will be integrated in a Digital Twin model for production equipment, as part of the identification phase in the design process of this type of model.

References

1. Barbieri, G., Battilani, N., Fantuzzi, C.: A PackML-based Design Pattern for Modular PLC Code. IFAC-PapersOnLine **48**(10), 178–183 (2015). DOI 10.1016/j.ifacol.2015.08.128
2. Bekrar, R., Messai, N., Essounbouli, N., Hamzaoui, A., Riera, B.: Off-line identification for a class of discrete event systems using safe Petri nets. IFAC Proceedings Volumes **39**(17), 221–226 (2006). DOI 10.3182/20060926-3-PL-4904.00037
3. Commission, I.E., others: International standard IEC 1131-3, Programmable Controllers, Part 3: Programming Languages. International standard (1992)
4. Estrada-Vargas, A.P., López-Mellado, E., Lesage, J.J.: A Black-box Identification Method for Automated Discrete Event Systems. IEEE Transactions on Automation Science and Engineering **14**(3), 1321–1336 (2015). DOI 10.1109/TASE.2015.2445332
5. Ghazivakili, M., Demartini, C., Zunino, C.: Industrial data-collector by enabling OPC-UA standard for Industry 4.0. In: 2018 14th IEEE International Workshop on Factory Communication Systems (WFCS), pp. 1–8 (2018). DOI 10.1109/WFCS.2018.8402364
6. Henßen, R., Schleipen, M.: Interoperability between OPC UA and AutomationML. Procedia CIRP **25**, 297–304 (2014). DOI 10.1016/j.procir.2014.10.042
7. Hennig, C., Kneupner, K., Kinna, D.: Connecting programmable logic controllers (PLC) to control and data acquisition a comparison of the JET and Wendelstein 7-X approach. Fusion Engineering and Design **87**(12), 1972–1976 (2012). DOI 10.1016/j.fusengdes.2012.05.009
8. Hetland, M.L., Last, M., Kandel, A., Bunke, H.: A survey of recent methods for efficient retrieval of similar time sequences. In: Series in Machine Perception and Artificial Intelligence, vol. 57, pp. 23–42. WORLD SCIENTIFIC (2004). DOI 10.1142/9789812565402
9. Koehler, W., Jing, Y.: Automated, Nomenclature Based Data Point Selection for Industrial Event Log Generation. In: H. Yin, D. Camacho, P. Novais, A.J. Tallón-Ballesteros (eds.) Intelligent Data Engineering and Automated Learning – IDEAL 2018, Lecture Notes in Computer Science, pp. 31–40. Springer International Publishing (2018)
10. Lee, E.A., Varaiya, P.: Signals and Systems. Structure and Interpretation of Signals and Systems p. 441 (2000)
11. Lucas-Estan, M.d.C., Raptis, T.P., Sepulcre, M., Passarella, A., Gozalvez, J., Conti, M.: Communication and Data Management in Industry 4.0. In: The Digital Shopfloor: Industrial Automation in the Industry 4.0 Era, Automation, Control and Robotics. River Publishers (2019)
12. Moen, P.: Attribute, Event Sequence, and Event Type Similarity Notions for Data Mining. Ph.D. thesis, University of Helsinki - Finland (2000)
13. Negri, E., Fumagalli, L., Macchi, M.: A Review of the Roles of Digital Twin in CPS-based Production Systems. Procedia Manufacturing **11**, 939–948 (2017). DOI 10.1016/j.promfg.2017.07.198
14. Roblek, V., Meško, M., Krapež, A.: A Complex View of Industry 4.0. SAGE Open **6**(2) (2016). DOI 10.1177/2158244016653987
15. Schlund, S., Baaij, F.: Describing the technological scope of Industry 4.0 – a review of survey publications. Scientific Journal of Logistics **14** (2018). DOI 10.17270/J.LOG.2018.289
16. Theiss, S., Naake, J., Dibowski, H., Kabitzsch, K.: PLC-integrated process monitoring and prediction of the resulting real-time load. In: 2006 4th IEEE International Conference on Industrial Informatics, pp. 880–885. IEEE (2006)
17. Uhlemann, T.H.J., Schock, C., Lehmann, C., Freiberger, S., Steinhilper, R.: The Digital Twin: Demonstrating the Potential of Real Time Data Acquisition in Production Systems. Procedia Manufacturing **9**, 113–120 (2017). DOI 10.1016/j.promfg.2017.04.043
18. Younis, M.B., Frey, G.: Formalization of existing PLC Programs: A Survey. In: Proceedings of CESA, pp. 234–239 (2003)
19. Younis, M.B., Frey, G.: A Formal Method Based Re-Implementation Concept for PLC Programs and Its Application. In: 2006 IEEE Conference on Emerging Technologies and Factory Automation, pp. 1340–1347. IEEE, Prague, Czech Republic (2006). DOI 10.1109/ETFA.2006.355346