DAS Departamento de Automação e Sistemas
CTC Centro Tecnológico
UFSC Universidade Federal de Santa Catarina

Fault Detection for Mobile Robots based on Integrated Sensor Systems

Relatório submetido à Universidade Federal de Santa Catarina

como requisito para a aprovação da disciplina:

DAS 5511: Projeto de Final de Curso

André C. Bittencourt

Florianópolis, abril de 2009

Fault Detection for Mobile Robots based on Integrated Sensor Systems

André C. Bittencourt

Orientador:

Prof. Bo Wahlberg

Este relatório foi julgado no contexto da disciplina

DAS 5511: Projeto de Fim de Curso

e aprovado na sua forma final pelo

Curso de Engenharia de Controle e Automação Industrial

Banca Examinadora:

Prof. Bo Wahlberg
Orientador Instituto

Prof. Alexandre Trofino
Orientador do Curso

Prof. Augusto Humberto Bruciapaglia
Responsável pela disciplina

Prof. Ubirajara Franco Moreno, Avaliador

Daniel Fernandes de Souza, Debatedor

Caio Merlini Giuliani, Debatedor

Abstract

Most fault detection algorithms are based on residuals, i.e. the difference between a measured signal and the corresponding model based prediction. However, in many more advanced sensors the raw measurements are internally processed before refined information is provided to the user.

The contribution of this thesis is a study of the fault detection problem when only the state estimate from an observer/Kalmanfilter is available and not the measured residual/innovation. The idea is to look at an extended state space model where the true states and the observer states are combined. This extended model is then used to generate residuals viewing the observer outputs as measurements. Results for fault observability of such extended models are given. The approach is rather straightforward in case the internal structure of the observer is exactly known. For the Kalman filter this corresponds to knowing the observer gain. If this is not the case certain model approximations can be done to generate a simplified model to be used for standard fault detection.

Our motivating application has been mobile robots where the so-called pose, the position and orientation of the robot, is an important quantity that has to be estimated. The pose can be measured indirectly from several different sensor systems such as odometry, computer vision, sonar and laser. The output from these so-called pose providers are often state estimates together with, in best cases, an error covariance matrix estimate from which it might be difficult or even impossible to access the raw sensor data since the sensor and state estimator/observer are often integrated and encapsulated. In this thesis we discuss some of the main pose estimators in mobile robots and validate a fault detector filter through experiments using the our proposed framework.

Acknowledgments

This thesis was held at the Royal Institute of Technology (KTH) Automatic Control Department, Sweden. I would like to thank all my colleagues, students and employees, at KTH for the nice moments we shared during my stay in Stockholm proportionating a nice work atmosphere. Special thanks for my supervisor, Bo Wahlberg, for providing me the opportunity to join the project who, together with Patric Jensfelt at the Centre of Autonomous Systems, have shared their knowledge/expertizing and provided an always nice but challenging environment.

Finally, this work would have never come true if it was not for the help and support given from good friends and specially from my family. Thank you very much for everything.

Resumo Estendido

Grande parte dos algoritmos de detecção de falhas são baseados em resíduos, i.e. a diferença entre um sinal medido e uma correspondente predição baseada em modelo. No entanto, em muitos sensores mais avançados, as medidas puras são internamente processadas antes que informação refinada seja repassada ao usuário.

A primeira contribuição deste trabalho é o estudo do problema de detecção de falhas quando somente estimação de estados obtidas por um observador ou filtro de Kalman estão disponíveis, mas não seus resíduos/inovações. A idéia é olhar para um modelo de espaço de estados extendido onde os estados reais e os estados do observador estão combinados. Este modelo extendido pode então ser utilizado para gerar resíduos utilizando as saídas do sensor integrado e suas entradas como valores medidos. Resultados para observabilidade de falhas utilizando tal modelo extendido são dadas. A abordagem é consideravelmente simples em caso a estrutura interna do sensor é exatamente conhecida. Para o filtro de Kalman, correspondendo a saber o ganho do observer usado no sensor. Se este não é o caso, simplificações podem ser realizadas para gerar um modelo simplificado a ser usado na detecção de falhas.

Uma questão importante é discutir os ganhos utilizados usando-se tal modelo, indicações para o problema são apresentados através da análise das funções de transferência falha-resíduo, para os diversos casos. Especialmente interessante é a comparação entre as abordagens em que se utiliza do conhecimento da estrutura interna do sensor (ganho do observador, por exemplo) em contrapartida ao que se faz simplificações quanto ao sensor.

Nossa aplicação motivadora tem sido robótica móvel onde a conhecida pose do robô, sua posição e orientação, é uma importante grandeza a ser estimada. A pose pode ser medida indiretamente por diferentes sensores como odometria, visão computacional, sonar e laser. A saída destes “provedores de pose” são geralmente estados estimados, juntamente com, nos melhores dos casos, uma estimativa da matriz de covariância de erros, pode ser difícil ou até mesmo impossível de se ter acesso às grandezas diretamente medidas por estes sensores, uma vez que os mesmos estão geralmente integrados e encapsulados com observadores/estimadores de estados.

Neste trabalho, nós também discutimos alguns dos provedores de pose em robôs móveis e apresentamos e validamos uma framework para detecção e atuenação de falhas de localização em alguns cenários relevantes.

Um sistema de monitoramento de condição supervisiona um sistema dinâmico com o objetivo de:

- Detectar falhas: o sistema reconhece que uma falha ocorreu.
- Isolar falhas: o sistema reconhece onde e quando uma falha ocorreu (alguns sistemas incluem as funções de reconhecer o tipo de falha, o tamanho ou a causa da falha).
- Atenuar falhas: o sistema toma medidas necessárias para contra atuar os efeitos das falhas.

Nossa framework define cada uma dessas funções. Para tal, primeiramente, apresentamos dois provedores de localização utilizados comumente em robótica móvel, odometria e sobreposição de scans laser. Suas princípias características e alguns algoritmos são motivos de estudo. Os mesmos são utilizados em um robô móvel para na nossa framework que inclue as tarefas de detecção de falhas, estimação do seu tempo de ocorrência, estimação do seu tamanho e atenuação sobre as estimações de pose providas pela odometria.

As contribuições mais relevantes deste trabalho são:

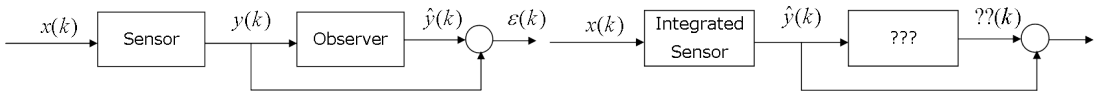
- As condições de detectabilidade de falhas para tais sensores integrados através da análise de observabilidade do sistema augmentando as falhas nas matrizes de dinâmica apresentadas no Capítulo 4.
- A análise da sensibilidade das funções de transferência falha-resíduo para as soluções propostas presente no Capítulo 4.
- A framework utilizada para detectar, isolar e atenuar falhas na localização de robôs móveis, objeto de estudo do Capítulo 6.
- O paper aceito para o 7th IFAC Symposium on Fault Detection, Supervision and Safety of Technical Processes incluso no Anexo A.

Summary

1 Introduction

Sensors and observers/estimators are often closely integrated in intelligent sensor systems. This situation is common in distributed sensor processing applications. It may be very difficult or even impossible to access the raw sensor data since the sensor and state estimator/observer are often integrated and encapsulated.

An important application of sensor based systems is model based fault detection, where the sensor information is used to detect abnormal behavior. The typical approach is to study the size of certain residuals (differences between a direct measured output $y(k)$ and a redundant model-based prediction of the same, $\hat{y}(k)$), that should be small in case of no fault, and large in case of faults. Most of these methods rely on the direct sensor measurements. The problem when only state estimates are available, as depicted in Figure 1.1, is less studied. In [1] the fault detection using such



(a) Raw measurements available. (b) Only an integrated sensor output available.

Figure 1.1: Residual generation problem arising when raw measures are not available.

sensors in mobile robotics is discussed from mainly an experimental point of view. The primary objective of this thesis is to investigate the theoretical foundation of observer data only fault detection, where it is not possible to directly access the raw sensor data.

Our motivating application has been mobile robots, where the localization task (estimating the robot positioning) plays an important role for increasing the general performance and reliability of the robot. The methods available today commonly provide an estimate which is the result of some internal computations from raw measurements, including in this category relative localization methods, .i.e dead-reckoning, and even global positioning systems, i.e. triangularization/trilateron. It may be difficult or or even impossible to access the raw measured data used in such methods and one should deal with the fault detection problem only with the estimates provided.

Description: Study the system state space description considering the following model:

$$x(k+1) \;=\; Ax(k) + B_u u_0(k) + B_v v(k) + F_a f_a(k)$$

where $x(k)$ denotes the states vector, $u_0(k)$ is a fault-free input to the system, $f_a(k)$ models actuator faults unknown inputs and $v(k)$ is process unknown disturbances.

The behavior of the system can be observed from sensors. To simplify the analysis we assume that sensors are integrated with standard observers/Kalman filters.

$$\hat{x}_j(k+1) \;=\; A\hat{x}_j(k) + B_u u(k) + L_j(y_j(k) - C_j\hat{x}_j(k))$$

The input to observer j is the measured output signal $y_j(k)$ and the input $u(k)$ which might also be subject to fault and noise as

$$\begin{aligned} u(k) &= u_0(k) + F_u f_u(k) + e_u(k) \\ y_j(k) &= C_j x(k) + F_{y,j} f_{y,j}(k) + e_{y,j}(k) \end{aligned}$$

The output from the observer is $\hat{x}_j(t)$, i.e., an estimate of the state. If for example the Kalman filter is used this could be come with a corresponding error covariance matrix

$$\mathrm{Cov}(\hat{x}_j(t) - x(t)) = P_j$$

Problem: We will study the problem when it is only possible to obtain $\hat{x}_j(k)$, and **not** the raw data $y_j(k)$. This seems to be a severe restriction, but from a practical point of view the measurement process could be integrated in the sensor system. One common example is standard GPS, where the measurement is based on satellite tracking and triangularization based techniques. In many applications the state estimate is obtained by more sophisticated methods then a simple linear observer. We will however use this structure for analysis and design purposes.

So far we have not taken the fault contribution $f(k)$ into account. One possibility is to also estimate $f(k)$ by for example extending the state vector to $\bar{x}(k) = [x(k)\, f(k)]^T$ and apply the Kalman filter or another observer method to estimate the extended state vector $\bar{x}(k)$. Recently, there has been quite a lot of progress in the area of input estimation using Kalman filtering, see [3].

1.1: Residual Based Fault Detection using Observer Data Only

Residual based techniques are all based on comparing the predicted output $\hat{y}(k)$, based on a model, with the observed output $y(k)$. In case of a systematic difference we will alarm. If only $\hat{x}(k)$ is available the first two ideas for fault detection ideas would be:

- Try to reconstruct $y(k)$ using a model of the observer, e.g.

  $$L_j y_j(k) = \hat{x}_j(k+1) - (A - L_j C_j)\hat{x}_j(k) - B_u u(k)$$

  Here we need a very accurate model and the internal structure of the observer, e.g. the gain L, otherwise the estimations will be easily biased. In many practical cases this would be difficult. Notice also that if L_j is not full rank, it allows for multiple solutions.

- Assume that there are at least two observers providing $\hat{x}_1(k)$ and $\hat{x}_2(k)$. Define the residual vector

  $$\varepsilon(t) = \hat{x}_1(k) - \hat{x}_2(k)$$

  which should be sensible to fault that affects the two observers in different ways, e.g. sensor faults. This approach does not, however, make direct use of the model of the system.

We will start by analyzing the case with only one observer.

Idea: View $\hat{x}(k)$ as the output from the extended system

$$\begin{aligned} x(k+1) &= Ax(k) + B_u u(k) + F_a f_a(k) \\ \hat{x}(k+1) &= A\hat{x}(k) + B_u u(k) + L(y(k) - C\hat{x}(t)) \\ \hat{y}(k) &= C^* \hat{x}(k), \quad \text{where} \quad y(k) = Cx(k) + F_y f_y(k) \end{aligned}$$

where C^* depicts which estimates are available. By using the extended state $\bar{x}(k) = [x^T(k) \quad \hat{x}^T(k)]^T$ and the corresponding state space matrices we can interpret this a standard fault detection problem, which could be approached by a parity space method or a Kalman filter based method. The problem when we have m different observers can be approached by augmenting the state space $x(k)$ with all observer states $\hat{x}_j(k)$, $j = 1, \ldots m$.

There are some basic questions that need to be addressed with such framework which are studied in this thesis:

- Are the faults detectable using this model?
- What to do if the sensor gain L is unknown?
- How to compare/validate the performance of different methods?

1.2: Outline

This thesis is divided in two parts. Part I approaches the problem in a more theoretical manner, analyzing some of its properties and proposed solutions. Part II presents a practical example in detection of localization faults in mobile robots. Starting with Part I, it follows as:

Chapter 2 presents a review on fault detection with a brief introduction to the main different methods and paradigms.

Chapter 3 is an introduction to state observers and some relevant properties for this work.

Chapter 4 presents some theoretical analysis on observer-based fault detection for both when one has access to $y(k)$ or only $\hat{y}(k)$, and addresses the questions on fault observability, fault sensitivity and unknown sensor structure (for the latter case).

Chapter 5 starts Part II with our application example. It describes the platform used, wheeled mobile robots. It includes a discussion on modeling and localization methods with more emphases on odometry and laser scan matching.

Chapter 6 presents an approach for fault detection in wheeled mobile robots. Different methods are used which are compared and analyzed.

Finally, Chapter 7 concludes this work and leave comments on remaining challenges.

Moreover, in Appendix A, we include the conference paper to be presented in the 7th SAFEPROCESS.

1.3: Contributions

The main contributions of this thesis are:

- The fault observability conditions for integrated sensors presented in Chapter 4.
- The analysis on residuals fault sensitivity for the different methods and comparison presented in Chapter 4.
- The fault detection framework for localization faults in mobile robots presented in Chapter 6.
- The conference paper in Appendix A.

Part I

Fault detection

2 Fault detection review

This Chapter presents a review on the main aspects of fault detection and isolation (FDI) methods with focus on the application to localization fault detection in mobile robots.

2.1: Fault Detection and Isolation Overview

This Section was based on [5, 12, 17] and provides the reader with a brief overview of FDI (Fault Detection and Isolation) methods. Basically, the purpose of FDI is to monitor dynamic systems and should be able to perform the following tasks:

- Fault detection: FDI recognizes that *a fail has occurred*.
- Fault isolation: FDI recognizes *where and when* a fail has occurred (some FDI extend this concept to include the type, size or cause of the fail).

To choose the algorithm of the FDI it is important to know which kind of fault is present in the system. Basically the fault types can be classified by its *time behavior* and *effects on the system*.

The first category of fault can be summarized as:

- Abrupt: faults that affect the system in a stepwise manner. Ex: A wrong calibration of a sensor causing an offset like error.
- Incipient: faults that occur gradually with time (drift-like). Ex: increase of friction inside a gearbox due to wear.
- Intermittent: faults that affect the system during certain time intervals, with interrupts. Ex: undesired air bubbles in a gas pipeline.

The manner a fault affects the system's behavior can be categorized as:

- Additive: faults that are effectively added to the system's input or output. Ex: sensor faults.
- Multiplicative: faults that changes the parameters of the system. Ex: resistance changes in an electric motor caused by an overheat.
- Structural: faults that introduces new governing terms to the describing equations of the system. Ex: a change on the dynamic behavior of an aircraft caused by a loss of power in the engines.

Figure 2.1 illustrates additive faults in the input signal (f_u) and output signal (f_y) as well as a multiplicative faults (f_{par}) of a system.

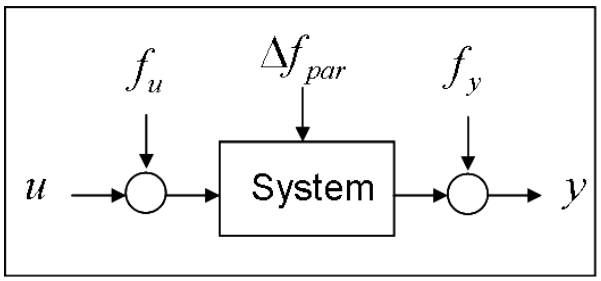


Figure 2.1: Additive and Multiplicative faults.

Remark 1 *Localization faults are generally associated with a malfunction of a localization provider and affects an output of the system. Therefore, it can be classified as an additive fault that may occur abruptly or in an incipient manner.*

Fault detection methods can either run on-line (concomitantly to the normal operation of the system) or off-line (achieved for example with special experiments for diagnosis). The methods and complexity for each of these category vary considerably, with on-line methods remaining the biggest challenge.

2.1.1: Fault Detection and Isolation Methods

FDI methods can rely or not on a model of the system. Three convenient categories for *model-free methods* are:

- Hardware redundancy: these systems rely on extra hardware which are specially used to detect faults.

- Spectral analysis: utilize mechanical vibration, noise, ultrasonic, current or voltage signals to detect and diagnose faults.

- Expert/Logic systems: rely on previous knowledge about the behavior and characteristics of the system (age, statistical data, operating condition, etc) under different circumstances. It is a logical method therefore it does not need extra hardware.

- Dimensionality reduction: PCA or Principal Component Analysis is a classical approach that transforms a number of possibly correlated variables into a smaller number of uncorrelated variables, called principal components. The first principal components will explain the larger variances of the signal (system dynamics for example), while smaller ones will relate to the parts carrying smaller variances, noise for example. Whenever a fault appears, the PCA cannot explain the signal and therefore detection is made possible.

  Other methods are for example FDA (Fisher Discrimant Analysis) which is related to PCA with the property of considering different classes of data into account and PLS (Partial Least Squares). See [27] for more.

Model free methods have been applied successfully in the industry but these methods present some clear drawbacks. In the case of hardware redundancy, extra costs and weight is added to the system; model free methods make use of *a priori* (and often empirical) knowledge of the system signal characteristics, which are dependent on the system operational point and can be costly to define if no previous knowledge about the signals are available.

As pointed in [32], model-free methods are not suitable for mobile robots application since these systems operates over a wide range of different conditions and might be difficult, too costly or even impossible to obtain data for the failure cases. Therefore, this work will focus on the use of the so-called model-based methods. The next Section gives an overview of the several model-based FDI techniques emphasizing its application to robotic systems.

2.2: Model-based FDI methods

This class of methods are based on the principle of *analytical redundancy*. Instead of comparing several signals outputs for the same variable as in hardware redun-

dancy, they compare system ouputs with analytically generated signals that resemble to the ouput of the system.

Figure 2.2 displays the general flowchart of a model-based FDI method.

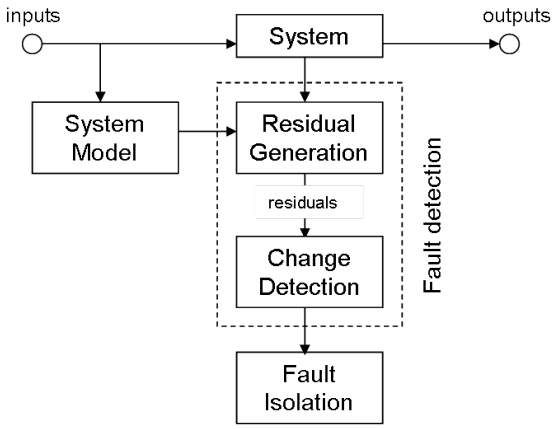


Figure 2.2: Model-based FDI flowchart

Residual is a fault indicator, based on a deviation between measurements and model-equation-based computations. The residuals are usually generated by filtering techniques that take measured signals (i.e. inputs and outputs) and transform them into a sequence of residuals that resemble white noise before a change occurs and drifts in case of a fault.

Regardless the method used to generate a residual, a residual generator, as illustrated in Figure 2.3 is just a filter that takes as input measured inputs $u(s)$ and outputs $y(s)$ from the monitored system. Where $H_u(s)$ and $H_y(s)$ are realizable transfer

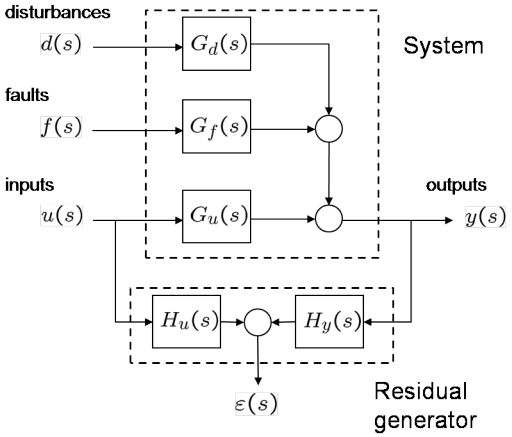


Figure 2.3: General structure of a residual generator.

functions. Since the residual should be insensitive to the input u, we have the condition that

$$H_u(s) + H_y(s)G_u = 0$$

There are three main methods to generate residuals in a model-based approach:

- Parity space: the system model directly produces outputs that are comparable with the measured outputs.
- Diagnostic observers or state estimators: an observer is designed to reconstruct the states of a system, which are compared to the real states generating the residual.
- Parameter estimation: an estimation of some physical parameters of the system is compared with its healthy values generating the residuals.

Remark 2 *According to [6], more than 50% of the applications to detect additive faults uses observer methods while more than 50% of the applications to multiplicative faults utilizes parameter estimation methods.*

Nevertheless, one can find several successful applications of observer methods to detect multiplicative faults utilizing for example augmented states where the unknown parameters are modeled as a state of the system. The parity space will not be further detailed since these methods work in an open-loop fashion which requires a precise model of the system with fixed parameters, which is generally not the case in mobile robots. An example of the use of a parity space approach to fault diagnosis in industrial robots can be found at [22].

In the following Subsections, some of the methods for residual generation and fault detection found in the literature are reviewed and discussed, the focus will be in parameter estimation and diagnostic observers.

2.2.1: Residual generation methods - parameter estimation

Parameter estimation is the process of estimating some or all parameters of a system model using its input and output measurements. Residuals can be generated when the estimated parameters are compared with fault-free values of such parameters (Figure 2.4). In [28] for example, the friction parameters inside an industrial robot arm joint is estimated and monitored for diagnosis.

Since the measured signals are stochastic (corrupted by noise) and physical systems are generally nonlinear, recursive estimators like nonlinear observers, extended

Kalman filters or recursive least squares are generally used to update the parameter estimates. These parameters are usually initially guessed and then converge to a final value after multiple recursive steps.

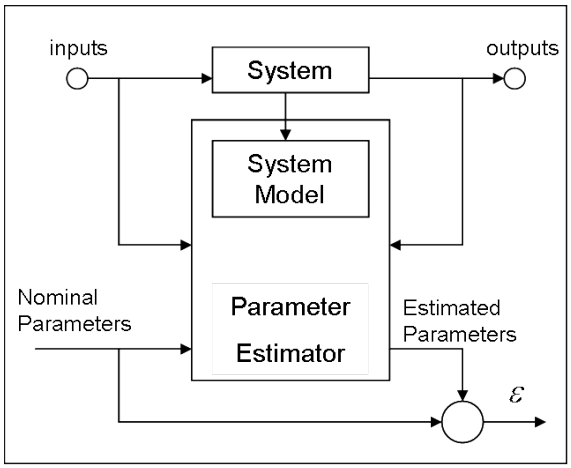


Figure 2.4: Parameter Estimation Block Diagram

There are various different but related conceptual bases for continuous-time system parameter estimation see [26] for a basic literature in system identification. They are briefly described here.

1. Output error methods, OE: this is maybe the most intuitive parameter estimation approach. The parameters are estimated in order to minimize the error between the model output and the system output.

$$\varepsilon(t) = y(t) - \frac{\hat{B}}{\hat{A}}u(t) \tag{2.1}$$

where $\hat{A}$ and $\hat{B}$ are the estimates of the governing polynomials of the system. In this case, no direct calculation of the parameters is possible, because $\varepsilon(t)$ is nonlinear in the parameters. The loss function is therefore minimized as an optimization problem.

2. Equation error methods, EE: this approach is clearly derived from an analogy with static regression analysis and linear least squares estimation. The error function is generated directly from the input-output equations of the model.

$$\varepsilon(t) = \hat{A}y(t) - \hat{B}u(t) \tag{2.2}$$

From Equation 2.2 it is clearly seen that it implies the generation of the time derivatives of the signal, which might be a problem when the signal is too noisy.

Young, [21], proposes a solution for this utilizing a 'generalized equation error' that filters the measured signals and provides filtered derivatives of the signals.

After sampling, the estimation can be solved as a least square estimate or in a recursive form (recursive least squares). Isermann, [4], emphasizes that for numerical properties improvement, square-root filters algorithms are recommended.

3. Prediction error methods, PE: the equation of the error is the same as the OE case, Equation 2.1, the difference is that the output estimate is defined as a "best prediction" depending on the current estimates of the parameters θ which characterize the system and the noise models, $\hat{y}(t)\widehat{=}\hat{y}(t|\theta)$. $\hat{y}(t|\theta)$ is a conditional estimate of $y(t)$ given all current and past information of the system, while $\varepsilon(t)$ is an 'innovations' process with serially uncorrelated white noise characteristics (see [26] for more). The PE can be written as an OE (2.3) or EE (2.4) approach.

$$\varepsilon(t) = \frac{\hat{C}}{\hat{D}}\left[y(t) - \frac{\hat{B}}{\hat{A}}u(t)\right] \tag{2.3}$$

$$\varepsilon(t) = \frac{\hat{C}}{\hat{D}\hat{A}}\left[\hat{A}y(t) - \hat{B}u(t)\right] \tag{2.4}$$

4. Maximum likelihood methods, ML: a special case of PE methods, separated here because of its importance, with the additional restriction that the stochastic disturbances to the system have specified amplitude probability distribution functions. In several applications, this assumption is restricted further for analytical tractability to the case of a Gaussian distribution.

5. Bayesian methods: extension of ML where *a priori* information on the probability distributions is included in the formulation of the problem. It is important in the FDI context because most recursive methods can be interpreted as being a Bayesian type.

An usual approach to deal with the problem of parameter estimation is to consider linear models. Here we summarize the main parameter estimation methods for *linear* continuous-time models based on sampled signals [9].

1. Least-squares parameter estimation:this is a well-known case of optimization where the estimated parameters vector $\hat{\theta}$ are estimated by the non recursive estimation equation below.

$$\hat{\theta} = [\psi^T\psi]^{-1}\psi^T y \tag{2.5}$$

where ψ is the data vector and y is the measured output. These parameters are biased by any noise, therefore, a good signal-to-noise ratio must be achieved to use this method.

2. Determination of the time derivatives: as mentioned before, the estimation of the signals derivatives by numerical differentiation is not a good approach because of the inherent noise in the signals. A state variable filter is therefore utilized that calculates the derivatives and filter the noise.

3. Instrumental variables parameter estimation: instrumental variables can be used to overcome the bias problem due to noise. The instrumental variables introduced are only insignificantly correlated with the noise-free process output. A major advantage of instrumental variables is that no strong assumptions and knowledge on the noise is required. However, when dealing with closed loop configurations biased estimates are obtained because the input signal is correlated with the noise.

4. Parameter estimation via discrete-time models: one can try to estimate the variables in discrete-time models and then calculate the parameters of the continuous-time model. These methods, however, require extensive computational effort and are not so straightforward.

Parameter estimation for fault detection has been used successfully for several applications. However, the fundamental tasks of modelling and identification in such framework might not be trivial and certainly corresponds to its main design tasks. Consider for example a chemical process, where there might be no precise physical model, and operating in closed-loop, it would be very difficult or even unfeasible to either model it or to perform an informative enough excitation for a proper identification.

2.2.2: Residual generation methods - state estimation (observers)

This category of FDI methods uses a state observer (a Kalman filter for example) to reconstruct the unmeasurable state variables based on the measured inputs and outputs. It can be shown that an additive fault is easily detected with this technique, this kind of fault makes the residual (generally taken as the estimation error) deviate from zero with a bias.

Remark 3 *The influence of multiplicative faults in residuals generated by state observers is not as straightforward recognizable because in this case the changes in the residuals could be caused either by parameter, input and state variable changes and are not easily seem in the system output.*

The main advantage of observer-based methods is that they do not require special excitation of the system, making it a *good choice for on-line fault detection*.

Observer-based FDI methods also require an accurate mathematical model of the process, therefore it is important to try to robustify the residual evaluation in order to cope with the inherited uncertainties of any physical model. Investigations of robust observer-based approach can be found for example at [16, 14, 56].

