The International Journal of Advanced Manufacturing Technology (2019) 102:3243–3264
https://doi.org/10.1007/s00170-018-03263-z

ORIGINAL ARTICLE

Observations on developing reliability information utilization in a manufacturing environment with case study: robotic arm manipulators

Michael Sharp[1]

Received: 15 October 2018 / Accepted: 26 December 2018 / Published online: 13 February 2019

Abstract
Manufacturing environments face many unique challenges with regard to balancing high standards of both product quality and production efficiency. Proper diagnostic health assessment is essential for maximizing uptime and maintaining product and process quality. Information for diagnostic assessments, and reliability information in general, can come from a myriad of sources that can be processed and managed through numerous algorithms that range from simplistic to hypercomplex. One area that typifies the assortment of information sources in a modern manufacturing setting is found with the use of industrial robotics and automated manipulators. Although several monitoring methods and technologies have been previously proposed for this and other assets, adoption has been sporadic with returns on investment not always meeting expectations. Practical concerns regarding data limitations, variability of setup, and scarcity of ground truth points of validation from active industrial sites have contributed to this. This paper seeks to provide an overview of barriers and offer a feasible action plan for developing a practical condition monitoring information utilization program, matching available capabilities and assets to maximize knowledge gain. Observations are made on a real-world case study involving industrial 6 degrees of freedom (DOF) robots actively deployed in a manufacturing facility with a variety of operational tasks.

Keywords Diagnostics · Machine learning · Maintenance · Manufacturing · Monitoring · Operations management · Robotics

1 Introduction

As the digital age provides cheaper and more plentiful options for sensing and recording, the task of curating and utilizing the wealth of information collected in modern manufacturing facilities can seem daunting to both the uninitiated and seasoned veterans alike. In some cases, the massive influx of information and digital processing options has prevented the development or deployment of cost-saving technologies for plant condition assessment and decision support due to indecision or fear of overcommitting to an endeavor with unknown or fuzzy expected returns on investment. Often, even simple steps and basic technologies can provide quick positive returns without the need to commit to a fully developed monitoring and reliability program. This paper seeks to identify barriers and provide basic thought processes and recommendations for assessing and developing a "first steps" level monitoring information utilization program.

The demand for process flexibility and task automation has motivated a sharp increase in the utilization of high-value assets in the manufacturing industry, such as the increasingly popular 6 degrees of freedom (DOF) robotic manipulators. Additionally, the demands for asset availability and product quality have also increased as manufacturers strive to remain competitive in a global environment. Toward this end, accurate asset monitoring and assessment remain forefront in providing optimal process efficiency as well as maintaining the required quality tolerances demanded by the industry customer base.

Equipment monitoring methods, and specifically those developed for highly adaptable systems such as robotic arms, often rely on having precise knowledge of the asset to be monitored as well as the specific operations performed or encountered as part of the planned duty cycle. These explicit rule-based monitoring programs can relate physics data,

✉ Michael Sharp
michael.sharp@nist.gov

[1] National Institute of Standards and Technology, 100 Bureau Drive, Gaithersburg, MD 20899, USA

heuristics, or other well-known operational characteristics of the equipment into some form of diagnostic expected behavior model. Unfortunately, the information needed to create these models is not always available, or may change too rapidly to effectively update and redeploy custom monitoring programs that capitalize on this style of information. With the demand for increasing flexibility in production lines, an adjustable equipment asset may undergo several reconfigurations of action sets throughout their lifetime. Expanding the example of robotic manipulators, one may be tasked to perform multiple movement paths with differing end effectors within a single production cycle, demanding specific behavioral rule sets for each path.

As opposed to explicit rule-based programs, other monitoring programs utilize data-driven techniques, like machine learning, to rapidly develop or adjust underlying models to the specific system. Being less cumbersome to construct, these models can be very powerful in situations where a company does not have the resources to create a specific monitoring program for multiple, possibly dissimilar, systems and assets. However, these models have limitations as the data available directly determine the level of sensitivity and confidence that can be placed in their output. The validation of these models notably requires some degree of "ground truth" or reference point that is often difficult to obtain. A straightforward countermeasure to this comes from proper documentation and contextualization of the data, but unfortunately these are not always available in practical applications.

This paper presents observations from the development of a real-world industrial robot monitoring program and explores best practices to matching the available system information to provide the highest possible level of monitoring and health information. Data from an active industrial manufacturing facility is used to create a generic robotic health monitoring program to detect and discriminate changes in process versus incipient faults or degradation in the system. Observations on the difficulties and barriers during the creation of the monitoring program are used to help develop a road map and best practices guide to developing similar monitoring programs that could hold a broad range of applicability. As opposed to trying to design the optimal integrated monitoring infrastructure, this paper takes a more bottom–up perspective of trying to determine "what can I do with what I have?" as well as looking forward to ask, "what do I need to do more?"

2 Background

Research on holistic applications of Prognostic and Health Management (PHM) technologies has often focused on functional approaches for developing a system-level program, such as in [1]. These approaches look at decomposing a system into functional groups, systems, and assets, then mapping their interfaces to design an optimal monitoring scheme. Research has also gone into defining requirements, architecture, and needs for verification and validation of PHM systems [2]. A good summary of much of the work in this area can be found in [3] and [4]. All of these efforts mostly follow a top–down approach that requires a significant initial investment of effort and/or money, with the goal of fully deploying some optimal program or solution.

Some technologies focus on the methods of deployment for PHM. There has been a recent push for cloud-based services and technologies to offload some of the stress and in-house analysis requirements for manufacturers [5]. Specific examples of technologies are developed for asset monitoring of distributed factories and environments [6]. There are also some works exploring the required architecture of the data for cloud-based software services [7].

Many efforts of developing workable PHM technologies for industrial and manufacturing settings have focused primarily on the development of specific algorithms and use cases. Prior to, and leading into the 1990s, specialized mathematical models were developed and used to monitor and diagnose incipient faults in industrial machines [8–11]. One example of this form of modeling relied on differential equations relating a specific robot's multiple axes and characteristic physical quantities [12]. The advantage of these models is that they require little prior operational data and have the potential to incorporate a priori parameter estimations augmented with collected data to improve the parameter estimations. These rule-based models centered on principles of known physical relationships, first principles modeling, and, in some cases, observed and constructed heuristics about a system.

One major downside to models that rely on analytic equations is that the parametric equations must be specifically developed for, or adapted, to each robot configuration. Such models are time consuming to create, maintain, and optimize. Further, a non-trivial level of understanding of both the robot and the representative math involved is required to create and maintain these models. Modern manufacturing facilities may have multiple different robotic manipulators, each with a unique set of physical properties that will influence physics-driven or other heuristic rule-based models. Many small to medium enterprises (SMEs) simply do not have the time and in-house expertise to deploy these types of models. Original Equipment Manufacturers (OEMs) are best positioned to develop and embed these more rule-based models as they have access to many tiers of performance and quality testing, but as of yet, there has not been broad implementation of access to OEM-embedded health models by equipment operators.

Some work have investigated using information currently available from many OEMs to create base-level expected performance models [13], but comparatively few OEMs offer access to embedded active health monitoring of their equipment.

Late 1990s and early 2000s saw an increase in efforts spent toward the development of less explicit "black box" modeling approaches. Neural networks were at the forefront of these heath monitoring models. Vemuri proposed a method of using a neural network to model the eccentricities of the difference between the commanded actions of a developed control model and the observed robot behavior [14]. This allowed for more general failure alerts that did not follow linear or additive fault behavior (i.e., faults that can be represented as external additive inputs). Vemuri's example case study focused on a simplistic two-joint robot, and although this still relied on having a good kinematics model of the robot, the presented idea of moving away from pure explicitly explainable models exemplifies a trend that would largely continue in many fields up through present times.

Datta et al. proposed a method that extended the generality of robot fault classification using neural networks [15]. By using the coefficients from a three-level wavelet decomposition as input, they trained a neural network to classify the output torque of a robot as either nominal or one of five specific fault conditions. This generic methodology allows for the capture and classification of any arbitrary fault mode; however, it requires correctly classified a priori exemplar data (labeled data) and knowledge of each fault that a robot would be expected to experience. Additionally, results showed that not all faults were easily distinguished in the experimental data set, nor were any variations in operations addressed. Although a good step forward, these shortcomings are typical of models purely derived from operation data without any expert knowledge of the system.

One method for generically capturing all potential anomalies, without explicit prior knowledge of all possible fault modes, is to narrow the range of the modeling set. By exploiting the similarity of signal patterns generated from robots performing repetitive actions, pure data-driven approaches can more easily identify anomalies or "off model" residuals that likely result from some incipient fault or degradation [16]. In their work, Bittencourt et al. rightfully point out that a key step in developing these, and truly most types of data-driven models, is the identification of a proper transformation of the data that will highlight or extract the important predictable aspects of the signal pattern (i.e., feature learning). Metrics for characterizing the distance between distributions were utilized, giving the benefits of inherent fault-type classification, or even potentially accounting for multiple nominal operating conditions, provided exemplar distributions of each potential class exist for comparison. One notable drawback when using distribution matching is that it loses any temporal-based information. While, in some cases, this could allow for smoothing and robustness against minor time-based variances, it also overooks time-dependent indicators of either degradation or faults that may provide earlier warnings of impending risks or health hazards.

Additional work has expanded on the need for proper extraction techniques based on both the type of signal that is available and the type of information that needs to be extracted [17]. The transient nature of robotic arm movement requires specialized techniques to best capture the temporally dependent aspects of the generated signals and information. Wavelets and short-time Fourier transforms have been identified to provide suitable information extraction in such cases, but due to the high variability between physical setups of robots and their respective data capture abilities, no widely available standard methodology for applying these technologies has yet to have been developed and adopted.

