Masterarbeit

Name:	Colin	Klein	
Matr.-Nr.:	368642		
Thema:	FAIR Sensor Health Monitoring of Flight Test Data		
Betreuender Assistent:	M. Sc.	Matthias	Bodenbenner
Aachen, den	15.06.2015		

Diese Arbeit wurde vorgelegt am Werkzeugmaschinenlabor WZL, Lehrstuhl für Fertigungsmesstechnik und Qualitätsmanagement.

Aachen, 9. May 2023

V. Name - Tel. 0241-80 xxxxx

Master Thesis

for Ms./Mrs. Cand.-Ing. Erika Mustermann

Matriculation number: 081511

Topic: FAIR Sensor Health Monitoring of Flight Test Data.

The start of production for series production represents a major uncertainty and cost factor for both manufacturers and users of automated production systems. In particular, the poor planning of the production start-up requires new approaches that support the early safeguarding of the functionality and performance of automated production systems. Within the BMBF joint project Ramp-Up/2, the step from two-dimensional alphanumeric planning to an integral 3D-based digital verification of plant development and commissioning is aimed at. For this purpose, a kinematic 3D model of the production plant and all control components (NC/PLC) are simulated by virtual NC/PLC software modules and the mechanical behaviour of a machine is depicted by the Siemens Machine Simulator (MS). Based on this virtual production system, the aim of the plant development is to enable a preliminary verification of the production control software. In doing so, technical errors as well as operating and software errors are to be simulated and the reaction of the control software is to be analysed by means of diagnostic tools.

Within the scope of the work, concepts and tools are to be developed which enable the testing of the functionality of a production control software. In particular, the following questions are to be dealt with: which test cases can occur, how errors/tests can be reproduced, which data are necessary for the clear diagnosis of an error and to what extent tools can be used for error correction. Based on these considerations, a concept for information visualization is to be developed and realized exemplarily. The information content as well as the temporal sequence of the information flow within the production control software should be mapped and the possibility should be provided to reset the system into a freely defined state (time). The developed concepts are to be realized exemplarily on the basis of the control software cosmos4. The functionality and performance of the developed tools will be verified using an example scenario in the Integrated Manufacturing and Assembly System (IFMS) of the WZL.

In detail, the following subtasks have to be solved:

- Introduction with the leading software cosmos4
- Development of a comprehensive concept for error diagnosis and correction
- Exemplary realization of a scalable information visualization and recovery
- Documentation of the work

Prof. Dr.-Ing. Robert Schmitt

I Contents

II Acronyms

Symbol	Unit	Description
a_e	mm	Width
a_p	mm	Cutting depth
t		Number of teeth [Note: Sorting is alphabetical]
α		angle

Abbreviation	Description
DP	Polycrystalline diamond
ADC	Analog Digital Converter

III List of Figures

IV List of Tables

1 Introduction

Sensors fail. Failure rates often depend on the sensor as well as the complexity of the system into which the sensor is integrated. Historically, sensors (from latin *sensus*, perception) are defined as entities that measure physical conditions. This information then gets transmitted by signals that are emitted by the sensors [PEÑA23]. Aircraft sensors generally emit an electric current that gets transformed into a digital signal by an Analog to Digital Converter (ADC). This digital signal then flows through other stages until it reaches the Main Aircraft Bus where it gets used by the aircrafts systems.

Within this works scope the ISTAR aircraft (In-flight Systems Technology Airborne Research) gets examined. The ISTAR is a Falcon 2000LX business jet that is equipped with various additional sensors including strain gauges, accelerometers and changing experimental configurations. öö-

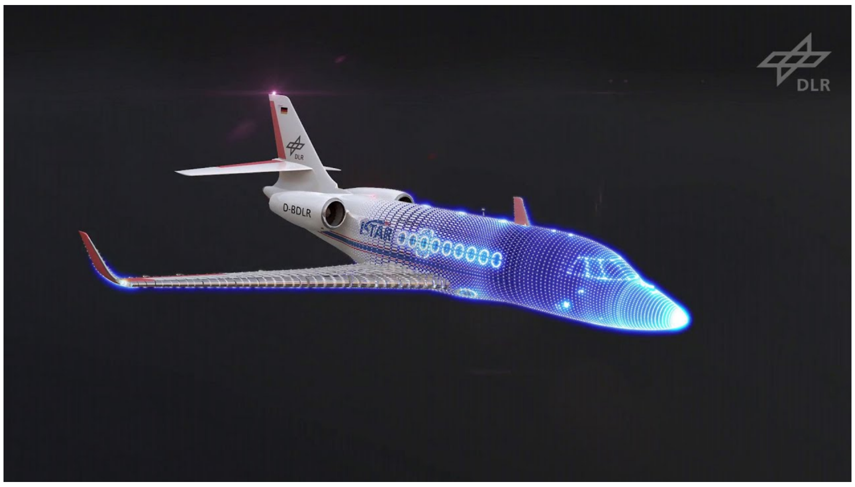


Figure 1.1: The ISTARs sensor data systems will be analysed within this work [**dlr_dlr-forschungsflugzeug_2020**]

??

1.1 What

Q:What is going to be examined within this thesis? E:The ISTAR aircraft is currently the aircraft generating the most data for the DLR's research aircraft fleet at the Brunswick Research Site. The DLR also owns the largest European fleet of research aircraft, operating 12 aircraft for research purposes ranging from atmospheric research to testing operating limits, pushing the aircraft operating envelope. -the data acquisitioning system (DAQ) as well as the sensor behaviour is going to be focal point of this work -also in conjunction with the FAIR principles to allow reusability of the components of this work as well as interoperability with outside algorithms, Accessible due to upload to stash and Findable due to condensing errors into tags and using conventions to catalyze digitalization.

C: examine ISTAR DAQ in detail to detect sensor behaviour anomalies and implement a clear infrastructure to fulfill FAIR principles.

Q: Why is this important? E:-reduce manual overview of sensors. Currently, much manual postprocessing is needed to get an overview over the large datasets. Large datasets are difficult to work with due to limited computing power. Solving this by working with a Server architecture may be of assistance -reduce downtime due to sensors. A diagnostic tool for detecting sensor behavior can facilitate the processes and allow a quicker follow up to detect sensor errors since currently system information is distributed and not clearly set -increase detection rate (sensor faults often not detected). Once a sensor fails during a flight experiment and does not get fixed, the experiment needs to be reflown. This is economically painful since aircraft configurations get customized to experiments needs and the aircraft free slots during a year are low. So a worst case scenario could be that a whole experiment needs to be reflown and a configuration needs to be recustomized and refitted which is a time- and funding intensive task. -increase reaction time. Often errors are detected weeks after the flight is over. With a software detecting errors directly after sensor data is extracted from the aircraft errors may be detected on the same day. allowing a quicker reaction maybe repeating the experiment during the affixed timeslot. -build a foundation for a reliability index for sensors. Vital for big data operations within digitalization

Figure 1.2: The DLRs research fleet without its newest member, a Dornier-228 for Hybrid-Electric Propulsion Research [**dlr_dlr-fleet_2018**]

??

and the stash project. Should allow for quick implementation of custom data quality algorithms. -FAIR principles are the new standard within the DLR. Operations however are far from the fulfillment. Documents containing descriptions of sensors and setups are distributed among multiple employees obstructing any efforts to comprehend the already complex aircraft system and its inherent generation of data. -Safety critical errors may be detected earlier since the proposed routine has potential to go beyond the scope of commercially tested software that is generally employed within aircraft. This software allows a quicker development cycle due to not being bound to the amount of certification aerospace software needs to go through C:A tool for detection and avoidance of sensor failures is needed to mitigate risks associated with experimental setups since aircraft hours are expensive and the amount of sensors is vast.

Q:Where does ISTAR data come from? E:-Generally, the ISTAR has a base DAQ system by IMC co. This base contains inputs from the main aircraft data bus, the experimental nose boom, experimental strain gauges and acc sensors as well as an experimental Inertial Measuring Unit (IMU) combined with a GNSS-System. -In addition, complementary DAQs may be added to fulfill given experiment requirements. -ascb bus data originates from various sources, ranging from air data that gets measured by a sensor working on an electric current which then gets transformed by an Analog Digital Converter (ADC) into a digital, discretized signal which then gets fed into the main aircraft bus system (simplified). It then gets transmitted to the IMC DAQ. Trouble is a multifaceted effort within this system.

C: Data originates from all over the aircraft and gets collected in the aircraft experimental DAQ.

1.2 Origin

ISTAR DAQ DLR data generation (DAQs) (How Data is checked)

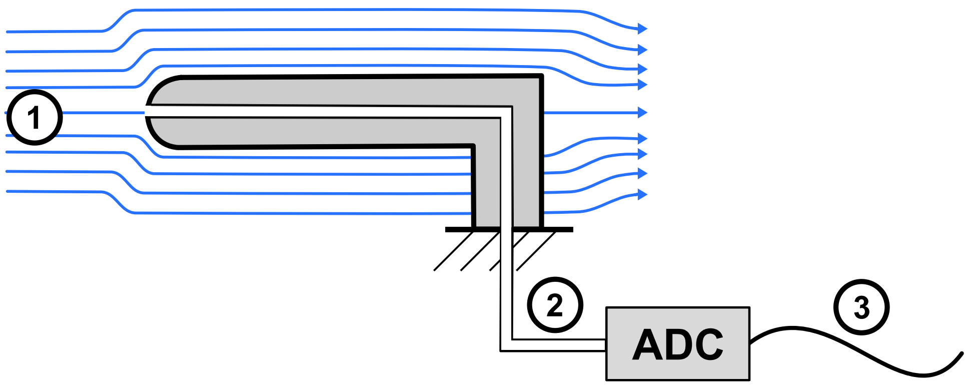
Matthews flowchart(dataflow from sensor voltage through computer-computer-user) Exemplary for a single sensor.

The ISTARs DAQ records the aircrafts own flight data bus (ASCB, avionics standard communications bus), the experimental Noseboom, an additional GPS unit and various additional strain gauges and acceleration sensors that are distributed across the aircraft. configuration currently is described differently within each sub-group.

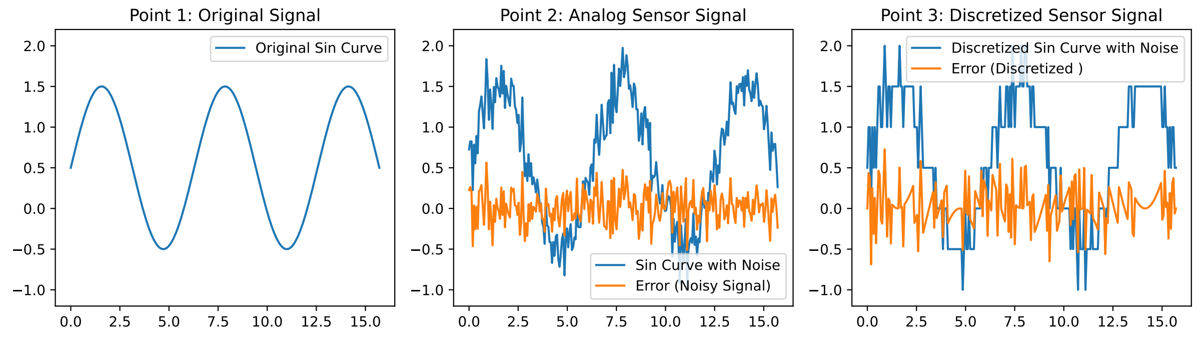
Different sensors Different formats/sources

1.3 Signal Discretization

State values of the real system are never measured without some error. Upon measurement, sensor values are discretized by converting it to a digital signal. This happens in two steps that are presented in an exemplary setup in figure 1.2a. In the first step the real state value gets converted into a sensor signal by measuring it within a pitot tube. Within this step, some white noise generally occurs. Within the second step the sensor value gets converted to a digital signal by feeding it into an Analog Digital Converter (ADC). During the ADC step a discretization error occurs. Based upon sampling rate discretization errors occur in time and value direction. After the transformation by both steps, the data series contain a white noise sensor error as well as a discretization error (see figure 1.2b, Point 3). Great effort is made to avoid such errors. For sensor errors i.e. a nose boom is fitted to the aircraft to measure undisturbed stream conditions. For value and time discretization the resolution of the ADC is chosen to guarantee necessary parameter precision. An exemplary signal transforming process is shown in figure 1.2b.



(a) pitot tube setup for measuring dynamic pressure



(b) Original signal into measured and discretized signal

Figure 1.3: Simplified, exemplary signal processing flow for measurement of a dynamic pressure for real (1), analog sensor (2) and discretized sensor (3) values

The dynamic pressure in the air (denoted by 1) is ideally undisturbed and smooth for the actual state value. Within the sensor and stream close to the aircraft, the air is disturbed by the aircraft itself and also by the sensor, leading to the measured value at position 2. signal conversion from pressure to a voltage as well as discretization of the analog voltage to a digital signal is summarized within the ADC-Box since it is assumed that the errorwithin the pressure conversion is small compared to the errors within sensor and actual ADC. A sample signal conversion is shown within figure 1.2b. Of course, errors are exaggerated for illustration since state of the art systems possess a far smaller level of sensor as well as ADC error.

All recorded sensors are present in values that are already is a process that is integral to digitally recorded sensor values.

-mention data information -information loss through discretization

Time discretization value quantization

Q: How can to solve this rather extensive problem? E:-Fault Detection and Mode Analysis (FMEA) is a common problem in engineering disciplines and everywhere where systems become complex. Literature is rich in this regard, so finding an appropriate solution should be a fesible undertaking -work on the already working dataspace skystash upon which sensor data is already present -allow clear software interfaces to facilitate future extensions and modes -Goal is to allow quick overview over sensor errors. -further methods presented in chapter 2 C: FMEA is a rich field in which many smart people have already worked on similar problems. A new challenge arises from the implementation into a dynamic digital system also keeping in mind to remain open for new changes and addons.

1.4 How

Proposal SHM Detect errors and report them Check ISTAR Data Filtering out sensor errors is the goal of this thesis. As described in figure 1.3 the error can be imagined as a disturbance added upon the original sensor value.