The following FDI methods with state estimation are known, [4, 5]:

1. Dedicated observers for multi-output processes: the design of specific observers allows the detection of specific faults, combining and arranging the observers one can detect multiple faults.
   (a) Observer excited by one input: one observer is driven by one sensor output while the other outputs are estimated and compared with the measurements allowing the detection of single sensor faults (additive faults).
   (b) Bank of observers, excited by single outputs: several of the first case allowing the detection of multiple sensor faults.
   (c) Kalman filter, excited by all outputs: the residuum changes the characteristic of zero mean white noise with known covariance if a fault appears, which is detected by a hypothesis test.
   (d) Bank of observers, excited by all outputs: several of the above designed to detect a definite fault signal.
   (e) Bank of observers, excited by all outputs except one: as before, but each observer is excited by all outputs except one sensor output which is supervised.
2. Fault detection filters for multi-output processes: the feedback state observer is chosen so that particular fault signals in the input change in a definite direction and fault signals at the output change in a specific plane.

3. Output observers: another possibility is the use of output observers (unknown input observers) if the reconstruction of the state variable is not of primary interest. A linear transformation is applied so that the residuals are dependent only on additive input/output faults.

2.3: Change (fault) detection methods

After the generation of the residuals, it is needed to establish whether there was a change (fault) on the system or not. This role is done by the change detector which can be classified under three categories [17]:

1. One model approach: The filter residuals $\varepsilon(k)$ are transformed to a distance measure $s(k)$ (computed from the no-fault values), a stopping rule decide whether the change is relevant or not. A schematic is show at Figure 2.5.

Figure 2.5: One model approach for change detection

The most natural distance measures are:

- Change in the mean, $s(k) = \varepsilon(k)$.
- Change in the variance, $s(k) = \varepsilon(k)^2 - \lambda$, where λ is a known fault-free variance.
- Change in correlation, $s(k) = \varepsilon(k)y(k-d)$ or $s(k) = \varepsilon(k)u(k-d)$ for some d.
- Change in sign correlation, $s(k) = sign(\varepsilon(k)\varepsilon(k-1))$, this test is used due to the fact that white residuals should change sign every second sample in the average.

2. Two model approach: In this case the residuals are generated by two filters, a slow (with a great data window or the whole data) and a fast one (with a small data window) which are compared, Figure 2.6 illustrates the procedure. If the model based on the smaller data window gives larger residuals a change is detected. The main problem is to choose an adequate norm for the comparison, typical norms are:

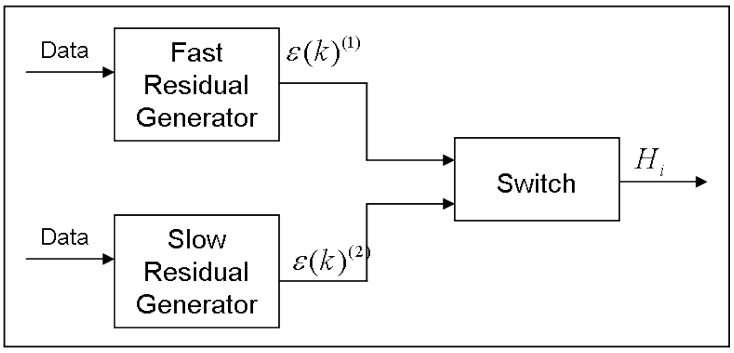


Figure 2.6: Two residual generators running in parallel, one slow to get good noise attenuation and other fast to get fast tracking. The switch decides whether a change occurred or not.

- The Generalized Likelihood Ratio (correlation between fault signatures).
- The divergence test.
- Change in spectral distance.

3. Multi-model approach: This approach makes use of the so-called *matched filters*, that can generate white residuals for a specific change even after it was inserted in the system. The idea is to enumerate all conceivable hyphoteses about changes and compare the residuals generated from the matched filters, the one with the 'smallest' residuals will be an indication of the change, Figure 2.7 shows the procedure. Since a batch of data is needed, this approach is off-line, but many proposed algorithms makes the calculations recursively, and are consequently on-line.

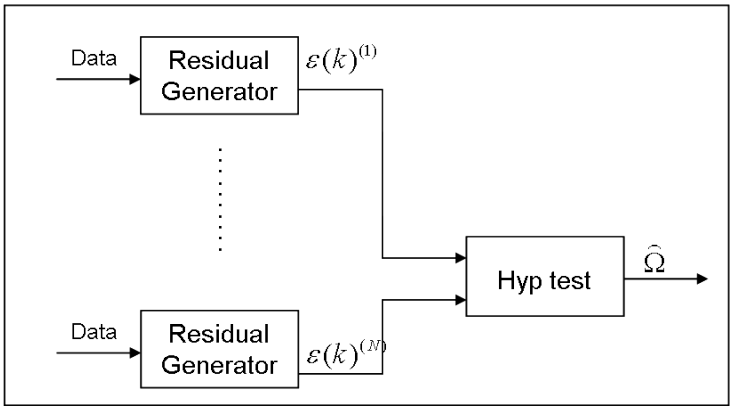


Figure 2.7: Several matched filters (residual generators) that are compared in a hyphotesis test.

Remark 4 *The performance of change detection method will always be a trade-off between speed (how fast the method detect a fault) and false alarm rate.*

2.4: Concluding Remarks

The Chapter presented a review of some fault detection methods. Emphasis has been given on methods for monitoring unmeasurable quantities like process parameters and process state variables.

In designing of a model-based FDI method, the following aspects should be considered [9]:

- Process models: since the methods are based on the deviation of a normal operation, one should define the normal operation of the system (for example, nominal values of parameters) and also which kind of model. If the system or process is running only with small changes of the variables, linearized models can be used. However for many applications this is not the case ([15, 13, 23]).

  Besides the use of analytical models, a diagnosis system can be combined with heuristics of the system, which can be translated for example in fault-symptom-trees or fuzzy logic and are important for the fault isolation.

- Parameter and state estimation: as discussed through the Chapter, state estimation has its main applications in the detection of additive faults and has the drawback that it is difficult to identify the source of the fault since the residuals are deviations of the system states.

  On the other hand, parameter estimation techniques are the most indicated approach for the identification of multiplicative faults. In [8, 18] several multiplicative faults could be identified utilizing parameter estimation, testifying its relevance. A main drawback of parameter estimation however is that the system input signal must be informative enough to perform identification, which makes it difficult to implement in on-line diagnosis systems.

  In a complete diagnosis system, where both additive and multiplicative faults are required, the FDI methods utilizing parameter and state estimation complements each other. For example, an observer-based FDI method that detects faults on sensor and actuators could run on-line, whether an actuator fail is detected an off-line test utilizing a parameter estimation FDI method could be used to fault diagnosis and isolation.

- Faults: the way that a fault affects the system is very important when designing FDI methods. One should also map the faults that affect the system.

- Performance: fault detectors must be sensitive to the appearance of faults but insensitive to other changes (noise, operating points, modeling errors, etc.). Because these requirements often contradict each other, the following trade-offs must be analyzed:
  - size of fault vs detection time;
  - speed of fault appearance vs detection time;
  - speed of fault appearance vs process response time;
  - size and speed of fault vs speed of process parameters changes;
  - detection time vs false alarm rate.

  Methods that are sensitive to abrupt faults for example, might not be suitable to detect incipient faults.

- Practical aspects: a FDI method should consider the practical aspects when defining the experiments to detect the faults, for example safety-critical system, systems under a closed-loop, network system, etc. and should be robust enough to cope with them.

- Testing: the introduction of artificial faults is also important to validate the system reliability and should try to approximate to real faults.

3 State observers

As mentioned in the previous Chapter, one of the most important tasks in a fault detection scheme is the residual generation method design. A state observer takes the system input and output signals and utilizing a system model, estimates its states, which can be used as an analytical redundancy for fault detection.

In this Chapter, we introduce the concept of state observers depicting some of its properties and different configurations.

3.1: State observers

Given a liner-time invariant system described by the state equations below:

$$\begin{aligned} x(k+1) &= Ax(k) + B_u u(k) \\ y(k) &= Cx(k) \end{aligned} \tag{3.1}$$

Here $x(k)$ denotes the state vector, $u(k)$ is a known input signal, $y(k)$ are the measured output signals.

Under the condition that the system is observable (see Section 3.3) and assumptions of a known model structure and parameters, the internal state variables $x(k)$ can estimated with the Luenberger observer:

$$\begin{aligned} \hat{x}(k+1) &= (A - LC)\hat{x}(k) + L(y(k)) + B_u u(k) \\ \hat{y}(k) &= C\hat{x}(k) \end{aligned} \tag{3.2}$$

If system and model parameters are equal, the state error

$$\tilde{x}(k+1) = x(k+1) - \hat{x}(k+1)$$

becomes

$$\tilde{x}(k+1) = (A - LC)\tilde{x}(k)$$

And hence,

$$\lim_{k\to\infty} \tilde{x}(k) = 0$$

for any initial state error $\tilde{x}(0) = x(0) - \hat{x}(0)$ if the $A - LC$ has only stable poles and will be zero if the initial error is zero.

When $L = 0$, Equation 3.2 is turned into a pure simulator, in which the measured output does not affect the observer. With $A = LC$, the observer is called a predictor and the estimates are taken directly from the measurements.

The observer gain L is an important design parameter that will influence the observer behavior, it can be adjust for example with pole placement methods.

3.2: State estimation - The Kalman filter

The Kalman filter, initially presented in 1960 [29] is a special case of observer. Also called estate estimator, it minimizes the mean of squared error for the observations of stochastic systems in the form below:

$$\begin{aligned} x(k+1) &= Ax(k) + B_u u(k) + w(k) \\ y(k) &= Cx(k) + v(k) \end{aligned} \tag{3.3}$$

in which $w(k)$, process noise, and $v(k)$, measurement noise, are assumed to be independent from each other, and to attend a white gaussian distribution,

$$p(w) \sim N(0, Q)$$
$$p(v) \sim N(0, R)$$

with process and measurement covariances, Q and R. Given the *a priori* state estimate $\hat{x}(k|k-1)$, computed with knowledge on the process before time k and *a posteriori* estimate $\hat{x}(k|k)$ given measurement $z(k)$, one can define the *a priori* and *a posteriori* covariance error estimates as

$$\begin{aligned} P(k|k-1) &= E[e(k|k-1) :\ e(k|k-1)^T], \quad \text{where :}\ e(k|k-1) = x(k) - \hat{x}(k|k-1) \\ P(k|k) &= E[e(k|k) :\ e(k|k)^T], \quad \text{where :}\ e(k|k) = x(k) - \hat{x}(k|k) \end{aligned}$$

The Kalman filter finds a linear relation between the *a posteriori* estimate and *a priori* estimate and measurement as

$$\hat{x}(k|k) = \hat{x}(k|k-1) + K(y(k) - C\hat{x}(k|k-1)) \tag{3.4}$$

where the term $(y(k) - C\hat{x}(k|k-1))$ is also known as the filter innovations, $\tilde{x}(k)$, which under the above mentioned conditions will follow a white gaussian distribution. The also called Kalman filter gain, K, is the optimal solution that minimizes the *a posteriori* error covariance, $P(k|k)$, and therefore the estimates. A possible solution for the problem is the form as:

$$K(k) = P(k|k-1)C^T(CP(k|k-1)C^T + R)^{-1} \tag{3.5}$$

The above Equation can be achieved by minimizing the trace of the error covariance in respect to K given the state estimate in form of Equation (3.4). Check [29] for a detailed proof.

Analyzing Equation (3.5), one can depict the smaller measurement noise covariance, R, the greater is the computed gain and therefore, the greater will the innovations influence be in the estimates. In other words, the filter will thrust more on the measures when the measures itself are more reliable.

On the other hand, the greater *a priori* error covariance $P(k|k-1)$, which is directly related to the process covariance Q, the smaller is the gain. Which in other words, means that the estimate update will thrust more on the model, rather than the innovations process.

3.2.1: The filter algorithm

The Kalman filter can be written in a recursive manner, separated in two steps. First, a *prediction* of the estimations given the previous estimate and system model. This prediction is then *corrected* by the, so-called, innovations process in which the measures are taken in a feedback manner.

The prediction step (or time update) equations fall in the group:

$$\hat{x}(k|k-1) = A\hat{x}(k-1|k-1) + Bu(k-1) \tag{3.6}$$

$$P(k|k-1) = AP(k-1|k-1)A^T + Q \tag{3.7}$$

In the correction step the actual measurements are used to correct the estimations and filter equations:

$$K(k) = P(k|k-1)C^T(CP(k|k-1)C^T + R)^{-1} \quad (3.8)$$

$$\hat{x}(k|k) = \hat{x}(k|k-1) + K(k)\left(y(k) - C\hat{x}(k|k-1)\right) \quad (3.9)$$

$$P(k|k) = \left(I - K(k)C\right)P(k|k-1) \quad (3.10)$$

The algorithm steps can be translated as:

Prediction	
(3.6)	State prediction given model
(3.7)	Covariance prediction given model

Correction	
(3.8)	Gain computation
(3.9)	State correction given innovations
(3.10)	Covariance update

Table 3.1: Kalman filter algorithm steps

Finally, an initial guess for $P(0)$ and $\hat{x}(0)$ is needed. A comparison with the state observer shown in Section 3.1 shows that the observer only uses past information and not predicted ones.

Under the assumption of constant Q, R, and model, the gain will converge quickly and can be pre-computed in an off-line manner solving a Riccati equation of the error covariance matrix P. In such case the gain is computed as

$$\bar{K} = PC^T\left(CPC^T + R\right)^{-1}$$

and it can be shown, [5], that, with the prediction inserted, the filter becomes

$$\hat{x}(k+1|k) = A\hat{x}(k|k-1) + Bu(k) + A\bar{K}\left(y(k) - C\hat{x}(k|k-1)\right)$$

which in comparison, is the same as an observer with $L = A\bar{K}$.

3.2.2: The extended Kalman filter

The Kalman filter as presented until now is the optimal solution for the state estimation problem in stochastic *linear* systems and supposes that both the transition state matrix A and measurement matrix C to be constant.

It is still possible to use the Kalman filter for nonlinear systems by linearizing the system at each time step and computing the gain and covariance error. With this approach, the filter is not optimal but still finds itself useful in several cases, for example in GPS and navigation systems.

Considering the non-linear system:

$$\begin{aligned} x(k+1) &= f\left(x(k), u(k)\right) + w(k) \\ y(k) &= h\left(x(k)\right) + v(k) \end{aligned} \tag{3.11}$$

The nonlinear functions *f* and *h* can be used to compute the predicted estate and measurement from the previous estimations, Equation 3.6. However, to update the covariance and, consequently, the gain, these functions should be linearized over the system operational point at each time step, using, for example Jacobian matrices.

3.3: Observability conditions

A system is said observable if its internal states at a time can be computed from a finite set of output observations and the respective inputs. It is an internal property of the system determined by the system matrices A and C.

Below we present two different but equivalent criteria for the observability of a system, these are not new results but will be relevant for the continuation of this thesis. The demonstrations are based in [30], but might be found in any usual system theory book.

3.3.1: The observability matrix

With the representation of the system output as a time series, the system equations can be rewritten as

$$\begin{pmatrix} \Delta y(k_0) \\ \Delta y(k_0+1) \\ \vdots \\ \Delta y(k_0+p-1) \end{pmatrix} = \begin{pmatrix} C \\ CA \\ \vdots \\ CA^{(p-1)} \end{pmatrix} x(k_0) \tag{3.12}$$

where $\Delta y(k)$ represents all know terms, function of the measured output $y(k)$ and input $u(k)$:

$$\begin{aligned}
\Delta y(k_0) &= y(k_0) - Du(k_0) \\
\Delta y(k_0+1) &= y(k_0+1) - CBu(k_0) - Du(k_0+1) \\
\Delta y(k_0+2) &= y(k_0+2) - CABu(k_0) - CBu(k_0+1) - Du(k_0+2) \\
&\vdots \\
\Delta y(k_0+p-1) &= y(k_0+p-1) - CA^{(p-2)}Bu(k_0) - \dots \\
&\quad -CBu(k_0-p-2) - Du(k_0+p-1)
\end{aligned}$$

Equation 3.12 has a unique solution for $x(k_0)$ if and only if the matrix that multiplies $x(k_0)$ has the same rank as the n total states in the system.

The Cayley-Hamilton theorem, shows that any matrix A^n linearly depend on $I, A, \dots, A^{(n-1)}$ (see [30] for a proof), and therefore the criteria can be checked directly from the matrix:

$$\mathcal{O} = \begin{pmatrix} C \\ CA \\ \vdots \\ CA^{(n-1)} \end{pmatrix} \tag{3.13}$$

The matrix $\mathcal{O}$ is known as the observability matrix and shows that the system is observable if

$$rank\, \mathcal{O} \;=\; n$$

Which is equivalent to check if $\mathcal{O}$ has full column rank since it has n columns.

3.3.2: The Popov-Belevitch-Hautus criteria

Another important criteria for testing the observability of systems is known as the Popov-Belevith-Hautus criteria (PBH), firstly studied by these authors. The test depicts a pair $\begin{pmatrix} A & C \end{pmatrix}$ to be observable if and only if

$$rank \begin{pmatrix} C \\ A - sI \end{pmatrix} \;=\; n \quad \forall s \tag{3.14}$$

Where n is the dimension of A. It is easy to see that these conditions are met for all s that are not eigenvalues of A. The relevance of the theorem is that the rank must

be n even when s is an eigenvalue of A. For a proof and more insights on system observability and its dual, controllability, the reader is advised to check [49].

3.4: Concluding remarks

This brief Chapter presented a review on state observers and state estimators (Kalman filter) as well as relevant observability conditions. The theory of observers has developed mainly in the 60's and has been, since then, used successfully in both industry and academy, specially the (optimal) Kalman filter is applied in different areas and is still an important tool. The Kalman filter was first designed for linear systems, and with such constrain, it is actually optimal. For non-linear systems, approaches as the Extended Kalman filter may work considerably well, but there is no guarantee on optimality.

With the theoretical background presented so far, the next Section finalizes this Part presenting the problem of observer-based fault detection in a more thorough manner, holding a discussion on our primary problem description, fault detection through integrated sensors.

4 Observer-based fault detection

With the introduction to state observers given in the previous Chapter, we finally present a framework for residual generation using state observers.

We present both the classical approach for residual generation through state observers (when the raw measurements are available) and the problem that arises when these measurements are not available but only an estimated output. Such situation is common in many more advanced sensors where raw measurements are internally processed before refined information is provided to the user.

The problem of fault detection (mostly related here to the residual generation task) is studied in both the classical and in the case of observer-integrated sensors. Conditions for fault observability using the concepts found in the previous Chapter are depicted as well as a discussion on the methods performance and residual analysis.

4.1: Observer-based residual generation - classical approach

Considering the following system description:

$$x(k+1) = Ax(k) + B_u u_0(k) + F_a f_a(k) \tag{4.1}$$

where $x(k)$ denotes the states vector, $u_0(k)$ is a fault-free input to the system and $f_a(k)$ model actuator fault inputs, treated as unknown.

When the system inputs and outputs are measured through sensors, which can also be subject to additive faults as

$$u(k) = u_0(k) + F_u f_u(k) \tag{4.2}$$

$$y(k) = Cx(k) + F_y f_y(k) \tag{4.3}$$

we can use the measured $u(k)$, $y(k)$ and a system model to design a state observer

(see earlier Chapter) as

$$\begin{aligned} \hat{x}(k+1) &= (A-LC)\hat{x}(k) + L(y(k)) + B_u u(k) \\ \hat{y}(k) &= C^* \hat{x}(k) \end{aligned} \tag{4.4}$$

where C^* depicts which estimates are available. The output of the observer $\hat{y}(k)$ can then be used to generate a residual $\varepsilon(k)$ that is the difference between observer and system output

$$\varepsilon(k) = y(k) - \hat{y}(k)$$

which can be used for fault detection purposes. This is the classical approach for residual generation through observers. Section 4.1.1 presents some analysis on fault observability while Section 4.1.2 analyzes the residual behavior.

4.1.1: Fault observability

As described in [55], stochastic biases in linear time invariant systems can be identified by augmenting the system state with a bias and implement a Kalman filter. The author utilizes this technique to identify biases in noisy measurements.

In Chapter 3 in [52] these results are extended to check the observability of additive faults with the constraint that $f(t+1) = f(t)$ (faults moving as a random walk). The faults f_a and f_y are augmented in the system states, so that we have

$$\begin{aligned} \bar{x} &= \begin{bmatrix} x & f_a & f_y \end{bmatrix}^T \\ \bar{A} &= \begin{bmatrix} A & F_a & 0 \\ 0 & I & 0 \\ 0 & 0 & I \end{bmatrix} \\ \bar{C} &= \begin{bmatrix} C & 0 & F_y \end{bmatrix} \end{aligned} \tag{4.5}$$

Given that the pair (A, C) is observable, the author depicts that the augmented system observability is provided by checking the rank of the matrix

$$\begin{pmatrix} C & F_y \\ (A-I) & F_a \end{pmatrix}$$

It is also shown that for the especial cases when only measurement or process fault are present we have the especial conditions:

- For output measurement faults only (f_y)

  If the system has no integrator dynamics, the faults will be observable as long as F_y is full rank.

  If there is an integrator in the system dynamics, then the system is not observable if F_y is full column rank and the faults should be orthogonal to the measured integrating part of the system (modes with eigenvalue equals to 1).

- For process faults only (f_a)

  Changes introduced by the dynamics of the system which are not directly measured must be distinguishable (orthogonal) to the disturbances.

In Section 4.2.1 this approach is used to analyze the fault observability for systems with observer-integrated sensors where the conditions are presented in a more thorough manner.

4.1.2: Residual analysis

Assuming zero as initial conditions, with $C^* = C$ and q as the time shift operator, the observer as in Equation 4.4 can be rewritten in an input output form

$$\hat{\mathbf{y}}(k) = C\mathcal{H}_{(A-LC)}B_u\,\mathbf{u}(k) + C\mathcal{H}_{(A-LC)}L\,\mathbf{y}(k)$$
$$\text{where} \quad \mathcal{H}_M = [qI - M]^{-1}$$

$\mathcal{H}_M$ is also known as the matrix resolvent of M and represents the dynamics introduced by M. The residual is then

$$\varepsilon(k) = (I - C\mathcal{H}_{(A-LC)}L)\,\mathbf{y}(k) - C\mathcal{H}_{(A-LC)}\,\mathbf{u}(k)$$

Using Equations (4.1)-(4.3) we can finally write the residual as function of $u_0(k)$ and faults

$$\begin{aligned}
\varepsilon(k) = \quad & \left[(I - C\mathcal{H}_{(A-LC)}L)C\mathcal{H}_A - C\mathcal{H}_{(A-LC)}\right] B_u\,\mathbf{u_0}(k) \\
+ \quad & \left[(I - C\mathcal{H}_{(A-LC)}L)C\mathcal{H}_A\right] F_a\,\mathbf{f_a}(k) \\
+ \quad & \left[(I - C\mathcal{H}_{(A-LC)}L)\right] F_y\,\mathbf{f_y}(k) \\
+ \quad & \left[-C\mathcal{H}_{(A-LC)}B_u\right] F_u\,\mathbf{f_u}(k)
\end{aligned} \tag{4.6}$$

Also, using the matrix identity below

Theorem 5

$$A^{-1} - B^{-1} = A^{-1}(B - A)B^{-1}$$

Proof:

$$\begin{aligned} A^{-1} - B^{-1} &= A^{-1} - B^{-1} \\ &= A^{-1}BB^{-1} - A^{-1}AB^{-1} \\ &= A^{-1}(BB^{-1} - AB^{-1}) \\ &= A^{-1}(B - A)B^{-1} \end{aligned}$$

we can show that $\left[(I - C\mathcal{H}_{(A-LC)}L)C\mathcal{H}_A - C\mathcal{H}_{(A-LC)}\right] = 0$, meaning that *the residual is not influenced by the system input, only by the faults*. Note that we are **not** considering noise and model uncertainties.

This is a very important property for the observer based residual and the faults will appear in the residual governed by the observer and system dynamics as described in Equation (4.6).

4.1.2.1: Fault sensitivity

As presented in [30], the sensitivity of the residual to a fault is a transfer function relating a fault to the residual

$$S_f(q) = \frac{\partial \varepsilon}{\partial f}$$

These functions are easily seem in Equation (4.6). We will analyze them as a function of the observer gain L. The results are related to [50] in which the observer gain is analyzed for linear interval models (models including uncertainties in the parameters).

For additive input sensor faults we have

$$\begin{aligned} S_{fu}(q) &= -C\mathcal{H}_{(A-LC)}B_u\, F_u \\ &= -C\left[qI - (A - LC)\right]^{-1} B_u\ F_u \end{aligned} \tag{4.7}$$

Analyzing its value at instant times $k = 0$ and $k \to \infty$ we have

$$\begin{aligned} S_{fu}(0) &= \lim_{q\to\infty} S_{fu}(q) = 0 \quad (4.8) \\ S_{fu}(\infty) &= \lim_{q\to 1} S_{fu}(q) \\ &= -C\left[I-(A-LC)\right]^{-1} B_u \ F_u \quad (4.9) \end{aligned}$$

Showing that at time instant $k = 0$ the fault is not visible, which is due to the fact that the fault needs to travel through the observer dynamics before it is visible in the output. We also depict that *the value of the fault in steady state is larger than at the initial instant*, that is

$$\|\mathbf{S_{fu}}(\infty)\| \ > \ \|\mathbf{S_{fu}}(\mathbf{0})\| \quad (4.10)$$

and *evolves as a function of the observer dynamics only*. We can also check the influence of the gain in steady-state for two extreme cases. The simulator, $L = 0$, and the predictor, $LC = A$, configurations.

$$\begin{aligned} S_{fu}(\infty)|_{L=0} &= -C\left[I-A\right]^{-1} B_u \ F_u \quad (4.11) \\ S_{fu}(\infty)|_{LC=A} &= -CB_u \ F_u \quad (4.12) \end{aligned}$$

Considering a positive A (all its elements positive) and a stable observer with $\|L\| > 0$, can compare Equations (4.9), (4.11) and (4.12) to conclude

$$\|\mathbf{S_{fu}}(\infty)|_{\mathbf{L=0}}\| > \|\mathbf{S_{fu}}(\infty)|_{\mathbf{0<LC<A}}\| > \|\mathbf{S_{fu}}(\infty)|_{\mathbf{LC=A}}\| \quad (4.13)$$

The condition above is related to the fact that $\|[qI-(A-LC)]\| \geq 1$ for any stable observer. Which comes from the fact that for any consistent norm, such that $\|AB\| \leq \|A\| \cdot \|B\|$, we have [53]:

Theorem 6

$$\|\mathcal{H}_M\| = \left\|(sI-M)^{-1}\right\| \geq \frac{1}{\min\limits_{1\leq i\leq n} |s-\lambda_i|}$$

Proof: *Let $(\lambda_i,\ v)$ be an eigenvector-value pair of M, then:*

$$(sI-M)v = sv - Mv = (s-\lambda_i)v$$

showing that $(s-\lambda_i)$ is an eigenvalue of $(sI-M)$. $(s-\lambda_i)^{-1}$ is then an eigenvalue of $(sI-M)^{-1}$ finally, since for any consistent norm $|\lambda| \leq \|M\|$, we have:

$$\max_{1\leq i\leq n} \frac{1}{|s-\lambda_i|} \leq \|(sI-M)^{-1}\|$$

whence

$$\|\mathcal{H}_M\| = \|(sI - M)^{-1}\| \geq \frac{1}{\min\limits_{1\leq i\leq n} |s - \lambda_i|}$$

A physical interpretation of Theorem (6) is that the eigenvalue measures the gain of a matrix only in one direction (given by eigenvectors) and must be smaller than for the whole matrix which allows any direction [60]. The norm of the resolvent matrix $\left\|(qI - M)^{-1}\right\|$ has, in fact, its lowest bound as $\frac{1}{\min\limits_{1\leq i\leq n} |s-\lambda_i|}$.

Since the observer is considered stable, we have that all its eigenvalues are smaller than 1, $\lambda_i^{A-LC} < 1$, depicting a relation to the norm and the eigenvalues of M. As faster it becomes (smaller eigenvalues), the smaller becomes $\|S_{fu}\|$ (worst case when $LC = A$, observer configured as a predictor). Notice that this is just an indication, the resolvent norm will vary, for example, with frequency.

The above condition gives some insights on the effect of the observer gain L. Because the predictor configuration uses the measurements directly in the estimations, these estimations will be contaminated by the fault very quickly (depending on the system order), while in the observer case, the estimations are balanced with the observer dynamics. Ultimately, for the simulator case, no measurements are used and the sensitivity function is directly taken by its dynamics in the system.

