EC3.21

Methodological Guidelines for Rule based Monitoring

L3.5.2.3

contact@confiance-ai.fr | www.confiance.ai

Document reference: 321BA

Contributors

	Name	Organisation	Role
Responsible for the deliverable	Paul-Marie RAFFI	IRT SystemX	Research Engineer
Scientific responsible	Fateh Kaakai	Thales	Research Director
Co-authors	Fateh Kaakai	Thales	Research Director
	Paul-Marie RAFFI	IRT SystemX	Research Engineer
	Thibault ROYET	IRT SystemX	Research Engineer

Document Control

Revision	Date	Commentary	Author
v1.0	04/10/2023	Initialization	Paul-Marie RAFFI
v1.1	18/12/2023	Delivery	Paul-Marie RAFFI

Contents

A. Introduction and abstract

A.1. General introduction to trustworthy AI challenges

Trustworthiness in AI within critical systems (systems that can directly or indirectly affect human life and moral entities) is essential for its widespread adoption (by the industry, the decision makers, the general public, etc.) and poses the following significant challenges.

- First, how to design AI models, so that, by construction, they satisfy trustworthy properties (accuracy, robustness. . .).
- Secondly, how to characterize these AI models, for example to understand and explain their behavior and their adequacy to the operational domain.
- Then, how to implement and embed those AI models on hardware, by making them fit for the target without losing their trustworthy properties.
- Another question is, what methods of data engineering to apply in order to, among other topics, manage important volumes of data and adapt to the evolution of the operational domain.
- At system level, what verification and certification processes to consider specifically for AI-based systems.
- Finally, a federation of all these matters is necessary to build an end-to-end methodological approach, supported by a consistent engineering environment compatible with industrial practices.

These are the challenges, among others, that the Confiance.ai program addresses.

A.2. Introduction to Rule-based monitoring

A.2.1 Rationale for this methodology

Rule-based monitoring is part of the methodologies that can be used to control the AI models and their input data in operation.

A.2.2 Scientific Challenges

This document addresses the scientific challenges:

- Components with integrated self-monitoring of the detection of the exit from the operating zone.
- Calculation of the confidence score by various approaches.

A.2.3 Trustworthiness Attributes

Maintainability Robustness

A.2.4 Target audience and disclaimer

This document has been written in a focus to be easy to comprehend for any reader having a few notions in Mathematics, Digital Imagery and Time Series. It is intended to give rule-based

methods for:

- Data Scientists and Data Managers who want to improve the quality of their training datasets by detecting and annotating anomalies.
- AI Algo Engineers and Safety Engineers who want to control the inputs and outputs of AI models in operation.

A.2.5 Glossary and terminology

This section presents useful definitions of key concepts that will be frequently used in the rest of this report. It is recommended to have a good understanding of these definitions before deep diving into the next chapters.

A.2.5.1 Operational Design Domain (ODD) Definition

According to the taxonomy document developed by EC2 [L1.1.1 - 1st version of Confiance.ai Taxonomy], the Operational Design Domain (ODD) is defined as follows:

A.2.5.1.1 ODD definition in EC2 Taxonomy

Operating conditions under which a given driving automation system or feature thereof is specifically designed to function, including, but not limited to, environmental, geographical, and time-of-day restrictions, and/or the requisite presence or absence of certain traffic or roadway characteristic SAE J3016 (2018).

This definition in EC2 taxonomy document is coming from the automotive domain (Road Motor Vehicle Automated Driving Systems) has been considered by project EC6 to develop a top-down approach to specify ODD from system considerations.

A.2.5.1.2 Concept of AI Model ODD

Now, if we assume that this "ODD as specified" exists and has driven the development of an AI model as depicted in in Figure A.1. Let also assume that the AI Model has been verified against its allocated requirements and is ready for deployment in the field. Therefore, at the end of the development process, we obtain what we call an "ODD of the AI Model as designed, implemented and verified)" or simply and shortly "AI Model ODD". To characterize this AI Model ODD, we need to measure the performance of the AI Model in operational conditions as close as possible to real one (i.e., nominal and abnormal conditions). By using inference results of the AI Model to input data containing nominal values and anomalies, we characterize the AI Model ODD through the measured ranges of parameters where the AI Model provides a correct output within an acceptable margin error. This AI Model ODD will be the baseline for the online monitoring capability since it corresponds really to the domain where the AI Model guarantee the expected behavior and the expected level of performance (since it has been verified against its allocated requirements during the development process). At this point, many questions arise like: *does the AI Model ODD perfectly fit with the specified ODD? Is the AI Model ODD a superset or a subset of the specified ODD?* All these questions are legitimate and should be managed by relevant process interface between the system engineering processes and AI Model development processes (EC2, EC4 and EC6 scopes). In this report, we develop more deeply this concept of AI Model ODD since it is a key concept for the online monitoring capability.

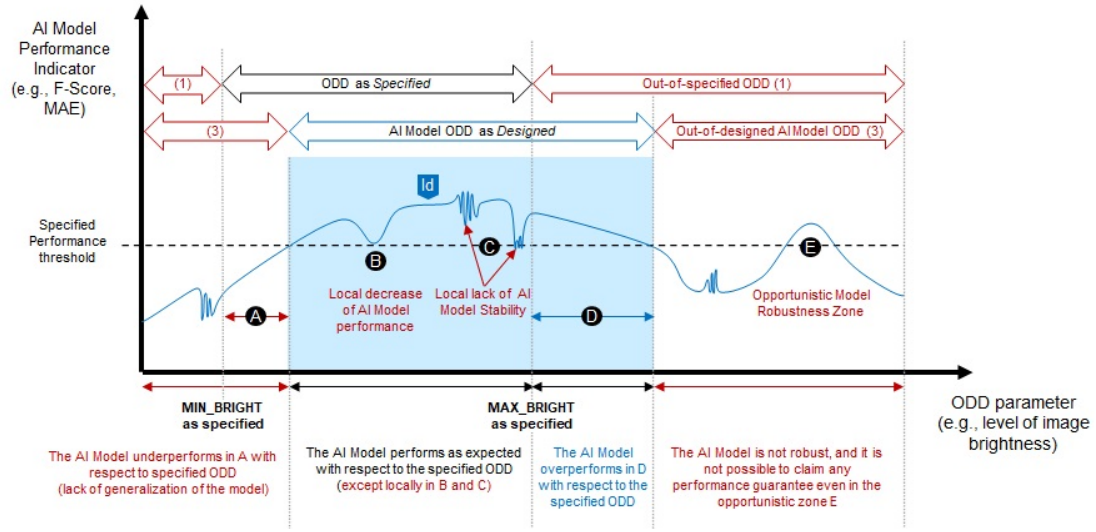

Figure A.1: Specified ODD at system level and AI Model ODD as designed

A.2.5.1.3 Principles of the AI Model ODD

There are various potential classes of anomalies depending on the data type (images, time series, natural language, ...) and on the type of problem the industrial is trying to address (classification, object detection, semantic segmentation, regression task, ...). For example, the Renault welding Use Case is a problem of image classification into 3 classes: "Compliant welding", "non-compliant welding", or "Unknown" (see the use case description in Part III-Appendixes). Based on the system ODD as specified with the use case owner, we identified potential anomalies of interest for this type of problem (images with too much color, blur, translation, rotation, brightness, darkness, ...) and defined ranges and increment steps (sampling) for these anomalies. Some anomalies such as Rotation can be selected in their whole range, from 0 degree to 360 degrees, and only the increment step is chosen. Other anomalies such as Blur or Brightness cannot be estimated easily until tested against the AI Model during a calibration phase (few cycles of tests before finding a range of interest). There are also hardware limitations that can reduce the number of steps. A trade-off has been found during the Batch 2 for each class of anomalies between storage, time of computation and performance of the Model ODD.

A.2.5.2 Model Robustness and Stability Definitions

According to the taxonomy document developed by EC2 [L1.1.1 - 1st version of Confiance.ai Taxonomy], the definition of model robustness and model stability are the following F. et al. (2021):

- The Global robustness of a model or a system is the ability of the model/system to perform the intended function in the presence of abnormal or unknown inputs.
- The local robustness is the extent to which the system provides equivalent responses for similar inputs.

The above definitions are very general and are reworded in the EUROCAE/SAE AS8963 (draft4) SAE (2022) in more accurate terms using the ODD concept and where the global robustness is called robustness and the local robustness is called stability.

A.2.5.2.0.1 Robustness of an AI-based model/system The capacity of an AI-based model/system to preserve its expected / intended performance under well-characterized abnormalities or deviations to its inputs and operating conditions outside its operational design domain (ODD).

A.2.5.2.0.2 Stability of an AI-based model/system The capacity of an AI-based model/system to preserve its expected / intended output(s) under well-characterized and bounded perturbations to its inputs and operating conditions within its operational design domain (ODD)
In the rest of this report, we will use the definitions of SAE (2022) which are consistent with EC2 taxonomy since they provide an important reference to the ODD and a clear distinction between robustness to invalid inputs outside the ODD and stability within the ODD.

A.2.5.3 Taxonomy of anomalies for time series

A.2.5.3.1 Classes of time series anomalies

In this paragraph are described 3 classes of anomalies often used in the scientific papers Audibert (2021) for time series to classify anomalies.

- **Punctual anomalies**
  Punctual anomalies correspond to data points that are outside the ODD.

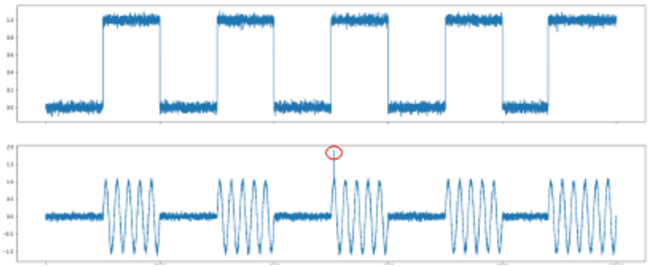

Figure A.2: Punctual anomaly example

- **Contextual anomalies**
  Contextual anomalies correspond to a group of points that is outside the ODD in a specific context.

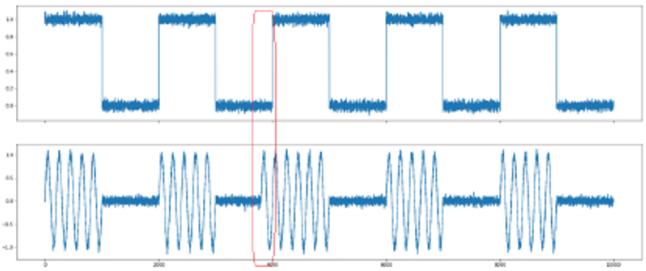

Figure A.3: Contextual anomaly example

- **Collective anomalies**
  Collective anomalies correspond to a group of points that are inside the ODD individually but their appearance together is outside the ODD.

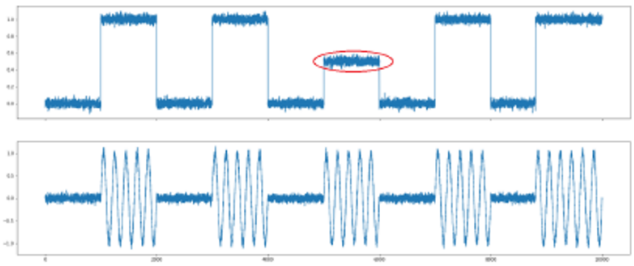

Figure A.4: Collective anomaly example

A.2.5.3.2 Types of time series anomalies

In this paragraph, we describe most of the types of anomalies that can be found in time series.

- **Phase modulation**
  The phase modulation correspond to the shifting in time of a periodic signal.

$$g(x) = f(x \pm \Delta T) \tag{A.1}$$

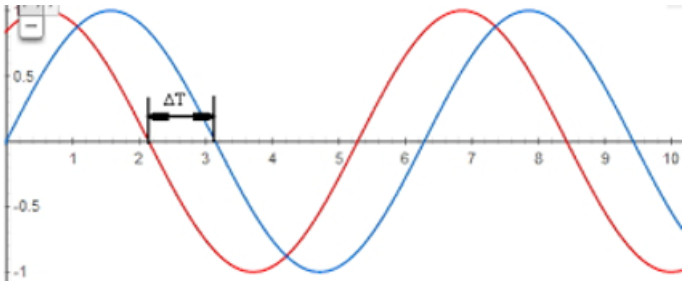

Figure A.5: Phase modulation example

- **Amplitude modulation**
  An amplitude modulation (AM) corresponds to the modification of the signal amplitude. The following formula is commonly used in telecommunication Wikipedia (2022a):

$$X_p(t) = A_p * cos(w_p * t) \tag{A.2}$$

  We will generalize as:

$$g(x) = f(x) * A(x) \tag{A.3}$$

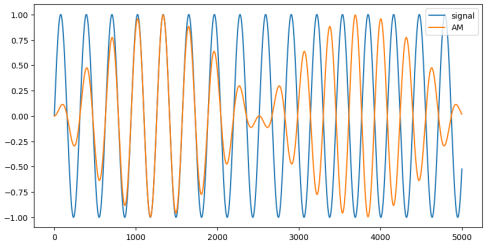

Figure A.6: Amplitude modulation example

- **Frequency modulation**
  The frequency modulation (FM) corresponds to the modification of the signal frequency.

$$g(x) = f(x * f_m(x)) \tag{A.4}$$

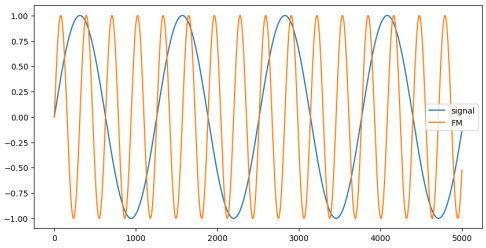

Figure A.7: Frequency modulation example

- **Abnormal Offset/Drift**
  The abnormal offset corresponds to the an additional component in the distribution of the data that is outside the ODD.
  The drift is a sub-type of an offset corresponding to the accumulation of error in time.

$$g(x) = f(x) + k(x) \tag{A.5}$$

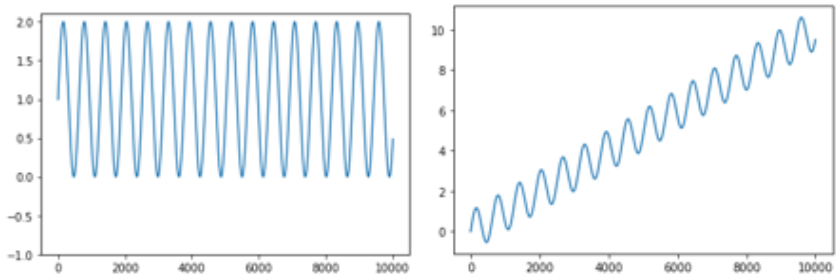

Figure A.8: Abnormal Offset/Drift example

- **Noise/White noise**
  Noise is the realisation of a random process, it creates random irregularity in the data. A process ε_k with discrete time is qualified as white noise Wikipedia (2022b) if:

$$\begin{aligned} &E[\varepsilon_k] = 0; \\ &E[\varepsilon_k^2] = sigma^2; \\ &E[\varepsilon_k \varepsilon_l] = 0, \forall k \neq l; \end{aligned}$$

  Most of the time, in the scientific papers, the white noise is defined as additive noise.

$$g(x) = f(x) + N(x) \tag{A.6}$$

  Usually the noise is a random phenomenon that is modifying the signal according to a distribution, the most common way to represent the noise is with a Gaussian distribution.

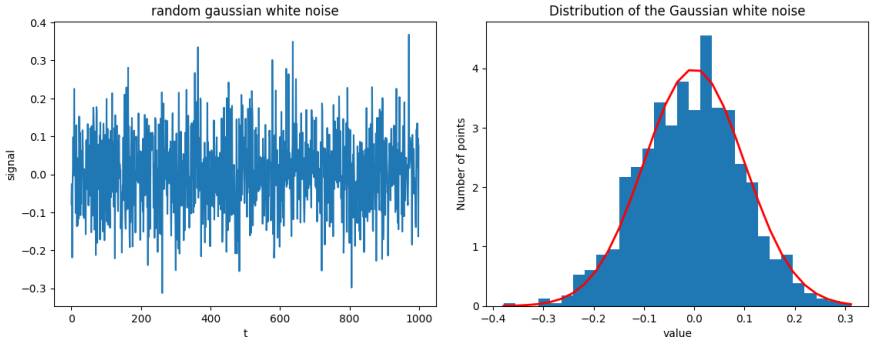

Figure A.9: Example of Gaussian noise and distribution

- **Abnormal seasonality**
  Seasonal compounding or seasonality corresponds to a phenomenon that repeats itself at regular time intervals (periodicity) Audibert (2021). So we defined an abnormal seasonality as periodic pattern that is outside of the ODD.

$$g(x) = f(x) + S(x), \text{S a periodic signal} \tag{A.7}$$

- **Abnormal pattern**
  Abnormal pattern corresponds to the appearance of a group of points (pattern) that is defined as outside of the ODD.
- **Multivariate anomaly**
  Multivariate anomaly corresponds to points or patterns in two or more variables that are normal locally but defined as outside of the ODD when they appear together.

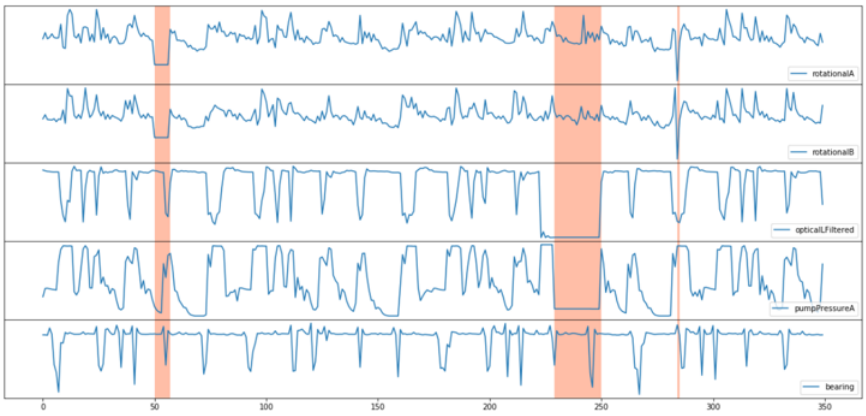

Figure A.10: Multivariate anomaly example

- **Dropout**
  The dropout corresponds to missing values. It can happen during a sensor failure.