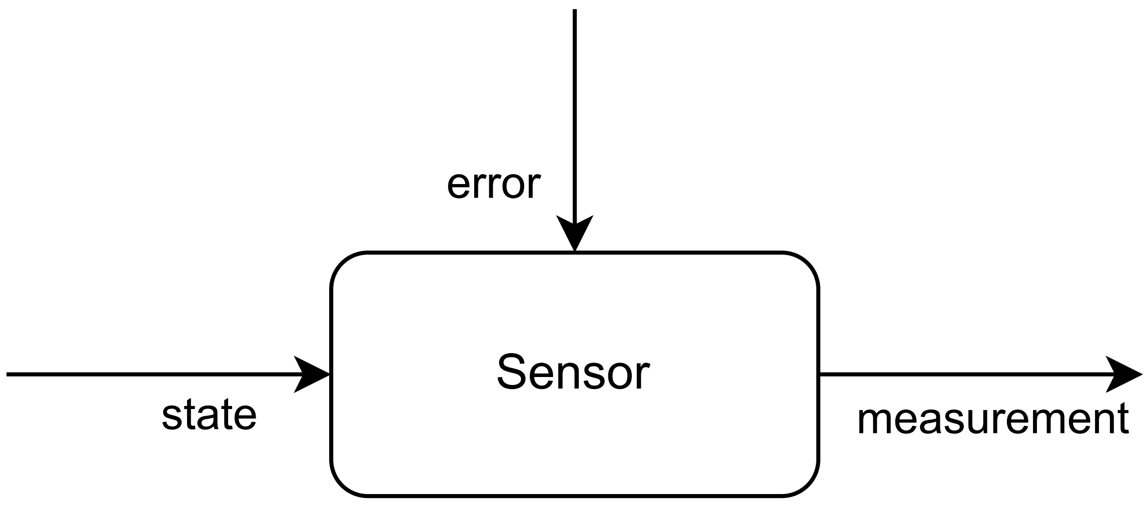


Figure 1.4: Sensor Signal

Q: whats the context on this work? E: this work happens within the DLRs effort to digitize and digitalize the research data and update its research data management strategy. -Part of DigECat project for ISTAR digital twin -past work: development of skystash architecture, upload and supply of research data into cloud -future work: metadata management on a larger scale, metadata frameworks already exists but challenges arise from indexing and providing such large amounts of paper trails in the context of an aircraft. Aircraft are highly complex systems that amass a great amount of paperwork since even its screws require a highly documented certification process. further big data efforts regarding data analytics allowing for easy scaling of this data base format -present work: work on prototypes of metadata management to represent sensor data. Calculations then can happen based on datasets with provided metadata without additional inputs.

C: digitalization and digital twin are the companions of this work. A database with data already exists. now it is time to structure metadata allowing for this work to happen dynamically. Building upon this, further metadata may be fed into the system allowing further analytics to be developed.

Conclusion

This work will examine the sensor data of the ISTAR aircraft, examining various algorithms and methods for FMEA. Then developing a software that allows dynamic implementations of various FMEAs to generate a dynamic backend facilitating development and finally generating reports on data quality for the aircraft systems. FAIR principles are vital for future legibility and exchangeability of data and hence become a central part of this thesis facilitating future use of the results generated within this work.

2 State of the art and a theoretical Background

Chapter 2 introduces the theoretic baselines for data acquisition, data metrics as well as standardized data formats in which to save metadata.

Topics to consider when starting the Sensor Health Monitoring process are mainly that of providing a structured overview of the Sensor Metadata which in itself consists of many layers as a dynamically generated set of metadata is desired. This should be able to accomodate changing Data Acquisitioning (DAQ) System configuration changes. Consideration is given to the SOIL data model and its' ability to accomodate the many demands that are expected of sensor data management. [BODE21] The second major part to consider is that of physical crossrelations and d̈eep checksẅhich are a experimental mode of checking for inconsistencies among the data. Major research and implementation work shall go into developing a dynamic model that is generated from the data and then checks back upon the data for possible discrepancies. This approach is chosen as it is estimated to be the most structured approach for a first prototype.

This topic shall give an overview over the state of the art technology as well as systems and describe the systems employed by this work. Structuring the data

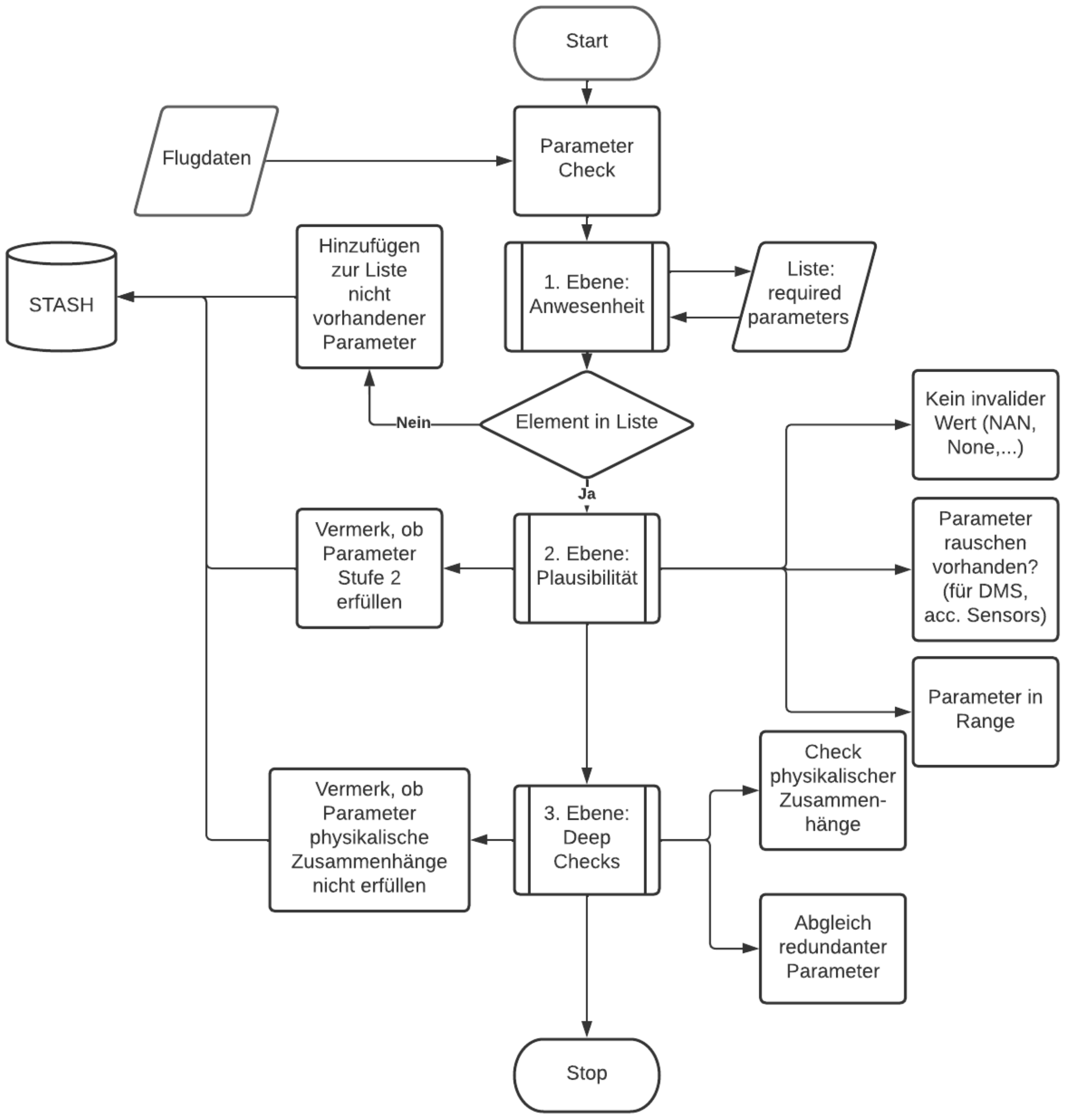

Start
Parameter Check
Flugdaten
1. Ebene: Anwesenheit
Liste: required parameters
Hinzufügen zur Liste nicht vorhandener Parameter
STASH
Element in Liste
Nein
Ja
2. Ebene: Plausibilität
Vermerk, ob Parameter Stufe 2 erfüllen
Kein invalider Wert (NAN, None,...)
Parameter rauschen vorhanden? (für DMS, acc. Sensors)
Parameter in Range
3. Ebene: Deep Checks
Vermerk, ob Parameter physikalische Zusammenhänge nicht erfüllen
Check physikalischer Zusammen-hänge
Abgleich redundanter Parameter
Stop

Figure 2.1

2.1 What is SHM?

-detect failures -

When talking about Sensor Health Monitoring (SHM) generally one requirement is made which is to detect failures and suspicious behaviours within a system. The failures then need to be quantified to allow writing algorithms to detect them. If one had a sensor value that differed strongly from its expected values a decision needed to be made how much the sensor value differed from the expected value and where the system would receive the information from. This and further considerations are examined in the following chapter to generate a broad overview over the topic of Sensor Health Monitoring and Fault Detection.

2.1.1 Scope

In addition to detecting faults it is also necessary for a SHM to display faults to the systems operator. This may happen in the shape of a display interface or a dashboard that generates fault and reliability ratings. Condensing information for operators is also a necessary part of SHM since suspicious occurences may be frequent and raw information about events is less helpful than already preprocessed data.

Within this chapter various algorithms and methods that may be used to detect faults are presented first and then followed by considerations in the direction of metadata. Standards for metadata as well as new propositions are examined and the skystash architecture which is used for data hosting as well as analytic tasks is presented.

2.1.2 Requirements

Notwendig -detect errors within previous flights. -develop a toolchain that accesses configuration data for parameters -find a format to communicate metadata as well as SHM findings -develop first on a small batch size optional -develop smarter algorithms -include all parameters

2.2 Fundamentals

Lets talk about Semantics and descriptors of data/metadata. Hence, a quick insert about semantics

2.2.1 Semantics

Semantics are defined in various way by various people. One definition that has found some acceptance is the definition by Metzlers Lexikon which relies on theories by Blackburn and Kutschera. [SHOE87; KUTS75]

First off, semiotics from the greek , semeion for sign, describes the theory of signs and their usage. Semiotics is divided into the areas semantics, syntactics and pragmatics. Within the definition at hand syntactics is given as the internal structure of signs within sign systems, pragmatics are defined as the theory of sign usage effectively thinking about how interaction with signs works. Finally, semantics define the relationship between signs and described objects. They work by allocating a structure/model to a predefined expressions

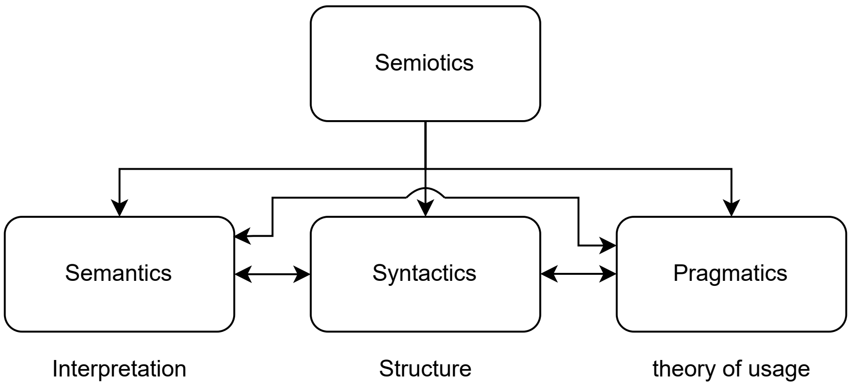


Figure 2.2: Semiology, according to Kutschera [KUTS75] and Shoemaker and Blackburn [SHOE87]

And now to the actual semantics that have been mentioned previously:

2.2.2 Metadata descriptors

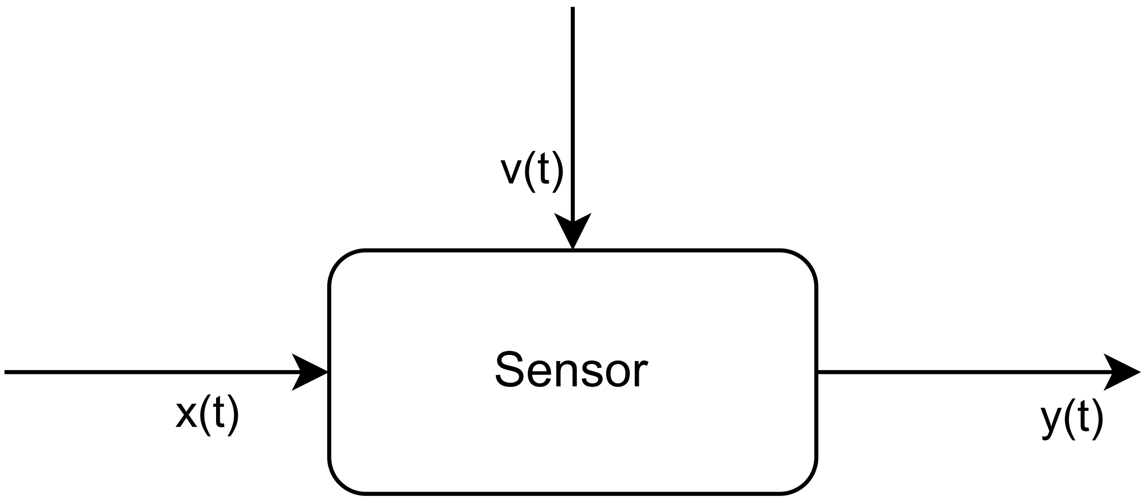


Figure 2.3: Sensor Signal with control systems

Unambiguous terms are needed to provide clear information within the sensor health monitoring data structure. This is needed in regards to basic terms like semantics as well as data metrics. In the following, industry standards and definitions are collected to accompany this work.