This paper seeks to establish a starting point for creating a best practices guide to relating what models, algorithms, and technologies are best applied given the available information from an established equipment system with the presented example of robotic arms. The methods explored here are expected to yield a process useful beyond exclusively robotic systems and may be expanded into a development guide for creating a monitoring program for any active physical industrial asset. Although some specific techniques and models are presented in this paper, this work is not intended to be a comprehensive survey of all classes of PHM technology. Instead, this paper should act as a general starting guide to help lead developers and industry professionals toward the types of technology and information that are tailored to their specific needs. A more comprehensive survey of available health monitoring techniques is provided in other works, such as [18].

3 Methodology

This section establishes a foundation for relating and choosing the best robot monitoring and assessment strategy based on the data and information available. For the scope of this paper, monetary concerns and broader scoped Return on Investment (ROI) investigations are left for planned future work. The focus here is on mapping what is possible from a resource availability perspective. These resources include, but are not limited to, personnel knowledge, physical sensor data, physics-based or relational system knowledge, maintenance logs, operations, and logistics information.

Information assets can come in many forms. Static information about the properties of the system, such as brand information or OEM specifications, can be an asset. Variable information that changes or updates with some frequency are also information assets, such as the data recorded from a thermometer. An important distinction here is that, in this case, a sensor is not an information asset, instead the data recorded from the sensor is the information asset. In terms of this paper, an information asset is not the device, algorithm, expert, etc., that produces data or information, instead it is the data itself. Sources of information assets can be sensors, reports, models, or operation plans: nearly anything that collects, produces, or conveys information.

An information asset provides one or more data streams that can be fed into other information assets, making a network of information flow. The capacity of a given asset is defined based on how its output could be used, i.e., what knowledge could potentially be gained from an information asset? Conversely, the capability of an asset relates to the intrinsic characteristics that enable it to fulfill that potential use. Capability answers the question of how effectively the asset can be used. The first step, before determining the abilities of any information asset, is to simply list and recognize what assets are available. Both existing assets and potential assets should be listed in this process not only to determine what is currently possible, but also to help guide and direct decisions for creation of new assets to ensure optimal levels of system evaluation and health monitoring. This is true for both existing systems that are being retrofitted with a monitoring program, and new systems being designed with integrated active monitoring programs.

Based on the steps outlined in the next sections, the process of developing an information flow network for monitoring a system can be represented in the simple flowchart shown in Fig. 1.

3.1 Determination of information assets

A defining characteristic about information assets pertinent to a well-developed monitoring program is that they provide useful data about the system or its environment (both physical and digital environments). Data about the system can exist in one of several states based on its collection time and frequency. Table 1 provides a detailed list of the various collection states data can exist in.

Information which can be related to a singular value or a single relationship that does not change or is static needs only to be collected once. This kind of data is mostly used as references or for distinguishing groups and conditional hazard rates of the system or to help structure models and prioritize information gathering. An example of a static concept is the physical relationship between cyclic motion and the vibrations of a system. Knowing that a motor is spinning at a particular speed could guide the monitoring program to include a vibration sensor collecting frequencies in a pertinent range. Other more dynamic types of data can be categorized based on when they are collected relative to the use and development of the monitoring program. Historic data is collected prior to or during the development of the monitoring program. Active data is that data which will be collected in an ongoing fashion after deployment of the program. Potential data sources are those that could exist at some future point after the deployment of the program, but are currently unavailable.

Examples of information asset sources include physical sensors (ex. real-time current and voltage probes), or periodic offline testing information (ex. oil analysis results). Even standard "paper trail" logs such as work orders and maintenance history can provide insight and information useful in developing a total health monitoring program if their information can be properly structured and captured by the monitoring program.

3.1.1 Determining sources of information assets

Information assets can come from many places: sensors connected to the system, boiler plate information about equipment, intuition from an expert operator, and predicted behavior from a simulation model are all examples of sources of information assets. The source of an information asset can help determine its use and function in a monitoring program. As shown in Table 2, there are three basic progressive levels of the utilization of information asset sources determined by the source of that asset: probe, processed, and fully contextualized.

Probe sources are those that directly interact with the system or the environment and provide only raw data or information from a singular source. Although these information

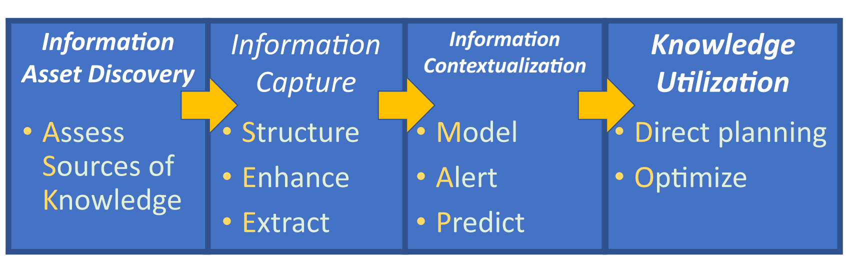

Fig. 1 Information utilization network creation and utilization workflow

Table 1 Collection status of information assets

Data collection status	Description	Examples
Static	– An unchanging property of the system or its environment – Does not change over time	– OEM suggested operational life – Max operating capacity – Deployment system location – Specific gravity of working fluid
Historic	– Past examples of recorded data from or about the system or its environment – Recorded prior and during development of monitoring program	– Past maintenance logs – Past sensor recordings – Past operations log
Active	– Examples of data that are being gathered or are queryable on an actively deployed system that relates information about the system or its environment – Available at and after deployment of a monitoring program	– Streaming sensor values – Incoming maintenance requests – Current atmospheric data for deployed unit
Potential	– Data about the system or its environment for which sources exist or could exist but no examples have currently been recorded – Not available during development of monitoring program	– Deployed sensor not recording – Unstructured information not in computationally useful format – Additional sensor that could feasibly be deployed

sources are the basic building blocks for most monitoring programs, they are rarely sufficient to support any appreciable amount of informed decision making without additional processing or interpretation via an expert agent.

Processed information sources are one level above probes. These sources have taken data and performed some operation to make that data more useful to some aspect of the monitoring program. This could be in the form of noise filtering, information extraction (ex. frequency analysis), combining information from several information assets, simulation modeling, etc. It is possible, and even common, that one base-level probe data source will pass through several steps of processing within a monitoring program becoming one or more processed information sources along the way.

The final levels of sources for information assets are those that can directly answer questions such as “how good/bad is the system?” or “what state is the system in?” These contextualized data sources can come from computerized agents or models, as well as interpretive input from human investigators. Although these sources of information produce information assets that are singularly informative about the system by definition, those assets may still be combined with other information assets from any level to further inform and support decisions within a monitoring program.

Table 2 Information utilization levels

Information contextualization status	Description	Examples
Probe	– Direct recording from the system or environment	– Thermocouple – Voltmeter – Flow meter
Processed	– Data or information that has been refined one or more times to better facilitate use in monitoring program	– Noise filter – Frequency monitoring sensor – Principal component analysis
Contextualized	– Produces information that is directly useful in supporting or implementing decisions regarding the system	– Human agent – Fault detection/classification model – Predictive simulation model

In some cases, the physical agent that interacts with the system may produce either processed or contextualized information. Such cases as smart sensors or human investigators can directly supply second- or third-level information because they have implicit or "hidden" probe-level data recorder that may or may not be part of that physical instigator. For categorization and implementation within a monitoring program, the level of an information source is the same as the information output from that source, which is not necessarily the same as the level it collects. By connecting various information assets from sources across the levels, an information network can be created that targets and implements a fully contextualized system health monitoring program which could be interrogated at different points to provide higher or lower levels of investigative and explanatory information.

3.1.2 Categorization of information assets

After determining the sources of available information, the next step is to determine the type of their descriptiveness. This includes asking what each source describes about the system, how well it informs about your system, and what form the data take.

Determine basic information type Broadly available data can be grouped into several categories based on what it conveys about the system: external, descriptive, or indicative. Detailed in Table 3, all available, or potentially available, information assets of each of these types of data should be listed and have their data categorized appropriately.

One type of data captures external information that could relate to the robotic system. This data holds information about something that either directs the systems' actions or affects its performance, but is largely unaffected by those actions or performance. An example of this could be the conditions and physical environment of the system. Robots operating in a hot and humid atmosphere may suffer higher rates of degradation than those in a cool and dry factory floor. This type of data can be used often as a conditional modifier for prescriptive models. One less straightforward example of external information is planned duty cycle. While this may not receive direct input from the system, it can (and often should) iterate with the predictive output from the monitoring program that is attached to the system. Thus, it may not be completely decoupled from the system's performance, but should still be categorized as a source of external information.

Another, and arguably most important type of information, is descriptive information. This data tells you how the machine is supposed to act, or how it would be expected to perform. This is the broadest category of information, and without which, very little monitoring can be done. More information will be presented on the gathering, generation, and utilization of this style of information in a later section, but the key idea is that prescriptive or descriptive information can be used as a reference to compare indicative information against, thus quantifying the performance of the robot. Most examples of descriptive information are generated by models to predict the output of a robot under certain conditions. These can be physics-based models or data driven, or some combination thereof. The architectures of these models can also vary from probabilistic to rule based, or even utilize machine learning methodologies, such as neural nets and Bayesian networks.