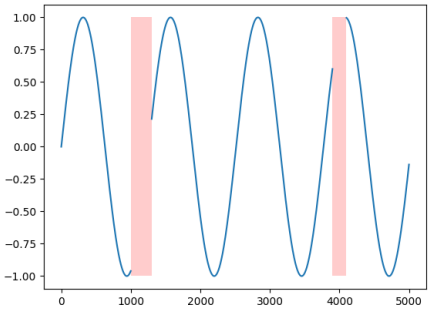

Figure A.11: Dropout example

- **Drag**
  The drag corresponds to data points that are equal to the previous data point. It can happen during a sensor returning a default value which is locked due to a failure.

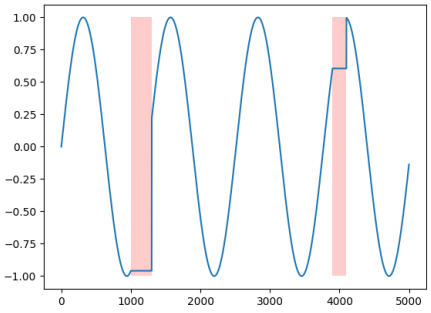

Figure A.12: Drag example

- **Categorical anomaly**
  The categorical anomaly corresponds to categorical data point that are defined as outside the ODD.

A.2.5.3.3 Classification of time series anomalies

The types of anomalies can be classified into the 3 classes of anomalies previously mentioned. This methodology allows to evaluate the difficulty to generate and detect anomalies. For instance the contextual anomalies will require knowledge about the context that won't be needed in punctual anomalies. It also allows to extrapolate insight on the detection process and similarities between types of anomalies. For example with punctual anomalies, only a few data points will be needed in contrary to collective anomaly.

For forecasting problems, we have matched the different Rules-based monitoring classes with the anomaly classes:

A.2.5.4 Taxonomy of anomalies for images

There is no generic taxonomy of anomalies for images in this version of the report since the applications on images are various: classification, object detection, semantic segmentation, etc.

Anomaly\Class	Punctual	Contextual	Collective
Phase modulation	X	✓	X
Amplitude modulation	X	✓	✓
Frequency modulation	X	✓	✓
Offset/Drift	X	✓	✓
Noise/White noise	X	X	✓
Abnormal seasonality	X	X	✓
Abnormal pattern	✓	X	✓
Multivariate anomaly	X	✓	X
Dropout	✓	X	✓
Drag	✓	X	✓
Categorical anomaly	✓	✓	X

Class\Monitoring	NPM	PTM	NFM
Punctual	✓	✓	X
Contextual	✓	✓	✓
Collective	✓	X	X

The types of anomaly are defined on a case by case basis during the analysis of the use case.

A.2.5.5 Multi-timescale Online Monitoring Framework

This chapter starts by presenting the Multi-timescale Online Monitoring Framework, continues with design methodological considerations, and finished with key design principles to develop an efficient monitor for your AI-based product.

A.2.5.5.1 What is Multi-timescale Online Monitoring? The context of the online monitoring device is described in the figure A.13. The product incorporating one or more AI models is shown in black. This product can be a component of a system or a system as a whole. The generic term "product" is used in the following. Above the product, in a white oval, the on-line monitoring device receives both the external inputs and outputs of the product, as well as some information about the internal state of the product using pre-designed probes placed in the product's software code or hardware. A device for collecting all relevant operational data is usually required to calculate off-line metrics and to fine-tune some of the on-line monitor parameters. Finally, a controller is responsible for synthesising the output produced by the product and the verdict of the monitor to calculate the final output, which is so-called the "safe output". Consider now the deliberately simplified example in Figure A.13, which represents a product incorporating an AI model that approximates a linear physical phenomenon in the form of an linear function $y = a * x + b$ (solid green segment in Figure A.13).
The principle of multi-scale monitoring is described in Figure A.14. It consists in combining several monitoring time scales: monitoring of the product or system behaviour at the present time, monitoring over a configurable time window of the near past, monitoring over a configurable time window of the near future.
In system engineering analyses, robustness bounds were defined (green dashed segments) that

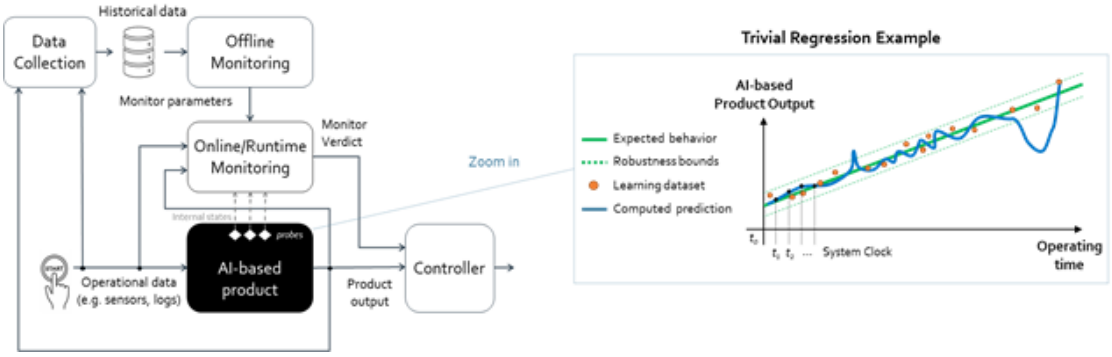

Figure A.13: Context of the AI Model and its online monitoring function

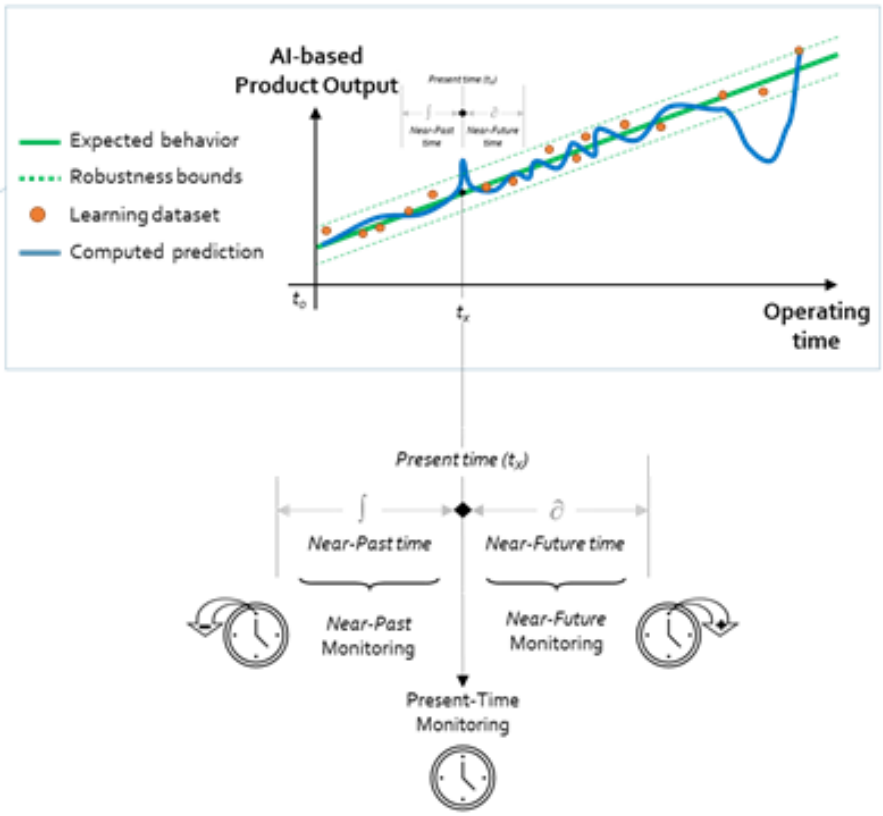

Figure A.14: Principle of the multi-timescale online monitoring

define the validity domain of the product output y. As sufficient data was available to characterise the physical phenomenon, it was decided to use machine learning technology to design the product (e.g. using deep neural network technology). The model obtained after learning is described by the blue curve in Figure A.13 where one can see several operating points of the output y that fall outside the validity domain. The aim of the monitoring device is to detect all operating points that violate the monitoring properties previously defined during the design of the monitoring device. The operation of the product in production is clocked by a system clock tk, which also clocks the monitoring device (at each time tk the device acquires data to produce a verdict). To illustrate the combination of these 3 different time-scale monitoring methods, let

us continue the discussion of the trivial example of Figure A.15, the learning datasets (orange points) have been used to train and verify the regression model (e.g. feed-forward neural network). After several training iterations, the final model (M) is obtained as represented by the blue solid line. It is possible to see in this figure that M has several issues.

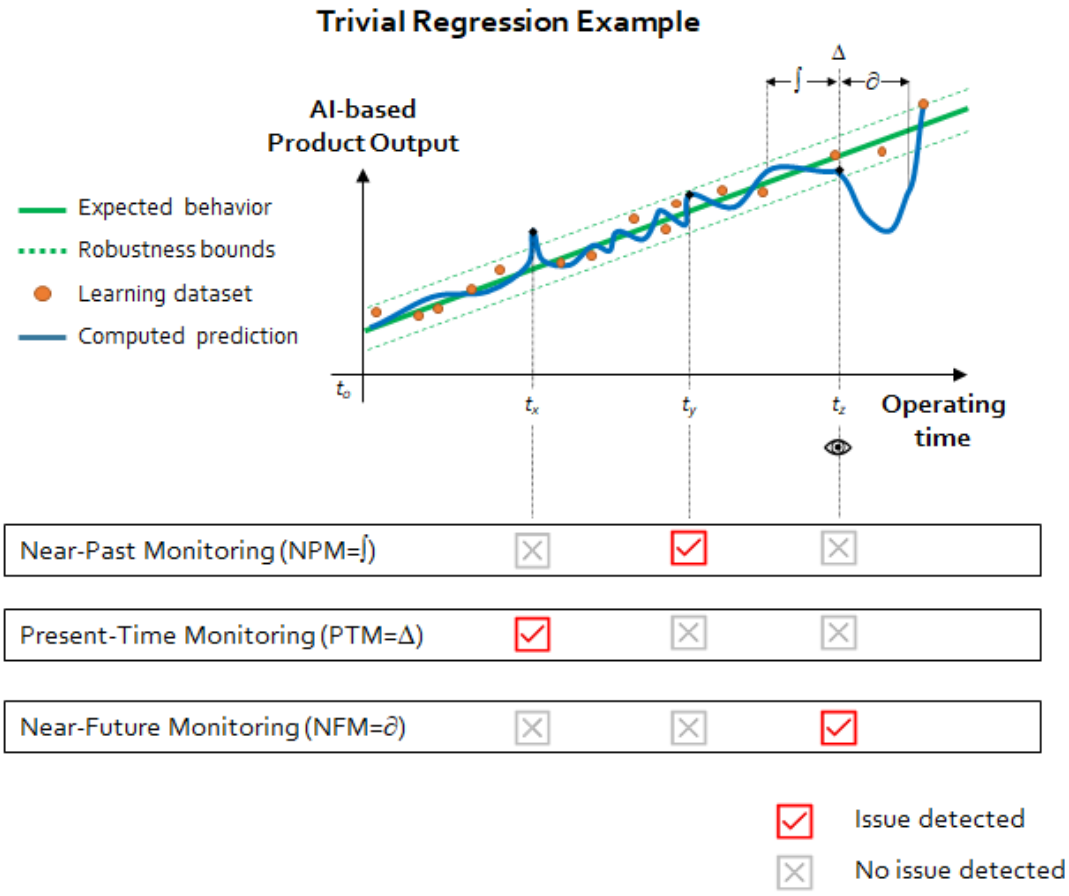

Figure A.15: Trivial example of multi-timescale online monitoring

At t_x, M output overpasses the robustness boundaries, and it is expected that the Present Time Monitoring (PTM) will be able to detect such abnormal behavior. Between t_x and t_y, it is possible to observe that M output starts to unexpectedly oscillate. It is clearly an unintended behavior that could be a precursor of a failure of the model. Since this oscillation phenomenon should be observed and confirmed on several clock cycle, it is expected that the Near Past Monitoring (NPM) will be the appropriate monitoring timescale to detect such anomalies. At t_z, the output of M has a abrupt trend that will make it overpassing the robustness boundaries at the next clock cycle. Here, the Near-Future Monitoring (NFM) is the most appropriate method to detect such abnormal behavior since its is based on trend analysis. Through this didactic example, one can observe that an efficient combination of these 3 different monitoring timescale (NPM, PTM, NFM) allows to detect several classes of anomalies and to achieve by design a high rate of detection of failures that could occur in operational conditions when the AI model is in production and the minimisation of false alarms. The next chapters will present how this multi-timescale monitoring is implemented in practice through different industrial use cases.

A.2.5.5.2 Some Methodological Considerations to Design a Monitor The development of monitors should follow two complementary design methods, one which is top-down and the other bottom-up as depicted in Figure A.16. The top-down approach is based on a traditional system engineering where the intended behavior of the AI-based product is specified in system requirements from CONOPS, Human Factor assessments, and System Safety Assessments. These system requirements are analyzed to characterize the ODD and the high level properties of the AI-based product. From this knowledge of the ODD and high level properties, it is possible to identify the classes of anomalies to be detected by the online monitor. In addition to this top-down approach, a complementary bottom-up design method is necessary to capture the classes of anomalies that are specific to the selected AI technology used to implement the AI model. For example, some AI technologies may be more sensitive to noise or geometric transformation compared to others and it is important to capture these intrinsic technology weaknesses in order to monitor the corresponding anomalies (e.g., monitoring the level of noise, or monitoring the level of translation/rotation of an image).

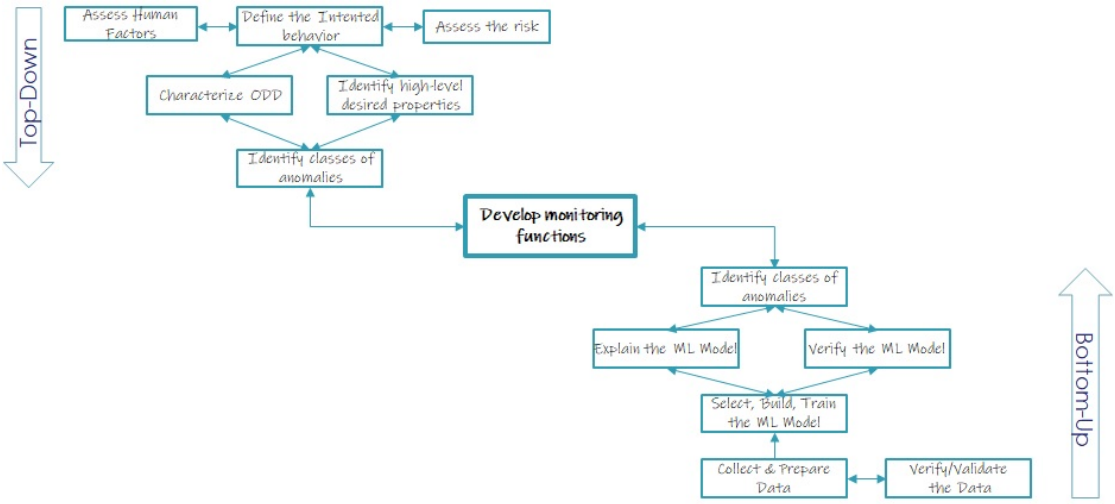

Figure A.16: Top-Down and Bottom-Up Design Methods

A necessary condition to reach a higher anomaly detection completeness of the monitor is to carefully select the monitoring assets: inputs of the AI model, output(s) of the AI model, internal states and variables of the AI model and also the assumptions considered during the design of the AI model, as illustrated in Figure A.17. For example, a current assumption that is taken for machine learning models is the *identically and independently distributed* (i.i.d.) property of the input data. Some verification proofs, like the underlying statistical theory to compute generalization guarantee, will no more hold if this assumption is not verified on the input data received from the sensors at inference time. Therefore, it is important to monitor online that this kind of assumption is still valid or not, and trigger an appropriate recovery mechanism at system level in case of invalid assumption.
The development of PTM, NPM and NFM monitoring functions can be done using different AI technologies: rule-based (or knowledge-based) technology, data-driven technology (Machine Learning (ML)), or hybrid technologies combining rule-based and data-driven. This report details only rule-based methods. It allows the monitored function to be independent of the AI model in terms of technology, specification, design and implementation.

