FACULDADE DE ENGENHARIA DA UNIVERSIDADE DO PORTO

Dependability Evaluation of Wireless Visual Sensor Networks

Thiago Cerqueira de Jesus

U.PORTO
FEUP FACULDADE DE ENGENHARIA
UNIVERSIDADE DO PORTO

Doctoral Programme in Electrical and Computer Engineering

Supervisor: Paulo Portugal

Co-Supervisor: Daniel Gouveia Costa

September 21, 2021

Dependability Evaluation of Wireless Visual Sensor Networks

Thiago Cerqueira de Jesus

Doctoral Programme in Electrical and Computer Engineering

September 21, 2021

Resumo

Redes de sensores sem fio têm sido consideradas uma solução eficaz para uma ampla gama de aplicações devido às suas características proeminentes no que diz respeito à recuperação de informações e processamento distribuído. Quando as informações visuais podem também ser recuperadas por nós sensores, as aplicações adquirem uma percepção mais abrangente dos ambientes monitorados, promovendo a criação de Redes de Sensores Visuais Sem Fio. Essas redes estão sendo consideradas de forma mais frequente para o desenvolvimento de aplicações críticas de monitoramento e controle, geralmente relacionadas à prevenção de situações catastróficas, aprimoramento da segurança e gerenciamento de crises. Como consequência, a tolerância a falhas se torna um grande problema para as redes de sensores visuais sem fio, sendo uma forma de alcançar *dependability*. Por sua vez, a *dependability* do sistema pode ser estimada por meio de atributos quantitativos (por exemplo, confiabilidade e disponibilidade), que requerem uma estratégia de modelagem adequada para descrever o comportamento do sistema.

Dessa forma, nesta tese propomos uma metodologia para modelar e avaliar analiticamente a *dependability* de redes de sensores visuais sem fio usando Análise de Árvores de Falhas e Cadeias de Markov. A metodologia define um modelo unificado compreensivo, considerando especialmente as particularidades das aplicações baseadas em sensores. A estratégia de modelagem proposta considera falhas de hardware, de bateria, de links de comunicação e de cobertura visual, além de considerar protocolos de roteamento na operação da rede. A metodologia proposta é automatizada por um framework desenvolvido e integrado com a ferramenta SHARPE. A fim de tornar a metodologia útil para aplicações reais, é discutido como os parâmetros do modelo podem ser alcançados em cenários práticos, potencialmente suportando avaliações mais eficazes de *dependability*.

A metodologia também explora vários aspectos visuais relacionados à *dependability* do sistema. Um desses aspectos é a presença de obstáculos na área de monitoramento, que podem obstruir a visão das câmeras, reduzindo a área de cobertura efetiva. Também é abordada a qualidade do monitoramento em relação à distância de visão. Essa qualidade está relacionada à nitidez das imagens coletadas das câmeras, e mostramos que é importante definir a utilidade desta informação visual na aplicação. A metodologia proposta também pode ser utilizada para guiar a reimplantação de nós visuais de forma a maximizar a *dependability* do sistema, sendo útil também para avaliar a *dependability* da rede resultante, no sentido de garantir a otimização.

Os resultados obtidos mostram que esta metodologia é útil para comparar diferentes cenários de rede e a respectiva *dependability*, possibilitando a identificação de potenciais pontos fracos associados à *dependability*, além de ser uma contribuição para auxiliar na etapa de projeto de aplicações baseadas em redes de sensores visuais sem fio.

Abstract

Wireless sensor networks have been considered as an effective solution to a wide range of applications due to their prominent characteristics concerning information retrieving and distributed processing. When visual information can also be retrieved by sensor nodes, applications acquire a more comprehensive perception of monitored environments, fostering the creation of Wireless Visual Sensor Networks. Such networks are being more often considered for the development of critical monitoring and control applications, usually related to prevention of catastrophic situations, security enhancement and crises management. As consequence, fault tolerance becomes a major issue for wireless visual sensor networks, being a way to achieve dependability. In turn, system dependability can be estimated through quantitative attributes (*e.g.*, reliability and availability), which require a proper modelling strategy to describe the system behavior.

That way, in this thesis we propose a methodology to analytically model and evaluate the dependability of wireless visual sensor networks using Fault Tree Analysis and Markov Chains. The methodology defines a comprehensive unified model, specially considering the particularities of sensors-based applications. The proposed modelling strategy considers hardware, battery, communication links and visual coverage failures, besides to consider routing protocols on the network operation. The proposed methodology is automated by a framework developed and integrated with the SHARPE tool. In order to become the methodology useful for real applications, it is discussed how the parameters of the model can be achieved in practical scenarios, potentially supporting more effective dependability evaluations.

The methodology also exploits several visual aspects related to system dependability. One of these aspects is the presence of obstacles in the monitoring area, which can occlude the sight of view of cameras, reducing the effective coverage area. It is also addressed the quality of monitoring regarding the distance of view. This quality is related to the sharpness of the gathered images from the cameras, and we show that it is important to define the utility of this visual information in the application. The proposed methodology can also be used to planning redeployment of visual sensor nodes in order to maximize the system dependability, being also useful to assess the dependability of the resulting network in order to assure the optimization.

The achieved results show that this methodology is useful to compare different network scenarios and the corresponding dependability, enabling the uncovering of potentially weak points associated to dependability, besides to be a contribution for aid the design stage of wireless visual sensor networks-based applications.

Acknowledgment

I dedicate this section especially to my family and close friends. So I will allow myself to write it in Portuguese.

Em relação a essa caminhada para a obtenção do grau de doutor, eu gostaria de dedicar os meus mais sinceros agradecimentos para:

– Deus, antes de tudo, pois creio que “d’Ele, por Ele e para Ele são todas as coisas” (Romanos 11:36). Muito obrigado pelo cuidado e provisão sempre!

– a minha amada esposa, Cristina, por estar ao meu lado desde o primeiro dia deste sonho, me apoiando, incentivando e enchendo de amor e carinho. Te amo muito!

– a minha filha, Luísa, que embora tenha me tomado muito tempo na reta final com muitas risadas, brincadeiras e noites em claro, veio para me dar mais uma razão para querer alcançar a excelência. Espero ser um bom exemplo para você.

– os meus pais, Noemia e Luiz, por terem me ensinado desde menino, entre centenas de valores, a importância da educação, perseverança, respeito, humildade, ambição e equilíbrio. Muito obrigado por mesmo longe se fazerem tão presentes.

– o meu irmão, Diego, por acreditar em mim muito mais do que eu mesmo. Espero um dia ser o homem que você acha que eu sou.

– todos meus demais familiares, em especial as minhas sobrinhas, sogros e cunhados(as), que sempre me incentivaram indistintamente.

– o meu orientador e amigo, Paulo Portugal, pela sinceridade, respeito, comprometimento, paciência, dedicação, confiança, conselhos, dicas de viagem e aulas de história e da cultura portuguesa. Muito obrigado!

– os meus co-orientadores e amigos, Daniel Costa e Francisco Vasques, pelas provocações sempre assertivas, pelo “toque de Midas” nas revisões de artigos, pelo apoio e incentivo durante todo o trabalho.

– a minha imensa família da Igreja Evangélica Baptista de Cedofeita. Vocês são espetaculares. Que vontade de nomear e abraçar cada um de vocês. Obrigado por nos acolher e cuidar de nós com tanta devoção. Serei eternamente grato! Em especial, às pessoas que primeiro nos receberam lá, o Pastor Abel Pego e sua esposa, Isabel Pego.

– os meus senhorios, Dona Nina (*em memória*) e Sr. Jorge, pelo carinho e amizade cultivados nos acalentadores encontros mensais para chá com biscoitos. Sentiremos muita saudade!

– os amigos de laboratório, Benedito, Renata, Anurag, Rogério e Suelen, pela descontração e troca de experiência em momentos críticos.

– todos meus outros verdadeiros amigos, que prefiro não citar seus nomes para não cometer nenhuma injustiça, por estarem sempre dispostos para o que fosse necessário.

– a Universidade Estadual de Feira de Santana, por me proporcionar esse tempo de aprimoramento.

– os professores Manuel Barranco, Mário Alves, Ana Aguiar e Manuel Ricardo, por avaliarem este trabalho dando dicas valiosas para seu melhoramento e continuidade.

– todos que, direta ou indiretamente, contribuíram para a realização deste trabalho, cujos nomes não citei.

A todos vocês, muito obrigado.

Thiago C. Jesus, PhD! =D

“All truths are easy to understand
once they are discovered.
The point is to discover them.”

Galileo Galilei

“...when you have eliminated the impossible,
whatever remains, however improbable,
must be the truth.”

Sherlock Holmes
(Sir Arthur Conan Doyle)

Contents

List of Figures

List of Tables

List of Abbreviations

AQM	Area Quality Metric
CA	Coverage Area
CCF	Common Cause Failure
CDF	Cumulative Distribution Functions
CTMC	Continuous Time Markov Chain
FoV	Field of View
FT	Fault Tree
FTA	Fault Tree Analysis
IoT	Internet of Things
MA	Monitoring Area
MB	Monitoring Block
MC	Markov Chain
MTTF	Mean Time To Failure
NFC	Network Failure Condition
OFoV	Occluded Field of View
PDF	Probability Density Functions
PRR	Packet Reception Rate
QoM	Quality of Monitoring
QoS	Quality of Service
RBD	Reliability Block Diagrams
SDP	Sum of Disjoint Products
SMP	Semi-Markov Process
SPN	Stochastic Petri Nets
VCF	Visual Coverage Failure
WSN	Wireless Sensor Network
WVSN	Wireless Visual Sensor Network

List of Symbols

$\mathbb{A}$	monitoring field
AQM	area quality metric
AQM_{abs}	absolute area quality metric
AQM_{min}	minimum quality of monitoring of a network deployment
AQM_{rel}	relative area quality metric
$A(t)$	system availability
α	orientation angle of camera from a Visual sensor node
B_i	i-th battery charge interval (stage)
β	rotation angle of a monitoring area MA
CA_{mb}	coverage area computed through monitoring blocks
CA_{min}	minimum visual coverage percentage
c_0	initial capacity of the battery
FoV_{min}	minimum field of view of a visual sensor node
H	hour rating
h	height of a monitoring area MA
hs	height of a monitoring block
I	average continuous discharge current
i	index of visual nodes
j	index of scalar nodes
k	index of monitoring block horizontally
l	index of monitoring block vertically
λ	failure rate
m	number of scalar nodes
mb	monitoring block
μ	repair rate
n	number of visual nodes
$nStages$	number of battery stages
η	Peukert's constant
$Q_0(t)$	system unreliability
$R(t)$	system reliability
R_s	sensing radius of a visual sensor
SS	set of scalar sensor nodes
ss	scalar sensor node
T	dependability evaluation period
t_s	dependability evaluation time step
θ	viewing angle of a visual sensor
VS	set of visual sensor nodes
vs	visual sensor node

ws	width of a monitoring block
w	width of a monitoring area MA

Chapter 1

Introduction

Advances in technology are conducing the world to a scenario of intense automatization of processes with distributed and connected operations, in which Wireless Sensor Networks (WSN) have been considered as an effective solution for a wide range of applications. This is due to the prominent characteristics of WSN concerning information retrieving and distributed processing. When visual information (images and videos) can be also retrieved by sensor nodes, applications acquire a more comprehensive perception of monitored environments, fostering the creation of Wireless Visual Sensor Networks (WVSN). Examples are applications in the context of Industry 4.0, smart cities, intelligent transportation systems, Internet of Things (IoT), agriculture and military operations. For such applications, WVSN can be used for intrusion detection, face or pattern recognition, manufacturing inspection, environmental monitoring, security footage, object tracking, etc. Those sensor-based applications have become part of everyday life, which are, in many cases, safety-critical, which means that a failure in such kind of systems may put people in danger, lead to environmental damages or result in economic losses.

Therefore, it is mandatory to evaluate the system dependability in those situations in order to assess its successful operation behavior through time. This evaluation brings several benefits to the application during the design, operation and maintenance phases. In the design phase it is possible to identify weak points in the application, focusing on their mitigation. The time span while the system operates without the occurrence of any failure can be computed in sense to adequate system's parameters to adjust the duration of the application operation or to guide a (re)design. For this last case, metrics can be provided for comparison among different application, finding an improved network implementation and deployment (or redeployment, in case of the maintenance phase).

In this context, expecting to provide applicable metrics to evaluate dependability of wireless visual sensor networks and leverage the phases of an application, several problems have emerged, as it is discussed in the next section. Most of these problems are intrinsically associated to the visual nature of the data gathered from these networks, which impacts significantly the notion of coverage.

1.1 Problem Statement

Dependability is a generic concept including attributes as availability, reliability, safety, integrity and maintainability, being commonly addressed by quantitative reliability and availability metrics. Reliability is associated with the continuity of correct system behavior, while availability is associated with the readiness of the system (*i.e.*, the system is operating at that time), which requires repair actions to take the system from a failed state to an operational state (Avizienis et al., 2004).

Proper dependability evaluation demands an adequate modelling to describe the system behavior when faults occur. This has fostered the development of models for several network elements, such as sensor nodes, their connectivity, different types of failures, eventual repairments and application requirements, in the most feasible realistic description. This can be a huge challenge, due to the several uncertainties or lack of sufficient data. On the other hand, an useful system model must be generic enough to be applied to a wide range of systems. However, a very detailed model requires much more parameters to estimate, which can also be unpractical to achieve. The trade-off between detailed and practical model must be handled for a valuable dependability evaluation.

Several works have addressed dependability evaluation on WSN, but there are still relevant aspects to be approached regarding assessment on WVSN. Jointly with a new category of wireless sensor network, new challenges related to dependability modelling and evaluation have arisen. The notion of visual coverage is the paramount change in WVSN, since the retrieved visual can represent individual cameras, the network as a whole or the relationships between cameras in terms of their coverage, such as overlapping (Mavrinac and Chen, 2013).

It is common on visual sensors applications to consider one of three types of coverage: area coverage, target coverage or barrier coverage (Akyildiz et al., 2007; Wang and Cao, 2011). Area coverage is related to monitoring one or more areas of the monitored field. Target coverage is focused on monitoring of a set of targets. Finally, barrier coverage creates a conceptual barrier that avoids undetected penetration. Several works have addressed dependability on target coverage WVSN, but there is still a lack of research related to both area and barrier coverage (Costa and Guedes, 2010; Costa et al., 2014a; Si et al., 2017).

Area coverage brings some special issues that directly affect dependability, which are not yet properly approached in the literature. The correct covered area computation have demanded the adequate perception of redundant covered areas, which is a combinatorial problem and it can imply a solution of exponential complexity. In general, differently of target and barrier coverage, area coverage is not affected by perspective, *i.e.*, the angle of coverage from cameras does not alter the information importance. As a consequence, the solution space for deployments and redeployments is much more vast. Another characteristic inherent to area coverage is the higher computational complexity when addressing issues such as Quality of Monitoring (QoM) and the occlusion, since the monitoring task is extended to entire sub-regions covered by each camera, instead of to be limited to a specific point (target) or line (barrier).

Quality of monitoring and occlusion are also related to other important aspects of WVSN: visual coverage failures. Applications based on such networks do not fail only when its nodes fail or when the connectivity among them has been interrupted. It is also necessary to associate the capability of gathering visual information with the application requirements. Hence, it is expected that the quality of gathered data, such as the definition and sharpness of images, be considered in the applications, as well as occlusion issues on active cameras, when their field of view is blocked by obstacles. As a consequence, it is natural to consider the possibility of network redeployment in order to optimize the system dependability by replacing or re-orientating the cameras.

Redeployment in sensor networks is generally performed by adjusting or improving some network parameter, such as network connectivity (Banfi et al., 2018), lifetime (energy consumption) (Kuawattanaphan et al., 2013) or coverage of a well-defined objective (area (Costa et al., 2017a), target (Rangel et al., 2018) or barrier (Nguyen and So-In, 2018)). An expected consequence of the "improvement" of one of these elements should be the increase of the application dependability. However, it is common to evaluate these parameters separately or individually, in spite of the intrinsic relationship that exists among them. Such independent approaches can affect the application dependability in an undesirable way. Therefore, it is worth and necessary to guide redeployment by dependability metrics that consider all these elements in an integrated way, providing a certain level of dependability guarantees to the redeployed network.

Considering the described issues, the next section defines the scope of this thesis.

1.2 Research Objectives and Thesis Hypothesis

The main objective of this thesis is to develop an automated methodology to evaluate analytically the dependability of Wireless Visual Sensor Networks for area coverage, considering visual coverage failures beyond the typical hardware and link failures. It is also intended to analyze the impact of different routing protocols upon the network communication behavior and others aspects related to system dependability of WVSN, such as occlusion, quality of monitoring and optimization. All these topics must be addressed together within an integrative modelling, allowing a better understanding of the perceived applications quality.

Therefore, the aim of this thesis is to prove the following hypothesis: "*It is possible to evaluate quantitatively the dependability of wireless visual sensor networks for area coverage under visual coverage failures based on models that are realistic, simple to build and computationally feasible*".

1.3 Main Contributions

1.3.1 Methodology

The strategy to achieve the research objectives begins with designing of a methodology to evaluate dependability of WVSN for area coverage. The proposed methodology can be useful to compare different network scenarios, being also applied to ordinary WSN, since its network behavior models also consider battery, communication links and hardware failures, and the impact of different routing strategies. This methodology is based on an analytical approach, using Continuous Time Markov Chains (CTMCs) models hierarchically integrated to a Fault Tree (FT).

The methodology is designed to be able to evaluate networks under several visual failures related with area coverage, quality of monitoring and occlusion. For that, it is necessary to develop new metrics and models of QoM and occlusion in order to describe the covered area and do not impose any restrictions to visual sensor nodes in terms of position, orientation and viewing angle in a 2D scenario.

1.3.2 Framework

Since the methodology follows an analytical evaluation, it requires the development of analytical models, which can be a time-consuming task. To circumvent this problem, we develop a framework to automate the entire methodology, consisting at the generation and combination of individual models for each network element (hardware, battery and link), according to network topology and following the predefined steps.

The framework is integrated with the SHARPE (Symbolic Hierarchical Automated Reliability and Performance Evaluator) tool (Trivedi and Sahner, 2009), taking advantage of SHARPE's support to hierarchical models and different modelling techniques. The most relevant aspect of the framework is to be able to compute automatically the network failure condition (NFC) of a WVSN based on application requirements. In order to make the methodology execution feasible, it is necessary to propose efficient approaches for computing area coverage and QoM modelling and calculation, allowing therefore the usage of the methodology into optimization processes.

1.3.3 Redeployment

This thesis also proposes a procedure to use the developed framework to redeploy visual nodes in order to maximize system dependability. Since the problem of optimally placing wireless sensor nodes has been proven to be a NP-hard problem (Al-Karaki and Gawanmeh, 2017), a variety of heuristic approaches have been developed to find sub-optimal solutions in a reasonable time (Senouci and Abdellaoui, 2017), from simpler approaches like greedy algorithms, to more powerful and complex solutions like genetic algorithms. In this thesis, optimization algorithms and heuristics are proposed, tested and compared, always guided by dependability metrics that consider connectivity, energy consumption and visual coverage in an integrated manner.

1.3.4 Practical Aspects

Finally, this thesis also aims at filling a gap from previous evaluation methodologies addressing practical aspects and clarifying how to define or to estimate several application and model's parameters. Furthermore, it is also shown how to use the methodology to guide the system design steps and a case study of the dependability assessment of an industrial network.

1.4 Publications

The work presented in this thesis has led to the following publications:

Journals

1. **T. C. Jesus**, D. G. Costa, P. Portugal and F. Vasques, "Quality Assessment of Wireless Visual Sensor Networks: a Review," *in International Journal of Distributed Sensor Networks*, submitted for publication.
2. **T. C. Jesus**, P. Portugal, D. G. Costa, F. Vasques, "A Comprehensive Dependability Model for QoM-Aware Industrial WSN when Performing Visual Area Coverage in Occluded Scenarios," *Sensors*, vol. 20, no. 22, 2020, doi: 10.3390/s20226542 (Jesus et al., 2020).
3. **T. C. Jesus**, D. G. Costa, P. Portugal and F. Vasques, "FoV-Based Quality Assessment and Optimization for Area Coverage in Wireless Visual Sensor Networks," *in IEEE Access*, vol. 8, pp. 109568-109580, 2020, doi: 10.1109/ACCESS.2020.3002206 (Jesus et al., 2020a).
4. **T. C. Jesus**, D. G. Costa, P. Portugal, F. Vasques and A. Aguiar, "Modelling Coverage Failures Caused by Mobile Obstacles for the Selection of Faultless Visual Nodes in Wireless Sensor Networks," *in IEEE Access*, vol. 8, pp. 41537-41550, 2020, doi: 10.1109/ACCESS.2020.2977173 (Jesus et al., 2020b).
5. **T. C. Jesus**, P. Portugal, F. Vasques, D. G. Costa, "Automated Methodology for Dependability Evaluation of Wireless Visual Sensor Networks," *Sensors*, vol. 18, no. 8, 2018, doi: 10.3390/s18082629 (Jesus et al., 2018b).

Conferences

1. **T. C. Jesus**, D. G. Costa and P. Portugal, "Wireless visual sensor networks redeployment based on dependability optimization," *IEEE 17th Int'l. Conf. on Industrial Informatics (INDIN)*, Helsinki, Finland, 2019, pp. 1111-1116, doi: 10.1109/INDIN41052.2019.8972128 (Jesus et al., 2019).
2. D. G. Costa, E. Rangel, J. P. J. Peixoto and **T. C. Jesus**, "An Availability Metric and Optimization Algorithms for Simultaneous Coverage of Targets and Areas by Wireless Visual Sensor Networks," *IEEE 17th International Conference on Industrial Informatics (INDIN)*, Helsinki, Finland, 2019, pp. 617-622, doi: 10.1109/INDIN41052.2019.8972176 (Costa et al., 2019).
3. **T. C. Jesus**, D. G. Costa and P. Portugal, "On the Computing of Area Coverage by Visual Sensor Networks: Assessing Performance of Approximate and Precise Algorithms," *IEEE 16th International Conference on Industrial Informatics (INDIN)*, Porto, 2018, pp. 193-198, doi: 10.1109/INDIN.2018.8471997 (Jesus et al., 2018a).

1.5 Thesis Organization

The remainder of the thesis is organized as follows. The background knowledge necessary to support this thesis' contributions is described in Chapters 2, 3 and 4. In Chapter 2 the basic dependability concepts and terminology are presented, as well as the dependability modelling and evaluation methods used in this thesis. The problem addressed and its theoretical background are presented and formulated in detail in Chapter 3. This encompasses the formal modelling of wireless visual sensor networks and visual coverage failures. Chapter 4 presents related works regarding dependability on WSNs and WVSNs, as well as regarding related topics, like visual coverage, quality of monitoring and occlusion.

The main thesis' contributions are described in Chapters 5 and 6. The proposed dependability methodology is presented in Chapter 5, detailing the modelling and evaluation phases, considering visual coverage failures related only network deployment. It is also presented and discussed some results according to the evaluation of different WVSN scenarios. In Chapter 6, the methodology scope is extended to address the visual aspects that affect system dependability. This is achieved by considering visual coverage failures related to occlusion and quality of monitoring. It is discussed the modelling of obstacles for occlusion computation, the proposal of quality monitoring metrics and the optimization algorithms to improve system dependability.

Finally, conclusions are stated and possible future works are discussed in Chapter 7.

Part I

Background

Chapter 2

Dependability Concepts, Modelling, Evaluation

This chapter presents notions and required preliminary concepts to understand the proposals presented in this thesis. Among these concepts are the threats for a dependable system, dependability attributes and the means to achieve dependability. Furthermore, some modelling and evaluation techniques used to assess system dependability are also introduced.

Considering that a system is an entity that interacts with other entities, *i.e.*, other systems or subsystems (including hardware, software, humans and the physical world with its natural phenomena), then dependability can be defined as *the system ability to deliver a service that can be justifiably trusted, avoiding service failures more frequent or more severe than is acceptable* (Avizienis et al., 2004). Dependability can be measured by several *attributes* such as reliability, availability, safety, integrity, maintainability. The attributes can be achieved dealing properly with dependability *threats* (faults, errors, failures) by different *means*, like fault prevention, fault tolerance, fault removal and fault forecasting, according to dependability tree shown in Figure 2.1. These concepts are clarified next.

2.1 Threats

A dependability threat is any event or phenomenon that results into undesired circumstances. The primary threat is a *fault*, that is the adjudged or hypothesized cause of an *error*, which, in turn, can possibly lead to a *failure*. An error is the internal part of the total state of the system that may produce its subsequent failure. Finally, a failure occurs when the system service deviates from the correct (specified) service (Avizienis et al., 2004). Notice that the failures of a subsystem can characterize the faults of the superior system in which the subsystem is inserted.

Since an error affects the internal states of a system, it will lead to a failure if some trigger event propagates that error until it reflects at the external boundary of the system. That way we focus this thesis discussion in identifying, describing and modelling faults and failures that can affect

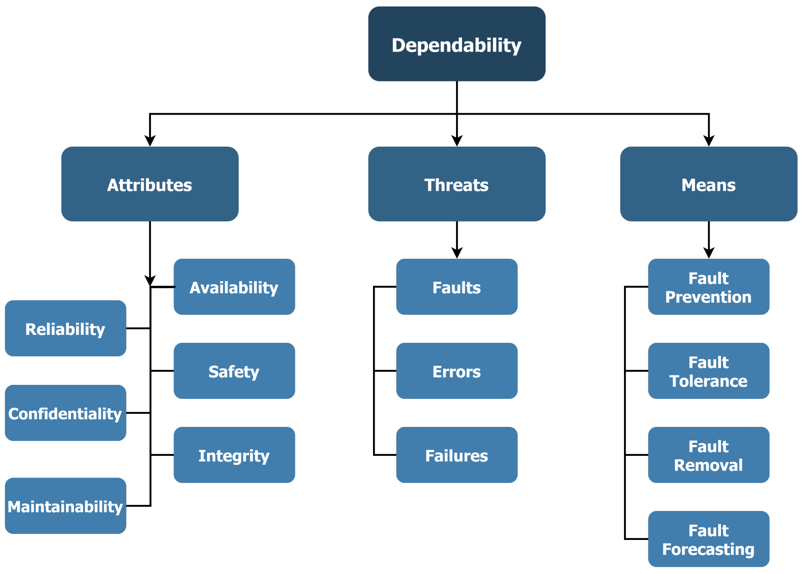


Figure 2.1: Dependability Tree (Avizienis et al., 2004).

wireless visual sensor networks, which are the initial cause of a malfunction and the perceived undesired behavior, respectively.

2.1.1 Faults

The prevention and contingency measures that may be taken to assure dependability of a system should be planned according to the nature of faults, which can be classified according to eight basic viewpoints Avizienis et al. (2004). Each viewpoint may assume two types of faults, deriving the elementary fault classes, as shown in Figure 2.2. A fault can be categorized by more than one viewpoint, totaling 256 different combined fault classes. However, some of these combinations make no sense, remaining only 31 likely combinations of fault classes, according to Avizienis et al. (2004), shown in Figure 2.3. For instance, a fault due to physical deterioration can be defined by the classes combination 12 or 13, *i.e.*, they occur during the *operational* phase, *internal* to the system boundaries, by *natural* phenomenon, in *hardware* dimension, with *non-malicious* objective, with *non-deliberate* intent, *accidental* capability and both *persistent* or *transient*.

Once defined the possible fault classes combinations, they should be used to specify the fault model, which consists in the set of fault classes that can affect the system. A fault-tolerant system must be designed and evaluated to tolerate all faults that are described by a fault model (Warns, 2010). Therefore, this model is useful to predict the consequences of a given fault and so to determine test routines, prevention and contingency measures (Pretschner et al., 2013).

2.1.2 Failures

A system failure is the ultimate manifestation of a fault, implying in a deviation from the intended service and being perceived through the system behavior. That way, in order to deal properly with the cause (fault) of a malfunction and to facilitate the design of a dependable system, it is necessary

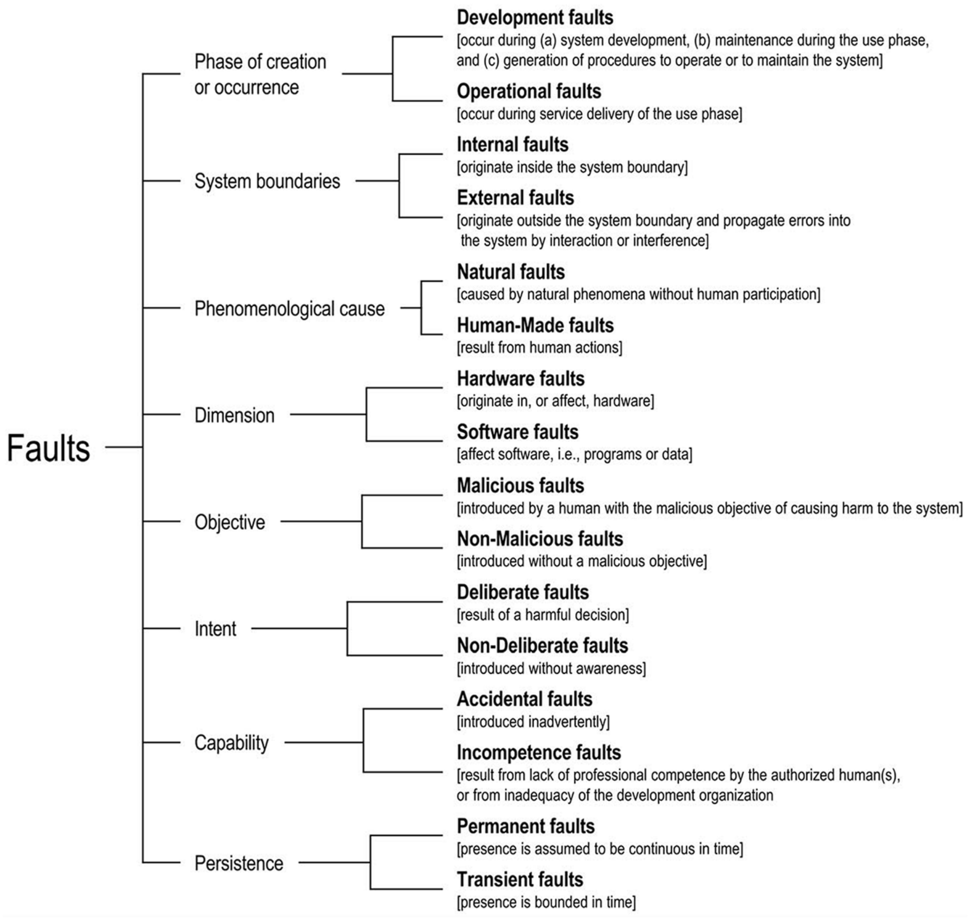

Faults
Phase of creation or occurrence
Development faults
[occur during (a) system development, (b) maintenance during the use phase, and (c) generation of procedures to operate or to maintain the system]
Operational faults
[occur during service delivery of the use phase]
System boundaries
Internal faults
[originate inside the system boundary]
External faults
[originate outside the system boundary and propagate errors into the system by interaction or interference]
Phenomenological cause
Natural faults
[caused by natural phenomena without human participation]
Human-Made faults
[result from human actions]
Dimension
Hardware faults
[originate in, or affect, hardware]
Software faults
[affect software, i.e., programs or data]
Objective
Malicious faults
[introduced by a human with the malicious objective of causing harm to the system]
Non-Malicious faults
[introduced without a malicious objective]
Intent
Deliberate faults
[result of a harmful decision]
Non-Deliberate faults
[introduced without awareness]
Capability
Accidental faults
[introduced inadvertently]
Incompetence faults
[result from lack of professional competence by the authorized human(s), or from inadequacy of the development organization
Persistence
Permanent faults
[presence is assumed to be continuous in time]
Transient faults
[presence is bounded in time]

Figure 2.2: Elementary fault classes (Avizienis et al., 2004).

to better understand that undesired behavior (failure). For this, failure modes are defined to identify and classify the nature of failures, describing how a component that is failed may behave (Warns, 2010). Figure 2.4 shows a scheme of failure modes and the inclusion relationship between them.

The more generic failure mode is the *Byzantine failure*, characterized by an arbitrary system failure. In this case, the system shows an arbitrary behavior (Warns, 2010). Similarly, an *authenticated Byzantine failure* presents the same byzantine behavior, however the system messages (information) are authenticated. This means that a system cannot lie about facts which are sent by other systems, which is commonly accomplished by authenticated messages (Derasevic, 2018).

The *incorrect computation failures* consists into deliver incorrect results, either in the time domain or in the value domain, without malicious or inconsistent behavior (Derasevic, 2018). On the other hand, *timing failures* occur when the system delivers correct results in the value domain, however they are faulty in the time domain, *i.e.*, the results may be delivered early or late (Poledna, 2007). If the fault in the time domain is a late delivery of results with infinite delay, then it is characterized a *omission failure* (Poledna, 2007).

In case of a *crash failure*, a system may behave as usual, but eventually halts prematurely, not responding to the current or any subsequent service request (Warns, 2010; Derasevic, 2018). If

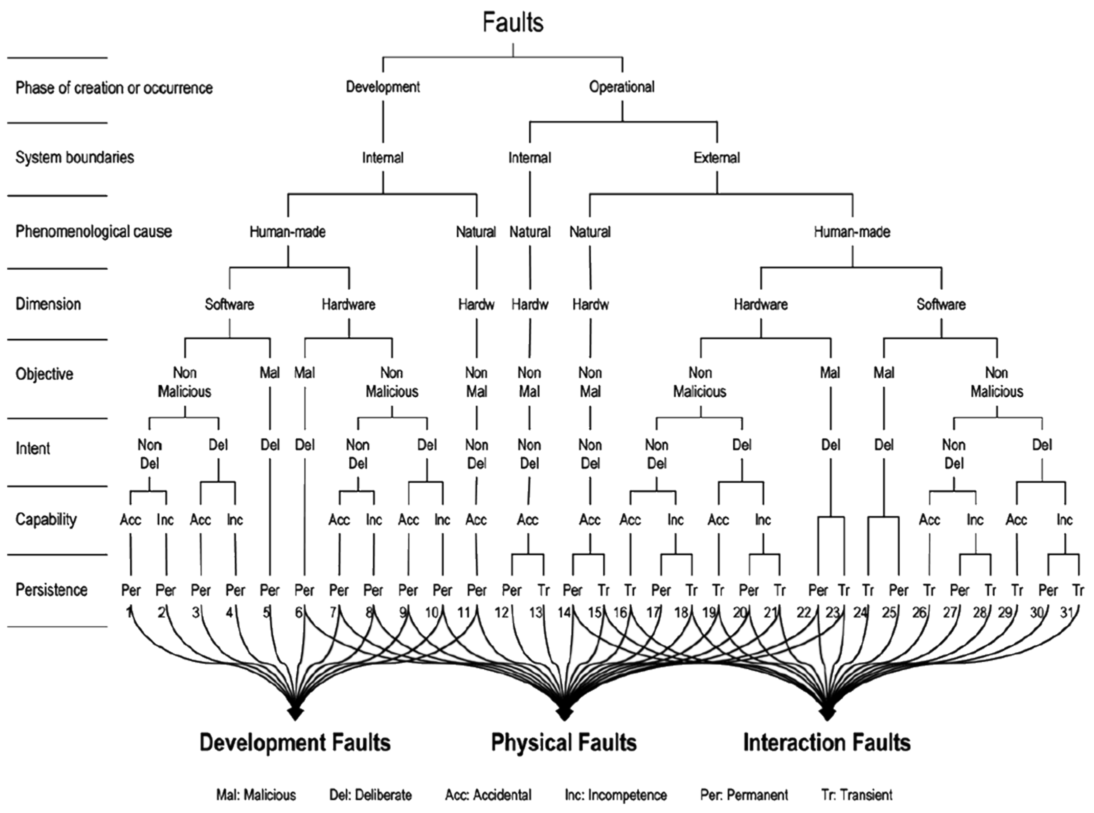


Figure 2.3: Tree of classes of faults (Avizienis et al., 2004).

this failure occurs in a way that can be detected by other systems, it is called a *fail-stop failure*. For this it is assumed that the failed system keeps a stable storage which reflects the last correct service state and can be read by other systems (Poledna, 2007).