Lastly is indicative information: this is information about how the system is currently operating. A simplistic example of this could be tracking how many hours of operation the system has seen, or the date it was first put into operation. More complex information sources could involve directly sensed signals (e.g., motor torque, current) being generated and recorded from the active robot. Sources of indicative information are necessary to characterize the current

Table 3 Types of information sources

Information source	Description / Use	Examples
External	– Affects or directs system but is not impacted by the system – Can be used to provide conditional information	–Environment or location – Requested/planned duty cycle
Descriptive	– Informs how the system should behave – Can be used as reference to check against when quantifying how "good/bad" a system is behaving – Provides "Target" parameter values of system	– Physics model output – Expected mean time to failure – Probability of failure from a Bayesian belief network – Specific commanded operations
Indicative	– Informs about the current state of the system – Lists "actual" parameter values of system	– Maintenance reports – Voltage signal – Accumulated operational hours

behavior of your system and form the foundation for a monitoring platform.

An example for the types of information could be a robot programmed to pull a lever underwater that only achieves half the desired lever position. The external information is the underwater environment. The descriptive information is “pull the lever.” The indicative information is “half desired position.” In this simplistic example, there are no quantitative values, but in practice, these concepts extend to explicit values, sets, and categories that can be used to further inform a monitoring program.

Determine information asset capacity Nearly all data useful to a monitoring program need to be structured as some quantitative value: the amperage of a motor during operations, the number of failure incidents in the last month, accumulated operational hours, etc. While not all available information is directly applicable to evaluating the system in question, it is the task of a monitoring program to contextualize this information to provide useful decision support. This section focuses on determining what a given information asset can be used to inform (not on what the quantitative values directly are). Table 4 provides a concise list of how information can inform about a system, or the types of capacity of a given information source. A given source of information may have multiple aspects of informative capacity.

Determining the value of the information is not always a straightforward task. Often, it will require expertise about the system, statistics, and general signal processing. Even without expertise in these areas, there are some easy questions that can help guide and inform about the potential for each data source. These questions include:

- Can the data be used to characterize normal operations?
- Does the data provide access to a relative measure of how good/bad the system is performing?
- Can this data be used to infer to what degree some symptom or action is being exhibited?
- Is the data useful for being able to discriminate between different system conditions?
- Does the data characterize one or more fault modes?
- Is the data labeled in some way or provide examples of various possible operational modes of the system?
- Can the information be used to predict current states of the system, either health states or operations?
- Does it have potential future workloads or similar information planned duty cycles?
- Could the data be used to infer what the system will look like or do in the future based on its current and past states?

Answering these questions centers around discovering information that falls into one of three categories: quantitative/qualitative, diagnostic, or prognostic. Qualitative/quantitative information provide indications of how much or to what degree something is. Diagnostic information give indications of a classification, typical discrimination between faults, or some classification of the nominal operations and/or environment of the system. Prognostic information is that which can be used to infer or predict upcoming events and conditions. Many sources of information will have capabilities multiple of these categories, even though a given algorithm may only capture a piece of that information. Similarly, a single data stream may have differing capacities based on the question it is being used to answer. For example, readings from a temperature sensor may be very reliable in measuring the temperature, but have low capacity for quantifying the health of a robot due to the overall higher dependency of temperature on the ambient atmospheric conditions.

If an information asset does not, to some degree, provide at least one of the types of desirable informative capacity about the target system (or its environment), then it is most likely not useful for a monitoring program. However, care must be taken not to discard an asset just because it is not useful at first glance. Matching available information capabilities to interpretive algorithm requirements is an iterative process, as the information that becomes available from one model or algorithm may be useful or even necessary as an input for another algorithm or model. It is sometimes

Table 4 Types of informative capacity

Capablity	Describes / Informs about	Examples
Qualitative/quantitative	– Relative/absolute state of the system – How good/bad and/or by how much	– Inspection reports – Motor voltage signal – Performance model
Diagnostic	– Categorical state or condition of the system – Category of fault	– Maintenance report/fault data – Operations command logs
Prognostic	– Upcoming influences on the system – Where are current trends leading	– Planned duty cycles – Trended past system states

difficult to assess which information will be useful for future activities, which is why it can be convenient to keep an active list of all discovered information assets, updating any changes to their collection states as needed.

Determine capability characteristics of information assets

The characteristics of an information asset can help determine the capability of that asset to be utilized at its fullest capacity. Just because a data stream or information asset contains information about a subject does not mean that it should always be used in every situation. Especially if there could be another source that contains better or more easily accessed information. Some key considerations to determine when classifying the capability of a data source include frequency, availability, reliability, and resolution. A summary of these characteristic is provided in Table 5.

The frequency of a data stream or information asset relates to the speed at which it is collected and used. The frequency of a data stream can be either continuous, on demand, or periodic. On-demand information is queried at irregular intervals based on triggering action. Continuous data streams are those that arrive at regular intervals. It is always the rate of information output (or queried by the monitoring system) that defines the frequency of an information asset, not necessarily the sample rate of the physical probe or sensor. An example of this could be a device that samples 2 s of data once every 5 min at a sample rate of 10,240 Hz then outputs a single vibration spectrum accurate up to approximately 500 Hz. In this instance, the frequency of the data stream that comes from the information asset of the device is once per 5 min (0.0028 Hz).

The availability of an information asset is defined from the ability to query new data from that asset. This can be considered the speed at which new information is available in the data stream. Some data streams will naturally update in batches, perhaps after being held in some long- or short-term buffer; others will continuously update as the data becomes available. Understanding the availability of a data stream allows for the alignment of different data streams and matching the update requirements of various modeling and contextualization algorithms.

The reliability of data can be thought of as a relative indication of its consistency and trustworthiness. Does the data stream provide correct values with a low occurrence of random values that cannot be compensated for? This is similar, but not necessarily the same, as having a high signal to noise ratio and/or low uncertainty in the context of the desired information. Reliability can more appropriately be thought of as a combination of the precision, accuracy, and consistency of the information asset. Broader than just the signal to noise ratio, reliability should also encompass aspects such as rates of missing data, i.e., “lost” or expected sample outputs that are unavailable at the time of query. These lost samples are usually attributed to some form of error in the data stream. For some information assets, it is more convenient to think of accuracy in terms of false-positive rate or frequency of false indications. This too should be considered when estimating reliability.

One occasionally overlooked characteristic of an information asset is its resolution, or the level of detail/granularity a signal can be viewed. This lowest possible change of a signal relates such aspects how continuous or discrete a data stream is. Important to this work, the resolution can help answer the question: are progressive changes in the data stream of a sufficient level to capture important changes in the desired information? While more explicitly discrete variables often require special considerations in model selection, any digitized value must be considered discrete. It is important to ask and recognize if the minimum possible changes in output are sufficient to represent the desired information in a timely and accurate manner.

Asset capability tells which assets contain the most relevant information for a given question, while the capability characteristics help to show how well that information can be conveyed or captured by a given model or algorithm. Different models/algorithms may have different requirements for the quality and type of information they accept and it is important to understand and match assets within these limitations whenever possible.

There is another characteristic that relates to this synchronicity of different data streams, or the *temporal alignment* of those streams. This aspect conveys when the data from multiple data streams are recorded relative to one another. Closely related to the frequency of the data stream recordings, proper management of processing information with independent triggering mechanisms for recording is an important issue.

When utilizing multiple information assets together in a single model, it is important to match characteristics such as availability, reliability, and temporal alignment. When this is not possible, it becomes necessary to have some contingency. For example, imagine the scenario where four data streams are used as input to a Principle Component Analysis algorithm, with three streams updating every 6 s and one updating every 12. To avoid mismatched data processing, the output of the PCA algorithm could be set to only update just after the slow stream changes, approximately every 12 s. This method ignores some of the information from the more rapidly updating streams, but avoids potential anomalies arising from using the “stale” data in the slower stream. Matching or aligning to the lowest characteristic feature of a set of data streams is not always the best solution, but is almost always the simplest and generally can be framed into the most conservative outputs. The larger problem of handling asynchronous data, or other mismatched characteristics, is out of scope for this work, but will be addressed in future documents.

3.2 Capturing and processing information assets

The goal of a well-developed monitoring program is to convert information assets into insights about a system and provide contextualized decision support. Most assets will not directly deliver insights needed to assess the condition of the system. Many assets simply do not have the ability to provide any total system insight without considerable processing and/or input from additional information assets. However, with proper applications of transformations, information-merging, modeling, and similar algorithms, one or more information assets can be utilized to form a new asset with output more pertinent to the state of the system. In terms of data science, this can be thought of as feature extraction and data modeling. In practice for general applications, this may need to be done through several stages to create a network of information flow that eventually leads down to an assessment of the various aspects of the state of the system.

An example of an information asset network for assessing the health of a 6DOF robot could be structured as shown in Fig. 2. In the example, the base-level information assets are represented in blue. These probes are the raw current and voltage signals collected from each motor joint in the robot. They can be combined with modeling techniques to create a new contextualized information asset that represents the health of the respective joint. These joint health information assets can then be combined to create a higher level contextualized model that assesses the total health of the robotic system. From a high level, this simple process can be extended as needed to "collapse" the information in stages until a minimum set of system assessment data streams are accessible.

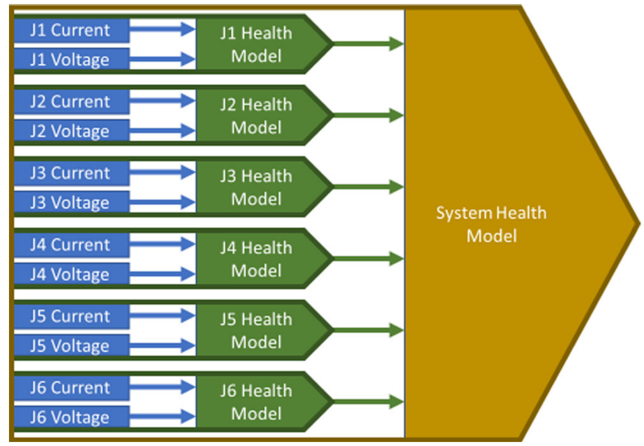

Fig. 2 Example information asset flow network

Although some connections are intuitive from a basic understanding of the system, others require more detailed study. The methods for determining the more complex links are left to future work in another paper.