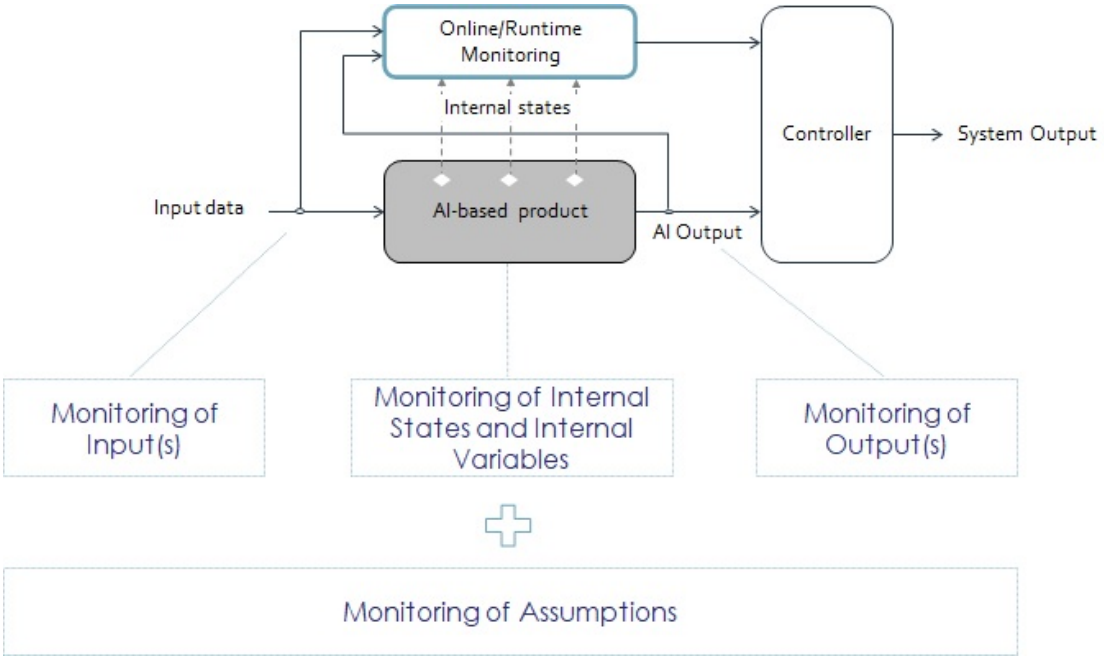

Figure A.17: Multiple Monitoring Assets

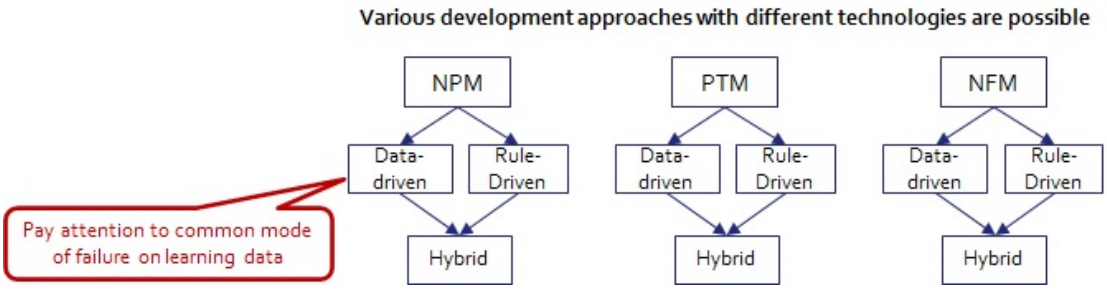

Figure A.18: Various development technologies are possible

A.2.6 Overall vision, context, motivation, objective, added value of the proposed method

A.2.6.1 General context

Artificial Intelligence (AI) technologies are increasingly used in the development of mission-critical and safety-critical products in many application fields such as automotive, aviation, defense, energy, naval, railway, space, etc. AI technologies are diverse, including Machine Learning (ML), Symbolic AI and Hybrid AI . The models designed with these AI technologies are not easily testable and certifiable due to several difficulties well known to specialists in the field. For example, in the case of ML models, the certification challenges are the availability and the quality of the data used for the learning, the stability of the model against small perturbations on input data within the Operational Design Domain (ODD), the robustness of the model against invalid inputs outside the ODD, the ability of the model to provide a correct prediction for data not seen during the design (generalisation capability), the explainability of the model composed of a very large number of internal parameters, the resilience of the model against concept drift, etc. In the industry, it is commonly established that the main objective of online monitoring

of AI models (also called Run Time Assurance in [21] or safety control structure in [25]) is to detect (i) any deviation of the AI component deployed in production from the expected behavior (i.e., intent specified at the system level and allocated to the AI model), and (ii) precursors of the occurrence of failure conditions (i.e., feared events at the system boundaries) based on a predefined set of safety properties. Deploying a monitoring component running in parallel with the AI model is a practical way to manage the risk induced by a model for which it is not possible or feasible to formally demonstrate the achievement of the performance and the safety objectives resulting from the system analyses. Online monitoring is an architectural pattern well-known to safety engineers, but it had to be adapted to AI technologies.

A.2.6.2 Motivation and scope

In an ideal world, the AI model can perform its prediction over its entire Operational Design Domain (ODD) with the expected level of performance (e.g., 99.9 % correct predictions, and this accuracy is maintained over time in operation). However, in practice, if we consider for example machine learning models in many recent papers, most of the time it is very difficult to achieve more than 99 % accuracy, which is an average of one wrong prediction out of 100 inferences in production. But should we hastily conclude that 1 % of bad prediction systematically triggers unexpected behavior leading to a system failure condition? In practice, from the extensive industrial feedback collected in Confiance.ai program and for a wide range of industrial applications, a single error does not directly lead to hazardous or catastrophic events, because the system design has eliminated Single Points Of Failure (SPOF) (e.g., application of the following guidelines ARP4754 Aerospace (2010) and ARP4761 Aerospace (1996) in the aviation, IEC 61508 IEC (2010) in industry, ISO 26261 ISO (2011) in the automotive, etc.). Therefore, based on this assumption (no SPOF in the system), it implies that a single failure of an AI component (i.e., an incorrect prediction at a given time) cannot directly lead to hazardous or catastrophic events. However, what about the case where the AI component has persistent failures (i.e., the model fails to infer the correct prediction at a given point in time, and continues to fail during a subsequent time interval)? This could increase the residual risk due to a higher probability of having (during that time interval where the AI component continues to fail) a combination of multiple internal failures in the system leading to a system failure condition. To detect persistent failures, considering the dynamics of the system and thus the "time" variable is a major issue.

A.2.6.3 Objectives

In this document, the rule-based part of an innovative multi-time scale online monitoring architecture is presented. The main idea is to combine several monitoring timescales - Present-Time Monitoring (PTM), Near-Past Monitoring (NPM), and Near-Future Monitoring (NFM) - on different monitoring assets (inputs, internal states, and outputs of the AI model) to ensure a high anomaly detection rate by design of the online monitor while minimizing the rate of false alarms. Within this multi-timescale monitoring framework, the main objectives are the following:

- Study the Confiance.ai use cases and analyze their ODD
- Identify the properties (e.g., anomaly detection) to monitor
- Design and develop monitoring functions
- Evaluate the monitoring functions
- Integrate the monitoring functions in the Confiance.ai trusted environment

B. Description of the method

B.1. Position in the End-to-End approach

This method can be used in several places of the End to End pipeline.
C.7 “Data Engineering”
C.7.4 Develop data
Given a model trained into several generations to increase its performance, this method can help to identify OOD samples from the datasets used to train, validate and test the generation n+1 of a model generation n. By detecting and cleaning those samples from the training dataset, this should improve the performance of the model. The potential OOD samples in the validation and test datasets could also be annotated and used to improve the model validation. In this scenario, the monitor has already been calibrated on the generation n of the model and data.
C.5 “ML Model monitoring and control: specification, development, implementation”
The rule-based detectors of the monitor should be calibrated after the training, validation and test dataset have been validated and after the model of generation n has been trained.
C.15 “System Validation/Qualification” from AI perspective
Once the monitor has been calibrated and the AI system is in operation, this method aims at monitoring the AI system to analyze and detect anomalies on the input data, on the internal states and variables of the model and/or on the outputs of the model. The robustness and stability of the input data and of the model predictions can also be monitored.

B.2. Prerequisites

This method of rule-based monitoring requires the output of the method MOODD (Model ODD Characterization) in order to calibrate the detectors on the model that will be monitored. The method is generic for any type of problem, data and models, as long as a model has been trained and is capable of inference on this data.

B.3. Level of maturity

This method has been implemented and tested on Confiance use cases of image classification, object detection, time series regression and time series classification. The results on those use cases have been evaluated by the LNE, the National Laboratory of Metrology and Testing.

B.4. Generic monitor calibration Guidelines

Most monitor functions are calibrated with a generic method.
This generic method has 2 prerequisites:

- The anomaly should have intensities sorted in a way that they can be represented as a decreasing or increasing curve.
- The detector on a dataset augmented with these sorted anomalies should give values represented as a decreasing or increasing curve.

An example of image anomaly that didn't fulfill those prerequisites during our experiments was Color anomaly. An example of datatype that doesn't seem to fullfill those prerequisites is NLP. Some types of NLP anomalies would be hard to scale in intensity as a small change in a sentence can easily change the whole meaning. In reality, different anomalies are often mixed together when human make mistakes while writing. This is quite challenging to apply this generic method on such combinations. Other methods of rule-based calibration and detection should be explored for NLP.
This method contains 3 steps as shown in Fig. B.1:

- Given an anomaly (ODD parameter) A, a supervised model M, and samples of a training dataset S, the Model ODD should first be characterized for A using M predictions on augmented dataset S'. This MOODD method will output a curve C1 representing the confiance on the prediction of the model M for the degrees of anomaly A.
- The detection function F to detect the anomaly A is run on the augmented dataset S'. The ouput is a curve C2 of detected median values per anomaly levels.
- Basic detection thresholds can be computed by interpolating ODD zone delimiters on C1 and detected values on C2 for these delimiters. For each anomaly, an value of uncertainty for the thresholds can be arbitrary chosen. Those thresholds and uncertainties will then be used during anomaly detection to trigger or not alarms. Rather than using thresholds, a more advanced method is to compute uncertainty ranges directly on C2, which gives C2' and interpolating C1 and C2' in the detection functions. This method is described in Fig. B.33

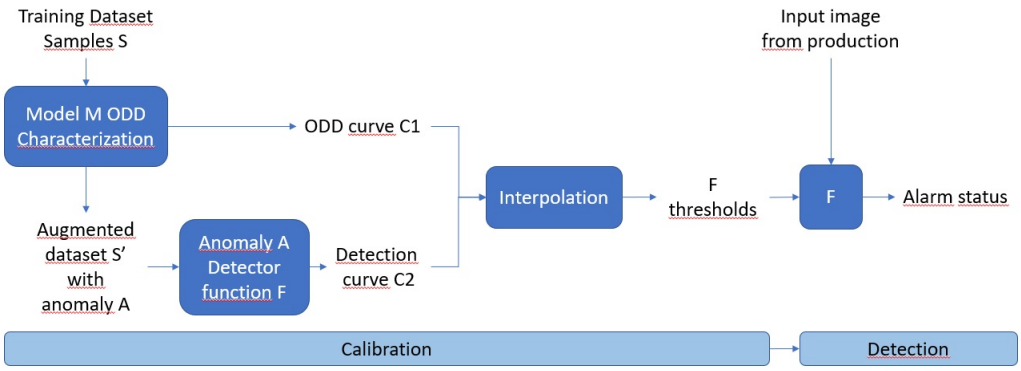

Figure B.1: Detection calibration

Fig. B.2 shows an example of curves C1 (on the left and C2 (on the right) for the ODD parameter Brightness. On C1, the ODD zone is starting at intensity 23.58 and ending at intensity 328.68. On C2, interpolating those values on the perceived brightness medians gives the calibrated thresholds starting at 10.75 and ending at 138.25. The perceived brightness detected on an input image from production will then be compared with those thresholds and an arbitrary

uncertainty range to give an alarm, no alarm or potential alarm status.

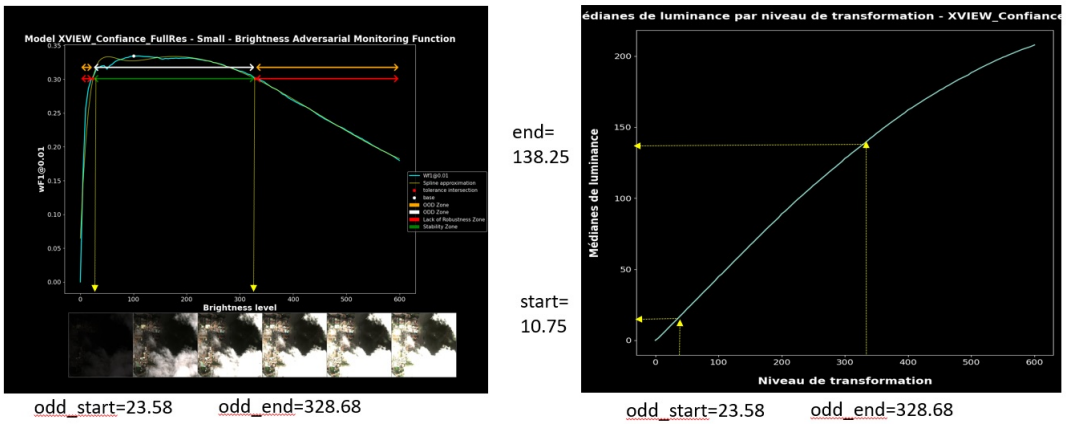

Figure B.2: Curves used to interpolate Detection thresholds

B.5. Black-Box Images Guidelines

B.5.1 Basic monitoring functions

B.5.1.1 Standard Defocus (Out of focus) Blur Detection

In optics, defocus is the aberration in which an image is simply out of focus. In general, defocus reduces the sharpness and contrast of the image. What should be sharp, high-contrast edges in a scene become gradual transitions. Fine detail in the scene is blurred or even becomes invisible. When the degree of defocus blur in an area of interest of an image can impact the prediction of a model on this image, the monitor should raise an alarm about it.

B.5.1.1.1 Objective

This function aims at extracting the degree of defocus blur from an image and raising an alarm if this degree of blur can impact the prediction of a model on this image.

B.5.1.1.2 Inputs and outputs

The function takes in input the image, and parameters calibrated for the category of this image: the area of interest and the value of reference. It returns as output the defocus blur status; whether there is an alarm, no alarm or a potential alarm.

B.5.1.1.3 Monitoring class

Rules-based Present-Time Monitoring (PTM)
Subclass: Out of Distribution Monitoring (ODM)
Monitoring type: Monitoring of the AI Model Input

B.5.1.1.4 Design principles

- The Area of Interest is extracted from the image.
- The Area of Interest is converted to gray.
- Variance of Laplacian is computed on the color values of this gray Area of Interest.
- This variance is compared with the value of reference tuned for this category of images
- Compute an estimation of the degree of blur: batch 2

B.5.1.1.5 Detailed design

A measure of the defocus blur in an image can be approximated by computing the variance (the spread of the distribution) of the high-contrast edges in the image. Laplacian is often used on digital images to detect such edges. In mathematics, the Laplace operator or Laplacian is a differential operator given by the divergence of the gradient of a scalar function on Euclidean space. Considering an image in 2 dimensions I(x,y), the Laplacian would be the sum of second-order partial derivatives of its function:

$$Laplacian(x,y) = \frac{\partial^2 I(x,y)}{\partial^2 x} + \frac{\partial^2 I(x,y)}{\partial^2 y}$$

In image processing, the input image isn't a continuous function but a discrete function. The Laplacian can still be approximated by filtering the image with a 3×3 kernel. In digital images there are essentially two ways to look at the neighborhood of a given pixel:

- Moore neighborhood based on Chebyshef distance: B.3 .
- Von Neumann neighborhood based on Manhattan distance: B.4.

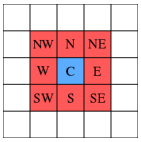

Figure B.3: Moore

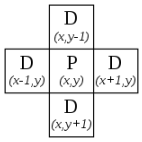

Figure B.4: Von Neumann

Moore neighborhood gives the same weight to all eight surrounding pixels. Von Neumann neighborhood considers that the North, East, South, West gray pixels are closer (distance 1) than the ones in the corners (distance $\sqrt{2}$). On GPU, vector operations are usually faster between small integers. The 3x3 convolution kernel for Laplacian should use the smallest integers and the sum of its elements should be 0. There is 1 such kernel for Moore neighborhood:

$$\begin{bmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{bmatrix}$$

And 2 kernels for Von Neumann neighborhood:

$$\begin{bmatrix} 0 & -1 & 0 \\ -1 & 4 & -1 \\ 0 & -1 & 0 \end{bmatrix} \begin{bmatrix} 1 & -2 & 1 \\ -2 & 4 & -2 \\ 1 & -2 & 1 \end{bmatrix}$$

The method Laplacian() from OpenCV called in the monitor is using this kernel by default:

$$\begin{bmatrix} 0 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & 0 \end{bmatrix}$$

The image is first converted to gray levels to increase the contrast before applying Laplacian as shown in figure B.5.
The variance of the Laplacian is a measure of the spread of its distribution. The less high-contrast edges are given by Laplacian, the smaller is the variance. In a sharp image, there are many high-contrast edges so the variance is high. In a blurry image the edges are smoothed so the variance is little. The Laplacian variance measured in an image is compared to a threshold to determine if the degree of defocus blur can impact the prediction of a model.

B.5.1.1.6 Tuning

The tuning step's consists in the identification of the detection thresholds. The tuning method described in figure B.6 is applicable for other monitoring functions presented in this report. Tuning is performed offline. It has an automatic phase of detection within a list of values of reference to quantify the blur of each images. Then an automatic phase of identification of the blur threshold value corresponding to the minimal measure under which the model does not infer correctly the incoming image. Before saving the threshold to the online configuration, an optional manual validation can be performed on a specific dataset to fine-tune the threshold and ensure a satisfying answer of the model.