2.2 Means

Dependability means are the methods and techniques enabling the development of a dependable system, such as fault prevention, fault tolerance, fault removal and fault forecasting. *Fault prevention* aims to prevent the occurrences or introduction of faults in the system in the first place. It can be achieved by quality control techniques during the specification, implementation and fabrication stages of the design process. *Fault removal* aims to reduce the number and severity of faults that are present in the system. It can be performed during the development phase, as well as during the operational life of a system. During the development phase, fault removal involves verification, diagnosis and correction. During the operational life of the system, it consists of corrective and preventive maintenance. *Fault tolerance* targets the development of systems that still operate correctly in the presence of faults, which is generally achieved by using some kind of redundancy. *Fault forecasting* aims to estimate how many faults are present, possible future occurrences of faults and the impact of the faults on the system. It can be achieved by performing an evaluation of the system behavior with respect to fault occurrences or activation. This evaluation

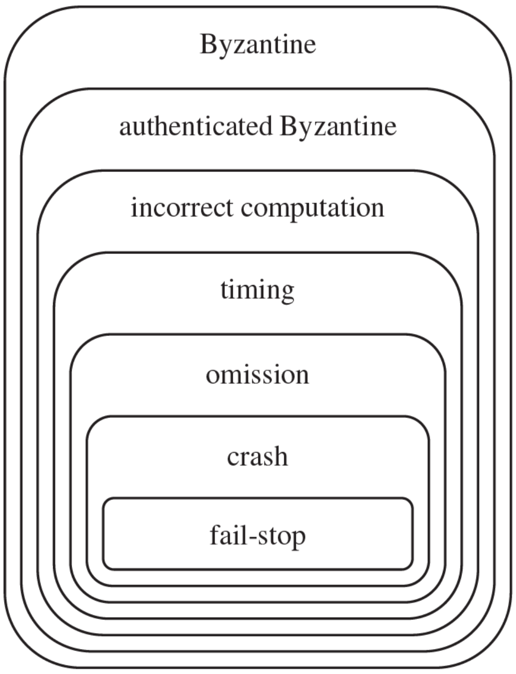


Figure 2.4: Failure modes (Warns, 2010).

can be *qualitative*, which aims to rank the failure modes or event combinations that lead to system failure, or *quantitative*, which aims to evaluate the system in terms of probabilities of how much some dependability attributes are satisfied. This thesis focuses on fault forecasting by quantitative evaluation (Avizienis et al., 2004; Bernardi et al., 2012; Dubrova, 2013).

2.3 Dependability Attributes

Dependability is an integrating concept that measures system properties by several attributes. *Availability* is related to system readiness for correct service. *Reliability* is associated to continuity of provision of correct service. *Safety* regards absence of catastrophic consequences on the user(s) and the environment. *Integrity* is associated to absence of improper system alterations. *Maintainability* is the ability to undergo modifications and repairs. *Confidentiality* is related to absence of unauthorized disclosure of information (Avizienis et al., 2004; Huang et al., 2014). These attributes can vary in number and degree of importance considering the nature of the application (Elghazel et al., 2015) and can be assessed to determine system overall dependability using qualitative or quantitative measures.

In the literature, dependability is commonly addressed by availability and reliability metrics (Macedo et al., 2014; Frühwirth et al., 2015; Dar et al., 2016), by quantitative evaluation, and so these attributes receive more attention in this thesis. Follow, the concepts of availability and reliability are expanded and later, in Section 2.4, we show how to quantitatively assess these attributes. An easy way to determine these attributes is to assume, without loss of generality, that the system behavior evolves through a set of states that can be grouped into two complementary sets: *operational*, corresponding to the correct service, and *nonoperational* (failed), corresponding to the incorrect service, as shown in Figure 2.5. The transition between those states is driven by the respectives failures and repairments of the system.

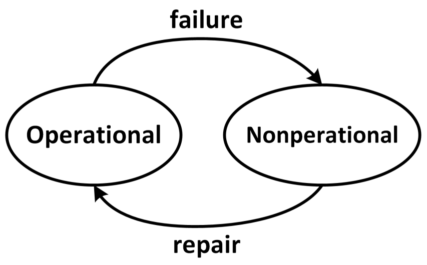


Figure 2.5: Transitions between state of the system.

2.3.1 Reliability

Reliability is the ability of a system or component to perform continuously its required functions under stated conditions for a specified period of time, without interruption, or in another perspective, it is the probability that an item will not fail. That way, the system reliability $R(t)$ can be stated as the probability of any system failure does not occur at the time interval $[0,t[$, which means a continuous presence at the state *operational* in the model of Figure 2.5 (Avizienis et al., 2004; Misra, 2008; Sandborn and Myers, 2008). This is represented by Equation 2.1, being T the random variable referring the occurrence of a system failure.

$$R(t) = Prob(T > t) \tag{2.1}$$

2.3.2 Availability

Availability is the probability of a system or component operating correctly at any given point in time when used under stated conditions, where the total time considered includes operating and repair time (Misra, 2008). In the model of Figure 2.5, this means a presence at the state *operational* in a determined instant of time t.

$$A(t) = Prob(X_t = Operational) \tag{2.2}$$

This can be formalized by Equation 2.2, being X_t the system state at time t. This probability can be computed as the probability that the system will be available at a time t (instantaneous or point availability), as the proportion of time available within a specified interval of time (average up-time availability), or as the limit as $t \to \infty$ of the average up-time availability (steady-state availability) (Sandborn and Myers, 2008). In this thesis we will consider steady-state availability, except when it is clearly distinguished.

2.4 Dependability Modelling and Evaluation

As it was mentioned before, dependability will be addressed in this thesis by two quantitative attributes: availability and reliability. In order to evaluate them, different approaches can be used, commonly categorized as *analytical*, *simulation* and *experimental*, each one presenting an different level of accuracy, cost, duration or comprehensiveness. Some authors also define the *formal* category (Coronato and Testa, 2013), but it is assumed herein that it is equivalent to the analytical approach, since the tools to model systems in a formal way can also be described analytically.

While experimental approaches require the existence of a real system to perform the evaluation at run-time, analytical and simulation approaches are easier to implement because they are based on models. However, the accuracy of their results depends on the accuracy of the values assigned to the model parameters and the considered hypothesis. Moreover, the use of simulation for dependability evaluation brings an important concern: for an accurate estimation of dependability measures, frequent observations of the system-failure event are necessary, which, by definition are rare events. This results in a substantial increase of the simulation time, which could lead to impractical values (Bondavalli et al., 2010; Coronato and Testa, 2013; Cinque et al., 2012; Martins et al., 2015). Therefore, considering this complex scenario for dependability evaluation, in this thesis, we will focus on analytical modelling, because: (a) it can be used during the design phase of the system; (b) it is generally cheaper, since it does not requires an actual system; and (c) it is faster, especially when using effective modelling tools, since does not require several iterations of evaluation to attain a credible result (Gokhale and Trivedi, 1998).

However, it is important to remark that analytical models present some disadvantages, especially the difficulty to capture and reproduce the exact behavior of the system to be modelled. Examples of this behavior are the occurrence of events with constant duration or the occurrence of failures and repairs that are difficult (or even impossible) to model by the stochastic processes underlying those analytical approaches. Generally, these aspects are approximated by the composition of distributions supported by the models (*e.g.*, sum of exponential distributions). Numerical errors are unavoidable in any model solution approach and may derive from many causes, such as discretization of continuous variables, series truncation errors or roundoff errors. More than that, due to largeness and stiffness in the models, sometimes it is necessary to make simplifying approximations by resorting to decomposition techniques, state truncation and fixed-point iteration, which may introduce approximation errors (Trivedi et al., 2010; Trivedi and Bobbio, 2017).

Among the analytical techniques to model systems dependability, we can highlight two categories regarding the structure of models: *combinatorial* and *state space*. Combinatorial methods describe a system based on the static structural relationship between the system components, usually solved by retrieving the structure function of the system, which is a combination of events that leads to the system failure. These methods have flexibility in component lifetime distributions and low computational complexity (Zhao et al., 2018; Manno, 2012).

However, some real-life system presents structures and failure/repair dependencies that violate assumptions necessary for analysis using combinatorial models, such as shared repair, warm/cold spares, imperfect coverage, non-zero switching time, detection and recovery delays, reliability with repair, multiple failure modes or hot swap (Trivedi et al., 2010; Manno, 2012). For these cases, state space oriented models are more suitable to incorporate a more realistic and dynamic system behavior, since they enumerate several states associated to the evolution of the system. A system state is a collection of variables whose values define the state of the system at a given time instant. These models can be described by stochastic processes, which are mathematical models useful for the description of probabilistic phenomenons as a function of a parameter that usually has the meaning of time (Marsan, 1988). The main drawback of a state space-based model is the possible large state space when modelling complex systems, which can represent an exponential growth in the number of components (Gokhale and Trivedi, 1998; Malhotra and Trivedi, 1994; Trivedi et al., 2010; Distefano et al., 2012).

Among the combinatorial techniques to model systems dependability, the most common include Fault Tree Analysis (FTA) and Reliability Block Diagrams (RBDs), while some works focus the state space-based formalisms regarding Markov Chains (MC) and Stochastic Petri Nets (SPN) (Trivedi et al., 2009; Billinton and Allan, 1992). FTA and RBD are very similar graphical formalisms, both based on to describe the system behavior through combination of events of the system, generally in a static way. FTA uses a treelike structure (a Fault Tree) composed by events and logic gates (*e.g.* **AND**, **OR**) to describe the combinations that lead to the system fail. RBDs express the system successful behavior by composing a set of blocks, where each block represents the reliability of an element of the system (Trivedi and Bobbio, 2017; Ahmed et al., 2017; Høyland and Rausand, 1994). Figure 2.6 shows an example of an hypothetical network and its RBD and Fault Tree (FT) models. In this case, the network must delivery information from nodes A, B or C, to node D, and after that to node E.

Regarding the state space-based approaches, SPN appear as a graphical and mathematical modelling formalism to discrete events systems, with the capacity to model dynamic behavior of heterogeneous systems, providing a integrated approach to dependability models (Sanders and Meyer, 2001; Malhotra and Trivedi, 1995). In a SPN the system states are graphically represented (see Figure 2.7) by places (circles) and the system evolution is driven by transitions (rectangles). The transitions are interconnected with places through arcs, that indicate which objects are changed by a certain activity. The firing of a transition is driven by a random variable associated to a distribution function specifing the amount of time that must elapse before the transition can fire (Marsan, 1988; Bause and Kritzinger, 2002). The SPN model shown in Figure 2.7 describes the behavior of a sensor node. It is considered operational (*Sensor_Up* place) in normal conditions, or nonoperational (*Sensor_Down* place) either if a failure occurs or if the sensor is undergoing maintenance.

Another modelling technique of dynamic behavior of a system is Markov Chains, which is suitable to continuous and discrete time systems, and provide the appropriate mathematical formalism to process with non-negligible or non-instantaneous repair time (Meyn and Tweedie, 2009). As shown in Figure 2.8 a MC is graphically represented by a labeled directed graph whose vertices are assigned to the MC states, and whose arcs are labeled with the rate of the exponential distri-

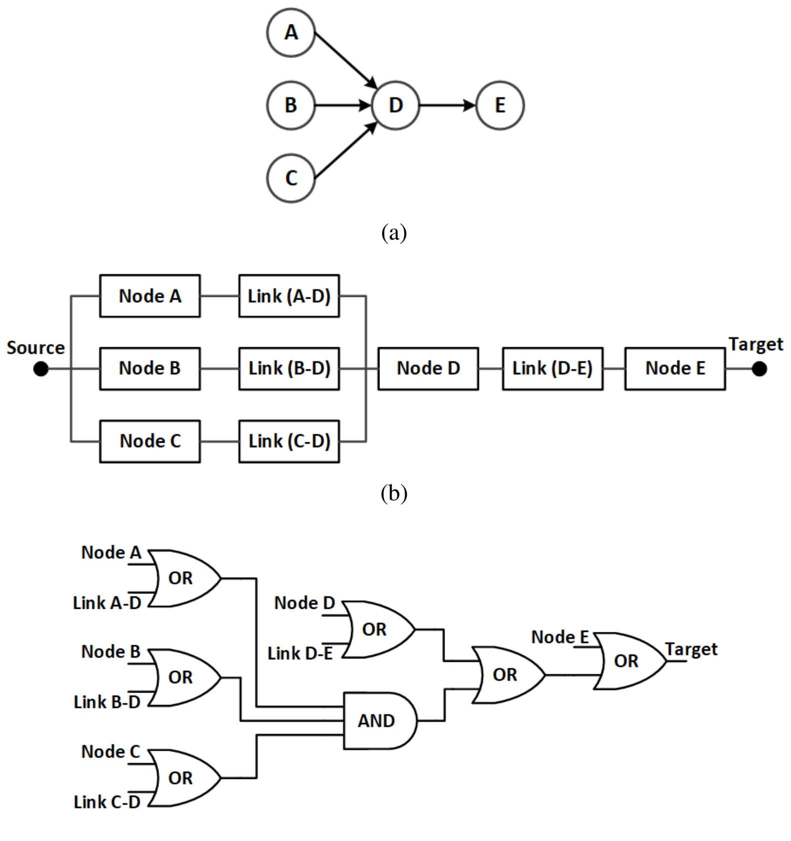


Figure 2.6: Example of (a) a network region, (b) its RBD and (c) its Fault Tree (Dâmaso et al., 2014).

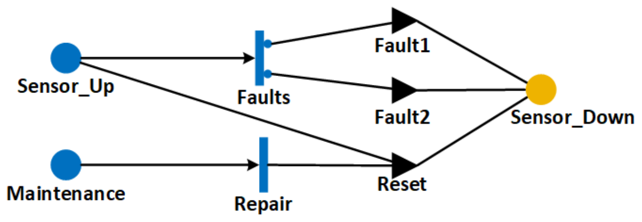


Figure 2.7: SAN model of a sensor node (Maza, 2013).

bution associated with the transition from a state to another (Marsan, 1988). Figure 2.8 shows a Markov Chain of a network component, describing its states as *GOOD* (operational) and *BAD* (failed), being λ the rate of failure occurrence and μ the repairment rate.

As it was mentioned before, the major disadvantage of state space oriented models (SPN and MC) is their exponential growth in the number of states, which turns these techniques prohibitive for large and complex systems. However, it is possible to model large and complex systems exploiting, at the same time, the modelling of static and dynamic system behavior, which circumvents the exponential growth of states. In this case an hierarchical model is developed combining two or more techniques into the construction and solution of a single model. For this, submodels

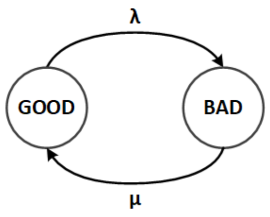


Figure 2.8: Reliability model by Markov Chain of a device (Macedo et al., 2014).

are specified in one formalism and the result of the submodel analysis are embedded in a higher-level model (Trivedi and Bobbio, 2017; Bause and Kritzinger, 2002). In Chapter 5 we discuss hierarchical models based on Fault Trees and Markov Chains for dependability evaluation. Equivalent results could be achieved using RBD instead of FT, but FTs were chosen since they are more intuitive, or using SPN instead of Markov Chains, but we have slightly more experience with MC.

That way, more details related with Fault Trees and Markov Chains are presented later in this chapter. Independently of the system modelling formalism, it is necessary to input into the model the probability distribution of failure (in the reliability case) and the probability distribution of repair (in the availability case). This aspects will be discussed in the next section.

2.4.1 Probability Distributions

Probability distributions describe the random behavior of a system or a component from a set of data. These distributions are used to model random events for which the outcome is uncertain such as the time of failure for a component (O'Connor et al., 2016). In practice, the parameters that are normally associated with reliability and availability evaluation are described by probability distributions, more specifically, *Probability Density Functions* (PDF) or *Cumulative Distribution Functions* (CDF) (Shooman, 2002).

The PDF $f(t)$ is the global rate of occurrence of an random event represented by a unit area function, according to Equation 2.3. In the case of assess the probability of occurrence of a failure, the PDF is also called *failure density function*. Meanwhile, the CDF $F(t)$ is the probability of the random event occurring before the time t, as shown Equation 2.4.

$$\int_{-\infty}^{\infty} f(t)dt = 1 \qquad (2.3)$$

$$F(t) = P(T \leq t) = \int_{-\infty}^{t} f(x)dx \qquad (2.4)$$

The reliability function (or survival function) $R(t)$, defined by the probability of a failure (random event) occurs after the time t can be written as:

$$R(t) = P(T > t) = 1 - F(t) \tag{2.5a}$$

$$R(t) = 1 - \int_{-\infty}^{t} f(x)dx \tag{2.5b}$$

$$R(t) = \int_{t}^{\infty} f(x)dx \tag{2.5c}$$

Reliability can also be defined through the hazard rate $\lambda(t)$, which is the relation between the number of failures per unit time and the number of components exposed to failure. This function denotes a conditional probability that a component fails in a small time interval, given that it has survived from time zero until the beginning of the time interval. The relation between hazard rate and reliability is stated in Equation 2.6 (Billinton and Allan, 1992; Shooman, 2002; O'Connor et al., 2016).

$$\lambda(t) = \frac{f(t)}{R(t)} \tag{2.6}$$

For many systems or components, the hazard rate presents a characteristic shape which is similar to a bathtub curve (see Figure 2.9). When the system is young, the probability of occurrence of a failure is higher (infant mortality), and then quickly decreases until it stabilizes (useful life). As the system/component gets older it increases again (wear out). For electrical/electronic systems it is common to consider that the hazard rate is constant during the useful life period, *i.e.*, $\lambda(t) = \lambda$. In such circumstances, the hazard rate is called *failure rate*. It can be proved that $R(t)$ and $\lambda(t)$ are related according to the following expression (Shooman, 2002):

$$R(t) = \exp\left[-\int_{0}^{t} \lambda(u)du\right] \tag{2.7}$$

If it is considered that $\lambda(t) = \lambda$, then this is equivalent to say that failures occur according to Poisson process with constant rate. In this case, the time interval between the events occurrences is defined by an exponential distribution, with CDF $F(t) = 1 - e^{-\lambda t}$. Therefore, using Equation 2.5a, Equation 2.7 can be rewritten and the reliability function $R(t)$ is stated as in Equation 2.8 (Trivedi and Bobbio, 2017).

$$R(t) = e^{-\lambda t} \tag{2.8}$$

From Equation 2.8 it is possible to state an important property of exponential distributions. Consider that a complex system can be defined as a set of components organized in series, *i.e.*, the system is functioning if and only if all of its components are functioning. This is a reasonable assumption for non-redundant systems and leads to realistic lifetime models (Rausand and

Høyland, 2004; Høyland and Rausand, 1994). Thus, the failure rate of a complex system is simply the sum of the individual failure rates of its components, as shown in Equation 2.9, being $\lambda_e = \lambda_1 + \cdots + \lambda_n$. This means that the reliability of a series system can be evaluated very rapidly, since the components can be described by exponential distributions (Billinton and Allan, 1992).

$$R(t) = R_1(t) \times \cdots \times R_n(t) \tag{2.9a}$$

$$R(t) = e^{-\lambda_1 t} \times \cdots \times e^{-\lambda_1 t} \tag{2.9b}$$

$$R(t) = e^{-(\lambda_1 + \cdots + \lambda_n)t} = e^{-\lambda_e t} \tag{2.9c}$$

$$\lambda_e = \lambda_1 + \cdots + \lambda_n \tag{2.9d}$$

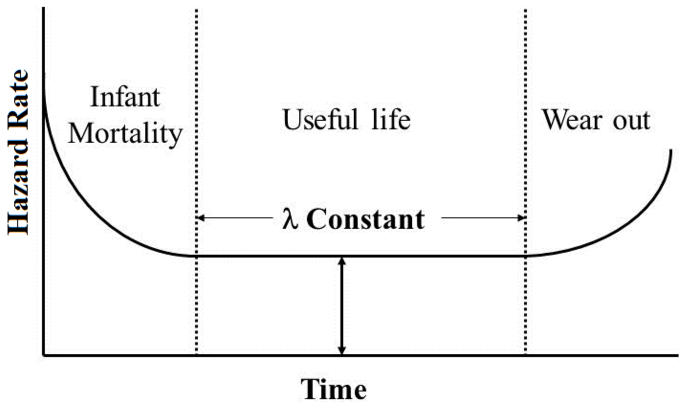


Figure 2.9: Bathtub curve.

Regarding to availability distribution function, and following a reasoning similar to the reliability, it can be considered that, once a system fails, it can be repaired. We assume here that the repair rate μ is constant. Thereby, considering a system behaving as shown in Figure 2.8 (operational or failed), according to Høyland and Rausand (1994), instantaneous availability $A(t)$ can be expressed as Equation 2.10, representing the probability of the system be found available at time t. On the other, average availability $A_m(t)$ assess the percentage of time that the system has been operational in the interval $(0,t]$, as stated by Equation 2.11. Finally, the steady-state availability A_∞ can be found in Equation 2.12, computed from the average availability when $t \to \infty$, representing the asymptotic availability of the system. These same results can be achieved by Markov Chains, as shown in detail in Section 2.4.2. Notice that, if $\mu = 0$, *i.e.*, if it is assumed the system cannot be repaired, then, from Equations 2.10 and 2.8, $A(t) = R(t)$, which means that the concept of availability becomes identical to the reliability definition.

$$A(t) = \frac{\mu}{\mu + \lambda} + \frac{\lambda}{\lambda + \mu} e^{-(\lambda + \mu)t} \tag{2.10}$$

$$A_m(t) = \frac{1}{t} \int_0^t A(\tau)\, d\tau \tag{2.11}$$

$$A_\infty = \frac{\mu}{\mu + \lambda} \tag{2.12}$$

2.4.2 Markov Chains

As mentioned before, a Markov Chain is a state space-based formalism used to model stochastic systems. That means that a set of random variables X_i can assume any value from a finite or countable state space S, which describe the system or component behavior at an specific instant of time t. The system evolution between states are described by non-negative transition rates corresponding to exponential distributions, which results from a stochastic Poisson process (the interval between occurrences is modelled by an exponential distribution) (Rausand and Høyland, 2004; Meyn and Tweedie, 2009).

Formally, a stochastic process $\{X(t) : t \geq 0\}$ is a continuous time Markov chain if, for all states i and j, and for all times $t \geq 0$ and $s \geq 0$, the Markovian property stated in Equation 2.13 remains true. This property is the main characteristic of a MC, which defines that the future behavior of a stochastic process towards to a state j (at some time $t+s$) depends only on the present situation in state i (at time s), and not on the past history of the system. This makes a MC a process without any memory of the trajectory i_r followed to reach the present state (Marsan, 1988; Bujorianu, 2012; Trivedi and Bobbio, 2017).

$$P\{X(t+s) = j \,|X(s) = i, X(r) = i_r, r < s\} = P\{X(t+s) = j \,|X(s) = i\} \tag{2.13}$$

The Figure 2.10 present the Markov Chain model of a repairable component. The state 0 is related to the correct functioning of the component, while the state 1 is related to the failed behavior of the component. So, if the Markov Chain is found in state 0, the component is operable, and if the Markov Chain is found in state 1, the component is failed. The transition rate of the Markov Chain from state 0 to state 1 is equals to the failure rate λ, which means the frequency that a failure occurs. In the same way, the Markov Chain change from state 1 to state 0 with the repair rate μ, which means the frequency that the component is repaired.

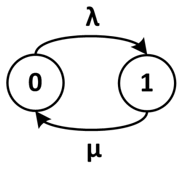


Figure 2.10: Markov chain of a repairable component.

Since the reliability is a special case of availability, that the system is not repairable (repair rate $\mu = 0$), we will focus in show how to evaluate availability. For this purpose, it is necessary to compute the probabilities $P_0(t)$ and $P_1(t)$. $P_0(t)$ is the probability that the component is in state 0, *i.e*, operable at time t, and $P_1(t)$ is the probability that the component is in state 1, *i.e*, failed at time t. The failure density of a component with a constant hazard rate λ, based on Equation 2.5b and Equation 2.8 is given as:

$$f(t) = \frac{-dR(t)}{dt} \tag{2.14a}$$

$$f(t) = \lambda e^{-\lambda t} \tag{2.14b}$$

The availability is the percentage of time that a system is actually operating with relation to the total time it should be operating, or in another way, it is the probability of finding the system in the operating state at some time into the future, *i.e.*, $A(t) = P_0(t)$.

In order to analytically evaluate $P_0(t)$, lets consider an incremental interval of time dt which is made sufficiently small so that the probability of two or more events occurring during this increment of time is negligible. So, the probability of the component being in operating state after this time interval, *i.e.*, the probability of the component being in state 0 at time $t+dt$ is the probability of being operable at time t **AND** does not fail in time dt, **PLUS** the probability of being failed at time t **AND** be repaired in time dt. Formally,

$$P_0(t+dt) = P_0(t)(1-\lambda dt) + P_1(t)(\mu dt) \tag{2.15}$$

In the same way,

$$P_1(t+dt) = P_1(t)(1-\mu dt) + P_0(t)(\lambda dt) \tag{2.16}$$

From Equation 2.15,

$$\frac{P_0(t+dt)-P_0(t)}{dt} = -\lambda P_0(t) + \mu P_1(t) \tag{2.17a}$$

$$\left.\frac{P_0(t+dt)-P_0(t)}{dt}\right|_{dt\to 0} = \frac{dP_0(t)}{dt} = P'_0(t) \tag{2.17b}$$

$$P'_0(t) = -\lambda P_0(t) + \mu P_1(t) \tag{2.17c}$$

In the same way,

$$P'_1(t) = \lambda P_0(t) - \mu P_1(t) \tag{2.18}$$

The equations 2.17c and 2.18 can be expressed in matrix form as

$$\begin{bmatrix} P'_0(t) & P'_1(t) \end{bmatrix} = \begin{bmatrix} P_0(t) & P_1(t) \end{bmatrix} \begin{bmatrix} -\lambda & \lambda \\ \mu & -\mu \end{bmatrix} \tag{2.19}$$

The Equations 2.17c and 2.18 or the Equation 2.19 compose a linear differential equations system with constants coefficients. There are several ways to solve that system, such as Laplace transforms. If we consider that the system initiates its execution in operable state, then the initial conditions are $P_0(0) = 1$ $P_1(0) = 0$, which leads to the following solution:

$$P_0(t) = \frac{\mu}{\mu+\lambda} + \frac{\lambda}{\lambda+\mu} e^{-(\lambda+\mu)t} \tag{2.20a}$$

$$P_1(t) = \frac{\lambda}{\mu+\lambda} + \frac{\lambda}{\lambda+\mu} e^{-(\lambda+\mu)t} \tag{2.20b}$$

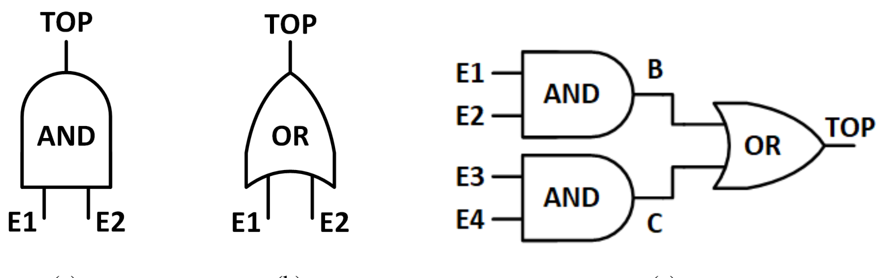
As mentioned before, $A(t) = P_0(t)$, which achieves the same result of Equation 2.10. Also, if it consider a component without repair, $\mu = 0$, and the availability of Equation 2.20a is identical to the reliability, as shown in Equation 2.8.

2.4.3 Fault Trees

A Fault Tree is an intuitive graphical formalism used to evaluate the failure probability of a system. It describes the combination of events that lead to system failures in a treelike structure composed by events and logic gates, as shown in Figure 2.11(c). The system failure is called the TOP event of the fault tree. That way, evaluating a Fault Tree means to determine the probability of the TOP event occurs at time t, which is the system *unreliability* stated in Equation 2.21 (Rausand and Høyland, 2004; Høyland and Rausand, 1994; Ahmad et al., 2016).

$$Q_0(t) = 1 - R(t) \tag{2.21}$$

A Fault Tree is composed by combination of several types of gates, being the most common the **AND**-gate and the **OR**-gate, as shown in Figures 2.11(a) and 2.11(b), respectively. The inputs of a gate are called *events*. An event can be associated to the failure probability or the probability distribution function of a subsystem, as the events B and C in Figure 2.11(c). An event can be also related to a basic system component, appearing in the lowest level in a fault tree branch, terminating this branch, as the events E_1, E_2, E_3 and E_4 in Figure 2.11. These events require no further development of failure causes, being called *basic events*. If a basic event occurs two or more times in an FT, it is called a *repeated event*.



(a) (b) (c)

Figure 2.11: Fault Tree gates (Rausand and Høyland, 2004)

The **AND**-gate indicates that the output event TOP occurs only when all the input events E_i occur at the same time. In this case, the probability of the TOP event can be defined according to Equation 2.22, *i.e.*, in Figure 2.11(a) the probability of the system fails, $P(TOP)$, is equals to the product of the probability of E_1 by the probability of E_2 given $E1$. Considering that the events E_i are independent, then $P(E_2|E_1) = P(E_2)$. Additionally, since the system reliability $R(t)$ can be stated as the probability of a system does not fail at the time interval $[0,t[$, the Equation 2.22 can be rewritten and the system unreliability $Q_0(t)$ (probability of the system fails over the time) can be expressed by the product of the probabilities of all components or subsystems (input events) fail, being $R_i(t)$ the reliability of the event E_i, as shown in Equation 2.23.

$$P(TOP) = P(E_1 \cap E_2) = P(E_1) \times P(E_2|E_1) \tag{2.22}$$

$$Q_0(t) = \prod_{i=1}^{n} (1 - R_i(t)) \tag{2.23}$$

The **OR**-gate indicates that the output event TOP occurs if any of the input events E_i occur. That means that the probability of the TOP event in Figure 2.11(b) is the probability of at least one component or subsystem fails. In other words, as shown in Equation 2.24, the probability of the TOP event can be computed by the sum of the probabilities of E_1 and E_2, minus the probability of E_1 and E_2 occur at the same time, which is their intersection. If the events E_1 and E_2 are mutually exclusive, then $P(E_1 \cap E_2) = 0$. Thus, the Equation 2.24 can be rewritten and the system unreliability $Q_0(t)$ can be defined as the complement of the product of the probabilities of all components or subsystems do not fail, as shown in Equation 2.25. If it is considered that the input events E_i are independent and present low probability of occurrence (say $P(E_i) < 10^{-1}$, which are the case of failure probabilities), this means that $P(E_1) + P(E_2) \gg P(E_1 \cap E_2)$, then $P(E_1 \cap E_2)$ can be approximated to zero (Rausand and Høyland, 2004; Høyland and Rausand, 1994; Vesely et al., 1981).

$$P(TOP) = P(E_1 \cup E_2) = P(E_1) + P(E_2) - P(E_1 \cap E_2) \tag{2.24}$$

$$Q_0(t) = 1 - \prod_{i=1}^{n} R_i(t) \tag{2.25}$$

Equations 2.23 and 2.25 can be combined according to the Fault Tree structure to determine the probability of the TOP event occurs. For instance, following the structure of Figure 2.11(c), the TOP event probability of occurrence $Q_0(t)$ is the probability of occurrence of events B ($Q_B(t)$) **OR** C ($Q_B(t)$), as defined by Equation 2.26a. As well, the probabilities of B and C are defined by Equations 2.26b and 2.26c as the probabilities of occurrence of events E_1 **AND** E_2, and E_3 **AND**

E_4, respectively.

$$Q_0(t) = 1 - R(t) = 1 - (R_B(t) \times R_C(t)) \tag{2.26a}$$

$$Q_B(t) = 1 - R_B(t) = (1 - R_1(t)) \times (1 - R_2(t)) \tag{2.26b}$$

$$Q_C(t) = 1 - R_C(t) = (1 - R_3(t)) \times (1 - R_4(t)) \tag{2.26c}$$

All these equations are valid only when the FT does not contain any repeated event. In the presence of these events, the FT must by evaluated by other methods, such as inclusion-exclusion principle, sum of disjoint products, factorization and direct/indirect recursive methods (Limnios, 2010). In this thesis we applied the sum of disjoint products (SDP), since it is efficient and can be easily automated. The SDP defines a system failure condition (TOP event of a FT) through a function $\phi(\mathbf{x})$ based on the failure probability of each system component i.

Consider a system with n components and a state vector, $\mathbf{x} = (\mathrm{x}_1, \mathrm{x}_2, \ldots, \mathrm{x}_n)$. Each element x_i is a Boolean variable representing the state of component i, as shown in Equation 2.27. The failure information of each state x_i can be jointly analyzed in order to determine the system failure condition based on Equation 2.28. That way, if $\phi(\mathbf{x}) = 1$ for a determined configuration of $\mathbf{x}$, then the elements $\mathrm{x}_i = 1$ in this configuration define a subset called *cut set*. This means that, if all elements of a cut set K fails simultaneously, the system will also fail (Limnios, 2010).

$$\mathrm{x}_i = \begin{cases} 1, & \text{if the component } i \text{ has failed} \\ 0, & \text{if the component } i \text{ has not failed} \end{cases} \tag{2.27}$$

$$\phi(\mathbf{x}) = \begin{cases} 1, & \text{if the system has failed} \\ 0, & \text{if the system has not failed} \end{cases} \tag{2.28}$$

Cut sets can be used to redefine $\phi(\mathbf{x})$ according to 2.29, being K_i the i-th cut set. Finally, $\phi(\mathbf{x})$ can be transformed in a sum of disjoint products, as shown in Equation 2.30, being $\overline{K_i}$ the complement of the i-th cut set. Since the terms are pairwise disjoint, the probability of the TOP event can be obtained as the sum of the probabilities of the individual terms. The last step to compute probability of a system failure is to replace each event i, in the respective cut set, by its reliability function $R_i(t)$. After that, the reliability of the system $R(t)$ can be easily computed using simple probability laws (*i.e.*, probability of union and intersection of events) (Limnios, 2010).

$$\phi(\mathbf{x}) = \bigcup_{i=1}^{m} K_i \tag{2.29}$$

$$\phi(\mathbf{x}) = \bigcup_{i=1}^{m} K_i = K_1 \cup \overline{K_1} K_2 \cup \ldots \cup \overline{K_1} \ldots \overline{K_{m-1}} K_m \tag{2.30}$$

2.5 Conclusion

In this chapter, dependability concepts were presented, from basic definitions to evaluation techniques. First it was discussed the threats to the system dependability, specially faults and failures, since they are the probable cause and the manifestation of a system malfunction, respectively. The means to deal with these threats were also presented, with a highlight to quantitative fault forecasting, which is the core of this thesis. Fault forecasting can be performed related to several dependability attributes, using different approaches. These attributes were enumerated, inclusive reliability and availability, and some analytical approaches (either combinatorial or state space-based) were discussed in order to determine the probability of the system fails or the probability of the system to be found operational. Among these approaches, we focused on Markov chains and fault trees, presenting more details related to theses formalisms, since they are applied later in the proposed methodology.

Chapter 3

Wireless Visual Sensor Networks

In this chapter it is presented the concepts related to Wireless Visual Sensor Networks, as well as the mathematical modelling of structural elements of such networks, like nodes' positioning and orientation, coverage, monitoring field, etc. We also discuss different types of visual coverage as well as some events that can trigger a visual failure. The dependability evaluation proposed in this thesis is based on this type of failure.

Similarly to an ordinary wireless sensor networks, a wireless visual sensor network can be seen as a distributed system formed by sensor nodes spread into an interest area, aiming the extraction of relevant information from that area. In the case of a WVSN, some nodes have an integrated a camera that allows them to collect, process and deliver visual information, in format of image or video (Soro and Heinzelman, 2009; Charfi et al., 2009). These visual networks achieve a deeper perception of the monitored environment, leveraging several applications, such as smart cities (Giyenko and Cho, 2016), smart street lighting (Kumar et al., 2016), smart homes (Tanwar et al., 2017), smart grids (Toth and Gilpin-Jackson, 2010), traffic and pedestrian control (Shah et al., 2016), living assistance (Pirsiavash and Ramanan, 2012), driving assistance (Westhofen et al., 2012), waste collection (Medvedev et al., 2015), surveillance (Shao et al., 2017). For example, a street lighting triggered by an identified human shape can avoid a wrong activation from motion sensing of birds or other animals. A traffic light control system can better manage its time intervals with a image of how many people or cars are occupying the side walk or the road, respectively. An intelligent transportation system can easily use images to detect a car accident and then call the proper authorities. Besides the obvious usage of visual data, like surveillance, face detection, intrusion detection, which can be solved using WVSN.

Formally, consider a wireless visual sensor network being defined as $wvsn = \{\{snk\} \cup VS \cup SS\}$, where *snk* is the sink node (base station), the common destination of all information collected by nodes in the network, $VS = \{vs_i | i = 1, \ldots, n\}$ a set of visual sensor nodes and $SS = \{ss_j | j = 1, \ldots, m\}$ is a set of scalar sensor nodes. Visual nodes are equipped with cameras and are responsible for monitoring the area $\mathbb{A}$ (monitoring field), gathering visual information (videos or photos), while scalar nodes are unable to gather visual information, but can be used as routers (relaying nodes) to support the delivery of visual information to the sink node, re-transmitting messages (Costa et al., 2014a). Figure 3.1 shows a WVSN with visual nodes vs_1, vs_2 and vs_3, that can reach the sink by assistance of the scalar nodes ss_1, ss_2, ss_3 and ss_4.