3.2.1 Choosing proper information asset links

Generally, the utilization of an information asset can take on two forms: univariate and multivariate. Univariate information utilization refers to manipulating a single information asset to extract or contextualize information about the system. Multivariate utilization combines two or more information assets to create a new asset. Any processed or contextualized information asset, regardless of number of inputs, can potentially output multiple data streams that can then be further linked through the information flow network as needed.

Table 5 Types of information characteristics

Characteristics	Description	Examples
Frequency	– How quickly the information is recorded – What triggers collection	– Continuous – Periodic – On-demand
Availability	– How often is new information accessible	– Continuous – Batch
Reliability	– How precise/accurate is the collected information – How often is information lost	– Measured signal variance – Frequency of missing data – False-positive rate
Resolution	– What level of detail is presented by the information asset – Is it sufficient for the target information	– Discrete – Continuous – Granular

Univariate vs multivariate information asset utilization The number of inputs an analysis tool or algorithm uses can help direct the selection and use of that tool within the context of the available data streams. As set out in Table 6, the primary classes of algorithms are those that utilize only one input versus those that take in two or more.

Any transformation, model, or algorithm that only requires a single data stream is considered a univariate information utilization technique. Some of the simpler versions of these contextualization methods come from statistical methods, such as calculating a mean, median, or mode. Examples of more complex methods could be wavelet analysis or performing a Hilbert Huang decomposition [19].

The goal of any univariate process is to highlight and represent the aspects of a data stream that are most informative about the system. This makes the processed data more effective when used by later models or algorithms of the information network. Univariate techniques may also be a forced necessity if there is a limited number of available useful data sources and/or none of the information assets have relationships that can be exploited for additional contextualization or gained insights.

Much like univariate methods, multivariate techniques can be used to create new information asset output as one or more data streams. Typically, multivariate techniques are intended to help capture similar aspects of the input data streams and enhance them. Ideally, this will help isolate the information most useful for assessing the state of the system.

Multivariate techniques can be used for extracting direct system information that is obscured, or has a low signal to noise ratio and is spread across multiple sources. Another use of multivariate methods is highlighting relationship changes between related data streams. Although any individual data stream may indicate "nominal behavior," if the relationship between them starts to significantly alter, it could be indicative of an incipient fault or other anomalies that a user should be alerted about. Extending from that, multivariate methods are necessary when monitoring and quantifying overall system health from a collection of subsystems.

Explicit rule models vs implicit learning models Another important decision faced when creating a monitoring program is the question of whether to use an explicit "hard-coded rules model" or an implicit "learned behavior" model (Table 7). While explicit models can have more intuitive or explanatory internal workings, a more implicit, black box model may more easily provide the desired results with less needed expertise or effort. This unfortunately comes at the potential cost of being unable to understand the internal activations or why some input scenarios do not produce the expected output. To help delineate between the two types of models, consider the example of choosing between using a physics-based model versus a neural network. The physics model gives explicit justifications and explanations for each value in and out of it, but may make some general assumptions that are not exactly true within the real system. The neural network does not make assumptions, but requires volumes of properly encoded data to provide full coverage and interpretation of the system. There are many other models that are built with either explicit or implicitly learned rule sets, even some in-between (ex. hybrid models), but this example is useful for capturing the essence of the benefits and drawbacks of the two model types.

Explicit rule-based models, expert systems, or physics equations all rely on having a deep understanding of the system that you are trying to represent. This has the obvious benefit that for most given states of the system, the model inputs can be directly mapped to outputs that have an intuitive and informative meaning to anyone familiar with the system and/or the model. The drawback to this is that each different system will require a unique model to be created for it, which may not be a trivial task, especially for complex systems. Even between similar systems, the main trade-off for this type of model says that generally

Table 6 Algorithm input classes

Algorithm input	Description/when to use	Examples
Univariate (single data stream input)	– Contextualize data stream – Remove unneeded information – Highlight/extract desired information – Collect and condense series of data points	– Fourier transform – Kernel filter – Statistical values
Multivariate (multiple data stream inputs)	– Capture group system behavior – Combine and enhance weak information sources	– Neural network – Physics models – Principle component analysis

broader assumptions give wider applicability, but also higher misspecification. Some physics-based models will have parameters that can be tuned to help adjust the model to a specific system, but even this does not always guarantee a perfect match to the system. Some systems are just too complex to be effectively modeled with explicit rule-based models. However, for those systems that can be, rule-based models require little (and in some cases no) prior data about the specific system being modeled.

Counter to rule-based models, implicit learning models rely on data and algorithms to extract, capture, and model information about a system with minimal levels of required expertise about the system. Neural networks, latent semantic analysis, or Kernel regressions are examples of this type of model. This group of modeling that overlaps heavily with the modern concepts of artificial intelligence (AI) and machine learning (ML). Many models within the area of AI have matured to the point of being rapidly and widely deployable with minimal training, or cost in time and effort. However, there are trade-offs with such ease of use. These generic modeling techniques require significantly more data to develop effectively than a corresponding heuristics or physics-based model. There are methods to overcome this, but most require some knowledge about the system and how the employed algorithm works.

In general, the rule for picking a model is the less prior explicit knowledge about the system, the more data from that system is required. The more you understand a system, the more likely that some useful explicit rule-based model can be made. The more explanatory and historic data that exist, the more likely some implicit model can learn some useful rule set about the system.

When validating any model, knowledge about or data representing confirmed differing states of the system is a necessity. Without this, all the model can say is if the current state is the same as what it knows. There is no scale or context to relate severity of health (or other performance indicators) to a given difference between the current system and its nominal state. That does not mean that such models are worthless, but it does severely limit their usefulness.

3.3 Information contextualization

3.3.1 Diagnostic assessment methods

Once an indicative system model or information asset has been established, the next step is to contextualize it by scaling the output relative to some descriptive model (or information asset). This step sets limits or ranges quantifying the current state of the system. There are two main approaches to this step based upon the available system information. The first relates to the question "is the system in a faulted state?", while the second question asks, "is the system NOT in a normal state?"

All the steps prior to this point relate to developing one or more information assets that indicate or describe the state of the system in some way. This section relates to how to interpret these data streams, adding the final layer(s) of contextualization needed to produce actionable information for decision support. For some data streams, this is as simple as setting performance limits and alarming when the data stream exceeds specified tolerances. For more complex systems, or when the data streams representing the system state do not directly translate into performance indications, this process may not be so intuitive. As detailed in the previous sections, the methods for diagnosing or quantifying system states can be explicit rule-based or inferred rule-based, univariate or multivariate.

When contextualizing the overall state of the system, one approach is to attempt to answer which of a set of possible faulted and not faulted states the system is in. Framing this as a classification problem implies that you have detailed knowledge of multiple states of the system. For example, by comparing the behavior of a system to both the nominal expected and that of some degraded state, it is intuitive to ask if it is more like one than the other.

When information about possible states of the system is unavailable, in either example data or expert knowledge, one available option is to quantify the system's state away from nominal. Models based on implicitly learned rules can often benefit from this approach that relies on the statistical

Table 7 Analysis rules type selection criteria

Model type	Description/when to use	Examples
Explicit rules based	– Little to no prior data about the system – "Simple" systems or relationships – Expert knowledge about system exists	– Physics models – Expert systems – Bayesian network
Implicit "learned" rules models	– Little to no prior knowledge of system – Large amounts of prior data about system – Highly complex systems or relationships	– Neural networks – Gaussian process models – Principle component analysis – Kernel regression – Support vector machine

significance and natural variation of the system to quantify how far away from normal, in any sense, is the system based on its current behavior. This method can be more difficult in terms of setting correct tolerances for alerting a user to anomalies, and should certainly be updated as better information becomes available through the lifetime of the system. This off-model or distance from nominal method of contextualization can also be applied if multiple possible states are being monitored for. In such a case, this is the equivalent of a "none of the above" or "other" option. Table 8 shows some examples of algorithms that can utilize these two approaches.

Setting tolerance limits on values is just one way to alert end users of anomalies. In some cases, it is better to provide leveled alerts or fuzzy ranking of alarms. In other cases, an accumulation of values might best trigger an alarm. Still in others, some post processing alarm logic that requires a certain number of alarms within a given window is required to alert the user. Matching the alerting method for the user is based both on the characteristics of the incident data streams and the end goals of both the user and the system. While specific details will not be explored in this paper, a guiding principle is to balance the trade-off between minimizing false positives and maximizing early detection.

3.3.2 Prognostic methods

A fully developed heath monitoring program will not only attempt to quantify the current state of the system, but will also look ahead to provide insight to possible future states of the system. Based on knowledge of the current, past, and planned future inputs to the system, what is the most likely state of the system after some finite amount of time? Although answering this may appear complicated, the process can be simplified by grouping most algorithms into three practical use categories: distribution-based, extrapolation, and simulation. This does not mean other methods do not exist, but these methods and combinations of them represent more common approaches.

Distribution-based predictions refer primarily to those that report the expected value of some known distribution given a set of current and/or future conditions. Usually, in this context, these are failure distributions relating to the system being monitored. It is possible to combine results from multiple distributions to calculate overall system expected values, but this is not always an intuitive task and nearly always requires some level of understanding of the system. In the simple case, distribution-based prediction could be reporting an expected time to failure based on nothing but the current accumulated operating hours of the system and a corresponding time of failure distribution. These types of predictions are considered accurate, but also generally the least specific and precise. Extrapolation is a formal method for continuing current trends forward in time to estimate an eventual outcome. For univariate data streams, this can be as simple as fitting an appropriate polynomial function to the most recent recorded values, then solving the function for an arbitrary time step ahead. This becomes more complicated when additionally applying appropriate uncertainty bounds and accounting for errors due to misspecification of the fitting function or window size. Understanding these limitations is beyond this paper, but should at least be acknowledged before further delving into prognostication.