For additive process faults the sensitivity function is

$$\begin{aligned} S_{fa}(q) &= \left(I - C\mathcal{H}_{(A-LC)}L\right) C\mathcal{H}_A\, F_a \\ &= \left(I - C\left[qI - (A - LC)\right]^{-1} L\right)\, C\left[qI - A\right]^{-1}\, F_a \end{aligned}$$

Which can be simplified using Theorem (5) as

$$\begin{aligned} S_{fa}(q) &= \left(I - C\mathcal{H}_{(A-LC)}L\right)\, C\mathcal{H}_A\, F_a \\ &= \left(C\mathcal{H}_A - C\mathcal{H}_{(A-LC)}LC\mathcal{H}_A\right)\, F_a \\ &= C\left(\mathcal{H}_A - \mathcal{H}_{(A-LC)}LC\mathcal{H}_A\right)\, F_a \\ &= C\left(\mathcal{H}_A + \mathcal{H}_{(A-LC)} - \mathcal{H}_A\right)\, F_a \\ &= C\mathcal{H}_{(A-LC)}\, F_a \end{aligned} \tag{4.14}$$

Which depicts that though f_a affects the states of the system, it appears in the residual only as a function of the observer dynamics. And has a similar form as S_{fu}, but with opposing signals.

Similarly to an additive input fault, f_a will only be visible after a short interval and dependent *only* on the observer dynamics since the system dynamics does *not* affect f_a. The remain analysis follows similar to $S_{fu}(q)$.

For additive output sensor faults we have

$$\begin{aligned} S_{fy}(q) &= \left(I - C\mathcal{H}_{(A-LC)}L\right) F_y \\ &= \left(I - C\left[qI - (A-LC)\right]^{-1}L\right) F_y \end{aligned}$$

Analyzing its value at instant times $k = 0$ and $k \to \infty$ we have

$$S_{fy}(0) = \lim_{q\to\infty} S_{fy}(q) = F_y \tag{4.15}$$

$$S_{fy}(\infty) = \lim_{q\to 1} S_{fy}(q) = \left(I - C\left[I - (A-LC)\right]^{-1}L\right) F_y \tag{4.16}$$

Showing that at time $k = 0$, it is independent of L and that *the value of the fault in steady state is smaller than at the initial instant* and is a function of the observer gain and its dynamics *only*. With $\|L\| > 0$ we have ([50])

$$\|\mathbf{S_{fy}}(\infty)\| < \|\mathbf{S_{fy}}(\mathbf{0})\| \tag{4.17}$$

Checking the influence of the gain,

$$S_{fy}(\infty)|_{L=0} = F_y \tag{4.18}$$

$$S_{fy}(\infty)|_{LC=A} = \lim_{q\to 1} S_{fy}(q)|_{LC=A} = (I - CL)\ F_y \tag{4.19}$$

Yielding, for a stable observer with $\|L\| > 0$ and a positive A (all its elements positive):

$$\|\mathbf{S_{fy}}(\infty)|_{\mathbf{L=0}}\| > \|\mathbf{S_{fy}}(\infty)|_{\mathbf{0<LC<A}}\| > \|\mathbf{S_{fy}}(\infty)|_{\mathbf{LC=A}}\| \tag{4.20}$$

Moreover, in the especial case where $C = I$, $S_{fy}(q)$ can be simplified as:

$$\begin{aligned} S_{fy}(q) &= \left(I - C\mathcal{H}_{(A-LC)}L\right)\ F_y \\ &= \left(I - \mathcal{H}_{(A-L)}L\right)\ F_y \\ &= \left(\mathcal{H}_{(A-L)}\mathcal{H}^{-1}_{(A-L)} - \mathcal{H}_{(A-L)}L\right)\ F_y \\ &= \mathcal{H}_{(A-L)}\left(\mathcal{H}^{-1}_{(A-L)} - L\right)\ F_y \\ &= \mathcal{H}_{(A-L)}\mathcal{H}^{-1}_{A}\ F_y \\ &= \mathcal{H}_{(A-LC)}\mathcal{H}^{-1}_{A}\ F_y \end{aligned} \tag{4.21}$$

The result of the simplification in Equation (4.21) shows (using Theorem (6)), that S_{fy} is *related to the relative speed of the observer (eigenvalues of $A - LC$) and system (eigenvalues of A)*. That is, if the observer is faster than the system indicates, $\|S_{fy}\| < \|F_y\|$, conversely, if system has faster dynamics would cause $\|S_{fy}\| > \|F_y\|$ and if they are equal ($L = 0$ for instance) $\|S_{fy}\| = \|F_y\|$.

4.1.2.2: Some remarks

Observe that f_u and f_a have similar effects in the residual (check Equations (4.14) and (4.8)). Both are dependent only on the observer dynamics but affect the residual in opposite directions, that is, if f_a and f_u are affecting the same state, for example $F_a = B_u F_u$, and growimg in the same direction, their effects in the residual will be opposing to each other and with same value.

Moreover, notice that the simplification for deriving S_{fa} in Equation (4.14) implies a perfect system model while S_{fu} was derived directly and the user should be careful when analyzing it in a practical situation.

4.2: Observer-based residual generation through integrated sensors

The previous approach *requires* the availability of $y(k)$. As pointed in the introductory Chapter, a sensor output might be provided only after the raw measurements have been internally processed. Here, we will approach this problem with the *simplification that such sensors are integrated with standard obsevers/Kalman filters*.

The idea is to model the sensor as an augmented system model that includes

both the system and observer dynamics with *only* the estimates available

$$\begin{aligned} x(k+1) &= Ax(k) + B_u u_0(k) + F_a f_a(k) \\ \hat{x}(k+1) &= A\hat{x}(k) + B_u u(k) + L(y(k) - C\hat{x}(k)) \\ \hat{y}(k) &= C^* \hat{x}(k) \end{aligned} \tag{4.22}$$

where C^* depicts which *estimates* are available. With measured inputs $u(k)$ and outputs $y(k)$ also subject to additive faults as

$$u(k) = u_0(k) + F_u f_u(k) \tag{4.23}$$

$$y(k) = Cx(k) + F_y f_y(k) \tag{4.24}$$

Notice that only $\hat{x}(k)$ are affected by $f_u(k)$ but not $x(k)$.

By using the extended state $\bar{x}(k) = [x^T(k) \quad \hat{x}^T(k)]^T$ and the corresponding state space matrices

$$\bar{A} = \begin{bmatrix} A & 0 \\ LC & (A - LC) \end{bmatrix} \quad \bar{B}_u = \begin{bmatrix} B_u \\ B_u \end{bmatrix} \quad \bar{C} = \begin{bmatrix} 0 & C^* \end{bmatrix} \tag{4.25}$$

we can interpret this a standard fault detection problem, which could be approached by a parity space method or a Kalman filter based method and use for example

$$\bar{\varepsilon}(k) = \hat{y}(k) - \hat{\bar{y}}(k)$$

where $\hat{y}(k)$ is output from the integrated sensor simplified as Equation (4.22) and $\hat{\bar{y}}(k)$ is the observed output using the extended model as in Equation (4.25).

4.2.1: Fault observability

Considering a system as in Equation (4.22), where only estimated outputs $\hat{y}(k)$ are available but not $y(k)$, we can augment the faults in the states as

$$\bar{x}(t) = [x(k)^T \quad \hat{x}(k)^T \quad f_a(k)^T \quad f_y(k)^T \quad f_u(k)^T]^T$$

with the resulting matrices

$$\bar{C} = \begin{bmatrix} 0 & C^* & 0 & 0 & 0 \end{bmatrix} \tag{4.26}$$

$$\bar{A} = \begin{bmatrix} A & 0 & F_a & 0 & 0 \\ LC & (A-LC) & 0 & LF_y & B_uF_u \\ & & \mathbf{0} & & \mathbf{I} \end{bmatrix} \tag{4.27}$$

we can now analyze the observability for such system when considering faults as driven by the random-walk, $f(k+1) = f(k)$. We summarize the results first and present a more detailed proof in the sequence.

Given that the original pair (A, C) *is observable*, if the followings **conditions** are met:

1. All estimates are available and distinguishable (C is full column rank, for instance, $C^* = I$).
2. L is full column rank, such that

$$Lz = 0 \quad \Rightarrow \quad z = 0$$

the extended system observability can be tested by checking the rank of the matrix

$$\begin{pmatrix} (A-I) & F_a & 0 & 0 \\ LC & 0 & LF_y & B_uF_u \end{pmatrix}$$

which has a similar form as for the case when the actual measurements are available. This can be explained by the fact that the conditions on the gain and state availability provides the same information as when we have access to the raw measurements.

The same conditions for process and sensor output faults are achieved as for the usual case. While input sensor faults f_u follows similar conditions as f_y:

- For measurement faults only (f_y or f_u)

  If the system has no integrator dynamics, the faults will be observable as long as F_y (or B_uF_u) is full rank.

  If there is an integrator in the system dynamics, then the system is not observable if F_y (or B_uF_u) is full column rank and the states through which the fault

propagates should be orthogonal to the measured integrating part of the system (modes with eigenvalue equals to 1).

- For process faults only (f_a)

  Changes introduced by the dynamics of the system which are not directly measured must distinguishable (orthogonal) to the disturbances.

The demonstrations follows below.

Demonstration: We will use the Popov- Belevitch-Hautus (PHB) criteria, as presented in Chapter 3, to check the augmented system observability. The PHB depicts a pair (A,C) to be observable if

$$\begin{pmatrix} C \\ A - sI \end{pmatrix}$$

has rank equals n (the number of states) for all s. We will use the following theorems

Theorem 7 *The matrix*

$$\begin{pmatrix} A \\ B \end{pmatrix}$$

has full column rank if $\mathcal{N}_A \cap \mathcal{N}_B = 0$.

Proof:

$$\begin{pmatrix} A \\ B \end{pmatrix}$$

has full column rank if for any $v \neq 0$

$$\begin{pmatrix} A \\ B \end{pmatrix} v = \begin{pmatrix} Av \\ Bv \end{pmatrix} \neq 0$$

which means that the null spaces of A *and* B *are non-intersecting, or*

$$\mathcal{N}_A \cap \mathcal{N}_B = 0$$

Theorem 8 *Given* K *is full column rank, for any* B *we have:*

$$\mathcal{N}_{KB} = \mathcal{N}_B$$

Proof: *Suppose $\mathcal{N}_K = \emptyset$. Such that*

$$Kv = 0 \quad \rightarrow v = 0$$

Now, let us check the null space of KB

$$(KB)w = 0, \quad K(Bw) = 0 \quad \rightarrow Bw = r \in \mathcal{N}_K \quad \rightarrow Bw = 0$$

finally, the solution for $Bw = 0$ is the null-space of B and therefore,

$$\mathcal{N}_{KB} = \mathcal{N}_B$$

The faults modes, as shown in Equation (4.27), are $s = 1$, therefore the analysis will be separated for $s \neq 1$ and $s = 1$.

For $\mathbf{s \neq 1}$ the observability is given by:

$$\begin{pmatrix} 0 & C^* & 0 & 0 & 0 \\ A - sI & 0 & F_a & 0 & 0 \\ LC & (A - LC) - sI & 0 & LF_y & B_u F_u \\ & \mathbf{0} & & (1-s)\mathbf{I} & \end{pmatrix}$$

Following similar simplifications as in [52], we can check that the matrices $(1 - s)I$ will have full rank and with row operations (rank conserving) it can be rewritten as:

$$\begin{pmatrix} 0 & C^* & 0 & 0 & 0 \\ A - sI & 0 & 0 & 0 & 0 \\ LC & (A - LC) - sI & 0 & 0 & 0 \\ & \mathbf{0} & & (1-s)\mathbf{I} & \end{pmatrix}$$

And therefore, it is sufficient to check the rank of the matrix:

$$\begin{pmatrix} 0 & C^* \\ A - sI & 0 \\ LC & (A - LC) - sI \end{pmatrix} \tag{4.28}$$

which for $\mathbf{C}^*$ full rank, i.e $C^* = I$ is equivalent to analyze

$$\begin{pmatrix} A - sI \\ LC \end{pmatrix}$$

Finally, using Theorem 7 we have the condition that $\mathcal{N}_{A-sI} \cap \mathcal{N}_{LC} = \emptyset$ which using Theorem 8 is equivalent to $\mathcal{N}_{A-sI} \cap \mathcal{N}_C = \emptyset$ which is known to be true since we consider that the pair (A, C) is observable and the observability for $s \neq 1$ is achieved.

Notice that last step depicts that the observability is independent on the observer dynamics in case C^* is full rank. In case this is not true, $C^* = C$ for example with C non full rank, it is possible to find a full column rank L and a pair (A, C) that turns the system not observable and each case should be considered separately using Equation (4.28).

Example 9 *Take $C^* = C$ and*

$$A = \begin{pmatrix} 1 & 1 \\ -2 & -2 \end{pmatrix} \quad C = \begin{pmatrix} 1 & 0 \end{pmatrix} \quad L = \begin{pmatrix} l_1 \\ l_2 \end{pmatrix}$$

Note that we have $\mathcal{N}_A \cap \mathcal{N}_C = \emptyset$ (the system is observable). Considering that analyzing if a matrix M is full column rank is equivalent to

$$Mv = 0 \quad \Leftrightarrow v = 0$$

Then, the conditions are achieved by checking Equation (4.28) (with $s = 0$ for instance):

$$\begin{pmatrix} 0 & C \\ A & 0 \\ LC & (A - LC) \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} = 0$$

From which we get that $LCx + (A - LC)y = 0$. Moreover, since we also have that $y \in \mathcal{N}_C$ and $x \in \mathcal{N}_A$ it is equivalent to analyze x and y for such. To do so, we can form a basis $\tilde{A}$ which is formed by the eigenvectors of $\mathcal{N}_A$, $(1 \quad -1)^T$ for example, and equivalently take $\tilde{C}$ formed by eigenvectors of $\mathcal{N}_C$, for instance $(0 \quad 1)^T$. And the criteria can be rewritten as

$$LC\tilde{A}w + (A - LC)\tilde{C}v = 0$$

yielding

$$l_1 w + v = 0$$
$$l_2 w - 2v = 0$$

which only implies $w = v = 0$ *if* $-2l_1 \neq l_2$ *and therefore, there is a full column rank* L *that makes the system not observable (even when* $(A - LC)$ *is stable).*

For $\mathbf{s = 1,}$ the observability matrix is

$$\left(\begin{array}{ccccc} 0 & C^* & 0 & 0 & 0 \\ A-I & 0 & F_a & 0 & 0 \\ LC & (A-LC)-I & 0 & LF_y & B_uF_u \\ & & & & \\ & \mathbf{0} & & \mathbf{0} & \\ & & & & \end{array}\right)$$

and it is equivalent to analyze

$$\left(\begin{array}{cccc} A-I & F_a & 0 & 0 \\ LC & 0 & LF_y & B_uF_u \end{array}\right)$$

For process faults only we can analyze

$$\left(\begin{array}{cc} A-I & F_a \\ LC & 0 \end{array}\right)\left(\begin{array}{c} x \\ y \end{array}\right)$$

and we have the conditions, $(A-I)x + F_a y = 0$ and $x \in \mathcal{N}_{LC}$. For a full column rank L, the last condition can be rewritten as $x \in \mathcal{N}_C$. And it is equivalent to analyze such x, to do so we take a basis $\tilde{C}$ formed by the eigenvectors $\mathcal{N}_C$ and rewrite the first condition as

$$(A-I)\tilde{C}w + F_a y = 0$$

. This condition can only be true if $(A-I)\tilde{C}$ and F_a share image spaces and the condition on the observability can be rewritten as

$$\mathcal{R}_{(A-I)\tilde{C}} \cap \mathcal{R}_{F_a} = \emptyset$$

Depicting that changes introduced by the dynamics of the system which are not directly measured must distinguishable (orthogonal) to the disturbances.

For measurement faults only we can analyze

$$\begin{pmatrix} (A-I) & 0 \\ LC & W \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix}$$

where W can either relate to input sensor faults ($W = B_u F_u$) or output sensor faults ($W = LF_y$). We have the conditions that $x \in \mathcal{N}_{A-I}$ and $LCx + Wy = 0$. The first condition is true if and only if x is zero or an eigenvector of A with eigenvalue equals to 1. Hence, it is sufficient to analyze such x. Notice, though, that if A has no integrators (no eigenvalues equals to 1), then the conditions is achieved by checking if $W = B_u F_u$ (for input faults) or $W = LF_y$ (for output sensor faults) is full column rank.

If there are integrators in A with a basis $\tilde{A}$ formed by the eigenvectors of $\mathcal{N}_{A-I}$ the condition is rewritten as $LC\tilde{A}v + Wy = 0$

For f_y we have $LC\tilde{A}v + LF_y y = 0$. Which can be rewritten as

$$\mathcal{R}_{LC\tilde{A}} \cap \mathcal{R}_{LF_y} = \emptyset$$

Which means that the system is not observable if LF_y is full rank and the states through which f_y propagates should be orthogonal to the measured integrating part of the system propagated by L.

For f_u we have

$$\mathcal{R}_{LC\tilde{A}} \cap \mathcal{R}_{B_u F_u} = \emptyset$$

Which means that the system is not observable if $B_u F_u$ is full rank and the states through which f_u propagates should be orthogonal to the measured integrating part of the system propagated by L.

4.2.2: Residual analysis

As proposed in Section 4.2, we can use the extended state space model, with $C^* = C$

$$\bar{A} = \begin{bmatrix} A & 0 \\ LC & (A - LC) \end{bmatrix} \quad \bar{B_u} = \begin{bmatrix} B_u \\ B_u \end{bmatrix} \quad \bar{C} = \begin{bmatrix} 0 & C \end{bmatrix}$$

to design an observer producing a redundancy $\hat{\bar{y}}$

$$\begin{aligned}\hat{\bar{x}}(k+1) &= (\bar{A}-K\bar{C})\hat{\bar{x}}(k)+K(y(k))+\bar{B}_u u(k)\\ \hat{\bar{y}}(k) &= \bar{C}\hat{\bar{x}}(k) \quad \text{where} \quad K=[K_1^T \quad K_2^T]^T\end{aligned}$$

and use the residual

$$\bar{\varepsilon}(k)=\hat{y}(k)-\hat{\bar{y}}(k)$$

to fault detection purposes.

Following similar steps as in Section 4.1.2, we rewrite the residual in an input-output form yielding

$$\begin{aligned}\bar{\varepsilon}(k)= &\quad \left[\left(I-\bar{C}\bar{\mathcal{H}}_K K\right)C\mathcal{H}_L\left(I+LC\mathcal{H}_A\right)B_u-\bar{C}\bar{\mathcal{H}}_K\bar{B}_u\right]\ \mathbf{u_0}(k)\\ + &\quad \left[(I-\bar{C}\bar{\mathcal{H}}_K K)C\mathcal{H}_L LC\mathcal{H}_A\right]F_a\,\mathbf{f_a}(k)\\ + &\quad \left[(I-\bar{C}\bar{\mathcal{H}}_K K)C\mathcal{H}_L L\right]F_y\,\mathbf{f_y}(k)\\ + &\quad \left[\left(I-\bar{C}\bar{\mathcal{H}}_K K\right)C\mathcal{H}_L B_u-\bar{C}\bar{\mathcal{H}}_K\bar{B}_u\right]F_u\,\mathbf{f_u}(k)\end{aligned} \tag{4.29}$$

to simplify the notation, we have written

$$\begin{aligned}\mathcal{H}_{(A-LC)} &= \mathcal{H}_L\\ \mathcal{H}_{(\bar{A}-K\bar{C})} &= \bar{\mathcal{H}}_K\\ &\text{where}\\ \mathcal{H}_M &= (qI-N)^{-1}\end{aligned}$$

Using the block matrix inversion:

Theorem 10 *If A is invertible then:*

$$\begin{bmatrix} A & B\\ C & D\end{bmatrix} = \begin{bmatrix} A^{-1}+A^{-1}B\Delta^{-1}CA^{-1} & -A^{-1}B\Delta^{-1}\\ -\Delta^{-1}CA^{-1} & \Delta^{-1}\end{bmatrix}$$

where Δ, known as the Schur complement, is

$$\Delta=\left(D-CA^{-1}B\right)$$

Proof: *found in any linear algebra book, [53] for example.*

we can solve $\bar{\mathcal{H}}_K$ and show that the term multiplying the fault-free input u_0 equals to zero.

4.2.2.1: Fault sensitivity

As presented before, we can analyze the transfer functions $S_f(q) = \frac{\partial \varepsilon}{\partial f}$ from each fault to the residual in order to analyze how the faults will behave.

For additive input sensor faults we have:

$$\begin{aligned}\bar{S}_{fu}(q) &= \left[\left(I - \bar{C}\bar{\mathcal{H}}_K K\right) C\mathcal{H}_L B_u - \bar{C}\bar{\mathcal{H}}_K \bar{B}_u\right] F_u \\ &= [\Big(I - \bar{C}\left(qI - (\bar{A} - K\bar{C})\right)^{-1} K\Big) C\left(qI - (A - LC)\right)^{-1} B_u \\ &\quad -\bar{C}\left(qI - (\bar{A} - K\bar{C})\right)^{-1} \bar{B}_u] F_u\end{aligned}$$

which can be simplified using Theorem 10 and matrix identities to

$$\begin{aligned}\bar{S}_{fu}(q) &= -C\Delta^{-1} LC\left(qI - A\right)^{-1} B_u F_u \\ &= -C\Delta^{-1} LC\mathcal{H}_A B_u F_u \end{aligned} \tag{4.30}$$

where Δ is the Schur complement of $\bar{A}$ as

$$\begin{aligned}\Delta &= \left(qI - (A - LC - K_2 C) + LC\left(qI - A\right)^{-1} K_1 C\right) \\ &= \left(\mathcal{H}^{-1}_{(A-LC-K_2C)} + LC\mathcal{H}_A K_1 C\right)\end{aligned} \tag{4.31}$$

Equation (4.30) shows that the f_u will depend only on the sensor gain L and the augmented states observer and system dynamics. The system dynamics appear because the sensor is affected by the input fault also which is propagated by the augmented observer. The sensor dynamics however are compensated with use of the augmented observer, though, the sensor gain L still affects the residual.

Analyzing $\bar{S}_{fu}$ at instant times $k = 0$ and $k \to \infty$ we have

$$\bar{S}_{fu}(0) = \lim_{q\to\infty} \bar{S}_{fu}(q) = 0 \tag{4.32}$$

$$\begin{aligned}\bar{S}_{fu}(\infty) &= \lim_{q\to 1} \bar{S}_{fu}(q) \\ &= -C\left[I - (A - LC - K_2 C) + LC\left(I - A\right)^{-1} K_1 C\right]^{-1} \\ &\quad LC\left(I - A\right)^{-1} B_u \ F_u\end{aligned} \tag{4.33}$$

and consequently

$$\|\bar{\mathbf{S}}_{\mathbf{fu}}(\infty)\| > \|\bar{\mathbf{S}}_{\mathbf{fu}}(\mathbf{0})\| \tag{4.34}$$

It might be difficult to analyze the influence of the gain K in such residual through Δ. One should notice, for example, that it will be impossible to have $K\bar{C} = \bar{A}$ because $\bar{C} = [0 \quad C]$ is not full column rank and $K\bar{C}$ will have the form

$$K\bar{C} = \begin{bmatrix} 0 & K_1 C \\ 0 & K_2 C \end{bmatrix}$$

To be feasible to have $K\bar{C} = \bar{A}$ would require that the raw measurements, $y(k)$, are also available.

Some special cases are:

$$\begin{aligned}
\bar{S}_{fu}(\infty)|_{K=0} &= -CB_uF_u && (4.35)\\
\bar{S}_{fu}(\infty)|_{K_1=0} &= -C\left[I-(A-LC-K_2C)\right]^{-1} \\
&\quad LC\left[I-A\right]^{-1}B_u\ F_u && (4.36)\\
\bar{S}_{fu}(\infty)|_{K_2=0} &= -C\left[I-(A-LC)+LC\left(I-A\right)^{-1}K_1C\right]^{-1} \\
&\quad LC\left[I-A\right]^{-1}B_u\ F_u && (4.37)\\
\bar{S}_{fu}(\infty)|_{K_2=-L} &= -C\left[I-(A)+LC\left(I-A\right)^{-1}K_1C\right]^{-1} \\
&\quad LC\left[I-A\right]^{-1}B_u\ F_u && (4.38)\\
\bar{S}_{fu}(\infty)|_{K_2C=-(I-A+LC)} &= -C\left[LC\left(I-A\right)^{-1}K_1C\right]^{-1} \\
&\quad LC\left[I-A\right]^{-1}B_u\ F_u && (4.39)\\
\bar{S}_{fu}(\infty)|_{K_2C=-(I-A+LC),K_1=0} &= 0 && (4.40)
\end{aligned}$$

For additive process faults we have:

$$\bar{S}_{fa}(q) = \left[(I-\bar{C}\bar{\mathcal{H}}_K K)C\mathcal{H}_L LC\mathcal{H}_A\right]F_a$$

which can be simplified similarly as for $\bar{S}_{fu}$ to

$$\begin{aligned}
\bar{S}_{fa}(q) &= C\Delta^{-1}LC\left(qI-A\right)^{-1}F_a \\
&= C\Delta^{-1}LC\mathcal{H}_A F_a && (4.41)
\end{aligned}$$

where Δ is the Schur complement of $\bar{A}$ as in Equation (4.31). Which as in the case when the raw measurements were available, is similar to $\bar{S}_{fu}$ with opposite direction. The remaining analysis follows similarly to $\bar{S}_{fu}$.

For additive output sensor faults we have:

$$\bar{S}_{fy}(q) = \left[(I - \bar{C}\bar{\mathcal{H}}_K K)C\mathcal{H}_L L\right] F_y$$

which can be simplified to

$$\bar{S}_{fy}(q) \;=\; C\Delta^{-1}LF_y \tag{4.42}$$

where Δ is the Schur complement of $\bar{A}$ as in Equation (4.31).

It is interesting to notice here that the $\bar{S}_{fy}$ is only dependent on the augmented observer dynamics through Δ and on the sensor gain L. As occurred with $\bar{S}_{fu}$, and consequently to $\bar{S}_{fa}$, the augmented observer took away the influence of the sensor dynamics $\mathcal{H}_L$.

Continuing the analysis for initial and steady conditions we have:

$$\bar{S}_{fy}(0) \;=\; \lim_{q\to\infty} \bar{S}_{fy}(q) = 0 \tag{4.43}$$

$$\begin{aligned}\bar{S}_{fy}(\infty) &= \lim_{q\to 1} \bar{S}_{fy}(q) \\ &= -C\left[I - (A - LC - K_2C) + LC\left(I - A\right)^{-1} K_1 C\right]^{-1} LF_y\end{aligned} \tag{4.44}$$

and consequently

$$\|\bar{\mathbf{S}}_{\mathbf{fy}}(\infty)\| \;>\; \|\bar{\mathbf{S}}_{\mathbf{fy}}(\mathbf{0})\| \tag{4.45}$$

Again, it might be difficult to analyze the behavior of Δ with K. We can extend the results presented for f_u from Equations (4.35) to (4.40) by noticing that $\bar{S}_{fu} = \bar{S}_{fy}$ when $F_y = C\mathcal{H}_A B_u F_u$.

4.2.2.2: Some remarks

With the use of the augmented observer, the influence of the sensor dynamics $\mathcal{H}_{(A-LC)}$ is taken away for all S_f.

However, it might be difficult to analyze the influence of the augmented observer gain K through Δ. K_1 will affect the estimation of x (and consequently of $\bar{x}$ through the term LC) as it can be seem in the augmented observer matrix $\bar{A} - K\bar{C}$

$$\begin{bmatrix} A & -K_1C \\ LC & (A - LC) - K_2C \end{bmatrix}$$

while K_2 will affect the estimation of $\hat{x}$ only. Choosing $K_1 = 0$ would assume a perfect model for the internal structure of the measurement system, conversely $K_2 = 0$ assumes a perfect model of the sensor. A typical case would be when, in fact, the sensor is integrated through an observer (such as a Kalman filter or an extended Kalman filter) while the internal structure of the system is rather complex or nonlinear, in this case, if we are interested in state estimation for instance, one should choose a large K_1 but small K_2.

A great restriction with this approach is that it requires some knowledge of L. In fact, in case we have little knowledge on L, we can compensate it increasing K_2, that would try to correct its influence. In the next Section we present an approach where no knowledge at all in L is needed.