1What are metadata descriptors

u-is the input x-is the system state y-is the sensor value n-noise

2what are metadata descriptors

No clear, fully unambiguous and precise definition exists for most of following terms. However some assumptions have been made by e.g. isermann et al[ise97] within trends and apps.. in discussion with vdi/vde committees and the reliability, availability and maintainability (RAM) dictionary. Definitions are classified as:

deviation: difference to a reference value

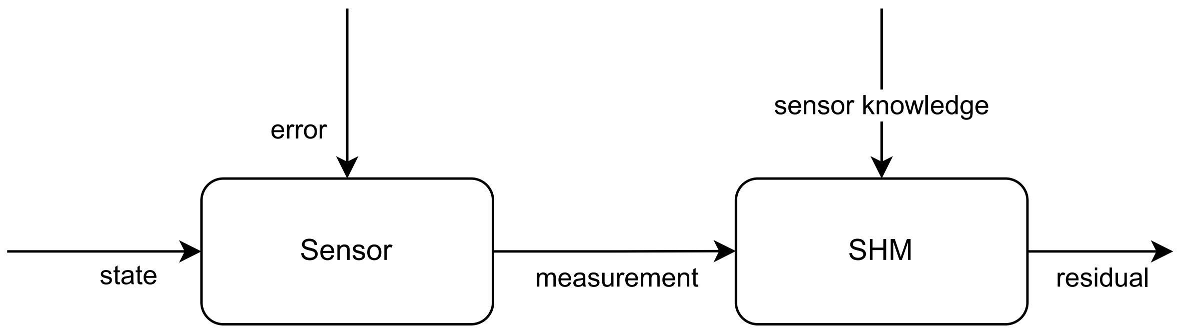


Figure 2.4: Sensor Signal Filter

- **states and signals (chapter 2.2.3)**
- **functions (chapter 2.2.4)**
- **models (chapter 2.2.5)**
- **system properties (chapter 2.2.6)**

2.2.3 States and Signals

Basic descriptors for occurences in signals (states). Faults are definedWithin this definition it is however not clearly defined

descriptor	description
fault	unpermitted deviation of one subset of the system
failure	permanent interruption
malfunction	intermittent regularity
disturbance	unknown, uncontrolled input
perturbation	input, leading to temporary departure from steady state
error	deviation between measurement and true, specified, theoretically correct value $y_e = \bar{y} - y$
residual	fault indicator based on deviations between measurements and model-based calculations $\hat{y} = \bar{y} - y_m$

Table 2.1: states and signals

2.2.4 functions

2.2.5 models

2.2.6 system properties

data quality: reliability (isermann) availability (isermann) accuracy

descriptor	description
fault detection	determination fault presence
fault isolation	Determination fault properties: kind, location, time of detection
fault identification	determination of size and time-variant behaviour of fault
fault diagnosis	includes fault detection, isolation, identification
monitoring	real-time determination of possible physical conditions and recognition and indication of behavioural ano
Supervision	monitoring and taking actions to maintain operation during faults
protection	means by which potentially dangerous behaviours are suppressed if possible or how consequences are avo

Table 2.2: functions

descriptor	description
quantitative	describe system in quantitative mathematical terms
qualitative	describe system in causalities and if-then rules
diagnostic	link specific inputs (symptoms) to outputs (faults)
analytical redundancy	determine a quantity in an additional way by using a mathematical process model

Table 2.3: models

descriptor	description
MTTF=$1/\lambda$	Mean time to failure
λ	rate of failure
MTTR $= 1/\mu$	mean time to repair
μ	rate of repair

Table 2.4: system properties

descriptor	description
reliability	ability to perform a function, measure $MTTF$, with λ as rate of failure per hour
safety	ability of a system not to cause danger to persons, equipment and environment
availability	$A = \frac{MTTF}{MTTF+MTTR}$

Table 2.5: system properties

3what are metadata descriptors

they need to be clearly defined as they have been now to set a common ground upon which this work can be built.

Data according to ISO 8000 cite [ISO22] iso5725: -accuracy(validity)+precision(reliability) Also see precision and accuracy definition [SMIT, S.33ff.]

Other measures for various positioning systems are found in FAA [FAA08]. Also, similar definitions are found as defined in Isermann [ISER11] $Reliability = 1 - Probability_{Failure}$ [FAA08, B.1.5]

B.1.10: Integrity: Display when system should not be used due to potential errors.

Validation: Black Box Testing. Results match expectations Verification: White Box Testing. Establish algorithm's truth

2.2.7 Statistics

[SMIT]

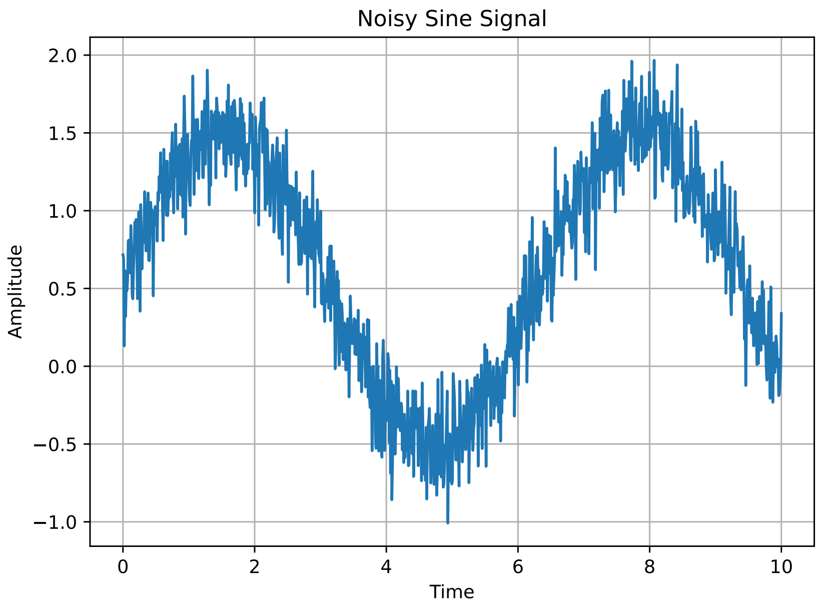


Figure 2.5: Noisy Sine Signal

Definition Fault Detection + Fault Diagnosis (Fault-Diagnosis Applications)

Signal properties are examined for a given signal in figure 2.5

Sensors generally produce errors within expected forms of output. -Noise -Measure using Covariance –> autocovariance -Offset -No response

Statistical representation of: Mean

[SMIT, S.13-17]

Time continuous mean

$$\mu = \frac{1}{T} \cdot \int_T x(t)dt \tag{2.1}$$

Time discrete mean:

$$\mu = \frac{1}{N} \cdot \sum_{i=1}^{N} x_i \tag{2.2}$$

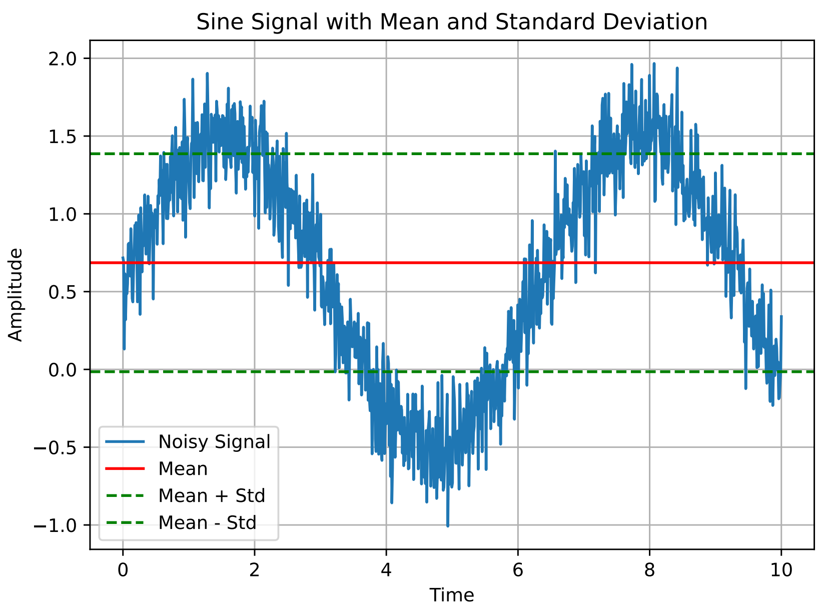


Figure 2.6: Sine Signal with mean and standard deviation

The variance σ^2 is a metric for the signal's behaviour. It expresses the mean squared deviation from the mean.

Time

$$\sigma^2 = \frac{1}{T} \cdot \int_T [x(t) - \mu]^2 dt \tag{2.3}$$

$$\sigma^2 = \frac{1}{N} \cdot \sum_{i=0}^{N} [x_i - \mu]^2 \tag{2.4}$$

The standard deviation is derived from the variance. Its value gets square-rooted to better represent the power (magnitude of the amplitude).

Standard Deviation

$$\sigma = \sqrt{\sigma^2} \tag{2.5}$$

Mean and the standard deviation don't represent the desired metrics in some use cases. Rather more important is a comparison between the two. Hence, the Signal-to-Noise ratio (SNR) is used to compare and condense the mean and standard deviation by dividing the mean by the standard deviation.

$$SNR = \frac{\mu}{\sigma} \tag{2.6}$$

Another parameter is the coefficient of variation (CV) which is the standard deviation divided by the mean and multiplied by 100%.

$$CV = \frac{\sigma}{\mu} 100\% \tag{2.7}$$

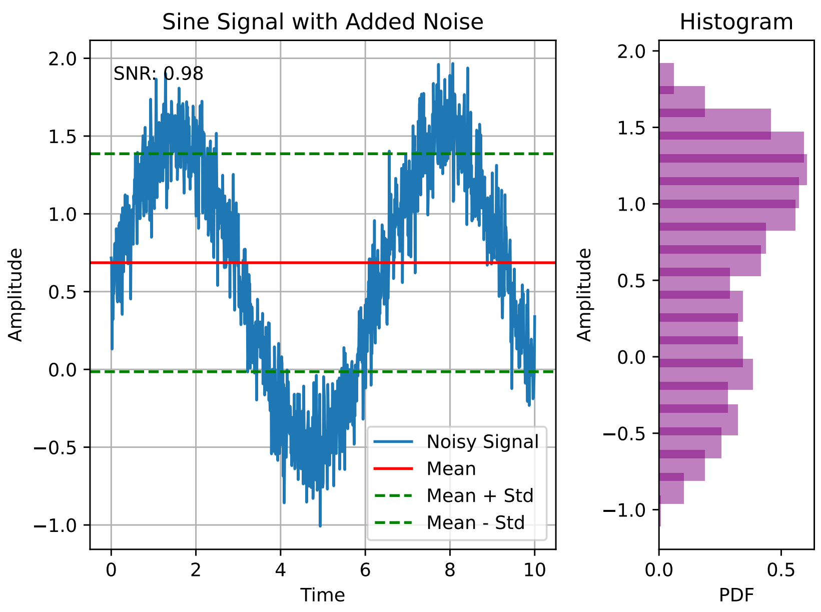


Figure 2.7: Sine Signal full analysis with mean, stdev, SNR and Histogram

An arising problem based on the SNR and CV are however that they scale based on the mean value. Should the mean value lie at about 0 for e.g. a sensor of an aircraft control surface, the signal to noise ratio will be relatively high compared to an acceleration sensor in z axis with a constant offset of 1g

Practical example for mean and standard deviation in a given signal are overlayed in figure 2.6

To evaluate a signal according to the quantities the next logical step for statistic Histogram

probability mass function

$$f(x) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(x-\mu)^2}{2\sigma^2}} \tag{2.8}$$

Covariance Zeitkontinuierliche Autokovarianzfunktion

$$\gamma(\tau) = \frac{1}{T} \cdot \int_T [x(t) - \mu] \cdot [x(t-\tau) - \mu] dt$$

Zeitdiskrete Autokovarianzfunktion

$$\gamma(k) = \frac{1}{N} \cdot \sum_N [x_i - \mu] \cdot [x_{i-k} - \mu]$$

–>Autocovariance autocov(x,x) = cov(x,x)

Correlation (Pearson coefficient)

$$\rho_{x,y} = corr(x,y) = \frac{cov(x,y)}{\sigma_x \sigma_y}$$

The maximum of a total correlation is 1 or that of a total negative correlation -1.

Total independence for $\rho_{x,y} = 0$ not given since Pearson coefficient only detects linear correlations

Other correlation methods as rank-correlation (Spearman, Kendall) that detect change correlations are possible but are more complex in the implementation.

2.3 Previous works

2.3.1 PCA

What is PCA (Intro)

The principal component analysis (PCA) is a method that allows to reduce the information parameters of a dataset into a few principal components. Generally this is specified as the operation of the original dataset X_{Nxm} with the timesteps N and the measurement parameters m into the transformed principal components (PC) T_{Nxr} with r being amount of PCs. This transformation is defined as:

$$T_{Nxr} = X_{Nxm} P_{mxr}$$

with P being the permutation matrix that rotates the original data into the principal components. The overall strategy to generate a PC is shown in figure 2.9 as finding a function between two parameters x_1 and x_2 that maximizes the variance of data in the newly generated principal component. However drawbacks arise using this approach since the variance optimization only happens linearly and is not able to function for nonlinear correlations.

-originally to reduce dimensionality within a dataset [HAND17] –>extract maximum variance (see variance chapter) from dataset ->transform onto 2 PCs

-Application in FMEA (Isermann, fault diag) [ISER06] ->determine number of principal components

What is PCA again? (Detail)

relative minimal input. Usable results without system knowledge. Works pretty much straight out of the box

the principal component analysis is based on correlation principles and allows reduction of datasets into independent components. Applications in an aerospace context include detection of previously unknown correlations between parameters and complex interactions in aeroelastics or otherwise large datasets. It also may allow generation of rules to detect a deviation from previously correlated values. Usage however is associated with a certain effort since information needs to be preprocessed to a certain degree, reducing dimensions previous to processing in order to avoid feeding in redundant parameters. Upsides of the PCA are that it acts as a blackbox that quickly allows to reduce data onto a given set of dimensions, requiring relatively little user input.