B.5.1.1.7 Usage recommendations or limitations

The first value of reference comes from the analysis of:

- the Robustness to Gaussian Blur transformations
- the distance between the camera and the object
- the focal ratio of the camera

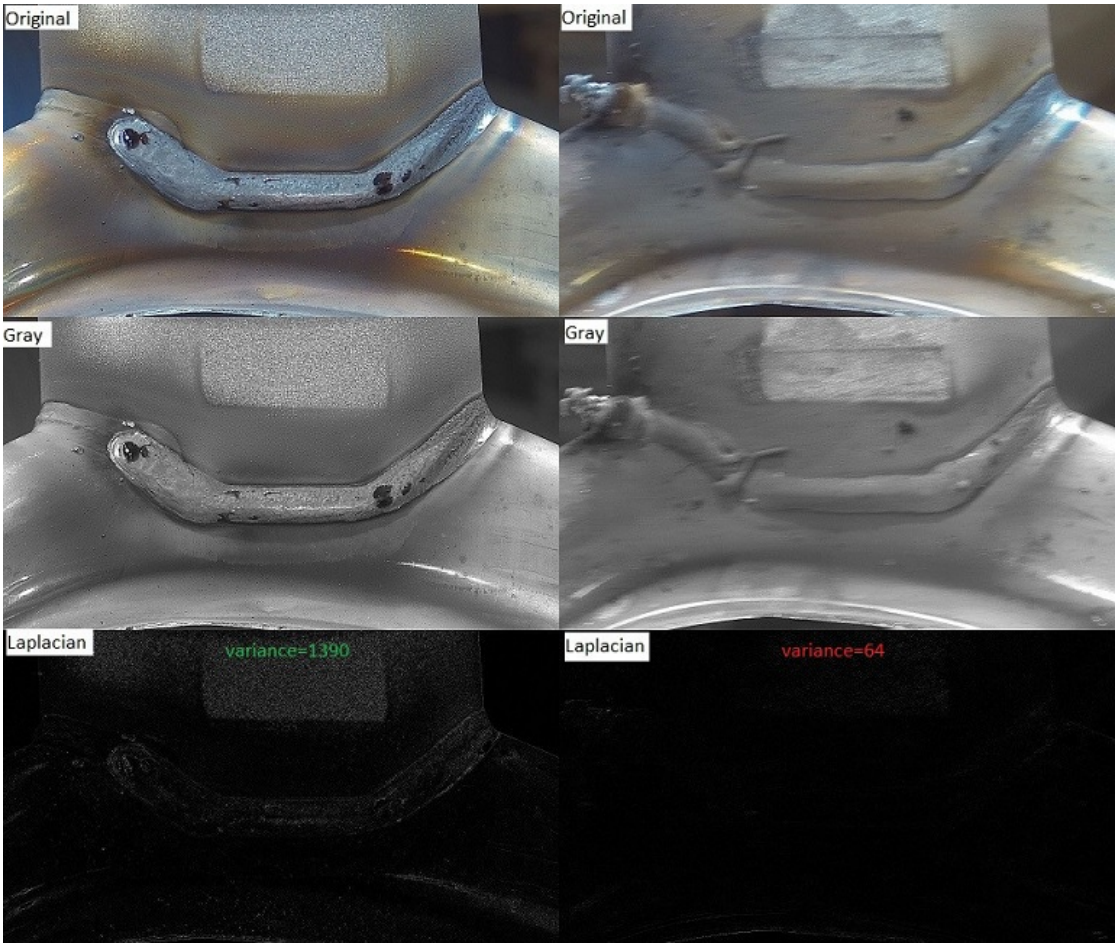

Figure B.5: Sharp (left) and Blurry (right) comparison in Renault Welding UC

- Graphs of distribution of defocus blur measured by category of images

B.5.1.2 Standard Motion Blur Detection

Motion blur is the apparent streaking of moving objects in a photograph. It results when the image being recorded changes during the recording of a single exposure, due to objects and/or camera movement or long exposure.

B.5.1.2.1 Objective

This function aims at extracting the degree of motion blur from an image and raising an alarm if this degree of blur can impact the prediction of a model on this image.

B.5.1.2.2 Inputs and outputs

The function takes in input the image, and parameters calibrated for the category of this image: the area of interest and the values of reference. It returns as output the motion blur status; whether there is an alarm, no alarm or a potential alarm.

B.5.1.2.3 Monitoring class

Rules-based Present-Time Monitoring (PTM)

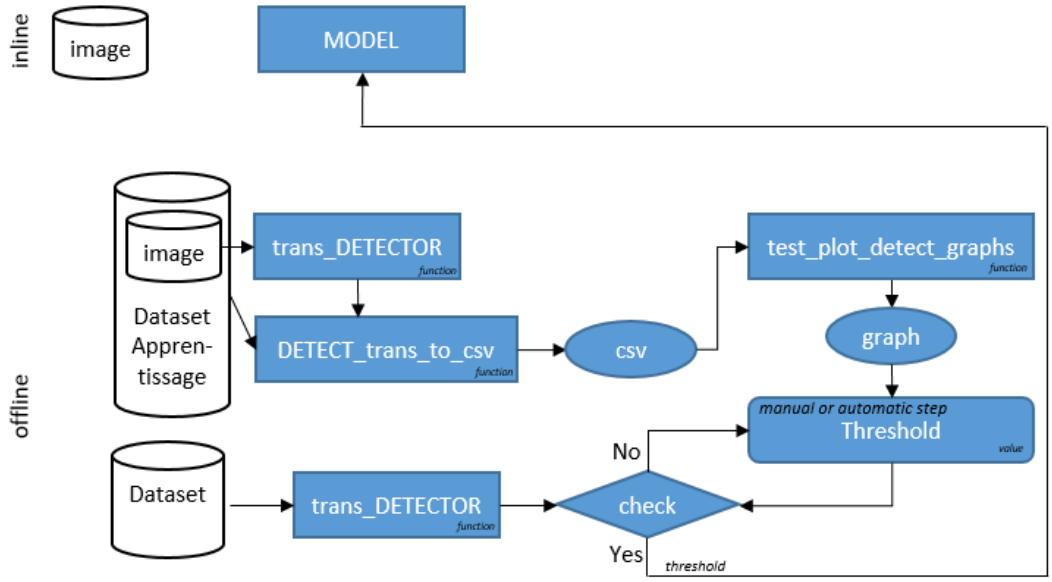

Figure B.6: Generic workflow of detection functions tuning

Subclass: Out of Distribution Monitoring (ODM)
Monitoring type: Monitoring of the AI Model Input

B.5.1.2.4 Design principles

Design principles are represented on figure B.7.

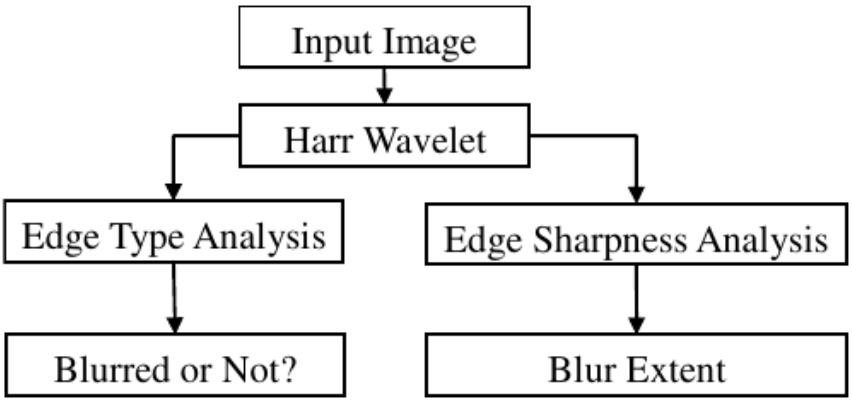

Figure B.7: Motion Blur Detector Scheme

B.5.1.2.5 Detailed design

The latest approach to detect motion blur is the analysis of edges type and sharpness. Edges are usually classified into three types: Dirac-Structure, Step-Structure and Roof-Structure In this approach Step-Structure are classified into Astep-Structure and Gstep-Structure according

to whether the change of intensity is gradual or not. Edges types are illustrated in Figure B.8 .

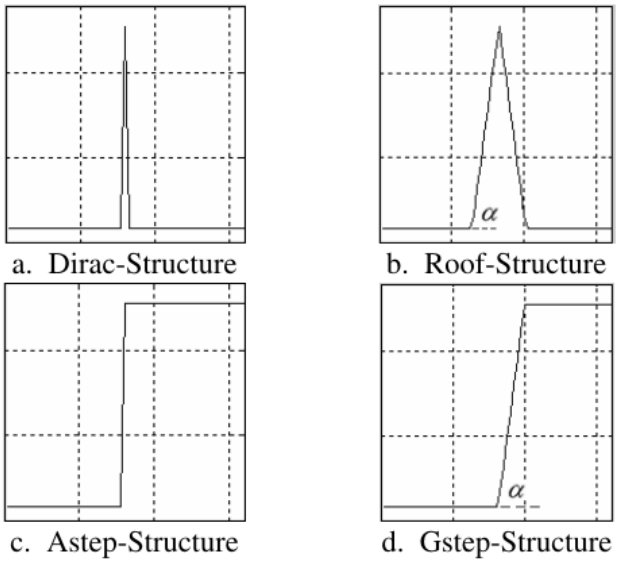

Figure B.8: Edge Types

Most original images contain all types of edges and most Gstep-Structure and Roof-Structure are sharp. When defocus or motion blur occurs, both Dirac-Structure and Astep-Structure will disappear and both Gstep-Structure and Roof-Structure tend to lose their sharpness. A given image is detected as blurred by the absence of Dirac-Structure or Astep-Structure, and the percentage of Gstep-Structure and Roof-Structure determine the blur extent (or blur degree).

Harr wavelet transform is an efficient algorithm to detect edges and thus motion blur in an image Tong et al. (2004). When this algorithm is performed on an image it returns the list of all edges. Then a set of rules is used to classify edges by types. The ratio of Dirac-Structure and Astep-Structure edges among all edges is compared to a threshold to determine if the image is blurred. Then another set of rules is used to measure the sharpness of Gstep-Structure and Roof-Structure and return the blur degree. This blur degree is compared to another threshold to determine if it can impact the prediction of a model.

B.5.1.2.6 Tuning The blur motion tuning uses the same method described in B.5.1.1.6. The threshold corresponds to the measure limit of the blur motion confiance.

B.5.1.2.7 Usage recommendations or limitations

The first value of reference comes from the analysis of:

- the Robustness to Motion Blur transformations
- the average speed between the camera and the object
- Graphs of distribution of motion blur measured by category of images

B.5.1.3 Standard Brightness Detection

B.5.1.3.1 Objective

This function aims at extracting the degree of brightness from an image and raising an alarm if

this degree of brightness can impact the prediction of the model on this image.

B.5.1.3.2 Inputs and outputs

The function takes in input the image and two parameters calibrated for the category of this image: the area of interest and the value of reference. It returns as output the brightness status; whether there is an alarm, no alarm or a potential alarm.

B.5.1.3.3 Monitoring class

Rules-based Present-Time Monitoring (PTM)
Subclass: Out of Distribution Monitoring (ODM)
Monitoring type: Monitoring of the AI Model Input

B.5.1.3.4 Design principles

- The Area of Interest is extracted from the image.
- A formula of perceived brightness is applied on the mean of RGB values of the Area of Interest
- This perceived brightness is compared with the values min and max of reference tuned for this category of images
- Compute an estimation of the degree of brightness: batch 2

B.5.1.3.5 Detailed design

In video and photography, luma represents the brightness in an image. It is the weighted sum of gamma-compressed R'G'B' components of an image. For digital formats following CCIR 601 (i.e. most digital standard definition formats), luma is calculated with this formula:

$$luma = \sqrt{(0.299 \times mean(R)^2 + 0.587 \times mean(G)^2 + 0.114 \times mean(B)^2)} \quad \text{(B.1)}$$

Formats following ITU-R Recommendation BT.709 use a different formula:

$$luma = \sqrt{(0.2126 \times mean(R)^2 + 0.7152 \times mean(G)^2 + 0.0722 \times mean(B)^2)} \quad \text{(B.2)}$$

One may also use the SMPTE 240M coefficients:

$$luma = \sqrt{(0.212 \times mean(R)^2 + 0.701 \times mean(G)^2 + 0.087 \times mean(B)^2)} \quad \text{(B.3)}$$

The first equation has been implemented and tested in batch1. The two other equations or other custom equations could be added in batch3 and enabled through the monitor configuration.

B.5.1.3.6 Tuning

The tuning of this function consists in the definition of the minimum and maximum measures of perceived brightness which permit an acceptable inference of the image by the model. The method to obtain them is the same than the one detailed in B.5.1.1.6. Graphs displayed the brightness in function of the response from the model on a leaning dataset of images in order to identify the minimum and maximum value.

B.5.1.3.7 Usage recommendations or limitations

The first value of reference comes from the analysis of:

- the Robustness to brightness transformations
- Graphs of distribution of brightness by category of images

B.5.1.4 Standard Color Detection

Robustness analysis on Renault Welding UC pointed out that a color filter applied to the whole (or a part of) area of interest of an image impacts the prediction of the model on this image if this color was not part of dominant colors in the model training dataset.
No existing method has been found during our research that could detect if colors of an input image are close to the colors in a dataset. We propose the method described in the following paragraphs after having explored and tested it on Renault Welding UC.

B.5.1.4.1 Objective

This function aims at extracting the dominant colors from an image and raising an alarm if those dominant colors can impact the prediction of the model on this image.

B.5.1.4.2 Inputs and outputs

The function takes in input the image, and two parameters calibrated for the category of this image: the area of interest and the dominant color values of reference. It returns as output the color status; whether there is an alarm, no alarm or a potential alarm.

B.5.1.4.3 Monitoring class

Rules-based Present-Time Monitoring (PTM)
Subclass: Out of Distribution Monitoring (ODM)
Monitoring type: Monitoring of the AI Model Input

B.5.1.4.4 Design principles

- The Area of Interest is extracted from the image.
- The dominant colors are extracted from this Area of Interest.
- Each dominant color is checked against the values of reference.
- If the RGB point is contained (or close enough) in the 3D clouds (approximated by a convex hull) of the RGB points of reference, the color is accepted.
- A status is given based on the number of dominant colors accepted.

B.5.1.4.5 Detailed design

The RGB values of each pixel are extracted from the image interest area. Similar colors are regrouped following a tolerance T based on a scale from 0 to 100 where 0 won't group any color and 100 will group all colors into one. The smaller the distribution of colors is in the interest area, the lower this tolerance should be. The colors are then sorted from highest number of pixels to lowest. The N first colors are called dominant colors. Colors extracted after this limit N are ignored. The more different colors an interest area can have, the higher this limit should be. Those dominant colors are then compared with dominant colors of reference.
Dominant colors of reference are created during the calibration phase on the offline monitor for each category of image. For Renault Welding UC, the labelled dataset used to train the model

has been used as reference. Each category and state (conform or defective) of welding is a category of image. The distinction between states comes from the fact that conform weldings are usually gray and defective weldings have a large panel of additional colors such as yellow, orange, brown and black.
The first step of color calibration on a category of image is to extract the N dominant colors of all images from this category. Each dominant color $i \in \{1, N\}$ can be represented in a 3D space where (x, y, z) axes represent (R, G, B). Figure B.9 shows this step on Renault welding of type C33 and state conform (OK) for the first dominant color. The first dominant colors of the 1224 image of welding C33 OK are represented in the 3D plan. 4 of those 1224 images are shown on the left of the figure with a yellow rectangle delimiting the area of interest. A yellow arrow is linking each image to the RGB point of the first dominant color extracted from the area of interest.

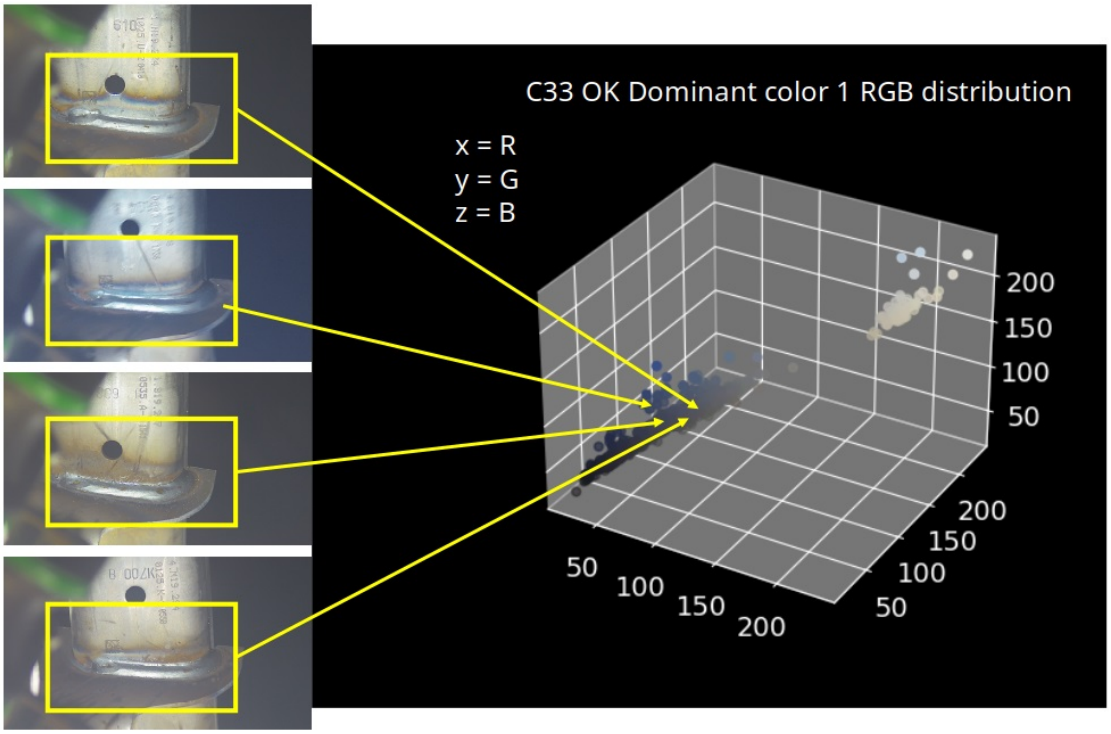

Figure B.9: Reference dominant colors extraction (i=1)

Next step is to group colors points closer than a color maximum distance together. Any group with less points than a minimum is filtered out. Figure B.10 shows in red circle some points that are filtered out with a maximum points distance of 30 and a minimum points per group of 5.
The third step is the calculation of the surface of each group of colors remaining after filtering. In geometry, the minimum surface enveloping a set of 3D points is called a convex hull. Qhull is a C++ library (with python bindings) which implements the Quickhull algorithm for computing the convex hull of a finite set of points in n-dimensional space. It uses a divide and conquer approach. The result of the algorithm on previous step gives Figure B.11.
The last step inflates each vertex of the convex hull on every axe (r, g, b, -r, -g, -b) by an inflation tolerance distance (ITD) as seen in Figure B.12. Those inflated convex hulls become the values of reference for detection.
Once those steps have been executed for N dominant colors on each category of image on offline