4.2.3: Residual generation - Unknown sensor structure

We study now, the case when the observer structure is not given, for example if L is unknown, we cannot directly use the augmented observer from Equation (4.22) for fault detection because we cannot build the extended model. To overcome this problem, let us make two approximations. Looking at the integrated sensor model

$$\begin{aligned} x(k+1) &= Ax(k) + B_u u_0(k) + F_a f_a(k) \\ \hat{x}(k+1) &= A\hat{x}(k) + B_u u(k) + L(y(k) - C\hat{x}(t)) \\ \hat{y}(k) &= C^* \hat{x}(k) \end{aligned}$$

With both sensor faults

$$\begin{aligned} y(k) &= Cx(k) + F_y f_y(k) \\ u(k) &= B_u \left(u(k) + F_u f_u(k)\right) \end{aligned}$$

The first approximation is

$$L(Cx_f(k) + F_y f_y(k)) \approx \bar{F}_f \bar{f}(k)$$

Here

$$x_f(k+1) = Ax_f(k) + F_a f_a(k) \tag{4.46}$$

and $x_f(k)$ is the fault contribution in $x(k)$. Since $f_i(k)$ is zero or a constant vector it is most important that this approximation holds stationary i.e. after transients.

The second approximation is

$$L(y(k) - C\hat{x}(k)) \approx \bar{v}(k)$$

where $\bar{v}(k)$ is a white noise process with a certain covariance matrix. This makes sense from a Kalman filter point of view where the innovations can be viewed as process noise, $\bar{v}(k) = L\epsilon(k)$ where $\epsilon(k) = y(k) - C\hat{x}(k)$ is a white innovation process.

This leads to the simplified model

$$\begin{aligned} \hat{x}(k+1) &= A\hat{x}(k) + B_u u(t) + \bar{F}_f \bar{f}(k) + \bar{v}(k) \\ \bar{y}(k) &= C^* \hat{x}(k) + \bar{e}(k) \end{aligned} \tag{4.47}$$

The artificial measurement noise $\bar{e}(k)$ can be used to cope with unmodeled characteristics of the system. For example, for sensors over a network or with a weak real-time performance, one can use $\bar{e}(k)$ to include jitter, missed samples, delays, etc. or to cope with sensor/system unknown dynamics.

After defining $\bar{e}(k)$, it can be used to tune a Kalman filter observer for the system as in Equation (4.47) and a standard parity space method or Kalman filter based method can be used to design a fault detection algorithm. For notation reasons, we will call the above filter a **tracker**.

4.2.3.1: Residual analysis

A residual defined as

$$\breve{\varepsilon}(k) = \hat{y}(k) - \breve{y}(k) \tag{4.48}$$

can be used to perform the detection. Here $\breve{y}(k)$ is the output estimated from an observer tuned from the model in Equation (4.47) and can be represented by the model

$$\begin{aligned} \breve{x}(k+1) &= (A - K_t C^*)\, \breve{x}(k) + B_u u(k) + K_t \hat{y}(k) \\ \breve{y}(k) &= C^* \breve{x}(k) \end{aligned} \tag{4.49}$$

Writing the residual in an input-output form we have

$$\begin{aligned}\breve{\varepsilon}(k) &= \left[\left(I - C^*\mathcal{H}_{(A-K_tC)}K_t\right)C\mathcal{H}_{(A-LC)}\left(I + LC\mathcal{H}_A\right) - C^*\mathcal{H}_{(A-K_tC)}\right]\, B_u\mathbf{u_0}(k) \\ &+ \left(I - C^*\mathcal{H}_{(A-K_tC)}K_t\right)C\mathcal{H}_{(A-LC)}LC\mathcal{H}_A F_a\, \mathbf{f_a}(k) \\ &+ \left(I - C^*\mathcal{H}_{(A-K_tC)}K_t\right)C\mathcal{H}_{(A-LC)}LF_y\, \mathbf{f_y}(k) \\ &+ \left[\left(I - C^*\mathcal{H}_{(A-K_tC)}K_t\right)C\mathcal{H}_{(A-LC)} - C^*\mathcal{H}_{(A-K_tC)}\right]B_uF_u\, \mathbf{f_u}(k) \end{aligned} \quad (4.50)$$

from which is easy to show that is independent to $u_0(k)$ when $C^* = C$.

4.2.3.2: Fault sensitivity

We can analyze the fault sensitivity transfer functions in this case for each fault. We will consider $C^* = C$ for all the analysis.

For additive input sensor faults we have:

$$\breve{S}_{fu}(q) = \left[\left(I - C\mathcal{H}_{(A-K_tC)}K_t\right)C\mathcal{H}_{(A-LC)} - C\mathcal{H}_{(A-K_tC)}\right]B_uF_u$$

which can be simplified to

$$\breve{S}_{fu}(q) = -C\mathcal{H}_{(A-K_tC)}LC\mathcal{H}_{(A-LC)}B_uF_u \quad (4.51)$$

which reveals that it will be sensitive to both sensor and tracker dynamics, but **not** to system dynamics.

Analyzing it for initial and steady states:

$$\breve{S}_{fu}(0) = \lim_{q\to\infty}\breve{S}_{fu}(q) = 0 \quad (4.52)$$

$$\begin{aligned}\breve{S}_{fu}(\infty) &= \lim_{q\to 1}\breve{S}_{fu}(q) \\ &= -C\left[I - (A - K_tC)\right]^{-1}LC\left[I - (A - LC)\right]^{-1}B_uF_u \end{aligned} \quad (4.53)$$

yielding

$$\|\breve{\mathbf{S}}_{\mathbf{fu}}(\infty)\| > \|\breve{\mathbf{S}}_{\mathbf{fu}}(\mathbf{0})\| \quad (4.54)$$

Checking Equation (4.51) and Theorem (6) that the $\breve{S}_{fu}$ is related to the speed of the tracker filter, which will eventually have smaller values as it becomes faster.

Analyzing the tracker gain K_t for some cases we can depict the steady state

relations

$$\breve{S}_{fu}(\infty)|_{K_t=0} = -C\left[I-A\right]^{-1}LC\left[I-(A-LC)\right]^{-1}B_uF_u \quad (4.55)$$

$$\breve{S}_{fu}(\infty)|_{K_t=L} = -C\left[I-(A-LC)\right]^{-1}LC\left[I-(A-LC)\right]^{-1}B_uF_u \quad (4.56)$$

$$\breve{S}_{fu}(\infty)|_{K_tC=A} = -CLC\left[I-(A-LC)\right]^{-1}B_uF_u \quad (4.57)$$

For additive process faults we have:

$$\breve{S}_{fa}(q) = \left(I - C\mathcal{H}_{(A-K_tC)}K_t\right)C\mathcal{H}_{(A-LC)}LC\mathcal{H}_AF_a$$

which can be simplified, to

$$\breve{S}_{fa}(q) = C\mathcal{H}_{(A-K_tC)}LC\mathcal{H}_{(A-LC)}F_a \quad (4.58)$$

and can be analyzed in a similar manner as for an additive input sensor fault, f_u.

For additive output sensor faults we have:

$$\breve{S}_{fy}(q) = \left(I - C\mathcal{H}_{(A-K_tC)}K_t\right)C\mathcal{H}_{(A-LC)}LF_y$$

which can be simplified to

$$\breve{S}_{fy}(q) = C\mathcal{H}_{(A-K_tC)}\mathcal{H}_A^{-1}\mathcal{H}_{(A-LC)}LF_y \quad (4.59)$$

which reveals that it will be sensitive to system, sensor and tracker dynamics. Notice though that is has a similar form as S_{fy} when $C = I$ (see Section 4.1.2.1).

Analyzing it for initial and steady states:

$$\breve{S}_{fy}(0) = \lim_{q\to\infty}\breve{S}_{fy}(q) = 0 \quad (4.60)$$

$$\begin{aligned}\breve{S}_{fy}(\infty) &= \lim_{q\to 1}\breve{S}_{fy}(q)\\ &= C\left[I-(A-K_tC)\right]^{-1}\left[I-A\right]\left[I-(A-LC)\right]^{-1}LF_y \quad (4.61)\end{aligned}$$

yielding

$$\|\breve{\mathbf{S}}_{\mathbf{fy}}(\infty)\| > \|\breve{\mathbf{S}}_{\mathbf{fy}}(\mathbf{0})\| \quad (4.62)$$

Similarly to S_{fy} when $C = I$ in Section 4.1.2.1, the value of $\|\breve{S}_{fu}\|$ will be related to the relative location of the eigenvalues of the tracker filter, sensor and system.

We can analyze the influence of K_t for some cases in steady state:

$$\breve{S}_{fy}(\infty)|_{K_t=0} = C\left[I - A - LC\right]^{-1} LF_y \tag{4.63}$$

$$\begin{aligned}\breve{S}_{fy}(\infty)|_{K_t=L} &= C\left[I - (A - LC)\right]^{-1}\left[I - A\right] \\ &\quad \left[I - (A - LC)\right]^{-1} F_y\end{aligned} \tag{4.64}$$

$$\breve{S}_{fy}(\infty)|_{K_tC=A} = -C\left[I - A\right]\left[I - (A - LC)\right]^{-1} LF_y \tag{4.65}$$

4.2.3.3: Some remarks

The results are similar to when we have access to the raw measurements, with the inclusion of the tracker dynamics in the fault sensitivity transfer functions.

4.3: Concluding remarks

In this Chapter, we presented some characteristics of observer based residual generation for fault detection. Special attention was paid for the case when there is no direct access to the raw measurements $y(k)$, only to the output of an observer-integrated sensor. $\hat{y}$.

We addressed the question on fault observability for both cases and have shown that under the conditions for sensor gain L and state availability through C^*

1. All estimates are available (for instance, $C^* = I$).
2. L is full column rank, such that

$$Lz = 0 \quad \Rightarrow \quad z = 0$$

one will have a similar case as if the original output was available.

In section 4.2 we proposed two solutions for the case when one has only access to $\hat{y}(k)$: using an augmented observer which requires knowledge on the sensor gain L and estimates both sensor and system states; using a regular observer (tracker) with the sensor output with a simplified model, with this approach, no *a priori* knowledge on L is needed.

The fault sensitivities functions for additive faults have been analyzed for

- $\varepsilon(k)$: the residual generated when $y(k)$ is available

- $\bar{\varepsilon}(k)$: the residual generated with augmented observer (requires some knowledge on L) and only $\hat{y}(k)$ is available.
- $\breve{\varepsilon}(k)$: the residual generated with tracker observer and only $\hat{y}$ is available.

We summarize these results in Table 4.1. The reason why $\breve{\varepsilon}(k)$ would be worse than

	$\varepsilon(\mathbf{k}) = \mathbf{y}(\mathbf{k}) - \hat{\mathbf{y}}(\mathbf{k})$	$\bar{\varepsilon}(\mathbf{k}) = \hat{\mathbf{y}}(\mathbf{k}) - \bar{\hat{\mathbf{y}}}(\mathbf{k})$	$\breve{\varepsilon}(\mathbf{k}) = \hat{\mathbf{y}}(\mathbf{k}) - \breve{\mathbf{y}}(\mathbf{k})$
$\mathbf{S_{fu}}$	$-C\mathcal{H}_{(A-LC)}B_u\, F_u$	$-C\Delta^{-1}LC\mathcal{H}_A B_u\, F_u$	$-C\mathcal{H}_{(A-K_t)}LC\mathcal{H}_{(A-LC)}B_u\, F_u$
$\mathbf{S_{fa}}$	$C\mathcal{H}_{(A-LC)}\, F_a$	$C\Delta^{-1}LC\mathcal{H}_A\, F_a$	$C\mathcal{H}_{(A-K_t)}LC\mathcal{H}_{(A-LC)}\, F_a$
$\mathbf{S_{fy}}$	$\left(I - C\mathcal{H}_{(A-LC)}L\right)\, F_y$	$C\Delta^{-1}L\, F_y$	$C\mathcal{H}_{(A-K_t)}\mathcal{H}_A^{-1}\mathcal{H}_{(A-LC)}\, F_y$
Pros	Straightforward. Good fault sensitivity. Easy to tune.	Does not require $y(k)$. Simplify the sensor influence in the sensitivity functions.	Simple. Easy to tune. Does not require L. Does not require $y(k)$.
Cons	Require $y(k)$.	Require L. Complex. Difficult to tune.	Influenced by sensor. Low fault sensitivity.

Table 4.1: Comparison of residual generation approaches.

$\bar{\varepsilon}$ is related to the fact that we have the sensor dynamics affecting the fault sensitivity functions as design invariant while with $\bar{\varepsilon}(k)$ they are affected by the system dynamics as design invariant, which are slower, therefore increasing the sensitivity function (see Theorem (6)). For similar reasons we have also that $\varepsilon(k)$ would yield a possible best residual.

Noise and uncertainties The reasoning that the simulator should always be used is only true when dealing with a perfect system model and no noise, which is not practical. Nevertheless, it would be advisable to choose lower values for the gain since that would increase the fault sensitivity. In [30], Chapter 12, the author presents a more detailed discussion on modeling errors, how they affect the residual relative to input $u(k)$ (first order error effect) and to the faults $f_i(k)$ (second order error effect).

Optimal residual and the observer design The problem of selecting the gain of the observer is actually an optimization problem where we would like to have the least influence of model uncertainties and disturbances in the residual with the most sensitivity to faults. The way faults, noise and model disturbances affect the system is usually different. It is expected that noise, for example, would appear with low power and high frequency, while model disturbances with an average power and low frequencies. These

information are important and can be used to design the residual generator. However, it might be difficult or even impossible to quantify/estimate the effects of these variables in the output.

Optimal residual generation is a live research topic in the research community, see for example [56, 57, 58, 59]. In [56], the author presents closed form solutions for the optimal residual generator in relation to disturbances, however, it requires enough information on disturbances and faults (a perfect model). A main limitation common to the techniques developed so far is that the faults need to be directly contained in the output $y(k)$. In other words, considering the model

$$\begin{aligned} x(k+1) &= Ax(k) + B_u u(k) + B_d d(k) + B_f f(k) \\ y(k) &= Cx(k) + D_u u_(k) + D_d d(k) + D_f f(k) \end{aligned}$$

would recquire D_f to be full column rank ([56], page 18). This is usually not the case for process faults and in case we only have the output from a sensor integrated with an observer,

$$\begin{aligned} \hat{x}(k+1) &= (A - LC)\hat{x}(k) + B_u u(k) + Ly(k, \mathbf{d}, \mathbf{f}) \\ \hat{y}(k) &= C\hat{x}(k) + D_u u_(k) \end{aligned}$$

this will never be the case, since the faults appear through the observer dynamics in the output $\hat{y}(k)$.

Kalman filtering Notice that the application of a Kalman filter for fault detection is similar to the one with an observer. As shown in the previous Chapter, under the assumptions of constant known noises covariances and model, the Kalman filter is analogous to an observer with $L = AK_o$, where K_o is the Kalman filter gain. Its use is specially suitable for systems with large noise since it will attenuate its influence, however it demands the user extra information about the system and sensors.

Part II

Application example

5 Wheeled Mobile robots

A wheeled mobile robot (WMR) is an automatic machine that is capable to move within an environment utilizing wheels for this purpose. As illustrated in Figure 5.1, WMRs finds a very wide range of applications in both industry and service, with very different demands.

Figure 5.1: A container AGV, a robot car, aan utomated vacuum cleaner and a multi-purpose mobile platform illustrates the wide range of applications of mobile robots.

Independent of the robot structure and wheel configuration, the task of localization is crucial. This is probably still the biggest challenge for the research community in the area and there has been several improvements and development of different methods and sensors to achieve this task. So far, the main focus has been in the improvement of sensors accuracy and reliability of the methods. It is a great challenge for any localization method to deal with unexpected changes/conditions in both environment and robot. Localization reliability can be improved by adding constraints to the environment the robot interacts with, for example restricting a robot use to even floor, with a drawback of restricting its application.

Fault/change detection can be used to improve these systems reliability with relaxed operational constraints. By detecting a fault in the localization system of the robot, its consequences can be attenuated, increasing the system robustness. In the past decades, there has been an increased interest in change detection methods for

mobile robots, for example [31, 32, 1]. The field, however, is comparatively recent in the mobile robots research community with a lot of space for improvements.

In this work we discuss observer-based fault detection for WMR localization systems. As in any model-based detection framework, the understanding of the system is crucial. In this Chapter, we present both WMR modeling and localization methods characteristics. We focus our analysis in differential wheel drive robots and in odometry and scan matching localization methods which will provide an introduction to main challenges to the problem and to the following Chapter which presents a localization fault detection framework and provide some comparative analyses.

5.1: Robot modeling

A mobile robot is a complex system where the dynamics and kinematics aspects are very important to its general performance. Different models are required according to different robot structures and applications. Though kinematic models have been successfully used in several applications, for fast moving robots, such as cars, dynamical models are crucial for performance increase.

Different robot structures will, obviously, require different models. The usual approach to model a robot mobility is to analyze the contribution of each wheel to its motion [33]. Active wheels enable the robot to move but also add constraints to the motion (for example, a car can not move in a 0 radius circular path).

In this work we will only consider a simple kinematic model, assuming the robot as a rigid body moving in a horizontal plane. The Figure below illustrates the robot in a global reference frame, where θ is the orientation of the robot relative to the reference frame, $[u_f \quad u_w]^T$ are the forward and rotational velocities of the robot in respect to its own frame. It is easy to show the following relation between the robot position and orientation in the place, called here as *pose*, $[x \quad y \quad \theta]^T$, and its directional speeds, $[u_f \quad u_w]^T$:

$$\begin{bmatrix} x \\ y \\ \theta \end{bmatrix}_{t+1} = \begin{bmatrix} x \\ y \\ \theta \end{bmatrix}_t + \begin{bmatrix} T\,cos(\theta) & 0 \\ T\,sin(\theta) & 0 \\ 0 & T \end{bmatrix}_t \begin{bmatrix} u_f \\ u_w \end{bmatrix}_t \tag{5.1}$$

where T is the sampling period. Though this model is a rough approximation, it will be enough for the use within the scope of this thesis. For a more thorough modeling, the

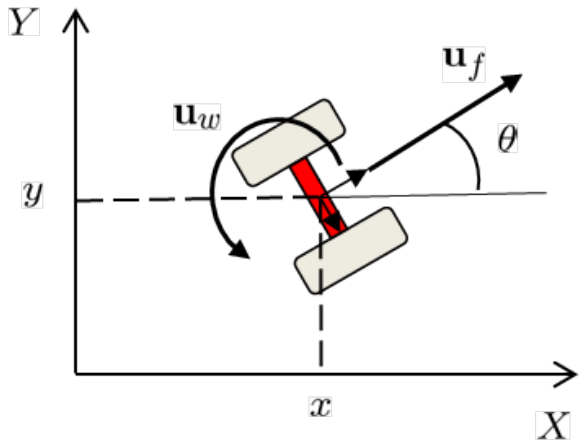


Figure 5.2: A mobile robot in a reference frame.

reader is advised to check the vast literature on vehicle modeling, [34] for example.

For a differential wheel drive robot, the following relation can be depicted between the directional speeds $[u_f \quad u_w]$ and the wheel speeds:

$$\begin{bmatrix} u_f \\ u_w \end{bmatrix}_t = \begin{bmatrix} \frac{r_R}{2} & \frac{r_L}{2} \\ \frac{r_R}{b} & \frac{r_L}{b} \end{bmatrix}_t \begin{bmatrix} w_R \\ w_L \end{bmatrix}_t \tag{5.2}$$

where b is the distance between wheels along their common axis, r_i and w_i are the wheel radius and wheel rotational speed at side i. This model consider the parameters r_i and b to be constant and, also common, with $r_R = r_L$, these are simplifications that the wheels and chassis are rigid with only one contact point to the ground.

5.2: Pose providers

There are several sensors/methods available to estimate a robot pose, from simple odometry to Global Positioning Systems (GPS). One might argue that with the advent of global positioning systems such as GPS, the localization problem is now solved, this reasoning is partly true and, in fact, such sensors play a significant role in solving the problem. However, the achieved accuracy for a GPS may vary from centimeters to tenths of meters (depending, amongst other factors, on atmospheric conditions) which is unviable in more demanding applications such as service robots.

The Figure below, extracted from [35] gives a scaling perspective of the several different pose providers methods/sensors:

In the following Section we explore two of the several available methods available

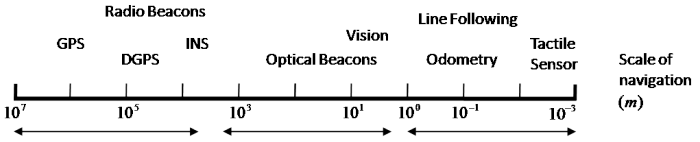


Figure 5.3: Pose providers methods/sensors in a navigation scale.

for pose estimation. At first, odometry, calculated from wheel speeds rotation is analyzed, followed by scan matching. A discussion on the error sources and challenges with the methods is also held.

5.3: Odometry

Odometry was one of the first localization methods developed historically and is still important for its simplicity, low cost and good short-term accuracy.

The position estimations are achieved by first solving the robot kinematics from wheel speeds to directional speeds using a kinematics model such as found in Equation (5.2) for a differential drive robot. With the estimates u_f and u_w, and given an initial pose, it is possible to achieve an estimate of the current pose through the model presented in Equation (5.1).

The velocity can be measured in several different manners. A common and very usual one is the use of **optical encoders**. As presented in Figure 5.4, such devices, when coupled to a rotating axis, estimate velocity by checking the time interval between consecutive blanks seen by a photo sensor. There are several different configurations for the device with increased accuracy depending on the number of ticks in the encoder wheels and the clock speed. For the application in mobile robots, the errors with these measures remain bounded and in a much smaller range when compared to other uncertainty sources and are often assumed to be insignificant [33].

5.3.1: Error sources

Odometry is based on the assumption that wheel revolutions can directly relate to linear displacements, for example, Equation (5.2) presents such relations for a differential drive robot. This model is only valid under some simplifications such as a robot moving in a fixed plan, a constant and unique contact point to the wheels, perfect alignment of wheels and so on. These simplifications and other stochastic effects may

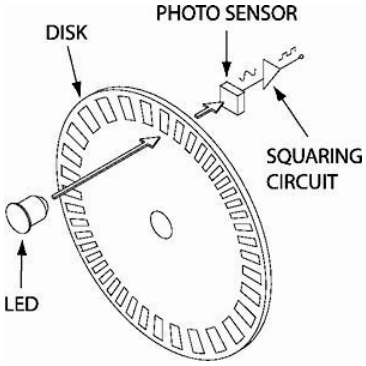


Figure 5.4: Optical encoder scheme. Extracted from the Wikipedia.

and will lead to errors in the odometry readings.

The errors can be categorized under two groups, systematic and random errors. Systematic errors are caused by known sources and/or assumptions that can be (sometimes) easily handled, some examples:

- Error introduced by uneven wheel radii.

  Caused for example by a badly calibrated tire.

- Actual wheelbase differing from nominal (used to solve the odometry equations).

  Caused for example by the use of a thick inflatable tire with several and not constant contact point to the ground.

- Misalignment of wheels.

  Introduced by a bad mechanical design or fatigue caused by extensive use.

- Finite wheel encoder resolution and finite sampling rate.

  A wheel encoder based sensor measures a discrete quantity and therefore resolution and sampling rate should be considered.

While most of the systematic error can be reduced with a better calibration of the system (see [36] for an example), there are other sources of error that affect the system in a stochastic manner and cannot be compensated for in a predictive manner, some examples:

- Travel over uneven floors.

  Should be faced in any outdoor application in which the robot path may vary considerable.

- Unexpected interaction with other entities.

  A common situation for robots that are designed to interact with people, when someone may push it away for example.

- Wheel slippage.

  There are several causes for wheel slippage: overacceleration of the wheels, driving over slippery floors, forces due to the castor wheels, skidding, etc.

Depending on where the robot is being operated, systematic errors can be less or more important to the overall performance of the robot. For example, if the robot is traveling in a flat and clean floor, the systematic errors will be much larger than the random ones.

5.3.2: Error modeling

A good understanding of the errors evolution, causes and behavior is of great relevance for the localization task as well as for a detection scheme. There has been several efforts in modeling odometry errors, [33, 37, 38], a model for odometry errors is very important in allowing sensor fusion techniques and the use of Kalman filter observers.

In [33], for example, the equations for the covariance error matrix of odometry are derived for differential drive robots by considering piecewise travel motions updated according to a kinematics robot model, Equation (5.1), and robot mobility model, Equation (5.2). The incremental traveled distances are:

$$p_{t+\Delta_t} = \begin{bmatrix} x \\ y \\ \theta \end{bmatrix}_{t+\Delta_t} = \begin{bmatrix} x \\ y \\ \theta \end{bmatrix}_t + \begin{bmatrix} \Delta s \, \cos(\theta_t + \frac{\Delta\theta}{2}) \\ \Delta s \, \sin(\theta_t + \frac{\Delta\theta}{2}) \\ \Delta\theta \end{bmatrix} \tag{5.3}$$

with

$$\Delta\theta = \frac{\Delta s_R - \Delta s_L}{b}, \quad \Delta s = \frac{\Delta s_R + \Delta s_L}{2}$$

where Δs_i is the traveled distance at side i. The final odometry position update is then

written as:

$$p_{t+\Delta t} = p_t + \begin{bmatrix} \frac{\Delta s_R + \Delta s_L}{2} \cos(\theta_t + \frac{\Delta s_R - \Delta s_L}{2b}) \\ \frac{\Delta s_R + \Delta s_L}{2} \sin(\theta_t + \frac{\Delta s_R - \Delta s_L}{2b}) \\ \frac{\Delta s_R - \Delta s_L}{b} \end{bmatrix} \tag{5.4}$$

Which is a function $f(p_t, \Delta s_L, \Delta s_R)$. Finally, with the assumptions that Δs_L and Δs_R are uncorrelated and proportional to the absolute traveled distance $|\Delta s_L|$, $|\Delta s_R|$ respectively, the covariance matrix Σ is computed as

$$\Sigma_{t+\Delta t} = \nabla_{p_t}(f)\, \Sigma_{p_t}\, \nabla_{p_t}(f)^T \quad + \quad \nabla_{\Delta s_L, s_R}(f)\, \Sigma_{\Delta s_L, s_R}\, \nabla_{\Delta s_L, s_R}(f)^T$$

where ∇_i is the Jacobian matrix in relation to i. From the Equation above, Σ_{p_t} is resulted from the previous estimation of Σ, while $\Sigma_{\Delta s_L, s_R}$ at initial condition is a design parameter that include motion errors due to a deformed wheel, slippage, encoder errors, etc. Figure 5.5 shows the odometry error estimated using this approach for a straight line

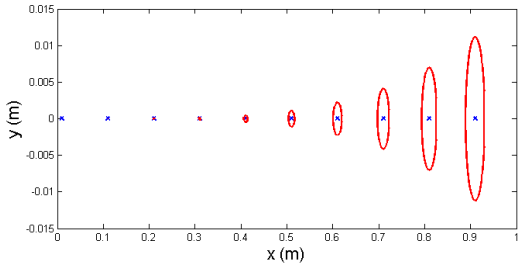


Figure 5.5: Odometry error for a straight line path following. The ellipses represents the computed uncertainty in x and y directions.

path. It is easy to depict the larger influences on the perpendicular direction of motion, which is due to the integration of the uncertainty of the robots orientation.

5.4: Laser scan matching

A laser range finder is a device that measures distances from a laser source and objects in the surroundings. There are several ways to estimate the distance, one is to measure the time of flight a laser beam takes to return to the device after reflected by an object, it can be expressed as $d = \frac{c\,t}{2}$, where c is the speed of light.

With the use of a mechanical mechanism with a mirror, the device sweeps the surroundings generating a rough estimate of the scene over a plane or even in 3D. Figure 5.6 shows a SICK LMS200, which can be configured with a $180°$ scanning angle,

Figure 5.6: Laser range finder from SICK.

resolution of $0.5°$ and $10m$ range, the device takes 4 scans in 1 second. A result of such scan is shown in Figure 5.7 where 361 points are taken in one scan.

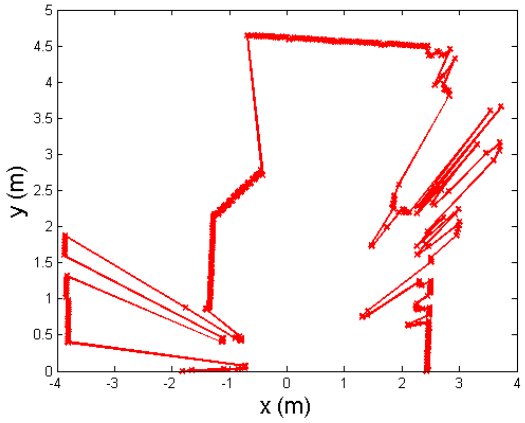


Figure 5.7: 2D laser scan over $180°$. The $(0, 0)$ position coincides to the laser source.

Scan matching consists of estimating the roto-translation relation between two different scans. Consider, for example, Figure 5.8 where two scans were taken from a laser range finder mounted in a mobile robot that was rotated clockwise from time t_0 to t_k. The scan matching algorithm finds the transform between the scans and consequently an estimate of how the robot has moved.