One drawback of nearly all pure extrapolation methods is that they do not account for planned future actions or knowledge of potential, future inputs to the system. Simulation-based methods can provide a solution to this. This broader category of prediction methods is embodied by the idea of running one or more possible scenarios to find the most likely state of the system after a series of possible new inputs to the system occur. Monte Carlo methods, state estimators, and particle filters are some of the more intuitive simulation techniques. Monte Carlo methods, which simulate many possible future scenarios to infer a most probable outcome, also have the added benefit of intrinsically providing a quantifiable level of uncertainty around any prediction made. Although the setup for many simulators is non-trivial, tools are becoming more accessible to aid in their setup and use. Table 9 provides a list of some popular prognostic algorithms and some recommended scenarios for their use in the simplest case.

Table 8 Popular diagnostic approaches

Model type	When to use	Examples
Comparative Classification	– Having knowledge or examples of behavior for multiple states	– Kmeans – Support vector machines
Off models	– Having knowledge or example data of natural variation within some nominal state(s)	– Kolmogorov–Smirnov test – Sequential probability ratio test – Uncertainty limit crossing

3.4 Motivation and summary

This paper proposes four standard steps as a fundamental starting point in the creation of any practical system monitoring solution. The first step being to assess what assets are available (or could become available) and are able to provide information about your system. Next, structure information from those assets into data streams, and transform it to best relate all the information about the system it can provide. Then, contextualize that information to accurately assess the state of your system, alerting users to anomalies and predicting the possible future states of the system. Finally, use the knowledge generated from those system models for directing operations planning, maintenance scheduling, and decision support for logistics and control.

Obviously, not all systems, and their subsequent data and information, are the same. Not every step of this process can always be practically implemented. This could be due to the lack of needed information or data, low quality of information assets, etc. Being able to understand such limitations at the beginning of developing a monitoring program is why, in many regard, the first step of assessing what information is available is the most important. This step can be vital in developing the next steps and informing you what is and is not possible or practical with regard to extracting knowledge about your system. In a landscape where the perception is often "more is better," it is important to remember that more data do not always equal more knowledge, and even when it does, there may not be a one to one return. Instead, it is best to think in terms of better data and information to generate knowledge. The basic breakdown of maximizing information utilization and knowledge gain follows the steps:

- I. Determine available assets
  - a. Identify sources of information/knowledge
  - b. Determine asset potential/ability
- II. Process assets to maximize insight potential
  - a. Structure as data stream
  - b. Enhance/extract information useful for next layer of information network
- III. Contextualize assets to provide knowledge about the system
  - a. Create models of expected vs observed behavior
  - b. Detect, diagnose, and alert users to undesirable behavior or incipient damage
  - c. Predict future states and/or behavior to assess planned goals and operations
- IV. Utilize knowledge and insights about system to inform decision making
  - a. Direct operations planning and scheduling
  - b. Optimize system usage levels and maintenance to meet logistics goals

This procedure is designed to help available data to be optimally utilized for knowledge and decision support.

Figure 3 is an approximate guide to inferring what can be learned from a system given a current set of information assets. This is not meant to be a comprehensive list, or even all possible options for the information assets on the list. Instead, this table is meant to show the most probable knowledge that could be gained by providing exclusively the listed asset. In other words, with median effort, what knowledge about the system could be obtained from each information asset?

A fully developed system monitoring program can yield tangible benefits in the areas of maintenance scheduling, problem area identification, criticality assessment, equipment availability, and more. Understanding the state of a system as it evolves through time can allow informed operations and logistical decisions. Even so, modeling or monitoring every aspect of a system is not required for all levels of decision making. Layering and contextualizing available information assets in a structured information network can allow for multiple levels of probing with a "just what's needed" approach to information management. Although a common approach is to ask "what do I want to

Table 9 Popular prognostic approaches

Prediction type	When to use	Examples
Conditional probability	– Time based estimations distributions – Multiple condition indicators	– Cox proportional hazards model
Extrapolation	– Univariate system state indicator	– General path model
Simulation	– Incorporating future planned states – Complicated systems – Multiple state indicators	– Monte Carlo methods – Particle filters

	Diagnose				Predict			
	Explicit Rules-Based Models	Implicit Learned Rules Models	Comparative Classification	Off-Model Distance	Conditional Distribution Models	Extrapolation	Simulation	Expert Knowledge Needed
Historic Failure Times	None	None	None	None	Medium	None	None*	Low
External Factors Data	Low	Low*	Medium	None	High	None	Medium	Low
Maintenance Reports	Low	Medium*	Medium*	None	Medium	Low	None	Medium
Active Sensors and:								
Physics Based Model	High	Low*	Medium*	High	Medium	High	High	High
Historic Data From:								
Similar Systems	Low*	High	Medium*	High	Medium	High*	High	Medium
This System	Medium*	High	Medium*	High	Medium	High	High	Low*
Planning Data	Low	Low	Medium	Low	Medium	Low	High	High*

* = Varies With Quality of Data

Fig. 3 Example information asset usage potential

know," then build a monitoring program to support that, it can sometimes be more convenient to ask "what knowledge can I gain from existing infrastructure?" then decide what from that matters to my process. Figure 3 and Table 10 aim to give some insight on this second approach.

4 Example case study

This section presents the development of a simple health monitoring program through the application of a generic anomaly detection algorithm and applied to real industry robot data. The data was captured over the course of approximately 2 years from an automotive manufacturing facility. What follows is a brief discussion of the creation of a monitoring program via the process described in the methodology section. There is special emphasis on the exploration of what can and cannot be done based on the available information assets.

Please note that although the data presented in this paper was captured with real values in standard units, representations of data in this document have had their scales altered to obfuscate the true values at the request of the industry partner that provided the data. All values are reported as a percentage of range.

4.1 Determine available factory information assets

The availability of information assets for this case study is deceptively limited. The manufacturer had been collecting data from over 200 similar robotic units performing various tasks throughout the factory. Each of these robots was a 6DOF arm configured to operate with one or more end effectors during their process operations. Although this seems like a large volume of data, there are key elements missing that greatly limit the effectiveness of the available data. Namely, the specific process data was unavailable, and so too was the specific commanded operations data. Additionally, the vast majority of the collected robot data has no corresponding maintenance or efficiency reports that could be used to assess and/or label the performance or infer faults during the recorded times.

For each robot, the measured signals across each motor of the robot were recorded during the same 100-s time window roughly once per week. These measured signals included time-stamped entries for each joint's velocity, electrical current, and temperature. Each signal is recorded simultaneously at a rate of 50 Hz for approximately 100 s. Figure 4 shows an example of a single data capture for one of the six robotic joints.

Most of the robots recorded during the span of this data have no way to directly identify operations activities or health state. This lack of information indicates that implementing a categorical diagnostic modeling algorithm is impractical (Table 8). Instead, an off-model behavior model can be made, such that observations significantly differing from the expected behavior can be labeled as anomalies.

Among the available information assets, maintenance reports were for four specific robots. These reports identified several anomalies that can be used as points of reference in qualifying the developed model. Additional maintenance documents or information about failures were unavailable in a format useful for digital analysis. This is a comparatively small sample of the total set of robots, meaning that any conclusion drawn about the failure

Table 10 Example simple monitoring program layers

Information source	Processing step(s)	Contextualization method	Resulting information
Maintenance records	– NLP – Human agent	Historic frequency and timing	– Problem area identification – Mean time between failures
Vibration signal during motor operations	– 1.Noise filter – 2.Frequency analysis	Bearing pass frequency physics model	– Bearing health – Load monitoring
Historic electrical recordings from a single cutter	– 1.Windowed RMS – 2.Support vector estimation model	Residual monitoring of support vector model	– Relative health and anomaly detection for specific cutter
Historic electrical recordings for population of actuators with labeled operations and fault data	– Auto encoder (neural network)	Recurrent neural network classifier	– Classified state of system – Probable fault detection

distribution of the total population from these reports may have high uncertainty and thus be undesirable.

Beyond the more obvious assets, the knowledge of general relationships between the components of a robotic arm could be counted as an asset. Each joint is driven by an electric motor, and each of those motors should have definable relationships between the electricity flowing through them and the speed at which they move. Recognizing and listing even seemingly simple knowledge or relationships can occasionally reveal unexpected connections or patterns that would have otherwise been unnoticed, thus leading to more sensible and efficient construction of a monitoring program. Table 11 gives a brief listing of the available information assets within this case study. In this case study, no assumptions are made about operations or planning. Even with no explicit operations data, patterns in the available data show that some robots have otherwise unaccounted for shifts in captured signals that seem to indicate a change in process or assigned movement path. This style of abductive reasoning could act as a source of implicit operations information.

4.2 Capture and processing of factory information assets

For this case study, there is limited availability of information assets. Because there is no explicit rule set or predefined relational behavior of the robot and the interaction between its joints, we are forced into the construction of an implicit "learned" behavior model or some set of ad hoc heuristic rules to describe any discovered relationships within the data (Table 7). Unfortunately, this is also not ideal as far as available information assets are concerned, primarily due to the shortage of labeled health information to verify and validate any inferred rule or relationship model. This limitation reduces the capabilities of any developed diagnostic or prognostic algorithm. Despite this, the first step remains to start with the most concrete or explicit knowledge of the system available and utilize that to its fullest potential before attempting to implicitly learn about the system or develop more abstract relationships.