Generally, we want to transform the Matrix X into the transformed Matrix T containing all Principal components

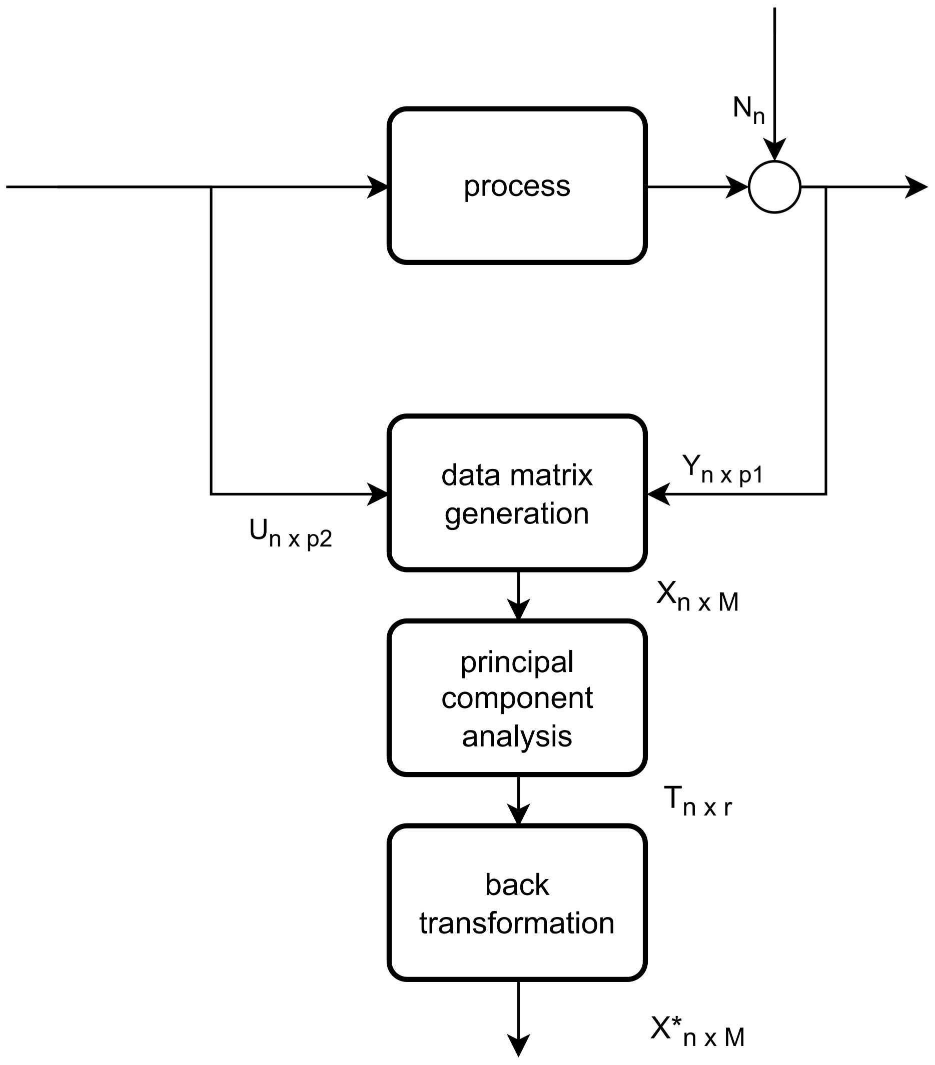


Figure 2.8: Known process input U as well as Sensor Measurement Y get collected into X and fed into a PCA. N indexes timesteps.[ISER06, p.268]

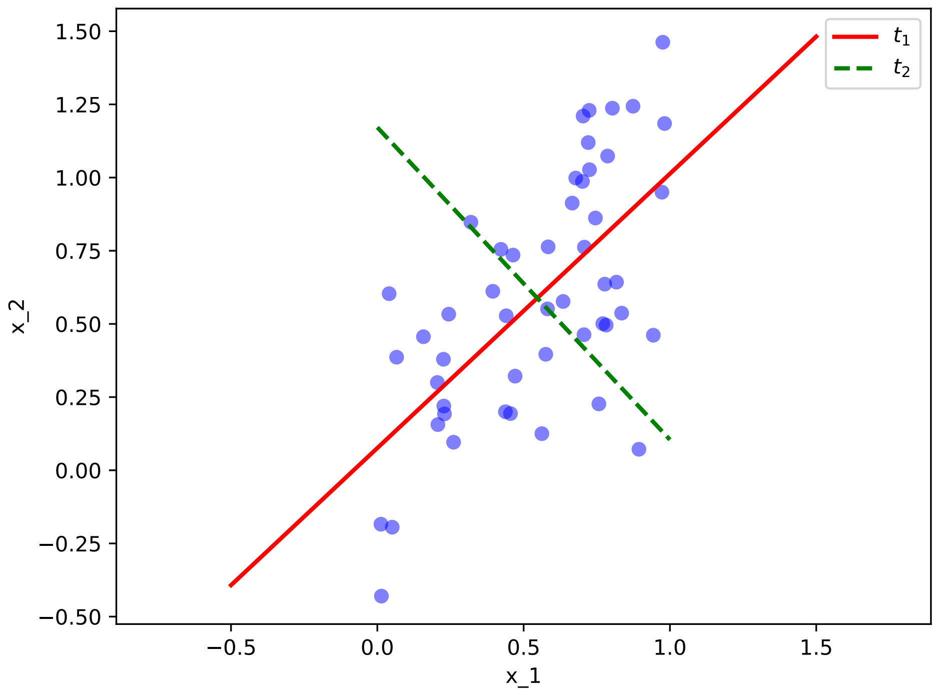


Figure 2.9: Parameters x_1 and x_2 get converted into principal components t_1 and t_2 by optimizing for minimal variance.

with the count $r < m$. To perform this operation we introduce the Permutation Matrix P leading to:

$$T_{Nxr} = X_{Nxm} P_{mxr}$$

Full derivations are available in previously cited sources (isermann, han17). Rather more important is however, that isermann presents a process for an autotuning PCA that can be used for real-time Fault Detection applications.

A principal component gets created by condensing two parameters in the direction of most variance to each other. This is a classical optimization problem

PCs can be created in any direction. Where input is needed however is the number of Principal components that shall be kept. A cutoff value can be defaulted to but this may not be feasible for multidimensional systems such as the flight test data that is at hand.

Eliminating a principal component becomes easy if parameters show a direct correlation such as in figure 2.9. The gatherings of t_2 then mostly consist of noise and can be safely discarded.

The PCA can be defined as a condensation of a dataset into fewer parameters. The so called principal components. These principal components can also be transformed back into the original parameters. This process can also be used to create a data model and then calculate the residual to possibly quantify the error.

2. create principal components by maximizing variance

3. for all

analogy to covariance matrix

SVD.

What is PCA (Concluding remarks. Lessons learned)

2.3.2 PTF

control systems approach

what are state values?

Following the recipe for a modeled aircraft based on sensor data we try to simulate the parameters x and u of the aircraft with the sensor data y. Khaled shows that this approach works for linking omega$_x omega_y, delta_d elta (drift angle) with omega_z . and Transmissibility function$T

The examined approaches include:

1. Transmissibility functions $\mathcal{T}$ that model the system output without having to take the unknown system input into consideration
2. Bond Graphs to model a physical rigid aircraft system
3. Physical relations
4. redundancies between sensors

model based fault detection(isermann) -parameter estimation (process modeling with linear or nonlinear functions, unknown process parameters are modeled by residual minimization) -parity equations () -state estimation (kalman), state/output observers (for known process parameters,) -principle component analysis

2.3.3 STFT

stft.

Test noise analysis over short time window of 256 samples.

Define here

Also, next to the simple and known models to display an error is the Luenberger Beobachter. enter placeholder for image: Measurement equals signal + error and image: luenberger beobachter. Kalman Implementierung (Lie13) Parameter Correlation Studies (Li15)

2.3.4 API

2.3.5 GNSS

The aircraft altitude generally is defined as the displacement of the aircraft from Mean Sea Level (MSL). On a geodetic scale, the earth can be described as an ellipsoid due to its rotation. However, varying density levels of the earth's crust cause the elevation and sea level to deviate from the ellipsoid shape. This results in a lopsided model that is modeled in the WGS84 (ref and image here) system. This is also the altitude that the gps measures. And will be the reference altitude for the following calculations.

The main existing altitudes are the:

1. Geodetic Altitude (GNSS)
2. Barometric Altitude (used in conjunction with reference pressure)
3. Inertial Altitude
4. Radar Altitude (can be used in conjunction with a terrain model to derive geodetic altitude)

Possible errors for each are: geodetic: inconvenient satellite placements, deflection of signals in the atmosphere and signal problems Barometric Altitude: Possible errors due to drift and meteorological atmospherical pressure shifts, Reference Altitude.

Going into WGS84 and the GNSS Altitude however exceeds the scope of this work. In the following, the satellite altitude above Mean Sea Level (MSL) is considered as the reference altitude.

Fault detection algorithms within GNSS are described as Receiver Autonomous Integrity Monitoring (RAIM) are tried and tested within GNSS implementations since high accuracy positioning is valuable for various applications, reaching precisions of up to a few centimeters. Explaining the full function of position calculation exceeds this works' scope. To summarize however, GNSS inputs form an overdefined system of equations which needs to be compensated within some algorithm to form one position based on multiple inputs.

ica18 annex10 ABAS, RAIM, AAIM

RAIMS

Examination of Receiver autonomous integrity monitoring (RAIM) from GNSS applications.

basic principle 1 in 1 out basic

2 in 1 out mean solution. Detect discrepancies 1. simple solution: take average 2. detect discrepancies but still take average

3+ in 1 out. 1. take average 2. detect value with strong variations and isolate (it is assumed that only 1 sensor is faulty) 2.1 predict value and deny value if it is larger than 3 times standard deviation (Wen07, 239).

implementation details. see [Bro92]

2.3.6 The barometric altitude based on the International Standard Atmosphere (ISA) [ISO75]

The international standard atmosphere is the conventional method of measuring an aircrafts altitude based on air pressure. The method can be understood as a function of pressure and reference pressure (differing based on weather) that returns an altitude. In the following this equation based on fundamental scientific correlations will be derived.

In the beginning, a finitesimally small element of the atmosphere is considered that is in equilibrium. It has pressure acting upon it from all its sides. The pressures acting upon all its sides generate a force that cancels out within the horizontal directions. It is notable however that the pressure on the top marginally differs from the pressure on the bottom. This arises from the elements desire to remain in stationary as it is attracted by gravity. Were the pressure difference on top and bottom equal, the element would begin moving towards the origin of the gravity vector.

Since we are interested in the difference in pressure we formulate the force equilibrium for direction h in equation 2.9.

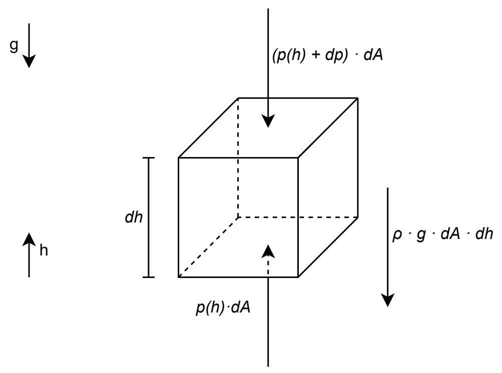


Figure 2.10: Forces around a finitesimally small element

Please note that the original gravity term of course is $\rho \cdot dV \cdot g$ but we can transform it to $\rho \cdot dA \cdot dh \cdot g$ using the correlation $dV = dA * dh$ with dA being the area of the elements top and bottom side.

From equation 2.9

$$\sum F_h = 0 = p(h) \cdot dA - (p(h) + dp) \cdot dA - \rho \cdot g \cdot dh \cdot dA \tag{2.9}$$

follows 2.10.

$$dp = -\rho \cdot g \cdot dh \tag{2.10}$$

ρ can be taken out of eq. 2.10 using the perfect gas formula $p = \rho RT$ and equation 2.11 emerges.

$$\frac{dp}{p} = \frac{-g}{R \cdot T} \cdot dh \tag{2.11}$$

At this point we need to introduce the Temperature model of the ISA. It assumes linear temperature gradients for each layer of the atmosphere, meaning that each atmospheric layer can be modeled using a single coefficient that models the gradient. In the following, this gradient will be used as $a = dT/dh$. This allows modeling the temperature for each atmospheric layer with equation 2.12.

$$T(h) = a \cdot h + T_0 \tag{2.12}$$

Atmospheric Layer	Geopotential Altitude [km]	Temperature T at bottom [K]	Temperature gradient a [K/km]
Troposphere	-2-0	301.15	-6.5
Troposphere	0-11	288.15	-6.5
Tropopause	11-20	216.65	0
Stratosphere	20-32	216.65	1
Stratosphere	32-47	228.65	2.8
Stratopause	47-51	270.65	0
Mesosphere	51-71	270.65	-2.8
Mesosphere	71-80	214.65	-2

Table 2.6: International Standard Atmosphere cite ISO75. Atmospheric Layers model the average physical correlations and do not represent real atmospheric values which vary greatly based on latitude and local factors.

Using the different coefficients from table 2.6 we now can model the Temperature based on a standard temperature of 15° Celsius or 288.15Kelvin at Mean Sea Level (MSL), meaning that $h = 0$. The temperature gradients drastically simplify real circumstances but work as an empirical estimate that is good enough to guarantee a common understanding of barometric altitudes. This is especially important for aeronautic applications since altitudes for directing flights are generally given in barometric altitudes.

Now, substituting the ISA-equation 2.12 into 2.11 delivers 2.13.

$$\frac{dp}{p} = \frac{-g}{R * (a * h + T_0)} * dh \tag{2.13}$$

To get this differential equation into a regular formula we need to integrate which resolves into equation 2.14.

$$\ln(\frac{p}{p_0}) = -\frac{g}{a \cdot R} \cdot ln(\frac{a * h + T_0}{a * h_0 + T_0}) \tag{2.14}$$

The borders chosen for integration are the lower border of the atmospheric layer referenced with the index 0 and the other layer being the running variable.

Substitution for the single factors follow in equations2.15.

$$\ln(a) = b * \ln(c) \tag{2.15}$$

$a = p/p_0$

$b = \frac{-g}{a \cdot R}$

$c = \frac{a \cdot + T_0}{a \cdot h_0 + T_0} = 1 + \frac{a \cdot (h - h_0)}{a \cdot h_0 + T_0} = 1 + \frac{a \cdot (h - h_0)}{T(h_0)}$

Subequation c is then simplified by pulling T_0 into the denominator and substituting the ISA-equation 2.12.

After substituting, the term 2.16 emerges.

$$\ln(a) = b * \ln(c) = \ln(c^b) \tag{2.16}$$

Brief transformation then resolves into 2.17.

$$\rightarrow a = c^b \tag{2.17}$$

Resubstituting then gives equation 2.18, the barometric altitude equation.

$$\frac{p}{p_0} = \left(1 + \frac{a \cdot (h - h_0)}{T\left(h_0\right)}\right)^{\frac{-g}{a \cdot R}} \tag{2.18}$$

p	= pressure at current altitude [Pa]
p_0	= reference pressure [Pa]
h	= current altitude [m]
h_0	= reference altitude [m]
$T(h_0)$	= temperature at reference altitude[K]
a	= ISA Temperature Coefficient depending on altitude $(-6.5e-3)[K/m]$
g	= gravity constant $(9.80665)[m/s^2]$
R	= Ideal Gas Constant $(287.05287)[m^2/(K \cdot s^2)]$

Transforming this formula for altitude resolves into equation 2.19

$$\rightarrow h = \left(\frac{p}{p_0}^{\frac{-a*R}{g}} - 1\right) \frac{T(h_0)}{a} + h_0 \tag{2.19}$$

Barometric equation for constant layer temperature

This formula however is not valid for constant layer temperature and $a = 0$. So the barometric equation 2.13 needs to adapted to $a = 0$ **before** integration. From equation 2.13 henceforth follows equation 2.20.

$$\frac{dp}{p} = \frac{-g}{RT_0} \cdot dh \tag{2.20}$$

Integrating then yields 2.21:

$$\ln\left(\frac{p}{p_0}\right) = \frac{-g}{RT_0} \cdot h \tag{2.21}$$

And resolving for pressure into equation 2.22 follows into the barometric equation for a constant temperature coefficient.

$$\frac{p}{p_0} = exp\left(\frac{-g}{RT(h_0)} \cdot (h - h_o)\right) \tag{2.22}$$

The following equation 2.23 then resolves for the altitude based on pressure ratio within a layer with constant temperature.

$$h = ln(\frac{p}{p_0}) \cdot \frac{RT(h_0)}{-g} + h_0 \tag{2.23}$$

Process to determine altitude

The process to then actually calculate a barometric altitude is displayed in figure 2.11. The input is p and p_0 with p being the measured pressure via the aircrafts static ports and p_0 being the reference pressure that is manually set by the pilot with the standard pressure being $p_{std} = 1013.25hPA$ which varies according to the weather.