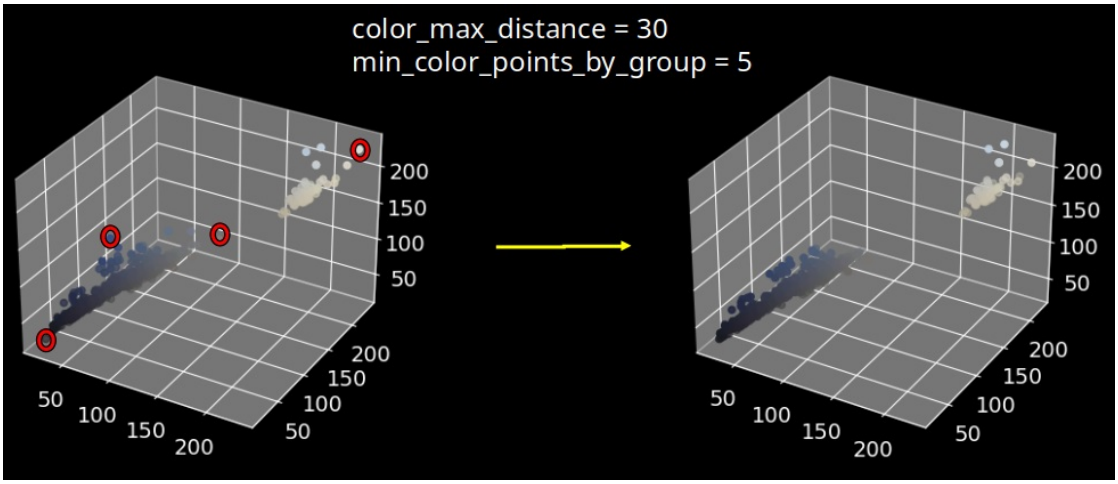

Figure B.10: Reference dominant colors filtering (i=1)

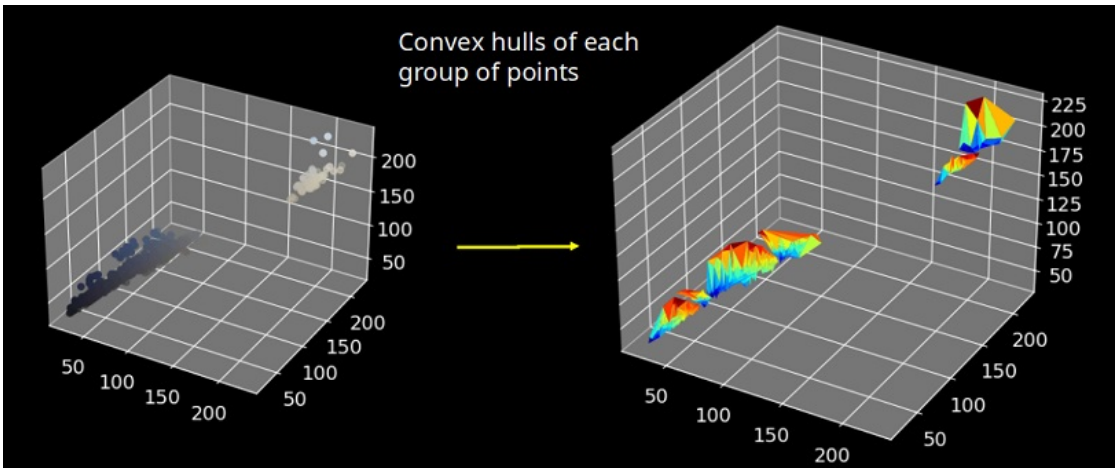

Figure B.11: Reference dominant colors convex hull (i=1)

monitor, the values of references are then loaded to online monitor for detection.
The detection function takes as input an image and the convex hulls for the N dominant colors of the image category. The N dominant colors are extracted from the image as RGBs. For i from 1 to N, vertices of the reference convex hull are compared with the vertices of the reference convex hull recomputed after adding the RGB point of the image dominant color i. If the reference and recomputed convex hulls have the same vertices, it means that the RGB point was inside or on the surface of the reference convex hull and no color alarm is raised for the dominant color i. Otherwise, it means that the RGB point was outside of the reference convex hull and a color alarm is raised for i. If an image has only 1 color extracted, an alarm is raised because the image is too uniform; an object couldn't be interpreted by the model.

Figure B.13 details a detection made on an image from Renault Welding UC by comparing N=4 dominant colors. The red arrow shows the RGB point of the first dominant color. It is outside the reference convex hull and therefore an alarm is raised for color 1. Even if the three other dominant colors are fine, a global color alarm is raised by the function.

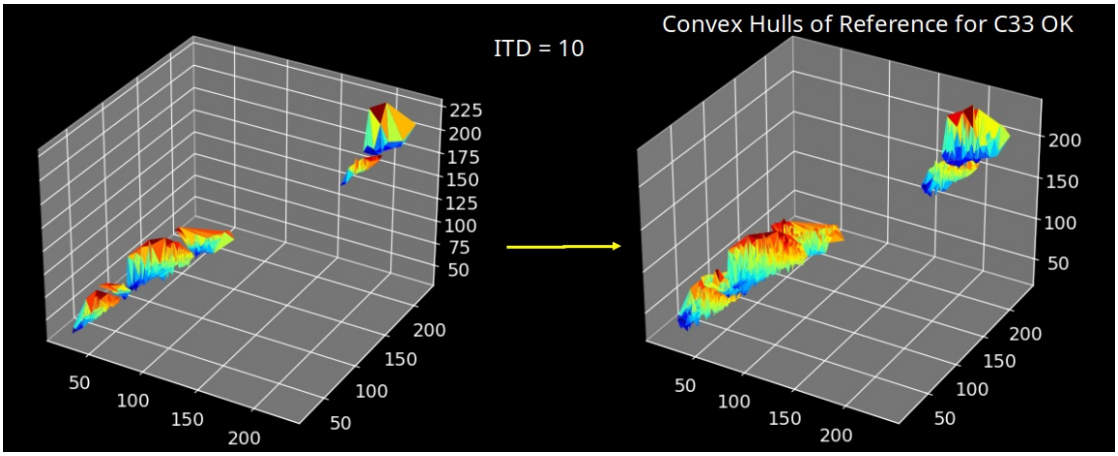

Figure B.12: Reference dominant colors convex hull inflation (i=1)

B.5.1.4.6 Tuning

Tuning consists in the optimisation of the convex hull of reference taking into consideration the number of points and their distance to be considered in the same area. Theses areas are tested on images classified outside the reference area to control if alarm is raised by the monitoring function.

B.5.1.4.7 Usage recommendations or limitations

The first value of reference and detection rules comes from the analysis of:

- The Use Case
- The average number of objects in each category of images
- The Robustness to color filters transformations
- Graphs of distribution of color by category of images

The images should have at least 2 distinct colors.

B.5.1.5 Standard Image Rotation Detection & Image Translation Detection (IRD & ITD)

B.5.1.5.1 Objective The following functions expect a known inspected object to be present at a relatively fixed location in an image. The rotation detection function aims at estimating the rotation angle of the inspected object and raising an alarm if this degree of rotation can impact the prediction of the model on this image. The horizontal and vertical translation detection functions aim at estimating the translation of the inspected object and raising an alarm if this degree of translation can impact the prediction of the model on this image.

B.5.1.5.2 Inputs and outputs The function takes in input the image, and two parameters calibrated for the category of this image: the area of interest, the optimal grid values and the optimal detection curve. It returns as output the geometric status; whether there is an alarm, no alarm or a potential alarm.

B.5.1.5.3 Monitoring class Rules-based Present-Time Monitoring (PTM)
Subclass: Out of Distribution Monitoring (ODM)

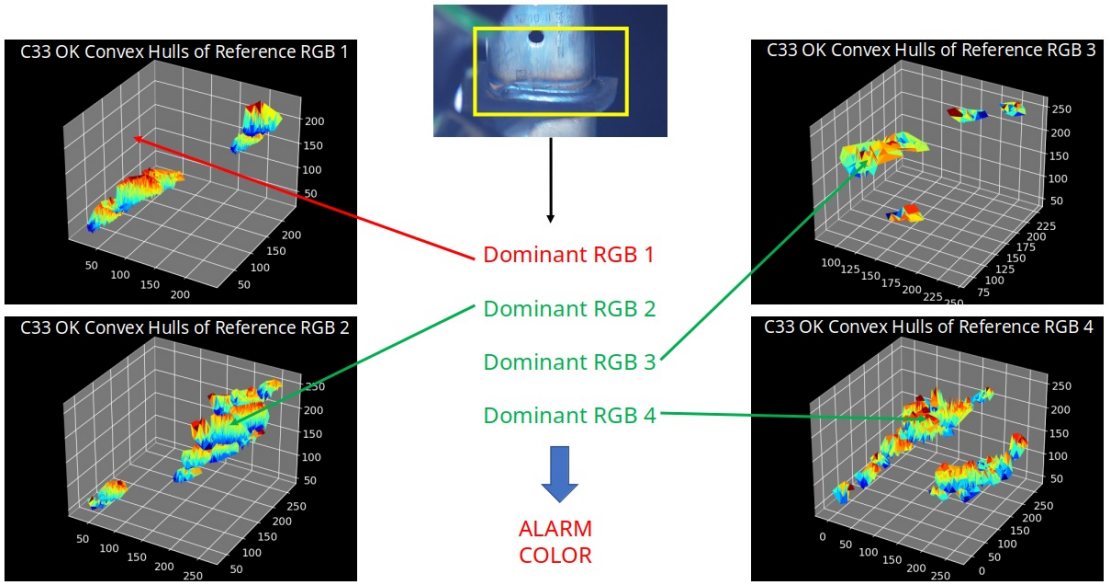

Figure B.13: Color detection

Monitoring type: Monitoring of the AI Model Input
Problem Type: Image classification

B.5.1.5.4 Design principles

- The Area of Interest is extracted from the image.
- The Area of Interest is resized.
- The resized image is converted to gray.
- The grayed image is denoised by applying Gaussian Blur.
- The contrast in this denoised image is increased.
- This contrasted image is cut into cells following an optimal grid for this image category.
- The dominant colors are extracted from each pixel of each cell.
- The percentage of black pixels is computed for each cell.
- This percentage is used to interpolate the transformation level on witness cells based on a calibrated function.
- Each witness cell gives a status of alarm.
- A majority vote is made between all witness cells to decide the final status.

B.5.1.5.5 Detailed design The first step of the detection is to transform the image (PREFORM) before extracting the color values of its pixels. The detection is limited to an area of interest where the object of interest is supposed to be. This area of interest is configured during the calibration of the detectors. Then the image is proportionaly resized to a preconfigured base. The default base is 128 pixels. The higher value between width and height of the image will take the value of this base, and the lower value will be rescaled proportionaly. This resized image is converted to gray levels to first improve the contrast, then it is denoised by applying a Gaussian Blur filter. The last step consists in binarization of the image through an adaptive thresholding. An input image from Renault Welding Use Case and its PREFORM image are shown in Figure B.14 .
Those PREFORM steps aim at delimiting the objects in the image. But the object of interest

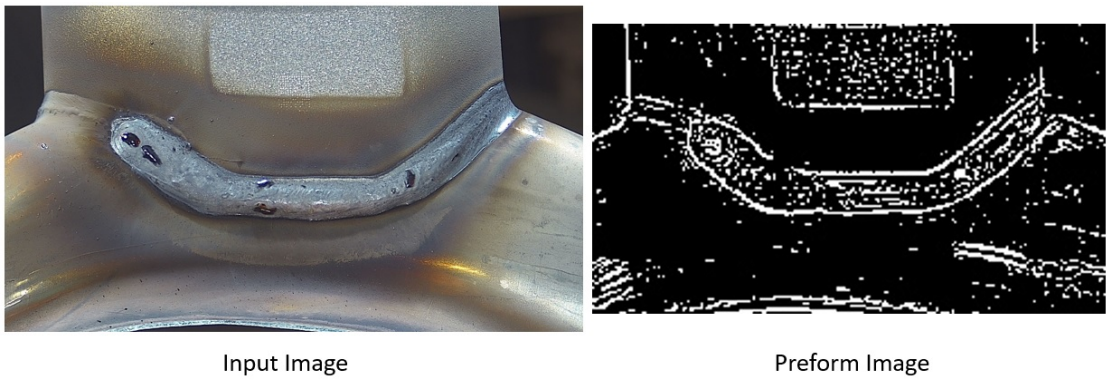

Figure B.14: Rotation Detection PREFORM

is not separated enough to allow the precise coordinates of its countours to be known by this rule based techniques in a generic manner. The proposed approach is to compare the percentage of black pixels in cells of this PREFORM image with calibrated data to interpolate geometric anomalies such as rotation, horizontal and vertical translations. The size and location of those cells are calibrated per object of interest. Which means each class of object to be detected by the image classification model is calibrated and has its own configuration. For Renault Use Case it translates into one calibration configuration per type of welding. The RGB values of pixels in the cells of interest are extracted and the percentage of black pixels if computed. This percentage is given as input of an interpolation function whose parameters have been calibrated. This function returns the estimated translation level. This level is compared to thresholds to give an alarm status. When all cells of interest (also called witness areas) have been processed, a majority vote is made between these witness areas to decide the final status. This majority vote is shown in Figure B.15.

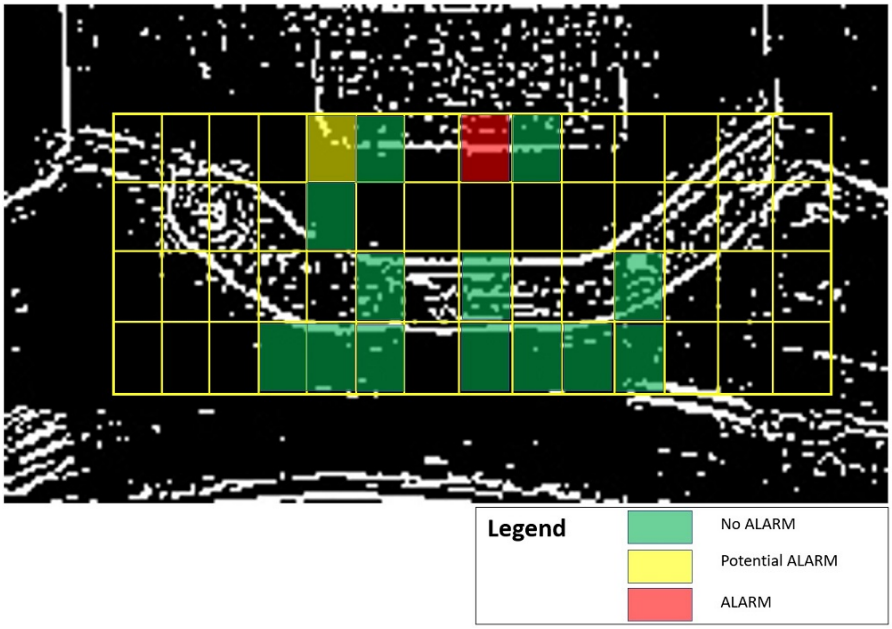

Figure B.15: Rotation Detection Majority Vote

B.5.1.5.6 Tuning The first thing to configure is the area of interest. This is the smallest squared area in the image where the object to classify is most likely to be find. This optional parameter is intended to optimize computation time and improve the precision during the detection, but can also be left empty to let the detectors scan the whole image if optimization is not an issue, or if the object has almost the same size than the image.
Then the calibration of geometric detectors follows 4 major steps:

- Prepare calibration data
- Compute calibration metrics
- Calibrate detection methods
- Compute calibration detection metrics

B.5.1.5.6.1 Prepare calibration data This step requires the outputs of the characterization of the Model ODD for geometric transformations on a subset of N images. It also require a configuration to tune the grid parameters with the ranges of numbers of lines and columns that will cut the interest area into grid cells, and a range of transformation levels. It will run the PREFORM steps as detailed in the first part of B.5.1.5.5 on N transformed images for each level of transformation and for each grid configuration. Figure B.16 shows the PREFORM outputs of two images of transformation level 0 and 20 degrees. Their interest area is being cut into a grid of 5 lines and 11 columns. Then the RGB values of pixels in each cell of the grid are extracted and the percentage of black pixels is computed. The percentages of black are stored by level of transformation and by grid configuration. When all the results are available, the mean of the black percentages is computed on the N images per level of transformation and per grid configuration. These averaged results by transformation level are used in the next step to compute metrics.

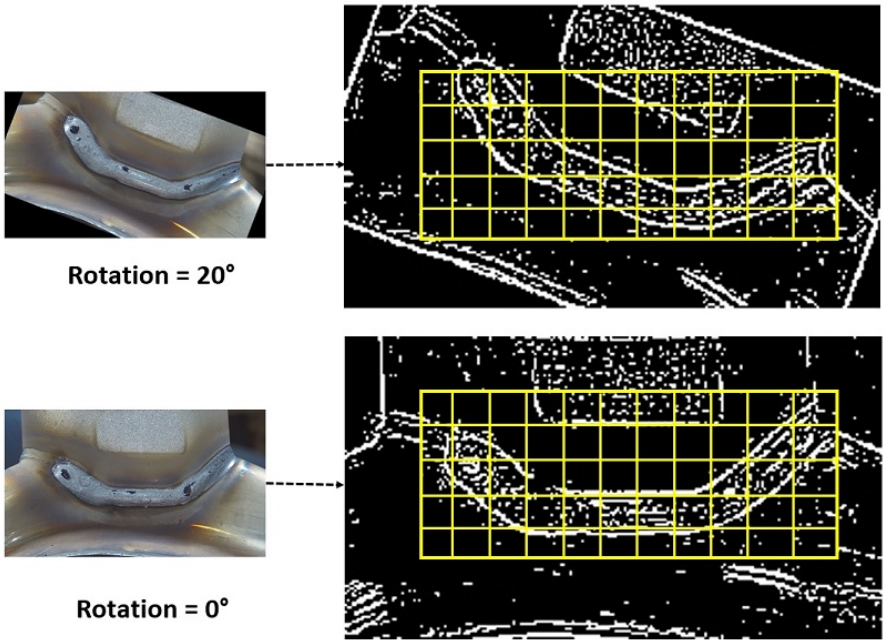

Figure B.16: Prepare calibration data for rotation

B.5.1.5.6.2 Compute calibration metrics This step requires a tuning configuration with a range of desired coefficient of determination (r^2) and a range of resizing values in pixels. A tuning loop is executed on the previous averaged results by looping the grid configurations, the r^2 and the resizing values. Each step of the loop is interpolating the averaged black percentages of pixels by transformation levels per grid cell into a polynomial function of third order. A polynomial r^2 in computed between the polynome and the data. Then a linear regression and its r^2 are calculated from this polynomial interpolation. These polynomial and linear r^2 are compared with the tuned desired r^2 to decide whether the grid cell is a candidate to detect the transformation level. If the comparisons and tests give good results, then the cell becomes a witness area and its coordinates within the grid are saved. Figure B.17 zooms on 4 cells in a tuning grid for rotation. The black curves are the averaged black percentages of pixels by transformation level. The red curves are the polynomial interpolations. The interpolations in cells X1,Y5 and X1,Y6 have been selected for being monotonic curves with a r^2 of 0.99, and in cell X2,Y5 for being a bell curve centered in the rotation identity transformation level equal to 0 degree. The transformation levels were tuned in this example from -20 to 20 degrees.