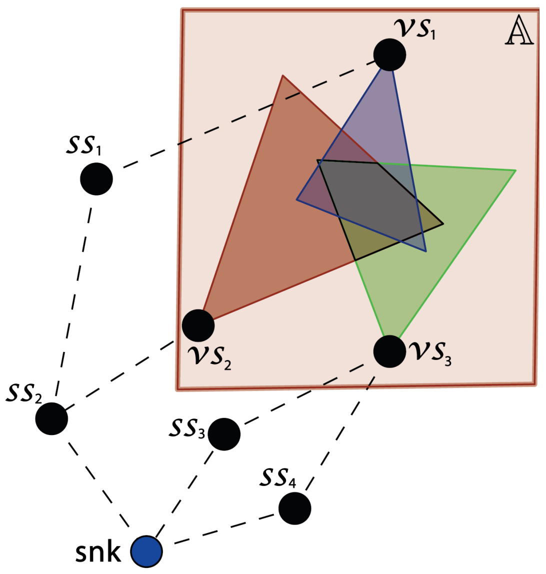


Figure 3.1: Area monitoring by visual sensors (Costa et al., 2014a).

Visual sensors are deployed over a two-dimensional monitoring field $\mathbb{A}$, assuming that those sensors may be randomly or deterministically positioned. Each sensor vs_i is located at coordinate (x_i, y_i) within $\mathbb{A}$ or nearby, and it is expected to be equipped with a low-power camera, with a viewing angle θ_{vs_i} and an orientation α_{vs_i} (Costa et al., 2014a). The embedded camera also defines a sensing radius R_s representing an approximation of the camera's Depth of Field (DoF), which is the region between the nearest and farthest point that can be sharply sensed (Almalkawi et al., 2010; Costa and Duran-Faundez, 2016). For simplification and in order to make this problem tractable and computationally feasible for the Wireless Sensor Network (WSN) context, sensor nodes should be assumed as having limited hardware and low-power requirements. A camera has a Field of View (FoV), which is the maximum region visible from a camera, represented by a sector-like visible region emanating from the camera (Soro and Heinzelman, 2005). In this thesis, the FoV of any visual sensor is defined as the area of an isosceles triangle composed of three vertices, A, B and C, being (A_x, A_y) the Cartesian coordinates of the camera, represented in Figure 3.2 by the red circle. The coordinates of vertices B and C can be computed by Equation 3.1 and the FoV of any visual sensor vs can be computed using trigonometry, as expressed in Equation 3.2 (Costa et al., 2014a,b).

$$\begin{aligned}
B_x &= A_x + R_s \cos(\alpha_{vs}) \\
B_y &= A_y + R_s \sin(\alpha_{vs}) \\
C_x &= A_x + R_s \cos((\alpha_{vs} + \theta_{vs}) \bmod 2\pi) \\
C_y &= A_y + R_s \sin((\alpha_{vs} + \theta_{vs}) \bmod 2\pi)
\end{aligned} \tag{3.1}$$

$$FoV_{vs} = \frac{R_s^2 \cdot \sin(\theta_{vs})}{2} \tag{3.2}$$

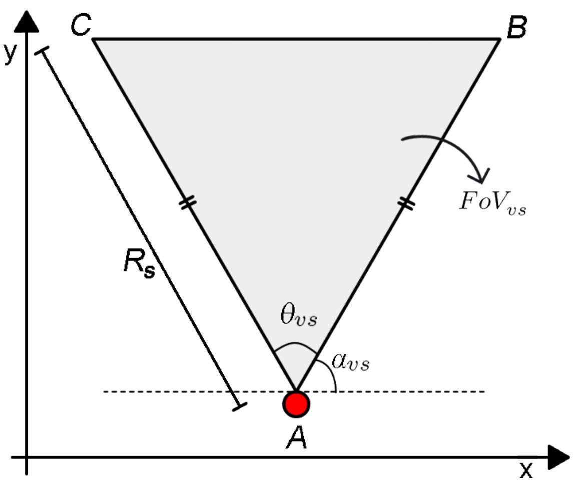


Figure 3.2: Field of View of a visual sensor (Costa and Duran-Faundez, 2016).

The monitoring field $\mathbb{A}$ may be composed of one or more *Monitoring Areas* (MA). A MA is the area of interest of a visual application and only visual information from this sub-region is relevant to the considered WVSN. Each MA is described as a rectangle defined by its origins ($x1$, $y1$), a width w, a height h and a rotation angle β, as shown in Figure 3.3. The coordinates of the other vertices of a MA (vertices 2, 3 and 4) can be computed as presented in Equation 3.3. The coverage of a MA can be performed oriented to objectives, as discussed follow.

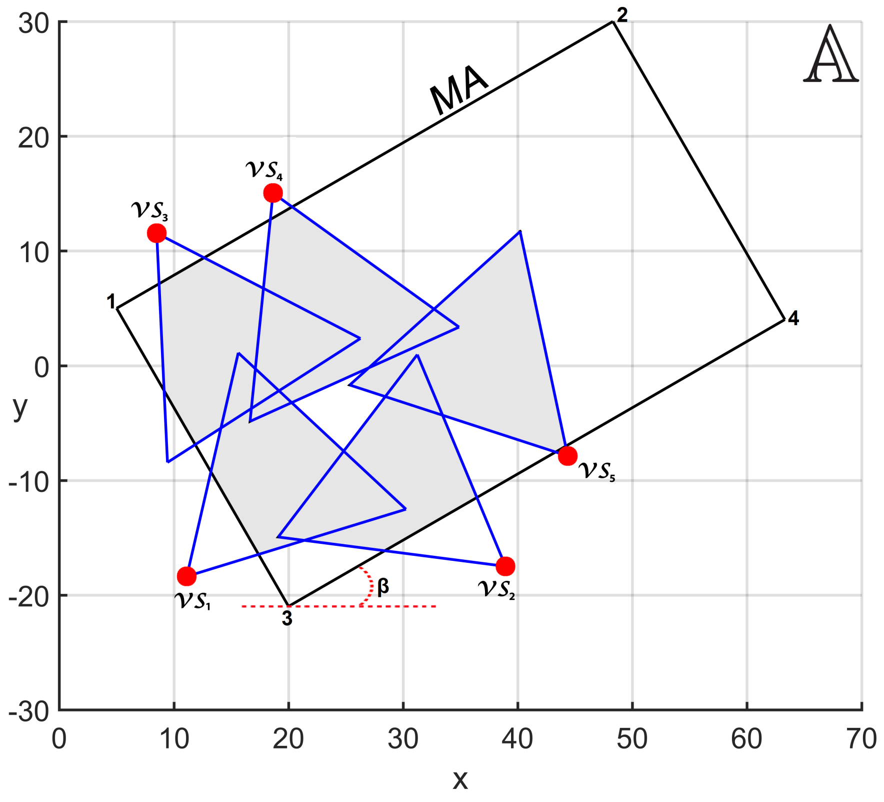


Figure 3.3: Area coverage of a monitoring area MA.

$$\begin{aligned}
x2 &= x1 + w.\cos(\beta), & y2 &= y1 + w.\sin(\beta) \\
x3 &= x1 + h.\sin(\beta), & y3 &= y1 - h.\cos(\beta) \\
x4 &= x1 + h.\sin(\beta) + w.\cos(\beta), & y4 &= y1 - h.\cos(\beta) + w.\sin(\beta)
\end{aligned} \tag{3.3}$$

3.1 Visual Coverage

In Wireless Visual Sensor Networks, visual coverage is an important quantifiable property of camera-based networks, describing from a pragmatic standpoint what the system can see, that is, what visual data is physically capable of collecting, and thus informing the most fundamental requirement of any computer vision task (Mavrinac and Chen, 2013; Jia et al., 2019). Visual coverage can present different characteristics and requirements according to the objective of coverage. The most common objectives are: *area coverage*, *target coverage* or *barrier coverage*. *Area coverage* is related to monitoring of one or more areas of the monitored field. The *target coverage* approach is focused on monitoring of a set of targets. At last, the *barrier coverage* monitors a conceptual, long and narrow barrier belt area of sensors, aiming the detection of intruders that attempt to cross the deployed region (Costa and Guedes, 2010; Costa et al., 2014a; Si et al., 2017). In this thesis we consider the area coverage problem with a target coverage-based approximation.

3.1.1 Target Coverage

In a WVSN performing target coverage, each target is required to be covered by a certain number of cameras. The goal is to achieve the maximum coverage of targets, considering the available number of cameras and cost constraints, such as energy consumption (Hanoun et al., 2016). In general, a target is any moving or static element (person, animal, object) that can be viewed by a camera and that may have different formats and sizes. For simplification purposes, all targets can be modelled equally, being considered shaped as a polygon or a circle. A target is said covered by a visual sensor vs if it is entirely within the region marked by the sensor's field of view FoV_{vs}. A common approximation in literature is considering the target modelled as a point, which can be one of its vertices or its centroid. To make coverage solutions more realistic, multiple points can be used to represent bigger sized targets. (Neishaboori et al., 2014; Zannat et al., 2016; Rangel et al., 2018; Zhang et al., 2019).

In order to verify whether a target is covered by a visual sensor node, let's consider the target is represented by the point P, with Cartesian coordinates (xc, yc), as shown in Figure 3.4. It is necessary to compute the area of the three triangles when selecting P as an inner vertex for sensor's FoV $\triangle ABC$, according to Equation 3.4. Then, the equality in Equation 3.5 must be true if $P \in ABC$, *i.e.*, the visual sensor covers the target. If the equality is not fulfilled, the target P is outside of the sensor's FoV, which means that the visual sensor does not cover this target (Pure and Durrani, 2015; Costa et al., 2017b).

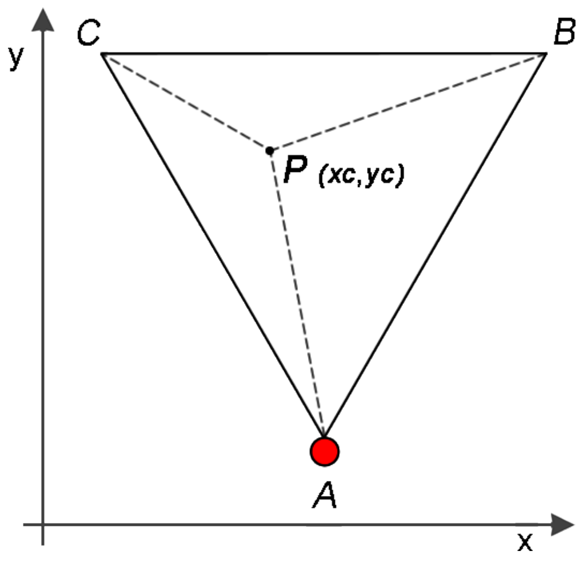


Figure 3.4: Scenario for target coverage verification.

$$\begin{aligned}\triangle APB &= \left| A_x.(B_y - yc) + B_x.(yc - A_y) + xc.(A_y - B_y) \right| \big/ 2 \\ \triangle APC &= \left| A_x.(yc - C_y) + xc.(C_y - A_y) + C_x.(A_y - yc) \right| \big/ 2 \\ \triangle BPC &= \left| xc.(B_y - C_y) + B_x.(C_y - yc) + C_x.(yc - B_y) \right| \big/ 2\end{aligned} \tag{3.4}$$

$$\triangle APB + \triangle APC + \triangle BPC = \triangle ABC \tag{3.5}$$

3.1.2 Area Coverage

In order to perform area coverage, a WVSN must compute the total area sensed by all visual sensor nodes together. Figure 3.5(a) shows the monitored area $\mathbb{A}$ and the FoV of four visual nodes. The coverage area CA is the sum of all FoV, considering properly the overlapped area, as shown in Figure 3.5(b). In this case, the coverage area can be computed according to the Inclusion–Exclusion Principle, which is a counting technique from combinatorial mathematics. That principle computes the number of objects in a union of sets, for the most general of circumstances in which the sets are free to overlap without restriction (Brualdi, 1977; Gross, 2008). It is stated in Theorem 3.1, and so the coverage area can be computed according to Definition 3.2.

Theorem 3.1 (Inclusion–Exclusion Principle)**.** *Suppose $n \in \mathbb{N}$ and A_i is a finite set for $1 \leq i \leq n$. It follows that (Brualdi, 1977; Andreescu and Feng, 2003; Gross, 2008)*

$$\left| \bigcup_{i=1}^{n} A_i \right| = \sum_{\emptyset \neq Q \subseteq \{1,2,\ldots,n\}} (-1)^{|Q|-1} \cdot \left| \bigcap_{q \in Q} A_q \right| \tag{3.6}$$

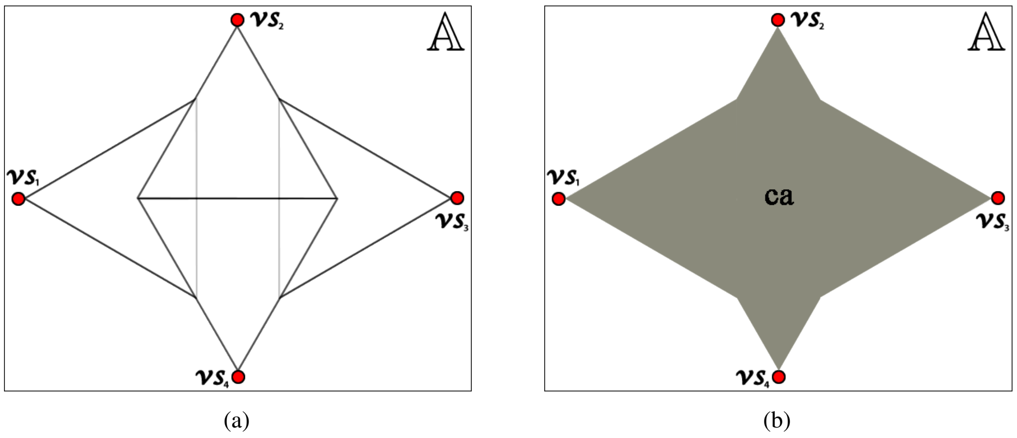


Figure 3.5: (a) Monitored area $\mathbb{A}$ and (b) the coverage area $ca = CA(\{1,2,3,4\})$.

Proof. See (Brualdi, 1977). □

If it is considered the monitoring field $\mathbb{A}$ as a set of points and the cardinality of that set as the area of that region, then the Inclusion-Exclusion Principle can be adjusted and the resulting coverage area can be computed according to Definition 3.2.

Definition 3.2 (Coverage Area)**.** Let $\mathbb{A}$ to be a monitoring field, $VS = \{vs_i | i = 1, \ldots, n\}$ a set of visual nodes covering $\mathbb{A}$ and $Area(p)$ the area of the polygon p, which defines a covered region. The coverage area of VS is defined as:

$$CA(VS) = \sum_{\emptyset \neq Q \subseteq VS} (-1)^{|Q|-1} \cdot Area\left(\bigcap_{vs_q \in Q} vs_q \cap \mathbb{A} \right) \tag{3.7}$$

Since the monitored area is a continuous space, it is hard to determine which area has been covered by each visual sensors. Discrete solutions such as grid-based coverage approach have been proposed to approximate area coverage for WVSN (Wang and Wang, 2019). Thereby, a MA can be divided into smaller regions, called *Monitoring Blocks* (MB), each one defined as a rectangle represented by its origins ($x1s$, $y1s$), a width ws, a height hs, a centroid (xc, yc) and the same rotation angle β of the MA, as shown in Figure 3.6 (Costa et al., 2017a). The coordinates of the other vertices of a MB (vertices $2s$, $3s$ and $4s$) can be computed similarly to Equation (3.3).

Thus, a MA is composed of a grid of MB regions with size $M \times N$. In this case, the area coverage problem can be indirectly approached as several target coverage problems, where each point (xc, yc) is a target with infinitesimal size. This is an important abstraction aimed at higher efficiency, while keeping the computational cost low. We discuss the efficiency and accuracy of this approximation in Chapter 5.

A single monitoring block mb is considered to be covered by a visual sensor node vs if its center (xc, yc) is within the polygon area of FoV_{vs}. In this configuration, the area of a mb ($ws \times hs$) is accounted to the total coverage area. A covered MB is represented by notation $mb \in FoV_{vs}$. This

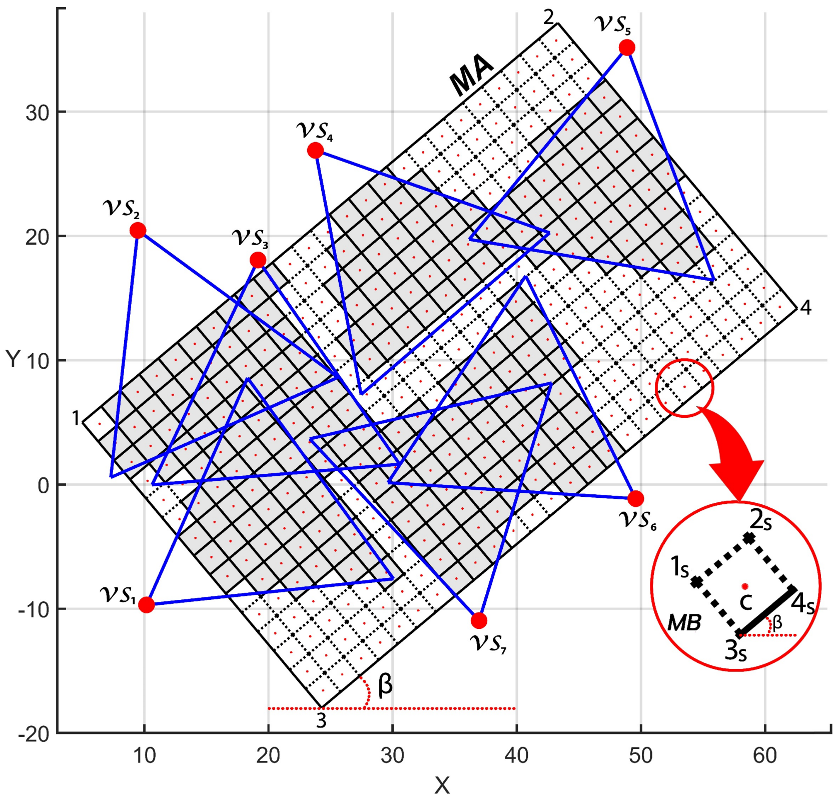


Figure 3.6: Monitoring area being covered by visual sensor nodes vs_i, $i = 1, \ldots, 7$.

definition can be extended for a set of visual sensors VS according to Equation (3.8). The total area covered by VS is computed according to Equation (3.9), where $mb_{k,l}$ is a monitoring block mb at the position (k,l) of the grid.

$$cover\,(mb, VS) = \begin{cases} 1, & \text{if } \exists vs \in VS \mid mb \in FoV_{vs} \\ 0, & \text{otherwise} \end{cases} \tag{3.8}$$

$$CA_{mb}\,(VS) = ws \cdot hs \cdot \sum_{k=1}^{M} \sum_{l=1}^{N} cover\,(mb_{k,l}, VS). \tag{3.9}$$

The computed visual coverage is used as metric to identify whether a application fulfills its requirements. If not, a coverage failure occurred.

3.2 Visual Coverage Failures

The main task of a Wireless Sensor Network is to gather information from the deployed environment and to deliver this data to be properly handled by the application. In the special case of a Wireless Visual Sensor Network, sensor nodes are equipped with cameras that gather and deliver visual information. If this service cannot be provided, an application (or system) failure occurs. This can be a consequence of failure in network elements, such as the hardware of a node or a communication failure. For instance, if a sensor node crashes or if the link between nodes goes down, the network is not able to gather and deliver the expected data. Particularly in WVSN, another issue rises: the visual coverage failures (VCF). Those failures occur when the gathered visual data does not fulfill the application requirements.

Several situations can trigger a VCF. The most common is a inadequate deployment, when sensor nodes are positioned and oriented in a configuration that, even if the network is fully operational, they cannot cover all expected targets, area or barrier belt. Other scenario that can generate a VCF is when obstacles occlude the cameras' FoV. In this case, the amount of visual data that a camera can retrieve is decreased and can not be enough for the application purposes. The quality of coverage can also be considered as VCF trigger. This is related to the sharpness and definition of images, which can be affected by distance of monitoring, perspective angle, luminosity (absence or excess) or even weather conditions (in case of outdoor monitoring) like fog, rain, snow and dust.

In all these situations, the application may not be able to extract the expected information from gathered visual data, being unable to deliver its expected service and, consequently, affecting the system dependability. Therefore, the evaluation methods applied to a system must encompass models to detect visual coverage failures in order to perform an adequate dependability assessment. This means that it is necessary to detail every possibility to the network do not be able to deliver the required visual information to the sink node. All these aspects are discussed in Chapters 5 and 6.

3.3 Conclusion

This chapter presented the visual elements that make up wireless visual sensor networks considered in this thesis. These elements are sensor nodes (including sink, scalar and visual nodes), cameras in the visual nodes, regions of interest to be covered by the network, types of coverage and visual failures that may occur in a WVSN. These elements were described and the mathematical models of their structure were provided in order to be used on the description of the dependability evaluation methodology.

Chapter 4

Related Work

This chapter overviews the state of the art in dependability of wireless visual sensors networks. First, dependability evaluation is investigated related to general wireless sensor networks, in order to get a starting point and comparison basis for potential contributions in visual networks. Then, the same investigation is performed related to WVSN, in order to identify existing problems, solutions and open issues. After that, we narrow the search and survey works that address specifically WVSN for area coverage. At the end of the chapter, a critical analysis is made, consolidating the bibliography review and placing this thesis in the literature.

4.1 Dependability Evaluation of WSN

Since there are just a few works addressing dependability evaluation in WVSN, we will start the literature review identifying elements in WSN that are common to WVSN in order to incorporate them to evaluation in WVSN.

In Bruneo et al. (2010) dependability issues are analytically evaluated in terms of reliability and *producibility* (a new attribute, similar to availability, introduced in that work), modelling the behavior of sensor nodes using Markov Reward models. In that work it is considered that the only node failure condition is due to the battery depletion. The battery charging is computed considering that a sensor can be in active or sleep state, and that network topology and sensor redundancy affect the node's power consumption.

New concepts related to dependability are also proposed in Huang et al. (2014). In that paper, it is presented an approach to quantify reliability and availability, besides maintainability, safety and integrity. These attributes are assessed by computing the sojourn time in some states of a semi-Markov process (SMP) model (a generalization of CTMC). It is provided a single SMP model for the entire system, without clear specifications about fault modelling, although the case study presented considers to six failure categories, namely facilities, hardware, software, network, human error and undetermined. Whatever the system failure, its occurrence or repair is estimated based on metrics such as *mean time to reliability/availability/maintainability/integrity failure*.

The authors in Maza (2013) use Stochastic Activity Networks (SAN) to model time-continuous systems, including maintenance and fault diagnosis aspects. The availability of the system is assessed through Monte Carlo simulation. For this purpose, it is shown how to model hardware failures, triggered by sensor faults. The considered faults are the measurement noises that can generate effects like drift, bias and oscillation.

In Macedo et al. (2014) the Internet of Things context is considered to evaluate dependability of such applications. IoT applications are modelled by Markov Chains, which are evaluated using the SHARPE tool. These models consider failures in devices and in mechanism that switches for activation of redundant devices, as well as Common Cause Failures (CCF), which are failures caused by an external event that causes all functioning components to fail at the same time. It is analyzed the impact of redundancy on mean time to failure (MTTF), a metric that is used to assess reliability.

IoT is also addressed in Andrade and Nogueira (2020), in order to evaluate dependability of a disaster recovery solution. The authors propose a SNP model to describe the system behavior considering energy consumption and failures on devices and network communication. These models are solved by the Mercury software, aiming evaluation of availability, Recovery Time Objective (RTO – the maximum downtime after a failure or disaster occurs) and Recovery Point Objective (RPO – the maximum data loss following disaster). However, the proposed modelling is focused on a specific case study (a healthcare IoT system), instead of a generic approach to be applied in other scenarios.

Since analytical modelling is a time-consuming task, demanding much effort to develop models for complex topologies, some authors propose automated model generation approaches (Cinque et al., 2012; Silva et al., 2012; Dâmaso et al., 2014; Martins et al., 2015; Dâmaso et al., 2017). The authors in Cinque et al. (2012) propose a framework to assess both dependability and performance of WSNs through automatic generation of analytical models. The proposed modelling approach considers unreliable devices and unreliable links, power consumption, routing protocols, workload and radio specifications. The proposed framework integrates behavioral models (analyzed by AVRORA simulator) and analytical models (described by SAN formalism). Besides, the proposed framework is capable to infer a realistic WSN model and to evaluate this model with respect to connection resiliency, coverage level of the monitored area, data delivery resiliency and efficiency, availability of nodes, lifetime of nodes and isolation time of nodes. Nevertheless, the model templates must be predefined *una-tantum* by a domain expert.

In Silva et al. (2012) a methodology is proposed for automatic generation of analytical dependability models based on Fault Tree Analysis, in industrial environments that are subject to permanent faults on network devices and network topology. The methodology is integrated with the SHARPE tool and evaluates the application behavior with regard to reliability, unreliability, availability, unavailability, MTTF and component importance (Birnbaum and Criticality), considering line, star, cluster and mesh topologies. In Silva et al. (2013) this methodology was adapted to be applied in IoT systems, including link failures considered as permanent failures.

The authors in Dâmaso et al. (2014) present a modelling strategy to evaluate the reliability of

WSNs, considering the battery level as a key factor. The WSN power consumption is evaluated by Coloured Petri Net (CPN) models, that are composed by basic models, which represent the power consumption of different components of the application or the network. This information is used into reliability block diagrams in order to evaluate the WSN reliability. A tool integrated with Mercury software and CPN Tools is proposed to automate the evaluation process. That work has been extended in Dâmaso et al. (2017), proposing more generic models and developing a fully automated toolbox to support the proposed evaluation methodology.

In Martins et al. (2015) the authors propose a toolset to support the evaluation of the reliability and availability of WSNs in industrial environments, focusing on the automatic generation of analytical dependability models from AADL (*Architecture Analysis and Design Language*) models. The proposed framework is integrated with the SHARPE tool and decides, in a flexible way, which modelling technique is the most appropriate for each case according to the system structure and dependability metrics. It also provides a high level abstraction environment to define the failure model.

The previous works are summarized in Table 4.1 with respect to the evaluation approach applied, dependability attributes assessed, failures of components considered, auxiliary tools used in the evaluation process, as well as the level of automation of this process. The approach proposed in this thesis is included (last row) also.

Table 4.1: Summary of research in dependability of wireless sensor networks.

Work	**Evaluation Approach**	**Attributes**	**Component failures**	**Tool**	**Auto-mated**
Bruneo et al. (2010)	CTMC	Reliability Producibility	Battery charge Network topology Node's redundancy	Not specified	No
Huang et al. (2014)	SMP	Reliability Availability Maintainability Safety Integrity	Not specified	Not specified	No
Maza (2013)	SAN Monte Carlo Simulation	Availability	Hardware	Mobïus	No
Macedo et al. (2014)	CTMC	Reliability Availability	Hardware Common Cause	SHARPE	No
Andrade and Nogueira (2020)	SPN	Availability RTO & RPO	Hardware Energy Communication	Mercury	No

Table 4.1: *Cont.*

Work	Evaluation Approach	Attributes	Component failures	Tool	Auto-mated
Cinque et al. (2012)	SAN	Resiliency Availability Lifetime	Battery depletion Hardware Communication	AVRORA	Yes
Silva et al. (2012, 2013)	FTA CTMC	Reliability Availability Importance	Hardware Network topology Link	SHARPE	Yes
Dâmaso et al. (2014, 2017)	RBD CPN	Reliability	Battery charge Network topology	CPN Tools Mercury	Yes
Martins et al. (2015)	Flexible	Reliability Availability	Not specified	SHARPE	Yes
Proposed approach in this thesis	FTA CTMC	Reliability Availability	Hardware Battery charge Link Network topology	SHARPE	Yes

4.2 Dependability Evaluation of WVSN

Regarding to wireless visual sensor networks, there is a lack of works addressing dependability evaluation concerning the visual aspects. Furthermore, the few existing works addressing this topic generally consider only target coverage. Costa et al. (2014a) address this issue using the methodology proposed by Silva et al. (2012) to perform availability assessments in WVSN. That evaluation assumes that the application is available if all targets are being covered by at least one visual sensor node. For this, it is considered hardware failures (battery discharging), communication failures (loss of path to a sink node) e and coverage failures (loss of view over targets). Communication failures are modelled in such a way that do not consider the effects of routing protocols. The evaluation methodology uses fault tree analysis and the network failure condition is identified by a voting gate (*k-out-of-N*). This means that whenever *k* out of the *N* visual sensors nodes monitoring the target fail, the application will fail, ignoring different importance degrees of sensor nodes.

On the other hand, some works discuss how other network elements can improve system dependability, namely availability, without a proper assessment. In this context, Costa et al. (2014c) identify and discuss availability issues of this application domain. That paper focuses on how redundancy should be considered to improve the availability level of WVSNs, with respect to camera's FoV overlapping, sensing similarity and sensing relevance. Moreover, the authors present some common hardware and coverage failures that can affect such availability level. The authors show that the availability evaluation in WVSN has to consider coverage quality, quality of viewing, barrier monitoring, directional k-coverage and users perceptions. In Costa et al. (2014c), it is also discussed practical approaches and mechanisms to evaluate and to enhance availability in WVSN.

Redundancy in WVSN is considered in Costa et al. (2014b) and Costa et al. (2017b), and used as a dependability metric. In these cases, availability is related to the redundancy level. The authors compute the FoV of each visual sensor node in order to know the application FoV and to select the redundant nodes. In Costa et al. (2014b) this metric is evaluated considering the minimum percentage of FoV and the maximum acceptable angle between sensors orientation. In Costa et al. (2017b) occlusion is added as a redundancy parameter, changing the way to compute FoV. Additionally, an algorithm to adjust cameras' orientations is proposed to enhance the availability of WVSN with occlusion.

In Costa and Duran-Faundez (2016), a new coverage metric is proposed, the *Effective Target Viewing* (ETV), which characterizes the percentage of viewed parts of targets' perimeters. That metric is exploited to assess the availability of WVSN monitoring applications. In this case, ETV is associated with an availability state, which may be "yes" (available) or "no" (unavailable), according to the defined *Minimum acceptable ETV* (M-ETV) threshold. Although addressing dependability issues of WVSN, Costa et al. (2014b), Costa and Duran-Faundez (2016) and Costa et al. (2017b) only address coverage aspects, ignoring communication and hardware issues.

It can also be found works that consider aspects that affect dependability of WVSN, although these works do not discuss the direct relation between these aspects and dependability issues, such as the aforementioned energy consumption, quality of coverage, distance and angle (perspective) of monitoring, occlusion. For instance, Mirzazadeh Moallem et al. (2020) consider the distance between camera node and target and the angle between the main view line of camera and the target to compute the image entropy. This metric is used to optimize energy consumption by selecting visual nodes that best cover a set of targets. Energy consumption is also optimized by Ghazalian et al. (2020), looking for a balancing with the image quality (quality of experience (QoE)) in target coverage applications. For this, the authors also perform a sensor selection, adjusting the focal length to guarantee the target tracking with the lowest energy consumption.

Yap and Yen (2017) proposed a new occlusion probability model to compute the occlusion probability from fixed and moving obstacles. Based on this occlusion probability model, an optimization formulation is developed to cover the largest number of uncovered objects with highest nonoccluded probability. Jiang et al. (2020) consider an approach that includes occlusion constraints in a 3-D scenario, aiming to reach a satisfactory coverage quality with a short processing time. They propose a fast, scalable and distributed deployment approach for coverage optimization of visual sensor networks.

Table 4.2 summarizes the previous works regarding different viewpoints. These papers are categorized analyzing if they consider the following aspects that can impact dependability of a WVSN: nodes redundancy, energy efficiency, quality of monitoring and occlusion. It is also considered which perspective these papers approach dependability, *i.e.*, which attributes are addressed and if they are evaluated. Our approach is also included, for comparison purposes.

Table 4.2: Summary of research in dependability of wireless visual sensor networks.

Work	**Dependability perspective**	**Redundancy**	**Energy efficiency**	**QoM**	**Occlusion**
Costa et al. (2014a)	Availability evaluation	X	X	–	–
Costa et al. (2014c)	Availability discussion	X	X	X	X
Costa et al. (2014b)	Availability discussion	X	–	–	–
Costa et al. (2017b)	Availability discussion	X	–	–	X
Costa and Duran-Faundez (2016)	Partial availability evaluation	–	–	X	–
Mirzazadeh Moallem et al. (2020)	–	–	X	X	–
Ghazalian et al. (2020)	–	–	X	X	–
Yap and Yen (2017)	–	–	–	–	X
Jiang et al. (2020)	–	–	–	X	X
Proposed approach in this thesis	Availability Reliability evaluation	X	X	X	X

4.3 Visual Coverage Failures in WVSN for Area Coverage

Visual coverage failures are related to insufficient area coverage, which may result from different causes such as incorrect deployment, occlusion and low quality of monitoring due to visual effects. These are some important topics that have been considered in different ways when performing visual monitoring or when assessing quality of monitoring for area coverage.

He et al. (2016) and Hsiao et al. (2017) address full-view area coverage in WVSN, aiming at the computing of the total area that is being covered by the sensors. For that, those works take into account the orientation of each point or region that is faced by the sensors. In these situations, the quality of monitoring is directly related with the facing angles.

Konda et al. (2016) also address area coverage, focusing in the automatic determination of the number, position and pan-tilt-zoom setting of a set of cameras to maximize the coverage. In this process, the authors consider QoM by handling the distortion models that can affect the quality and usability of the acquired data.

For Shriwastav and Song (2020), area coverage is considered in a different class of WVSN, which are the Unmanned Aerial Vehicle (UAV) networks. In this case, each UAV is equipped with a camera, being responsible to perform a persistent full coverage of the area. If eventually an UAV fails, it is considered "unrecoverable" and the network must be redeployed, adjusting position and loitering movements to keep the area covered. An important aspect is that the authors consider the loitering altitude constant in order to keep the coverage quality constant.

Although considering QoM in area coverage, neither the works in He et al. (2016); Hsiao et al. (2017); Konda et al. (2016); Shriwastav and Song (2020) actually compute a sensor node's or network's QoM. Tao et al. (2017) overcomes this issue by defining a weighted sensing quality, which is computed over a predefined deployment scheme. In that case, it is discussed the importance of sensing area to establish QoM in a full area coverage scenario. However, several restrictions are imposed to the network regarding to the nodes' position, orientation and viewing angle.

Some works also relate QoM with occlusion, which is commonly addressed as an aspect that reduces the coverage area. Scott et al. (2016) do not address the overall area coverage quality, since they consider occlusion to determine a satisfactory sensing quality of a monitoring area, when a set of waypoints are chosen for taking picture of some objects. The proposed coverage technique is used to achieve energy efficiency by minimizing the energy consumption among sensor nodes.

Still considering this overall scenario, Costa et al. (2017b) address occlusion in a more significant way, proposing computation of area coverage under occlusion. That work proposed a model to compute occlusion caused by simplified static obstacles, which were modelled as 2D walls (lines). In that work, occlusion affected FoV overlapping and selection of redundant nodes, directly impacting on availability assessment.

Table 4.3 summarizes the previous works, classifying them into areas related to QoM, occlusion and modelling of visual coverage failures. The approach proposed in this thesis appears in the last position.

4.4 Synthesis of Reviewed Works

Analyzing the reviewed works and looking at their summarizing Tables 4.1, 4.2 and 4.3, we can verify that dependability evaluation of general WSN has been a well addressed topic in the literature, having several works discussing a wide range of different perspectives. However, these works share some common features. It can be noticed the preferential usage of techniques to model the dynamic behavior of a system through a state space-based approach. These approaches are mostly focused on reliability and availability evaluation, generally considering in the evaluation the effect of battery depletion, hardware and communication failures. Also, it has been common to use a

Table 4.3: Visual coverage research in dependability of wireless sensor networks.

Work	QoM	Occlusion	VCF	Brief Description
He et al. (2016); Hsiao et al. (2017)	X	–	–	Area coverage analysis aiming at full-view coverage, considering QoM by the facing angle of points of interest.
Konda et al. (2016)	X	–	–	Automatic deployment of WVSN to maximize coverage and visual quality in indoor environments, considering perspective distortion of the acquired images.
Shriwastav and Song (2020)	X	–	–	UAV network for area coverage. Network parameters are carefully handled during redeployment to keep the coverage quality constant.
Tao et al. (2017)	X	X	–	QoM-enhancing coverage scheme in a full area coverage scenario, considering different area importance and weighted quality of captured image.
Scott et al. (2016)	X	X	–	Occlusion-aware area coverage with aerial sensing quality, providing satisfactory spatial resolution subject to the energy constraints of UAVs.
Costa et al. (2017b)	–	X	X	Selection of redundant visual nodes for enhanced resistance to visual failures, considering occlusion caused by obstacles modelled by lines.
Proposed approach in this thesis	X	X	X	Automated integrative methodology for dependability evaluation of WVSN, based on fault tree analysis, considering hardware, communication and visual failures, related to occlusion and QoM.

software to aid the evaluation process, with a more incidence of the SHARPE tool, mainly in the works that automate this process.

When the research context is extended to WVSN, just a few works have been found in the literature explicitly evaluating the dependability of such visual networks, and they are limited to target coverage. On the other hand, there are several works discussing well the aspects that impact on dependability of WVSN, even not directly associating these aspects to dependability evaluation. Such aspects are mostly nodes' redundancy, energy efficiency, quality of monitoring and occlusion.