4.2.1 Choosing asset links

Driven by the basic understanding of the components of a robotic arm and the available data streams, an intuitive first step is to investigate the relationship between the current and voltage of each motor during operations. Figure 5 shows the log scale probability density of the relationship between the motor speed and the electric current across all joints for four example robots. This density of simultaneous values between two signals (called a joint density function) is a useful tool for visually investigating potential relationships between signals [20]. This is quickly done by looking for shapes or forms that appear "not random." If unknown, an easy method to check the shape of a pure "random" relationship is to multiply the individual density function (or histogram) of the first signal by the transpose of the second according to the formula, $R = D_1 * D_2^T$. There are more formal statistical methods for confirming the existence of relationships, but this plus intuitive knowledge of the system is a good place to start.

By observing that the joint density of the two data streams have form and shape beyond that which would be expected from random chance, Fig. 5 clearly confirms the

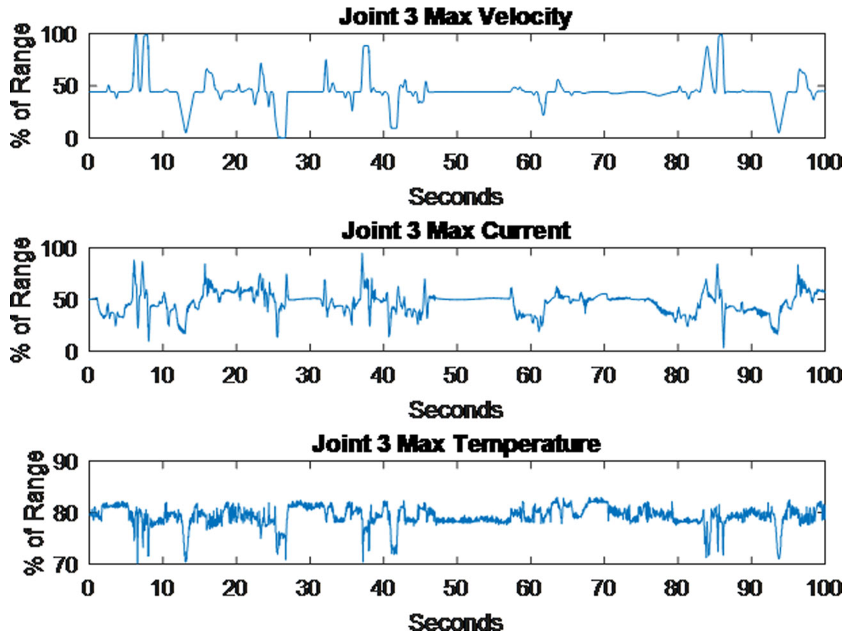

Fig. 4 Sample robot joint data capture

assertion that there appears be a relationship between motor current and the joint speed. The "no-relationship" shape in this case is a roughly symmetrical oval shape, or what is observed as a more peanut-shaped outline with multiple lines and tracks within. By looking at multiple robots, there appears to be distinct types of patterns for this relationship that are characteristic to the individual robot. Between the discernible patterns, there seem to be clusters of similar cross-density functions that may be indicative of the task the robot is performing. Notice robots 1 and 2 as examples of similar distributions versus robots 3 or 4 which seem to be more distinct.

Although within this case study there is no way to explicitly confirm the tasks or action set of each robot, noting these types of clusters could provide useful analytic information. Thus, even without explicit labeling of what physical task each robot is performing, we could infer classes or groups of actions via pattern matching. This construction and labeling of such groups could become a new asset that could be useful in later analysis.

In the absence of additional knowledge, the complex nature of the relationships exhibited by each robot implies that inferred learning models would be better suited than explicit rule-based models for capturing this relationship.

Looking at similar plots for temperature versus current and temperature versus speed, we again see patterns that lack definite structure (i.e., are likely produced by random chance). This seems to counter the possibly intuitive notion that more current might produce more heat and thus have a relationship to temperature. Such examples show that this method of visual analysis does not necessarily mean that there is *no* relationship, but instead that there is no strong or obvious relationship.

In fact, similar visual inspections showed there may be a slight relationship between temperature and speed. Even so, when trying to maximize the investment of effort and time in developing a monitoring program, such weak or questionable relationships are often best to forego during an initial pass. Enhancing and capturing that relationship would require more understanding of that physical relationship and/or deeper levels of expert analysis and algorithm development. Without such, even when applying significant volumes of data through a learning algorithm, the relationship is weak enough that there would be no guarantee that it would add a meaningful amount of contextualization to the system health.

It is reasonable to assert that by applying a black box machine learning algorithm to the significant volumes of data available in this case study, one could model the behavior of the robots without the need to explicitly investigate or discover the system relationships. Indeed, with comparatively few data streams (approx. 18 per robot), an auto-encoding neural network with moderate regularization methods might be able to implicitly utilize important links in the data, similar to work performed with bearings in 2017 [21]. While true, in this case, it is conceivable to employ such a method and expect reasonable-looking results, with no other points for comparison, there is still no way to quantify quality of the results. An exploration of the system, and/or development of rudimentary heuristic rules, can give at least a sense of the performance of any developed model. Further, by exploring the known physical relationships

Table 11 Factory case study: available asset assessment

	Asset(s)	Description	Characteristics	Potential
Expected failure time data	None			
External factors data	None			
Setup data	None			
Maintenance request data	Some	Request logs for 4 robots	Dates for reported anomalies	Confirm anomaly detection algorithms
Expert knowledge of system	Some	Electric motor relationships	Speed/current correlation	Information extraction
Sensor data	Yes	Velocity, current, temperature for each of 6 joints	Semi regular (weekly) snapshots of sensed values @50 hz for 100 s	Off nominal anomaly detection
Planned operations data	None			

of the system, you can ask more informed questions during model construction such as, is it better to group the data streams by joint number or condense them so that the joint number does not matter in the developed relationships? Such explorations are also useful and necessary for situations when such large volumes of data are not available. They can lead to the development of better understood and/or justified models even within the realm of black box modeling with at least a hybrid utilization of some explicit knowledge about the system. Even in cases where a pure machine learning approach is taken, the exploration of the system can lead to more sensible architectures that are predisposed to capture the relationships of the system and reach convergence more rapidly.

4.3 Contextualizing factory information

4.3.1 Implementing a diagnostic method

For implementing a diagnostic model, the first step is to identify a model or set of models that can utilize the identified data streams, and capture and exploit any identified relationships. From the previous section, this case study's primary data streams are continuous, time-varying sensor streams with little knowledge of the physical relationship, and a large volume of historic data from similar systems. Knowing that each joint ideally works in concert to carry out the robot actions, a tiered model approach appears best suited for creating a diagnostic model of the robotic system: a low level to monitor individual joints, then a high level to rate the overall health of the robot.

Joint health–level model According to Table 10, the best diagnostic model for this data set is an implicit learning model. After the development of some understanding of the relationships between the sensors (see Fig. 5), it is intuitive that we should choose an algorithm that can utilize multiple inputs and capture their relationships. Therefore, let us choose a multivariate implicit learning model that requires medium amounts of unlabeled data. For this example, a Kernel regression model is chosen, both for its ease of implementation and its intuitive inner workings [22]. Effectively, this model compares current to past behavior by quantifying the difference between current data samples and those used to build the model.

As used in this context, a Kernel regression is a memory-based interpolation model that compares some input to a set of historically recorded values and approximates an appropriate output based on the proximity to those historic values [22]. For this example, the input and output data streams are the same: the recorded electrical current and the corresponding voltage values. This configuration, termed auto-regressive modeling, allows for a given current and voltage value pair to be compared with historical value pairs. A useful analog for quantifying this difference is to calculate the difference between the input and the output of the auto-associative kernel regression model. These differences are sometimes called residual values.

The data selected for training is vitally important for determining the performance of any implicit learning model. Unfortunately, labeled data within this set is at a minimum, and there is no data explicitly identified as nominal or "healthy" data. When faced with this lack of information, and without better expert knowledge of expected system output, the best models that can be created must ultimately relate to "typical behavior" and not necessarily "healthy behavior." Equating typical with healthy behavior can, if used cautiously, yield comparable understandings of the system. This assumes that the bulk of the data exhibits behavior typical of low to no appreciable degradation because degraded or faulted operations would not be allowed to continue for an extended time in a functional

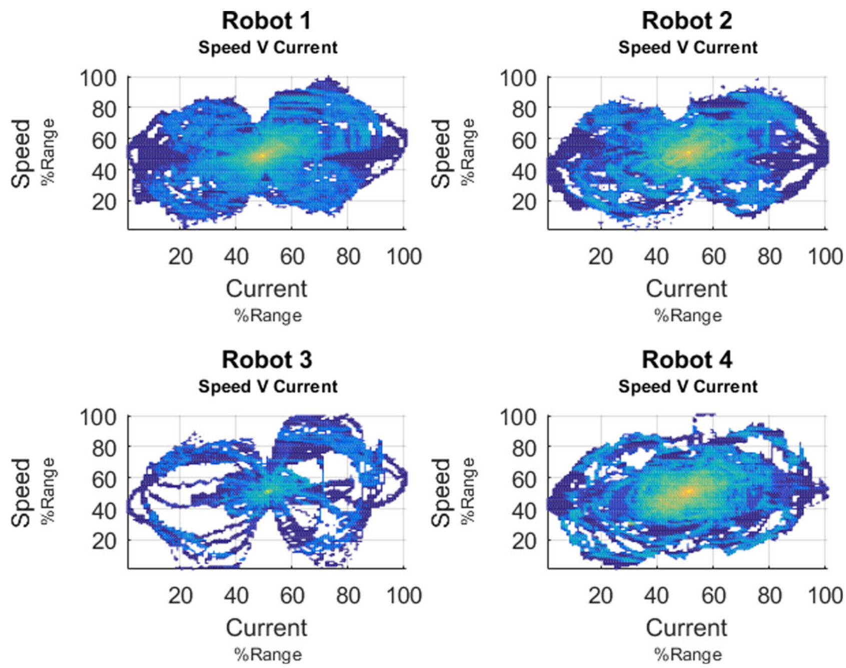

Fig. 5 Example density plots for current vs speed relationship

manufacturing plant. Data sets with no labeled healthy data are forced to adopt some approach near equivalent to saying, "I can't tell you if your system is good, I can only tell you if it is the same as it was in the past."