There are many algorithms available to perform scan matching. They might differ considerably from the scope of localization (global or local). Global localization takes no prior information of the robot position while for local localization some knowledge of previous estimates are available (from odometry for example).

Scan matching algorithms also differ by the kind of information they look into the scans in order to find the transform, some algorithms might only compare points in the scans while others might look at scene features (searching for lines for example), which in turn requires some assumptions on the environment. In the following Sections

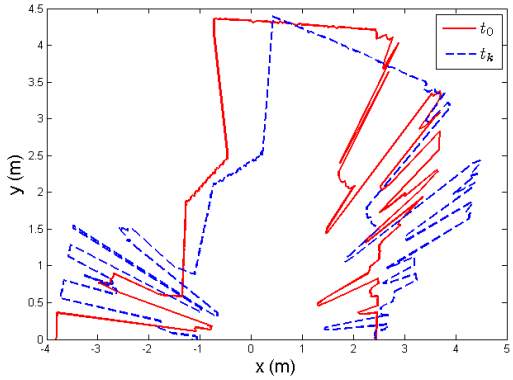


Figure 5.8: 2D laser scans over $180°$ with a rotation clockwise rotation between them.

we present two approaches for scan matching.

5.4.1: Iterative Closest Point - ICP

As the name points out, this is an iterative method which compares the scans in a point to point manner. It is one of the most common approaches for local scan matching and has a wide number of variants. It was first proposed in [40]. Given a reference, a current scan and an initial estimate of the transform relating them, ICP consists of the following tasks:

- **Transform** the new scan using the current estimate of the transformation.
- **Selection** of some set of points in one or both scans.

  Since the robot is moving, some points in the new scan might not even be seen in the reference scan, choosing which point should be used is therefore very important.
- **Matching** the selected points between the scans.

  In this task the points selected in one of the scans are corresponded to points in the other scan.
- **Rejecting** some of the pairs that might deviate too much.

  The matching is not perfect and some pairs should be discarded.
- **Updating** the transform given the matched points.

The algorithm iterates until it converges to a match exceeding some criteria. There are several variants of the algorithm, differing in how it solves the above tasks see [39, 41]

for example. The ICP tries to find the transform, T, that minimizes the total squared error between the scans.

$$T_{min} = \arg\min_{\mathbf{T}} \frac{1}{N} \sum_{i=1}^{N} \|\mathbf{T}r_i - s - i\| \tag{5.5}$$

where r_i and s_i are points in a reference and a current scan.

Rather than presenting a full proof of the solution of the above equation, we shall present a simple implementation of the algorithm, based in [40].

Given two scans (R, S) and an initial roto-translation transform T_0 between them, the algorithm iterates in a loop as shown in Algorithm (1) until some criteria is reached. To simplify some of the calculations one can use an homogeneous transform, including both rotation and translation in a joint transform as below:

$$T = \begin{bmatrix} Rot & Trans \\ 0 & 1 \end{bmatrix} = \begin{bmatrix} \cos(\theta) & \sin(\theta) & t_x \\ -\sin(\theta) & \cos(\theta) & t_y \\ 0 & 0 & 1 \end{bmatrix}$$

Algorithm 1 $(T) = \mathbf{match}(R, S, T_0)$. Main scan matching loop.

Require: $T_0 \neq \mathbf{0}$
  $T = T_0$
  while $\Delta T >$ criteria **do**
    $(S^*) = T \times S$ {apply current transform.}
    $(assoc) = \mathbf{getAssociatons}(R, S^*)$ {get p2p associations between scans.}
    $(R', S') = \mathbf{rejectOutliers}(R, S^*, assoc)$
    $(\Delta T) = \mathbf{getTransform}(R', S')$ {get matching transform.}
    $(T) = \Delta T \times T$ {update transform estimate.}
  end while
  return T

The $\mathbf{getAssociatons}$ function simply relates a point in the R scan with the closest point in the S scan, assuring that only one point is associated to another.

Due to changes in the scene between the scans, it could happen that a point in S is not present in R, the use of such points should be avoided in the matching process. The function $\mathbf{rejectOutliers}$ performs the task of excluding such points. There are several ways to reject outliers, here we use a very simple approach, taking away points that are below the median value of the distances of all associations.

Finally, the function $\mathbf{getTransform}$ takes the scans and current associations and return the transform between them.

Algorithm 2 $(T) = \mathbf{getTransform}(R,\ S)$. Compute the roto-translation transform between scans R and S.

$m_R = \frac{1}{N}\sum_{i=1}^{N} r_i$ {compute the centroid.}
$m_S = \frac{1}{N}\sum_{i=1}^{N} s_i$
$R' = R - m_R$ {subtract the centroid.}
$S' = S - m_S$
$M = \left[\sum_{i=1}^{N} r'_{ix}\ s'_{ix} \quad \sum_{i=1}^{N} r'_{iy}\ s'_{ix}\ ;\ \sum_{i=1}^{N} r'_{ix}\ s'_{iy} \quad \sum_{i=1}^{N} r'_{iy}\ s'_{iy}\right]$
$R = VU^T$ where $M = UDV^T$ {Singular Value Decomposition to solve rotation.}
$t = m_R - R\, m_S$ {translation parameters estimate.}
$T = [R \quad t\ ;\ 0 \quad 1]$ {return the transform in homogeneous coordinates.}
return T

As pointed out in [40] the ICP will always converge to a local minima. However, there are still important practical aspects that should be considered when utilizing ICP for localization.

- *Speed of convergence:* when the scans differ too much, the number of iterations needed might be to large in order to make it useful in practice. Some modifications of the original algorithm have been presented in order to increase the convergence speed [1, 39].

- *Update points of the reference scan:* one could always take two consecutive scans in order to perform the match, accumulating the pose estimates over time in order to get the robot current pose. However, in this manner, the current pose estimate will accumulate all past matching errors.

  Having a more invariant reference scan would then keep the errors on the current estimate lower.

  However, unless the robot mobility is previously constrained, the reference scan should always vary. For example, take a robot equipped with a $180°$ laser scanning range rotating around its initial position, the matching is expected to worsen as the reference scan is too far from the actual scan, with the worst case when they are $180°$ delayed.

- *Scene dependence:* the quality of the matching will be related to the scene. It is easy to understand that is easier to perform the matching for a robot moving in a squared room (rectilinear environment) rather than for a robot navigating in a jungle or in a crowded place (changing environment).

  In some cases, the estimates could be totally wrong. For instance, when the robot is following a straight path in a corridor longer than its laser range, the matchings

will be 'blind' to the changes occurring in the robot direction of movement. It is also the case for the orientation estimates for a robot navigating in a circle whose variance would be, theoretically, unbounded.

- *Error modeling:* for all the above mentioned aspects, modeling the error covariance of scan matching algorithms is a comprehensive task. There has been efforts in this direction, se for example [42, 43, 44].

A main drawback of the method is its need of a initial estimate of the transform relating the scans. A common approach is to use odometry measurements for this purpose. However, the estimates from scan matching would be coupled to odometry, which is not desirable in a fault detection scheme. The ICP algorithm would converge faster also in case there was some a priori knowledge on the rotation, since it is the parameter that recquires more iterations to be solved.

There are different algorithms to achieve global localization, [45, 46], in the next Section we shall elucidate the matching in the Hough domain as proposed in [46], with focus on the rotation estimations.

5.4.2: Matching in the Hough domain - HSM

The Hough transform describes an image in terms of the parameters of a feature, by doing so, it makes easy to identify features in an image by searching for local maxima in the Hough (parameters) domain. It was first introduced in a patent by P. Hough, [47], in 1962 as a method to detect lines in images, but later on the results were extended to detect also other features, such as circles and ellipses.

Background theory

The Hough transform for line extraction describes the image in line parameters. The idea is to vary the parameters of a straight line for each pair $(x,\ y)$ in an image, representing the image in the line parameters domain.

Because the usual representation of straight lines as $y = ax + b$ produces unbounded values for both slope and intercept parameters, a more convenient line representation [48] is

$$y = \left(-\frac{\cos(\theta)}{\sin(\theta)}\right) x + \left(\frac{\rho}{\sin(\theta)}\right)$$

The parameter ρ is the length of a normal from the origin to this line while θ is the

orientation of ρ in respect to the abscissas, see Figure 5.9. Finally, the representation

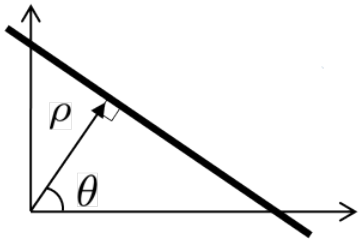
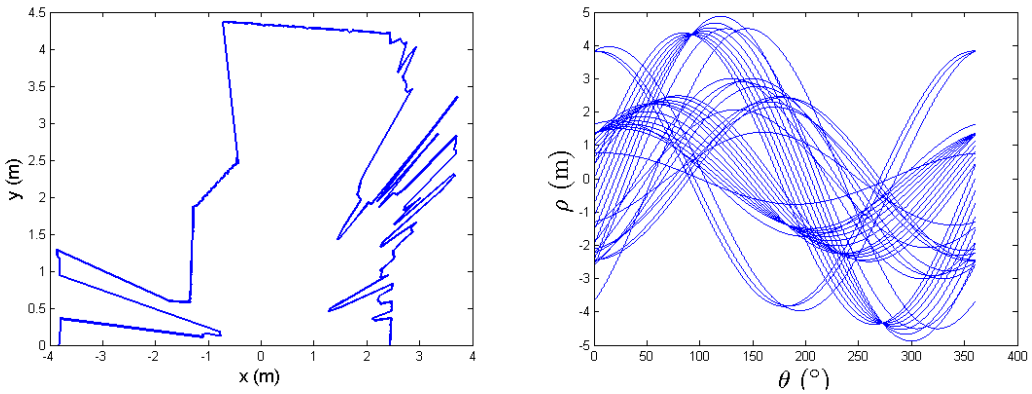


Figure 5.9: $(\rho,\ \theta)$ line parametrization.

can be rewritten as:

$$\rho = x\ \cos(\theta) + y\ \sin(\theta) \tag{5.6}$$

The result of a Hough transform for a selected set of points in a scan can be seen in Figure 5.10.



(a) Orginal scan (b) After HT

Figure 5.10: Hough transform representation.

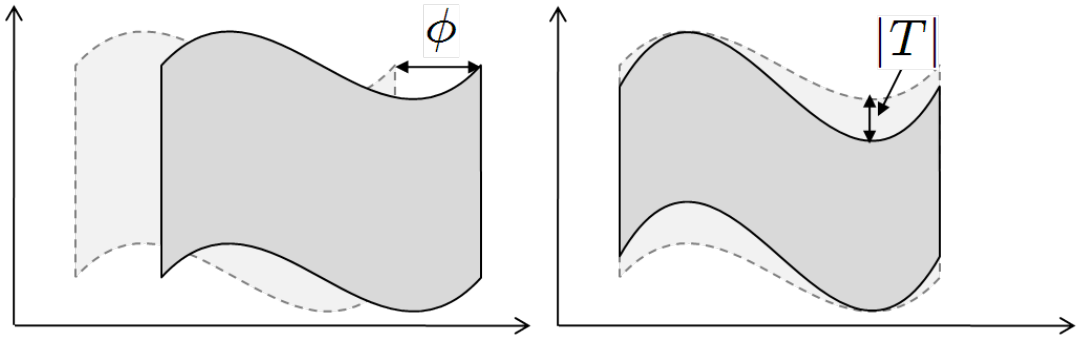
If we restrict the θ domain to $[0, \pi)$, than ρ is unique for a line. Some other interesting properties of the transform [48]:

- A point in the $(x,\ y)$ domain corresponds to a sinusoid in the $(\rho,\ \theta)$ domain.
- A point in the $(\rho,\ \theta)$ domain corresponds to a line in the $(x,\ y)$ domain.
- Collinear points in the $(x,\ y)$ corresponds to intercepting curves in the $(\rho,\ \theta)$ domain.
- Points on the same curve in the $(\rho,\ \theta)$ domain corresponds to lines through the same point in the $(x,\ y)$ domain.

Finally, and most relevant for our purposes, is the analysis of rigid body transformations ([46]):

- A rotation ϕ in the $(x,\, y)$ domain corresponds to a translation in the $(\rho,\, \theta)$ domain.
- A translation T in the $(x,\, y)$ produces a bend over ρ in the $(\rho,\, \theta)$ domain.

These properties are illustrated in Figure 5.11.



(a) Rotation ϕ (b) Translation $|T|$

Figure 5.11: Effects of image transformation in the Hough domain.

5.4.2.1: Rotation estimation through spectra correlation

As proposed by Censi in [46], it is possible to get an estimate of the rotation, ϕ, by searching for local maxima in the Hough transform spectra cross-correlation. The author computes the spectrum over a function g that is translation invariant. That is, for $f'(x, y) = f(R_\phi(x, y) + T)$:

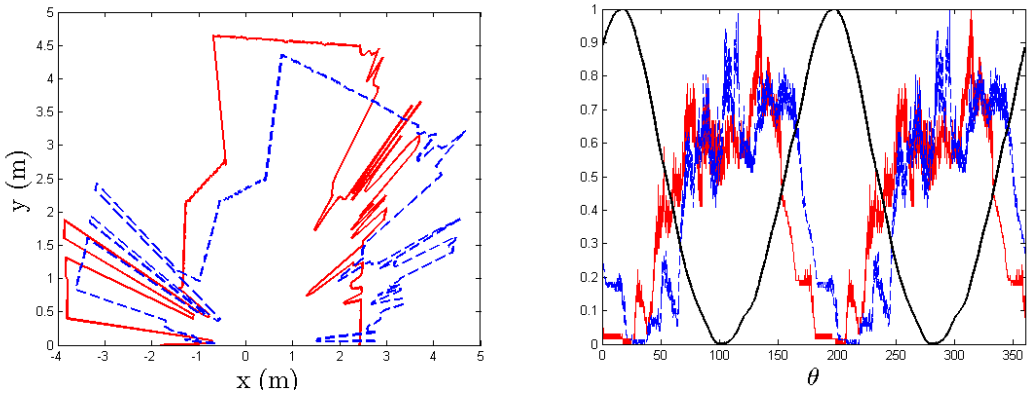
$$HS_g[f](\theta) = HS_g[f'](\theta + \phi) \tag{5.7}$$

where HS denotes the Hough spectrum. A simple choice for g, considered by the author, is the energy sequence, others could be used such as the Fourier transform, but only increasing the computational efforts.

Given a reference scan R and an actual scan S, because of (5.7), the rotation ϕ between the scans can be estimated by correlating the HS^R and HS^S. The author uses a circular correlation as:

$$corr(\phi) = \sum_{\theta \in \Theta} HS^S(\theta) \times HS^R(\theta - \phi), \quad \forall \phi \in \Theta$$

Figure 5.12 shows an example of the computed spectrum and its cross-correlation for two scans.



(a) Original scans

(b) Spectrum and cross-correlation (sinusoid like curve in the background)

Figure 5.12: ϕ estimation by spectra correlation.

The number of hypotheses generated using this method can be chosen arbitrarily, allowing global localization. The different hypotheses are relevant for example in a SLAM (simultaneous localization and mapping) context. Here, however, we are only interested in the pose tracking which is related to the previous estimations, this information can be used to restrict the search domain for ϕ, reducing the computational efforts.

With the ϕ hypotheses, it is possible to estimate the translation in several manners, for example using ICP. In [46] the author proposes its estimate in the Hough domain.

It is important to note that, even though the estimates are taken in the Hough domain, no assumptions on the scene geometry are needed, but the estimations are improved for rectilinear environments.

5.4.2.2: Complexity

As shown in [46] the complexity for finding the rotation estimation is

$$\mathcal{O}\left(R\sigma_\phi + \phi_{max}\sigma_\phi^2\right)$$

where R are the number of points in the scan, σ_ϕ is the required angular discretization (resolution of the solution) and ϕ_{max} is a bound for the search of ϕ. That is, linear in the

number of data points and quadratic on the resolution.

5.5: Concluding remarks

In this Chapter we have presented some relevant aspects of wheeled mobile robots, including some approaches to modeling and, most relevant to this work, the characteristics of two common pose providers. We summarize the relevant results:

In Section 5.1:

- The kinematics model for pose update as in Equation 5.1.

  This model, regardless its simplicity, can be used for sensor fusion, for example in an Extended Kalman filter.

- The model relating wheel speeds to direction speeds for a differential drive robot.

  This model is what is commonly used in pose estimation for odometry and an analysis of the assumptions in which the model is valid is important to understand what kind of model disturbances can affect odometry.

In Section 5.3 we explored odometry with a more comprehensive analysis including:

- Presentation of its principle of operation.

  Giving the reader a general understanding of the instrumentation used in odometry.

- Discussion on the main error sources.

  Very important to help the understanding of the estimates and behavior under different conditions.

- Discussion on error covariance matrix estimation.

  The unbounded error behavior related to the integrative method used in odometry was presented with a simple error modeling. Noticeable is the fact that it increases with the navigation time, with more impact in the direction perpendicular to the movement.

Finally, Section 5.4 explored the use of laser range finder sensors dealing with the localization problem, some remarks:

- Presentation of its principle of operation.

- Description of the Iterative Closes Point (ICP) scan matching algorithm.

  This is one of the most common approaches to scan matching and a version of the algorithm was presented. It has also been pointed out its current main challenges and a discussion on error modeling.

  A remarkable characteristic is its need for an initial transform to perform the matching with a reasonable convergence speed.

- Method for global estimate of the rotation transform between scans.

  We presented a method to estimate scan rotations using the Hough transform that has a bounded computational complexity. For this purpose we presented the basics of the Hough transform and its application to scan matching.

  The estimate of the rotation with such method can be used, for example, as a initial guess for an ICP algorithm.

6 Fault detection and recovery

The past Chapters have built the needed knowledge to design a detection and recovery framework. Part I introduced concepts and presented the observer-based fault detection while Chapter 5 introduced mobile robots and the main challenges arising in the localization task.

The objective of this Chapter is to define a localization fault detection and recovery method and to explore its performance over some case studies.

The platform used for our tests is shown in Figure 6.1. It is a Powerbot robot

Figure 6.1: Wheeled mobile robot platform used for the tests.

from MobileRobots, weighing $120kg$ and is designed for high payload tasks, able to carry more than $100kg$ total. Some important characteristics for our application:

- *Custom programs:* C/C++ programs can be used to control the robot. We here have two simple routines, one that handles the navigation of the robot while the second logs data from its sensors to a file, which is used as input for our algorithms.
- *Client-server communication:* the platform uses a client-server communication from applications to robot drives. It is important to note that this communication

is not deterministic and jitters/delays might occur and therefore the sampling rate will not be constant.

- *Differential drive:* the robot's mobility is achieved through two wheels driven by DC motors and two extra rear caster wheels for balance in an almost holonomic robot.

  The robot uses $25cm$ diameter inflatable tires. This type of tires requires more careful when using odometry, since the basic assumptions of constant wheels radii and one contact point between wheel and floor are weakened. It is then expected an increase of odometry errors caused by model uncertainties.

The robot is equipped with *odometry* and a SICK *laser range finder* which are used to provide two redundant localization outputs.

Our objective is to detect and attenuate faults causing a bias in the odometry pose estimation. Two different faults are prioritized:

- Hard faults, caused by external disturbances such as the robot being pushed, that affects considerably the robot pose, but are not visible in the odometry estimates.
- Slippage faults, that affects the robot positioning but slower and with smaller amplitudes. It can be caused for example by a robot navigating in a dusty terrain or grasping over a wall.

To be able to detect and attenuate these faults, the following Sections present both a change detection framework and a recovery method that include the detection information to attenuate the faults.

6.1: Change detection

Recapitulating from Chapter 2, a change detection takes a set of measured data from a system, subject to noise/disturbances as well as faults and filters it attempting to detect a change introduced by a fault. It is subdivided in different tasks:

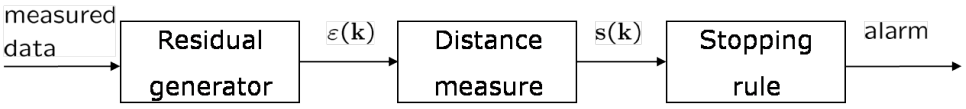


Figure 6.2: Change detection scheme.

- *Residual generator:* a filter that given a set of measured data provides a residual $\varepsilon(k)$, a quantity that should be sensible to faults but not noise/disturbances.
- *Distance measure:* is a metric of the changes occurring in the residual, outputs $s(k)$.
- *Stopping rule:* a decision making filter that generates alarms in case its input has exceeded a certain criteria.

We elucidate each of those subtasks in our framework.

6.1.1: Residual generation

This is probably the most important task in a fault detection scheme and is usually the one that require more design efforts. As shown in Chapter 2, there are different manners to define a residual generator, our focus here is observer-based residual generation. We suppose two localization methods (pose providers) are available, odometry and laser scan matching.

Odometry, p_{odo}, is very commonly used as a pose provider. As explained in the previous Chapter, its pose estimation error grows with the traveled distance in a non-linear manner. It is successfully used however, if it is possible to reset the errors, keeping it bounded under a certain region of operation or for short range tasks.

Scan matching, p_{sm}, is achieved from a laser range finder installed in the robot. The algorithm used mixes the rotation estimation in the Hough domain and Iterative Closest Point, as explained in the previous Chapter.

Since the robot heading ϕ estimate is the hardest, it is first estimated in the Hough domain by looking for local maxima in the spectra correlation. This provides a set of hypotheses for ϕ which are selected by checking the one that provides the smallest ΔT in the ICP algorithm (output of function $\mathbf{getTransform}$), check Algorithm (1). The main advantage of this approach is that we always find a reasonable estimate of ϕ, the accuracy of the estimate is related to the resolution of the transform.

Finally, the translation displacements $(\Delta x \quad \Delta y)$ are estimated with the ICP directly, since we have a good initial guess of ϕ, we use a 0.5° resolution Hough transform, the estimates converge in a couple of iterations of the ICP.

Figure 6.3 shows the pose estimations for both scan matching and odometry. Notice the larger variance from the laser estimates and the difference in the x estimation caused by odometry model errors, easily seem in this example due to a badly calibrated tire. With the pose providers described, we can introduce the different resid-

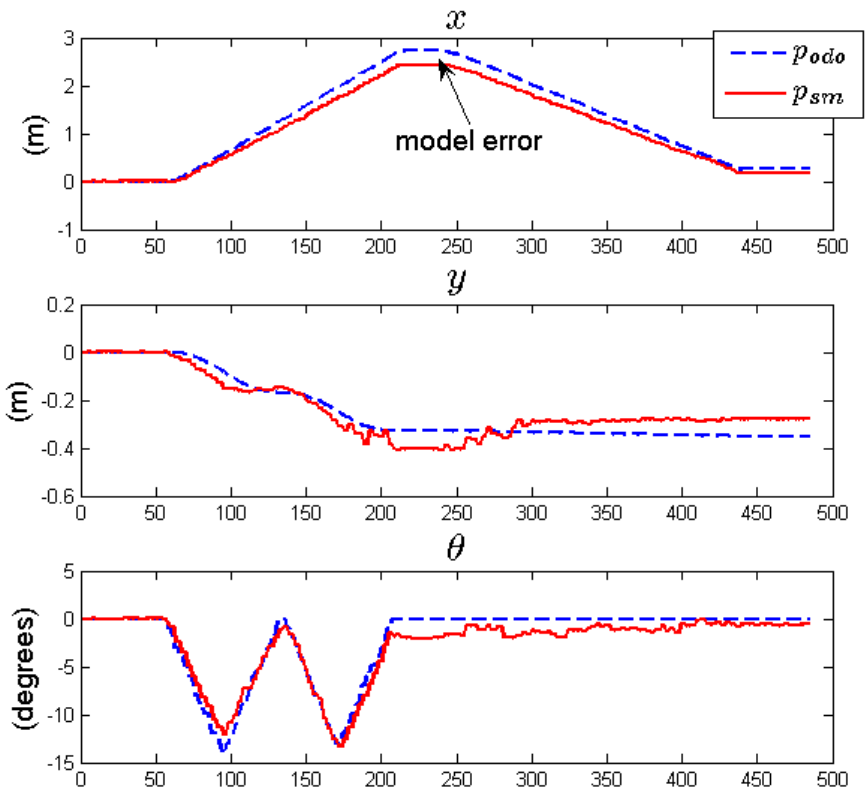


Figure 6.3: Output from odometry and laser scan matching.

uals that we will use for comparison reasons here.

6.1.1.1: ε_0 - difference between poses

The first one is the usual residual taken as the direct difference between the redundant sensors. In our case it is defined as

$$\varepsilon_0(k) = p_{odo}(k) - p_{sm}(k)$$

6.1.1.2: $\breve{\varepsilon}$ - observer with unknown sensor structure

The second residual is based on the tracker filter presented in Section 4.2.3. In this case, we neglect the sensor dynamics and directly apply an observer/Kalman

filter to provide a redundancy. The model we consider for the observer is the relation between the robots directional velocities $[u_f \quad u_w]$ and its poses $[x \quad y \quad \theta]$, this model was already described in the previous Chapter and is recapitulated here

$$\begin{bmatrix} x \\ y \\ \theta \end{bmatrix}_{k+1} = \begin{bmatrix} x \\ y \\ \theta \end{bmatrix}_k + \begin{bmatrix} T\,cos(\theta) & 0 \\ T\,sin(\theta) & 0 \\ 0 & T \end{bmatrix}_k \begin{bmatrix} u_f \\ u_w \end{bmatrix}_k$$

Notice that such model is nonlinear, and therefore, we use an extended Kalman filter (see Section 3.2.2).

In order to be possible to use this filter to provide the redundant output, we need the directional velocities $[u_f \quad u_w]$. For this purpose we use the velocity estimates $\begin{bmatrix} \dot{x} & \dot{y} & \dot{\theta} \end{bmatrix}$ available directly from odometry (it is quite common that the odometry pose provider gives information on the derivative of the pose also) and solve $[u_f \quad u_w]$ in Equation 6.1.

The resulting residual generator filter scheme is as shown in Figure 6.4. Note

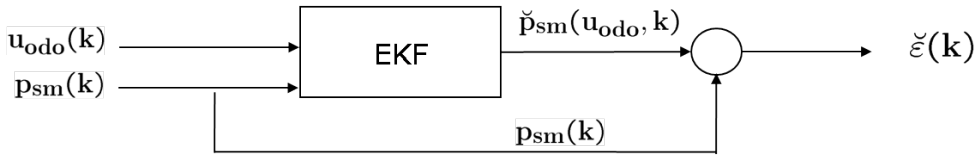


Figure 6.4: Redundant output generated through an EKF.

that we could have estimated $[u_f \quad u_w]$ from p_{sm} (the scan matching pose) however, this would require differentiation of the signals which would yield in a high variance estimate while odometry directly measures the velocity and therefore, no differentiation is needed. The residual is then

$$\breve{\varepsilon}(k) \quad = \quad p_{sm}(k) - \breve{p}_{sm}(u_{odo}, k)$$

where $\breve{p}_{sm}(u_{odo}, k)$ is the output of the tracker observer.

Tuning

Since a Kalman filter, as any stable observer, will adapt to the measurements, a fault will only be visible if the observer/Kalman filter counteractions to the data change (its speed) is slower than the change itself. Moreover, after a fault occurs, it will only be visible in the residual for a certain time until the observer reaches the measurements again. These reasonings are related to what has already been presented in Part I, that a fault effect in the residual is greater as slower the observer is. We cannot totally trust

the model (simulator case) though, since then the residual would also be considerably vulnerable to noise/disturbances. The observer tuning is then a tradeoff between noise/disturbance attenuation and sensitivity to faults.

To tune the EKF we need the measurement and process noise covariance matrices. As depicted in the previous Chapter the advantage of scan matching is that its errors can be kept bounded within a region. Using fault free data, we could depict our scan matching variance as

$$R = \begin{bmatrix} 0.03^2 & 0.05^2 & \left(\frac{0.5 * \pi}{180}\right)^2 \end{bmatrix}$$

that is, a $3cm$ standard deviation for the x estimates, $5cm$ for y and 0.5° in θ. The higher variance for y is due to the fact that estimates in the perpendicular direction of movement are more difficult.

The estimate of the process noise covariance matrix might be difficult, it is related, amongst other factors, to the robot structure, travel speed, etc. However, since we would like our filter not to be so fast (giving more relevance to the model estimates rather than the measurements), we simply consider the process noise as a tenth of our measurement noise

$$Q = \begin{bmatrix} 0.003^2 & 0.005^2 & \left(\frac{0.05 * \pi}{180}\right)^2 \end{bmatrix}$$

With Q and R set this way, the filter is set and we can use the residual $\breve{\varepsilon}(k)$ to perform the detection.