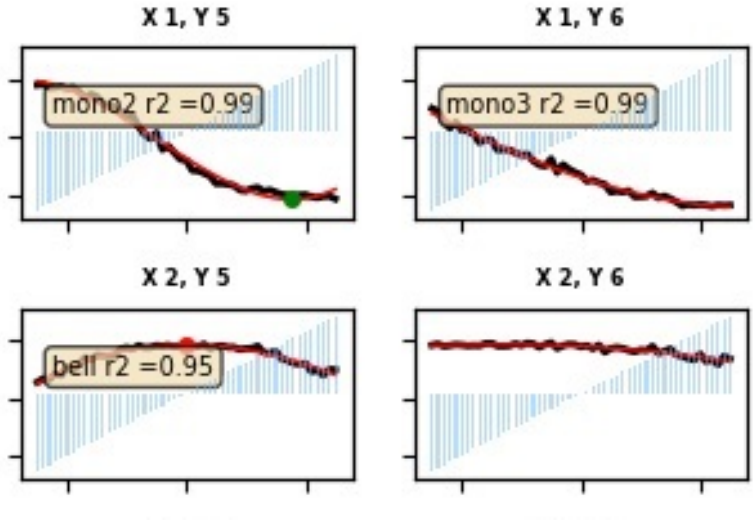

Figure B.17: Examples of candidates to witness areas

B.5.1.5.6.3 Calibrate detection methods A validation dataset V of Y images is first created by randomly selecting a subset A of Y/2 images from a given dataset (the test or the validation dataset for instance) and a subset B of Y/2 images from transformed images during Model ODD Characterization. The subset A is supposed to have no, or very few anomalies and are auto labelled as having the transformation level of the identity, 0 degree for rotation, 0 pixel for horizontal or vertical translations. The subset B is labelled by transformation level.
This validation dataset V is then detected several times by the function described in B.5.1.5.5 while varying the following parameters: the grid configuration, the r^2 and the resize values. For each step of this tuning loop, the witness areas (and their interpolations of black percentages) corresponding to the step parameters are used to give a status of anomaly on each image of the dataset V. Precision, rappel, fmesure, detection rate (the rate of true alarm statuses) and standard uncertainty are computed at each step. The optimal parameters are those who give the maximum fmesure at the end of the loop. The witness areas of the optimal parameters on a calibration run for Rotation on UC Renault Welding C27.2 are shown in green in Figure B.18. This optimal configuration is saved to be used by the geometric detector, as seen in Figure B.15, the grid configuration is the same, and the colored witness areas have the same coordinates in both Figures.

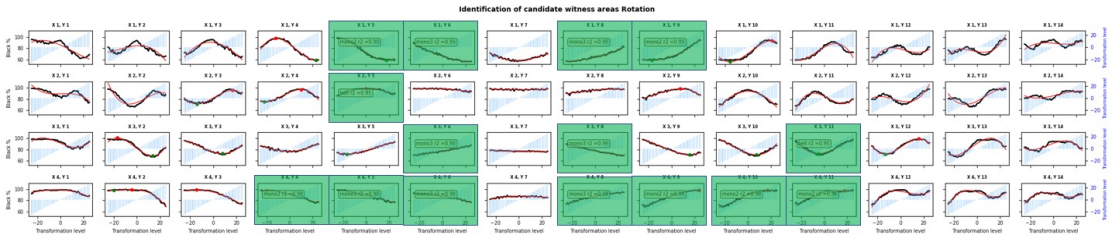

Figure B.18: UC Renault Welding C27.2 optimal witness areas for Rotation

B.5.1.5.6.4 Compute calibration detection metrics The last step generates 2d and 3d graphs to visualize the tuning metrics for all tuned parameters. This can help to confirm, expand or reduce the range of each parameters. The optimal fmesure of a calibration can be improved by starting another calibration with fined-tuned ranges of parameters chosen thanks to the analysis of the graphs. Such graphs are presented in Figure B.19

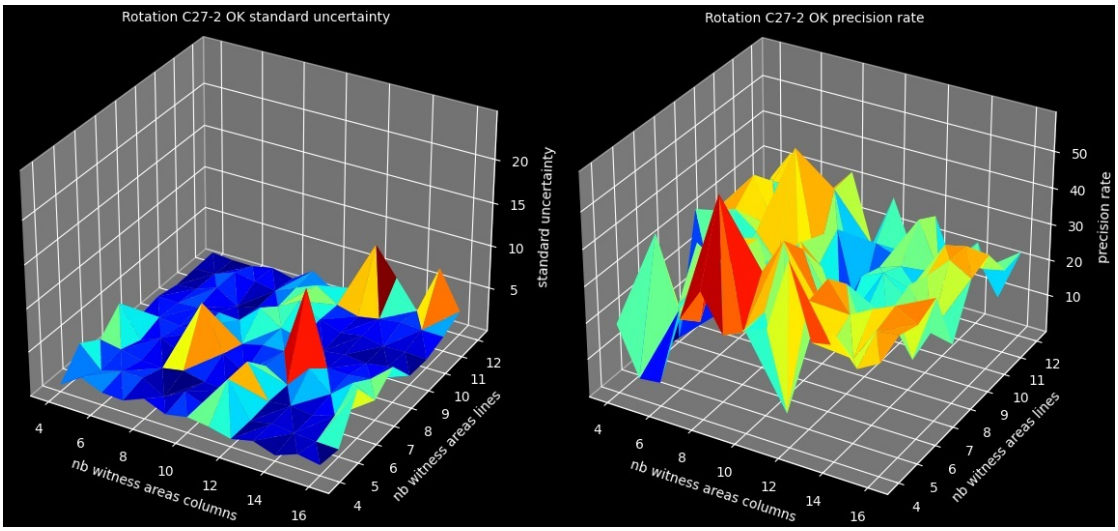

Figure B.19: UC Renault Welding C27.2 Rotation calibration metrics

B.5.1.5.7 Usage recommendations or limitations

This monitoring function is designed for single object inspection use cases. For a good precision, the inspected object should be present at a relatively fixed location and with the same average size in most of the images contained in the dataset chosen for the calibration. The inspected object should also have a minimum of contrast with its background. This function hasn't been designed to detect composition of rotation and translation.

B.5.2 Complex monitoring functions

B.5.2.1 Local anomalies detection (including Cloud detection)

B.5.2.1.1 Objective This function aims at detecting local anomalies on an image.

B.5.2.1.2 Inputs and outputs This function takes as input the list of anomaly to detect, the grid configuration, the image. It compresses the image, cuts it into cells and calls each detector on each cell. Each detector function takes as input the compressed image, the detection configuration calibrated for the associated anomaly and for this grid. The outputs of this function are an object with detectors status on each cell of the grid, and optionally heatmaps of anomalies.

B.5.2.1.3 Monitoring class
Rules-based Present-Time Monitoring (PTM)
Subclass: Out of Distribution Monitoring (ODM)
Monitoring type: Monitoring of the AI Model Input
Problem types: Object detection, image segmentation, image classification

B.5.2.1.4 Design principles

The monitoring functions are described in B.5.1.1 for Defocus Blur, B.5.1.2 for Motion Blur and B.5.1.3 for Brightness. For local anomaly detection, the input image is first divided into a grid of x rows and y columns. Then the monitoring functions are run on each cell of the grid. This creates a virtual heatmap of every classes of anomaly which can be saved into various image heatmaps. If the degree of anomaly in a given grid cell can impact the prediction of the model, the cell is colored in red. A potential impact colors a cell in yellow.

B.5.2.1.5 Tuning

The grid parameters must be chosen depending on the use case, the image size, the minimum, maximum and average size of the objects to be detected, the average size of anomalies and performance requirements. For instance if only one or two objects are expected to be detected per image in average, if these objects are bigger than 1/10 of the image and if the anomalies are mostly uniform on the whole image, then the grid is not necessary and the parameters of the grid can be 1 column and 1 row. But in some other cases with a huge image, relatively small objects, and local anomalies, a grid is required to detect the optimal impact of anomalies on expected objects. We tested a grid of 1 column and 1 row on Aerial Photograph Interpretation and it gave very poor anomaly detection rates with many false positives and bad precision. This was almost predictable and confirmed the need to benchmark the grid parameters to find a good trade-off between detection precision and performance.

B.5.2.2 Robustness & Stability Monitoring

Robustness is the analysis of the behavior of the model against very large disturbances in the input data.
Stability is the analysis of the behavior of the model against very small disturbances in the input data.

B.5.2.2.1 Objective

The robustness function aims at adding a very large disturbance to the input data and raising an alarm if the model decision doesn't change significantly.
The stability function aims at adding a very small disturbance to the input data and raising an alarm if the model decision changes significantly.

B.5.2.2.2 Inputs and outputs

The functions take in input the image, its category, the ODD of this category and the output of the PTM monitors. They return as output a local Robustness or Stability status and a global NFM SIREN verdict (GNSV).

B.5.2.2.3 Monitoring class

Rule-Based Near-Future Monitoring (NFM)
Subclass: Robustness and Stability Monitoring
Monitoring type: Monitoring of the AI Model Output

B.5.2.2.4 Design principles

- On every input image, a large disturbance for robustness or a small disturbance for stability is added for each class of anomalies to create one adversary sample per class of anomalies.
- Inference is run on those adversary samples by using a clone of the original model delivered by Renault Digital in Confiance environment.
- A robustness or a stability decision table returns an estimated status for each class of anomalies for the input image.
- A local robustness or stability SIREN function returns a local robustness or stability SIREN status for each class of anomalies on the N last local statuses.
- A global SIREN function returns a global NFM SIREN verdict (GNSV)

B.5.2.2.5 Detailed design

When an image and the metadata about its category arrive as input in the monitor, the first step is to run the PTM detectors on this image with the detection configuration for this category. This gives a list of measures and statuses for each class of anomalies. A first filter on the PTM statuses is then applied; if any detector returns a status with an alarm or a potential alarm, the NFM won't be run on this image. This allows to test the Stability and Robustness on images having the least anomalies as possible which reduces the number of NFM false alarms.
The second step is to wait for the inference result on this input image to be received from the industrial system to the monitor. In the implemented PoC the industrial system does not exist and this inference is done within the monitor by using the models and code delivered by the UC owners.
The third step is to estimate the level of anomaly in the input image for each class of anomalies through interpolation of detection measures from PTM with calibration data for the state given by the inference. Figure B.20 shows such anomaly interpolation on the Welding C9. If the inference returns the unknown class because none of the scores was higher than its associated inference threshold, the NFM won't be run on this image.
This estimated level gives the estimated location of the anomaly on the model ODD zones. If the estimated location is outside of a low-level stability zone of a high-level ODD zone, the image

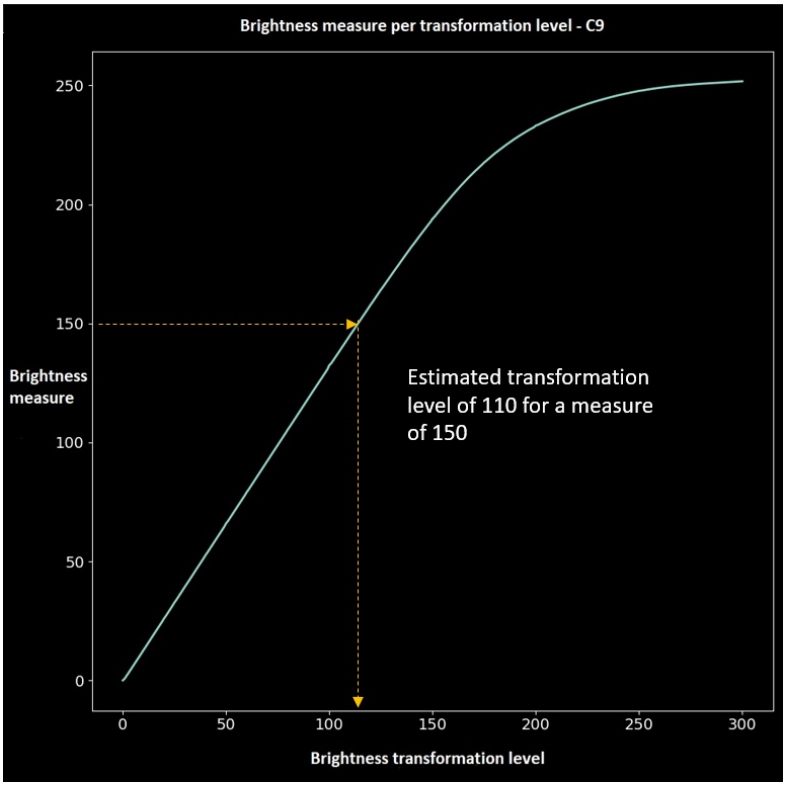

Figure B.20: Example of anomaly level interpolation for a detected measure of 150

is filtered out and NFM won't be run.
Next step is to apply a small disturbance for stability, or a large disturbance for robustness. The scale of disturbance is computed like this for each class of anomalies:
- Stability: A transformation level is randomly chosen between a minimum delta and a maximum delta as in Figure B.21. The minimum and maximum delta are pre-configured for each class of anomalies. If the minimum delta exceeds the stability zone, then NFM for stability is aborted. If the maximum delta exceeds the stability zone, then the border of this stability zone is used instead of the maximum delta.
- Robustness: A zone is randomly chosen within the low-level lack of robustness zones of the high-level OOD zones. A transformation level is randomly chosen between the boundaries of this zone as in Figure B.22.
Once the transformation for a given class of anomalies is generated, a clone of the model is used to run inference on this adversarial sample. The results of those inferences are compared to the inference of the input image through a robustness or a stability decision table to return an estimated local status.
Then a local robustness or stability SIREN function returns a local robustness or stability SIREN status for each class of anomalies on the N last local statuses.
Finally a global SIREN function returns a global NFM SIREN verdict (GNSV) by combining local robustness and stability SIREN statuses of all classes of anomalies

B.5.2.2.6 Tuning

The tuning of Robustness and Stability corresponds to the tuning of the model ODD charac-

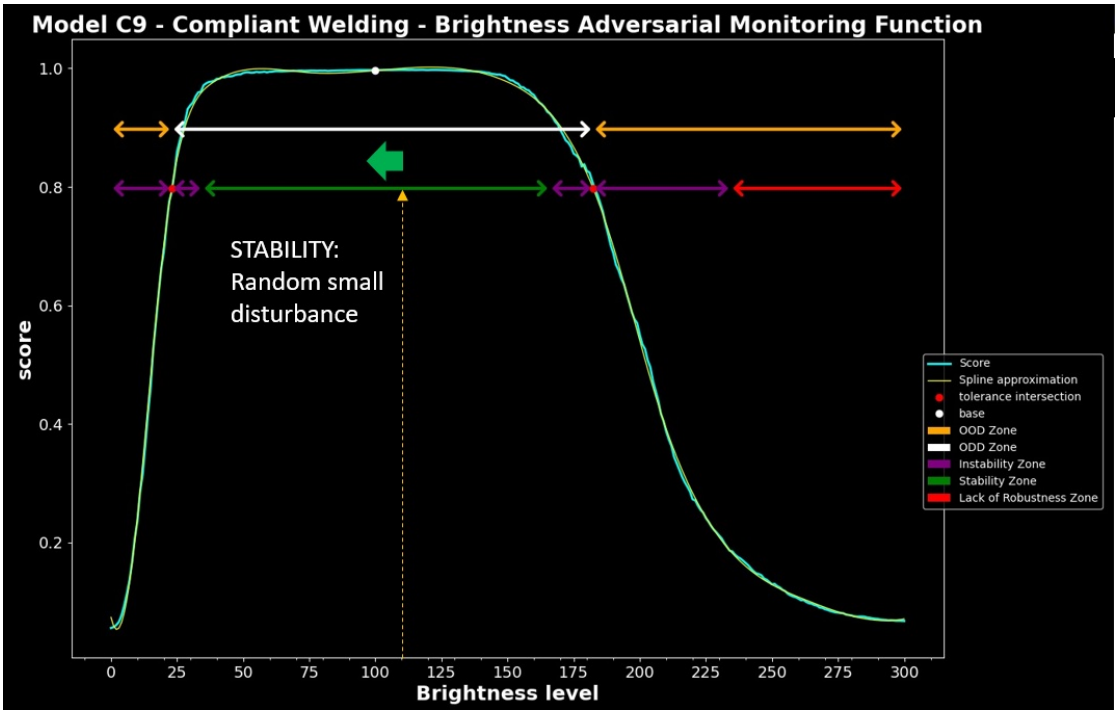

Figure B.21: Example of stability for an estimated anomaly level of 110

terization during calibration. The choice and the quality of the transformation functions, the selection of the samples to transform, and the parameters controlling how the high level and low level zones are defined, are levers that can improve the quality of the model ODD. The better the model is characterized and the most representative the samples are to the production data, the higher will be the precision of Robustness and Stability functions.