To maximize the potential to learn "nominal" motor behavior, historic data from multiple robots is concatenated and clustered via a Kmeans algorithm into approximately 500 prototype vectors [23]. These vectors will form the reference memory matrix of the Kernel regression as shown in Fig. 6. In practice, when using this approach, it is not necessary to compile data from all 200+ robots to perform this clustering. Exemplar data from randomly selected motors at various times can be enough to provide adequate coverage of the data space. The exact number will depend on the data set and system, but typically around 15–50 operation sets per configuration can be a convenient minimum. For this data set, that equates to 90 (15 examples $\times$ 6 joints) data cycles per known or inferred robot task. This comparatively lowered amount of data is one of the benefits of constructing a Kernel regression versus an auto-encoding neural network.

Once the Kernel regression model is constructed, it is a simple task to run data from a robotic joint motor through this expected behavior model and quantify the difference between the expected and observed values. By running a set of exemplar data captures separate from those used in the development of the model through the model, an expected level of model error can be created. This could be done across multiple robots and motors to get a value of expected deviations from the model prediction. When applicable, it is generally better to create expected deviation values for each specific robot joint being processed through the model. This allows for more precise adjustments for maximizing the sensitivity and robustness of any alarming criteria.

Robot system–level health model The second phase of creating a robot system monitoring program is to consider the fact that the joints operate together to perform a task. This means creating a higher level model to combine the lower level joint models and ideally help explain some of the non-independent behavior of those models' output.

For this case, based on the understanding of the physical setup, the assumption was made that joints could not spontaneously heal, and that their degradation patterns should

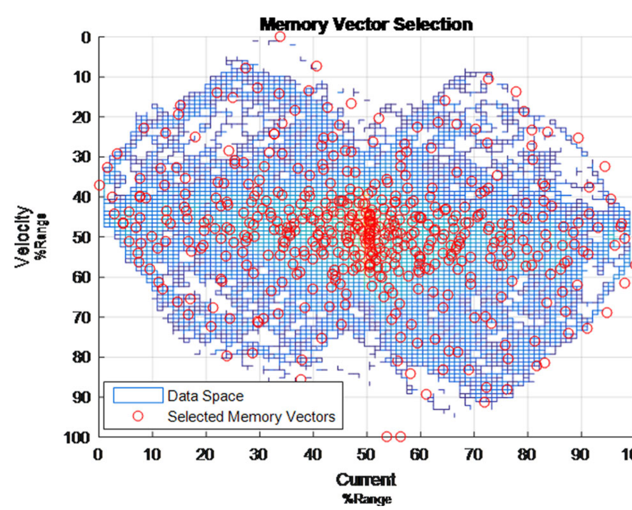

Fig. 6 Memory vector selection

mostly be independent, with the exception of long-term natural wear and/or common causes. Another intuitive assumption is that any rapid onset, short-term group (all joints) wear is most likely caused by an external factor such as load or operation change. This is an extension of the "no self-healing" assumption that can be used to provide probable explanations for short-term anomalies or outliers in the processed data stream.

These assumptions lead to the corollary that the relationship of the deviations from the joint models should remain consistent. A quick investigation (see Fig. 7) shows that most deviations from the joint health models have linear relationships with other joints of the same robot. Based on this, a linear Principle Component Analysis (PCA) [24] model was created to capture and explain correlated movement of the deviations from expected value between joints in a robot . This was accomplished by performing a singular value decomposition [25] on a linear correlation coefficient matrix of the RMS errors from the Kernel regression model for each joint of the robotic arm. Defining the RMS error from the Kernel regression as the corresponding joint's degradation, then the processed matrix had the form $M = [J1_{deg}, J2_{deg}, ..., J6_{deg}]$. To increase the general applicability of the PCA model, M matrices from multiple robotic arms were vertically concatenated then each column was scaled to be mean centered and unit variance. Defining the PCA model as the minimal number of Principle Components needed to explain 90% of the matrix variance, the measure of robot degradation is reported as the two metrics: movement in-model (collinear movement) and deviations from collinear movement.

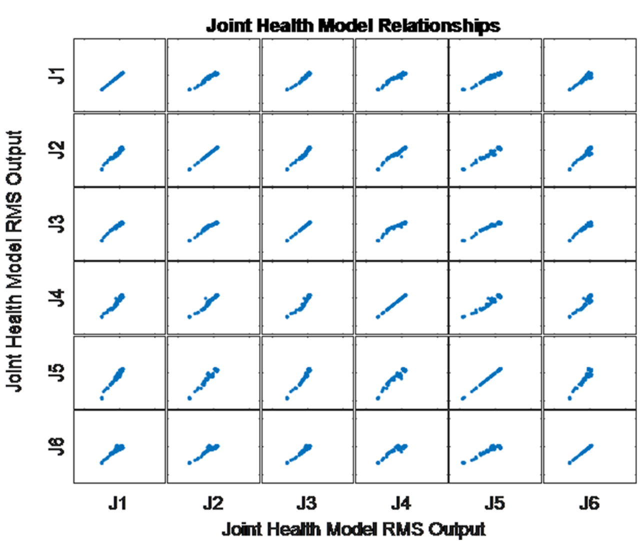

Fig. 7 Example robot joint health model relationships

By monitoring the deviations from the robot system–level expected behavior model, we can create stronger justification for alerting a user to a degraded joint. Conversely, if we observe elevated values that correspond to group movement, the alert would be more likely due to an environmental or process change. Within the framework of the developed PCA model, the first two Principle Components (PCs) capture the bulk of the collinear movement (greater than 90% for this data) and can be used to indicate changes that affect all the joints of a robot.

Finally, to discriminate the collinear components from the individual joint degradation, a simple secondary measure of distance from the group can be employed. Ranking the residual output from the joint expected behavior model as a distance from the median of the six corresponding joints at a given observation in time can help relate which joints are exhibiting increased stress compared to the group. This singular increased stress can intuitively be ascribed to individual joint wear as opposed to environmental or process stressors.

Because the PCA model is based on unit variance scaled inputs, a reasonable limit in to trigger alarms is any value exceeding 1, as this is the scaled variance of the historical training data. A more detailed analysis of the expected behavior of the processed values across individual robots could potentially yield more strict and individualized limits for alerting. Although such individual limits could provide more precise indications of the current condition of individual robots, there is a significantly increased effort involved to avoid potential overfitting which could drastically increase the number of false alarms over time. Such trade-offs should be managed within sight of the expected return, meaning that it may be worthwhile for high risk or highly critical physical assets, but less so for more commonplace equipment or equipment with lower work loads.

In less complicated terms, this whole process is just a way to optimally estimate the total degradation of a robot as a combination of the six joint degradation values, trying to differentiate collinear and non-collinear movement. But even if a combination is less than optimal, or is very simplistic (such as defining robot degradation as the average of the individual joints), the measures can still be useful so long as proper bounds and expected value ranges are characterized and used to set alarms. The important thing to recognize is that in many cases, when suboptimal amounts of information are available, absolute metrics are difficult or impossible to define. In such cases, relative metrics can still provide useful insight into the evolution and performance of equipment as it runs throughout its lifetime.

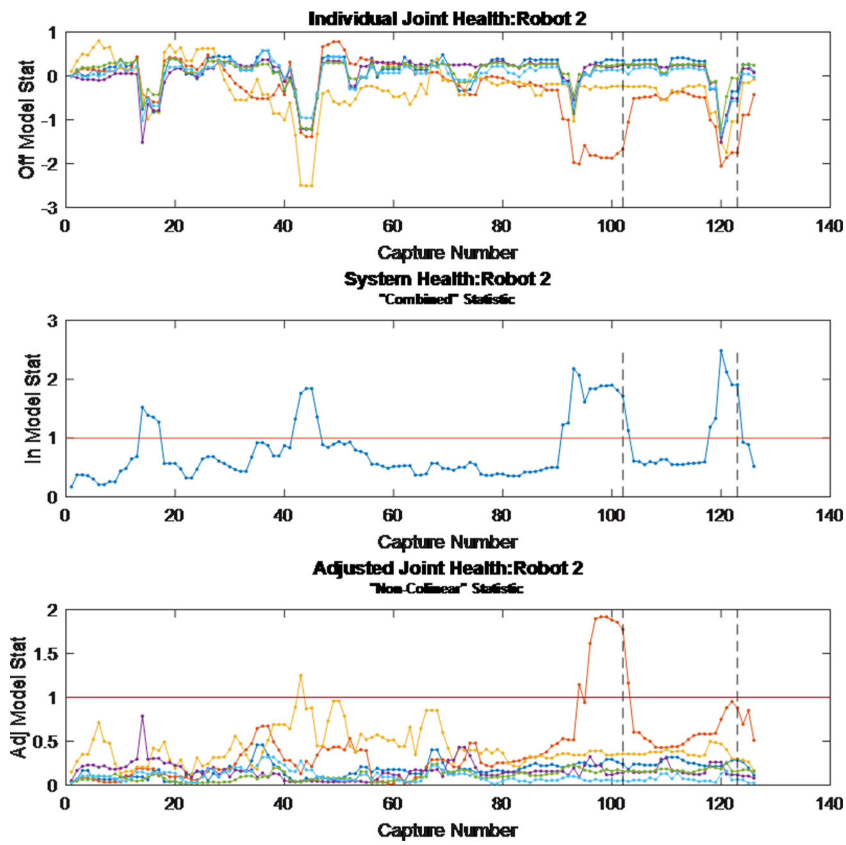

Fig. 8 Robot 2 example monitoring models output

4.3.2 Implementing a prognostic model

For this set of data, there is insufficient ground truth data to develop a fully validated prognostic model. The elevated values that precede the known maintenance requests do show promise for early detection of degradation and anomalous conditions, but with the few hard examples of known faults, it is difficult to characterize critical value limits