6.1.1.3: $\bar{\varepsilon}$ - augmented observer

As discussed in Chapter 4 Section 4.2, another possible approach to achieve fault detection with integrated/encapsulated sensors is to suppose sensors are integrated with an observer/kalman filter and build a model augmenting the internal sensor states $\bar{x} = \begin{bmatrix} x^T & \hat{x}^T \end{bmatrix}^T$ with the augmented system matrices

$$\bar{A} = \begin{bmatrix} A & 0 \\ LC & (A - LC) \end{bmatrix} \quad \bar{B_u} = \begin{bmatrix} B_u \\ B_u \end{bmatrix} \quad \bar{C} = \begin{bmatrix} 0 & C \end{bmatrix}$$

and use for example an extended Kalman filter with such augmented model to provide the redundancy. The final structure, shown in Figure 6.5, is similar to the tracker presented earlier with the difference that we use the augmented states model in the filter.

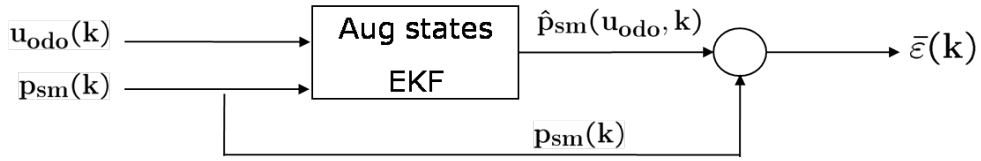


Figure 6.5: Redundant output generated through an augmented observer.

The residual is then

$$\bar{\varepsilon}(k) \quad = \quad p_{sm}(k) - \bar{p}_{sm}(u_{odo}, k)$$

where $\bar{p}_{sm}(u_{odo}, k)$ is the output of the augmented states observer.

Tunning

The EKF tuning in this case is a bit more comprehensive since the covariance matrice Q is now doubled in size and we also need to consider the internal sensor structure, which with our approximation that sensors are integrated with observers, is actually the sensor gain L.

Here, we are simply going to choose an L that yields an stable observer, that is

$$\lambda_{(A-LC)} < 1, \quad \forall\, \lambda$$

and that has faster dynamics then the system

$$\lambda_{(A-LC)} < \lambda_A, \quad \forall\, \lambda$$

notice that since our model is nonlinear, L needs to be solved at each time step with the linearized model of the system. Since a slower sensor and consequently smaller L would yield a residual more sensible to a fault we choose an L that makes the observer speed to be 10% faster than the system

$$\lambda_{(A-LC)} = 0.9 \times \lambda_A, \quad \forall\, \lambda$$

It is expected that the internal sensor states have a greater variance than the systems', the matrix Q is so that the sensor states are 2 times larger standard deviation than the ones available as output

$$Q \quad = \quad \begin{bmatrix} 0.003^2 & 0.005^2 & \left(\frac{0.05 * \pi}{180}\right)^2 & 0.0015^2 & 0.0025^2 & \left(\frac{0.025 * \pi}{180}\right)^2 \end{bmatrix}$$

Since, as previously, we use measurements from the scan matching, R is con-

sidered the same as in the tracker case

$$R = \begin{bmatrix} 0.03^2 & 0.05^2 & \left(\frac{0.5 * \pi}{180}\right)^2 \end{bmatrix}$$

and the filter is now defined and its output can be used to produce the residual $\bar{\varepsilon}(k)$

6.1.2: Distance measure

Several distance measures can be used (see Section 2.3). Here we are interested to changes in the mean and therefore, the direct residual is a good choice, that is

$$s(k) = \varepsilon(k)$$

since we assumed constant covariances matrices to tune the observers, noise/disturbances might influence too much on the residual, leading to false alarms. To improve the robustness of our framework we use a *moving average* filter, defined below.

$$s(k) = (1-\gamma)\ s(k-1) + \gamma\, \varepsilon(k) \tag{6.1}$$

For $\gamma > 0.5$, the result of the average gives more relevance for the last samples (the present estimations are more important). For $\gamma < 0.5$, the result of the average gives more relevance for the past samples (the past estimations are more important). For $\gamma = 0.5$, the moving average becomes the general mean. Of course we are more interested in the present measures so we take $\gamma = 0.8$.

6.1.3: Stopping rule

There are also several stopping rules available, a classical approach is to apply a test in the distance measure using a threshold as an estimate of the standard deviation at each time, such that $|s(k)| < 3\sigma(k)$, this is also known as the 3σ *test*. For more robustness, the variance using the non faulty data can be included as $|s(k)| < 3\sqrt{\sigma(k)^2 + \sigma_o(k)^2}$ where the subscript σ_o denotes the fault free value.

However, for signals with too high variance or for incipient faults, this approach can generate too many false and/or missed alarms. The CUSUM, defined below, is a test used to improve the robustness of the stopping rule.

The CUSUM test statistic, [17], is formulated by the following algorithm: The test

Algorithm 3 CUSUM test

$g(k) = g(k+1) + s(k) - v$
$g(k) = 0, \quad \text{and} \quad \widehat{k} = k \quad \text{if} \;\; g(k) < 0$
$g(k) = 0, \quad \text{and} \quad k_a = k \quad \text{and alarm if} \;\; g(k) > thold > 0$

statistics $g(k)$ sums up its input $s(k)$, with the idea to generate an alarm when the sum exceeds a threshold. The drift variable v is set to compensate for the variation of the parameter caused by noise and errors in the estimation, while the threshold choice is related to the trade off between false alarms and detection time. As presented in [17], the CUSUM can also be used to estimate the time a fault occurred as the last $\widehat{k}$ before an alarm. Note that the test as defined in Algorithm (3) is a one-sided test, if the residual can be negative (as in our case), another CUSUM must be run in parallel.

It is usual to set v as a standard deviation of $s(k)$ in a fault free case, this is the approach also used here. The threshold is then defined as $thold = 3v$.

Our change detection scheme is now set, recapitulating Figure 6.2 in the introduction of this Chapter, we have our detection translated as presented in Figure 6.6.

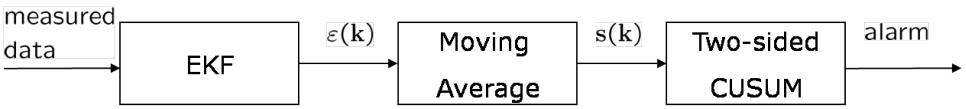


Figure 6.6: Change detection scheme used.

6.1.4: Case studies

In this Section the change detection framework is evaluated within some case studies. The cases are generated with real data. The same detection algorithm is used for all cases, with the parameters and structures as described earlier.

6.1.4.1: Case 1: Normal case

Objective

Check the behavior of the detection method in the case where there is no fault in the system, but with a badly calibrated odometry, showing how it behaves under model disturbances.

Experiment

The robot is moved in the room following a sinusoid trajectory, stops and return. The experiments were realized with the robot's tires calibrated with a lower pressure than the nominal. In this case, the positions calculated for odometry bias considerably from the robots real position causing a deviation between the poses.

Figure 6.7 shows the results of the residuals for the case. From Figure 6.7, it

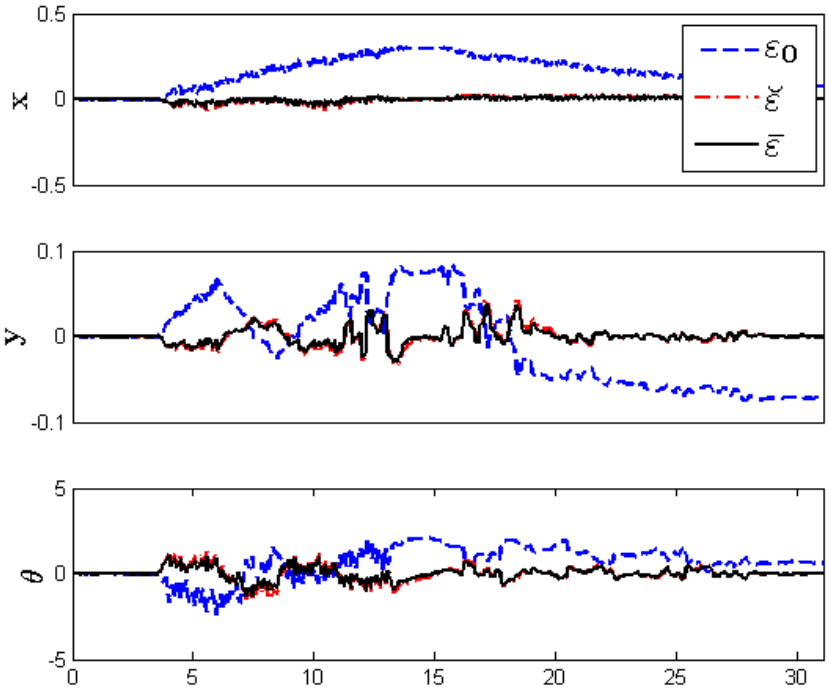


Figure 6.7: Residuals when there is no fault present in the robot. Notice the increase on the residual ε_0 caused by the bias in odometry.

is easy to realize that the residual $\varepsilon_0(k)$ is very sensible to noise/disturbances, especially in the x direction which would be expected to be most affected by this kind of disturbance.

The other two residuals though, could handle the disturbances much better. It can be seem that $\bar{\varepsilon}(k)$ has a slightly smaller variance than $\breve{\varepsilon}(k)$. Figure 6.8 shows the detection result from the CUSUM test for the three different residuals. Notice that while $\breve{\varepsilon}(k)$ and $\bar{\varepsilon}(k)$ remain close to zero, $\varepsilon_0(k)$ generates some false alarms.

Conclusions

The example showed us that $\varepsilon_0(k)$ is too sensible to undesired disturbances. As

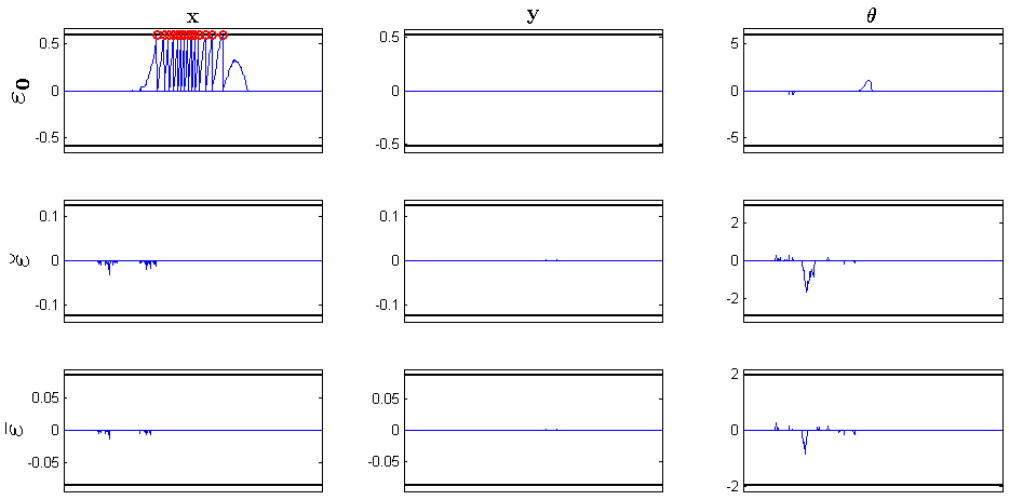


Figure 6.8: Detection result, $g(k)$ and thresholds, for all three residuals when there is no fault (two sided CUSUM test). Notice the alarms, marked as circles, generated for ε_0.

the other two residuals, $\breve{\varepsilon}(k)$ and $\bar{\varepsilon}(k)$, behave similarly, with a slight improve for $\bar{\varepsilon}(k)$.

6.1.4.2: Case 2: Hard fault

Objective

Check the behavior of the detection method in case of a large fault, caused by an external force, such as someone pushing the robot.

Experiment

The robot moves with the same trajectory as in the first case, when no fault was present, and it is pushed away, forcing a rotation and translation of the robot.

Figure 6.9 shows the residuals for $\breve{\varepsilon}(k)$ and $\bar{\varepsilon}(k)$, $\varepsilon_0(k)$ was omitted because, as it was shown in the previous example, it generates too many false alarms, so we will focus on the other two. Figure 6.10 shows the detection result, when both residuals are able to detect the faults correctly.

6.1.4.3: Case 3: Slippage fault

Objective

Check the behavior of the detection method when a slippage fault occurs, a

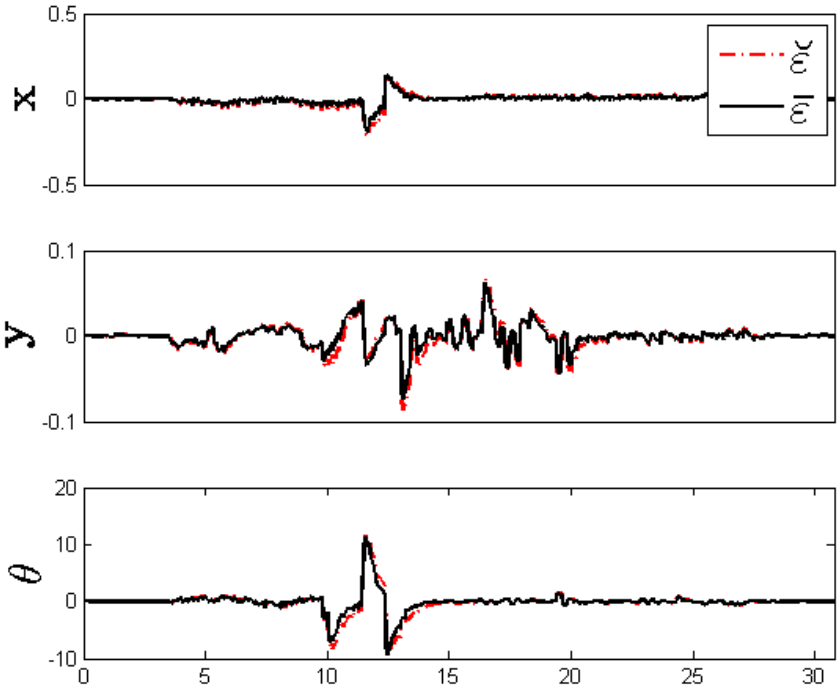


Figure 6.9: Residuals when there robot is hardly pushed away, forcing both translations and rotation. Notice that the residuals behave similarly.

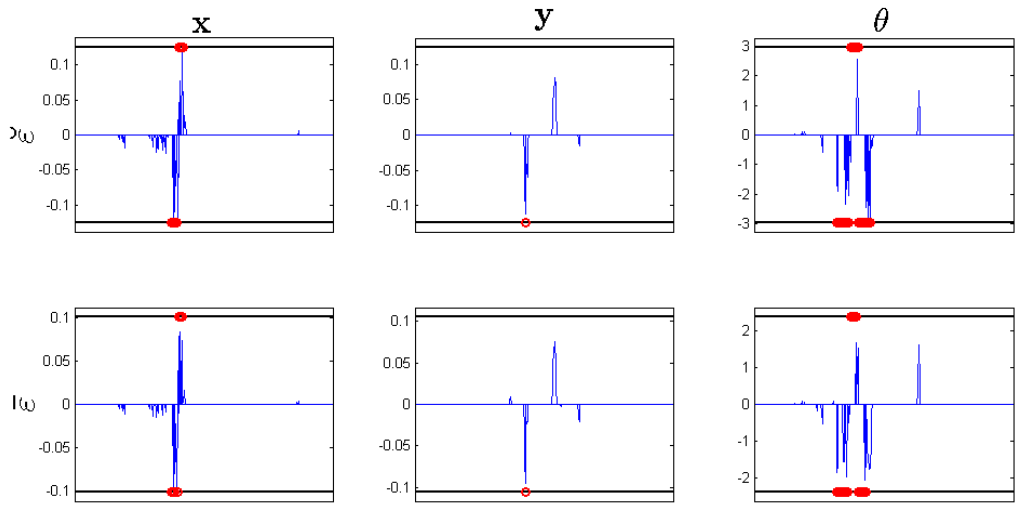


Figure 6.10: Detection result, $g(k)$ and thresholds, when the robot is hardly pushed away, forcing both translations and rotation. The circles marks when alarms were triggered.

smaller and slower position bias than in the previous case, caused for example when a robot is grasping an object or a wheel slippage. **Experiment**

The robot is held when it is moving forward, so that a fault appears in x, while the robot is held its heading is also slightly changed, approximately, $2°$. This case is similar to a wheel slippage, when the robot wheels rotates but no or less displacement occurs.

Figures 6.11 and 6.12 presents the results for the residuals and detection.

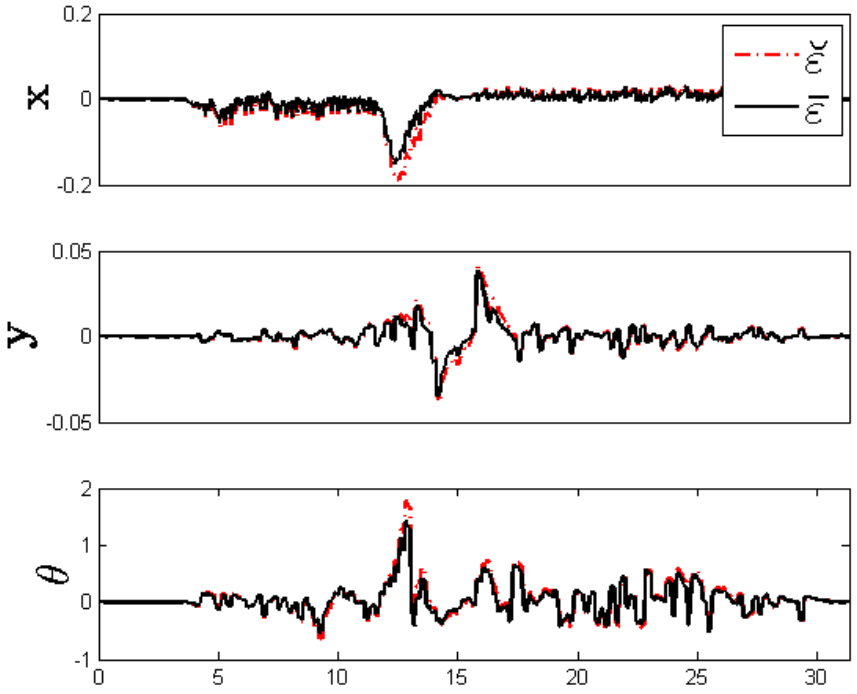


Figure 6.11: Residuals when the robot is held when moving forward, simulating a wheel slippage.

6.1.4.4: Conclusions

As seem from the results, both residuals were able to detect the faults, even the small change that occurred in the robot headings θ. It is also interesting to notice that though y varied around $\pm 3cm$, these changes were not detected, which is due to the fact that its normal variance as seen in Figure 6.7 are already inside this region.

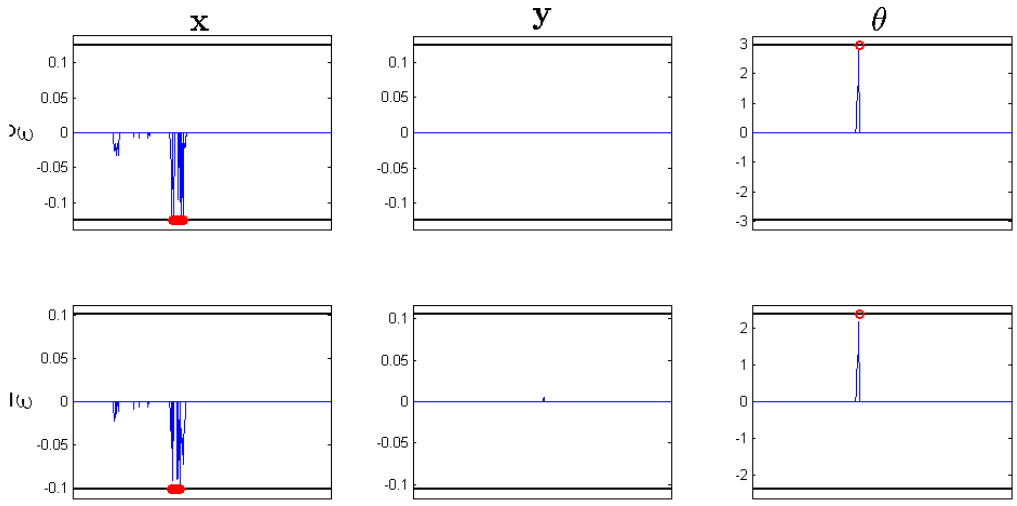


Figure 6.12: Detection result when the robot is held when moving forward, simulating a wheel slippage.

6.2: Fault isolation and recovery

We are interested in decreasing the influence of faults on the robots odometry localization, to do so, we need first to detect the faults (this was the subject of the previous Section) then, we need to estimate the fault time and its size to compensate for it in our pose. The task, as shown in Figure 6.13, can be divided in two subtasks

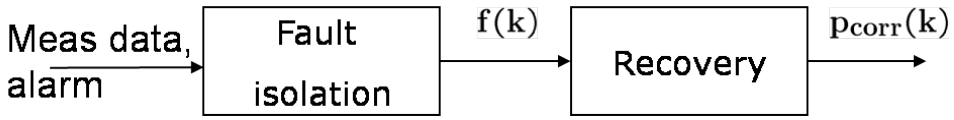


Figure 6.13: General recovery scheme.

- Fault isolation: estimate when, where and the size of the fault.
- Recovery: uses the fault isolation information to properly counteract its effects on the system.

The following Sections present an algorithm to first, estimate the fault time and size on-line (isolation), secondly, these informations are used to compensate the pose estimations given by odometry (recovery).

6.2.1: Fault isolation

As pointed earlier, we are interested in detecting and handle faults that affect our localization system. From our change detection framework as described earlier, it is easy to depict which direction is under the effect of a fault providing us the information of *where* a fault occurred is then, easy.

A useful method to indicate the kind of a fault that occurred is the use of hypothesis test and decision structure as described by Nyberg in [20]. In this approach, several test quantities T_i (any quantity sensible to a fault) are estimated and compared in a hypothesis test to determine which kind of fault occurred.

A hypothesis test can be defined as the decision between the two states possible for a fault (present or not). The hypothesis test here can be seem as a matrix where the rows represent the hypotheses and the columns the test quantities, the elements of such matrix are the result of the test quantity for each hypothesis (each monitored behavioral mode of the system), a 0 value means that the test quantity does not relate to the hypothesis and an X that it can relate. The illustration presents an example where

	T_0	T_1	T_2
F1	X	0	X
F2	X	X	0
NF	0	0	0

T_0 can affect F1 and F2 and T_1 can affect F2. NF stands for the non-fault hypothesis, where none of the test quantities affect it.

In our case such Table will look like

	ε_x	ε_y	ε_θ
F_x	X	0	X
F_y	0	X	X
F_θ	0	0	X
NF	0	0	0

Table 6.1: Hypothesis test for localization faults.

where ε_i is the residual in the direction i and F_i is the fault in the direction i. Notice that a fault affecting the robot headings θ can affect as well both translational hypothesis $F_{x,y}$.

To estimate the time and size of the fault is a bit more comprehensive. We can

notice that the behavior of our residual under a fault will look like Figure 6.14 which is

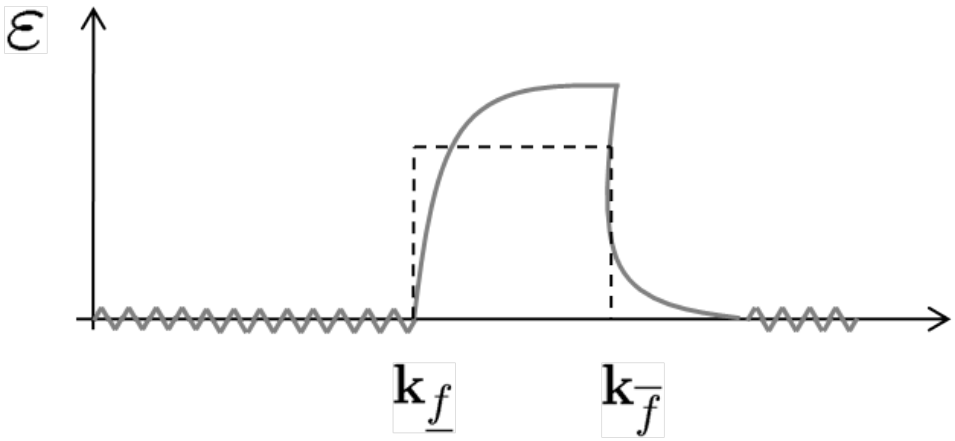


Figure 6.14: Residual under a fault affecting the system from $k_{\underline{f}}$ until $k_{\overline{f}}$.

a step response from fault to residual from time $k_{\underline{f}}$ until $k_{\overline{f}}$. If we can correctly estimate $k_{\underline{f}}$ and $k_{\overline{f}}$, it is possible to look at the pose deviations at these times to estimate the size of the fault.

Estimating the fault initial time

As we have pointed out earlier, the CUSUM test already gives an estimate of the size of the fault. In the CUSUM test, Algorithm (3), the parameter $\widehat{k}$ is the last time in which the residual was set to zero. If our drift parameter v is well set, when no fault is present the test statistic will be zero. After a fault occurred, $g(k)$ will start to grow and will not be reset until an alarm, so $\widehat{k}$ closest to the alarm time is related to the fault time (when the residual started growing).

Estimating the fault end time

Considering there is no overshoot on the response from fault to residual, as illustrated in Figure 6.14, the residual will grow under a fault until its steady-state value, which will be its maximum. We can use this reasoning to estimate $k_{\overline{f}}$.

When under a fault, the change detector will generate alarms, the basic idea is to track the largest value of the residual when consecutive alarms have been generated to update $k_{\overline{f}}$. This is illustrated in Figure 6.15 where the crosses and triangles mark when an alarm has been generated. At each time step under a fault the largest residual value is compared to actual one, whether it is greater or equal to current value stored, we update $k_{\overline{f}}$, these are marked as crosses in the illustration. When the fault halts, the residual starts to decrease but we still generate alarms, these are marked as triangles.

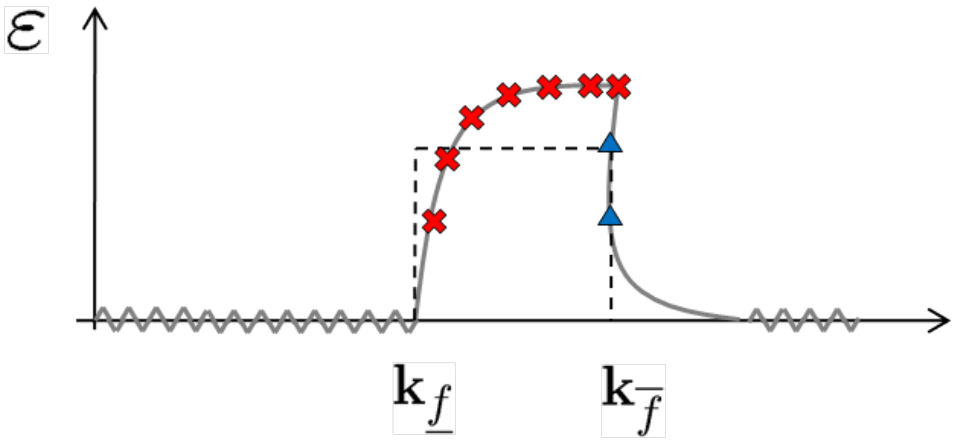


Figure 6.15: Residual under a fault affecting the system from $k_{\underline{f}}$ until $k_{\overline{f}}$. The crosses depict alarms with and increasing-equal residual while triangles depict alarms with a decreasing residual.

Notice that we cannot predict when the fault will end (the actual $k_{\overline{f}}$), we suppose that, when an alarm occurred, $k_{\overline{f}}$ is directly related to the current largest residual, at instant k.

Estimating the fault size

Finally, we can use the information of when a fault started and when it ended to estimate the size of the fault. Only the pose estimated from scan matching will be sensible to the faults since the odometry estimates are based on the integration of the wheels speeds translated to linear displacements.

We compute the size of the fault by checking the deviations of the pose in each direction that generated an alarm from the current estimates of $k_{\underline{f}}$ till $k_{\overline{f}}$. For example, Figure 6.16 presents the x estimations of a robot pose when it is moving forward with constant speed, in a certain moment, the robot is held, introducing a localization fault. Every time an alarm marked as a cross (whose residual is greater than the one when a previous alarm occurred) is generated, we look at the displacement that occurred in the time interval from $k_{\underline{f}}$ to k_a and estimate the size of the fault, which is taken as the difference between the scan matching pose at the time $k_{\underline{f}}$ and k_a.

Algorithm 4 presents the method for x, it should be repeated for the three parameters $[x \quad y \quad \theta]$, also notice that the CUSUM test is a one directional test, and therefore, it should also be repeated for the other direction with $CUSUM(-\varepsilon)$.