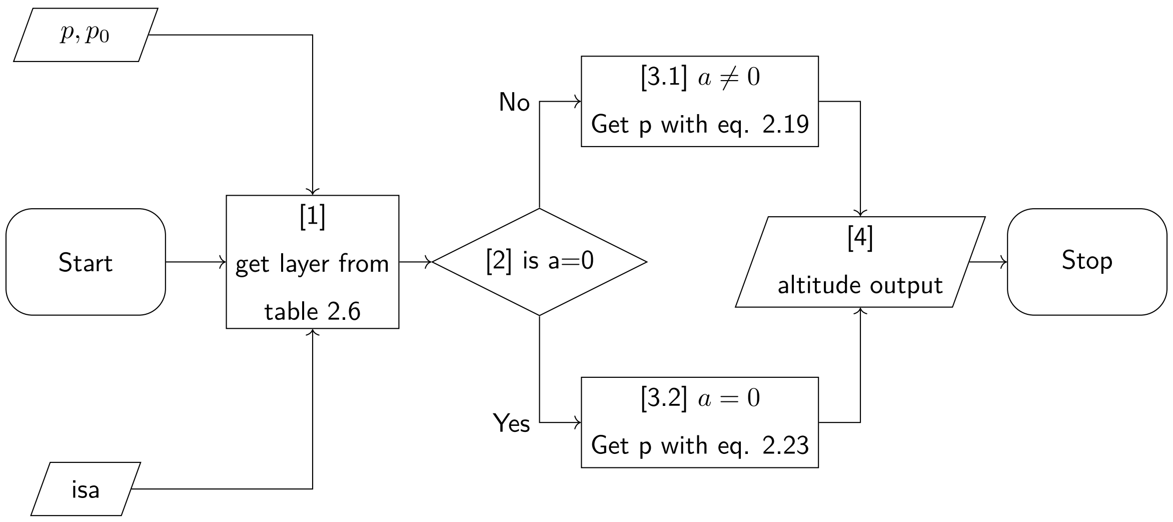

p, p_0
Start
[1] get layer from table 2.6
isa
[2] is a=0
No
Yes
[3.1] $a \neq 0$ Get p with eq. 2.19
[3.2] $a = 0$ Get p with eq. 2.23
[4] altitude output
Stop

Figure 2.11: Process for determining barometric altitude from pressure and reference pressure only.

The current atmospheric layer gets found by matching the MSL-pressure ratio $\frac{p}{p_0}$ to precalculated boundary values of layers based on table 2.6. After determining the layer and the value for temperature coefficient a, the altitude can quickly be determined by inserting layer properties as well as the pressure ratio into their according altitude equation (step 3.1 and 3.2 in flowchart 2.11). In the final step 4 the altitude is returned.

Notes: The ISA Norm defines barometric equations for both geometric as well as geopotential conditions. Both start at MSL but geopotential altitude assumes equal acceleration g for every altitude. Since its value as an industry-standard, the gepotential altitude is considered in this work. For completeness the correlation between geopotential and geometric altitude is given in equation 2.24. This formula is based on a derivation of gravity potentials and a full derivation can be found in ISO [ISO75].

$$h_{geometric} = \frac{r_{earth} \cdot h_{geopotential}}{r_{earth} - h_{geopotential}} \tag{2.24}$$

2.4 Structuring large datasets

2.4.1 Data Format, FAIR

Originating from a dutch alliance of biotechnology institutes in 2015, the FAIR Guiding principles emerged in 2016 as a general best practice guide for research data, referencing best practices

FAIR principles have been published in 2016 by the source here. They set the foundation for a standardized and open data culture. Within these principles values like open access for data, findable and well tagged datasets, interoperable data by using standardized formats and or semantics (define semantics as well) which guarantee a reusability of data to generate a sustainable process for data usage. Effectively meaning that similar experiments do not have to be performed multiple times when well tagged and formatted data is freely available.

json file format (FAIR-principles, INST-DLR, SOIL)

Explain which means are taken to guarantee an architecture throughout the work that ensures an implementation of the FAIR principles and the V-model. Architecture of the JSON-Tree structure and which data is inserted where.

-modern data management principles (storage not as important as readability)

Comparison with existing data structures. Implementation into existing architecture

–> Chosen Data Model, Semantics

2.4.2 SOIL

2.4.3 Skystash

Flight test data needs to be perceived within the context of its circumstances to extract all its information.

dlrk paper hier zitieren.

this work(flight data, sensor positions, isa table)

!!!!!!!!do a part page design here.

the skystash is a platform to distribute, analyze and visualize large flight data sets. It is currently in development within the DLR's digital twin project and aims to be a service platform for uploading and sharing the DLR's flight test data. Various requirements are posed from different stakeholders such as the topic of sampling rate. Flight guidance projects may be content with sampling rates as low as 1 Hz. Other stakeholders such as aeroelastic experts may require sampling rates as high as possible and may reach up to 2000Hz within the ISTAR. For this purpose a dynamic data export is sensible in which users can directly choose sampling rates as well as single parameters from a flight contrary to downloading and working with a whole flight data set averaging up to 2GB of data.

placeholder

This work builds upon the python api and tries to implement and test an algorithm that checks the data for the istar aircraft

file sizes too large –> Reduction for on demand parameters and resampling utility Handling of large data sets

2.4.4 optimal data structure

data metadata needs to be efficient

2.4.5 A word towards unit management

Units, especially within aerospace contexts, are far from standardized. Most scientific users prefer SI-units. However, the convention for Pilots and Flight Test Engineers remains imperial within units such as feet, Nautical Miles, knots and so on. Table **??** gives an overview over the conversions that are used within this work. Naturally SI-units are chosen for calculations since calculations are more efficient as well as less prone to errors in SI-systems.

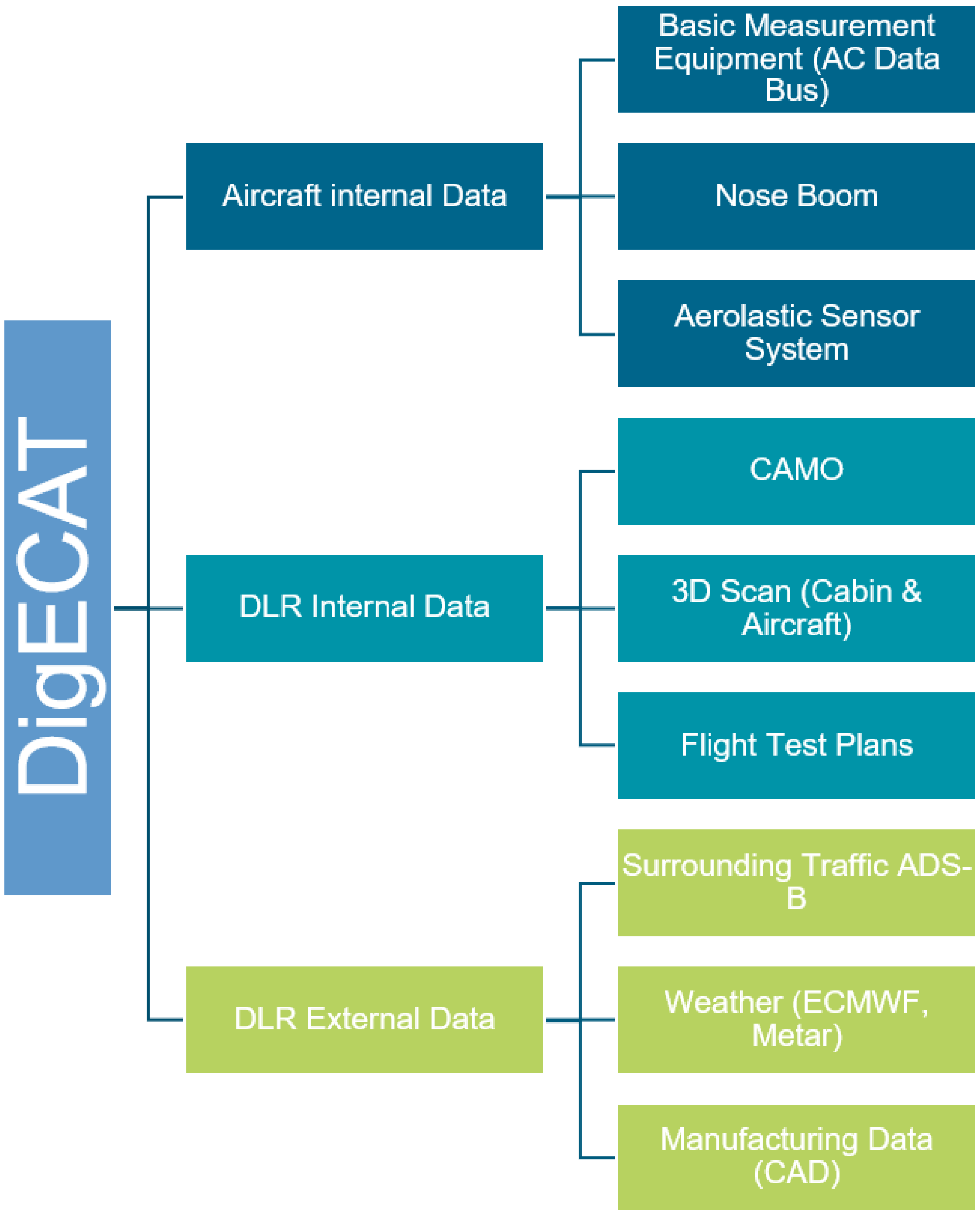


Figure 2.12: Generation of flight test data [ARTS]

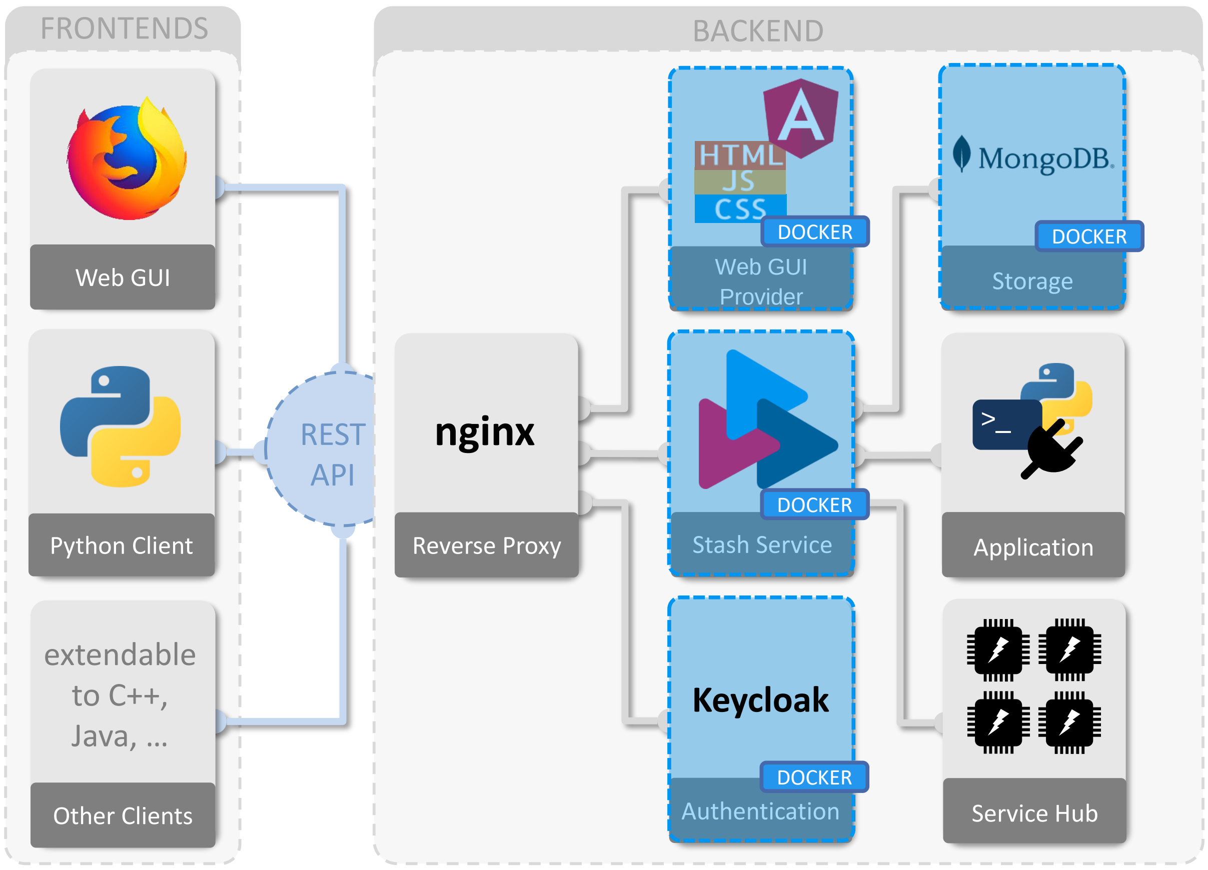


Figure 2.13: The Skystash architecture [ARTS]

2.5 conclusion theory and basics

BAsed upon these fundamentals, the effort is undertaken to implement the Sensor Health Monitoring

3 Problem Statement

- prototype for a complex sensor health monitoring structure. -employing standardized data classes -metadata semantics (SOIL) -pandas series -employing standardized interfaces for modular expansions

-structuring

▪ Precise detection of malfunctions and perturbances ▪ Reporting tool that displays relevant info and generates metrics Plug and play philosophy

3.1 Concrete Problem

existing proposal:

Sensor Health Monitoring Sensors fail. And have unexpected outputs. Not catching these faults can sometimes come at great costs when a research campaign proves to be useless when no sensible data has been produced. And even greater costs when the data acquisitioning systems have been changed and reconfigured for the next campaign. To solve this, an automatical fault detection/Health Monitoring of sensors shall be investigated and developed.