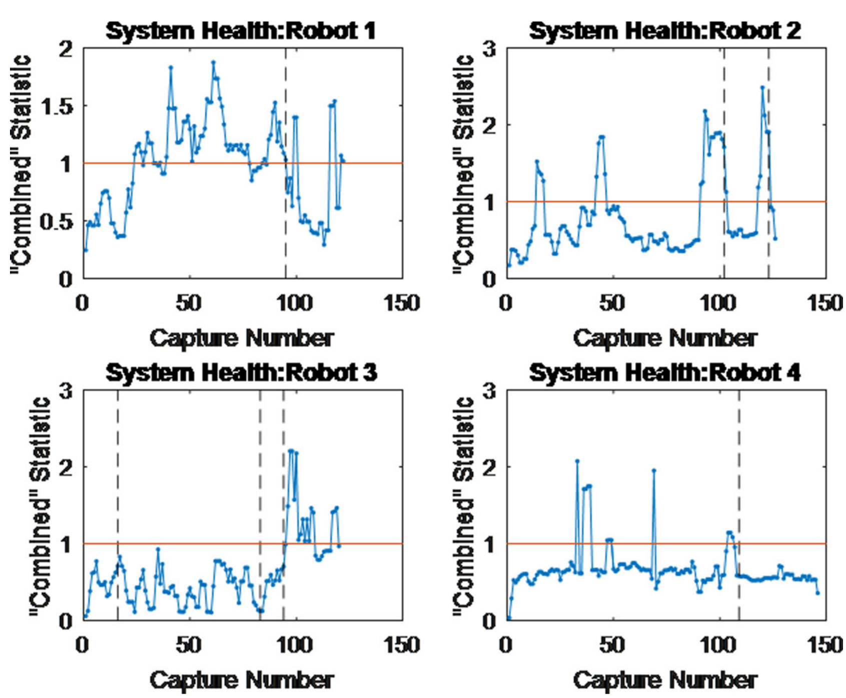

Fig. 9 System health metric for robots with maintenance requests

or characteristic fault progression forms. A greater number of confirmed examples or some more explicit understanding of the faults exhibited (or expected to be exhibited) would be necessary to create a fully developed prognostic model. For example, a physics simulator could be used in conjunction with the available data and developed models to characterize the expected behavior of a faulted robotic system, but such work is beyond the scope of this paper.

4.4 Results

This data set has four robots with at least one confirmed maintenance request report over the period of available data. In each of these cases, a distinct change can be seen in the monitored metrics before and after the dates of the requests. This can be seen that the developed models could in practice be used to isolate and alert users to various anomalies of a robotic system. Figure 8 shows an example of the various levels of the monitoring program. The known dates of reported anomalies are signified by vertical dashed lines.

In this figure, the upper chart displays the residual output from the expected behavior model for each of the individual joints. The second chart shows the combined robotic "system level" health indicator, whereas the bottom chart displays the individual joint health adjusted for system health. In each of the cases with known anomaly reports, there is a distinct change near each of the reported dates. The only outlier in this set is robot 3, shown in the lower left corner of Fig. 9. The first two anomaly dates show no significant change, while the last shows a strong and abrupt change past the standards values after the report. This would seem to indicate that the first two events were fast acting to a point not able to be captured by the periodic data, and that there may have been an induced maintenance error or change after the third. Joint no. 2 exhibits severe deviations from the expected behavior model from approximately data capture 100 and onwards.

5 Conclusions and future work

Based on available literature, research, industry interactions, and practical experience, there is a need for a standard set of guidelines for implementing a practical robotic arm monitoring program in an industrial environment. The procedures and examples shown in this paper begin to give directives and methods for developing monitoring programs with special emphasis on full utilization of the available data and information assets.

Future work has already begun to further develop and formalize these procedures with more example data sets by expanding beyond the special case of robotic data. The research team welcomes the opportunity to acquire additional data sets to further assess its monitoring program including the refinement of its diagnostics and prognostics approach.

6 NIST disclaimer

The views and opinions expressed herein do not necessarily state or reflect those of NIST. Certain commercial entities, equipment, or materials may be identified in this document to illustrate a point or concept. Such identification is not intended to imply recommendation or endorsement by NIST, nor is it intended to imply that the entities, materials, or equipment are necessarily the best available for the purpose.

Publisher's note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

References

1. Li R, Verhagen W, Curran R (2018) A functional architecture of prognostics and health management using a systems engineering approach. In: European conference of the prognostics and health management society
2. Saxena A, Roychoudhury I, Celaya J, Goebel K (2012) Requirements flowdown for prognostics and health management. In: AIAA Infotech at aerospace conference and exhibit 2012, p 13. https://doi.org/10.2514/6.2012-2554
3. Vogl GW, Weiss BA, Moneer Helu O (2016) A review of diagnostic and prognostic capabilities and best practices for manufacturing. J Intell Manuf 0:1–17
4. Lee G-Y, Kim M, Quan Y-J, Kim M-S, Kim TJY, Yoon H-S, Min S, Kim D-H, Mun J-W, Oh JW, ang Chung-Soo Kim IGC, Chu W-S, Yang J, Bhandari B, Lee C-M, IHn J-B, Ahn S-H (2018) Machine health management in smart factory: a review. J Mech Sci Technol 32(3):987–1009
5. Moa K, Chen Z, Tao X, Xue Q, Wu H, Hou J (2017) A visual model based evaluation framework of cloud-based prognostics and health management. In: IEEE international conference. Smart Cloud, pp 33–40
6. Wu D, Jennings C, Terpenny J, Kumara S, Gao RX (2018) Cloud-based parallel machine learning for tool wear prediction. Journal of Manufacturing Science and Engineering
7. Tountopoulos V, Kavakli E, Sakellariou R (2018) Towards a cloud-based controller for data-driven service orchestration in smart manufacturing. In: Sixth international conference on enterprise systems
8. Specht R, Isermann R (1988) On-line identification of inertia, friction and gravitational forces applied to an industrial robot. IFAC Proceedings Volumes 21(16):219–224
9. Mitsuishi M (1989) Diagnostic system for robot using a force-torque sensor. Robotersysteme 5:40–46
10. Isermann R (1990) Estimation of physical parameters for dynamic processes with application to an industrial robot. 1990 American control conference
11. Wünnenberg J, Frank P (1990) Dynamic model based incipient fault detection concept for robots. IFAC Proceedings Volumes 23(8):61–66

12. Freyermuth B (1991) An approach to model based fault diagnosis of industrial robots. Proceedings. 1991 IEEE International Conference on Robotics and Automation 2:1350–1356
13. Zhou C, Chinnam RB, Korostelev A (2012) Hazard rate models for early detection of reliability problems using information from warranty databases and upstream supply chain. Int J Prod Econ 139(1):180–195. Supply Chain Risk Management. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S092552731200165X
14. Vemuri A, Polycarpou M, Diakourtis S (1998) Neural network based fault detection in robotic manipulators. IEEE Trans Robot Autom 14(2):342–348
15. Datta A, Mavroidis C, Krishnasamy J, Hosek M (2007) Neural netowrk based fault diagnostics of industrial robots using wavelt multi-resolution analysis. 2007 American control conference
16. Bittencourt AC, Saarinen K, Sander-Tavallaey S, Gunnarsson S, Norrlöf M (2014) A data-driven approach to diagnostics of repetitive processes in the distribution domain – applications to gearbox diagnostics in industrial robots and rotating machines. Mechatronics 24(8):1032–1041
17. Jaber AA, Bicker R (2018) Development of a condition monitoring algorithm for industrial robots based on artificial intelligence and signal processing techniques. International Journal of Electrical and Computer Engineering (IJECE) 8(2):996
18. Kothamasu R, Huang S, VerDuin W (2006) System health monitoring and prognostics — a review of current paradigms and practices. Int J Adv Manuf Technol 24(28):1012–1024
19. Huang N, Shen Z, Long SR, Wu MC, Shih HH, Zheng Q, Yen N-C, Tung CC, Liu HH (1998) The empirical mode decomposition and the hilbert spectrum for nonlinear and non-stationary time series analysis. Proc R Soc Lond A 454:903–995
20. Prohorov Y, Rozanov Y (1969) Probability theory, basic concepts. Limit theorems, random processes. Springer, Berlin
21. Xia M, Li T, Liu L, Xu L, Silva CWD (2017) Intelligent fault diagnosis approach with unsupervised feature learning by stacked denoising autoencoder. IET Sci Meas Technol 11(6):687–695
22. Nadaraya EA (1964) On estimating regression. Theory of Probability and its Applications 9(1):141–142. [Online]. Available: https://epubs.siam.org/doi/abs/10.1137/1109020
23. Sculley D (2010) Web-scale k-means clustering. In: Proceedings of the 19th international conference on world wide web - WWW, p 10
24. Hotelling H (1933) Analysis of a complex of statistical variables into principal components. J Educ Psychol 24:417–441 and 498–520
25. Golub GH, Reinsch C (1970) Singular value decomposition and least squares solutions. Numer Math 14(5):403–420