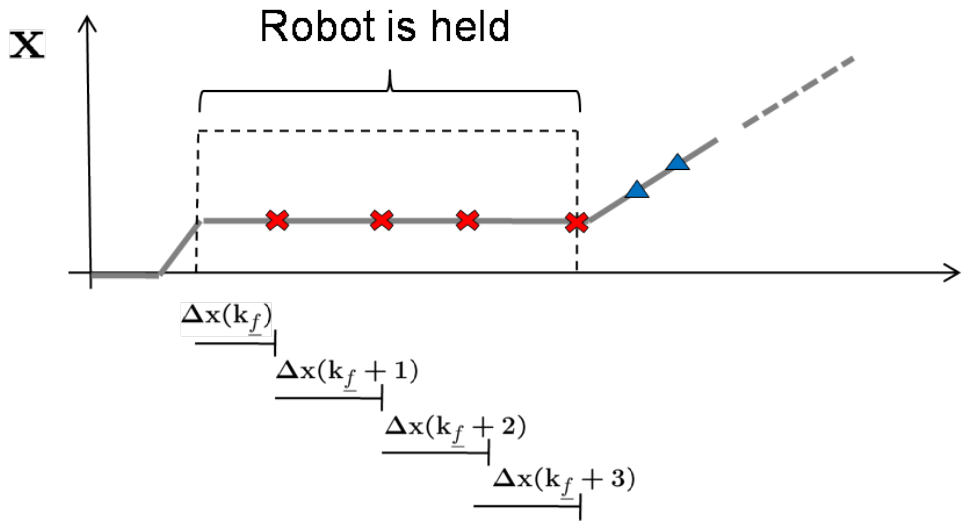


Figure 6.16: Fault size estimation illustration for x. The robot starts moving forward with a constant speed when it is held, introducing a localization fault. The fault size is estimated at each time step according to the estimated $k_{\underline{f}}$ and $k_{\overline{f}}$

Algorithm 4 Estimate fault size.

Let x be the monitored parameter,

$k_a = -1$ {alarm time initialization.}
$k_{\bar{\varepsilon}} = -1$ {time when largest residual occurred initialization.}
while k **do**
  $(\widehat{k}, alarm) = CUSUM(\varepsilon_x|_{1:k})$
  if alarm **then**
    $k_{\bar{\varepsilon}} = \max_k \left(\|\varepsilon_x(k)\|, \|\varepsilon_x(k_{\bar{\varepsilon}}) \times [k_{\bar{\varepsilon}} > 0]\| \right)$ {get time when largest residual occurred.}
    if $k_{\bar{\varepsilon}} == k$ **then**
      $k_{\underline{f}} = \max\left(\widehat{k},\, k_a\right)$ {update fault initial time.}
      $k_a = k$ {update alarm time.}
      $\Delta_x = x(k_a) - x(k_{\underline{f}})$ {estimate fault size.}
    end if
  end if
end while

6.2.2: Recovery

At last, given our detection and isolation framework, we can use the fault size and time of occurrence to define a recovery method. The objective is to use the fault estimate to update the odometry pose, the basic idea used here is to update the pose using an Extended Kalman Filter. Figure 6.17 illustrates the filter inputs and outputs.

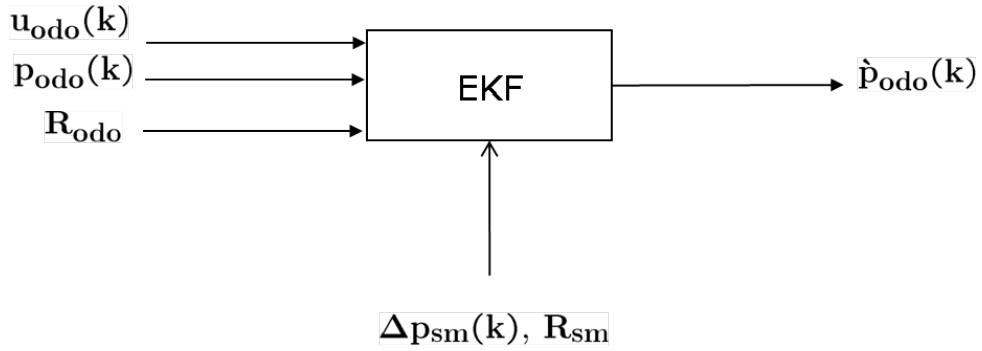


Figure 6.17: Fault recovery scheme.

Every time a fault is detected, we included its effect, $\Delta p(k)$, in the odometry pose to update the filter output, since the measurement variances from odometry and scan matching are different, we change also the measurement covariance matrix to consider its different values for odometry and scan matching.

Algorithm 5 Fault recovery.

Let $alarm|_k$ be a boolean diagonal matrix with its diagonal elements relating to a fault occurrence at each state at time k. And Q the process noise covariance matrix.

$\grave{p}(0) = p_{odo}(0)$ {set filter initial condition.}
while k **do**
  $p^*(k) = p_{odo}(k) + alarm|_k \times \Delta p(k)$ {actual measurement with fault correction.}
  $R = R_{odo} + alarm|_k \times R_{sm}$ {measurement noise covariance matrix with correction.}
  $\grave{p}(k) = EKF(u(k),\ \grave{p}(k-1),\ p^*(k),\ Q,\ R)$
end while

6.2.3: Case studies

Finally, with our recovery method defined, we can show some results achieved with the recovery. We take the same cases as in Section 6.1.4 so that we can follow with the conclusions already depicted about the detection for those cases. For simplicity reasons, we will only show the result of the recovery for the residual $\bar{\varepsilon}(k)$, which uses the augmented states observer.

Case 1 - Normal case

There is not much to discuss when no alarm is generated since the filter will simply track the odometry measurements so we follow and discuss the next cases.

Case 2: Hard faults

As seen in Figure 6.18, the corrected odometry pose $\grave{p}(k)$ could compensate

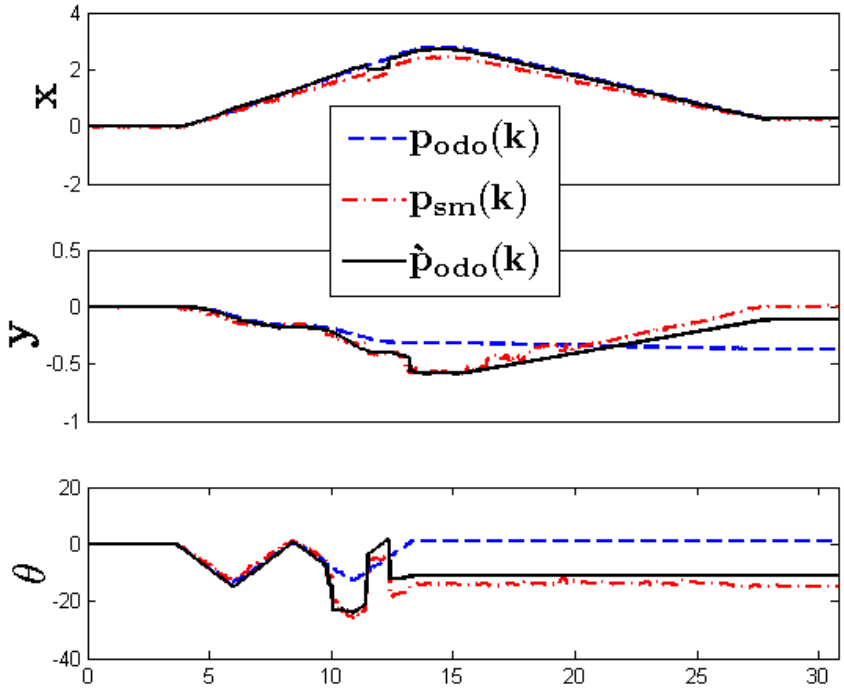


Figure 6.18: Residuals when there robot is hardly pushed away, forcing both translations and rotation. Notice that the residuals behave similarly.

quite well the faults helping to decrease its effects on the pose. The improvements are easily relevant to the robot headings estimate and consequently y, since rotation faults affects it more significantly.

Case 3: Slippage faults

Our last example, shown in Figure 6.19, depicts that, again, the recovery was successful. Even the small 2° fault in the robot headings was detected and handled, improving the whole localization performance

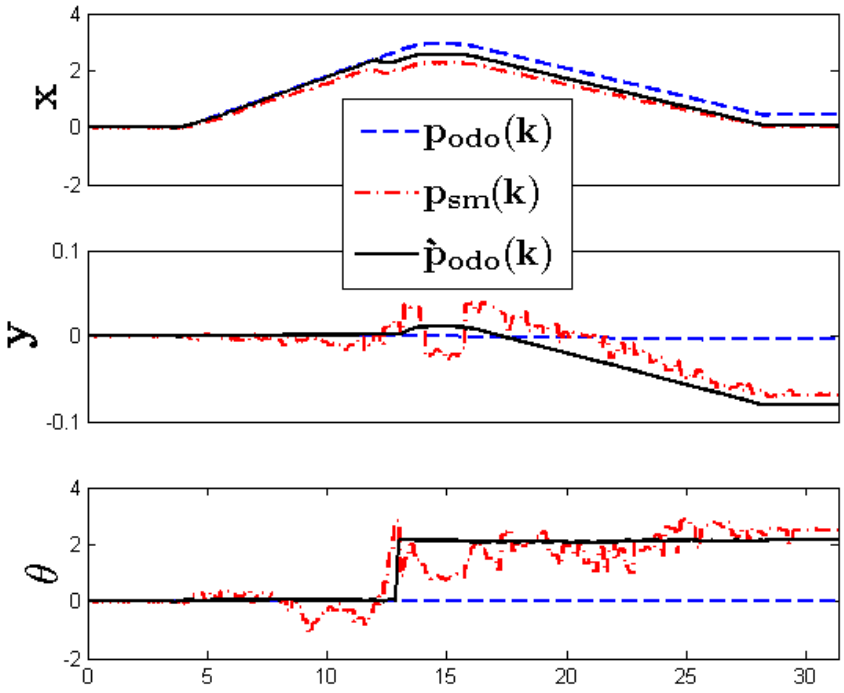


Figure 6.19: Residuals when the robot is held when moving forward, simulating a wheel slippage.

6.3: Concluding remarks

In this Chapter a fault detection, isolation and recovery scheme has been presented and evaluated within some scenarios for localization faults in mobile robots.

With the knowledge support from Part I and the previous Chapter, we first defined an observer-based fault detection scheme comparing the different approaches to the problem with the analyses taken on the residuals $\varepsilon_0(k)$, $\bar{\varepsilon}(k)$ and $\breve{\varepsilon}(k)$. Different case scenarios were taken to sudy these residuals performance, the conclusion of the analyses can be summarized as:

- The residual $\varepsilon_0(k)$, the direct difference of the poses, is too sensible to model disturbances. This was clear during our tests when the robot was moved with a badly calibrated tire.
- The residuals $\bar{\varepsilon}(k)$, using the augmented states observer, and $\breve{\varepsilon}(k)$, using the simplified model, both behaved well for the excited faults, with a slight improve in $\bar{\varepsilon}(k)$.
- Tuning the filter to generate $\breve{\varepsilon}(k)$ is easier than for $\bar{\varepsilon}(k)$. In the tuning procedure

we considered constants Q and R, process and measurement variances, this is of course a simplification and it is expected that using more refined models for these covariances would improve the residual performances even more.

With the definition of a fault detector, we have then presented a fault isolation and recovery method. The detection, isolation and recovery method has been evaluated within some case scenarios to comprove its efficacy in compensating localization faults in the odometric pose.

7 Conclusions

This Chapter summarizes the thesis report and leave comments on future work.

7.1: Summary

The first part of this work dealt mostly with providing theoritical background/results in fault detection, with focus to observer-integrated sensors fault detection, analyzing several structures for observers data only fault detection.

Chapter 2 and 3 discussed standard approaches for fault detection and some background in state observers/estimators; Chapter 4 presented ideas and addressed some basic questions for the problem, including a discussion over fault observability, knowledge on observer structure and residual performance measures through the analysis of the fault sensitivity functions.

Part II, illustrated the problem through a practical example, localization fault detection in mobile robots. Chapter 5 presented briefly the mobile robots field with special attention to modeling and understanding two common localization methods in mobile robots, odometry and laser scan matching. Chapter 6 finally accomplishes the main objective of this works defining a localization fault detection method through integrated sensors. It presented a framework that copes with the tasks of fault detection, isolation and recovery. Such framework is relevant to the community since the localization task is crucial for many mobile robots application, and detecting faults affecting it can decrease its influence.

7.2: Future work

There are yet some open problems such as methods to support the choice of the observer gains, analysis under model uncertainties, etc. We detail some of the

remaining challenges:

- Analysis on the internal sensor gain knowledge L: it would be interesting to analyze how uncertainties in the sensor structure, through L, affects the residual performance in terms of detection.
- Methods to support the choice of the observer gains (tuning): the detection method presented in Chapter 4 and 6 requires tuning which is not yet fully understood and/or there is no direct method to support it that does not require some hard constraints.
- More precise performance evaluation methods of the residuals: though we have given some indications in Chapter 4 of which residual would perform better through the analysis of the fault sensitivity functions they are not closed forms and more study in this area is welcome.
- Use more refined sensor models: the covariances matrices Q and R used in the extended Kalman filter as presented in Chapter 6 are considered constant. The results are supposed to improve if we would consider a more complete sensor model, using, for example a model for the error covariance in odometry and scan matching.
- Extensive evaluation of the method: it would be interesting to check the performance of the method presented in Chapter 6 in many cases as possible, specially in field applications, to testify its validity as a suitable tool for system monitoring.

Bibliography

[1] P. SUNDVALL: *Mobile Robot Fault Detection using Multiple Localization Modules.*, Kungliga Tekniska högskolan, TRITA-EE 2006:044, (2007).

[2] F. GUSTAFSSON: *Statistical signal processing approaches to fault detection.*, Annual Reviews in Control, 31:41–54, (2007).

[3] S. GILLIJNS AND B. DE MOOR.: *Unbiased minimum-variance input and state estimation for linear discrete-time systems.*, Automatica, 43:111–116, (2007).

[4] R. ISERMANN: *Supervision,fault-detection and fault-diagnosis methods - An Introduction*, Control Engineering Practice, **5**5, (1997), 639-652.

[5] R. ISERMANN: *Fault-Diagnosis Systems - An Introduction from Fault Detection to Fault Tolerance*. Springer-Verlag, London, UK, 1st Edition, (2006)

[6] R. ISERMANN AND P. BALLÉ : *Trends in the application of model-based fault detection and diagnosis of technical processes*, Control Engineering Practice, **5**5, (1997).

[7] R. ISERMANN AND B. FREYERMUTH: *Process fault diagnosis based on process model knowledge - Part I: Principles for fault diagnosis with parameter estimation*, ASME J. of Dynamic Systems, Measurement, and Control, **113**, (1991a), 620-626.

[8] R. ISERMANN AND B. FREYERMUTH: *Process fault diagnosis based on process model knowledge - Part II: Case study, Experiments*, ASME J. of Dynamic Systems, Measurement, and Control, **113**, (1991a), 620-626.

[9] R. ISERMANN: *Process Fault Detection Based on Modeling and Estimation Methods-A Survey*, Automatica, **20**4, (1984), 387-404.

[10] R. ISERMANN: *Estimation of physical parameters for dynamic processes with application to an insdustrial robot*, Int J. Control, **55**, (1992), 1287-1298.

[11] R. ISERMANN: *Fault Diagnosis of machines via parameter estimation and knowledge processing - Tutorial paper*, Automatica, **29**4, (1993), 815-835.

[12] P. B. HVASS AND D. TESAR: *Condition Based Maintenance for intelligent electromechanical actuators*, Robotics Research Group of the University of Austin at Texas, (2004).

[13] F. CACCAVALE: *Experiments of Oberver-based Fault Detection for an Industrial Robot*, Proceedings of the 1998 IEEE International Conference on Control Applications, (1998), 480-4 vol.1.

[14] F. CACCAVALE AND ID. WALKER: *Oberver-based Fault Detection for Robot Manipulators*, Proceedings. 1997 IEEE International Conference on Robotics and Automation, (1997), 2881-7 vol.4.

[15] H. SCHNEIDER AND P. M. FRANK: *Oberver-based supervision and fault detection in robots using nonlinear and fuzzy logic residual evaluation*, IEEE Transactions on Control Systems Technology, **4**3, (1996), 274-82.

[16] M. SAIF: *Robust discrete time observer with application to fault diagnosis*, IEEE Proceedings Control Theory and Applications, **145**3, (1998), 353-7.

[17] F. GUSTAFSSON: *Adaptative filtering and change detection*, John Wiley & Sons Ltd., (2000).

[18] M. ÖSTRING: *Identification, Diagnosis, and Control of a Flexible Robot Arm*, Licentiate thesis, Linköpings universitet, Linköping, Sweden. Linköping Studies in Science and Technology, (2002). Thesis No. 948.

[19] L. LJUNG AND T. SÖDERSTRÖM: *Theory and practice of recursive identification*, The MIT Press, (1983).

[20] M. NYBERG: *Model Based Fault Diagnosis: Methods, Theory, and Automotive Engine Applications*, PhD thesis, Linköpings universitet, Linköping, Sweden. Linköping Studies in Science and Technology, (1999). Dissertation No. 591.

[21] P. C. YOUNG: *Parameter estimation for continuous-time models - a survey*, Automatica, **17**23, (1981).

[22] V. F. FILARETOV, M. K. VUKOBRATOVIC AND A. N. ZHIRABOK: *Parity relation approach to fault diagnosis in manipulation robots*, Mechatronics, **13**2, (2003), 141-52.

[23] Q. WANG, Q. BI AND B. ZOU: *Parameter identification of continuous-time mechanical systems without sensing accelerations*, Computers in Industry, **82**, (1996), 207-217.

[24] E. WERNHOLT: *On Multivariable and Nonlinear Identification of Industrial Robots*, Licentiate thesis, Linköpings universitet, Linköping, Sweden. Linköping Studies in Science and Technology, (2004), Thesis No. 1131.

[25] L.-W. TSAI: *Robot Analysis - The Mechanics of Serial and Parallel Manipulators*, John Wiley & Sons Ltd., (1999).

[26] L. LJUNG: *System identification - theory for the user*. Prentice Hall, Upper Saddle River, New Jersey, USA, 2nd Edition, (1999)

[27] L.H CHIANG, E.L. RUSSELL AND R.D. BRAATZ: *Fault detection and diagnosis in industrial systems*. Springer-Verlag, London, UK, 2nd Edition, (2001)

[28] A. C. BITTENCOURT: *Friction change detection in industrial robots.*, Master Thesis Report XR-EE-RT 026/2007, Department of Automatic Control, The Royal Institute of Technology, (2007).

[29] R. E. KALMAN: *A new approach to linear filtering and prediction problems*, Journal of Basic Engineering 82 (1): 35–45, (1960)

[30] J. GERTLER: *Fault detection and diagnosis in engineering systems*, CRC Press, (1998)

[31] R. ISERMANN: *Model-based fault-detection and diagnosis - status and applications*, Annual Reviews in Control, Volume 29, Issue 1, Pages 71-85, (2005).

[32] M. LUO, D. WANG, M. PHAM, C.B. LOW, J.B. ZHANG, D.H. ZHANG AND Y.Z. ZHAO: *Model-based fault diagnosis/prognosis for wheeled mobile robots: A review*, Industrial Electronics Society, IECON 2005, 31st Annual Conference of IEEE, (2005).

[33] R. SIEGWART AND I. R. NOURBAKHSH: *Introduction to autonomous mobile robots*, The MIT Press, (2004).

[34] H. B. PACEJKA : *Tire and vehicle dynamics*, Elsevier Science & Technology Books, 2nd Edition, (2006).

[35] J. DIXON AND O. HENLICH: *Mobile robotics navigation*, http://www.doc.ic.ac.uk/~nd/surprise_97/journal/vol4/jmd/.

[36] G. ANTONELLI, S. CHIAVERINI AND G. FUSCO: *A Calibration Method for Odometry of Mobile Robots Based on the Least-Squares Technique: Theory and Experimental Validation*, IEEE Transactions on Robotics, 21(5), 994–1004, (2005).

[37] A. MARTINELLI: *The odometry error of a mobile robot with a synchronous drivesystem*, IEEE Transactions on Robotics, 18(3), 399–405, (2002).

[38] L. KLEEMAN: *Odometry error covariance estimation for two wheel robot vehicles*, Monash University, Department of Electrical and Computer Systems Engineering, Technical Report MECSE-95-1, (1995).

[39] S. RUSINKIEWICZ AND M. LEVOY: *Efficient variants of the ICP algorithm*, Proceedings of the third international conference on 3-D digital imaging and modeling, (2001).

[40] P. BESL AND N. MCKAY: *A method for Registration of 3-D shapes*, PIEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI), 14(2):239-256, (1992).

[41] E. LUNDSTRÖM: *An analysis of scan matching methods*, Master Thesis Report, The Royal Institute of Technology, to appear.

[42] F. LU: *Shape registration using optimization for mobile robot navigation*, PhD Thesis, University of Toronto, (1995).

[43] O. BENGTSSON AND A.-J. BAERVELDT: *Localization in changing environments - estimation of a covariance matrix for the IDC algorithm*, Proceedings of the 2001 International Conference on Intelligent Robots and Systems, (2001).

[44] O. BENGTSSON AND A.-J. BAERVELDT: *Robot localization based on scan-matching – estimating the covariance matrix for the IDC algorithm*, Robotics and Autonomous Systems 44(1):29-40, (2003).

[45] K. KONOLIGE AND K. CHOU: *Markov localization using correlation*, Proceedings of the Sixteenth International Joint Conference on Artificial Intelligence 1154–1159, (1999).

[46] A. CENSI, L. IOCCHI, G. GRISETTI: *Scan matching in the Hough domain*, Proceedings of the IEEE International Conference on Robotics and Automation,2739–2744, (2005).

[47] P.V.C. HOUGH: *Method and means for recognizing complex patterns*, U.S. Patent 3069654, (1962).

[48] R.O. DUDA AND P.E. HART: *Use of the Hough transform to detect lines and curves in pictures*, Communs Ass. comput. Mach. 15, pp. 11-15., (1975).

[49] T. KAILATH: *Linear systems*, Prentice-Hall Prentice-Hall Information and System Science Series, (1980).

[50] J. MESEGUER, V. PUIG, T. ESCOBET AND R. SARRATE: *Observer gain effect in linear interval observer-based fault detection*, 46th IEEE Conference on Decision and Control, 995–1002, (2007)

[51] J. CHEN AND R.J. PATTON: *Robust Model-Based Fault Diagnosis for Dynamic Systems*, Kluwer Academic Publishers, (1999).

[52] D. TÖRQUVIST.: *Statistical Fault Detection with Applications to IMU Disturbances.*, *Linköping Studies in Science and Technology*, Thesis No. 1258, (2006).

[53] R. A. HORN AND C: R. JOHNSON: *Matrix Analysis*. Cambridge University Press, (1985)

[54] G. F. FRANKLIN, J. D. POWELL AND A. EMAMI-NAEINI: *Feedback Control of Dynamic Systems*, Prentice Hall, Fifth edition.

[55] T. A. CHMIELEWSKI JR AND P. R. KALATA: *On the Identification of Stochastic Biases in Linear Time Invariant Systems*, In Proceedings of the American Control Conference, pages 4067-4071, Seatle, Washington, (1995).

[56] N. LIU: *Optimal robust fault detection*, PhD thesis,Louisiana State University, (2008).

[57] M. ZHONG, S. X. DING, B. TANG, P. ZHANG AND T. JEINSCH: *A LMI application to robust fault detection filter design for discrete-time systems with model uncertainty*, Proceedings of the 40th Conference on Desision and Control, 3613-3618, Orlando, USA, December (2001)

[58] H. B. WANG, L. LAM, S. X. DING AND M. Y. ZHONG: *Iterative linear matrix inequality algorithms for fault detection with unknown inputs*, Journal of Systems and Control Engineering, Vol. 219, No. 2, 161-172, (2005)

[59] F. TAO AND Q. ZHAO: *Fault detection observer design with unknown inputs*, Proceeding of the 2005 IEEE Conference on Control Applications, 1275-1280, Toronto, Canada, August (2005).

[60] S. SKOGESTAD AND I. POSTLETHWAITE: *Multivariable feedback control - analysis and design*, John Wiley & Sons, (1996).

ANNEX A – Paper I

Observers Data Only Fault Detection ⋆

B. Wahlberg * **A. C. Bittencourt** **

* *Automatic Control, KTH, SE 100 44 Stockholm, Sweden*
bo.wahlberg@ee.kth.se
** *Automatic Control, KTH, SE 100 44 Stockholm, Sweden*
andrecb@ee.kth.se

Abstract: Most fault detection algorithms are based on residuals, i.e. the difference between a measured signal and the corresponding model based prediction. However, in many more advanced sensors the raw measurements are internally processed before refined information is provided to the user. The contribution of this paper is to study the problem of fault detection when only the state estimate from an observer/Kalman filter is available and not the direct measured quantities. The idea is to look at an extended state space model where the true states and the observer states are combined. This extended model is then used to generate residuals viewing the observer outputs as measurements. Results for fault observability of such extended models are given. The approach is rather straightforward in case the internal structure of the observer is exactly known. For the Kalman filter this corresponds to knowing the observer gain. If this is not the case certain model approximations can be done to generate a simplified model to be used for standard fault detection. The corresponding methods are evaluated on a DC motor example. The next step is a real data robotics demonstrator.

1. INTRODUCTION

Sensors and observers/estimators are often closely integrated in intelligent sensor systems. This is common in distributed sensor processing applications. It may be very difficult or even impossible to access the raw sensor data since the sensor and state estimator/observer often are integrated and encapsulated. An important application of sensor based systems is model based fault detection, where the sensor information is used to detect abnormal behavior. The typical approach is to study the size of certain residuals, that should be small in case of no fault, and large in case of faults. Most of these methods rely on the direct sensor measurements. The problem when only state estimates are available is less studied. In Sundvall [2006a] and Sundvall [2006b] the problem of fault detection for such a system in mobile robotics is discussed from mainly an experimental point of view. The objective of this paper is to investigate the theoretical foundation of observer data only fault detection, where it is not possible to directly access the raw measured data.

Study the system state space description

$$x(t+1) = Ax(t) + B_u u(t) + B_v v(t) + B_f f(t).$$

Here $x(t)$ denotes the state vector, $u(t)$ is a known input signal, $v(t)$ is process disturbances and $f(t)$ is the unknown fault input. It is common to assume that $f(t)$ is either zero (no fault) or proportional to the the i:th unit vector $f(t) = f_i e_i$ in case of fault number i. Hence B_f is a matrix that determines how different faults affect the state. This covers, for example, faults in actuators.

⋆ This work was partially supported by the Swedish Research Council and the Linnaeus Center ACCESS at KTH.

The behavior of the system can be observed from different sensors j. To simplify the analysis we assume that sensors are integrated with standard observers/Kalman filters.

$$\begin{aligned} y_j(t) &= C_j x(t) + e_j(t) \\ \hat{x}_j(t+1) &= A\hat{x}_j(t) + B_u u(t) + K_j \left(y_j(t) - C_j \hat{x}_j(t)\right) \end{aligned}$$

The input to observer j is the measured output signal $y_j(t)$ and the input $u(t)$. The term $e_j(t)$ represents measurement noise. The output from the observer is $\hat{x}_j(t)$, i.e., an estimate of the state. If for example the Kalman filter is used this could come with a corresponding error covariance matrix

$$\mathrm{Cov}(\hat{x}_j(t) - x(t)) = P_j$$

Problem: We will study the problem when it is only possible to obtain $\hat{x}_j(t)$, and **not** the raw data $y_j(t)$. This seems to be a severe restriction, but from a practical point of view the measurement process could be integrated in the sensor system. One common example is standard GPS, where the measurement is based on satellite tracking and triangularization based techniques. In many applications the state estimate is obtained by more sophisticated methods then a simple linear observer. We will however use this structure for analysis and design purposes, so that the problem can be approached through well studied/standard FDI techniques.

Sofar we have not taken the fault contribution $f(t)$ into account. One possibility is to also estimate $f(t)$ by for example extending the state vector to $\bar{x}(t) = [x(t)\ f(t)]^T$ and apply the Kalman filter or another observer method to estimate the extended state vector $\bar{x}(t)$. Recently, there has been quite a lot of progress in the area of input estimation using Kalman filtering, see Gillijns et al. [2007].