Quality of monitoring and occlusion should receive a special attention since they exert higher influence on the dependability of WVSN, being some of the main causes of visual coverage failures. Several works have been found in the literature approaching QoM of a visual network, focusing on the level of sharpness and definition of the gathered visual data. A few works have considered occlusion as an impact factor of QoM, but not related to VCF. On the other hand, some works can be found associating occlusion as the cause of VFC, but not addressing QoM.

Therefore, we can enumerate some still opened issues to be addressed in the literature. Overall, dependability evaluation of wireless visual sensor networks is an incipient problem. It still needs to receive more attention, specially to provide for these networks the capability to be applied into safety-critical systems. That way, visual networks require the development of dependability models, metrics and tools specifically for their particularities. The few existing works are focused on target coverage, which makes dependability evaluation of WVSN for area and barrier coverage an open problem.

On the other hand, there are several works that provide valuable solutions for aspects that impact dependability of such networks, although they do not provide the proper addressing of dependability evaluation of WVSN. However, none of the works has addressed theses aspects in a integrated way, modelling the influence that one can exert on the other.

In this context, this thesis fills the gap in the literature of dependability evaluation of visual networks for area coverage. It is proposed a methodology to analytically evaluate dependability metrics of wireless visual sensor networks, centered on the concept of visual coverage failures, generated by occlusion or low monitoring quality. The methodology processes availability, reliability and effective coverage parameters, considering essential aspects in an integrated fashion, *i.e.*, the influence of power consumption, battery discharging, hardware, link and coverage failures, as well as the behavior of routing protocols. Therefore, the methodology proposed in this thesis arises as a valuable contribution to perform an unique and comprehensive dependability assessment.

4.5 Conclusion

Due to its importance, dependability evaluation is an issue widely addressed in the literature, especially on WSN. Several works can be founded surveying some aspects of dependability of WSNs, while others survey WVSNs. However, just a few works discuss dependability aspects related to the specificities of WVSNs. On the other hand, aspects that affect dependability of WVSN are well addressed, although not necessarily associated to their potential impact on dependability. That way, in this chapter it was provided a literature review of dependability evaluation on WSN and WVSN, focusing on the several aspects that can be applied and enhanced on WVSN. This review allowed the identification of existing problems, solutions and open issues regarding the investigated topic, which foster this thesis proposal. That way it is filled a gap in the literature of dependability evaluation of wireless visual networks for area coverage through the development of an evaluation methodology that integrates different types of failure in a comprehensive and unified mathematical modelling.

Part II

Main Contributions

Chapter 5

Dependability Evaluation Methodology

In this chapter is presented a new methodology for dependability evaluation of Wireless Visual Sensor Networks. Several models are discussed in order to describe faults, failures and behavior of the system and its components. These models are implemented in algorithms that compose an automated framework integrated with the SHARPE tool, which takes advantage of its support of hierarchical models. The algorithms are described and some examples of how to use the methodology are also presented, including how to use it to support important project decisions.

System dependability, roughly speaking, can be seen as a service that may or may not be correctly provided at a given time instant. A correct service provision is associated to the achievement of some metric related to the application. In this chapter, the imposed requirement is that the WVSN application needs to monitor a minimum percentage of the area of interest. The achievement of this goal depends on the components that are operating at each moment and the interaction among them.

This way, if the hardware of a visual node fails, it will be incapable to collect visual information. On the other hand, if a link or an intermediate node that compose a path to the sink node fails, then the visual information of one or more nodes will not be delivered to the sink node. In any case, failures will not allow the sink node to receive the whole visual information related to that minimum percentage. Thus, the WVSN application requirements are not achieved and the application fails. That way, in order to evaluate the dependability of a WVSN, it is necessary to define and to model how each element can fail, besides the relations among them. This is achieved first defining the possible failures and faults that can occur in the system.

5.1 Failure Model

A failure model describes the behavior of a failed component and the probable faults related (Warns, 2010). We consider three types of failures: coverage, node and link. A visual coverage failure is an abstraction of the network status related to its ability to gather information from environment. On the other hand, a node can fail if at least one of its hardware components fails, which can be the processor, radio, battery, memory and sensor units (*e.g.* camera). However, for modelling purposes we partitioned node components in two categories: battery and generic hardware (processor, memory, radio and sensors). This partition is made because these categories present different behaviors, so we propose different models for them. In this work we consider that a node failure is *permanent*, requiring a replacement or repair action to come back to the operational state. Nodes interact between them through message exchanges. This interaction can fail if, obviously, nodes fail, or if the communication links fail. In this thesis we consider that a link failure is *transient*, meaning that the failed link will reestablish its connection after a while, without an external intervention.

A coverage failure can be classified as an *incorrect computation failure*, *i.e*, besides the consistent behavior of the visual sensor nodes, the acquired visual data cannot provide the proper computed information, probably due to a bad deployment or a external aspect (occlusion, weather, luminosity). The node and link failures are approached as *crash failures*, which means that we consider that these network elements may suddenly halt, becoming nonoperational. Each one of these failures is differently characterized and associated to a repair action in order to mitigate it. In this thesis it is considered that a hardware failure can be a defective electronic component and the hardware repair can be a component replacement or a component fixing. We consider a battery failure as equivalent to its full discharge, which can be repaired by a replacement or a recharging. A link failure can be a radio interference, noise, an occupied channel or a data collision. The transient nature of a link guarantees an eventual repair. Regarding coverage failures, it is not considered an repair action for them, since these are systemic failures, which means that it is required knowledge of all visual nodes to deal with them. However, we approach some optimization processes that can be viewed as a solution to deal with visual coverage failures in order to guarantee the coverage requirement from the application.

5.2 Fault Model

Based on the previous failure model, and in order to define the scope of this work, we follow the dependable taxonomy proposed by Avizienis et al. (2004) to establish the fault model, *i.e.*, to specify the expected classes of faults that the system can exhibit during its lifetime, as shown in Figure 2.2.

First, we consider physical faults since we are interested in model the network infrastructure. Thus we do not approach faults in logical and application layers. Also, we do not consider any deliberate, malicious, incompetence or development faults. That way we only consider non-deliberate, non-malicious, accidental and operational faults, being natural or human-made, internal or external, transient or permanent faults. The approached fault can be mapped on Avizienis's dependable taxonomy as shown in the marked regions of Figure 5.1, generating the fault classes set $\mathscr{F} = \{f_{12}, f_{13}, f_{14}, f_{15}, f_{16}\}$, where f_i, $i = 12, 13, 14, 15, 16$, is the set of faults of the same class.

The next step is establishing which faults from network components can be classified into the fault classes in $\mathscr{F}$. Those network components faults will impact in determining some input data for dependability evaluation methodology, such as the failure rates. The proposed methodology

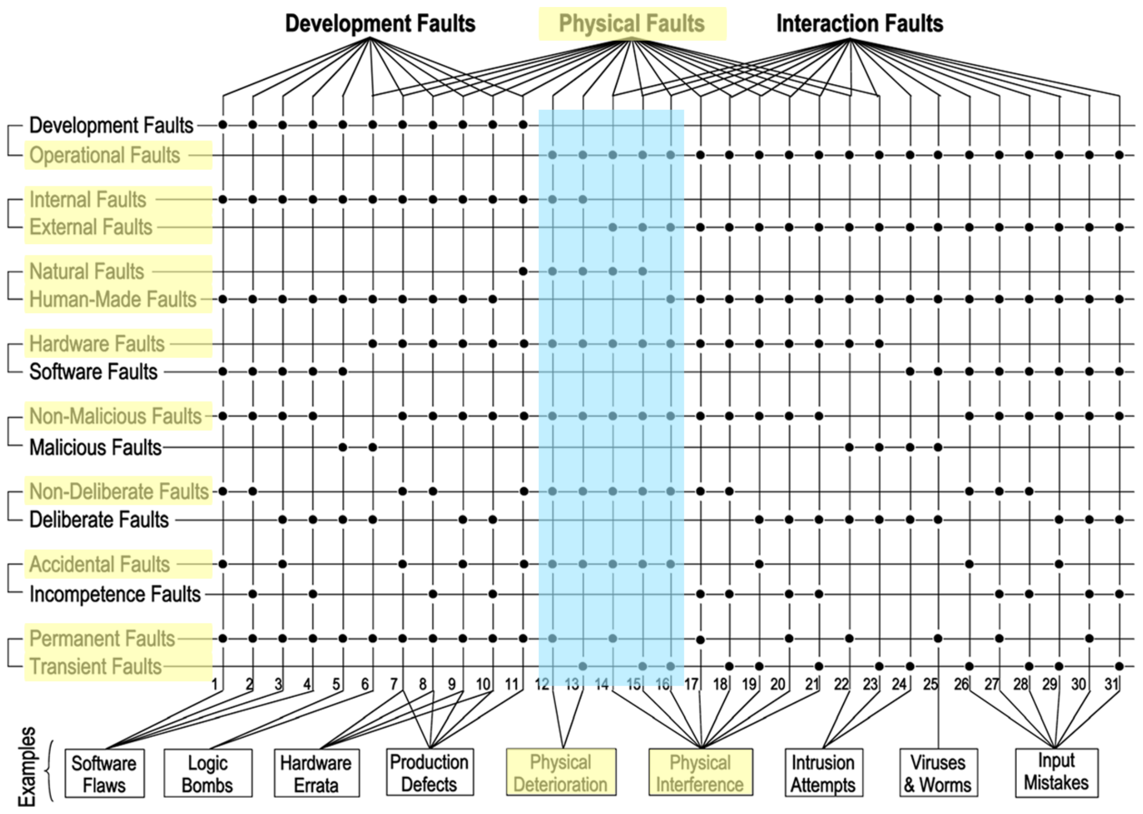


Figure 5.1: Classes of faults considered for the proposed methodology evaluation (Avizienis et al., 2004).

addresses essentially failures that can affect the visual coverage, which can be result of general hardware, battery or link failures.

A hardware failure can be consequence of faults $f_{hw} \in f_{12} \cup f_{14}$, which means permanent, natural faults, internal or external. In this case, external faults can be generated by environmental causes, radiation, power transients, noisy input lines, etc., and internal faults due to natural processes that cause physical deterioration (Avizienis et al., 2004). Due to its peculiar behavior, we distinguish battery failure from hardware failure, being the former failure equivalent only to a battery full discharge due to its operational usage, which means a battery fault $f_{bt} \in f_{12}$. Finally, a link failure can be generated by a link fault $f_{lk} \in f_{13} \cup f_{15} \cup f_{16}$, which are transient faults, being internal or external, natural or human-made.

An important aspect is a wireless link is an abstraction of communication between hardware element through a wireless medium. We consider that a wireless link consist of wireless channels jointly with all structures and entities that allow the successful information exchange between two close nodes. That way, regarding the dimension, we also classify link faults in hardware dimension. In this case, a link fault is internal ($f_{lk} \in f_{13}$) if, in hardware dimension, this fault originates within the system boundaries and only affects communication, such as channel error (Kafi et al., 2017). In a similar way, a link fault is external ($f_{lk} \in f_{15} \cup f_{16}$) if this fault affects communication due to phenomena outside the system boundary. The phenomenological cause can be natural ($f_{lk} \in f_{15}$) like bad weather, or can be human-made ($f_{lk} \in f_{16}$) such as insertion in environment of elements that can cause radiation, reflection, diffraction or scattering. Notice that we could consider a fault in the radio hardware as a link fault. But, since some sensor nodes can have their radio on-board, this ends up characterizing a fault of the general hardware. The faults' classification can be better visualized in Figure 5.2, where the faults considered in this thesis are highlighted.

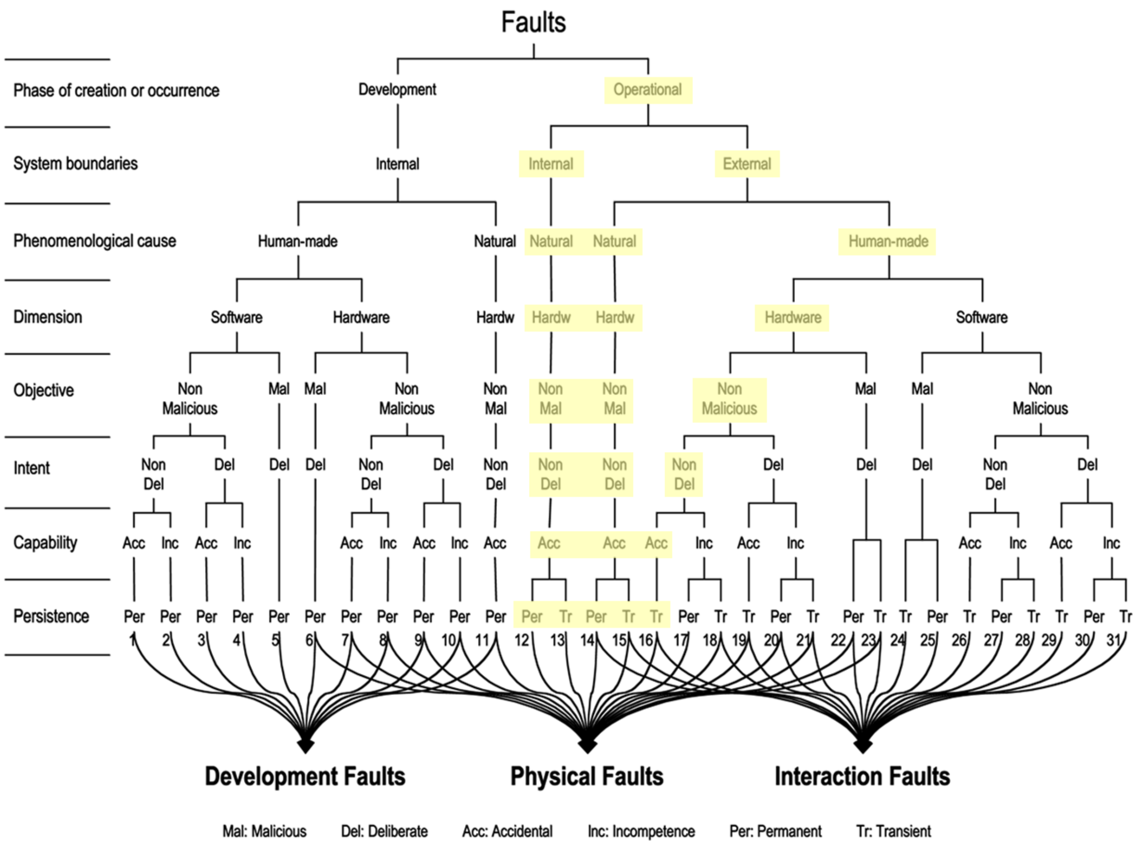


Figure 5.2: Tree representation of faults classes considered for the proposed methodology evaluation (Avizienis et al., 2004).

Once we have defined the possible reasons of a system malfunction and how to identify its undesired behavior, in the next sections this information is took into consideration to describe entirely the proposed dependability models for nodes and links, as well as components directly related to them. Also, it will be shown how to integrate all those models in the dependability evaluation process. Actually, dependability is modelled here in terms of availability. Notice that, as it was shown in Chapter 2, in order to perform a system reliability evaluation, it is only necessary to disregard the repair activities, which means setting the repair rates μ equal to zero.

5.3 Sensor Nodes Modelling

In this section, the dependability model for sensor nodes is presented. As stated previously, we split the proposed model in two parts: the (generic) hardware model and the battery model, presented as follows.

5.3.1 Hardware Modelling

In this thesis we consider the node hardware being composed by the following electronic components: processor, radio, memory and sensing unit (camera). For these elements, it is assumed that a failure is permanent and they occur with a constant rate during the useful life period of the node.

Regarding the repair processes, they are considered independent between them and the number of repairment (*i.e.*, number of repair actions) is not bounded (Limnios, 2010). We assume that the time to repair can be approximated by an exponential distribution (*i.e.* a constant repair rate). This kind of approximation is reasonable when failure and repair rates differ each other several orders of magnitude. We also assume that a repair action repairs all faulty components. Moreover, failure and repair actions are assumed *i.i.d.* (independent and identically distributed) random variables (Silva et al., 2012). That way it is possible to summarize the hardware failure rate of a node as a single and constant failure rate λ_{hw}, resulting of the sum of failure rates of each component. In an analogous way, the modelling of hardware repair actions of a node can be summarized by a single and constant hardware repair rate μ_{hw}.

The hardware behavior with respect to availability is described as a binary relationship of the components, which can be *operable* (UP) or *failed* (DOWN). In the former case the components are operational, and in the latter they are failed. This behavior can be represented by a CTMC with two states (UP_{hw}) and ($DOWN_{hw}$). The transitions between these two CTMC's states are described by the failure and repair rates, λ_{hw} and μ_{hw}, respectively. Figure 5.3 shows the described hardware model. Under the stated assumptions, the hardware availability $A_{hw}(t)$ can be computed as the probability of being on state UP_{hw}.

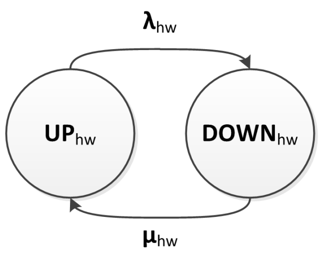


Figure 5.3: Hardware model.

5.3.2 Battery Modelling

Based on the Peukert's law (Doerffel and Sharkh, 2006; Omar et al., 2013; Rodrigues et al., 2017), which expresses the battery lifetime given an initial capacity C, the trend of the battery discharging process with respect to the time is represented by Equation 5.1. In this equation, c_0 is the initial capacity of the battery (expressed in Ampere·hour), I is the average continuous discharge current (measured in Ampere), H is the hour rating (hours), whereas η expresses the Peukert's constant, which depends on the battery material (*e.g.* 1.06 to 1.13 for lithium ion batteries and 1.2 to 1.4 for alkaline batteries).

$$c(t) = c_0 - I \cdot H \cdot \left(\frac{t}{H}\right)^{\frac{1}{\eta}} \tag{5.1}$$

The battery behavior modelled by Equation 5.1 is nonlinear and cannot be modelled using the same reasoning of the hardware modelling. To cope with this problem and to evaluate the battery availability, we propose an approximation of the nonlinear battery discharging behavior by a stochastic process, following the approach proposed by Bruneo et al. (2012).

First, it is identified the battery useful charge range as $[c_0, c_{\min}]$, splitting it into n contiguous intervals $[c_i, c_{i+1}]$ of equal size $(c_0 - c_{\min})/n$, with $i = 0, \ldots, n-1$. That way, the battery capacity is discretized into $n+1$ charge levels with generic value $c_i = c(t_i)$ $(i = 0, \ldots, n)$, where $c_n = c_{\min}$. It is assumed that the duration of the i-th time interval, $\tau_i = t_{i+1} - t_i$, with $i = 0, \ldots, n-1$, can be described by an exponential distribution, in which the charge assumes values ranging into $[c_i, c_{i+1}]$. Based on these assumptions, the discharge phenomenon can be represented by a CTMC with $n+1$ stages, defined by the stochastic process $B = \{B(t), t \geq 0\}$, as shown in Figure 5.4.

In this CTMC, the state B_i represents the i-th charge interval, τ_i can be considered as the sojourn time into the state B_i and, as a consequence, the transition rate between states B_i and B_{i+1} has to be set to $\lambda_{bti} = 1/\tau_i$. The discharge rates λ_{bti} would be analogous to a set of battery failure rates, since these rates imply that the system will eventually reach a failed state. B_n is an absorbing state that represents the $c_{\min}$ level, the probability of the battery being discharged $(c(t) \leq c_{\min})$ is $Prob\{B_n(t)\}$ and, consequently, the probability of the battery being working $(c(t) \geq c_{\min})$ can be computed as $1 - Prob\{B_n(t)\}$.

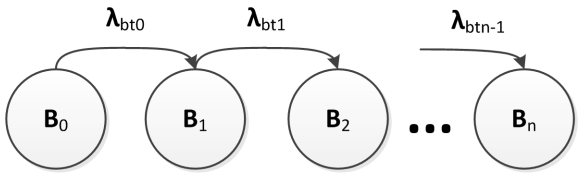


Figure 5.4: Battery discharging model.

When the battery discharges below the minimal operational level (c_{min}), it can be replaced or recharged, which is analogous to a repair. For the same reasons presented in Section 5.3.1, we can approximate the modelling of battery repair actions of a node by a constant battery repair rate μ_{bt}. For the battery availability evaluation, we propose the CTMC model presented in Figure 5.5, where the state **DOWN** is equivalent to the state B_n from Figure 5.4, and it is the only one which trigger the repair transition with battery repair rate μ_{bt}.

However, when the WVSN inherent active-sleep cycle operation is considered, this approach becomes inaccurate (Rodrigues et al., 2016). For this purpose, considering that the battery current

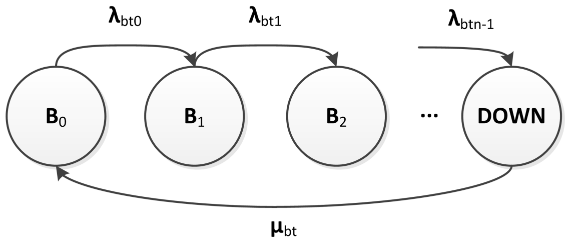


Figure 5.5: Battery model.

is almost constant in the active state and it is negligible (≈ 0) in the sleep state, the authors in Costa et al. (2014a) characterize the active-sleep cycle by a duty-cycle $DC \in [0,1]$, which is the percentage of time that a node stays in the active state. In this case, the amount of battery discharge in a interval τ_i is now proportional to DC. This is equivalent to assume that the sojourn time for each state B_i is τ_i / DC. Therefore, the transition rate λ_{bti} for each state can be redefined as follows:

$$\lambda_{bti} = \frac{DC}{\tau_i} \tag{5.2}$$

Analogous to the hardware modelling, the evaluation of the availability $bt = A_{bt}(t)$ is computed as the complementary probability of being on state $DOWN$.

5.4 Link Modelling

The link model consists in a description of the communication behavior between two nodes. Due to its wireless nature, we consider that link failures are transient. Also, as a link is an abstract concept, we cannot materialize its repair. So this repair action can be understood as the natural reestablishment of normal communication conditions after a failure, without a deliberate intervention.

Modelling the behavior of a wireless link is usually a complex task, since it depends on radio propagating modes, such as, line of sight radiation, reflections from a smooth surface, diffractions around a corner and scattering caused by an object with dimensions on the order of the wavelength (Bai and Atiquzzaman, 2003). These propagating modes can result in radio fading, signal attenuation, radio interference, background noise and other inherent characteristics of the wireless medium (Egeland and Engelstad, 2009). If we can consider that the environment where the networks are deployed are mostly composed by static elements, such as a urban area full of buildings, industries with fixed machinery and agricultural fields, then it is reasonable to consider that link failures occur according to a Poisson process, leading to a constant failure rate λ_{lk}. Repairs are modelled analogously to the hardware case, assuming a constant rate, μ_{lk}. These are optimistic assumptions which make the proposed link model an useful but constrained approximation.

That way, the link behavior with respect to availability is described as a binary relationship, which can be *operable* (UP) or *failed* (DOWN). In the former case the link is operational, and in the latter it is failed. This behavior can be represented by a CTMC with two states (UP_{lk}

and $DOWN_{lk}$). Transitions between these two states are described by the failure and repair rates, λ_{lk} and μ_{lk}, respectively. Figure 5.6 shows this link model. Under the stated assumptions, the availability, $A_{lk}(t)$, can be computed as the probability of being on state UP_{lk}.

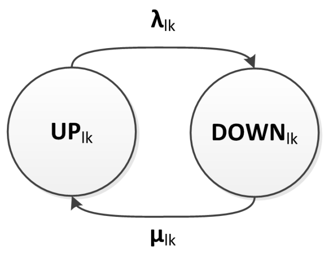


Figure 5.6: Link model.

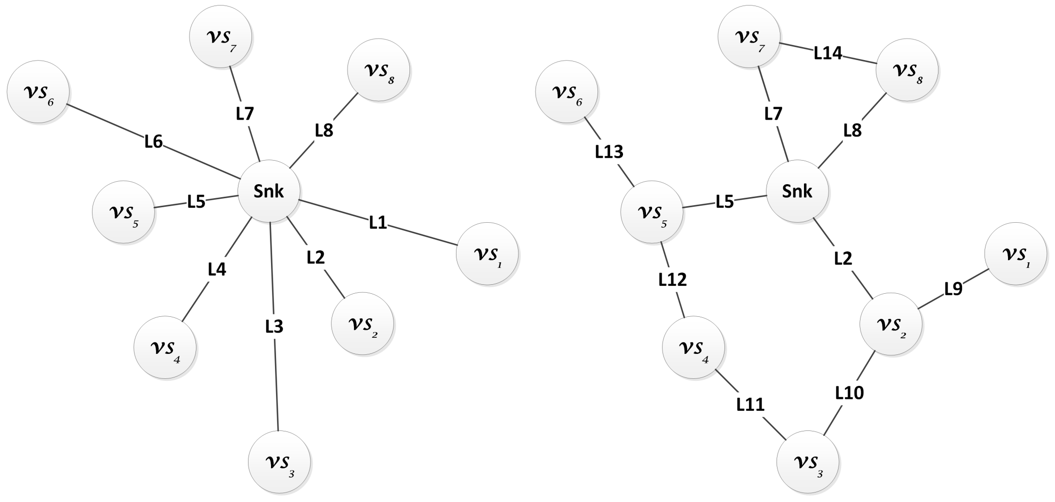
Link dependability is also related to the network routing protocol. A link failure can change the network topological arrangement, excluding a path to the sink node, disallowing or delaying the delivery of part of the network visual information. Searching a new path to the sink depends directly on the used routing strategy. In order to consider this behavior in the dependability evaluation, routing protocols are an important issue that needs to be properly evaluated.

The knowledge of the existent paths in a network is essential for the dependability evaluation, which implies that routing protocols must be considered. Since there are too many routing protocols, it is very difficult to model all of them or even to find a general pattern. On the other hand, it would be very restrictive to model a specific protocol. Instead, some authors discuss and describe routing strategies which are common to several protocols. The most used strategies that can be found are DIRECT, FLOODING, GOSSIPING and HIERARCHICAL (Karl and Willig, 2005; Dâmaso et al., 2014, 2017), with more attention to DIRECT and FLOODING strategies. That way, in this research we address and model these two strategies, and consider GOSSIPING and HIERARCHICAL as future works.

The DIRECT protocol guarantees direct connection between each node and the sink node through one single hop. Although being ideal for small networks, it may lead to high energy consumption, since it requires radios to be set with high transmission power on nodes that are far away from the sink (Senouci et al., 2012; Dâmaso et al., 2014). This protocol will be used for comparison purposes. In this work we consider the radio connectivity modelled by disk graphs, where each pair of nodes within a given distance threshold R_c (radio communication range) are connected and they can directly communicate with each other by a link (Karl and Willig, 2005).

The FLOODING protocol is a multi-hop strategy to discover multiple paths to the sink. Each sensor node broadcasts a message to all of its neighbors, which repeat that task until the message is delivered to the sink or it is dropped out due to a maximum number of hops. This protocol is easy to implement, but has some problems: duplicate messages and network overheads (Karl and Willig, 2005; Senouci et al., 2012). A good advantage of FLOODING is that it can deal with the losts of a intermediate node in a path, searching for new paths to the sink. This multiple path feature naturally provides a higher reliability, since the probability of a successful delivery of messages to the sink is higher with a higher number of possible paths. Also, as the nodes primarily communicate with their closest neighbors instead to communicate directly to the sink (which is probably more distant), each node can decrease the radio power to reach just the nodes in their vicinity, *i.e.*, with a smaller distance threshold R_c. This implies a smaller power consumption, a slower battery discharging and therefore an higher dependability.

Figure 5.7 illustrates the network topology of the same arrangement of nodes, managed by different routing strategies.



(a) (b)

Figure 5.7: Routing strategies: (a) Direct and (b) Flooding.

5.5 Integration of Models

In this section we present how to integrate hardware, battery and link models (with the routing protocols) in order to obtain a unified system model.

As aforementioned, the WVSN scenario considered in this research and presented in Figure 5.8[1] is affected by different classes of failures. The coverage failures are the most important in a WVSN, and they are related to the inability of essential visual nodes to deliver their information to the sink node. An essential node is one that directly participates on the required monitoring, so it determines the network failure condition. The NFC (network failure condition) of a WVSN application consists in a logical expression that indicates the combinations of components that, if failed, the application fails (Silva et al., 2012). Since we are considering WVSNs for area coverage, these combinations must be composed by visual nodes without which the visual information gathered by the rest of the network is not enough to achieve the required minimum area to be monitored (Costa et al., 2014a).

For example, analyzing Figure 5.8, it is possible to notice that the node vs_1 monitors a small part of the monitoring field $\mathbb{A}$, and the most part of that area is also monitored by nodes vs_2 and

[1]Figure 5.8 is the same Figure 3.1, which is presented here again to facilitate the thesis reading.

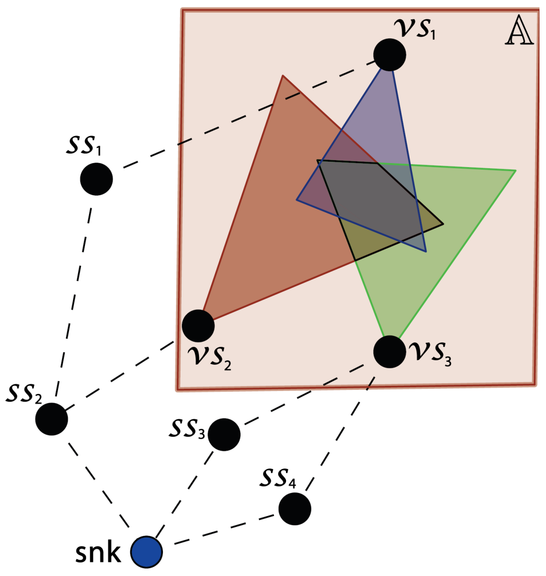


Figure 5.8: Area monitoring by visual sensors (Costa et al., 2014a).

vs_3. The node vs_2 monitors a larger area than vs_1, but also with some overlapping related to vs_1 and vs_3. That way, the NFC for this network should probably be $NFC = vs_2 \vee vs_3$, depending on the application requirement. This failure condition is assessed *true* if vs_2 is assessed *true* (if the hardware of vs_2 fails) or if vs_3 is assessed *true* (if the hardware of vs_3 fails), indicating that the nodes vs_2 and vs_3 are essential for the visual monitoring. In this case, if at least one of these nodes fails, then the remaining visual nodes are not able to collect enough visual information to meet the application monitoring requirements.

On the other hand, the application could require a minimum area slightly larger than the coverage area of any visual sensor, being necessary that at least two visual nodes are able to deliver their visual information to the sink in order to fulfill the application requirements. In other words, if any combination of two visual nodes fail, the application will fail, which implies $NFC = (vs_1 \wedge vs_2) \vee (vs_1 \wedge vs_3) \vee (vs_2 \wedge vs_3)$.

However, visual information of an essential node may also not reach the sink due to communication failures. For instance, if the scalar node ss_2 fails, then there will be no way to deliver the visual information from both visual nodes vs_1 and vs_2. In the same way, if the link connecting nodes ss_1 and ss_2 fails, or if the link connecting nodes vs_2 and ss_2 fails, then there will be no way to deliver the visual information from both nodes vs_1 and vs_2, respectively. Those elements must appear in the evaluation of the whole system since they indirectly affect its dependability.

To cope with these issues, we model this behavior using a Fault Tree, similarly to Silva et al. (2012) and Costa et al. (2014a), considering the previous classes of failures. A Fault Tree expresses the combination of events that leads to network failures. In this procedure, logic gates are used to represent cause-effect relationships among events and the NFC (the TOP event in the FT). Similarly to the NFC, a FT has to be logically evaluated as *true* to identify an application failure.

The inputs of the gates of a FT are either single events or combinations of events which result from the output of other gates. The events at the bottom of the tree are referred as basic events and must be assigned to their respective battery, hardware and link dependability functions (reliability or availability functions, according to the evaluation interest), resulting from the respective CTMCs evaluation. Considering the basic events as dependability functions of these network elements, a FT can be analytically evaluated to express the system dependability. Figure 5.9 shows the logical gates configuration for each failure condition in order to represent the events dependency in a FT structure.

Figure 5.9(a) indicates that the application fails (the TOP event is assessed equals one) if one or more given combination of nodes fail. According to Figure 5.9(b), a combination of nodes fails if all paths that connect the sink node to each node in the combination fail. As presented in Figure 5.9(c), a path fails if any link or node (also referred here as *device*) in the path from a node to the sink fails. Finally, a node fails if its hardware fails or if its battery discharges, as shown in Figure 5.9(d).

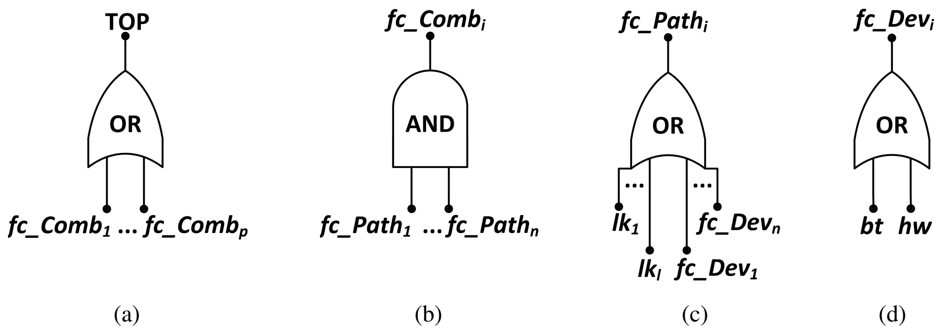


Figure 5.9: Fault Tree models of (a) network, (b) combinations of paths (c) paths and (d) devices.

In this thesis, the system dependability evaluation is considered to be, in the last stage, a Fault Tree Analysis, which requires the evaluation of how the basic events (availability or reliability functions) associated to the network elements interact between them. The proposed methodology takes advantage of the SHARPE tool to support *Hierarchical Models*. These models are structures that provide an overall model solution by composing individual model results, thus avoiding a large overall state space (Hirel et al., 2000; Trivedi and Sahner, 2009). That way, it is possible to model the events dependency (higher level) using Fault Trees. The complex behavior (lower level) can be described by state-based models (Markov Chains, for instance), which are used to model in detail the individual behavior of network elements (nodes, battery and links). That way, it is created several small models that are integrated using a Fault Tree, which allows to avoid the exponential growth of the state space. These combined formalisms are capable of modelling all the required behavior and to support the extraction of all the required metrics, like availability and reliability (Lanus et al., 2003).

Notice that the individual models integrating the hierarchical models are independent, otherwise it would be necessary an iterative process to solve the models. Through the CTMC of each individual model of the network elements it is possible to compute the availability $A(t)$ and the reliability $R(t)$ of each element, and use them as input events in the FT model. As mentioned in Chapter 2, the hierarchical models could be described based on other formalisms, such as using RBD instead of FT, or using SPN instead of Markov Chains. Whatever the case, the achieved results are equivalent. FTs were chosen since they are more intuitive.

Figure 5.10 presents an overview of the proposed methodology, which is detailed in the following sections.

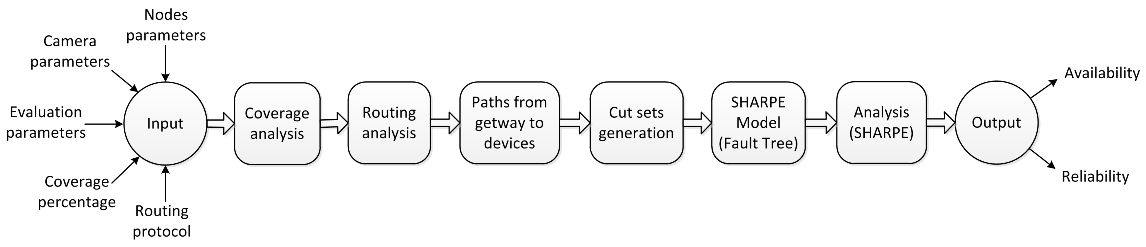


Figure 5.10: Overview of the methodology for dependability evaluation.

5.6 Data Input

The automated framework developed to implement the proposed methodology requires some supplementary data from the user in order to characterize the network and the application requirements. This information is related to network configuration, visual coverage attributes, nodes communication and the evaluation process itself, namely:

1. Nodes parameters:
    - Number of visual nodes, n;
    - Number of scalar nodes, m;
    - (x,y) position of nodes, (A_{x_i}, A_{y_i});
    - Radio communication range, R_c;
    - Hardware failure and repair rates, λ_{hw}, μ_{hw};
    - Battery discharge and repair rates, λ_{bt_i} and μ_{bt}, and battery stages, *nStages*;
    - Link failure and repair rates, λ_{lk}, μ_{lk};
2. Cameras parameters:
    - Viewing angle, θ_{vs};
    - Orientation, α_{vs};
    - Sensing radius, R_s;
3. Evaluation parameters:
    - Evaluation period, T;
    - Time step, t_s.

4. Minimum visual coverage percentage, CA_{min};
5. Routing protocol, DIRECT or FLOODING;

It is important to remark that, although for the sake of simplicity of notation it is assumed that all nodes and links have the same characteristics (same parameter values), the methodology supports, without any modifications, the assignment of individual values for each of them.