B.5.2.2.7 Usage recommendations or limitations

The robustness and stability functions are designed to detect data in production that doesn't verify behavior as measured during calibration on a specified dataset. It isn't designed to detect data that has been manipulated by attackers to fool the model.

B.6. Black-Box Time Series/Tabular Data Guidelines

B.6.1 Basic monitoring functions

B.6.1.1 Development of monitoring functions for dropout

B.6.1.1.1 Objective

The objective of this monitoring function is to detect the dropout (cf. Taxonomy A.2.5.3.2). For this monitoring function we will focus on 2 detectors: a chained missing values detector and a percentage missing values detector.

B.6.1.1.2 Inputs and outputs

The monitoring function takes as input the time series and a parameter called time window, this

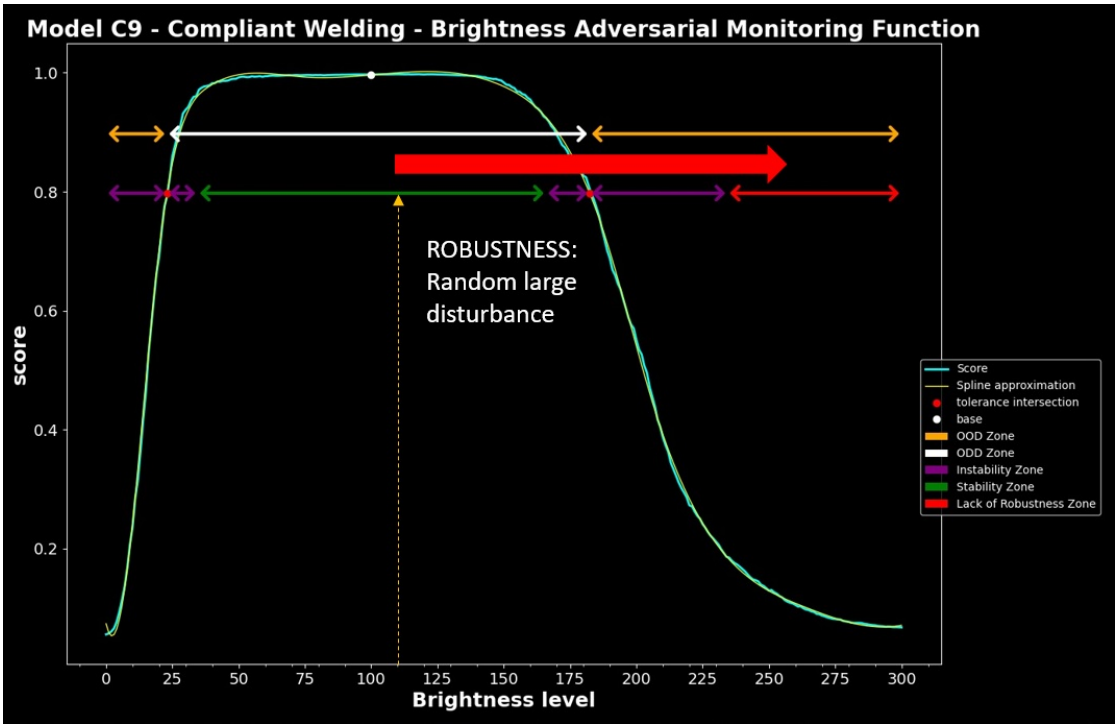

Figure B.22: Example of robustness for an estimated anomaly level of 110

parameter will be used for the sliding window to compute the percentage of missing value in each point of the time series.
The output of the chained missing values detector is the number of successive missing values. The output of the percentage missing values detector is the percentage of missing values. The monitoring function returns a status (Alarm, No Alarm) based on the model ODD.

B.6.1.1.3 Monitoring class

Rules-based Near-Past Monitoring (NPM) when used on past data.
Rules-based Present-time Monitoring (PTM) when used on actual data.

B.6.1.1.4 Design principles

The first detector computes the number of successive missing values (chained missing values) and the second detector computes the percentage within a time window (the time window will be set according to the memory of the model monitored). Both detectors complete each other for models that have high or low memory. They also fit together for forecasting models giving lot of importance to the closest points of the prediction. A status (Alarm/No Alarm) is raised at the end of the monitoring function depending on the output of the detector. The monitoring function process is summarized in Figure B.23.

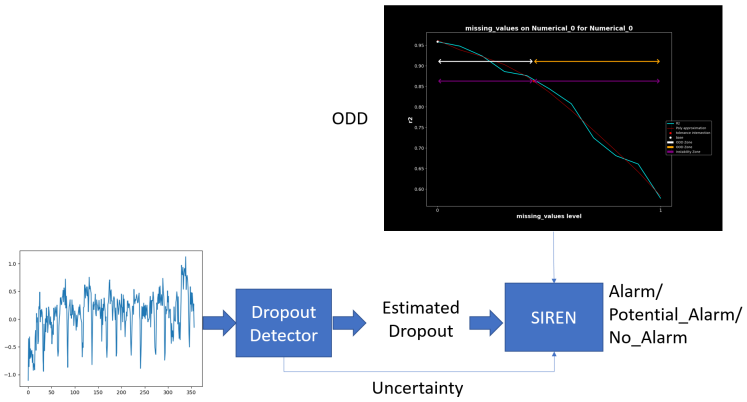

Figure B.23: monitoring process for dropout

B.6.1.1.5 Detailed design

- Percentage missing values detector:
  The concept of sliding window is used to compute the percentage of missing values. It consists in generating a value for each point of a time series by taking a subset of the data (a sliding window) and by computing a value on this subset. Using the sliding window, the percentage of missing value of the last n value is computed in each point of the time series, with n being the size of the sliding window (Fig. B.24).

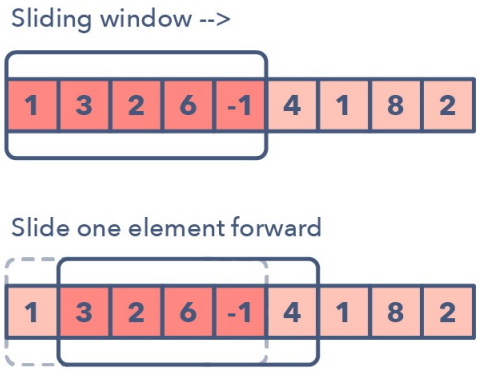

Figure B.24: Sliding window example

- Chained missing values detector:
  This detector is pretty simple, the data is browsed, each time a missing value appears, the counter of missing values is incremented and the value at the current position is saved. If the value is not a missing value, the counter is reset to zero.

Once the dropout effect is quantified, the representation of the model ODD threshold is used to raise a status (Alarm/No Alarm).

B.6.1.1.6 Usage recommendations or limitations

The monitoring function can be used on all time series, multivariate and uni-variate.

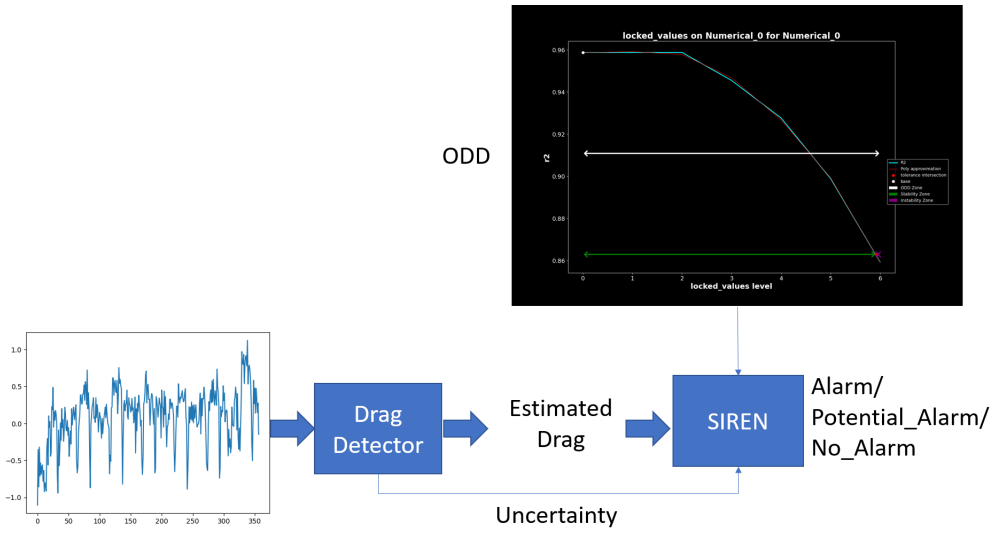

Figure B.25: monitoring process for drag

Usually in standard data files, missing values will be written using an empty string. Specific cases such as 'NaN' can be specified in the monitoring function parameters.

B.6.1.2 Development of monitoring functions for drag

B.6.1.2.1 Objective

The objective of this monitoring function is to detect drag (locked values) (cf. Taxonomy A.2.5.3.2).

B.6.1.2.2 Inputs and outputs

The monitoring function takes only as input the time series. The output of the detector is the number of successive equivalent values.
The monitoring function returns a status (Alarm, No Alarm) based on the model ODD.

B.6.1.2.3 Monitoring class

Rules-based Near-Past Monitoring (NPM) when used on past data Rules-based Present-time Monitoring (PTM) when used on actual data

B.6.1.2.4 Design principles

The detector computes the number of successive equivalent values. A status is raised (Alarm/No Alarm) at the end of the monitoring function by using the output of the detector. The monitoring function process is summarized in Figure B.25.

B.6.1.2.5 Detailed design

This detector is pretty simple, the data is browsed, each time the same value as the previous one appears the counter of locked value is incremented and the counter value is saved for the current position. If the value is different from the previous value, the counter is reset to zero.
Once the drag effect is quantified, the representation of the model ODD threshold is used to raise a status (Alarm/No Alarm).

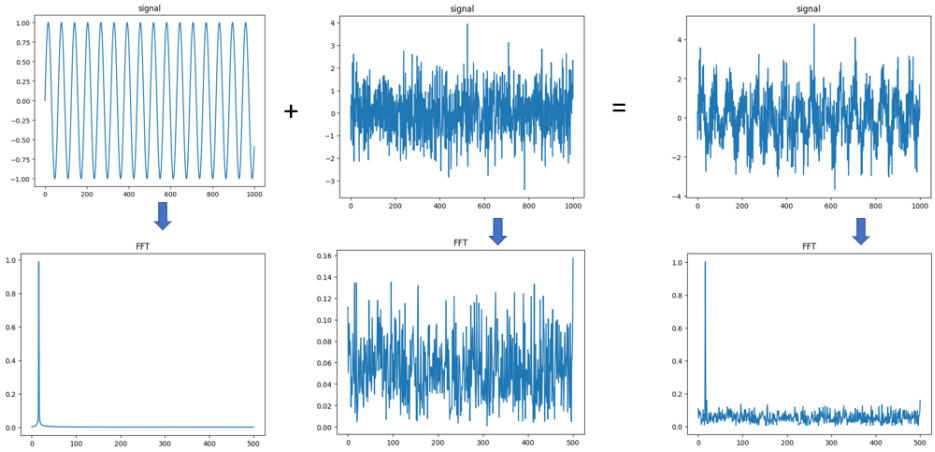

Figure B.26: Example of signals and their FFT

B.6.1.2.6 Usage recommendations or limitations

The monitoring function can be used on all time series, multivariate and uni-variate and there are no limitations.

B.6.1.3 Development of monitoring functions for noise

B.6.1.3.1 Objective

The objective of this monitoring function is to detect white noise (cf. Taxonomy A.2.5.3.2) The detector is first evaluated on how well it can quantify the noise using a confidence interval of 99%. The entire monitoring function is then evaluated by the LNE.

B.6.1.3.2 Inputs and outputs

The monitoring function takes as input the time series to detect the mean noise of the entire time series.
A time window can be specified to quantify the noise locally. The larger the window the better the detection will be but also the lesser the accuracy of the position of the noise.
The output is a status (Alarm, Potential Alarm, No Alarm) based on the approximated standard deviation and the model ODD threshold.

B.6.1.3.3 Monitoring class

Rules-based Near-Past Monitoring (NPM)

B.6.1.3.4 Design principles

This monitoring function is based on the following statistical particularity of the white noise Wikipedia (2022b): a white noise is a realisation of a random process where its Power Spectral Density (PSD) is the same in every frequency of the bandwidth as seen in (Fig. B.26). The monitoring function is implemented with FFT (B.4).

$$f_j = \sum_{j=0}^{n-1} x_k e^{-\frac{2\pi i}{n} jk} \tag{B.4}$$

Once the noise is quantified by the detector, the SIREN monitoring function to returns a status. The monitoring function process is summarized in Figure B.27.

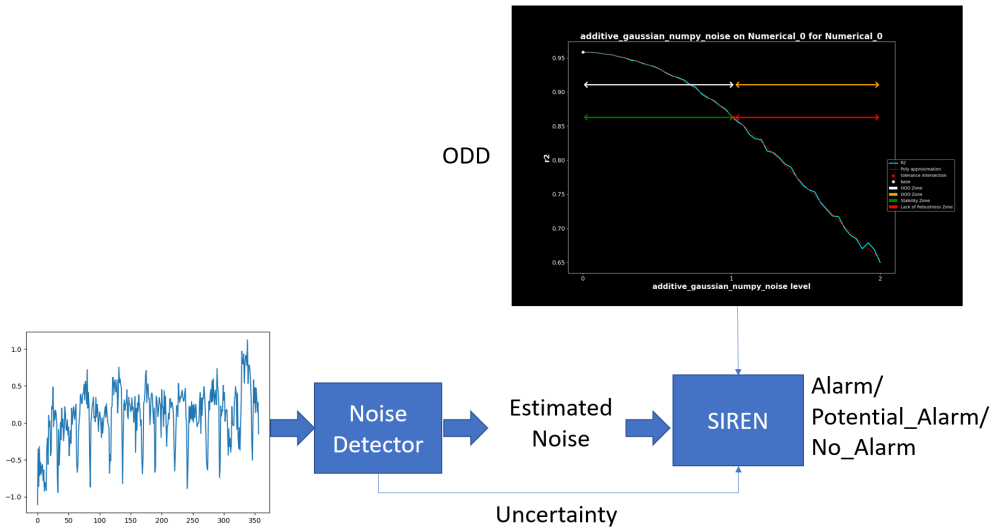

Figure B.27: monitoring process for noise

B.6.1.3.5 Detailed design

The design is composed of two parts, the detector and interpolation.
The detector computes the approximated average PSD of the noise in the data by following those steps:

- Transformation of the data using the FFT
- Remove the low frequency points
- Identify and remove the frequency peaks that has high chance to correspond to the real signal (ODD data).
- compute the mean of the result

Then once the noise in the domain of frequency is quantified, it is compared through interpolation with the real transformation (additive noise) which has been applied on the data during MOODD. This interpolation has been made possible by generating n samples for k transformation levels with the transformation being the additive noise and the transformation level being the standard deviation of the noise distribution. Then the mean and confidence interval (99%) are computed on the data for each transformation level. This interpolation is showed on curve (Fig. B.28 and is used to determine the approximate standard deviation of the noise based on the interpolation and the uncertainty of the detector based on the confidence interval.
Finally, the model ODD is used to determine the alarm thresholds. An uncertainty range is added around the threshold based on the uncertainty of the detector. To resume, based on the model ODD (Fig. B.29), the algorithm will set:

- An alarm if the noise estimated is out of the model ODD zone.
- A potential alarm if the noise estimated is close to the model ODD threshold and within the uncertainty range.
- No alarm if the noise estimated is in the model ODD.

To conclude this section, the data of the use case of Air liquid with local noise added is represented in the following graphic (Fig. B.30). The ground truth of where the noise has been added is in orange and the alarms raised by the siren algorithm based on the noise detector with a time window parameter set to 100 points are shown in red. This graphic shows our result but

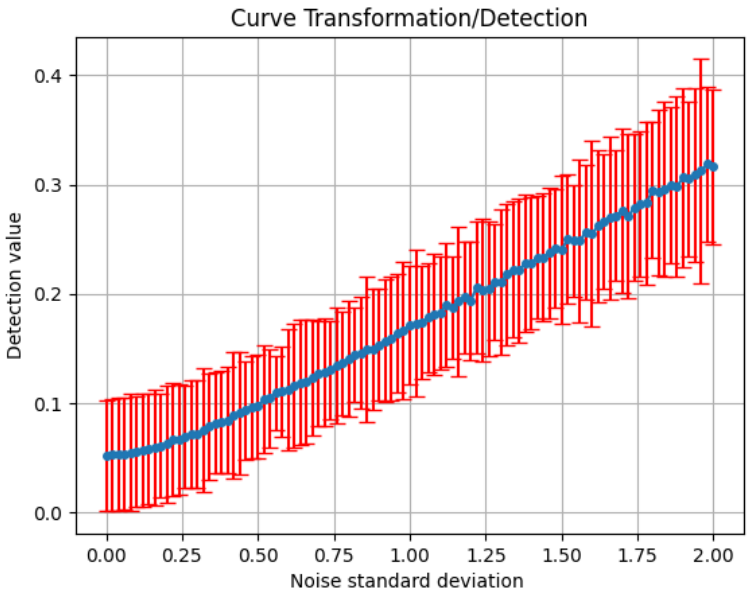

Figure B.28: Curve used to interpolate Detection/Transformation value

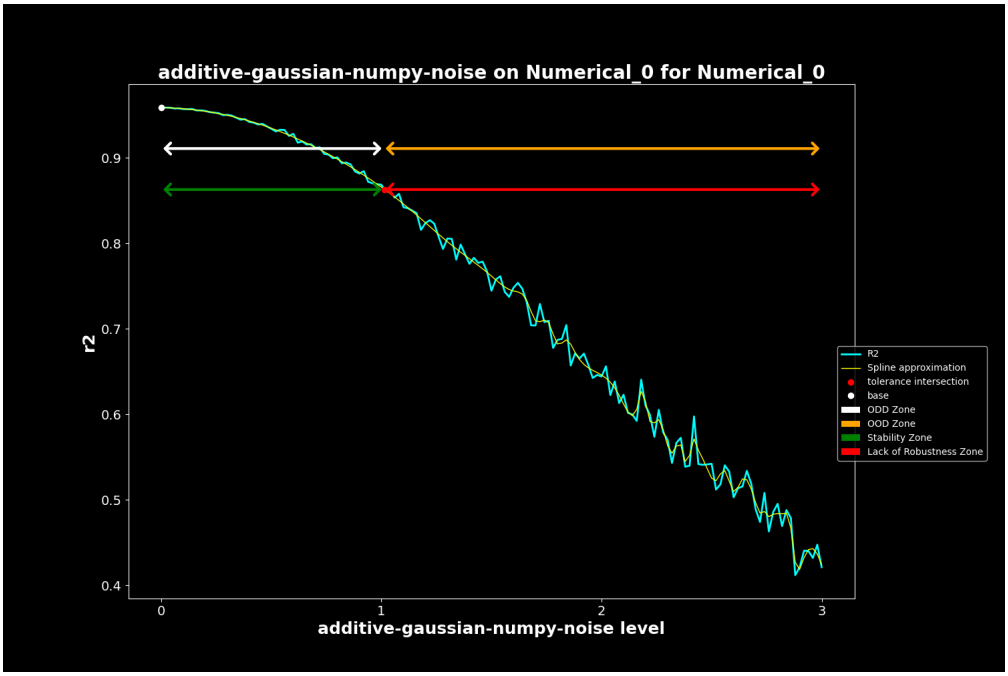

Figure B.29: model ODD for Gaussian noise applied on variable "Numerical_0"

also highlights our main challenge which is the difficulty to detect noise when it's not constant in the time window. This can be seen in both side of the noise zone where the 100 points around aren't always noisy data.