2. RESIDUAL BASED FAULT DETECTION

There are in principle two paradigms for residual based fault detection. We will start with the standard problem formulation, with a direct measurable output.

$$\begin{aligned} x(t+1) &= Ax(t) + B_u u(t) + B_v v(t) + B_f f(t) \\ y(t) &= Cx(t) + e(t) + D_f f(t) \end{aligned} \quad (1)$$

Here we also have the possibility to model and detect sensor faults via D_f. The dimension of $x(t)$ is n_x and the dimension of $y(t)$ is n_y.

The so-called *parity space* approach, recently reviewed in Gustafsson [2007], is based on a sliding window formulation of the state space equations

$$Y(t) = \mathcal{O}x(t-L+1) + H_u U(t) + H_v V(t) + H_f F(t) + E(t)$$

where $Y(t) = \left[y^T(t-L+1)\dots y^T(t)\right]^T$ and similar for the other signal vectors. The matrices are given by

$$\mathcal{O} = \begin{pmatrix} C \\ CA \\ \vdots \\ CA^{L-1} \end{pmatrix}, \quad H_f = \begin{pmatrix} D_f & 0 & \dots & 0 \\ CB_f & D_f & \dots & 0 \\ \vdots & & \ddots & \vdots \\ CA^{L-2}B_f & \dots & CB_f & D_f \end{pmatrix}$$

and H_v and H_u are constructed in the same way as H_f, i.e. from the corresponding impulse response coefficients. The residual is then defined by

$$R(t) = W^T \left(Y(t) - H_u U(t)\right)$$

where W is an $L \times n_r$ projection matrix such that $W^T\mathcal{O} = 0$. The dimension $n_r = Ln_y - n_x$, where n_y is the dimension of the output vector $y(t)$ and n_x is the state dimension. This still leaves freedom in the choice of W, which can be used to obtain more structured residuals. In Gustafsson [2007] gives detailed insights of the design and analysis of parity space based methods in case of stochastic disturbances and noise.

An alternative approach is to use an *observer* or a Kalman filter to estimate $x(t)$ and then study the size of the residuals

$$R(t) = \left(Y(t) - \mathcal{O}\hat{x}(t-L+1) - H_u U(t)\right)$$

The choice $L = 1$ just gives the standard innovation process $r(t) = y(t) - C\hat{x}(t)$ used in the observer and in the Kalman filter to update the state estimate.

It is most important that the effects of the faults are visible in the residual vector. A fault is detectable if the transfer function from fault to residual is non-zero, this condition holds even for faults that disappear in the residual after some transient and a stronger condition is that this transfer function is non-zero also in steady-state. Another way to check if certain faults are detectable is to calculate if the extended state space model $\bar{x}(t) = [x(t)\ f(t)]^T$ is observable in classical state space sense. It is also closely related to input estimation, for which conditions are given in Gillijns et al. [2007].

It is also desirable different faults to be distinguishable between each other through the residual. A classical approach is to define structured residual sets that are sensitive to one fault and whilst remaining unsensitive to others. An alternative is to design a directional residual vector; with each fault corresponding to a direction, isolation is achieved by deciding which direction the generated residual is closest to. The diagnosticability matrix as presented in Gustafsson [2007], is an off-line method to evaluate the isolation in a directional residual set, giving the probabilities of generating alarms for fault i given fault j occurred and also in case there is no fault.

3. RESIDUAL BASED FAULT DETECTION USING OBSERVER DATA ONLY

Residual based techniques are all based on comparing a predicted output $\hat{y}_j(t)$, based on a model, with the *observed* output $y_j(t)$ from a sensor. In case of a systematic difference we will alarm. If only the observer states $\hat{x}_j(t)$ are available the first two ideas for fault detection ideas would be:

- Try to reconstruct $y_j(t)$ using a model of the observer, e.g.
  $$K_j y_j(t) = \hat{x}_j(t+1) - (A - K_j C_j)\hat{x}_j(t) - B_u u(t)$$
  Here we need a very accurate model and the internal structure of the observer, e.g. the gain K_j, otherwise the estimations will be easily biased. In many practical cases this would be difficult. Notice also that if K_j is not full rank, it allows for multiple solutions.
- Assume that there are at least two observers providing $\hat{x}_1(t)$ and $\hat{x}_2(t)$. Define the residual vector
  $$\varepsilon(t) = \hat{x}_1(t) - \hat{x}_2(t)$$
  which should be sensible to fault that affects the two observers in different ways, e.g. sensor faults. This approach does not, however, make direct use of the model of the system and requires at least one redundant sensor.

We will start by analyzing the case with only one observer.

Idea: View $\hat{x}(t)$ as the output from the extended system

$$\begin{aligned} x(t+1) &= Ax(t) + B_u u(t) + B_f f(t) \\ \hat{x}(t+1) &= A\hat{x}(t) + B_u u(t) \\ &\quad + K\left(Cx(t) + D_f f(t) - C\hat{x}(t)\right) \\ \hat{y}(t) &= C^*\hat{x}(t) \end{aligned} \quad (2)$$

where C^* depicts which estimates are available. By using the extended state $\bar{x}(t) = [x^T(t)\ \hat{x}^T(t)]^T$ and the corresponding state space matrices we can interpret this a standard fault detection problem, which could be approached by a parity space method or a Kalman filter based method. The problem when we have m different observers can be approached by augmenting the state space with all observer states $\hat{x}_j(t)$, $j = 1, \dots m$.

There are some basic questions that need to be addressed

- Are the faults detectable using this model?
- What to do if the observer gain K is unknown?
- How to compare/validate the performance of different methods?

3.1 Fault Observability

As described in Kalata et al. [1995], stochastic biases in linear time invariant systems can be identified by

augmenting the system state with a bias and implement a Kalman filter. The author utilizes this technique to identify biases in noisy measurements. In Chapter 3 in Tornqvist [2006] these results are extended to check the observability of additive faults with the constrain that $f(t+1) = f(t)$. An important characteristic explored by both authors is the observability issue.

Considering a system as in Equation (2), where only estimated states $\hat{x}(t)$ are available but not $x(t)$, we can augment the faults in the states as $\bar{x}(t) = [x(t) \quad \hat{x}(t) \quad f(t)]$ and analyze the observability with the pair

$$\bar{C} = [\, 0 \;\; C^* \;\; 0 \,], \quad \bar{A} = \begin{bmatrix} A & 0 & B_f \\ KC & (A - KC) & KD_f \\ 0 & 0 & I \end{bmatrix} \tag{3}$$

Observability conditions: With a similar approach as in Tornqvist [2006] Appendix A shows that if the original system is observable (pair (A, C) observable) and if

- All state estimates are directly available at the output (i.e. $C^* = I$).
- K is full column rank, such that

$$Kz = 0 \quad \Rightarrow \quad z = 0$$

then the extended system will be observable under the same conditions for sensor and process faults as when the actual output $y(t)$ is available. In other words, we will have the same information as when $y(t)$ is directly accessible. The conditions can be summarized as:

- For measurement faults only, if the system has no integrator dynamics (modes with eigenvalue equals to 1), the faults will be observable as long as D_f is full rank. If there is an integrator, then the faults are not observable if D_f is full rank and the states through which the fault propagates should be orthogonal to the measured integrating part of the system.
- For process faults only, the faults should be orthogonal to the contribution of the non-measured part of the system.

3.2 Unknown Observer Structure

If the observer structure is not given, for example if K is unknown, we cannot directly use the extended state space model for fault detection. To overcome this problem, let us make two approximations. The first one is to approximate

$$K\left(Cx_f(t) + D_f f(t)\right) \approx \bar{F}_f \bar{f}(t)$$

Here

$$x_f(t+1) = Ax_f(t) + B_f f(t) \tag{4}$$

and $x_f(t)$ is the fault contribution in $x(t)$. Since $f(t)$ is zero or a constant vector it is most important that this approximation holds stationary i.e. after transients.

The second approximation is

$$K\left(y(t) - C\hat{x}(t)\right) \approx \bar{v}(t)$$

where $\bar{v}(t)$ is a white noise process with a certain covariance matrix. This makes sense from a Kalman filter point of view where the innovations can be viewed as process noise, $\bar{v}(t) = K\epsilon(t)$ where $\epsilon(t) = y(t) - C\hat{x}(t)$ is a white innovation process.

This leads to the simplified model

$$\begin{aligned} \hat{x}(t+1) &= A\hat{x}(t) + B_u u(t) + \bar{F}_f \bar{f}(t) + \bar{v}(t) \\ \bar{y}(t) &= \hat{x}(t) + \bar{e}(t) \end{aligned} \tag{5}$$

With such structure, actuator and sensor faults are mixed and the fault isolation step could be more difficult in this setting.

The artificial measurement noise $\bar{e}(t)$ can be used to cope with unmodeled characteristics of the system. For example, for sensors over a network or with a weak real-time performance, one can use $\bar{e}(t)$ to include jitter, missed samples, delays, etc. or to cope with sensor/system unknown dynamics.

After defining $\bar{e}(t)$, it can be used to tune a Kalman filter observer for the system as in Equation (5) and a standard parity space method or Kalman filter based method can be used to design a fault detection algorithm. It is easy to extend this approach to several observers by combining the observer states as

$$\begin{bmatrix} \hat{x}_1(t+1) \\ \hat{x}_2(t+1) \\ \vdots \\ \hat{x}_j(t+1) \end{bmatrix} = \mathbf{A} \begin{bmatrix} \hat{x}_1(t) \\ \hat{x}_2(t) \\ \vdots \\ \hat{x}_j(t) \end{bmatrix} + \mathbf{B_u} u(t) + \bar{\mathbf{F}}_\mathbf{f} \bar{f}(t) + \bar{v}(t)$$

where $\mathbf{A}$, $\bar{\mathbf{F}}_\mathbf{f}$ and $\mathbf{B_u}$ are diagonal matrices relating observers and fault states and actuator dynamics, and $u(t)$,$\bar{f}(t)$ and $v(t)$ are the extended control, fault and noise inputs.

3.3 Performance evaluation methods

We are interested in analyzing the quality of the residuals generated by the different methods rather than for a full detection scheme. Different factors influence a residual success in terms of detection, including noise/model disturbances decoupling and sensitivity to faults. These qualities are often in contradiction and can be seem as an optimization problem. Some recent results for optimal residual generation can be found in Liu [2008], where the author shows closed-form solutions for some sets of the problem, the solutions however, require that the faults are directly visible at the output (full column rank D_f), which will never be the case for sensors integrated with observer/Kalman filters since the faults travel through the observer dynamics before they appear in the output.

For the analysis of our proposed methods, we will use a definition of a good residual as one that resembles to white gaussian noise with no peaks or abrupt changes in a fault-free case (possibly decreasing the false-alarm rates) and that bias under a fault as great as possible (possibly increasing the detection rate). We depict two empirical quantities to relate these qualities.

The kurtosis statistic is a measure of the peakedness of a signal, we can use it to analyze the resemblance of the residuals *over a fault-free scenario* (NF) to a gaussian distribution. It is defined as

$$\kappa = \frac{E\left[\varepsilon(t)|_{NF} - \mu_{\varepsilon(t)|_{NF}}\right]^4}{\sigma^4_{\varepsilon(t)|_{NF}}} - 3$$

where $\mu_{\varepsilon(t)|_{NF}}$ and $\sigma_{\varepsilon(t)|_{NF}}$ are the mean and standard deviation of the residual under no-fault while $E[z]$ is

the expected value of z. For a gaussian residual the test approximates to zero, and it is usually used to detect abrupt variations over a gaussian signal. In Hadjileontiadis et al. [2005], for instance, it is used for crack detection over a beam with vibration analysis. Basically, a κ close to 0 will relate to a white gaussian distribution while higher κ means more of the variance is due to big sporadic deviations or biases.

The fault-to-noise ratio as defined in Gustafsson [2001] is a measure of a fault sensitivity relative to noise and is defined as a ratio between the expected value of a fault influence in the system output and the noise variance, it is, in fact, a similar concept as the signal-to-noise-ratio (SNR) but applied to a fault. For a known FNR, we can define the measure

$$\delta = \left| \frac{[\varepsilon(t)|_F]/\sigma_{\varepsilon(t)|_F}}{FNR} \right|$$

where $E[\varepsilon(t)|_F]$ and $\sigma_{\varepsilon(t)|_F}$ are the expected value and standard deviation of the residual *under a fault hypothesis* (F). In this manner, δ will vary from 0 to 1 (best possible residual, which will have the same quality as the direct fault influence to the output, only possible if we use a perfect simulator of the system to generate the redundancy).

4. ILLUSTRATIVE EXAMPLE

In order to explore the different configurations presented, we consider a simple linear DC motor, as shown in Figure 1. Non-linearities such as flexibilities and friction are simplified. The applied voltage in the motor terminals is the controlled input to the system V_{app} while angular speed is taken as output. The states are current and angular

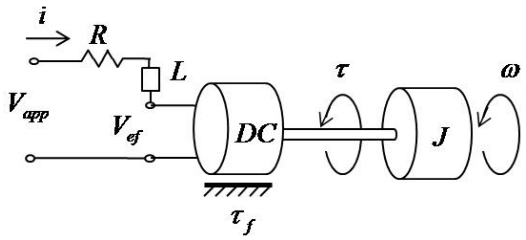


Fig. 1. DC motor model.

speed $x(t) = [i(t) \quad \omega(t)]^T$ and the governing matrices

$$A = \begin{bmatrix} \frac{-R}{L} & -\frac{k_b}{L} \\ \frac{k_m}{J} & \frac{-k_f}{J} \end{bmatrix}, \quad B = \begin{bmatrix} \frac{1}{L} & 0 \end{bmatrix}^T \qquad (6)$$
$$C = [0 \ 1], \qquad D = [0]$$

where k_m, k_b and k_f are armature, emf and friction constants. Process noise with variance Q, is considered to affect the system as random oscillations in the current $i(t)$. While measurement noise with variance R, appears in the angular speed measurements. So that $B_w = [1 \ 0]^T$ and $D_v = [1]$.

Integrated sensor model. We are interested in studying the residuals when only the state estimate from an observer/Kalman filter is available and not the direct measured quantity. For this purpose we use a sensor model integrated with an observer as in Equation (2). Taking $C^* = C$ and the observer gain K as the stationary Kalman filter gain for the given process and noise covariances, Q and R. During our example, this model should be referred to whenever we use the notation $\hat{y}(t)$.

Faults. Two step faults with $FNR = 10$ (that produces a 10 times greater amplitude in the sensor output $y(t)$ than the present noise) are considered: process faults, $f_p(t)$, appearing as a step torque opposing to the system; sensor faults, $f_s(t)$, an offset measurement deviation. So that $B_f = \begin{bmatrix} 0 & -\frac{1}{J} \end{bmatrix}^T$ and $D_f = [1]$.

In the following Sections we discuss the performance of different methods to generate the residual considering the output of such integrated sensors.

4.1 Augmented system observer

When K is known, a possible approach to generate a residual is to augment system and sensor states $\bar{x}(t) = [x(t) \quad \bar{x}(t)]$ and use the augmented model

$$\bar{A} = \begin{bmatrix} A & 0 \\ KC & (A - KC) \end{bmatrix}, \ \bar{B}_u = \begin{bmatrix} B_u \\ B_u \end{bmatrix}, \ \bar{C} = \begin{bmatrix} 0 \\ C \end{bmatrix}^T$$

to design an observer/Kalman filter. With the redundant output $\bar{y}(t)$ from this model we can set a residual as $\bar{\varepsilon}(t) = \hat{y}(t) - \bar{y}(t)$ to detect faults. Considering our DC motor example, when we configure our observer as a Kalman filter with gain $\bar{L}$, we have for process and sensor faults a response as shown in Figure 2 from which is easy to

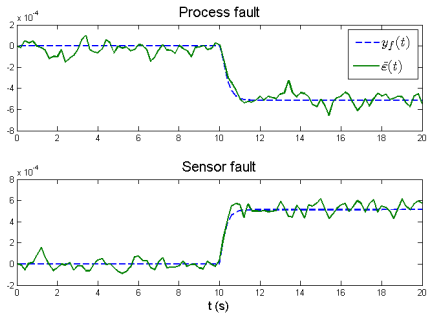


Fig. 2. Residual $\bar{y}(t)$ for process and sensor fault (solid line) together with direct fault contribution to the sensor output $y_f(t)$.

depict that the residual performance is good. In fact, for process faults we had $(\delta, \ \kappa) = (0.88, \ \approx 0)$ and sensor faults $(0.85, \approx 0)$.

Is is important to analyze how robust this method is to model errors introduced through errors in the sensor gain K. We analyze the sensitivity of the residual to K by varying it with a scaling factor as $K \times \alpha$ while $\bar{L}$ remains constant. The pair $(\delta, \ \kappa)$ is computed for α varying within $[10^{-1}, \ 10^5]$. The result is a pair $(\delta, \ \kappa) = (0.85, \ 0.30)$ in the worst case showing the robustness of the approach in this example.

4.2 Simplified system observer

As discussed in Section 3.2, we can simplify the sensor dynamics yielding a model as in Equation (5), with faults and internal observer dynamics appearing in the terms $\bar{F}_f \bar{f}(t)$ and $\bar{v}(t)$. With this simplification, we can design an observer to generate a redundancy $\breve{y}(t)$ and a residual $\breve{\varepsilon}(t) = \hat{y}(t) - \breve{y}(t)$ sensible to faults.

In our example, we set the observer gain equal to the sensor gain, $\breve{L} = K$. In this setting, the observer has similar dynamics to the sensor, in fact, if we could use the direct measured output $y(t)$ as input to our observer, sensor and observer would be equivalent. For a process fault, we have $(\delta,\ \kappa) = (0.87,\ \approx 0)$ and sensor fault $(0.84,\ \approx 0)$ which is a slightly worsened result when compared to the results in the earlier Section, with the augmented system observer.

To analyze the robustness on the gain selection, $\breve{L}$, we again, vary it with a scaling factor as $\breve{L} = K \times \alpha$ and plot the pair $(\delta,\ \kappa)$. The result, shown in Figure 3, indicates

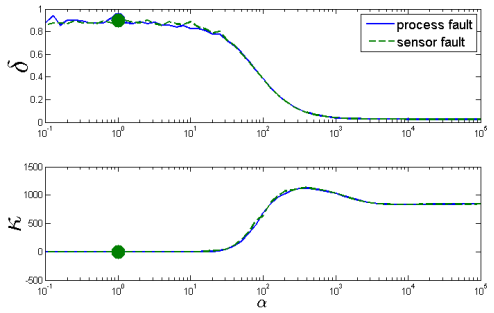


Fig. 3. δ and κ versus scaling factor α for process and sensor faults. The circle depicts the case when $\breve{L} = K$.

that the residual is worsened with the overestimation of K. This result is to be expected since, the larger the gain, the more relevance is given to the measurements and therefore, the residual will be less sensitive to faults (δ decrease). As well, it also increases the observer speed, with the observer trying to reach the signal faster and consequently increasing transient errors (κ increase).

4.3 Multiple sensors

A common approach to fault detection is to take a residual as the direct difference between two redundant sensors $\varepsilon(t) = y_i(t) - y_j(t)$. We study this case for our example comparing its performance with model-based generated residuals.

So far, our sensor estimates the states through angular speed measurements, $\hat{y}_\omega(t)$. To provide a redundancy, we depict a sensor that estimates the states through position measurements $\theta(t)$, $\hat{y}_\theta(t)$. Notice that the subscript in $\hat{y}_\theta(t)$ and $\hat{y}_\omega(t)$ denotes directly what is the measured quantity. For such redundant sensor, we have the states $[i(t) \quad \omega(t) \quad \theta(t)]$ and model,

$$A = \begin{bmatrix} \frac{-R}{L} & -\frac{k_b}{L} & 0 \\ \frac{k_m}{J} & \frac{-k_f}{J} & 0 \\ 0 & 1 & 0 \end{bmatrix}, \quad B = \begin{bmatrix} \frac{1}{L} \\ 0 \\ 0 \end{bmatrix}, \quad C = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}^T \tag{7}$$

we suppose this sensor is also integrated with a Kalman filter and outputs an estimate of $\omega(t)$. The sensor noise variance is set to produce the same error order in the output as for the first sensor so that both have similar qualities. We would like to compare the classical approach, $\varepsilon_0(t) = \hat{y}_\omega(t) - \hat{y}_\theta(t)$, with model based generated residuals. We consider one augmented states observer for each available sensor, $\bar{y}_\omega(t)$ and $\bar{y}_\theta(t)$ with the residuals as

$$\begin{aligned} \bar{\varepsilon}_\omega(t) &= \hat{y}_\omega(t) - \bar{y}_\omega(t) \\ \bar{\varepsilon}_\theta(t) &= \hat{y}_\theta(t) - \bar{y}_\theta(t) \\ \bar{\varepsilon}_{\omega,\,\theta}(t) &= \bar{y}_\omega(t) - \bar{y}_\theta(t) \end{aligned} \tag{8}$$

Because the sensors are different, the same fault may cause different influences in the output for each sensor, but since we are interested in showing the relative performance between model-based generated residuals and $\varepsilon_0(t)$, we compute a relative measure as

$$\Delta = \left| \frac{E[\varepsilon_i(t)|_F]/\sigma_{\varepsilon_i(t)|_F}}{E[\varepsilon_0(t)|_F]/\sigma_{\varepsilon_0(t)|_F}} \right|$$

where $\varepsilon_i(t)$ is one of the residuals from Equation 8. $\Delta > 1$ will depict a residual with a larger fault sensitivity than the one generated by $\varepsilon_0(t)$. Table 1 shows Δ for process and sensor faults. The results show that taking

Residual	Δ for process fault	Δ for sensor $\hat{y}_\omega(t)$ fault
$\bar{\varepsilon}_\omega(t)$	0.1498	1.6059
$\bar{\varepsilon}_\theta(t)$	1.3890	≈ 0.00
$\bar{\varepsilon}_{\omega,\,\theta}(t)$	3.0104	0.02

Table 1. Δ for different faults. In all cases $\kappa \approx 0$.

the residual as the direct difference between sensors, $\varepsilon_0(t)$, may not provide the best residual. It is expected this would be even more significant in case the noise variances differed considerably for each sensor, since the observers also attenuate noise.

4.4 Example summary

Different aspects have been analyzed trough our illustrative examples, some important remarks:

- Though this was not fully explored in the example, different tuning configurations have been used and it was noticed that the use of an augmented states observer is likely to improve the fault sensitivity when compared to the observer using a simplified sensor model as presented in Section 4.2.
- The analysis on the observer gain choice in Section 4.2 indicates that the fault sensitivity is improved as smaller we choose the gain. Such result is motivated by the fact that lower gains will thrust more on the model and therefore, the resulting residual will be more sensitive to unmodeled influences, such as faults. Nevertheless, the choice of the gain should actually be seem as a compromise between model uncertainties and the fault sensitivity.
- Finally, the example with multiple sensors depicted that using a model-based residual can improve the fault sensitivity.

5. CONCLUSIONS

The paper analyzed several structures for observers data only fault detection. Section 2 discussed standard approaches for fault detection; Section 3 presented ideas and addressed some basic questions for the problem, including a discussion over fault observability, knowledge on observer structure and residual performance measures. Finally, Section 4 illustrated the problem through a simulated example, covering the approaches for fault detection using redundant sensors and Kalman filter based methods with known and unknown sensor structure. Most of the methods have shown to be useful, with slight improvements when one consider both system and sensor states in the estimation.

There are yet some open problems such as more general observability conditions for faults, methods to support the choice of the observer gains, analysis under model uncertainties, etc. which shall be presented in future work, together with example from a real robotics application.

REFERENCES

P. Sundvall, P. Jensfelt and B. Wahlberg. Fault detection using redundant navigation modules. *Proc. of the 6th IFAC Symposium on Fault Detection, Supervision and Safety of Technical Processes*, SAFEPROCESS 2006.

P. Sundvall. Mobile Robot Fault Detection using Multiple Localization Modules. *Kungliga Tekniska högskolan*, TRITA-EE 2006:044.

F. Gustafsson. Statistical signal processing approaches to fault detection. *Annual Reviews in Control*, 31:41–54, 2007.

F. Gustafsson. Adaptive filtering and change detection. Wiley, 2001.

N. Liu and K. Zhou. Optimal Robust Fault Detection for Linear Discrete Time Systems. *Journal of Control Science and Engineering*, Volume 2008, Issue 1, 2008.

D. Törnqvist. Statistical Fault Detection with Applications to IMU Disturbances. *Linköping Studies in Science and Technology*, Thesis No. 1258, 2006.

G. F. Franklin, J. D. Powell and A. Emami-Naeini. Feedback Control of Dynamic Systems. Prentice Hall, 5th edition, 2006.

P. R. Kalata and T. A. Chmielewski Jr. On the Identification of Stochastic Biases in Linear Time Invariant Systems. *In Proceedings of the American Control Conference*, pp 4067-4071, Seattle, Washington, 1995.

L. J. Hadjileontiadis, E. Douka, A. Trochidis. Crack detection in beams using kurtosis. *Computers and Structures*, 83:-919, (2005).

S. Gillijns, B. De Moor. Unbiased minimum-variance input and state estimation for linear discrete-time systems. *Automatica*, 43:-116, (2007).

Appendix A. FAULT OBSERVABILITY

The Popov- Belevitch-Hautus test on the augmented extended systems depicts a pair (A,C) to be observable if

$$\begin{pmatrix} C \\ A - sI \end{pmatrix}$$

has full column rank for all s (see Franklin et al. [2006] for more). The faults modes are $s = 1$, therefore the analysis is separated for $s \neq 1$ and $s = 1$.

Given that our original system is observable (pair (A, C) observable), we analyze the observability for the augmented system as in Equation (3). First, when $s \neq 1$ the observability is given by:

$$\begin{pmatrix} 0 & C^* & 0 \\ (A - sI) & 0 & B_f \\ KC & ((A - KC) - sI) & KD_f \\ 0 & 0 & (1 - s)I \end{pmatrix}$$

The last two columns are full column rank for $s \neq 1$ and $C^* = I$, and the analysis can be simplified to

$$\begin{pmatrix} (A - sI) \\ KC \end{pmatrix} \tag{A.1}$$

Given that for a pair (A, C) to be observable, they should have non-intersecting null-spaces ($\mathcal{N}_{A-sI} \cap \mathcal{N}_C = 0$), the condition above is translated to $\mathcal{N}_{A-sI} \cap \mathcal{N}_{KC} = 0$. Using Theorem 1, it is rewritten as $\mathcal{N}_{A-sI} \cap \mathcal{N}_C = 0$ which is actually the observability condition on the original system, which holds and the condition for $s \neq 1$ is checked.

Theorem 1. Given K is full column rank, for any B we have:

$$\mathcal{N}_{KB} = \mathcal{N}_B$$

Proof: Suppose $\mathcal{N}_K = \emptyset$. Such that

$$Kv = 0 \quad \rightarrow v = 0$$

Now, let us check the null space of KB

$$(KB)w = 0, \quad K(Bw) = 0 \quad \rightarrow Bw = r \in \mathcal{N}_K \quad \rightarrow Bw = 0$$

finally, the solution for $Bw = 0$ is the null-space of B and therefore,

$$\mathcal{N}_{KB} = \mathcal{N}_B$$

When $s = 1$ and $C^* = I$, it is equivalent to analyze

$$\begin{pmatrix} (A - I) & B_f \\ KC & KD_f \end{pmatrix}$$

For *measurement faults* ($B_f = 0$) and considering that full column rank of a matrix A is equivalent to

$$Av = 0 \quad \Leftrightarrow v = 0$$

we have

$$\begin{pmatrix} (A - I) & 0 \\ KC & KD_f \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix}$$

and $x \in \mathcal{N}_{A-I}$, $KCx + KD_f y = 0$. Notice that if A has no integrators, then the condition is achieved if D_f is rank. If A has integrators, then D_f should not be full rank and we analyze that the first condition is only true when x is zero or an eigenvector of A with eigenvalue equals to 1. Hence, it is sufficient to analyze such x.

Take U as a basis formed by the eigenvectors of A with eigenvalues 1 and rewrite the last condition as $K(CUr + D_f y) = 0$ where r is any vector. For full column rank K, this condition can only be true if CU and D_f share image spaces and the condition on the observability can be rewritten as $\mathcal{R}_{CU} \cap \mathcal{R}_{D_f} = \emptyset$ Which means that *the faults should be orthogonal to the measured integrating part of the system.*

For *process faults* ($D_f = 0$) only we can analyze

$$\begin{pmatrix} (A - I) & B_f \\ KC & 0 \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix}$$

and we have $x \in \mathcal{N}_C$ (Theorem 1) and $(A-I)x + B_f y = 0$. Taking W as a basis for the null-space of C the final condition is $\mathcal{R}_{(A-I)W} \cap \mathcal{R}_{B_f} = \emptyset$.

Which means that the *faults must be orthogonal to the contribution of the non-measured part of the system.*