5.7 Coverage Analysis

The first step of the proposed methodology aims at establishing which visual sensor combinations are capable to monitor the interest monitoring field $\mathbb{A}$, in order to provide the minimum coverage area required by the application, CA_{min}. However, the proper area computation can be a difficult and time consuming task, since there may be many overlapping areas and many defined sub-regions. This scenario can be viewed as a combinatorial problem, resulting on exponential solutions, as shown in Section 3.1.2 (Mavrinac and Chen, 2013).

The work of Costa et al. (2017a) presents an interesting method to compute the covered area, based on monitoring blocks, as also shown in Section 3.1.2. That method provides an approximation to compute the total coverage area and it seems to be computationally efficient. Unfortunately, the authors Costa et al. (2017a) were not mainly focused on that method and did not provide a proper evaluation and analysis about complexity, performance and precision of it. That way, here we are interested in analyzing the area coverage method in order to analyze if it is accurate enough to be used as a faster solution to area coverage computation. To cope with that problem and to have a basis of comparison, we developed an accurate area coverage calculation method, which is based on the Inclusion-Exclusion principle (Brualdi, 1977; Andreescu and Feng, 2003; Gross, 2008). We describe and present both algorithms, analyzing their computational complexity and performing a comparison of their precision.

5.7.1 Overlapping Computing Algorithms

Roughly speaking, the main idea of the proposed algorithm herein is that, for each occurrence of elements in the intersection of an odd number of sets, the number of those elements must be incremented, and for each occurrence of elements in the intersection of an even number of sets, the number of those elements must be decremented, according to the stated Theorem 3.1.

The coverage area computation described in Algorithm 1 is based on Definition 3.2, which is applied for each combination of sensors in VS, in Line 4. In that same line a procedure is used to compute the overlapped areas of covered regions. That procedure is described in Algorithm 2 and it consists in the identification of the polygon formed by the vertices of overlapped regions. After that, the polygon area is computed according to Equation 5.3 (Line 10), based on the Shoelace equation (Pure and Durrani, 2015; Costa et al., 2017b), where $|V|$ is the number of vertices of the polygon and Vx_i and Vy_i are the (x,y) coordinates. Those vertices must be in a clockwise or anti-clockwise order, since it is a requirement for the Shoelace algorithm.

Algorithm 1: Proposed coverage area algorithm

Data: area = CA(VS, MA);
Input: List of sensors' parameters, monitoring area parameters.
Output: Accurate coverage area.

```
1  area = 0;
2  foreach subset of sensors Q in sensors set VS do
3      n = size(Q);
4      area = area + $(-1)^{n-1}$· overlapArea(Q); // Algorithm 2
5  end
6  return area;
```

Algorithm 2: Overlap area algorithm

Data: overArea = overlapArea(Q);
Input: List of parameters of subset of sensor.
Output: Overlap Area.

```
1  overArea = 0;
2  n = size(Q);
3  if n==1 then
4      overArea = getArea(Q[n]);
5  else
6      polyEdges = Q[1].edges;
7      for i ← 1 to nSSensors-1 do
8          polyEdges = getInterEdges(polyEdges, Q[i+1].edges);
9      end
10     overArea = getArea(polyEdges);
11 end
12 return overArea;
```

$$overArea = \frac{\left| Vx_{|V|}.Vy_1 - Vx_1.Vy_{|V|} + \sum_{i=1}^{|V|-1} \left(Vx_{i+1}.Vy_i - Vx_{i+1}.Vy_i\right) \right|}{2} \tag{5.3}$$

The proposed algorithm accurately computes the coverage area of all deployed visual sensors over the considered MA. However, such computation can be done using approximation, as presented by Costa et al. (2017a). Actually, that algorithm considers the scenario presented in Figure 3.6. In that case the MA is divided in smaller regions, called monitoring blocks (MB). The main idea of the method proposed by Costa et al. (2017a) is to verify if the center of a MB is inside of the FoV of a sensor. In affirmative case, that MB is covered and its area $ws \times hs$ is added to the coverage area. Since each MB is tested a single time, there is no way to compute an overlapped area more than once. Algorithm 3 details that method.

The verification if a monitoring block *mb* is covered by a visual node *vs* is performed in Line 18 of Algorithm 3, through the procedure *isInsideTriangle()* that considers the formulation in

Algorithm 3: Approximated coverage area algorithm

```
Data: approxArea = $CA_{mb}$(VS,MA,ws,hs);
Input: List of sensors' parameters, monitoring area parameters, width and height of
       approximation.
Output: Approximated coverage area.
approxArea = 0;
n = size(VS);
M = MA.width/ws;
N = MA.height/hs;
for $k \leftarrow 1$ to $M$ do
    for $l \leftarrow 1$ to $N$ do
        $x1s = x1 + (k-1) \cdot (w/M) \cdot cos(\beta) + (l-1) \cdot (h/N) \cdot sin(\beta)$;
        $y1s = y1 + (k-1) \cdot (w/M) \cdot sin(\beta) - (l-1) \cdot (h/N) \cdot cos(\beta)$;
        $x2s = x1 + k \cdot (w/M) \cdot cos(\beta) + (l-1) \cdot (h/N) \cdot sin(\beta)$;
        $y2s = y1 + k \cdot (w/M) \cdot sin(\beta) - (l-1) \cdot (h/N) \cdot cos(\beta)$;
        $x3s = x1 + (k-1) \cdot (w/M) \cdot cos(\beta) + l \cdot (h/N) \cdot sin(\beta)$;
        $y3s = y1 + (k-1) \cdot (w/M) \cdot sin(\beta) - l \cdot (h/N) \cdot cos(\beta)$;
        $x4s = x1 + k \cdot (w/M) \cdot cos(\beta) + l \cdot (h/N) \cdot sin(\beta)$;
        $y4s = y1 + k \cdot (w/M) \cdot sin(\beta) - l \cdot (h/N) \cdot cos(\beta)$;
        // Coordinates of the center of mb (See Figure 3.6).
        $xc = x1s + (x4s - x1s)/2$;
        $yc = y1s + (y4s - y1s)/2$;
        for $i \leftarrow 1$ to $n$ do
            inside = isInsideTriangle(xc,yc,VS[i]);
            if inside then
                approxArea = approxArea + ws·hs;
                break;
            end
        end
    end
end
return approxArea;
```

Equation 3.4 to compute the area of the triangles generated when by the center of *mb*, (xc, yc), and the vertices of the FoV_{vs} $\triangle ABC$. If the Equation 3.5 is true, then $mb \in FoV_{vs}$.

An initial complexity analysis was performed for both Algorithms 1 and 3, based on the influence of variations of the number of visual sensor nodes. We consider the asymptotic notation, where the worst case scenario is assumed, and we consider that each single command, as logic and arithmetic operations, comparisons, attributions, procedure invocations and parameters acquisition, have unity computational cost, implying in a complexity O(1).

Since Algorithm 1 invokes Algorithm 2, we will start the analysis by the last. The worst-case scenario is when it is executed the *else* block in Line 5. In that block, the procedure *getInterEdges()* in Line 8 searches for intersection points among edges of two polygons (P1 and P2), and for vertices from one polygon that are inside of the other. For the searching of intersection points, each edge built from the set of vertices V1 $\in$ P1 are tested against each edge built from the set of vertices V2 $\in$ P2. For the verification of inside vertices, each edge of V1 and V2 are tested once. That way, that procedure has computational cost $|V1| \times |V2| + |V1| + |V2|$. Since this procedure is initially executed for areas resulting of intersection of the *MA* rectangle and a sensor *FoV* triangle, this implies in 7 vertices per area, at most. Also, since the loop in Line 7 is executed $|VS| - 1$ times, that execution has a computational cost $(7 \times 7 + 7 + 7) \times (|VS| - 1)$, where $|VS|$ is the cardinality of the set VS, *i.e.*, the number of visual sensor. Besides, the procedure *getArea()* in Lines 4 and 10 computes the area of a polygon, according to Equation 5.3. That formulation is executed many times as the number of vertices in the overlap polygon. Since the overlap polygon starts with 7 vertices, at most, and a intersection of two polygons cannot create more edges than the sum of the existing edges in each polygon, in the worst case the intersection of all sensors *FoV* implies in a computational cost of $7 \times |VS|$ to the procedure *getArea()*. That way, disconsidering the single commands, Algorithm 2 has complexity O($(7 \times 7 + 7 + 7) \times (|VS| - 1) + 7 \times |VS|$) = O($|VS|$).

Considering Algorithm 1, since it basically calls the procedure *overlapArea()* in Line 4, described by Algorithm 2 for each subset of sensors, and there are $2^{|VS|} - 1$ possible subsets (we are excluding the empty set $\emptyset$), that algorithm has complexity O($|VS| \times (2^{|VS|} - 1)$) = O($|VS| \times 2^{|VS|}$), which is exponential.

Finally, the Algorithm 3 executes several single commands in a loop chain, including the procedure *isInsideTriangle()* in Line 18, which present 4 single commands (as shown in Eqs. 3.4 and 3.5), so it has complexity O(1). In the worst case scenario that algorithm executes a constant number of single commands, c, $k \times l \times |VS|$ times, where $k = w/ws$ and $l = h/hs$. That implies that Algorithm 3 has complexity O($c \times k \times l \times |VS|$) = O($k \times l \times |VS|$), which is polynomial.

In short, based on this initial complexity analysis, it is expected that, in average, the proposed accurate algorithm will have a higher computational cost than the algorithm described by Costa et al. (2017a).

5.7.2 Area Coverage Comparison

Aiming to support the analysis of the complexity, performance and accuracy of the considered algorithms, some numerical results are provided when computing the coverage area. As shown in Section 5.7.1, the method proposed by Costa et al. (2017a) presents a lower computational complexity. However, that method provides only an approximation of the real value of the coverage area for an application. For instance, the scenario in Figure 3.3 presents a coverage area of 664.9312 units of area (*u.a.*) when computed by the proposed method, while they present a coverage area of 672 *u.a.* when computed by the method of Costa et al. (2017a). For that scenario we considered $R_s = 20$ units of distance (*u.d.*), $\theta_{vs} = 60°$, $w = 50$ *u.d.*, $h = 30$ *u.d.*, $ws = hs = 2$ *u.d.* and $\beta = 30°$.

That discrepancy on the results are directly related to the size of the monitoring blocks. The smaller the MBs, the more precise will be the method of Costa et al. (2017a), since if *ws* and *hs* tend to zero, the monitoring block tends to its center point (xc, yc), and then the area computed by Algorithm 3 tends to the area computed by the proposed Algorithm 1.

In order to assess the precision of Algorithm 3 as function of the values *ws* and *hs*, we implemented those algorithms and designed some simulations in MATLAB. First it was defined a scenario with 20 visual sensors with $R_s = 20$ *u.d.* and $\theta_{vs} = 60°$ covering a monitoring area with $w = 120$ *u.d.*, $h = 60$ *u.d.* and $\beta = 30°$. The position and orientation of each sensor node were randomly generated. We performed 500 simulations with this same scenario, only assuming different random positions and orientations of sensor node. On each simulation we executed the proposed method once to get the real value of coverage area, and we executed the method of Costa et al. (2017a) once for each value of $ws = hs \in \{10, 5, 4, 3, 2, 1.5, 1, 0.5, 0.25, 0.15, 0.1\}$ *u.d.* Figure 5.11 shows the error between the methods computations taking the average values of simulations, and the standard deviation of each series, in order to better comprehend the dispersion of the simulations. The error is calculated as $100\% * (area - approxArea)/area$.

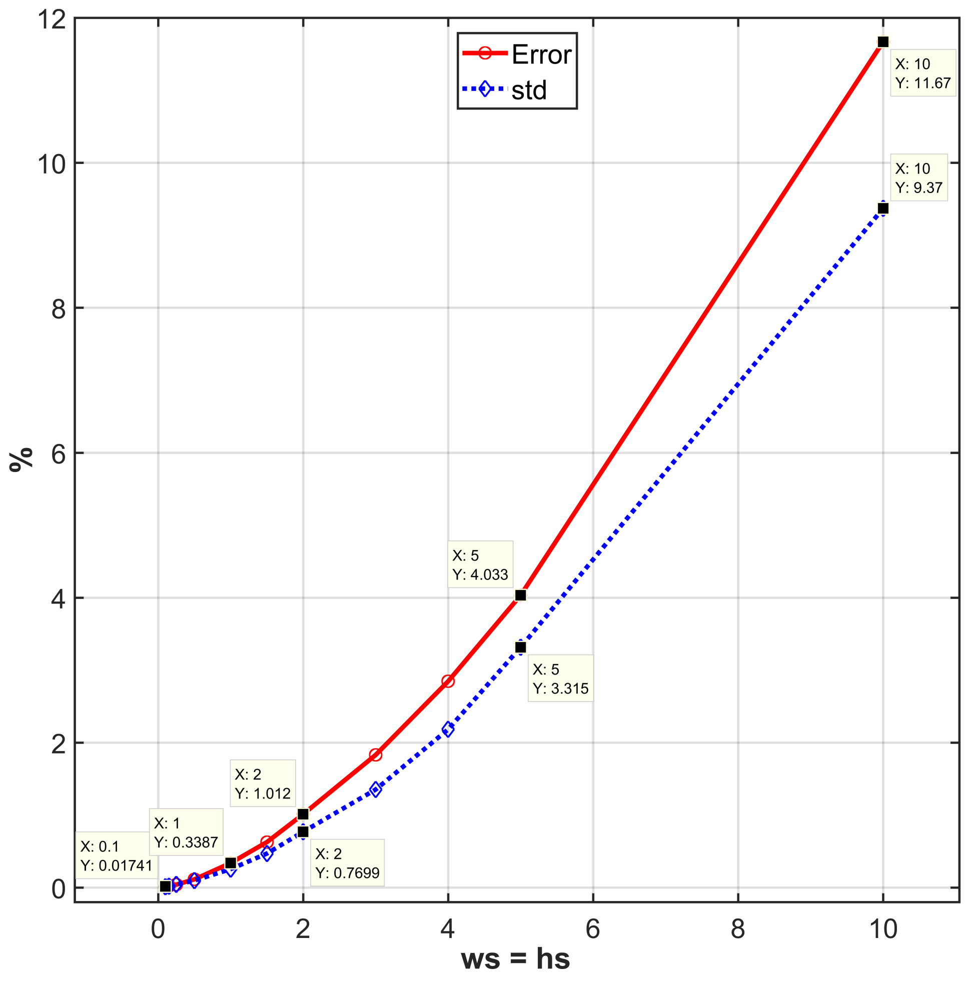


Figure 5.11: Error and standard deviation of simulations.

As expected, small values of *ws* and *hs* provide better results, with small error rates, making feasible the usage of the method of Costa et al. (2017a), even providing an approximated result.

For instance for *ws* = *hs* = 0.25, the error is 0.04% with 0.04% of dispersion, and for *ws* = *hs* = 2, the error is 1.01% with 0.77% of dispersion. However, as shown in Section 5.7.1, the execution time of that method depends directly on the values of *ws* and *hs*. Thus, it was measured the execution time of the method proposed in Costa et al. (2017a), as function of the parameters *ws* and *hs*. Since the execution time depends on several aspects, such as the computer configuration, the operating system, the background execution tasks in the processor, among other factors, it was only analyzed each execution time as a percentage of the previous execution time in order to better comprehend how slower is the execution if the number of MBs is increased. For that comparison, $ws = hs \leq 2$ were considered as parameters, since these values generate errors under 1%, which is assumed in this work as reasonable. This parameter configuration is executed in 98.5 *ms* and the comparison result can be viewed in Figure 5.12. Notice that a little increase on the precision implies in higher execution time. For instance, when we use $ws = hs = 1$ instead of *ws* = *hs* = 2 the gain of precision is only of 0.38% with a increase of 299.2% on the execution time (393.4 *ms*). If we change from *ws* = *hs* = 1 to *ws* = *hs* = 0.5, the gain of precision is only of 0.23% with a increase of 299.8% on the execution time (1.57 *s*). So, it is important to perform proper analysis to select the best value of *ws* and *hs* for the application requirements.

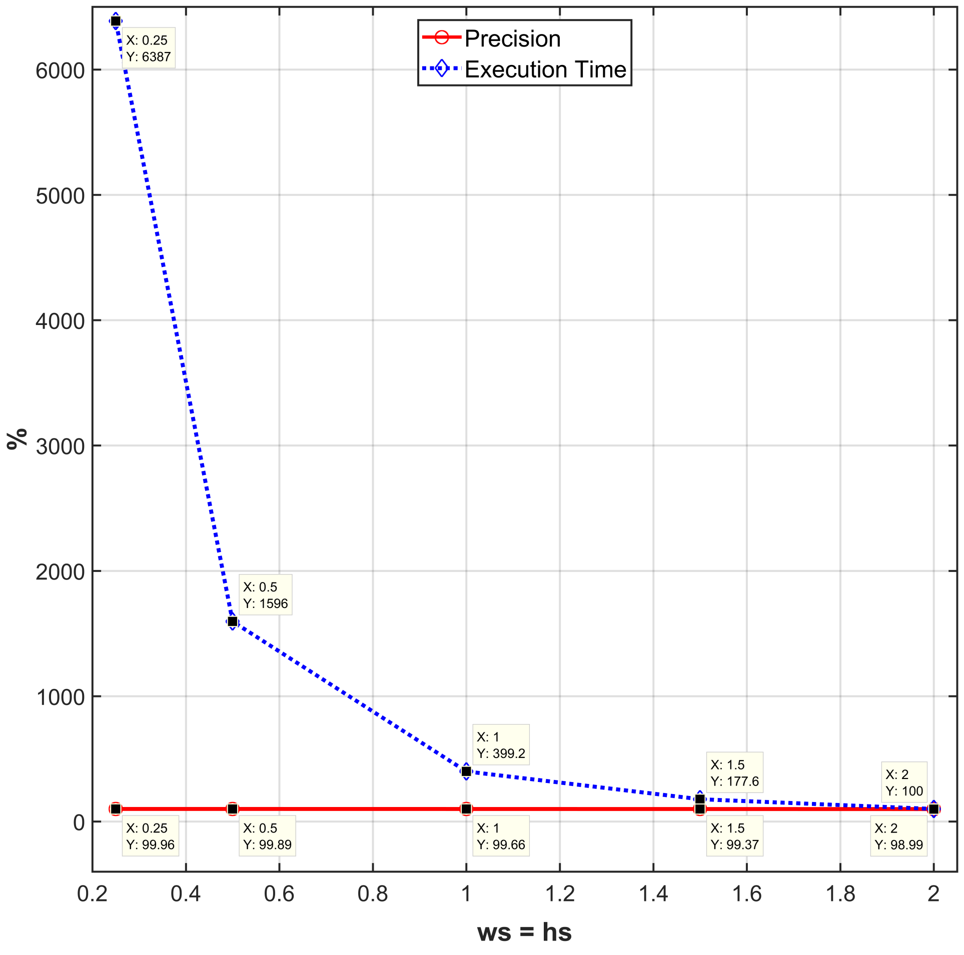


Figure 5.12: Precision vs. Execution Time.

These results show that the approximate approach presents results with a very low error rate, although having lower computational costs when compared with the proposed accurate algorithm. For the scenario of WVSN comprised of resource constrained sensor nodes, the evaluated approximate algorithm can be a worth choice and it is used in this thesis.

This approach is incorporated into the area coverage computation of the methodology. The coverage information will be used to compose the NFC: whenever none of combinations of essential nodes covering the minimum area can be formed, the application will fail. This procedure is described in Algorithm 4, where the coverage area CA_{mb} of each combination of sensors is computed and tested.

Algorithm 4 starts verifying which monitoring blocks are covered by each visual sensor (Lines 1–12). For this, the procedure $isInsideTriangle()$ in Line 5 checks if the center of a mb is within of the FoV of a visual sensor vs. According to how a sensor vs covers a monitoring block mb, the function $cover()$ is updated (Lines 6–10). This information is used to proper compute the overall coverage area $CA_{mb}(VS')$ provided by a subset of visual sensor $VS' \subseteq VS$ (Line 14). Finally, the NFC is updated if the sensors in this subset are able to fulfill the application requirements together, that is, if they cover an area greater or equal to CA_{min} (Lines 15–17).

Algorithm 4: NFC

```
Data: nfc = NFC($CA_{min}$,VS,ws,hs);
Input: Minimum Coverage Area, List of sensors' parameters,width and height of monitoring
       blocks.
Output: Network Failure Condition.
1  foreach $vs \in VS$ do
2      foreach Monitoring Block mb do
           // Coordinates of the center of mb (See Figure 3.6).
3          $xc = mb.x1s + (mb.x4s - mb.x1s)/2$;
4          $yc = mb.y1s + (mb.y4s - mb.y1s)/2$;
5          inside = isInsideTriangle($xc$,$yc$,$FoV_{vs}$);
6          if inside then
7              $cover(mb, vs) = 1$;
8          else
9              $cover(mb, vs) = 0$;
10         end
11     end
12 end
13 foreach $VS' \subseteq VS$ do
14     $CA_{mb}(VS') = ws \cdot hs \cdot \sum_{k=1}^{M} \sum_{l=1}^{N} cover(mb_{k,l}, VS')$; // Equation 3.9
15     if $CA_{mb} \geq CA_{min}$ then
16         nfc.addExpression($VS'$);
17     end
18 end
```

5.8 Routing Analysis

Supposing that visual nodes are able to monitor the minimum coverage area, the resulting visual information must reach to the sink. This task is managed by the routing protocol, which imposes a set of communication rules in order to optimize aspects like power consumption, overhead, throughput and delivered messages. This way, the routing analysis defines the possible connections between nodes based on the selected routing protocol and nodes position. Notice that it is considered as a node both visual and scalar sensor nodes.

These connections are mapped into an adjacency matrix Adj, which consists in a square matrix that represents an abstraction of the network topology. Each position Adj_{pq} of the matrix represents the binary relation between the nodes associated to that position. That way, if there is a link connection between nodes p and q, then $Adj_{pq} = 1$, otherwise $Adj_{pq} = 0$. It is important to notice that $Adj_{pq} = Adj_{qp}$, $\forall p,q$. Figure 5.7 shows the network arrangement for different routing strategies. Notice that a different selection of routing protocols preserves the same node positions, but generates different topological arrangements, where each arrangement will generate a different adjacency matrix.

Algorithm 5 details how to proceed the routing analysis, starting by creating the adjacency matrix, based on an identity matrix with dimension equals to the number of nodes (Line 3). This means that each node is connected with itself. Then, the adjacency matrix will be updated according to the selected routing protocol. If the selected protocol is the DIRECT, it will be created a connection between each sensor node and the sink, as shown in Line 7. For that, it is supposed that each node have enough radio transmission power to directly communicate to the sink using the DIRECT protocol. On the other hand, if the selected protocol is the FLOODING, a connection between two sensor nodes p and q will be created (Line 14) if the distance between them is less than or equal to their radio communication range, *i.e*, if $d(p,q) \leq R_c$ (Line 13). In this case, it is important to remember that a given node will probably communicate with the sink through a sequence of message re-transmissions. Therefore, it can be considered a smaller radio communication range, which implies that each node can reduce its radio power, generating a smaller power consumption and a slower battery discharging.

5.9 Paths and Cut Sets Generation

Using the network topology described by adjacency matrix, the next step is to discover which nodes and links are involved in the communication between the sink and the nodes that perform the successful area monitoring, *i.e*, which nodes and links are responsible to route the information from the nodes belonging to the NFC to the sink. For this task, it is performed a Depth-First Search (DFS) in the adjacency matrix starting from the leaves (nodes in the NFC), until finding the root (sink).

Algorithm 6 creates a set of paths from each node belonging to the NFC. This is performed by the invocation of Algorithm 7 (Line 7), which recursively goes through the adjacency matrix

Algorithm 5: Routing

Data: Adj = Routing(protocol, R_c, $VS \cup SS$);
Input: Routing protocol, Radio range, List of all nodes and its parameters.
Output: Adjacency matrix of network.

```
1  $nodes = VS \cup SS$;
2  $nNodes$ = size($nodes$);
3  $Adj$ = eye($nNodes$); // Identity Matrix with dimension $nNodes \times nNodes$
4  switch $protocol$ do
5      case $DIRECT$ do
           // Index 1 is reserved for the sink node
6          for $p \leftarrow 2$ to $nNodes$ do
7              $Adj$(1,p) = 1; $Adj$(p,1) = 1;
8          end
9      end
10     case $FLOODING$ do
11         for $p \leftarrow 1$ to $nNodes$ do
12             for $q \leftarrow p$ to $nNodes$ do
13                 if $d(nodes[p], nodes[q]) \leq R_c$ then
14                     $Adj(p,q)$ = 1; $Adj(q,p)$= 1;
15                 end
16             end
17         end
18     end
19 end
```

to find new neighbor nodes (Line 7) and adding these nodes and its links (Lines 9 and 10) until finding the sink (Line 3). In order to avoid cycles, nodes that have already been selected for the path are ignored (Line 8).

Each found path is called a *cut set*. In a Fault Tree analysis, a cut set is a subset of events whose simultaneous occurrence leads to the occurrence of the TOP event. Some authors go further (Silva et al., 2012) and find the *minimal cut set*, that is a cut set that does not contain any other cut set. A minimal cut set is important to reduce the number of mathematical operations required to compute the TOP event, which can be significant in a large FT. On the other hand, since the available computational tool (SHARPE) already cope with this issue, this task is ignored in this work.

An important aspect to remark is the repeated occurrence of network elements (nodes and links) in the cut sets. This is a crucial aspect for the proper analysis and solution of the FT model regarding basic and repeated events. That way, in Line 6 of Algorithm 6 and Lines 11 and 12 of Algorithm 7, each occurrence of a network element is computed in order to verify during the FT generation (Algorithm 8) whether this element must be assigned to a basic or repeated event.

Algorithm 6: Paths Discovery

Data: paths = PathsDiscovery(Adj[][], NFC);
Input: Adjacency Matrix, Network Failure Condition.
Output: Set of All Paths from nodes that interfere on NFC.

```
linkOccurrences = [];
nodeOccurrences = [];
// Find all paths between node i and sink
foreach node i in NFC do
    cPath ← null;
    cPath.addNode(i);
    nodeOccurrences.add(i);
    paths.append( DFS(null, Adj, i, cPath, linkOccurrences, nodeOccurrences) );
        // Alg. 7
end
return paths;
```

Algorithm 7: DFS

Data: paths = DFS(paths, Adj, cNode, cPath, linkOccurrences, nodeOccurrences);
Input: Collection of Paths, Adjacency Matrix, Node to be inserted, Current Path, number of occurrences of links and nodes in the paths.
Output: New Collection of Paths.

```
// Number of nodes is the dimension of squared Adjacency Matrix
nNodes = size(Adj);
for i ← 1 to nNodes do
    // Did you find the sink?
    if cNode == 1 then
        paths.add(cPath);
        return paths;
    end
    // Searching for neighbor nodes
    if Adj[cNode][i] == 1 then
        if ∼pathContains(cPath, i) then
            cPath.addLink(cNode,i);
            cPath.addNode(i);
            linkOccurrences.add(cNode,i);
            nodeOccurrences.add(i);
            paths = DFS(paths, Adj, i, cPath, linkOccurrences, nodeOccurrences);
                // Alg. 7
            cPath.removePathAndLink(cNode,i);
        end
    end
end
return paths;
```

5.10 Fault Tree Model Generation

Following the methodology flow, the model for the whole system must be generated according to the SHARPE language and syntax. For this purpose it is necessary to describe the system fault tree and the model of each network element (hardware, battery or link), which consists in a CTMC. This is performed by Algorithm 8, which looks into each path found in the previous step and, for each network element, the algorithm writes the corresponding SHARPE code to a text document that will be the input of SHARPE. This code describes the structure of the Fault Tree and the CTMC models. According to Figure 5.9, this structure is an OR-gate per node (device) including battery and hardware events as inputs (Line 8), and an OR-gate per path, including node and link events as inputs (Line 12). Then, the structure is completed by a AND-gate including each path event as input (Line 14). Finally, the generated code is organized in a text document, getting each reference between hardware, battery and link events and generating their CTMC code (Line 16). The availability events of hardware and link, according to Figures 5.3 and 5.6, and availability events of battery, according to Figure 5.5, are associated to their CTMCs in Lines 6, 10 and 7, respectively. At this point the repeated occurrence of network elements (nodes and links) in the cut sets is analyzed in order to determine whether these elements must be assigned to basic or repeated events.

Notice that, in order to use this methodology to perform a system reliability evaluation, it is only necessary to remove the repair activities from the CTMC models, which means to set the repair rates μ_{hw}, μ_{bt} and μ_{lk} equal to zero.

5.11 Fault Tree Analysis and Data Output

The Fault Tree analysis is totally performed by the SHARPE tool, which provides a hierarchical modelling structure, facilitating the time-consuming tasks of describing and evaluating large and complex systems. SHARPE receives a text document with the Fault Tree description, the related CTMCs and the dependability attributes to be assessed for the evaluation period T, considering the discrete time step, t_s. First, SHARPE tool evaluates the individual CTMC models of hardware, battery and link and obtains their respective dependability function. Due to SHARPE hierarchical models, these functions are inserted as input events of Fault Tree gates, according to the described model. Then SHARPE evaluates the FT and returns another text document with the values of the dependability metrics (*e.g.* availability) for each instant of time. These values are stored and graphically represented for the user. The integration with the SHARPE is fully automated by the implemented framework.

5.12 Results and Discussion

In this section we present some results obtained when using the proposed methodology to evaluate WVSNs dependability. The purpose of this section is to show how to use the methodology. We do

Algorithm 8: Fault Tree Generation

Data: FaultTree = FaultTreeGeneration(λ_{hw}, μ_{hw}, λ_{bti}, μ_{bt}, λ_{lk}, μ_{lk}, btStages, Adj[][], NFC);
Input: Failure and Repair Rates of Hardware, Battery and Link; Battery Stages; Adjacency Matrix, Network Failure Condition.
Output: Fault Tree associated to the Network Failure Condition.

```
paths = PathsDiscovery(Adj, NFC); // Alg.6
foreach node n in NFC do
    foreach path p in paths, from n to sink do
        foreach element item in path p do
            if isNode(item) then
                hw = Hardware_MarkovChain(item, $\lambda_{hw}$, $\mu_{hw}$, nodeOccurrences);
                bt = Battery_MarkovChain(item, $\lambda_{bti}$, $\mu_{bt}$, btStages, nodeOccurrences);
                input = OR(hw, bt);
            else if isLink(item) then
                input = Link_MarkovChain(item, $\lambda_{lk}$, $\mu_{lk}$, linkOccurrences);
            end
            path_Gate = OR(path_Gate, input);
        end
        AND_Gate = AND(AND_Gate, path_Gate);
    end
    NFC.subs(AND_Gate, n);
end
return NFC;
```

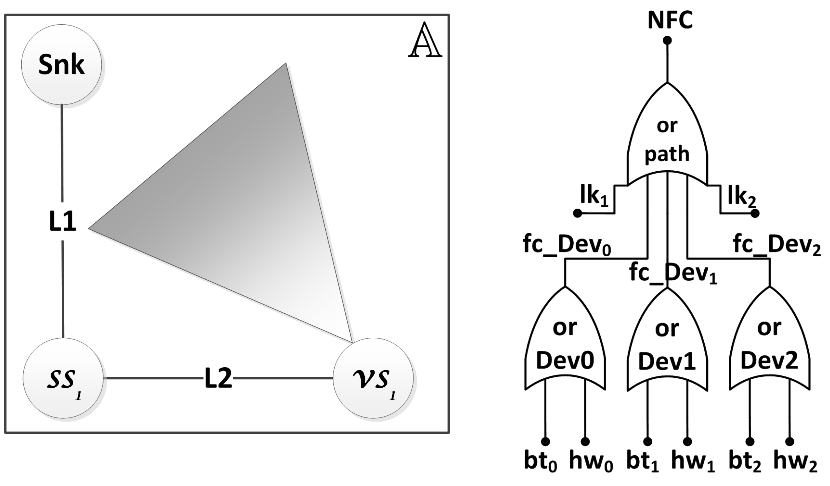
not intend to analyze a specific scenario. This would require a detailed parameters setting, which is discussed in Chapter 6. That way, some examples are given showing different types of analysis allowed by the proposed methodology. For all examples, it is assumed that each device uses the same model of battery, which is an alkaline battery (AA/R6) with c_0 = 3000 *mAh*, H = 25 *h*, I = 100 *mA*, η = 1.3 and DC = 50%.

For this case, it would take 25 hours to discharge the battery, considering an average operation current of 100 *mA*. Since we are considering a 50% duty-cycle, this will increase the operating time up to 50 hours. It is considered that a node cannot properly work with a battery capacity lower than $c_{\min} = 500$ *mAh*. So, the modelling of the battery by an approximation to a stochastic process is given as follows. Suppose that the battery discharges through $nStages = 4$ stages, and then, according to Section 5.3.2, each stage discharges $(c_0 - c_{\min})/nStages = 625$ *mAh*. This means that each stage duration $\tau_i = t_{i+1} - t_i$ $(i = 0, \dots, nStages - 1)$ can be found solving Equation 5.1 to the values $c_0 = c(t_0) = c(0) = 3000$ *mAh*, $c_1 = c(t_1) = 2375$ *mAh*, $c_2 = c(t_2) = 1750$ *mAh*, $c_3 = c(t_3) = 1125$ *mAh* and $c_4 = c(t_4) = 500$ *mAh*. That implies in $\tau_0 = 4.1235$ *h*, $\tau_1 = 6.0297$ *h*, $\tau_2 = 7.0465$ *h* and $\tau_3 = 7.8003$ *h*. So, according to Equation 5.2, λ_{bt0} = 0.1213, λ_{bt1} = 0.0829, λ_{bt2} = 0.0710 and λ_{bt3} = 0.0641. In this thesis, all failures, repair and discharging rates are measured by hour. Once discharged, we consider that it takes 2 hours to repair (replace or recharge) the battery, so μ_{bt} = 1/2 = 0.5. It is important to remark that increasing the number of battery stages results in a better approximation to the real battery discharging behavior (Bruneo et al., 2012).

With respect to the visual nodes and theirs parameters, we assume a viewing angle $\theta_{vs} = 60^\circ$ and a sensing radius $R_s = 150\ m$ for all considered visual sensors. Also, we consider that the radio communication range being $R_c = 180\ m$. These device parameters are used for the communication scenarios of Examples 5.12.1 and 5.12.2.

5.12.1 Example 1: Elucidating Scenario

In order to enlighten the methodology usage, first we executed an evaluation of a small network, composed by a sink node (*Snk*), a scalar node (ss_1) and a visual node (vs_1), as shown in Figure 5.13(a). The visual node is represented by a circle attached to a triangle, which is the camera's FoV. The system uses FLOODING protocol and considers the occurrence of 1 hardware failure per year and 1 link failure per 2 days, which implies hardware and link failure rates as $\lambda_{hw}01.1416 \times 10^{-4}$ and $\lambda_{lk} = 2.083 \times 10^{-2}$, respectively. Also, it is considered that it takes 72 hours to repair the hardware and 15 minutes for a link to be reestablished, which implies the hardware and link repair rates as $\mu_{hw} = 1.3894 \times 10^{-2}$ and $\mu_{lk} = 4$, respectively.



(a) (b)

Figure 5.13: (a) System topology and (b) its Fault Tree.

Since we have only one visual node, Algorithm 4 identifies node vs_1 as the unique responsible for the monitoring. Therefore, the network failure condition must be $NFC = vs_1$, *i.e.*, if the node vs_1 fails, the visual application fails. Then, the network topology is known by Algorithm 5, and it is represented by the following adjacency matrix:

$$mAdj = \begin{bmatrix} 1 & 1 & 0 \\ 1 & 1 & 1 \\ 0 & 1 & 1 \end{bmatrix} \tag{5.4}$$

This means that the sink (index 1 of the matrix) is connected only to the scalar node (index 2 of the matrix), whereas the scalar node is also connected to the visual node (index 3 of the matrix). This information is used by Algorithm 6 to find all paths from the NFC devices (device 2, in this case) to the sink, by a depth-first search (Algorithm 7). There is only one path with this feature, which is $vs_1 \rightarrow L2 \rightarrow ss_1 \rightarrow L1 \rightarrow Snk$, where $L1$ is the link connecting Snk and ss_1, and $L2$ is the link connecting ss_1 and vs_1. So, Algorithm 8 will generate a Fault Tree that represents this behavior, as shown in Figure 5.13(b), considering the mapping $Snk = Dev_0$, $ss_1 = Dev_1$ and $vs_1 = Dev_2$.

Notice that the logical diagram expressed in Figure 5.13(b) indicates that if any link (lk_1, lk_2) or device (fc_Dev_0, fc_Dev_1, fc_Dev_2) fails, the path will be interrupted and the application fails. Additionally, as this system has only one device associated to the NFC and one path from this device to the sink, then the AND-gate related to the failure condition of combinations (Figure 5.9(b)) has only one input, and the same happens to the OR-gate related to the network failure condition (TOP event – Figure 5.9(a)). These gates were replaced by a bypass connection in the Fault Tree structure.