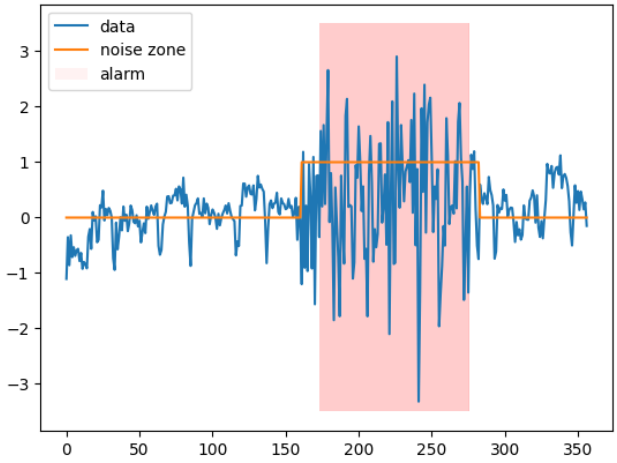

Figure B.30: Output of the monitoring function

B.6.1.3.6 Tuning

Tuning of the window size. The parameter window size can be modified to allow the detector to quantify the noise locally (Fig. B.31) by creating a sliding window (Fig. B.24).

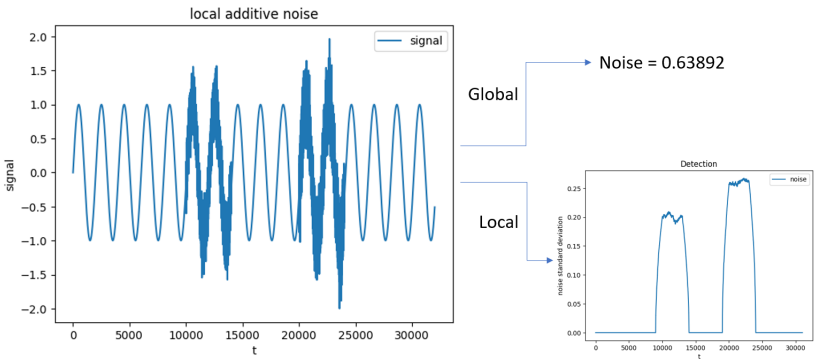

Figure B.31: Local vs Global

A lot of data points are needed to quantify a random process like white noise properly. The same is true when using time window to detect locally the noise. The larger the window the better the detection will be but also the lesser the accuracy of the position of the noise.
Therefore, a trade-off (Fig. B.32 is required, whether to have a precise localization of the noise or a precise quantification/detection of the noise.

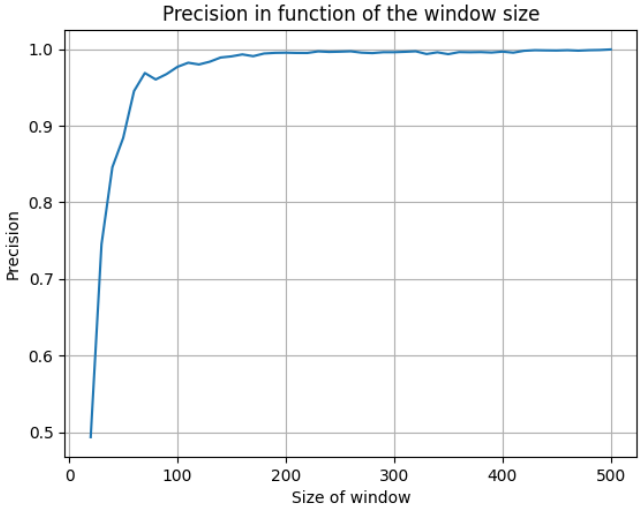

Figure B.32: Example of precision in function of the time window on a noised sinusoidal signal

B.6.1.3.7 Usage recommendations or limitations

- distribution problematic: The detection will perform better if the data comes from the same distribution. For example in the use case air liquid, when the detector is evaluated on all data for a time window set to 100 then the achieved confidence range is +/- 0.499. When evaluating the same detector on only one modulation (a sub part of the data set), confidence range of +/- 0.242 is achieved as seen in (Fig. B.33. So our recommendation is to calibrate the detector for each different industrial regime which could create different distribution in a time series.

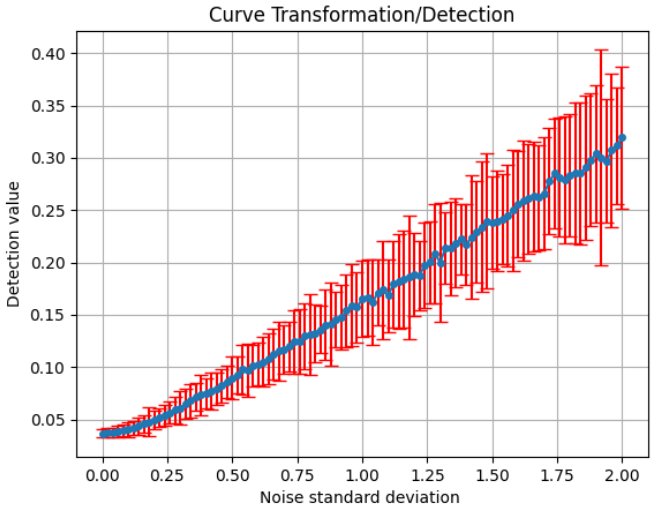

Figure B.33: Curve used to interpolate Detection/Transformation value

- time window tuning: As mentioned previously the parameter time window can impact greatly the performance of the detector. So it is important to correctly tune the parameter

depending on the industrial motivation. For classification problem, all the data in the time series should be used to compute the average noise in the time series; for segmentation problems we usually can afford to have a medium size time window. Finally for forecasting/regression, a small or a medium size time window should be used depending on the time step between data points and how precise we want to localize the noise.

B.7. Grey-box Images Guidelines

B.7.1 Monitoring of the Prediction Uncertainty applied to object detection task

In the literature, the Predictive Uncertainty (PU) is decomposed into two factors: the data uncertainty (aleatoric) and the model uncertainty (epistemic). To establish to the right diagnostic and define the appropriate corrective action, it is recommended to know as accurately as possible what is or are the source(s) of the PU. In the case of uncertainty in the data, additional data management considerations could be envisaged such as data source identification, data collection, data preparation, etc. In the case of model uncertainty, it would be more appropriate to rework on the model design in terms of AI technology selection, model building, model training and evaluation, model optimization, model verification, etc. In this section, we present a monitoring function of PU applied to object detection task through Thales aerial photograph interpretation use case.

B.7.1.1 Prediction Uncertainty Monitoring Function

B.7.1.1.1 Objective This Prediction Uncertainty Monitoring (PUM) function as depicted in B.34 aims at computing a PUM Score (PUMS) by combining two quantitative uncertainty measures:

- The Object Detection Probability (ODP), also known as *confidence score* or *objectness loss* in YOLO terminology, indicates the epistemic probability given by the model that an object is present in a bounding box, and that this object is of the predicted class.
- The Input Integrity Monitoring Score (I2MS) is a measure of the aleatoric uncertainty in the input image based on the detection of anomalies in the bounding box and its neighbourhood.

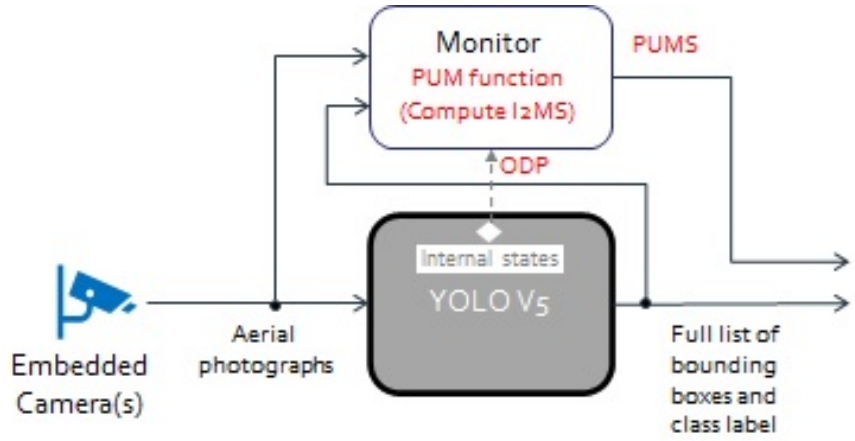

Figure B.34: Overview of the Prediction Uncertainty Monitoring Function

In the current version of the PUM function, ODP is obtained through a probe in the Yolo model (grey-box) whereas the I2MS is computed in the PUM function independently from the model. In an enhanced version of this PUM function, it is foreseen to also compute independently the ODP measure in the PUM function in order to remove any common mode of failure between the monitoring function and the monitored Yolo model.

B.7.1.1.2 Inputs and outputs The PUM function takes as input the ODP and the I2MS of a given bounding box. It returns the PUMS for this bounding box. This function updates an object containing the bounding box information as computed by the YoloV5 model and the scores with their ODP, I2MS and PUMS. When all these scores have been computed for all bounding boxes in the image, it is also possible to compute a heat-map showing coloured bounding boxes where the colours are configurable parameters depending on the PUMS value. More details are provided below in the paragraph dedicated to results.

B.7.1.1.3 Monitoring class

- Functional Task: object detection in images
- Monitoring level: grey-box
- Monitoring Technology: rule-based monitoring function
- Monitoring Timescale: present-time monitoring (PTM)
- Monitoring Function: predictive uncertainty monitoring (PUM)
- Monitoring Asset: inputs, internal states and output of the AI Model

B.7.1.1.4 Design principles

For a given bounding box i, $PUMS_i$ metric is simply computed by the following equation:

$$PUMS_i = ODP_i \times I2MS_i \tag{B.5}$$

where ODP_i is an external input for the PUM function that is computed by the YoloV5 model and is accessible from the bounding box object through an appropriate *get()* method (c.f. EC1 documentation on this use case), and $I2MS_i$ is a local metric computed within the PUM function by considering a grid on the image and by reusing detectors of blur, brightness, and colour anomalies already described in the subsection B.5.2.1.

For a bounding box $i \in \mathbb{N}^*$ in intersection with $j \in \mathbb{N}^*$ cell(s) of the image grid, for $k \in \mathbb{N}^*$ different types anomalies, and for $\alpha_1, ..., \alpha_k$ corresponding to real weights ($\alpha_k \in \mathbb{R}_+$ assigned to the different types of anomalies such that $\forall p \in [1,k], 0 \leq \alpha_p \leq 1$ and $\sum_{p=1}^{k} \alpha_p = 1$, the $I2MS_i$ metric can be computed by the following equation:

$$I2MS_i = 1 - \frac{1}{OBB_i} \times \sum_{p=1}^{k} \alpha_p \times \sum_{j} (PA_{j,p} \times (OBB_i \cap AMB_{j,p})) \tag{B.6}$$

where OBB_i is the area of bounding box i, $PA_{j,p}$ is the probability to have the anomaly p in the grid cell j, and $AMB_{j,p}$ is the area of the virtual grid cell j for the anomaly p.

C. Conclusion

The developments of monitoring functions has been structured using innovative multi-time scale online monitoring framework. This framework is based on the acknowledgment that the dynamics of the system and thus the "time" variable is a major factor in the detection of anomalies. The combination of several monitoring timescales - Present-Time Monitoring (PTM), Near-Past Monitoring (NPM), and Near-Future Monitoring (NFM) - on different monitoring assets (inputs, internal states, and outputs of the AI model) is a solution to ensure "by design" a high anomaly detection rate of the online monitor. The developments have also been structured using the principle of proportionality to risk with a sophistication level of the online monitor that is commensurate to the criticality of the AI-based product. As depicted in C.1, for low critical application the AI Model will be monitored as a black box (only external inputs and outputs of the AI Model are considered by the monitor). For medium critical application, a grey-box approach consists in adding to the former black-box approach the monitoring of internal states of the model obtained through an appropriate instrumentation of the AI Model (use of probes). For high critical application, which need the highest level of sophistication, in addition to grey-box approach, a full access to the detailed design of the AI Model is granted with the possibility to observe the evolution of several variables/internal states of the AI Model at the same time. This would be a white-box approach but it has not been explored in this program through rule-based methodologies due to lack of time.

Automotive (ISO 26262)	ASIL
Aviation: airborne (ED-12/DO-178/DO-254)	DAL
Aviation: ground (ED-109/DO-278)	AL/SWAL
Industries (IEC 61508)	SIL
Railway (CENELEC 50126/128/129)	SIL
Space (ECSS-Q-ST-80)	CAT

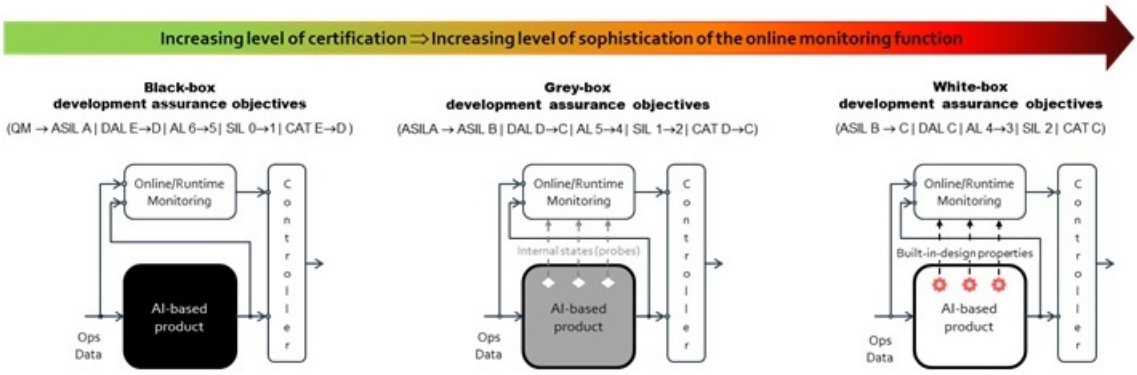

Figure C.1: Black-Grey-White box Monitoring Strategy

Bibliography

Aerospace, S. (1996). Guidelines and methods for conducting the safety assessment process on civil airborne systems and equipment. Technical report.

Aerospace, S. (2010). Guidelines for development of civil aircraft and systems. Technical report.

Audibert, J. (2021). *Unsupervised anomaly detection in time-series*. Theses, Sorbonne Université.

F., M., E., J., G., F., H., D., C., G., A., G., B., B., L., P., L., A., H., B., B., B., D., D., J.-B., G., A., H., S., P., K., D., C., P., J.-M., G., C., C., S., P., M., D., C., C., L., G., D., G. F., B., L., S., G., and A., A. (2021). White paper machine learning in certified systems. Technical report.

IEC (2010). Functional safety of electrical/electronic/programmable electronic safety-related systems (e/e/pe, or e/e/pes). Technical report.

ISO (2011). Véhicules routiers - sécurité fonctionnelle. Technical report.

SAE, E. . (December 2022). Process standard for development and certification/approval of aeronautical safety-related products implementing ai (draft 4b). Technical report.

SAE J3016 (2018). Taxonomy and Definitions for Terms Related to On-Road Motor Vehicle Automated Driving Systems.

Tong, H., Li, M., Zhang, H., and Zhang, C. (2004). Blur detection for digital images using wavelet transform. 1:17–20 Vol.1.

Wikipedia (2022a). Modulation d'amplitude Wikipedia, the free encyclopedia. [Online; accessed 07-December-2022].

Wikipedia (2022b). White noise Wikipedia, the free encyclopedia. [Online; accessed 07-December-2022].

Confiance ai

Title: Methodological Guideline for Rule based Monitoring

Keywords: monitoring, rule-based

The models designed with AI technologies are not easily testable and certifiable due to models complexity and data quality requirements. This guide proposes methods to implement a rule-based monitor able to detect and explain deviation of the AI component deployed in production from the expected behavior and precursors of the occurrence of failure conditions based on a predefined set of safety properties.

Our partners:

Air Liquide, AIRBUS, Atos, cea, Inria, NAVAL GROUP, GROUPE RENAULT, SAINT EXUPÉRY, SAFRAN, sopra steria, SystemX, THALES Building a future we can all trust, Valeo, INVESTIR L'AVENIR