Motivation and problem To create checks, the sensors have to be checked against a configuration database. As straightforward as it sounds this proves to have some challenges to it as the existing configuration formats need to first be understood and then transformed into a format in which they can be read by the python code. Of course, this step needs to be dynamic and expandable for different configurations since the aircrafts' configuration may change frequently. A nomenclature and procedure can be formulated upon segmenting the problem of faulty sensors into different fault catching steps. 1. Production of data 2. Production of data that is within an expected range of values 3. Production of data that acts sensible in reference to other sensors Since metadata regarding Steps one and two like the Sensor names as well as their ranges can be defined in the configuration database, the main work to solve these procedures lies in the creation of a thorough configuration management. Systems which simulate a dynamic model as well as neural networks or other algorithms can be implemented in step 3 for finding correlations between sensors and detecting perhaps previously unknown correlations.

Approaches Sensor Health Monitoring is a multilayered problem whose challenges not only lie in the single operational tasks themselves but also the integration into the existing infrastructure of the sensors. Since the DLRs aircraft sensor configurations are subject to change a database shall be devised that does not create an additional location for configurations to be updated. Thus, a dynamical sensor health monitoring over changing sensor configurations is proposed that at least shall be designed with the architecture in place to enable such a design Changing configurations may be incorporated into a standardized sensor configuration management system. Research has to be conducted on how this database of sensor metadata is going to be implemented in the best way. The SOIL system shall be examined closer for this task and rated and perhaps modified according to its' abilities. Once a sound sensor configuration database is found, simple checks on SHM Level 1 and 2 should be conducted without great effort. For Level 3 approaches, a systemic approach will have to be found that interfaces the sensors to a physical correlation model and then back to themselves. Simple physical correlations can be a starting point for more complex algorithms like control system approaches based upon Pseudo Transfer Functions. These would be able to reference the State Space variables to the sensor outputs without knowing the systems inputs. Also possible is the use of neural networks. However, the control systems approach is chosen first as it offers more transparency on its logic and hence facilitates the debugging process. Once satisfying solutions are reached in a simple version of the Level 3 approaches, additional sensors may be added while recursively checking that solutions remain sensible. A possible strategy would be to rebuild the approach from Aircraft Sensor Health Monitoring for omega$_{z}derivedfromomega_{x}, omega_{y}anddriftangle :$ $delta_{b}eta.Furtheruseofthismodelcouldbeexpandedonceithasbeenproventowork.$

Validation Debugging a fault detection for aircraft with around 400 sensors will certainly prove tedious. An approach using about 10 sensors might be easier to validate. Steps 1 and 2 will not need much validation after verifying the relatively simple logic. Step 3 however will create a physical model. This model can be validated by using known physical correlations to reference sensors to the simulated model and calculate the difference over an entire flight. This in turn can be investigated to validate the method. Further validation methods may be employed later on once more is known about the topic.

3.2 Methodology

The first consideration to be taken is that this is the development of a development system. Since SHM is a multifaceted problem that represents a grand undertaking with many moving parts and optimizable parameters, algorithms and systems that may be implemented, it is firstly considered necessary to develop an ecosystem and a process in which SHM algorithms may be tested, evaluated and optimized. To develop said ecosystem, the turbine model for development of Human-Machine Systems is featured [FLEM22]. Its process can be applied to this work without much changes. Starting from Ideation stage that represents the beginning phase as well as the first Chapter of this work. Within Ideation phase the project orientation takes place. Which goals are considered viable within the given constraints of time and manpower as well as ressources such as knowledge and available tooling. After Ideation Phase the vertical movement representing creativity and innovational ideas is centered into a level-headed assessment of the problem containing goals and boundary conditions. Using these condensed ideas and values, the Research and Technology Assessment Phase is kicked off, represented by Chapter 2 of this work in which various technologies are examined for usage within the given problem context. Based upon a wide and thorough research feasible ideas and approaches can now be arranged within an infrastructure to solve the given problem. Fitting together various approaches from the literature like puzzle-pieces and condensing the ideas into a more refined concept, concluding literature review and representing a first draft. Using this first draft, the development cycle can now be started which can also be represented by the spiral model displayed in figure 3.2. The challenge of this third stage is now to implement the draft given in the previous stage, this gets accomplished by bouncing from an exploration step in which the theoretical constructs from the previous steps are implemented into a review step in which the systems are rated upon usability. This also represents the procedure of chapter 4 in which the theory from chapter 2 is implemented and developed. After having then developed a prototype of this work we can assess the results for our test cases (sensor failures) in chapter 5.

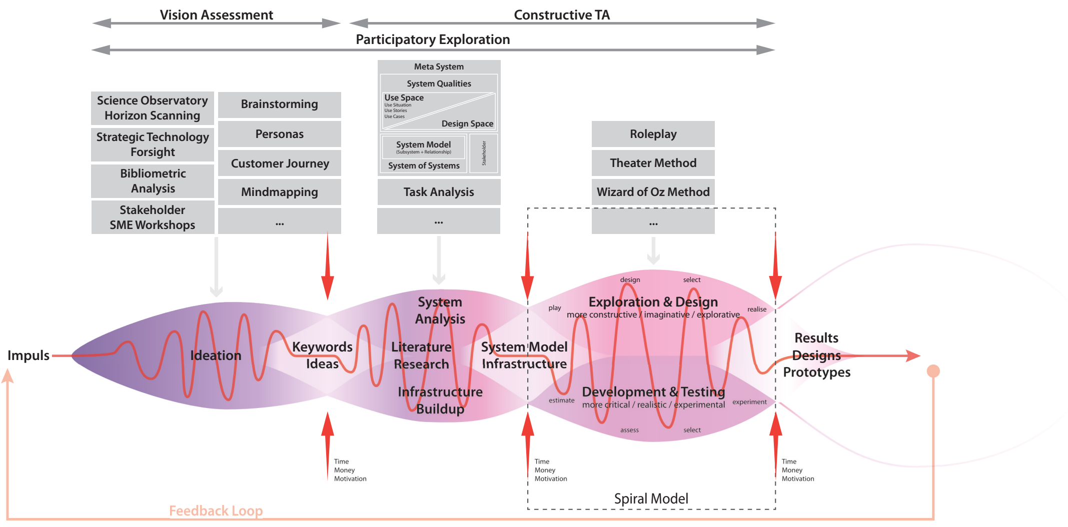


Figure 3.1: The innovation and exploration turbine in the context of this work as a two times nested feedback loop. [FLEM22]

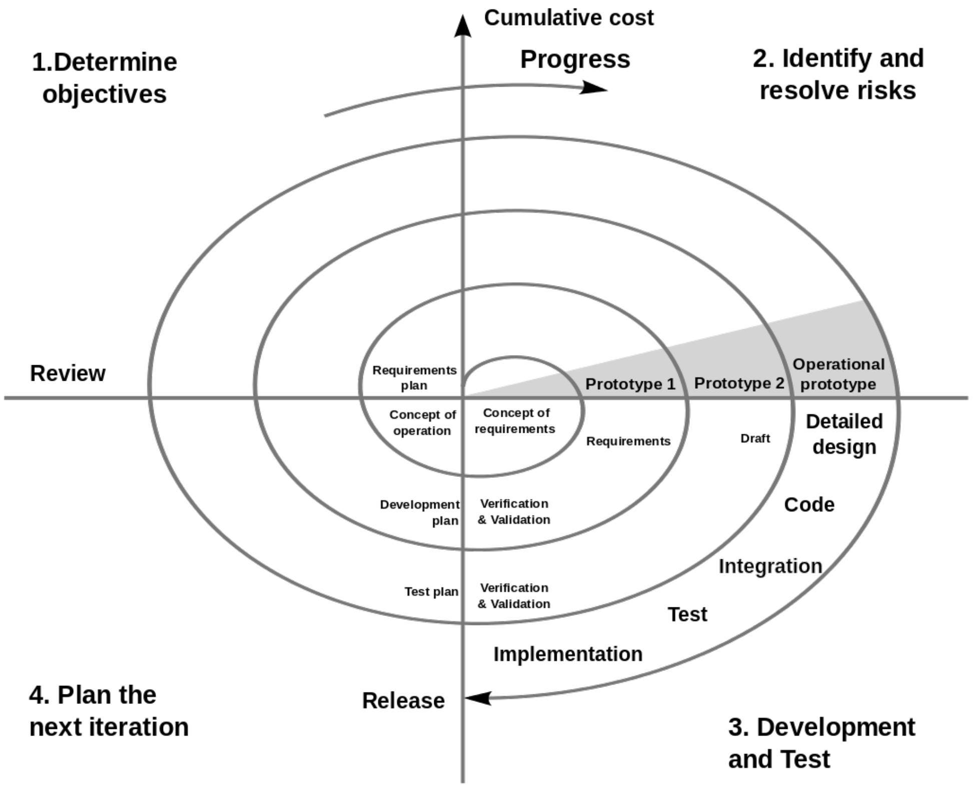


Figure 3.2: The spiral model featured within the innovation turbine [BOEH86]

4 Methods

The implementation efforts of this thesis are condensed into the following chapter. A holistic perspective on the data process is described in the following step following from the urgency that Sensor Health Monitoring needs to be perceived within its ecosystem. A SHM thus is directly reliant upon the data in the first place. It is then also reliant upon the configuration metadata since it will contain necessary information for processing the data further within the SHM. To then close the feedback loop it is also vital to generate a sensible data display that allows quick evaluation of the SHM since manual evaluation methods would slow the development process by orders of magnitude.

4.1 Introduction

The implementation can be detailed by figure 4.1. After having uploaded the Dataset the relevant information on the sensors as well as the aircraft properties is gathered and condensed into a single configuration metadata-set represented by a JSON file. This JSON file is then appended to the Dataset within the skystash architecture represented by the Online side that is accessed by the API. Processing the data now becomes a clean blackbox operation that is only fed by data received by the API returning its report containing notable occurences via the API and storing all relevant information online. Since this software is developed for flight operations and an interactive visualization of the report data facilitates evaluation of the generated report data a User Interface within a dashboard application is developed. Additional benefits contrary to the generation of PDF reports are dynamic updates as well as interactivity and an agile update cycle considering the spontaneous emergence of bugs.

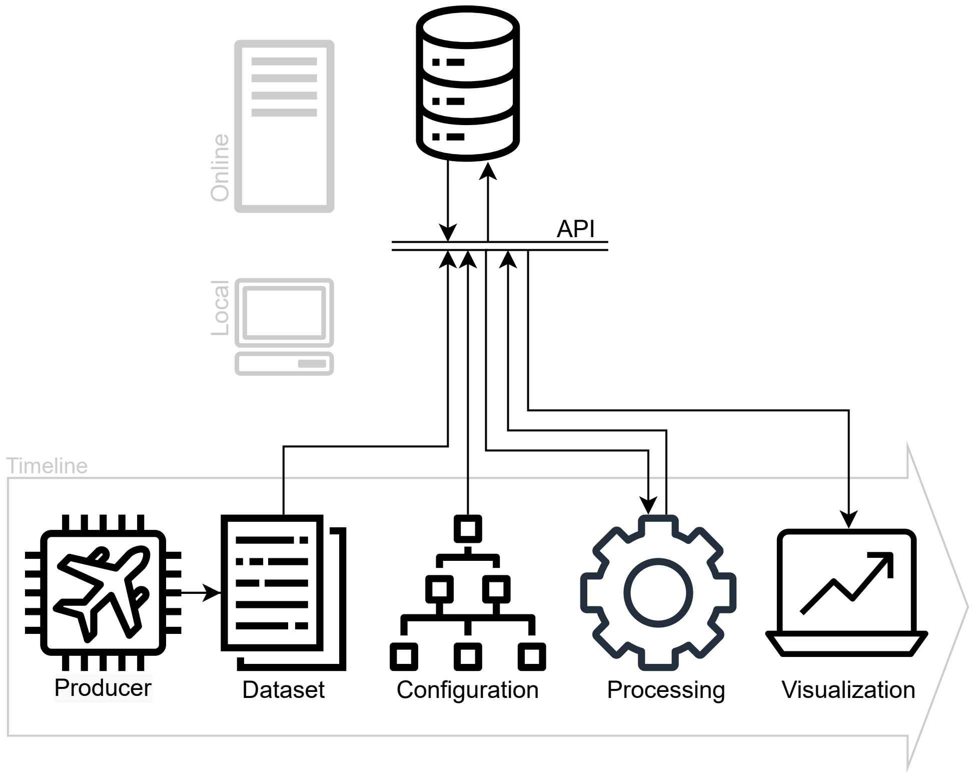


Figure 4.1: The data toolchain prospectively used for Sensor Health Monitoring

4.2 Data Parsing

Since the SHM has to run on something, the original data is briefly mentioned for completeness.

4.2.1 Parsing of ISTAR Data

Istar data is parsed for each flight for each sensor. Istar config data is present for each flight in the shape of an imcexp file.

ISTAR data originates from the ISTAR's DAQ system in the shape of .raw files for each parameter and is parsed and uploaded to the stash. It is then accessible via the stash api and can also be inspected in the stash webview. After the Dataset generation step the metadata available is only the one from the ISTAR DAQ system. The data is present in the shape of Projects->Collections->Series

The metadata is currently only the one from the ISTAR DAQ system. Since this metadata is not explicitly useful in itself it will be enriched further within the Configuration step (See figure 4.1).

4.3 Configuration-Metadata

get imcexp data enrich excel data -aircraft data -shm data -limits -type -tags -reference COS get excel data

generate merged view for metadata.

The metadata uploaded to the stash in itself is not very expressive and needs to be refined further into a usable, concise format. This step details the considerations and implementations taken to be able to generate a useful metadata model that applies necessary information to the dataset within the skystash.

4.3.1 Basics: Metadata Collection, formatting into SOIL

The configuration of the ISTAR is of great importance for the data processing. Also of great importance is the knowledge database which is currently in the shape of an excel document. Attributes that are generally transmitted within the DAQ configuration are sampling rates, data origins and other miscellaneous information related to the system. Not contained are information about general sensor description, position of sensors, overall setup of the aircraft sensor architecture and check limits for data values.

4.3.2 Parsing and assignment of ISTAR data config

-Additional Metadata -physical limits -amplitude limits -tag (physical entity) -reference ()

4.4 Data Processing

4.4.1 Software architecture considerations

Careful consideration needs to be given to the workflow of the level 1 check to allow scalability and minimal manual interaction in later stages. To achieve this architecture, manual steps are reduced as far as possible. However, some level of configuration must be implemented. Otherwise only relational sensor behaviour could be detected. Meaning that sensor faults occuring temporarily within an experiment can be detected but permanent behaviour is not noticed by a relative algorithm

4.4.2 Check 1 Implementation

For implementing the level 1 checks, the measured data needs to be compared to the expected data. To gain insight into what the the expected data is, the configuration file of the data acquisitioning system needs to get parsed. Luckily, the DAQ's format is a .zip directory. The 7zip command line tool gets used for opening the configuration file since the configuration file's format is not fully complying to the zip standard making several python libraries fail during the process. This stems from an issue with the zip header and footer parts of the file that are not at expected places (i.e. the front the back). Once opened, the configuration file contains multiple files as well as an essential xml file that contains the needed sensor metadata.