The generated Fault Tree (SHARPE) model can be seen in Listing 5.1. Notice that each event (Lines 2–9) is represented by an availability function (probability function) related to a hierarchical model. For instance, in Line 9 there is a reference to the availability of the Markov Chain model (*BT_model_Dev*0) of the battery of the sink (*Dev*0). This model is described in Listing 5.2. Finally, Listing 5.3 shows how to invoke the availability evaluation of the system. The evaluation period is T = 150 hours, with a time step of t_s = 15 min (0.25 hours) (Line 2). All these models have been generated by Algorithm 8. Figure 5.14 shows the graphical output, indicating that after 50 hours of execution, the availability of the system is slightly higher than 86%.

Listing 5.1: Fault Tree model.

```
 1 ftree FTA_model(time)
 2     basic ev_Dev2_hw prob(exrt(time; HW_model_Dev2; $\lambda_{hw}$, $\mu_{hw}$))
 3     basic ev_Dev1_hw prob(exrt(time; HW_model_Dev1; $\lambda_{hw}$, $\mu_{hw}$))
 4     basic ev_Dev0_hw prob(exrt(time; HW_model_Dev0; $\lambda_{hw}$, $\mu_{hw}$))
 5     basic evL2 prob(exrt(time; LK_model_L2; $\lambda_{lk}$, $\mu_{lk}$))
 6     basic evL1 prob(exrt(time; LK_model_L1;$\lambda_{lk}$, $\mu_{lk}$))
 7     basic ev_Dev2_bt prob(exrt(time; BT_model_Dev2; $\lambda_{bt0}$, $\lambda_{bt1}$, $\lambda_{bt2}$, $\lambda_{bt3}$, $\mu_{bt}$))
 8     basic ev_Dev1_bt prob(exrt(time; BT_model_Dev1; $\lambda_{bt0}$, $\lambda_{bt1}$, $\lambda_{bt2}$, $\lambda_{bt3}$, $\mu_{bt}$))
 9     basic ev_Dev0_bt prob(exrt(time; BT_model_Dev0; $\lambda_{bt0}$, $\lambda_{bt1}$, $\lambda_{bt2}$, $\lambda_{bt3}$, $\mu_{bt}$))
10
11     or or_Dev2 ev_Dev2_hw ev_Dev2_bt
12     or or_Dev1 ev_Dev1_hw ev_Dev1_bt
13     or or_Dev0 ev_Dev0_hw ev_Dev0_bt
14     or path or_Dev2 evL2 or_Dev1 evL1 or_Dev0
15 end
```

Listing 5.2: Markov Chain model.

```
bind $\lambda_{bt0}$ $0.1213$
bind $\lambda_{bt1}$ $0.0829$
bind $\lambda_{bt2}$ $0.0710$
bind $\lambda_{bt3}$ $0.0641$
bind $\mu_{bt}$ 0.5

markov BT_model_Dev0($\lambda_{bt0}$, $\lambda_{bt1}$, $\lambda_{bt2}$, $\lambda_{bt3}$, $\mu_{bt}$) readprobs
    B0_Dev0 B1_Dev0 $\lambda_{bt0}$
    B1_Dev0 B2_Dev0 $\lambda_{bt1}$
    B2_Dev0 B3_Dev0 $\lambda_{bt2}$
    B3_Dev0 B4_Dev0 $\lambda_{bt3}$
    B4_Dev0 B0_Dev0 $\mu_{bt}$

    * Reward configuration defined:
    reward
        B0_Dev0 0
        B1_Dev0 0
        B2_Dev0 0
        B3_Dev0 0
        B4_Dev0 1
    end

    * Initial Probabilities defined:
    B0_Dev0 1
    B1_Dev0 0
    B2_Dev0 0
    B3_Dev0 0
    B4_Dev0 0
end
```

Listing 5.3: Availability evaluation.

```
func Availability(t) 1-tvalue(t;FTA_model;t)
    loop t,0,150,0.25
        expr Availability(t)
    end
end
```

5.12.2 Example 2: General Usage

This example sets-up a larger network in order to analyze the effects of routing protocols, links and battery discharges upon the availability evaluation. Also, it is analyzed the availability with respect to parameters variation (failure and repair rates). For this purpose, we consider the WVSN represented in Figure 5.15, with 5 visual nodes, 4 scalar nodes and one sink node. The monitored area

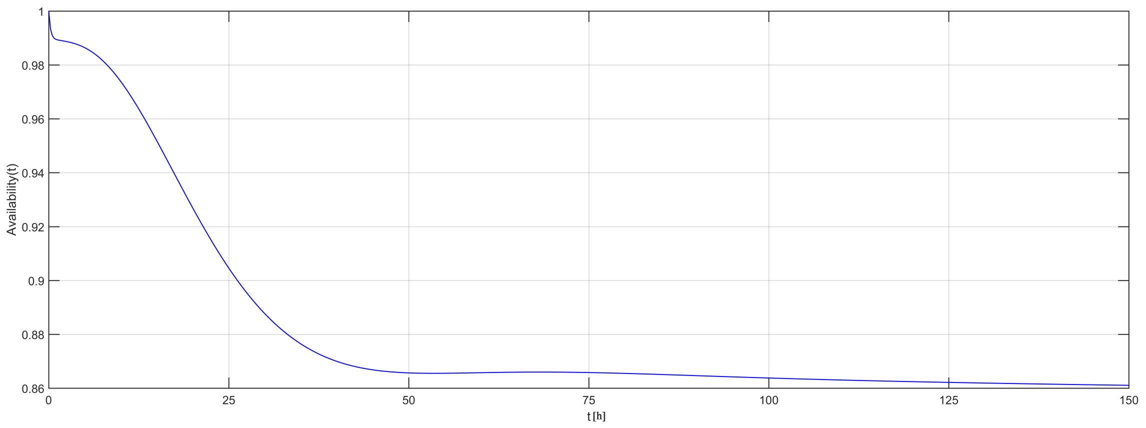


Figure 5.14: Graphical methodology output.

has $18 \times 10^4\ m^2$ and the minimum required coverage area is 20%, which is $CA_{min} = 36 \times 10^3\ m^2$. This implies the following network failure condition:

$$\text{NFC} = (vs_1 \wedge vs_2) \vee (vs_1 \wedge vs_3) \vee (vs_1 \wedge vs_4) \vee (vs_1 \wedge vs_5) \vee (vs_2 \wedge vs_3) \vee \\ \vee (vs_2 \wedge vs_4) \vee (vs_2 \wedge vs_5) \vee (vs_3 \wedge vs_4) \vee (vs_3 \wedge vs_5) \vee (vs_4 \wedge vs_5), \tag{5.5}$$

where operator "$\wedge$" indicates an **AND** gate and operator "$\vee$" indicates an **OR** gate. The hardware and link failure rates are $\lambda_{hw} = 1.1416 \times 10^-4$ and $\lambda_{lk} = 0.0417$, respectively, which means 1 hardware failure per year and 1 link failure per day, and hardware and link repair rates of $\mu_{hw} = 0.0208$ and $\mu_{lk} = 2$, respectively, which means 1 day to repair the hardware and 30 minutes to reestablish a link.

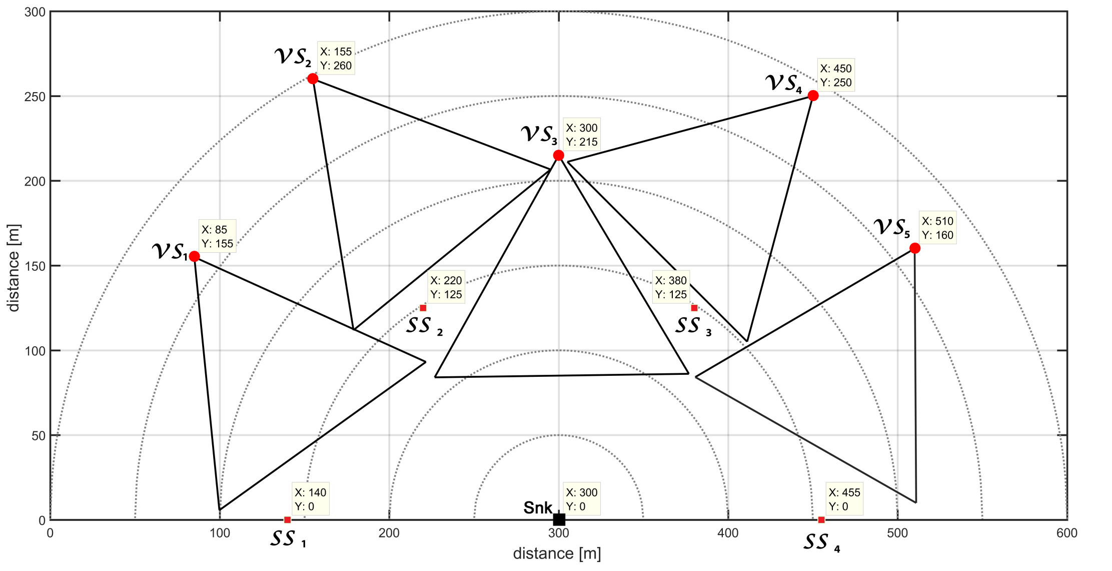
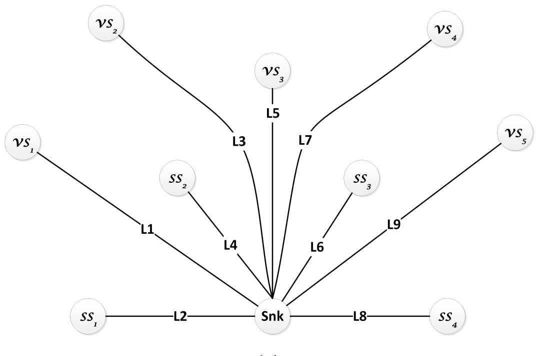


Figure 5.15: Area coverage in Example 2.

First we analyzed the effect of the routing protocol. Figure 5.16 shows the network topology for DIRECT and FLOODING protocols. Figure 5.17 shows the availability evaluation of those



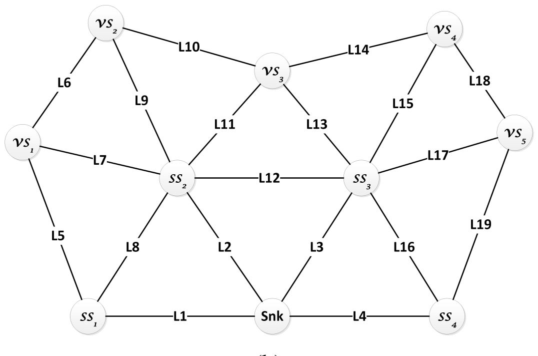


Figure 5.16: Network arrangement to routing protocols (a) DIRECT and (b) FLOODING.

topologies. As it was expected, the FLOODING protocol presents higher availability ($\approx 94\%$) than the DIRECT protocol ($\approx 92\%$) due to the fact that there are multiple possibilities to reach the sink. This is an interesting result, because besides providing a higher availability, FLOODING protocol also allows each visual node to reduce its radio transmission power and therefore to save battery. However, this methodology is not able to measure such power consumption savings yet. For this purpose, the battery modelling should be directly integrated to the routing protocol modelling, in order to associate the average discharging current according to the selected routing strategy. This task could be performed by simulation, which is out of the scope of this work.

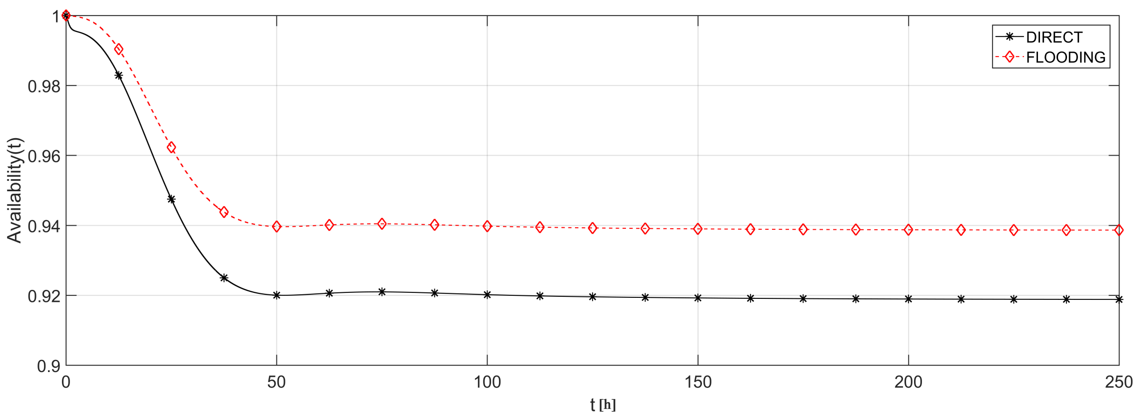


Figure 5.17: Availability of network from Example 2.

We also analyzed the effects of link and battery failures upon the system availability, using each of the routing protocols. To cope with these analysis, the system availability is evaluated in four different ways: (*i*) considering only hardware failures to provide a comparison basis; (*ii*) considering hardware and link failures to highlight just the link failure effects; (*iii*) considering hardware and battery failures to highlight just the battery failure effects; and (*iv*) considering hardware, link and battery failures together to obtain the overall system behavior. Figure 5.18 plots the result of that analysis based on DIRECT protocol and Figure 5.19 based on FLOODING protocol.

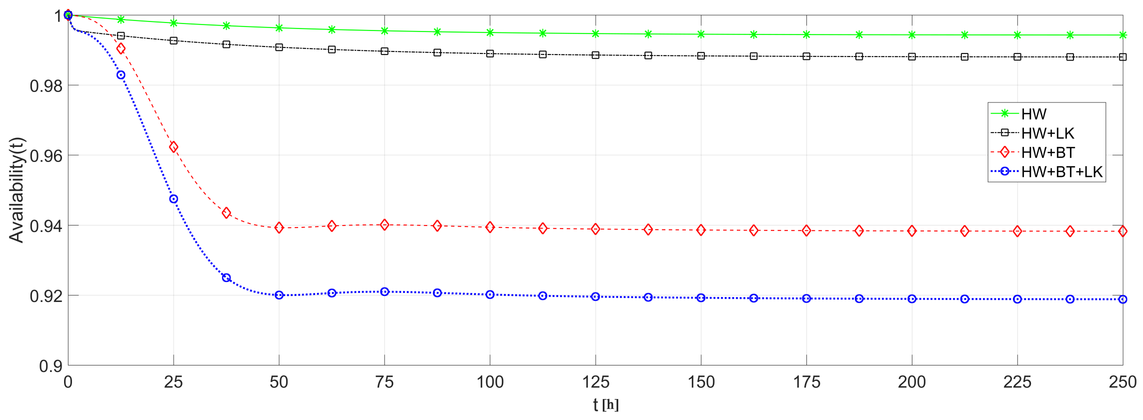


Figure 5.18: Comparison of effect of battery and link availability on network from Example 2, according to DIRECT protocol.

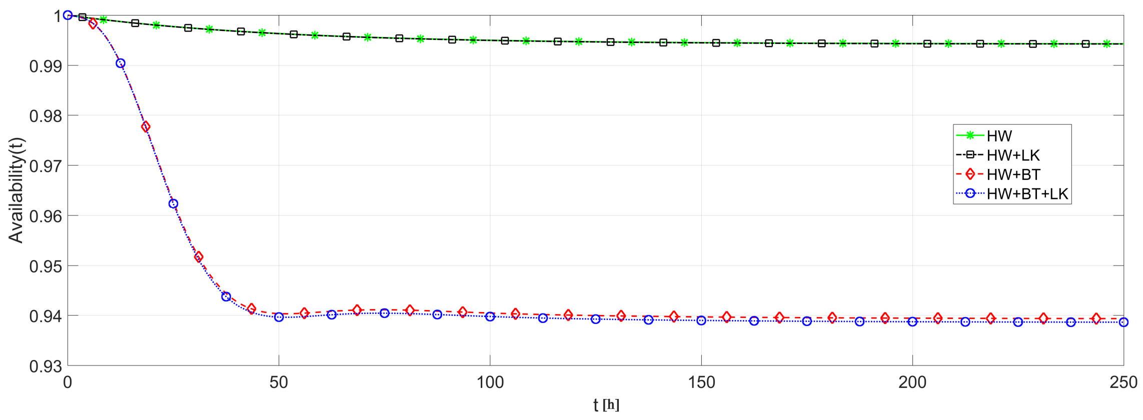


Figure 5.19: Comparison of effect of battery and link availability on network from Example 2, according to FLOODING protocol.

From Figures 5.18 and 5.19 it is possible to notice that the battery discharge is the component that causes the major effect upon the system availability, since the battery discharges faster than the other failures occurrences. On the other hand, link failures have smaller impact due to its transient behavior, which strongly depends on the routing protocol. Notice that link failures barely interfere in a network with FLOODING protocol, while it presents a considerable effect upon a network with DIRECT protocol. This is due to the fact that in a network with FLOODING protocol a lost path is quickly replaced by another one.

5.12.3 Example 3: System Design Assistance

Finally, we present an example of how to use the proposed methodology to guide the steps in the system design phase. In this case, we considered the network of Example 2, assuming that the cost of duplicating the quality of the battery is similar to the cost of reducing three times its repair time. This assumption means that the cost of buying a higher capacity battery is similar of buying a faster battery charger. In that case, the new battery would present failure rates with half of value from Example 2 or repair rates three times higher. So, which is the best decision?

Figure 5.20 provides information resulting from that analysis, comparing the availability of the system from Example 2 with the availability of the system with either the new battery, or the new repair approach. We considered similar failure and repair rates from Example 2, with FLOODING protocol.

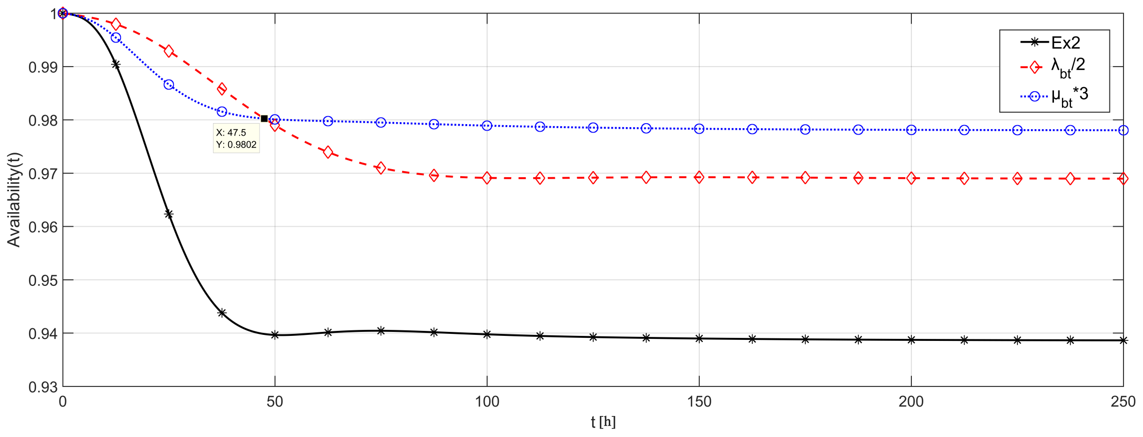


Figure 5.20: Availability Evaluation for system design.

Both changes imply a significant improvement of the system availability, where during the first 47.5 hours the new battery provides a higher availability, while the new battery repair approach provides a higher steady-state availability. If it is required that the application runs for less than 2 days, then is preferable to invest in a better battery. Otherwise, a faster battery repair will allow the application to be available during more time.

5.13 Conclusion

This chapter presented the proposed methodology for dependability evaluation on WVSN. This methodology follows a failure and fault models that classifies coverage failures as incorrect computation failures, and node and link failures as crash failures. Those failures can be eventually caused by faults with a nature non-deliberated, non-malicious, accidental and operational, natural or human-made, internal or external, transient or permanent.

The model of nodes and links based on these failures and faults were presented and integrated to the methodology. Each step of the methodology has been discussed and detailed by specific algorithms. Finally, some examples were given, showing how to properly use the methodology as a tool for assessment, comparison and design aid.

In the next chapter, we focus on visual aspects that affect system dependability and can be handled by the methodology. Practical aspects are approached in order to illustrate how useful is the methodology in realistic and more complex applications.

Chapter 6

Visual Aspects

This chapter discusses the relation of visual aspects and their impact on the system dependability. It is shown how to use dependability assessment information to guide the redeployment of a network, managing the network configuration to keep dependability requirements. Furthermore, dependability is assessed considering the quality of monitoring, *i.e.*, the ability of gathering sharpen images. This ability also varies according to regions eventually occluded by obstacles. Finally, practical evaluation issues are discussed and simulation results are presented to show how the proposed methodology can be applied in realistic scenarios to assess and enhance their dependability.

6.1 Optimization

Although several approaches have provided network deployments with dependability guarantees, sometimes the monitored environment or the application configurations can change during the network operation. These modifications can violate the dependability requirements and demand a network redeployment in order to keep those guarantees. We propose a WVSN redeployment guided by the optimization of the application dependability, considering changes on cameras' orientations. That redeployment is based on a pseudo-greedy heuristic, since it tries to find a global maximum while searching local solutions, although it uses some global information recovered in a lightweight way. The redeployment considers changes only on cameras' orientations, in order to demand no extra resources from network components.

6.1.1 Dependability Analysis

Considering the area coverage problem addressed in this thesis, application dependability evaluation according to the proposed methodology is directly related to visual coverage failures. These failures can be indirectly affected by battery, communication links and hardware failures, and also consider the impact of different routing strategies. In this case, in order to improve dependability, we could relocate the sensor nodes in such way that the new configuration can provide higher link quality or better network connectivity, leading to an increase of the amount of gathered visual information. We could also replace or add more reliable sensor nodes to probably postpone hardware failure occurrences. Moreover, relocation of sensor nodes could favour some routing strategy, *e.g.* allowing the selection of lower radio transmission power that could decrease energy consumption.

In this thesis we adopt a conservative approach, considering that the network is static, that is, the sensor nodes do not move. In this case, the cameras can rotate, assuming a new orientation that better fulfills the visual monitoring requirements of an application. With this choice it is desired to perform redeployments that do not demand extra resources from the network components, such as hardware replacement, topology alteration or excessive energy consumption.

In this context, it is necessary to analyse the monitoring scenarios considering qualitative and quantitative aspects of visual information. For instance, a camera will tend to gather little information if it is orientated with its FoV outside of a MA. On the other hand, a camera could be rotated to monitor an uncovered area or to increase the coverage redundancy of an already covered area, which occurs when the FoV of two or more cameras present an intersection, and a region is covered more than once. In another possible configuration, all cameras' FoV can be inside of a MA, but with high coverage redundancy, *i.e.*, some cameras *view* a high percentage of the same region. This generates a high amount of information, but with low aggregated value.

Once established that only the camera orientation is manageable, this means that the increase of the application dependability directly depends on the proper adjustment of the network failure condition. Its is important to remember that the NFC represents the failure combinations that lead to the lack of visual data and consequently to the application failure. Actually, without visual information from these nodes, the information gathered by the rest of the network is not enough to compose the required minimum area to be monitored. The cameras' orientations impact the total gathered visual information, which is used to define the NFC and ultimately the overall dependability.

The application dependability is as high as the number and variety of nodes combinations in the NFC. This is achieved by increasing the number of non-redundant nodes covering the minimum required area, potentially using a higher number of nodes. That way, if some nodes fail, the remaining combinations will probably still meet the coverage requirements. Another way to achieve high dependability is sharing high coverage redundancy among some nodes covering the minimum required area. Doing so, if some nodes fail, there may be some backup nodes to gather the same information of the failed ones, assuring that the application meets its expected coverage requirements.

Ideally, the best scenario is achieved combining these two approaches. However, the second approach is difficult to guarantee, since some nodes can have no redundancy correspondence. Therefore, in this thesis we focus on exploiting only the first approach, enhancing dependability indirectly through the increasing of visual coverage, and considering that each visual sensor may take one of a finite set of disjoint orientations. This assumption is aimed at making this problem tractable, still assuring near-optimal area coverage optimization (Costa et al., 2017b). Additionally, the changes in the visual coverage configurations are followed by dependability assessments, bringing an important contribution to this area.

6.1.2 Optimization Algorithm

- **Greedy**

First, a Greedy algorithm is presented to be used as comparison basis for the Pseudo-Greedy algorithm. A Greedy approach takes into account that a reasonable and feasible way to optimize the network is to compute the best orientations for each sensor individually, aiming at the maximization of covered monitoring blocks. A classical greedy heuristic looks for a global optimization, handling only local data. This is due to the complexity of dealing with global information, such as area coverage. That way, for a greedy algorithm, a visual sensor vs may assume S_{vs} different orientations (sectors), where each sector has the same angle γ_{sec}. The value of γ_{sec} can be computed by the number of sectors S_{vs}, according to Equation 6.1.

$$\gamma_{sec} = \lfloor 360^o / S_{vs} \rfloor \tag{6.1}$$

Therefore, for each sector $s = 1, ..., S_{vs}$, the possible new orientation of vs will be α^s_{vs}, according to Equation 6.2 (as shown in Figure 6.1, with $\gamma_{sec} = \theta_{vs} = 60^o$), being α_{vs} the original orientation of vs.

$$\alpha^s_{vs} = (s-1) \times \gamma_{sec} + \alpha_{vs} \tag{6.2}$$

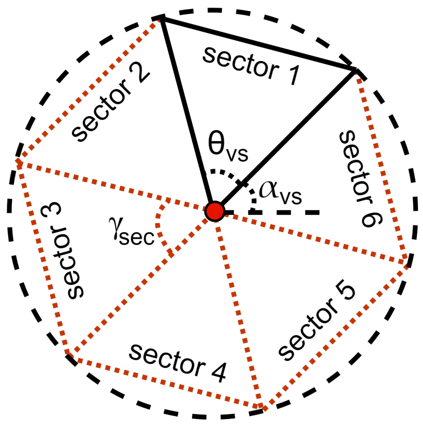


Figure 6.1: Sectors in the Greedy and Pseud-Greedy algorithms.

The proposed Greedy approach is based on individually testing which orientation provides the highest area coverage for each single visual sensor, then redeploying the sensor for that orientation. In this case, each sensor node will be re-orientated in order to compute the highest CA_{mb} and then the dependability evaluation is executed in order to verify any dependability improvement. Actually, this is a simple way to improve the value of CA_{mb}, while keeping lower computational costs as compared with other approaches, although greedy algorithms may provide sub-optimal results since they only evaluate local information. The proposed Greedy approach is detailed in Algorithm 9. It is worthy to note that the coverage area computed in Line 16 is related to a single visual sensor node vs, regarding the MB covered by vs computed in Lines 9 – 15.

Algorithm 9: Greedy optimization algorithm

Data: $\{$***VS***, dep$\}$ = Greedy(***VS***, ***MA***, S_{vs});
Input: List of sensors, monitoring area parameters and number of sectors.
Output: A set of reoriented visual sensors ***VS*** and dependability of optimized network.

```
foreach visual sensor vs in sensors set VS do
    angles = [];
    $\gamma_{sec} = \lfloor 360^o / S_{vs} \rfloor$;
    $\alpha' = vs.\alpha$;
    foreach sector $s = 1, ..., S_{vs}$ do
        MB = [];
        $\alpha^s_{vs} = (s-1) \times \gamma_{sec} + \alpha'$;
        // Re-orientate vs with $\alpha^s_{vs}$
        $vs.\alpha$ = $\alpha^s_{vs}$;
        foreach Monitoring Block $mb_{k,l}$ do
            if isCovered($mb_{k,l}$, vs) then
                MB[k,l] = 1;
            else
                MB[k,l] = 0;
            end
        end
        $CA_{mb}(vs)$ = $ws \times hs \times$ sum(MB);
        angles.add($\{\alpha^s_{vs}$, $CA_{mb}(vs)\}$);
    end
    $vs.\alpha$ = angles.getMaxCAmb();
end
dep = dependabilityEvaluation(VS);
return {VS, dep};
```

- **Pseudo-Greedy**

It is described in Algorithm 10 the proposed Pseudo-Greedy approach to optimize the application dependability through a coverage optimization. This algorithm searches for the orientation of each camera that decreases the area coverage redundancy. A classical greedy heuristic looks for a global optimization, handling only with local data. However, considering the subdivision of a monitoring area into several monitoring blocks, the complexity of dealing with global information, such as the total area coverage, is polynomial (as shown in Section 5.7.1), allowing us to consider the area coverage of the sensor's vicinity.

The algorithm searches for orientation of nodes that provides less redundancy, resulting in more variety of nodes combinations in the NFC. It begins by mapping the covered region from each sensor node, which is given by the monitoring blocks inside of the camera's FoV of the considered sensor (Lines 2 to 10). With this information, the next step is to verify the most likely orientation α^s_{vs} to each sensor node (Line 17), considering a finite set of disjoint orientations organized into S_{vs} sectors. This assumption is aimed at making this problem tractable, still assuring near-optimal optimization, as mentioned before.

Algorithm 10: Pseudo-Greedy optimization algorithm

Data: $\{\boldsymbol{VS}$, dep$\}$ = pseudoGreedy($\boldsymbol{VS}$, $\boldsymbol{MA}$, S_{vs}, ε);

Input: List of sensors, monitoring area parameters, number of sectors and error between iterations.

Output: A set of reoriented visual sensors $\boldsymbol{VS}$ and dependability of optimized network.

```
MB = [];
foreach sensor vs in sensors set VS do
    foreach Monitoring Block $mb_{k,l}$ do
        if isCovered($mb_{k,l}$, vs) then
            MB[$k,l,vs$] = 1;
        else
            MB[$k,l,vs$] = 0;
        end
    end
end

i=0; angles = [];
while $angles_{Err} > \varepsilon$ && $i$++ $< |VS|$ do
    foreach sensor vs in sensors set VS do
        $\gamma_{sec} = \lfloor 360^o / S_{vs} \rfloor$; $\alpha' = vs.\alpha$;
        foreach sector $s = 1, ..., S_{vs}$ do
            // Re-orientate vs with $\alpha_{vs}^{s}$
            $\alpha_{vs}^{s} = (s-1) \times \gamma_{sec} + \alpha'$;
            vs.$\alpha$ = $\alpha_{vs}^{s}$;
            update($MB[:,:,vs]$);
            $tmpMB = MB[:,:,vs] - \sum_{vs' \in VS/vs} MB[:,:,vs']$;
            // unique covered MB
            qtt = count( $tmpMB > 0$ );
            angles.add($\{\alpha_{vs}^{s}$, qtt$\}$);
        end
        vs.$\alpha$ = angles.getMaxQtt();
        update($MB[:,:,vs]$);
    end
    angles$_{Err}$ = $abs$(angles($i$) − angles($i-1$));
end

dep = dependabilityEvaluation($VS$);
return $\{VS$, dep$\}$;
```

The MBs covered by a sensor *vs* in its new orientation are computed (Line 18) and it is counted the number of MBs that are not covered by any other sensor instead of *vs* (Lines 19 and 20). The number of unique covered MBs in a given orientation is stored in Line 21. This procedure is repeated to every disjoint orientation. Then, the orientation that results in more unique covered MBs is taken by sensor *vs* (Line 23) and this heuristic is repeated for the remaining nodes.

This whole process is repeated until the orientations converge to an infinitesimal error between iterations, or if the number of iterations is equal to the number of sensor nodes. This is due to the fact that each iteration will best adjust only one sensor node. Finally, the dependability evaluation is executed once in order to verify any dependability improvement.

6.1.3 Optimization Results

In this section, numerical results are presented comparing the classical greedy approach with the proposed pseudo-greedy heuristic, through two different scenarios.

First, a specific WVSN with 10 sensor nodes and a sink is presented. Nodes are redeployed according to both greedy and pseudo-greedy heuristic in order to compare the area coverage and the dependability of the redeployed networks. As the second experiment, both approaches are applied to several random deployed networks in order to measure the average increase in coverage area and dependability from these approaches. In both scenarios dependability is assessed in terms of availability.

Since the purpose of the two presented experiments is a comparison between the heuristics, we do not focus on a detailed parameters setting, which is discussed later in this chapter. The cameras parameters are $R_s = 150$ units of distance (*u.d.*), $\theta_{vs} = 60°$, $w = 500$ *u.d.*, $h = 500$ *u.d.*, $ws = hs = 8.5$ *u.d.* and $\beta = 0°$. The monitored area has $\mathbb{A} = 2.5 \times 10^5$ units of area (*u.a.*) and the minimum required coverage area is defined according to the size of the network, being the area corresponding to the sum of FoV of half of sensor nodes in each evaluated network, *i.e.*, $CA_{min} = 0.5 \times |VS| \times FoV/A$. To cover this area we consider sensor nodes with similar characteristics to the ones presented in Chapter 5. That way, the radio communication range is $R_c = 180$ *u.d*. The hardware and link failure rates are $\lambda_{hw} = 1.1416 \times 10^{-4}$ and $\lambda_{lk} = 0.0417$, respectively, which means 1 hardware failure per year and 1 link failure per day. Moreover, the hardware and link repair rates is $\mu_{hw} = 0.0208$ and $\mu_{lk} = 2$, respectively, which means 2 day to repair the hardware and 30 minutes to reestablish a link. Considering a battery with 4 discharging stages, the battery failure rates are $\lambda_{bt0} = 0.1213$, $\lambda_{bt1} = 0.0829$, $\lambda_{bt2} = 0.0710$ and $\lambda_{bt3} = 0.0641$. Once discharged, we consider that it takes 2 hours to repair (replace or recharge) the battery, so $\mu_{bt} = 1/2 = 0.5$. The evaluation period is $T = 150$ hours, with a time step of $t_s = 15$ min (0.25 hours).

Considering all these parameters and the WVSN deployment presented in Figure 6.2, a classical greedy approach redeployment generates the network in Figure 6.3(a), while the pseudo-greedy redeployment generates the network in Figure 6.3(b). The covered area of the initial deployment is 7.65×10^4 *u.a.* ($\approx 30\%$ of MA), while the classical greedy redeployment presented a covered area of 7.79×10^4 $u.a.$ ($\approx 31\%$ of MA), and the pseudo-greedy redeployment presented a covered area of 9.68×10^4 $u.a.$ ($\approx 39\%$ of MA).

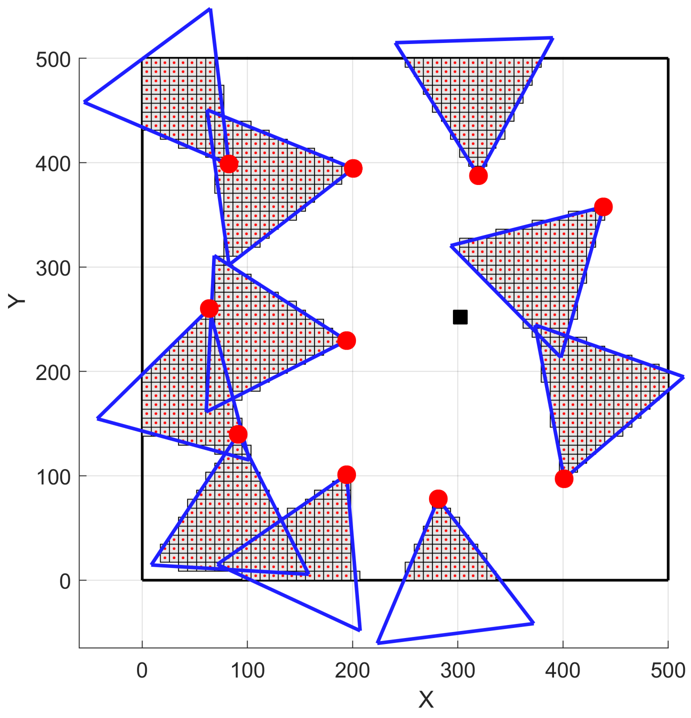


Figure 6.2: Initial deployment of WVSN.

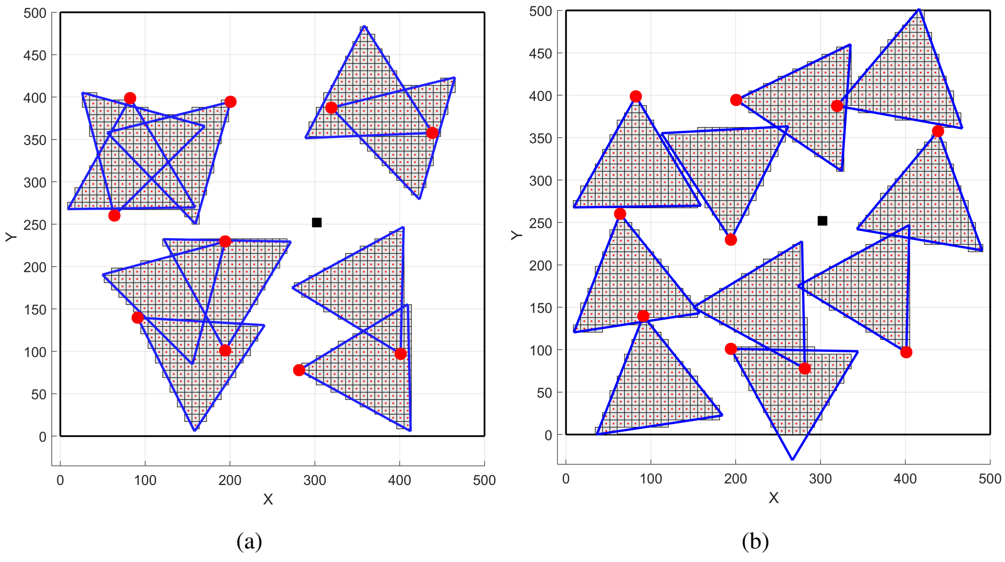


Figure 6.3: (a) Greedy and (b) Pseudo-Greedy redeployment of the WVSN.

As it was discussed in Section 6.1.1 and shown in Figure 6.4, the gain in area coverage reflects in gain of availability. In this case, the availability of the initial network tends to $\approx 88\%$, the availability of the greedy redeployed network tends to $\approx 89\%$, and the availability of the pseudo-greedy redeployed network (proposed approach) tends to $\approx 95\%$.

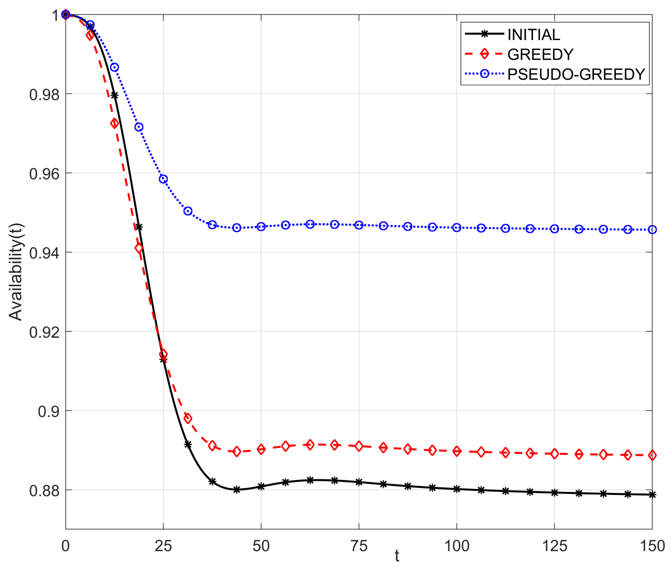


Figure 6.4: Availability comparison.

In order to confirm the trend of better results of the pseudo-greedy approach over the "classical" greedy algorithm, 300 simulated WVSN were randomly deployed, varying the number of sensor nodes in 5, 10 and 15, being 100 networks in each class. Table 6.1 shows the comparative results between the average gain (jointly with the standard deviation) in coverage area and availability when redeploying the networks by greedy and pseudo-greedy approaches. The average gain of coverage area (CA_{avg}) and availability (A_{avg}) are computed according to Equation 6.3, being CA_{int} and A_{int} the coverage area and the availability of the initial network, respectively, CA_{rdp} and A_{rdp} the coverage area and the availability of the redeployed network, respectively, and *sim* the number o simulations (100 for each class of networks).

$$CA_{avg} = \frac{\sum\left(\frac{CA_{rdp}-CA_{\text{int}}}{CA_{\text{int}}}\right)}{sim} \tag{6.3a}$$

$$A_{avg} = \frac{\sum\left(\frac{A_{rdp}-A_{\text{int}}}{A_{\text{int}}}\right)}{sim} \tag{6.3b}$$

Table 6.1: Area and availability comparison.

# Nodes	Area		Availability	
	Greedy	Pseudo-Greedy	Greedy	Pseudo-Greedy
5	$12.0\% \pm 17.6\%$	$23.0\% \pm 17.8\%$	$2.9\% \pm 5.8\%$	$4.2\% \pm 5.5\%$
10	$18.3\% \pm 17.0\%$	$42.8\% \pm 17.7\%$	$12.0\% \pm 13.7\%$	$17.7\% \pm 14.4\%$
15	$21.7\% \pm 13.3\%$	$52.3\% \pm 14.0\%$	$26.6\% \pm 19.1\%$	$35.1\% \pm 21.0\%$

These results show better performance of the proposed pseudo-greedy approach over the considered greedy reference algorithm, providing a higher area coverage improvement and consequently a higher availability improvement. In all executions, the pseudo-greedy heuristic increased the area coverage. Regarding the availability level, in some executions this parameter was not increased, but was just kept at least.

It is important to remark that the standard deviation values seem relatively high. This is due to the fact that, for the sake of tests, the networks were randomly deployed and thus they presented sometimes very low initial area coverage and availability level. This provides a lot of room for improvement. In these cases, the availability of the redeployed networks are too discrepant of the mean executions.

The presented results emphasize that a redeployment method guided by dependability metrics is an important achievement for flexible and dynamic safety critical systems, such as industrial networks, where neglected requirements may result in severe economical and safety consequences.

6.2 Visual Coverage Failures

Visual Coverage Failures (VCF) are addressed in this thesis as an important element of dependability, since they are related to the inability of the sensors to perform sufficient area coverage. In fact, VCF are resulted from total or partial lack of visual information retrieved from cameras, with different particularities depending on the way the environment is perceived. Two examples of main causes of visual failures that will affect the retrieved visual data and, consequently, the achieved dependability of an application are: (1) QoM due to distance of view and (2) occlusion. The QoM is associated to the sharpness of the retrieved visual data jeopardized by the distance of view (Shi et al., 2019). In a different way, occlusion will be produced by obstacles, potentially reducing a camera's Field of View and indirectly impacting the overall area coverage and monitoring quality. Both types of impairments may simultaneously impact visual sensor nodes and thus they will be modelled as the main causes for coverage degradation. The respective models will be incorporated to the dependability evaluation methodology in order to build a more comprehensive and realistic tool.

6.2.1 Quality Monitoring

Besides the coverage area, another important aspect related to visual sensor networks is the subjective perception of quality of monitoring. In this thesis, the QoM will be related to how "good" is the visual data definition that a camera can provide for a region located on a certain distance dov (distance of view). Actually, it is considered that the farther the monitored region is from the camera, the lower is the level of details related to that region in captured images, and consequently, the lower should be the amount of visual information extracted from that region. This means that the quality of captured images decreases as the distance from that camera increases, which lead us to consider different quality levels for the FoV of a camera. In other words, a FoV can be perceived as an area with different associated levels of monitoring quality over it.

The problem of quality of monitoring regarding area coverage was initially addressed in (Tao et al., 2017), but imposing several restrictions to the network deployment (position, orientation, viewing angle) and, as a consequence, to the network dependability. In a different way, we circumvent these constraints proposing a new approach for assessment of the quality of monitoring in WVSN, defining new QoM-based metrics. The proposed metrics, defined as the Area Quality Metric (*AQM*) and its variations, are types of QoM metrics that can be used for optimizations, comparisons or exploitation of different quality aspects, such as redundancy and dependability.

That way, as depicted in Figure 6.5, a sensor node $vs \in VS$ has its FoV divided into disjoint sub-regions FoV_{vs}^{H}, FoV_{vs}^{M} and FoV_{vs}^{L}, which determine the visual levels with *high*, *medium* and *low* quality, respectively. The first level is defined by an isosceles triangle $\triangle AFG$ with its high dov_H defined as the distance from vertex A to the end of FoV_{vs}^{H}. The second and third levels are defined by isosceles trapezoids $\pentagon DEGF$ and $\pentagon BCED$ with highs equal to $(dov_M - dov_H)$ and $(dov_L - dov_M)$, respectively. It is important to notice that the sensor FoV is not modified. It is only re-interpreted, being $FoV_{vs} = FoV_{vs}^{H} \cup FoV_{vs}^{M} \cup FoV_{vs}^{L}$ and $FoV_{vs}^{H} \cap FoV_{vs}^{M} \cap FoV_{vs}^{L} = \emptyset$.

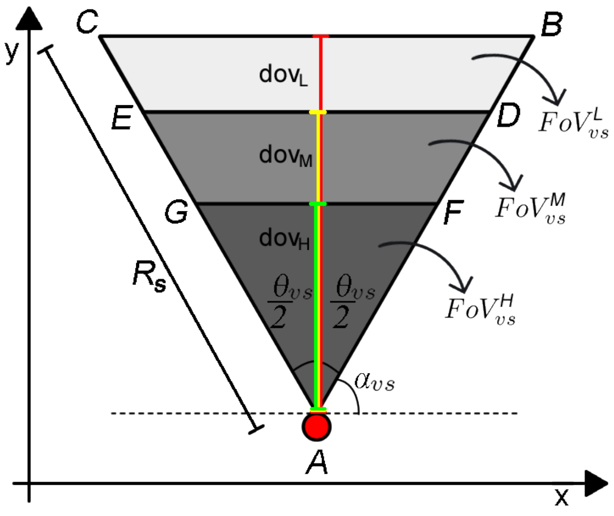


Figure 6.5: Quality perspective for a visual sensor's Field of View.

The distances dov_H, dov_M and dov_L can be computed according to Equation 6.4 and the proportion of dov_H and dov_M with respect to dov_L can be defined freely, considering the application requirements and camera's constraints. The coordinates of vertices D, E, F and G can be computed according to Equation 6.5. In this thesis, we consider only three quality levels, but they could be extended to incorporate additional levels, without loss of generality. In fact, the quality variation will be more realistic if the same FoV could be divided in more quality levels.

$$\begin{aligned} dov_L &= R_s . \cos\left(\theta_{vs}/2\right) \\ dov_M &= \left(3/4\right) dov_L \\ dov_H &= \left(2/4\right) dov_L . \end{aligned} \tag{6.4}$$

$$
\begin{aligned}
&\overline{AD} = \overline{AE} = \tfrac{dov_M}{\cos(\theta_{vs}/2)} \\
&\overline{AF} = \overline{AG} = \tfrac{dov_H}{\cos(\theta_{vs}/2)} \\
&D_x = A_x + \overline{AD}.\cos(\alpha_{vs}) \\
&D_y = A_y + \overline{AD}.\sin(\alpha_{vs}) \\
&E_x = A_x + \overline{AE}.\cos((\alpha_{vs} + \theta_{vs}) \bmod 2\pi) \\
&E_y = A_y + \overline{AE}.\sin((\alpha_{vs} + \theta_{vs}) \bmod 2\pi) \\
&F_x = A_x + \overline{AF}.\cos(\alpha_{vs}) \\
&F_y = A_y + \overline{AF}.\sin(\alpha_{vs}) \\
&G_x = A_x + \overline{AG}.\cos((\alpha_{vs} + \theta_{vs}) \bmod 2\pi) \\
&G_y = A_y + \overline{AG}.\sin((\alpha_{vs} + \theta_{vs}) \bmod 2\pi)
\end{aligned}
\tag{6.5}
$$

In order to provide quantitative assessment, we assign a weight value for each visual level, which is $w_H = 1$ for FoV_{vs}^H, $w_M = 0.5$ for FoV_{vs}^M and $w_L = 0.25$ for FoV_{vs}^L. Obviously, different values could be assigned, according to the application requirements. For example, following the definitions in (Tao et al., 2017), the assigned values would be $w_H = 4$ for FoV_{vs}^H, $w_M = 2$ for FoV_{vs}^M and $w_L = 1$ for FoV_{vs}^L. Actually, we use a percentage approach because an entire region "poorly" viewed would be equivalent to (a percentage) part of an "adequately" viewed region. Nevertheless, this does not mean that it is indifferent for the application to monitor a small area with good quality or a large area with low quality.

In fact, an application is probably not able to extract the same visual information from 100 *MB* poorly monitored ($w_L = 0.25$) and from 25 *MB* well monitored ($w_H = 1$), and vice-versa. However, we believe these quality weights are defined in a way that they can provide relevance equivalence between levels. For example, the information extracted from 25 well monitored *MB* can be as relevant as the information extracted from 100 poorly monitored *MB*, depending on the application. Actually, with less covered area, but with high coverage quality, it may be possible to make facial recognition. On the other hand, with a larger covered area, but with an associated lower coverage quality, it could be possible to detect intrusion or to perform pattern identification. Therefore, it is not necessarily about the importance of the task, but the possibility of adding value to visual information. This view-notion simplifies the understanding of the proposed metrics and indicates how practical they can be when performing quality assessment.

6.2.1.1 Proposed Quality Metrics

One of the challenges to assess the monitoring quality for area coverage is the need to deal with continuous variations of quality in function of the distance of view of a visual sensor. But this may be a prohibitive task if it is desired to compute the QoM of the entire network instead of a single visual sensor. For that, we treat this potential complex scenario as a discrete problem considering that the area to be monitored will be divided into monitoring blocks, thus approximating "area coverage" to the "coverage of several targets". In this case, the smaller the *MB*, the more realistic the QoM assessment will be.

In this context, we propose three new QoM metrics: AQM, AQM_{abs} and AQM_{rel}. These Area Quality Metrics consider that, similarly to the visual levels, each monitoring block $mb \in FoV_{vs}$ receives a weight w_l which is the weight of the FoV sub-region of vs where mb can be viewed, as expressed in Equation 6.6.

$$w_l\left(mb, vs\right) = \begin{cases} w_H, & \text{if } mb \in FoV_{vs}^{H} \\ w_M, & \text{if } mb \in FoV_{vs}^{M} \\ w_L, & \text{if } mb \in FoV_{vs}^{L} \end{cases} \tag{6.6}$$

If a monitoring block mb is redundantly covered by a set of visual sensor VS, then the weight of mb is the maximum weight among the associated sensor, as expressed in Equation 6.7. It is worthy to remark that in this thesis it is not considered the impact of perspective of coverage. This explains why it is taken the maximum weight instead other compositions: the visual information extracted from a MB by different sensors will be as good as the best quality of monitoring available among the associated sensor. In a different scenario, where the coverage direction is considered, sum or average should provide a better quality representation.

$$w_{\max}\left(mb, VS\right) = \max\left(wl\left(mb, vs\right)|_{\forall vs \in VS}\right) \tag{6.7}$$

Figure 6.6 illustrates the mapping of monitoring quality of the MB covered by two visual sensors, including the overlapping considerations. The MB marked with a green circle are in *level H* (highest quality), while the ones marked with a yellow star are in *level M* (medium quality) and the MB marked with a red square are in *level L* (lowest quality). Notice that there are some MB marked with more than one symbol: those MB are redundantly monitored by more than one visual sensor and its assigned weight is that one related to the highest quality.

We define the proposed metrics as presented in Equations 6.8, 6.9 and 6.10.

$$AQM_{abs}(VS) = hs \cdot ws \cdot \sum_{k=1}^{M} \sum_{l=1}^{N} w_{\max}\left(mb_{j,l}, VS\right) \tag{6.8}$$

$$AQM_{rel}(VS) = \frac{AQM_{abs}(VS)}{CA_{mb}\left(VS\right)} \tag{6.9}$$

$$AQM(VS) = \frac{AQM_{abs}(VS)}{h \cdot w} \tag{6.10}$$

The AQM_{abs} is an intermediate metric that provides an absolute perspective of the quality of monitoring, indicating the equivalent quantity of monitoring blocks. In a different way, the AQM_{rel} provides a relative perspective of the quality of monitoring, presenting the percentage of the equivalent monitoring blocks related to the covered area. Finally, AQM provides a global perspective of the quality of monitoring, indicating the percentage of the equivalent monitoring blocks related to the entire monitoring field. On the other hand, AQM_{rel} reveals an innermost panorama of the coverage, according to a threshold value. A low value of AQM_{rel} ($< 62.5\%$) implies that a larger area is covered with the majority of MB being "low quality" monitored, while

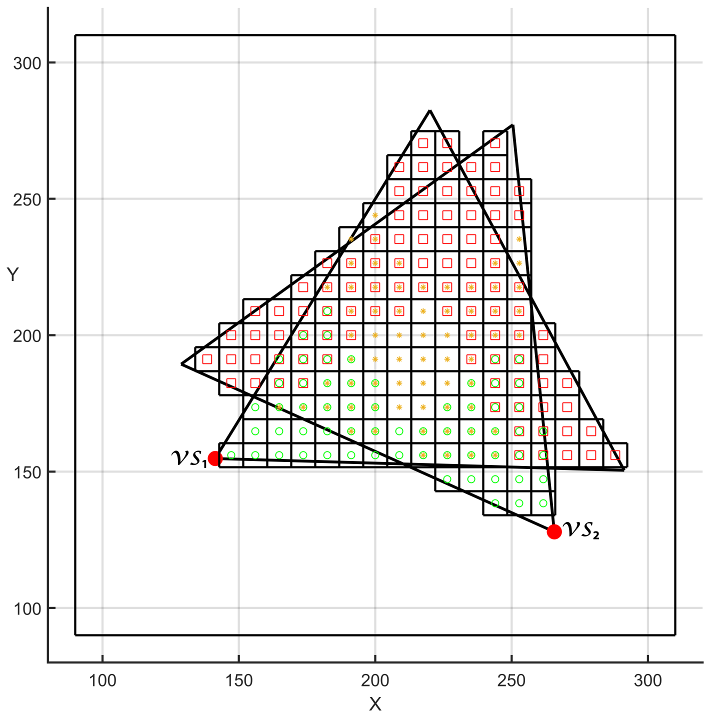


Figure 6.6: Quality of the performed area coverage when viewing monitoring blocks.

a high value of AQM_{rel} ($> 62.5\%$) implies that a smaller area is covered with the majority of MB being "high quality" monitored.

We can compute the threshold value of AQM_{rel} considering the worst and best case scenario for this metric, which varies from 25% to 100% as it can be easily verified. In the worst case scenario, all covered MBs would be in the lowest quality level (w_L), which generates $AQM_{rel} = 25\%$. In the best case scenario, all covered MBs would be in the highest quality level (w_H), which generates $AQM_{rel} = 100\%$. That way, we can state that threshold value of AQM_{rel}, th_{rel}, being the mean value between the worst and the best case scenario, starting from the minimum value, according Equation 6.11.

$$th_{rel} = 25\% + \frac{100\% - 25\%}{2} = 62.5\% \tag{6.11}$$

Table 6.2 shows different scenarios to better understand the meaning of the proposed metrics. The AQM is the fundamental metric, associating area coverage with the quality of monitoring. However, such monitoring can be performed in different ways. For example, a large area may be monitored with low quality, while a small area can be monitored with high quality. These two scenarios would probably present a similar AQM value, as presented in lines 1, 2 and 3 of Table 6.2. In those cases, it is difficult to perform worth assessment considering only the AQM metric and thus the other proposed metrics can be used for a better perception of the considered visual sensor network.

Therefore, the AQM_{rel} appears as an auxiliary metric to help to "untie" such comparisons.

Table 6.2: QoM metrics analysis for $ws = hs = 1$.

Number of covered *mb*			CA_{mb}	AQM_{abs}	AQM_{rel}	AQM
Level H	Level M	Level L				
10	20	60	90	35	39%	23%
10	30	40	80	35	44%	23%
10	40	20	70	35	50%	23%
10	20	40	70	30	43%	20%
4	30	36	70	28	40%	19%

Thus, it is possible to distinguish the monitoring quality between coverage schemes prioritizing area coverage (lower AQM_{rel}) or quality of monitoring (higher AQM_{rel}). The relation between these metrics can be used to improve objective functions in optimization processes used to increase the system dependability. Some authors use redundancy as dependability or, specifically, as availability metrics (Istin et al., 2010).

Aiming to provide some numerical results, it is analyzed the impact of the viewing angle on each *FoV* level, considering constant values of dov_H, dov_M and dov_L. All visual sensors are set having the same sensing radius $R_s = 150$ *u.d.*.

As a reference when setting the viewing angle, Figure 6.7 shows the total covered area by a visual node, as well as the covered area by each *FoV* level. It can be seen that for smaller angles (below 45^o) we have lower covered area. For 75^o and higher we have a greater covered area, specially for FoV_{vs}^H and FoV_{vs}^M. However, very wide angles bring the risk of loss of quality on peripheral areas. This could be solved increasing the image definition or setting an anisotropic QoM with respect to viewing angle (Wang and Wang, 2019). That being said, we set $\theta_{vs} = 60°$ which is an intermediate value and also it is the average viewing angle of several commercial cameras widely used on academy and industry, such as *RaspiCam* and *Cisco IP cameras* (Costa, 2020).

An example of distribution of visual sensors on the WVSN deployment considering QoM is shown in Figure 6.8. These visual nodes must cover a monitoring area with $w = 500$ *u.d.*, $h = 500$ *u.d.*, $ws = hs = 8.5$ *u.d.* and $\beta = 0°$. The position and orientation of each sensor node were randomly generated. For this example, the coverage area and quality metrics assessed are $CA_{mb}(VS) = 127972$ *u.a.*, $AQM = 29.8\%$ and $AQM_{rel} = 58.2\%$.

Another interesting result to notice is that the growth of coverage area is not directly related to the growth of the AQM, which means that a dependability optimization considering QoM cannot be dependent only of coverage area. Actually, it is natural that, increasing the covered area after an optimization process, more non-monitored regions will be encompassed, which tends to contribute to the gathering of more visual data and the improvement of the QoM perception. However, a non-optimal area coverage could provide less overlapping of regions with the same weight, providing a higher QoM. An example of this phenomenon can be seen in Figure 6.9. In this case, Network

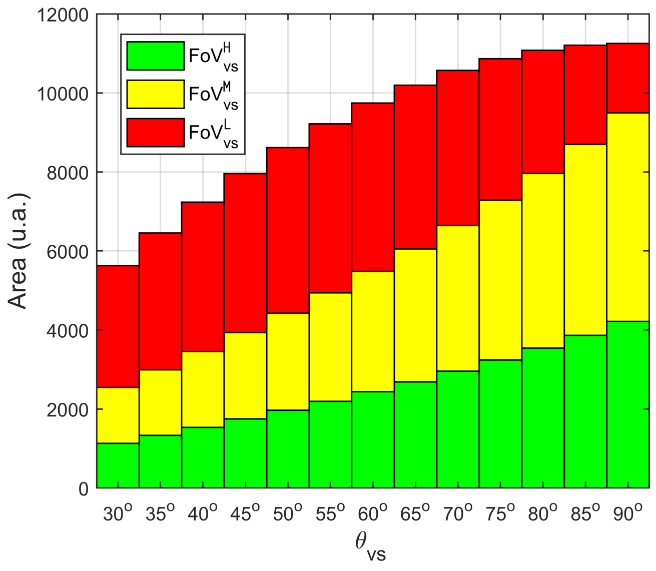


Figure 6.7: Quality variation related to the viewing angle.

1 presents $CA_{mb}(VS) = 28835$ *u.a.* and $AQM_{abs}(VS) = 14529$ *u.a.*, while Network 2 presents $CA_{mb}(VS) = 28166$ *u.a.* and $AQM_{abs}(VS) = 14696$ *u.a.*. Hence, Network 1 has a higher covered area and a lower QoM than Network 2. This means that we can not reduce the problem of QoM optimization to a simple area coverage maximization or redundancy minimization.

6.2.2 Occlusion

Occlusion will be produced by some static or mobile obstacle that may block the sight of visual nodes, potentially reducing a cameras' FoV and indirectly impacting the overall monitoring quality and system dependability. The computation of those Occluded FoV (OFoV) starts with the proper modelling of the obstacles, which must be mapped into the monitoring field and merged with the visual sensors model, in order to identify the effective coverage area.

Actually, there are different ways to geometrically model obstacles, with different levels of complexity. However, when modelling large WVSN and multiple objects, the perception of the obstacles may be simplified to a line, a square or a circle, using the original dimensions of the obstacles as reference. Such simplifications may allow the processing of visual occlusion as a simpler problem of Geometry, with lower computational complexity and approximated results.

In order to achieve a practical and yet effective mathematical model for the defined problem scope, obstacles will be modelled as rectangles, assuming that a WVSN may view a number O of obstacles with different dimensions. The use of rectangles allows a reasonable simplification of objects, while keeping the intersection of obstacles and FoV's triangles more tractable.

For any considered obstacle, which may be any moving or static object (*e.g.* a car, a forklift, a tractor, a wall, a building, a tree), an imaginary rectangle o, for $0 < o \leq O$, will be considered "circumscribing" the obstacle. This modelled rectangle, with width w_b and height h_b, will represent a virtual and mathematically defined instance of the real obstacle. Figure 6.10 presents the

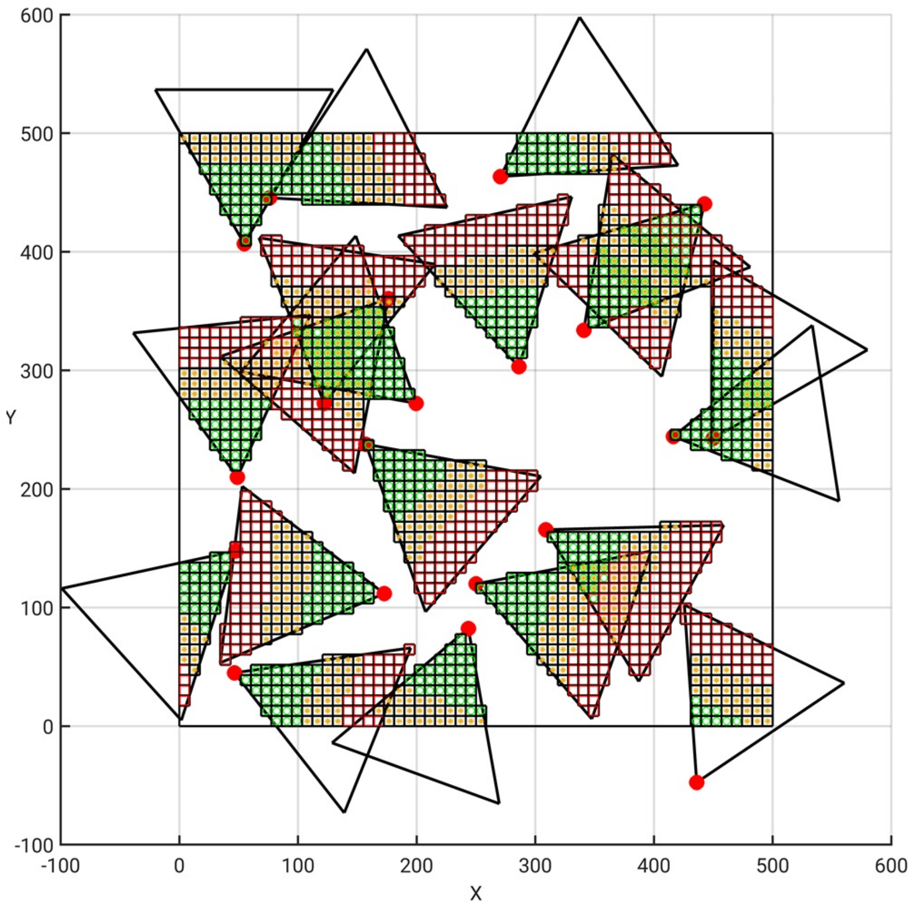


Figure 6.8: Example of a WVSN deployment under quality assessment.

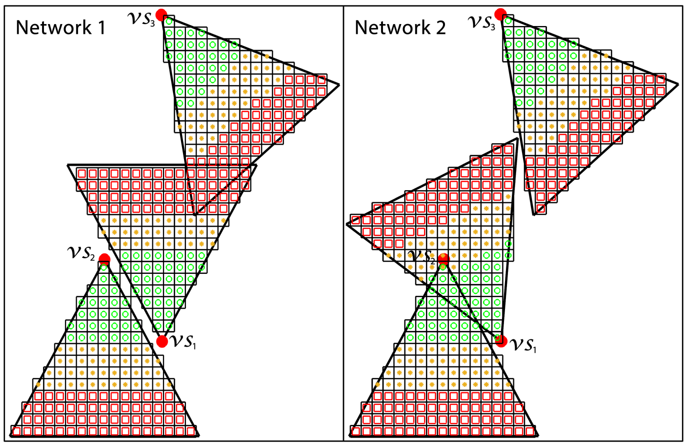


Figure 6.9: Association between covered area and QoM.

generic idea of modelling obstacles as rectangles, which are considered as being perceived from a top-view perspective.

An obstacle is defined by its position (H_x, H_y), which is mapped to the centroid of the rectangle, as well as its orientation, referred as β_b. Figure 6.11 depicts the proposed modelling of an obstacle, presenting its centroid, its orientation and the four vertices of the defined rectangle.

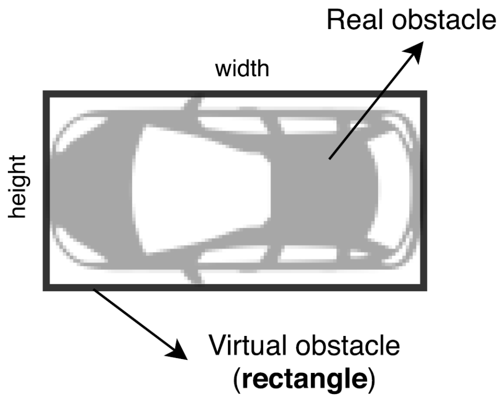


Figure 6.10: Abstraction of obstacles as rectangles.

With this information, the vertices' coordinates of an obstacle, (xi_b, yi_b), $i = 1, \ldots, 4$, can be computed. For this purpose, an auxiliary coordinate system (x', y') is defined with its origin at point H. In (x', y') it is easy to determine the coordinates (xi'_b, yi'_b) according to Equation 6.12. The next step is to rotate the obstacle in β_b degrees to establish its correct orientation. Finally, the auxiliary coordinate system is translated to the original system, obtaining the real (xi_b, yi_b) coordinates. Equation 6.13 and Equation 6.14 present the translation and rotation matrices, respectively. The coordinates (xi_b, yi_b) are defined by the multiplication of the matrices in the specified order, according to Equation 6.15.

$$\begin{aligned} x1'_b &= -{}^{w}/_{2},\ y1'_b = -{}^{h}/_{2} \\ x2'_b &= {}^{w}/_{2},\ \ y2'_b = -{}^{h}/_{2} \\ x3'_b &= {}^{w}/_{2},\ \ y3'_b = {}^{h}/_{2} \\ x4'_b &= -{}^{w}/_{2},\ y4'_b = {}^{h}/_{2} \end{aligned} \tag{6.12}$$

$$T(H_x, H_y) = \begin{bmatrix} 1 & 0 & H_x \\ 0 & 1 & H_y \\ 0 & 0 & 1 \end{bmatrix} \tag{6.13}$$

$$R(\beta_b) = \begin{bmatrix} \cos(\beta_b) & -\sin(\beta_b) & 0 \\ \sin(\beta_b) & \cos(\beta_b) & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{6.14}$$

$$\begin{bmatrix} xi_b \\ yi_b \\ 1 \end{bmatrix} = T(H_x, H_y) \cdot R(\beta_b) \cdot \begin{bmatrix} xi'_b \\ yi'_b \\ 1 \end{bmatrix} \tag{6.15}$$

The computed vertices of the rectangles are necessary to define the line equations of the obstacles. Actually, an rectangle is composed of four sides and each of those sides can be modelled by a line equation, easing the computation of occlusions exploiting Geometry rules. For any consecutive points P and Q taken from the list of vertices of a obstacle, *i.e.*, for vertices belonging to

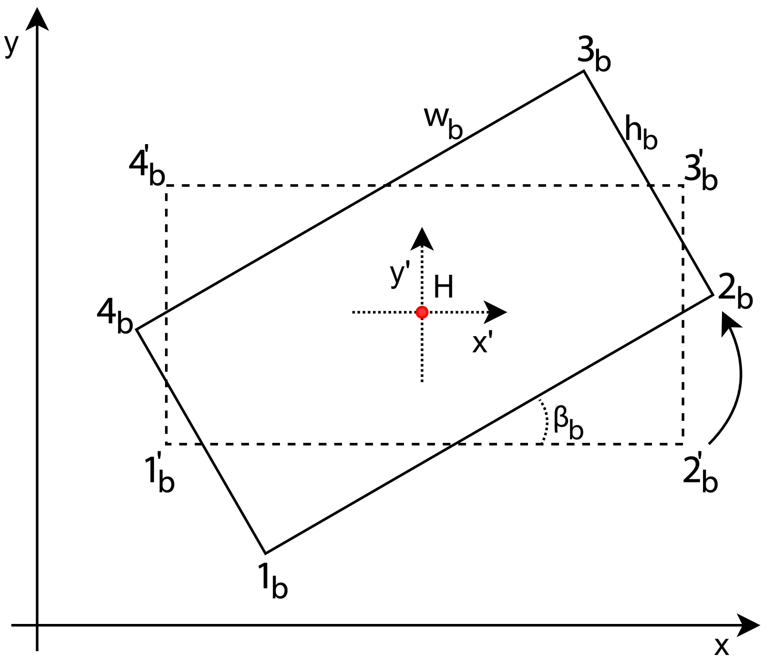


Figure 6.11: The defined mathematical model for the obstacles.

the same side of the rectangle ($\overline{x1_bx2_b}$, $\overline{x2_bx3_b}$, $\overline{x3_bx4_b}$ and $\overline{x4_bx1_b}$), Equation 6.16 will define each line equation of the sides of the obstacle o.

$$\overline{PQ} : y - P_y = \left(\frac{Q_y - P_y}{Q_x - P_x}\right).(x - P_x) \tag{6.16}$$

In the same way of the obstacles, the FoV of the visual sensors can be processed as a group of line equations. For simplicity, each of the three lines that define a FoV's triangle can also be modelled following a specific formulation, as presented in Equation 6.17.

$$\begin{aligned}
\overline{AB} &: y - A_y = (x - A_x).tan(\alpha_{vs}) \\
\overline{AC} &: y - A_y = (x - A_x).tan\left((\alpha_{vs} + \theta_{vs})mod 2\pi\right) \\
\overline{BC} &: y - B_y = \left(\frac{C_y - B_y}{C_x - B_x}\right)(x - B_x)
\end{aligned} \tag{6.17}$$

Having the line equations of both obstacles and visual sensors, the "interactions" between those elements can be computed and estimated, for any configuration of WVSN. Figure 6.12 depicts some of the situations that may happen when visual sensors are occluded by obstacles, presenting all related parameters.

6.2.2.1 Computing Visual Occlusion

The modelling of the visual sensors' FoV and the obstacles' rectangles are required to compute the estimated visual occlusion in each of the considered sensors. In fact, any obstacle may interfere in

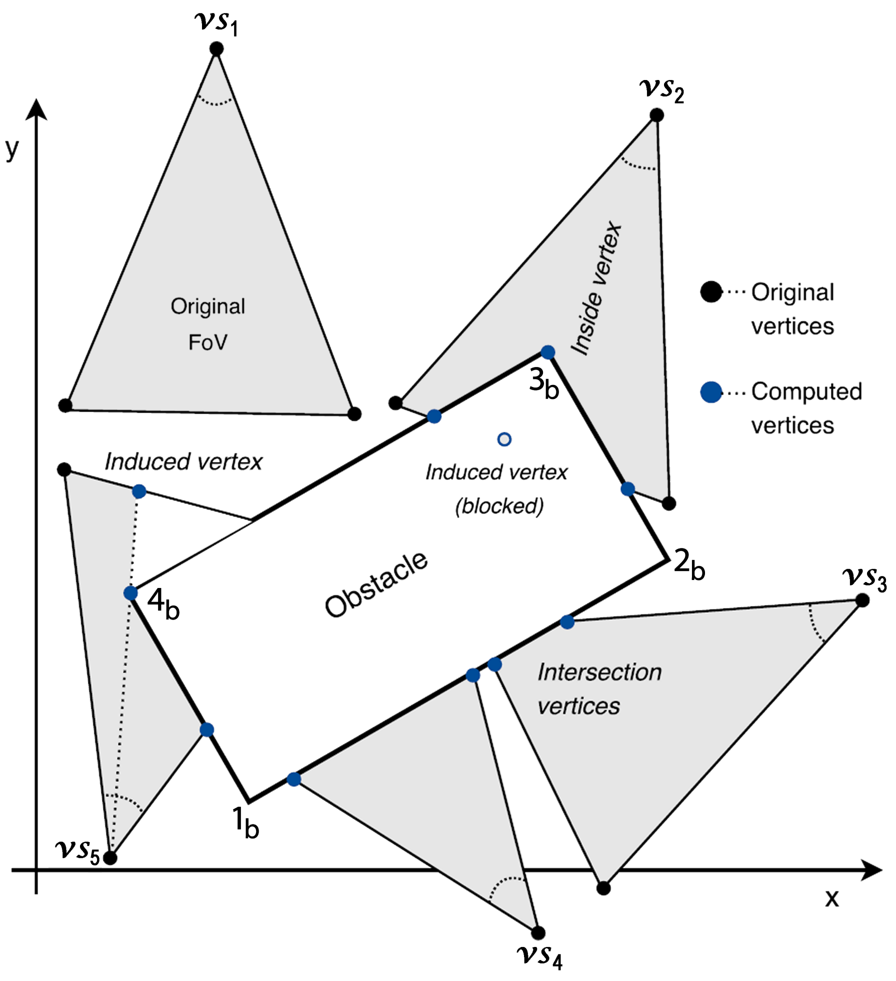


Figure 6.12: Different possibilities when visual sensors are being occluded. The shaded areas are the resulted FoV after occlusion (OFoV).

the expected viewed area, reducing the field of view of the visual sensors. But the actual impact of the obstacles will vary according to the configurations of the visual sensors and the positions and dimensions of the obstacles. In this sense, it is defined the concept of *Occluded FoV* (OFoV), which is a visual coverage area derived from an original FoV. The OFoV will be modelled as a concave or a convex polygon with an area smaller than the area of the corresponding original FoV triangle. Hence, the computed area of each OFoV will be exploited to identify if a certain visual sensor is under a coverage failure.