Missing datasets can now easily be found by comparing the expected values from the configuration file to the actual generated data.

For the second step.

All Parameters present?

4.4.3 Check 2 Implementation

Since the IMC DAQ System resamples the 20Hz Sensors to a straight timeframe data gets lost in the process. This makes potentially noisy sensors output the same value twice in a row since the DAQ hasn't yet received the new sensor data and outputs the same value multiple times in a row. Even sensors with a high noise ratio such as the fine part of a gps latitude signal outputs the same value twice in a row which is highly unlikely to happen on a statistical basis. Since this occurs quite often the question arises if the actual sampling may be lower than the actual data.

No signal (value)

A check implementation counterchecking the sensors sampling that generates fault detections based on deviation from predefined sampling. This takes into account the DAQ config since it contains the current sampling rate. Minimizing false positives.

Out of range

Generally, predefined limits are checked within SHM Level 2. For the first step, values are checked whether they are within a predefined range. This means e.g. a range of -1000-40000 ft for barometric altitudes. or 1Pa-120000Pa for static pressure ports. Of course, this selection is biased and may not be accurate for most mission profiles like atmospheric research aircraft that cruise in altitudes of up to 45,000 ft MSL.

movement too high

The next category examining the sensor behaviour can be classified into sensor movement being too low or sensor movement being too high. Movement being defined within this context as the difference of a new sensor value to the previous one. Of course, this topic could be examined within a depth that may exceed this work such as estimating white noise using discrete fourier Transform Subspace Decompositions [HEND08] or statistical methods such as covariance operators.

The goal of this work however, is preservation of scalability, minimal user and configuration inputs. Hence, an approach using a STFT where the amplitude is averaged across all frequencies from its spectrum. This guarantees a generalized feature extraction, meaning standardization. Based on this spectral analysis, the logarithmic order of the variable can be estimated within its dataset. Previously, the frequency is assumed via time difference of start and end time divided by $n_{datapoints}$. The STFT also implements preprocessing steps that would otherwise be necessary such as translating the signal by its average value as well as scaling it by its standard deviation. Would one be interested in examining a signal using functions from a statistical toolbox, a similar result may be achieved by performing the previous steps of averaging and scaling the signal and then examining the variance within a moving window of the signal, similar to the STFT. The size of the moving window for such an analysis is defaulted to 256 data points for each signal. Certain issues with methods of moving windows are however, that they are not very meaningful for the beginning and ending of datasets within which the window is not fully occupied. This may lead to spikes for the STFT, which is why it is found that the STFT is generally more powerful to examine low sensor movement.

TODO: Comparison variance, standard deviation, mean noise. Scalability for other sensors.

Noise Tracking using DFT can be used to estimate noise combined with a strenuous effort Since this however would exceed the scope of this work, a simpler algorithm is implemented by comparing moving window variance as well as STFT overall amplitude estimations.

Limits of variance and stdev.

To better approximate the errors concerning white noise, a STFT is employed.

White noise approximation methods. Rolling window variance.

movement too low

4.4.4 Check 3 Implementation:

I lied about single source of config. Level 2 and 3 also have their own config.

avoid weighting by structuring checks into a tree format for independent aircraft state variables.

Aims of Level 3 Implementation are to model the aircraft's state in a reference system. This reference system shall be geodetic and fixed to the earth while the aircraft moves through it. Moving aircraft coordinate systems like the aerodynamic and the along track Coordinate system may be derived from its geodetic position using ?angles and velocities.

Interfacing: Clean interfaces are generated throughout the model. Enabling a standardized state vector x. Meaning that A remains standardized for any aircraft while B, U and L need to be adjusted for any changes to the aircraft or sensor data.

The integration step may be omitted in early design stages since the necessary equations and equilibriums of Forces and Moments could only be modelled linearly while neglecting various unknown factors like shifting CG due to fuel burn, actual Inertia of Aircraft and aicraft mass. Hence, this step may be implemented if time allows it.

Finding a reference state

Necessary to compare parameters is a common reference system. Hence, reference systems are examined upon suitability for usage of parameter transformations.

A starting point which shall be considered is the geodetic reference. it measures the aircraft's position by its displacement from the previously (ref?) discussed WGS84 system. Meaning that altitude gets measured as the

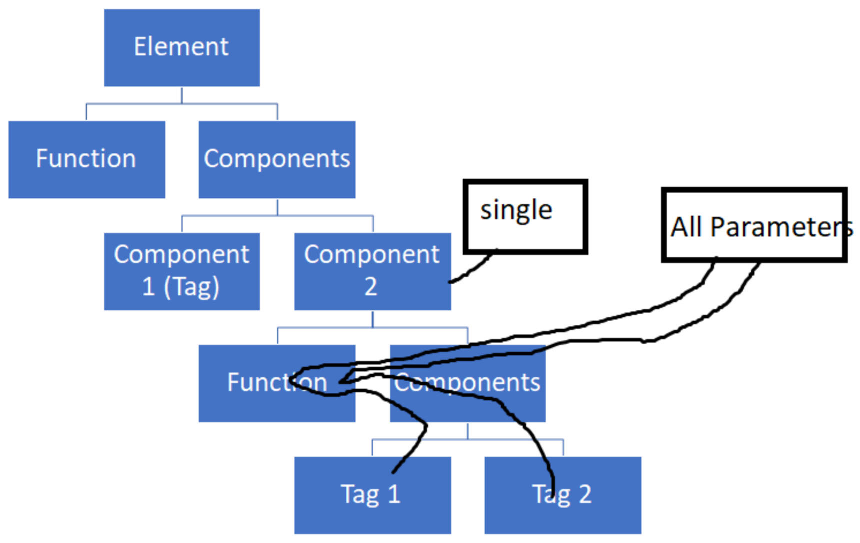


Figure 4.2: General structure of physical correlation setup

orthometric height (see ellipsoid height-geoid height).

Latitude and Longitude form the x and y axis of the COS. True heading forms the reference heading within the geodetic system.

All aircraft parameters related to motion and position of the aircraft should be attempted to be condensed into this form.

Dynamic configuration and correlation of parameters

Motivation: a dynamic description for parameter correlations is searched in order to implement an efficient and quick way to assign parameter correlations. It is then assumed that a tree structure is fit for this task since parameters can be correlated to other upstream parameters in a single flow direction.

To attach meaning to the parameters, they are specifically tagged with descriptions of their function. A simple example that is implemented is that of *static pressure*. The tag is defined in the excel configuration and furthermore gets configured into a dynamic configuration JSON-file. This file allows specification of sensors into subclasses and divides the aircraft into its degrees of freedom. This tree structure is then parsed and calculates a value for each level of detail taking into account its lower lying neighbours. Some weighting also has to be considered since a number of redundant pressure sensor could skew results against a single GNSS parameter. Thus a condensed parameter is calculated for each degree of freedom that is based on predefined algorithms and tags that can be freely defined within the config file that are then recursively parsed within the tree structure.

In figure 4.2 the general structure of the dynamic configuration in JSON format is shown. The configuration parent is an element similar to the configuration in SOIL-format [BODE21]. In our case the ISTAR-aircraft. The istar aircraft contains top-level components that are the independent state variables of the aircraft. Generally a

	A	B	C
A	0	0.25	2
B	0.25	0	1.75
C	2	1.75	0
Sum	2.25	2	3.75

component can contain other components or parameter tags that describe parameters that are referenced. A component can also contain a function that transforms its parameters into the parent parameter.

Residual generation

Based upon this previously defined model that is based on the aircraft configuration as a tree structure, residuals are generated based upon the difference of the parameters to the top level parameters (Single, fused value) and the single transformed sensor values (All Parameters) as previously discussed in Table 2.1.

An alternate method for fusion of redundant sensors is proposed that is similar to voting algorithms employed by Flight Control Systems of [TISC18]. An issue for voting algorithms is the abrupt exclusion of single sensors once their difference to the other sensors is too large. Hence, a weighting strategy for averaging is proposed that is based on the value difference.

An example is shown in figure **??**

This method tries to account dynamically for sensor distance by scaling parameters to small values compared to their neighbours. This works for a minimum of three redundant values.

Based on the single sensor values a distance matrix for $d_{i,j}$ is set up.

The column sum is then further defined $\hat{d}_i$. To generate a ratio we define:

$$w_i = \frac{1}{\hat{d}_i} \tag{4.1}$$

Since we desire that $\sum w_i \overset{!}{=} 1$ we need to scale the ratios using:

$$\bar{w}_i = \frac{w_i}{\sum w_i} \tag{4.2}$$

With this ratio we can now calculate the new fused value:

$$\bar{x} = \sum_{i}^{n} x_i \cdot \bar{w}_i \tag{4.3}$$

This works out to a fused value that is shown in **??**.

Compare results with averaging here using figure.

To further increase sensitivity for distance we can replace the term in equation 4.1 with a quadratic term:

$$w_i = \frac{1}{\hat{d_i}^2} \tag{4.4}$$

Residual Interpretation

The residuals are then checked against the limits of the standard deviation of the normal signal for a highpass filtering generated by taking a moving window of 256 points. If any value falls outside these values of the standard deviation of the condensed signal it is noted within the report. A fixed residual limit as well as a probability density function implemented in works such as [SVÄR14] may be implemented at a later point.

Examining Altitude Sensors

As seen in 4.3 gps altitudes have a base mean value of difference of about 3-4 meters. During flight level changes the base level changes significantly. Further investigation is needed how these changes may arise.

One possible approach would be considering different placements of gps antennas within the airframe. since the IMAR's position is precisely known and lies around the center of gravity but the ASCB's gps position is not known exactly the process is not facilitated. However using the process of lever arms the position could be roughly estimated and compensated from the altitude difference in figure 4.3.

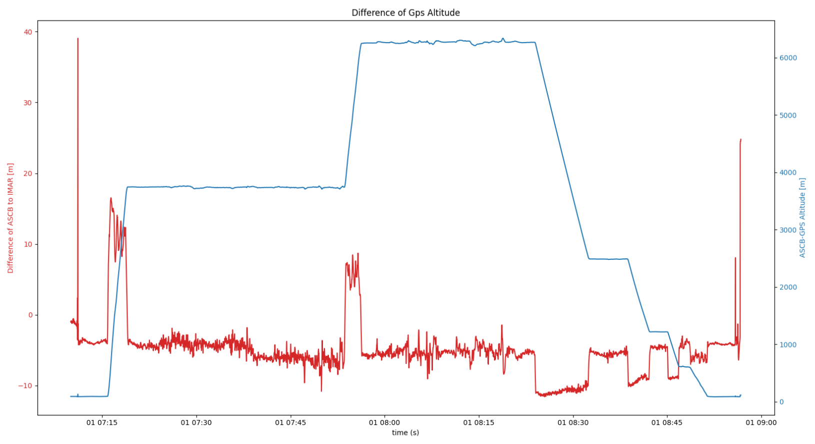


Figure 4.3: GPS difference for the aircraft base platform (ASCB) as well as experimental sensor (IMAR)

Another incurring deviation is investigating the 40m/150ft offset for gps altitudes. Possible causes may be Uncorrected Ellipsoid gnss altitudes. However, this appears unlikely since generally the offset in the region would be added and not subtracted [further investigation needed].

Difficulty diagnosing sensors without previous knowledge. limitations within sensor behaviour. Checks include: range, too much sensor movement, too few sensor movement.

Relative examinations possible for errors occuring for finite time within experiment but not if all of the experiment data is corrupt.

altitude comparison

plot here: -gps to gps -gps to baro. Beta overlayed as well as Ma Number

some mean ground has to be found to crossreference the various altitudes.

For standardization. The SI-unit meters is used for altitudes.

In the first draft. The altitude is sampled with 1Hz for computational speed and since gnss update rates are updated in a similar frequency.

Physical

GNSS

The following briefly glosses over GNSS and does not delve into the technical details. it rather tries to present GNSS Systems as a black box and examines inputs and outputs.

Generally speaking, GNSS systems calculate position based on run times to satellites. For output GNSS systems return position values for the ellipsoid model of the earth. Since the difference between MSL and ellipsoid varies for up to 100m vertically based on the receivers positions on the earth. Mostly, the difference is in the 40-50m range in germany.

Explain here why orthometric and ellipsoid value differs (vector difference, cg)

GNSS-units work with geoid models to deliver an orthometric value. Different Geoid models have different drawbacks and resolutions. For aviation use, the World Geodetic System (WGS84) generally proposes a geoid model that is used. Within this works scope it is omitted to implement an existing geoid model into the software-loop to reserve this workload for a later time.

4.5 Report Visualization

Motivation:

An interactive data report is chosen for data analytics since it works more efficiently and rather follows the paradigma of a single source of truth since it allows for dynamic updates and thus does not represent a data source in itself but rather view on the source.

Requirements: needs to be able to quickly gain an overview over state of sensors. Details can be looked up later. UX-centric design. Level1: Quickly see if many sensors are missing Level2: Identify outliers

4.5.1 Visualization

of missing parameter list

After monitoring, a list of missing parameters is generated. To show and detect this info quickly, a speed gauge style display is chosen to display this scalar value.

Displaying notable occurences of single sensor examination

To quickly get an overview of missing parameters, a timeline graph is implemented, showing cumulative errors over the flight. This allows a quick overview over all parameters.

Possible features: implement altitude graph later to contextualize sensor behaviour.

Examining sensor interactions

whole flight: occurences Single sensor: mean value, residual, stdev

It shows that while conservatively chosen ranges do not indicate many occurences, tighter limits enable more detailed monitoring.

4.5.2 Adapt sensitivity and algorithms (balance false positives)

The validation loop is driven by previously known error types (See OneNote Sensor Errors)

4.6 Conclusion

This parameter showed the methods used and the considerations made for the implementation of this SHM. Various methods and approaches were implemented to check for sensor faults. Next to the straight implementation, focus was put into developing solid interfaces to ease further implementation of future methods of fault detection. This applies for Levels 1, 2 as well as 3 which in this work was only minimally implemented to allow this prototype to see the light of day.

5 Results and Discussion

This chapter presents the findings of this work as well as the work that has been done and then discusses them in a wider context.

5.1 Metadata structure

Is this a sensible result representing reality and fulfilling demands in respect to working SHM and FAIR principles?

Thoughts on data structure flow.

A common ground has been found for the upload so that alterations to further processing parts of the cycle don't necessitate a full reupload of the original data. This common ground asserts that original is uploaded once with the downside that the DAQ configuration may not be fully interpretable at this stage due not being self describing metadata.

The following step of configuration solves this problem then by collecting and merging available metadata into a common scheme. This step works upon the DAQ metadata that is already uploaded to the skystash and then fuses it with other metadata sources that are locally available to generate a holistically valid metadataset. Notable is that this step will certainly be part of rather constant modification due to SHM configurations that are also provided within this step. Also the metadata schema is far from complete and will be in need of constant expansion to compensate the complex dimensions of flight test data as well as relevant test configuration data that also may be taken into consideration within this step 2.12. Since this step is far less computationally intensive than the parsing process this is considered as an acceptable tradeoff within this process.

This work's approach tries to implement the FAIR principles by using configuration files for the check algorithms that enable systemic control and extendability as well as by using a single source of truth, cleanly encapsulating every single step of the SHM process. However at the current point the metadata scheme is not clearly defined and work for a later point. Possible implementations may include the SOIL-scheme or other schemes that are possibly already suited for describing data quality and correlations. By also including the processing metadata configuration within the metadata, a reusability and traceability of the report logic is guaranteed.

An interactive frontend for the data report enables Accessibility of the SHM. Enabling a data quality overview for an entire flight dataset at a glance. Searchable metadata tags enable a dynamic search within the MongoDB Database.

5.2 Check testing 1,2,3 (test with real data)

Show plots of

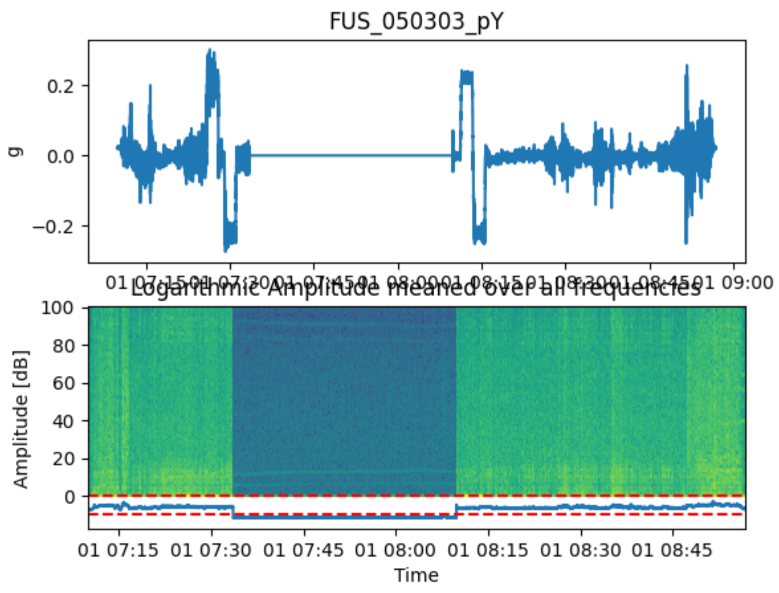


Figure 5.1: Capturing Errors

Check with faulty data (Plots and comparisons)

see onenote sensor errors

AE-errors F89: FUS$_0 50303$

-stft from onenote. show progression from spectrogram to averaged amplitude over time

20210923$_A VIA2terFlug_(1) : FUS_0 50303_p Y, FUS_0 50209_p Z(offset), DFUS_2 30306$

5.2.1 Comparison with previously developed FMEAs

PTF,Kalman,PCA

not nearly as good enough. However, focus of this work is to start off error detection within a FAIR context to generate interfaces for the skystash.

What could be reached with other FMEAs? Correlation Applications as well as Neural Networks.

5.2.2 Rate check quality with parameters

The data quality parameter fine tuning will still have to go a long way to fulfill all necessary and required conditions.

To summarize, the algorithms employed generally work by inspecting limits of parameter Series. These limits can also be set for modified parameter series that have e.g. been differentiated or fourier transformed. The interfaces have been created to allow for further implementation and optimization of parameters. The limits are defined within the configuration file (currently in excel format) and the Series transformation may also be customly defined.

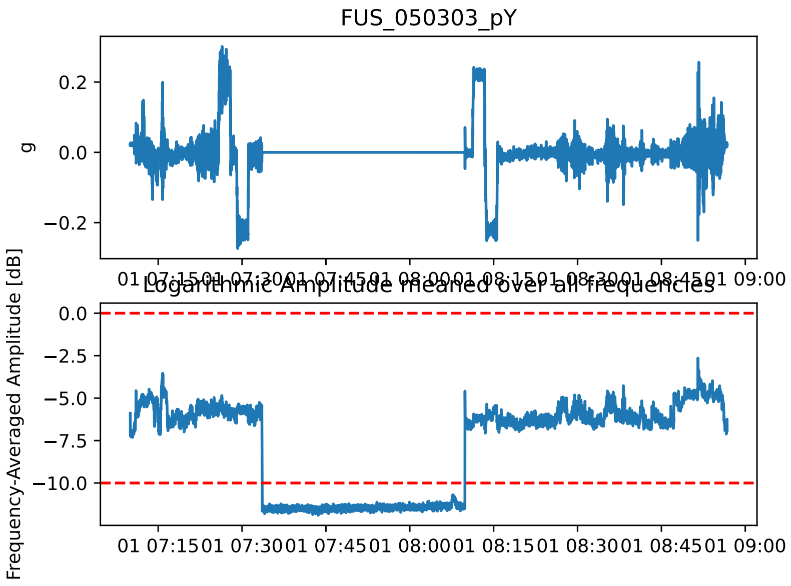


Figure 5.2: Capturing Errors

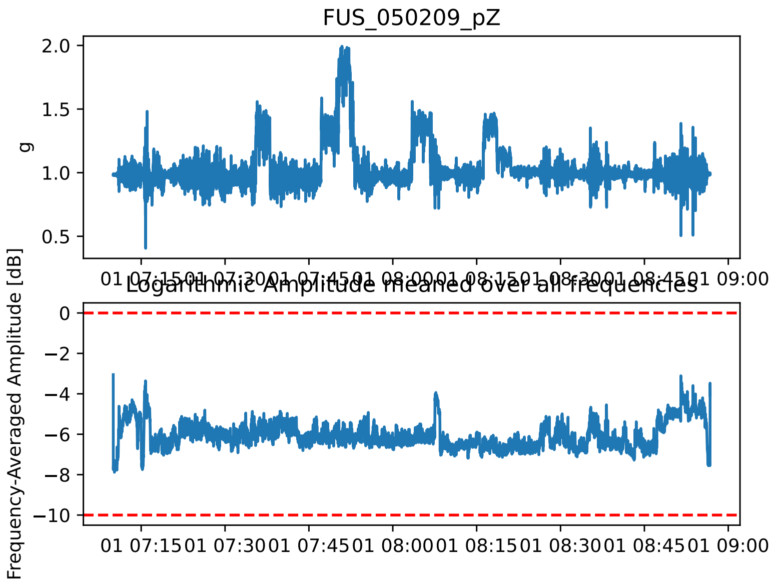


Figure 5.3: Capturing Errors

As for general formats it has been found to be difficult to generalize for all parameter formats since many employ strictly discrete values. For error detection using single sensors only, it is possible to divide into the following categories: Value is out of bounds, the movement of the value is to high or the movement of the value is too low.

In level 3, fault tolerant sensors methods are employed that use hardware as well as analytical redundancy. [ISER06, p.355-365]

5.3 Implementation ecosystem

How well is the implementation in the ecosystem? Modularity of components and extensibility with new functions.

-Embedding within the stash ecosystem. Clean Interfaces?

5.3.1 Toolchain FTI-tool

The following toolchain emerges for use of fti microservices.

Modular design allows encapsulation of algorithms as well as quick reaction towards faults and independence of fault modes. I.e. wrong config does not require the data to be reuploaded.

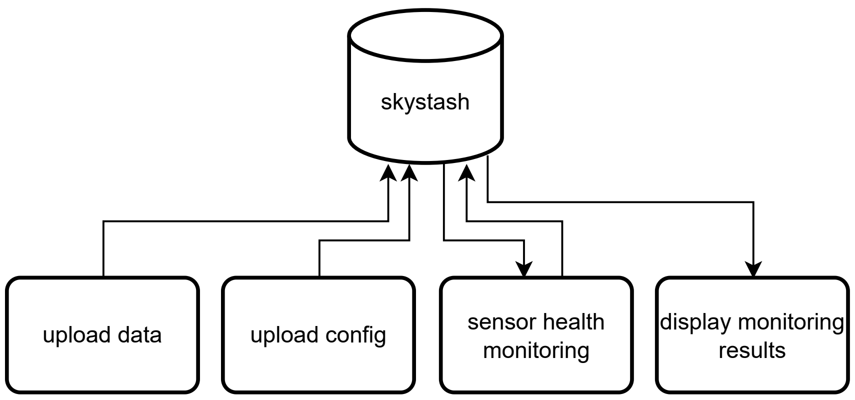


Figure 5.4: The emerging FTI-Toolchain

5.4 Discussion/Potential

Implementation list:tryouts -lvl3 rigid body simulation for flight dynamics -lvl3 implementation of multiple COS and their respective conversions -lvl3 residual interpretation (better than highpass filtering and checking for std)

-Real time implementation -Pseudo transfer functions (Generate a real time correlation matrix (MArkov parameters) and generate residuals based on that) -PCA -better parameter fusion algorithms (voting, or others see GNSS-RAIM) -aaim, -external factors integrity monitoring - GANomaly approach (neural network) since an aircraft dataset is similar to images in that it is a multidimensional dataset

-more and better evaluation of the report display to satisfy innovation turbine requirements.

5.5 Conclusion

These results show a promising prototype for a Sensor Health Monitoring Infrastructure. It establishes Interfaces and then

6 Conclusion

6.1 Stacked V-Model

This work per se can be modeled within a stacked v-diagram like in figure (stacked-v).

6.2 What has been done and accomplished in this work

6.3 What modular interfaces were built that can be extended in the future

References

[ARTS] E. Arts, M. Bäßler, S. Haufe, A. Kamtsiuris, H. Meyer, C. Pätzold, R. Schültzky, and M. Tchorzewski. "DIGITAL TWIN FOR RESEARCH AIRCRAFT". en. In: ().

[BODE21] M. Bodenbenner, M. P. Sanders, B. Montavon, and R. H. Schmitt. "Domain-Specific Language for Sensors in the Internet of Production". en. In: *Production at the leading edge of technology*. Ed. by B.-A. Behrens, A. Brosius, W. Hintze, S. Ihlenfeldt, and J. P. Wulfsberg. Series Title: Lecture Notes in Production Engineering. Berlin, Heidelberg: Springer Berlin Heidelberg, 2021, pp. 448–456. ISBN: 978-3-662-62137-0 978-3-662-62138-7. DOI: 10.1007/978-3-662-62138-7_45.

[BOEH86] B. Boehm. "A spiral model of software development and enhancement". en. In: *ACM SIGSOFT Software Engineering Notes* 11.4 (Aug. 1986), pp. 14–24. ISSN: 0163-5948. DOI: 10.1145/12944.12948.

[FAA08] FAA. *Federal_Radionavigation_Plan*. 2008.

[FLEM22] F. O. Flemisch, M. Preutenborbeck, M. Baltzer, J. Wasser, C. Kehl, R. Grünwald, H.-M. Pastuszka, and A. Dahlmann. "Human Systems Exploration for Ideation and Innovation in Potentially Disruptive Defense and Security Systems". en. In: *Disruption, Ideation and Innovation for Defence and Security*. Ed. by G. Adlakha-Hutcheon and A. Masys. Series Title: Advanced Sciences and Technologies for Security Applications. Cham: Springer International Publishing, 2022, pp. 79–117. ISBN: 978-3-031-06635-1 978-3-031-06636-8. DOI: 10.1007/978-3-031-06636-8_5.

[HAND17] A. Handl and T. Kuhlenkasper. *Multivariate Analysemethoden*. de. Berlin, Heidelberg: Springer Berlin Heidelberg, 2017. ISBN: 978-3-662-54753-3 978-3-662-54754-0. DOI: 10.1007/978-3-662-54754-0.

[HEND08] R. Hendriks, J. Jensen, and R. Heusdens. "Noise Tracking Using DFT Domain Subspace Decompositions". en. In: *IEEE Transactions on Audio, Speech, and Language Processing* 16.3 (Mar. 2008), pp. 541–553. ISSN: 1558-7916. DOI: 10.1109/TASL.2007.914977.

[ISER06] R. Isermann. *Fault-Diagnosis Systems*. en. Berlin, Heidelberg: Springer Berlin Heidelberg, 2006. ISBN: 978-3-540-24112-6 978-3-540-30368-8. DOI: 10.1007/3-540-30368-5.

[ISER11] R. Isermann. *Fault-diagnosis applications: model-based condition monitoring: actuators, drives, machinery, plants, sensors, and fault-tolerant systems*. eng. Berlin Heidelberg: Springer, 2011. ISBN: 978-3-642-12766-3.

[ISO75] ISO. *Standard Atmosphere ISO 2533.pdf*. 1975.

[KUTS75] F. v. Kutschera. *Sprachphilosophie*. 2., völlig neu bearb. u. erw. Aufl. Uni-Taschenbücher ; 80. München: Fink, 1975. ISBN: 978-3-7705-1182-2.

[PEÑA23] J. Peña-Consuegra, M. R. Pagnola, J. Useche, P. Madhukar, F. D. Saccone, and A. G. Marrugo. "Manufacturing and Measuring Techniques for Graphene-Silicone-Based Strain Sensors". en. In: *JOM* 75.3 (Mar. 2023), pp. 631–645. ISSN: 1047-4838, 1543-1851. DOI: 10.1007/s11837-022-05550-3.

[SHOE87] S. Shoemaker and S. Blackburn. “Spreading the Word.” In: *Noûs* 21.3 (Sept. 1987), p. 438. ISSN: 00294624. DOI: 10.2307/2215195.

[SMIT] S. W. Smith. “The Scientist and Engineer’s Guide to Digital Signal Processing”. en. In: ().

[SVÄR14] C. Svärd, M. Nyberg, E. Frisk, and M. Krysander. “Data-driven and adaptive statistical residual evaluation for fault detection with an automotive application”. en. In: *Mechanical Systems and Signal Processing* 45.1 (Mar. 2014), pp. 170–192. ISSN: 08883270. DOI: 10.1016/j.ymssp.2013.11.002.

[TISC18] M. Tischler. *Advances in Aircraft Flight Control*. en. Ed. by M. B. Tischler. 1st ed. Routledge, Apr. 2018. ISBN: 978-1-315-13682-0. DOI: 10.1201/9781315136820.