Visual occlusion will be computed through the definition of a set of *occlusion vertices*, which are mostly resulted from the intersection of the obstacles' lines with the visual sensors' lines. Actually, two lines may not intersect (parallel lines), may intersect in one point (which may be over a FoV's line or not, in the case the lines extensions meet) or in infinite points (when the lines are coincident). Whatever the case, we can expect that a FoV will be occluded for different configurations of the obstacles, which may intersect zero or more lines of any FoV, as depicted in Figure 6.12.

The OFoV of the visual sensors will be created by the influence of one or more obstacles and thus they may be concave or convex polygons composed of three or more vertices, resulting in $OFoV_{vs} < FoV_{vs}$. Although an OFoV may be affected by any number of obstacles, they are expected to be processed individually, line by line: the final OFoV is valid when all obstacles that may affect a sensor are considered.

Then, the computation of a set L of occlusion vertices will involve an original FoV or an OFoV that is not fully processed yet, which will be considered sequentially against unprocessed lines of obstacles. This process is repeated iteratively, until all obstacles' lines are considered. For that, 6 processing steps were defined, which may or may not achieve a new set of occlusion vertices at the end of each iteration. Therefore, for iteration z, the set of vertices $L(z-1)$ will be considered through 6 processing steps until a new set $L(z)$ is computed, which is the final OFoV of sensor vs if z is the last processed iteration. Moreover, for iteration $z = 1$ (the first iteration), the vertices in the set $L(0)$ will be considered as input, which will only contain the three original vertices of the corresponding sensor's FoV.

The defined steps are derived and extended from the work in (Costa et al., 2017b), resulting in more comprehensive and robust processing steps, described as follows:

- **Step 1**: The *intersection* vertices. These vertices are resulted from the intersection of an obstacle's line and one line of the considered FoV or OFoV. For any two different points $(x1,y1)$ and $(x2,y2)$, the linear equation to be considered will take the form $(x2-x1)(y-y1) = (y2-y1)(x-x1)$;

- **Step 2**: The *inside* vertices. When one or two vertices of the considered obstacle are inside the FoV or OFoV, they will be included as a vertex of the computed OFoV. The verification if a point is inside the original FoV (triangle) is easy, but it gets tricky when a point is checked against an OFoV defined as a concave polygon, as discussed in next subsection;

- **Step 3**: Inclusion of vertex A of the original FoV. The vertex A is the sensor's position and thus it must be in any computed OFoV. Because of this characteristic, the vertex A will guide the ordering of the vertices in any computed set L;

- **Step 4**: Selection of some vertices of the original FoV or OFoV. There are some vertices in $L(z-1)$ that will be replicated into $L(z)$. In this processing step, the line equation defined when taking vertex A and any "adjacent" vertex of the original FoV (B or C vertices) or of the considered OFoV must not intersect the processed line of an obstacle. If there is no intersection, the considered adjacent vertex is included into the set of occlusion vertices;

- **Step 5**: *Induced* vertices. They result from an intersection of the linear equation created by vertex A and by an *inside* vertex (from Step 2), with the line of the side of an original FoV or OFoV. In a simple analysis, the *induced* vertices are resulted from the "shadow" created by an obstacle, since part of the original FoV will not be viewed anymore;

- **Step 6**: Removal of internal *induced* vertices. The *induced* vertices are fundamental when computing the OFoV, but for z as the last iteration for a particular obstacle o, the set $L(z)$ must not have an *induced* vertex if it is "blocked" by two lines. Actually, for any *inside* vertex, the same *induced* vertex will always be computed twice since that vertex will be part of two concurrent lines. However, in the second computation, internal (blocked) *inducted* vertices must to be removed, as depicted in Figure 6.12 (in vs_2 there is no *induced* vertex resulted from the intersection of line $\overline{A_2 3_b}$ (being A_2 the vertex A of visual sensor 2) and a line of the original FoV, but an *induced* vertex is in the final set L in vs_5 since it is not "blocked").

Following all defined steps and after the last iteration, the final set L of occlusion vertices may be achieved for the visual sensor vs, resulting in $L_{vs} = \{(Vx_1, Vy_1), (Vx_2, Vy_2), \ldots, (Vx_v, Vy_v)\}$, for v vertices in L_{vs} and $(Vx_1, Vy_1) = (Ax_{vs}, Ay_{vs})$. As those vertices create a polygon, the viewed area can be computed using the Shoelace formula (Pure and Durrani, 2015), as showed in Equation 5.3.

Although the defined processing steps can be used to compute the vertices of the final OFoV, there are some important issues that must to be also considered, impacting on the selection of the faultless visual sensor nodes. Such issues are summarized as follows:

- If there is no computed *intersection* vertex (Step 1) or *inside* vertex (Step 2) in iteration z, then $L(z) = L(z-1)$. In other words, the first two steps will indicate if the remaining processing steps need to be considered for the current iteration;
- A special condition that needs proper processing to avoid inconsistencies is when the vertices of the FoV or OFoV are coinciding with vertices of the obstacles. In such case, if the coinciding points are the only computed *intersection* vertices, the current FoV/OFoV must be assumed as the computed OFoV in the considered iteration. Otherwise, the OFoV computation must to be performed as already defined;

Actually, all defined steps and processing remarks are sufficient to compute the occlusion vertices for each visual sensor. However, such computations require the correct ordering of the vertices in each L_{vs} set, as discussed in next section.

6.2.2.2 Ordering the Vertices in each OFoV

After computing all the occlusion vertices in each (intermediate) L set, the vertices must to be ordered to correctly represent the defined OFoV in each iteration. This is required since the area of the OFoV and the computing of new occlusion vertices in further iterations can only be performed if the OFoV is correctly defined. However, there are different possible ways to order the occlusion vertices, resulting in different polygons. Figure 6.13 presents examples of different polygons resulted from different orders for the vertices.

The computation of the OFoV's area through the Shoelace formula can only be performed if the vertices in the set L are in a clockwise or anti-clockwise order, which lead us to define the

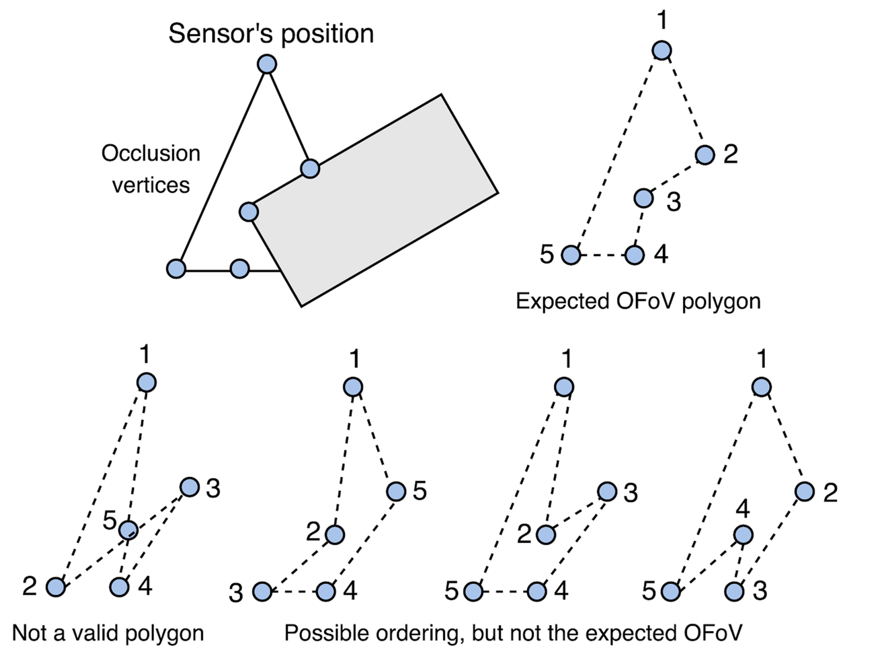


Figure 6.13: Ordering the list of occlusion vertices to create an OFoV.

ordered list L' as the result of this ordering process. But since there are different possible polygons depending on the number of occlusion vertices, a new method had to be developed to compute the correct (clockwise or anti-clockwise) order for the vertices. The defined method takes the vertex A as the reference for ordering, since it is the sensor's position, computing the order of the remaining vertices based on their angular distance to vertex A and a group of heuristics.

The angular distances are computed through the *arc tangent* function taking vertex A and each of the vertices in the original set L. This function is defined in a numeric value ranging from -90^o to 90^o (4th and 1st quadrant, respectively). Each of the computed angles are assigned to the considered vertices, allowing the sorting of the vertices according to their angular distances to vertex A. However, in order to consider only positive angles, we perform a quadrant shift (4th to 2nd, where the tangent values remain the same, inclusive keeping its negative sign) by summing 180^o to all negatives vertices, resulting in a sorting scope from 0^o to 180^o. After that, any sorting algorithm can be employed (*e.g.* bubblesort), producing a list L' with the vertices in a clockwise or anti-clockwise order (depending on the employed sorting algorithm).

Although this process is expected to efficiently organize the vertices, a recurrent problem will be resulted from vertices with the same angular distance to vertex A. Actually, this may be particularly common, as depicted in Figure 6.13: considering the correct ordering for the vertices, vertex 3 and vertex 4 have the same angular distance. In this case, any ordering algorithm will not differentiate those vertices, but there will be only one correct order when defining the OFoV. Therefore, we had to develop some heuristics to guide the correct ordering for the vertices.

Considering the expected formats for the OFoV, which is guided by the vertex A and the expected configurations of possible occlusions caused by rectangle-shaped obstacles, as expressed in Figure 6.12, there will be some well-defined viewing limits for the visual sensors when they are occluded. Therefore, as an effective solution for the problem of vertices with the same angular distance, the proposed ordering algorithm will differentiate vertices from the original FoV triangle and vertices created by occlusion. Then, the "type" of the vertices will be considered along with a second decision parameter: the Euclidean distance of the considered vertex to vertex A. Doing so, after the initial ordering of the vertices assuming only the angular distances as the single sorting parameter, a second round will process only vertices with the same angular distance, taking as reference the previous vertex in the computed order. For them, the Euclidean distance will be used as a second comparison parameter according to the following heuristic: the vertex with the highest distance must be "closer" to an original vertex of the FoV triangle, while the opposite applies for a vertex created by occlusion. Figure 6.14 presents an example of this solution for both clockwise and anti-clockwise orders, taking as reference the occlusion configuration previously depicted in Figure 6.13.

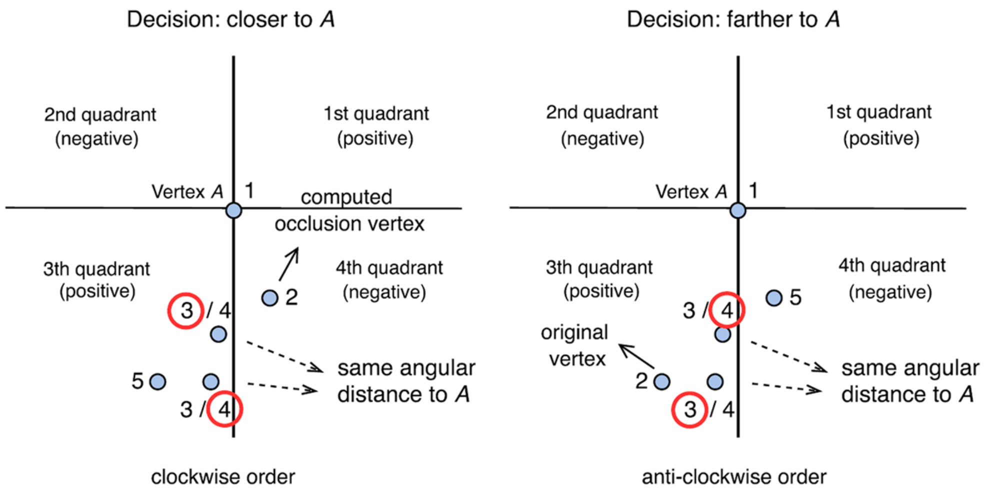


Figure 6.14: Applying the proposed heuristics for vertices with the same angular distance.

The ordering of the vertices must be performed for every iteration in a particular visual sensor, allowing the correct definition of intermediate OFoVs. Doing so, any number of non-coincident obstacles can be processed for a single visual sensor, resulting in the correct representation of the ultimately computed OFoV.

6.3 Integration of Models

The methodology proposed in Chapter 5 entirely follows the NFC generated based on the coverage area (Algorithm 4). However, as important as the amount of visual information, the quality of the gathered information is crucial for applications. Actually, both the amount and quality of this information can be jeopardized by occlusion, leading to a dependability decrease.

That way, in this section the methodology is extended to also incorporate the developed QoM and occlusion models. These coverage failures in the considered network have to be correctly processed. In fact, coverage failures may happen in different ways and in different moments of the network lifetime. Nevertheless, in order to properly evaluate the dependability of the applications, the proposed methodology processes all coverage failures at once, achieving a final dependability perception. Therefore, the standard procedure is to sequentially compute all modelled coverage failures, first computing occlusions on all visual sensors and then processing the QoM associated to the depth of view based on occluded FoV.

The computation of the OFoV will consider all obstacles that can produce some level of occlusion on the deployed visual sensors, achieving a set of occluded FoV for the considered network that reflects the impacts of the obstacles along time. Then, the QoM of that set of active visual sensors can be also processed, assuming that the FoV of any visual sensors may be not an isosceles triangle. In other words, two sequential layers of coverage failures will be computed for the visual sensors, and their resulted coverage on the monitored field will already reflect the impacts of both types of coverage failures. This general processing principle is depicted in Figure 6.15.

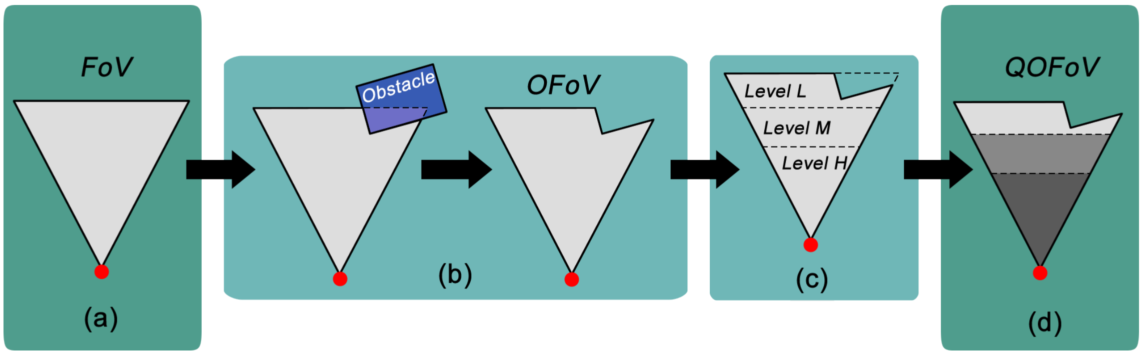


Figure 6.15: A sensor's field of view in different processing steps: (**a**) Original coverage, (**b**) Field of View (FoV) under occlusion, (**c**) Quality of Monitoring (QoM)-aware FoV and (**d**) Quality-Occluded FoV (QOFoV) for both occlusion and QoM-aware conditions.

As an important remark in Figure 6.15, while a visual sensor may be not occluded by an obstacle, which may result in an OFoV similar to the original FoV, the QoM-aware FoV of that sensor will always be computed.

Based on the processing of these coverage failures, we redefine the NFC considering that the application will fail if a combination of nodes are not able to cover a minimum amount of covered area, a minimum quality of monitoring and also considering the minimum percentage of FoV of each visual sensor. For this last case, it is considered that, if a sensor node cannot cover its minimum FoV, its camera is turned off, and the node operates as a *relay node*, that is, it can only retransmits messages, therefore consuming less power, saving battery, extending its lifetime, and so increasing the system dependability. Thereby, the initial supplementary data that the methodology requires from the user in order to characterize the network and the application requirements, must also encompass now the minimum QoM (AQM_{min}) and the minimum FoV (FoV_{min}).

The Algorithm 11 describes the redesigned NFC generation, when a combination of sensor nodes is included into the NFC expression if it provides the required area coverage and QoM (Line 31). Doing so, the dependability evaluation can be carried out considering the parameters of the scenario.

Algorithm 11: NFC.

```
Data: nfc = NFC($CA_{min}$, $AQM_{min}$, VS, Obs);
Input: Min. Coverage Area, Min. QoM, set of Visual Sensors, set of Obstacles.
Output: Network Failure Condition.
1  foreach $vs \in VS$ do
2      $vs.OFoV\_Vertices$ = getOFoV($vs$, Obs);
3      foreach Monitoring Block $mb$ do
           // Coordinates of the center of $mb$ (See Figure 3.6).
4          $xc = mb.x1s + (mb.x4s - mb.x1s)/2$;
5          $yc = mb.y1s + (mb.y4s - mb.y1s)/2$;
6          $isInsideOFoV$ = inpolygon( $(xc, yc)$, $vs.OFoV\_Vertices$); // Ray casting
               algorithm.
7          if $isInsideOFoV$ then
               // Coordinates of the vertices of each quality level:
               // High, Medium and Low (See Equation (3.1))
8              $vertices\_FoV_{vs}^{H}$ = $[(A_x, A_y), (F_x, F_y), (G_x, G_y)]$;
9              $vertices\_FoV_{vs}^{M}$ = $[(F_x, F_y), (D_x, D_y), (E_x, E_y), (G_x, G_y)]$;
10             $vertices\_FoV_{vs}^{L}$ = $[(D_x, D_y), (B_x, B_y), (C_x, C_y), (E_x, E_y)]$;
11             $isInside\_FoV_{vs}^{H}$ = inpolygon( $(xc, yc), vertices\_FoV_{vs}^{H}$);
12             $isInside\_FoV_{vs}^{M}$ = inpolygon( $(xc, yc), vertices\_FoV_{vs}^{M}$);
13             $isInside\_FoV_{vs}^{L}$ = inpolygon( $(xc, yc), vertices\_FoV_{vs}^{L}$);
14             if $isInside\_FoV_{vs}^{H}$ then
15                 $w_l(mb, vs) = w_H$;
16                 $cover(mb, vs) = 1$;
17             else if $isInside\_FoV_{vs}^{M}$ then
18                 $w_l(mb, vs) = w_M$;
19                 $cover(mb, vs) = 1$;
20             else if $isInside\_FoV_{vs}^{L}$ then
21                 $w_l(mb, vs) = w_L$;
22                 $cover(mb, vs) = 1$;
23             end
24         end
25     end
26 end
27 foreach $VS' \subseteq VS$ do
28     $CA_{mb}(VS') = ws \cdot hs \cdot \sum_{j=1}^{M} \sum_{l=1}^{N} cover\left(mb_{j,l}, VS'\right)$;
29     $AQM_{abs}(VS') = hs \cdot ws \cdot \sum_{j=1}^{M} \sum_{l=1}^{N} w_{\max}\left(mb_{j,l}, VS'\right)$;
30     $AQM(VS') = AQM_{abs}(VS') / (h \cdot w)$;
31     if $CA_{mb} \geq CA_{min}$ && $AQM \geq AQM_{min}$ then
32         nfc.addExpression($VS'$);
33     end
34 end
```

The Algorithm 11 starts computing the OFoV of each visual sensor (Line 2). Then, it is verified which monitoring blocks are covered by each visual sensor and their respective QoM. For this, the procedure $inpolygon()$ in Lines 6, 11–13 checks if the center of a mb is within of the OFoV of a visual sensor vs, and within of one of its quality level sub-regions FoV_{vs}^{H}, FoV_{vs}^{M} or FoV_{vs}^{L}, respectively. This procedure implements a Ray casting algorithm (Ye et al., 2013; Kularathne and Jayarathne, 2018). Notice that we first verify if a mb is within of the OFoV of vs (Line 6). This is important to not consider monitoring blocks in occluded areas. According to how a sensor vs covers a monitoring block mb, the functions $w_l(mb, vs)$ and $cover(mb, vs)$ are updated (Lines 14–23). This information is used to proper compute the overall coverage area $CA_{mb}(VS')$ and QoM $AQM(VS')$ provided by a subset of visual sensor $VS' \subseteq VS$ (Lines 28–30). Finally, the NFC is updated if the sensors in this subset are able to fulfill the application requirements together, that is, if they cover an area greater or equal to CA_{min} with QoM greater or equal to AQM_{min} (Lines 31–32).

6.4 Example of Dependability Assessment

In this section it is presented the dependability assessment of an industrial application in order to show the applicability of the proposed methodology, considering its extended model. Actually, for this task, the methodology parameters could be empirically set. However, in order to facilitate the usage of the methodology in real scenarios, some approaches are presented and employed to determine most of these parameters.

There are many possibilities to determine each parameter. Of course, each approach will be as accurate as the considered detailing level. In fact, we are not focused on strictly determining the parameters, but providing possible solutions when defining them. With a realistic parameters setting, we can more confidently evaluate the impact of visual coverage failures on the system dependability.

The proposed scenario used as case study is a merging of specifications of two real industrial shop floors into a single realistic model. This industrial scenario is similar to a paper mill proposed by Ahlen et al. (2019) (Iggesund Paperboard), which contains machinery such as boilers, mixing and storage tanks, conveyors and storage racks. We assume that the shop floor occupies an area $\mathbb{A}$ of 250 m $\times$200 m, where the monitored area MA is 240 m $\times$180 m. Over this MA are deployed 11 visual sensor nodes and one gateway node, following the same arrangement of nodes of the industrial wireless network presented by Wang and Wang (2010), implementing a network topology according to the WirelessHART standard (one of the most adopted wireless communication protocols in industrial applications (Wang and Jiang, 2016)). The visual nodes are oriented in a way to monitor regions of production more susceptible to failures, accidents and intrusion. Figure 6.16 shows the shop floor layout and Figure 6.17 presents the network topology. The percentage on the edges of the graph are the packet reception rates (PRR), which are link quality metrics that will be used to determine the link failure rates. Next we show how to obtain the dependability methodology parameters from the component specifications.

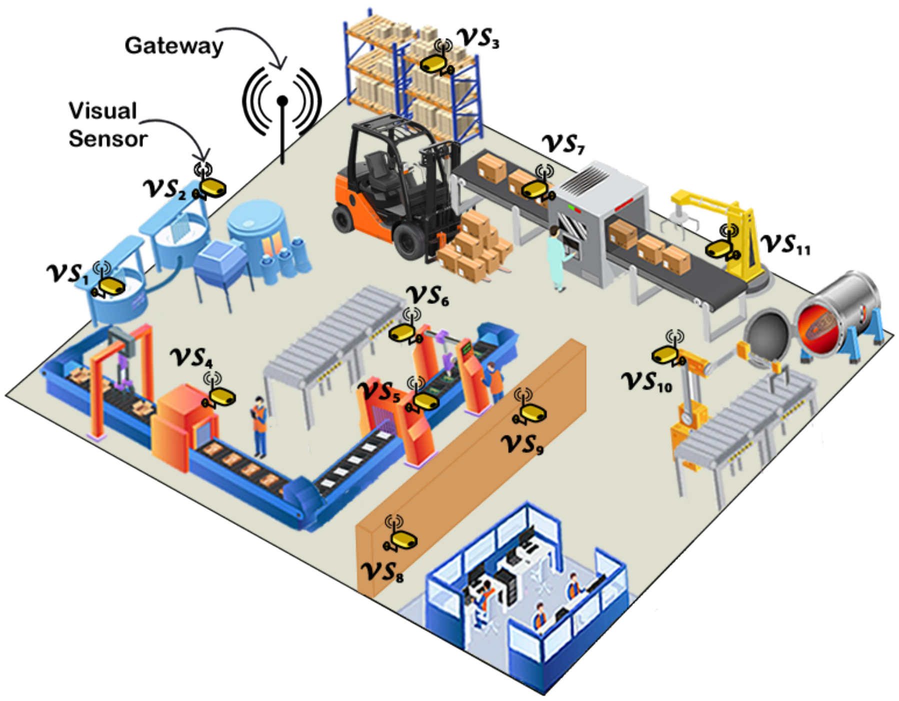


Figure 6.16: Shop floor layout.

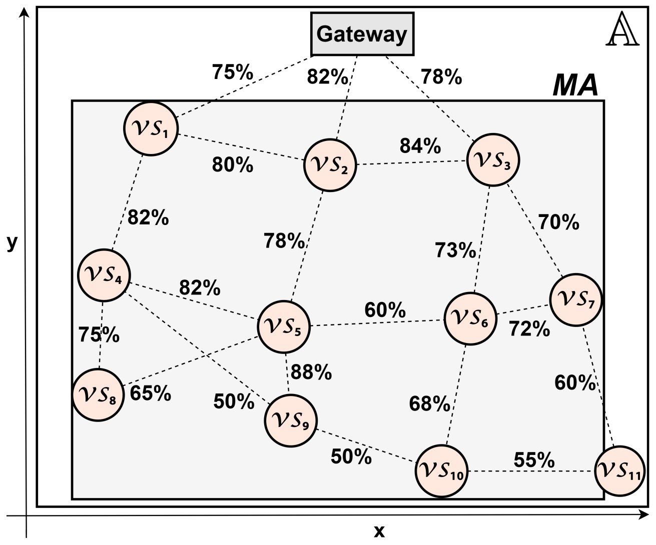


Figure 6.17: Shop floor network.

6.4.1 Experimental Parameter Settings

The methodology requires some data related to visual nodes, components failure and repair rates, network communication (topology, routing, radio range, etc.) and the application requirements. We assume that all the visual sensor nodes are composed of a camera FLIR BLACKFLY®S BFS-U3-13Y3 and that they communicate through an IEEE 802.15.4 radio interface at 2.4 GHz, using a CC2420 antenna which can support WirelessHART setting (Raptis et al., 2018). Each node is powered by a typical battery of WirelessHART field devices, a lithium battery unit BU 191 (Endress+Hauser, 2016) and they communicate based on flooding routing strategy.

We assume that the useful visual information from cameras can be retrieved of a distance not higher than 75 m from the camera, which implies in a sensing radius of $R_s = 75$ m with the viewing angle of $\theta_{vs} = 60°$ (an intermediate value and also the average viewing angle of several commercial cameras). Regarding the monitored area, its dimensions are $w = 240$ m and $h = 180$ m, with $\beta = 0°$. The MA is divided into monitoring blocks with $ws = hs = 2.15$ m, which implies a side size of, approximately, $0,09\%$ of the FoV area. As shown in Figure 5.11, that proportion generates a very low accuracy error, which was, in that case, a error around $0,02\%$ associated to $ws = hs = 0.15$. The position of each sensor node was obtained from the coordinates of Figure 6.17. The dependability parameters were pessimistically set, according to the information available in the datasheets and technical manuals of References (Texas Instruments, 2005; FLIR® Integrated Imaging Solutions Inc, 2017; Endress+Hauser, 2016), with the hardware failure and repair rates $\lambda_{hw} = 2.2831 \times 10^{-5}$ and $\mu_{hw} = 0.0833$, respectively, considering the worst case scenario of the hardware components specifications.

The battery has an initial capacity $c_0 = 19$ Ah rated by $H = 1$ h and a Peukert's constant $\eta = 1.1$. Considering that the average continuous discharge current needed to supply the sensor node (radio, camera, etc.) is about $I = 150$ mA, then the discharge rates of a 4-discharging stages of the battery model are $\lambda_{bt0} = 0.0225$, $\lambda_{bt1} = 0.0197$, $\lambda_{bt2} = 0.0187$ and $\lambda_{bt3} = 0.0181$ h. Once discharged, we consider that it takes 2 hours to repair (replace or recharge) the battery, so $\mu_{bt} = 0.5$.

The estimation of the link rates is more complex since the link has a transient behavior, which means that a failed link will reestablish its connection after a while, without an external intervention. A link failure could be caused by an obstruction produced by an external object, shadowing, signal attenuation, multi-path fading, radio interference, background noise, an occupied channel or a data collision (Zonouz et al., 2014). Due to several uncertainties or difficulties when modelling these aspects, usually resulted from a lack of sufficient data, we consider the fuzzy model approach presented by Purba et al. (2012) to estimate link failure rates based on link quality metrics. These metrics could be, for instance, Packet Reception Rate (PRR), Packet Error Rate (PER), link failure probability or Time to Link Failure (TLF). They can be obtained by simulation (Egeland and Engelstad, 2009), estimation (Gomez et al., 2006) or by radio propagation models (Miyazaki et al., 2012).

The approach proposed by Purba et al. (2012) expresses reliability data (failure probability) in a qualitative natural language and mathematically represented by a membership function of fuzzy numbers. That way, a defuzzification technique is needed to convert the membership function into a reliability score, and later into their corresponding fuzzy failure rates, which is a good approximation for probabilistic failure rates (Purba et al., 2012). We use an area defuzzification technique (ADT), since it presented the best results.

Let $f_f = (a, b, c, d)$ be a fuzzy function, where a, b, c, d are the parameters of a trapezoidal membership function, as shown in Figure 6.18. The parameter f_f can be converted to a crisp score by ADT (Equation (6.18)) and this score can be used to estimate the link failure rate associated to f_f according to Equation (6.19). We use the PRR available in (Wang and Wang, 2010) to define the values of the membership function of each link based on the fact that the higher is the PRR, the lower is the probability of the link to fail.

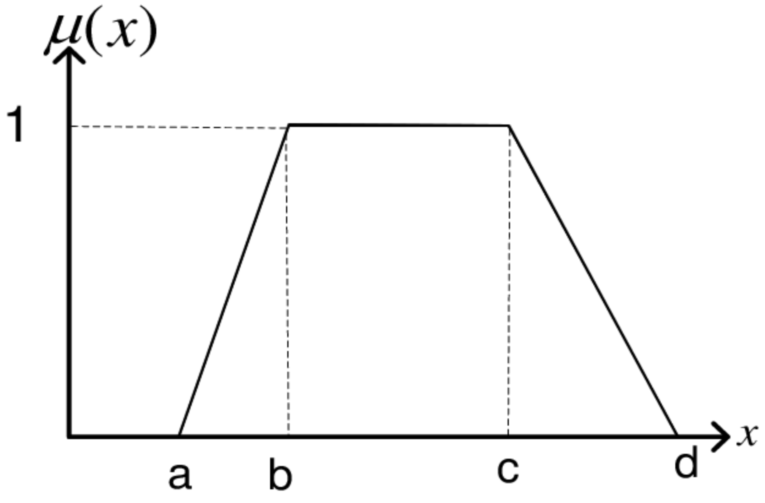


Figure 6.18: Example of a trapezoidal membership function.

It is important to remark that, according to Purba et al. (2012), the relation between the qualitative natural language of the failure probability and its mathematical representation through the membership function (defined by fuzzy function parameters *a, b, c, d*) must be stated by experts' belief of the most expected value that a failure event may occur. That way, the association between these parameters was a relatively subjective choice of us, the research team involved with this thesis, based on our familiarity, considerable training in and knowledge of this application field. The same was applied to the association between these parameters and the PRR.

Thereby, we consider that a PRR under 50% indicates a down link (nonoperational link). The fuzzy function was proportionally generated over the range of 50–100% and the link failure rates (λ_{lk}) were computed through Equations (6.18) and 6.19, according to Purba et al. (2012). These rates, as well as the other parameters of the fuzzy model approach, are shown in Table 6.3 and Figure 6.19 shows the graphic of the membership function defined. Once a communication failure occurs, due to the transient nature of links, the connection from a node to the gateway node could be reestablished naturally, eventually, or by the re-routing or self-healing features of routing protocols, which lead us to consider in this scenario the link repair rate $\mu_{lk} = 4$, which means about 15 min to reestablish.

$$\begin{aligned} ADT = \left[18(a+b-c-d)^2\right]^{-1} \times \Big[(a+2b-2c-d) \times \\ \times \Big((2a+2b)^2 + (c+d)(-3a+2c-d) - 2c(3b+d) - 4ab\Big)\Big] \end{aligned} \tag{6.18}$$

$$\lambda_{lk} = \begin{cases} 10^{-\left[((1-ADT)/ADT)^{1/3} \times 2.301\right]}, & ADT \neq 0 \\ 0, & ADT = 0. \end{cases} \tag{6.19}$$

Table 6.3: Parameters of link reliability and fuzzy model approach.

PRR	Failure Probability	Fuzzy Function Parameters (a,b,c,d)	λ_{lk}
0–50%	Down link (DL)	(1, 1, 1, 1)	∞
50–60%	Very High (VH)	(0.94, 0.95, 1, 1)	3.4820×10^{-3}
60–70%	High (H)	(0.865, 0.875, 0.925, 0.935)	2.3566×10^{-3}
70–80%	Reasonably High (RH)	(0.77, 0.80, 0.85, 0.86)	1.6623×10^{-3}
80–90%	Moderate (M)	(0.715, 0.725, 0.775, 0.785)	1.2916×10^{-3}
90–95%	Reasonably Low (RL)	(0.64, 0.65, 0.70, 0.71)	8.5774×10^{-4}
95–98%	Low (L)	(0.565, 0.575, 0.625, 0.635)	5.7197×10^{-4}
98–100%	Very Low (VL)	(0.5, 0.5, 0.55, 0.56)	4.2291×10^{-4}

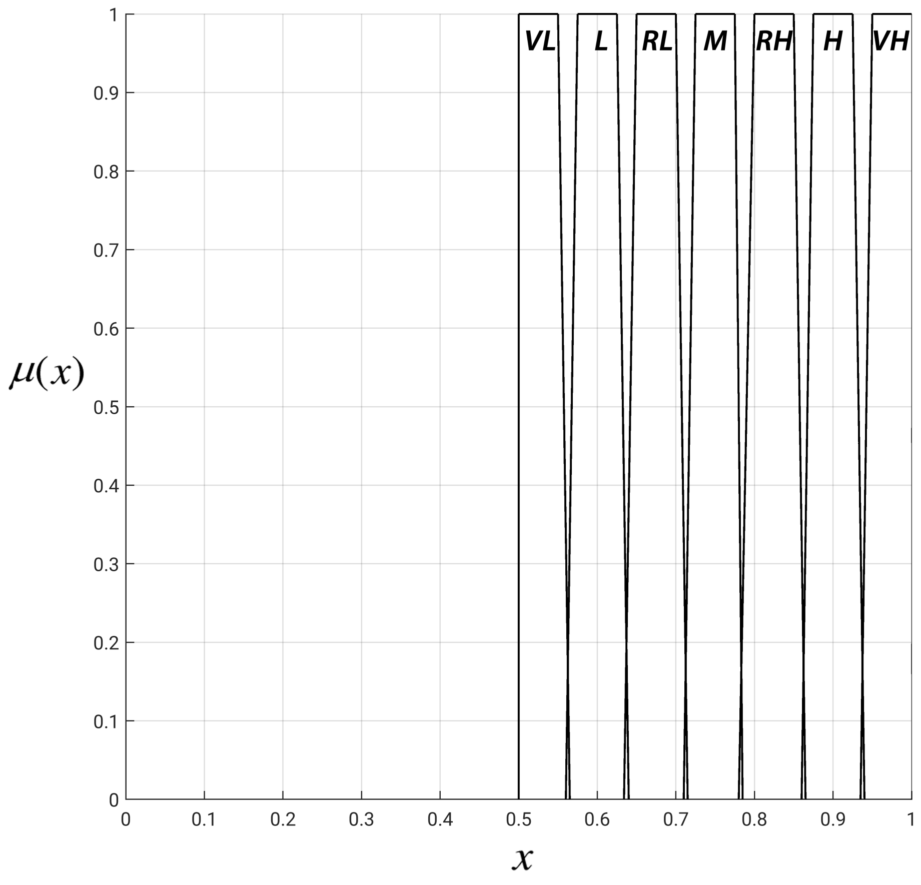


Figure 6.19: Graphic of the membership function defined.

6.4.2 Dependability Evaluation

Once the dependability parameters are set, the application requirements need to be defined to assess the system dependability. In this case, it is necessary to define the minimum acceptable area coverage (CA_{mb}) and quality of monitoring (AQM), CA_{min} and AQM_{min}, respectively. For this, based on the layout shown in Figure 6.16, we consider two obstacles in the monitoring area: a wall separating a management station to the chemical processes and a forklift that spends most of the time near to a conveyor belt to get the paperboard boxes. Figure 6.20 shows the network deployment, including the obstacles (blue rectangles) and the quality perspective of the *MA*. Notice that the FoV of visual sensors vs_3 and vs_7 are partially occluded by the forklift and the FoV of visual sensors vs_4 and vs_5 are partially occluded by the wall. The coverage area is $CA_{mb} = 21,691$ m^2, which is approximately 50% of total *MA*, and the quality of monitoring is approximately $AQM = 28\%$. Since in this example it is not considered network redeployment, the maximum coverage area and the maximum quality of monitoring are those ones presented by the network deployment. This means that $CA_{min} \leq 50\%$ and $AQM_{min} \leq 28\%$, otherwise this network will not fulfill the application requirements.

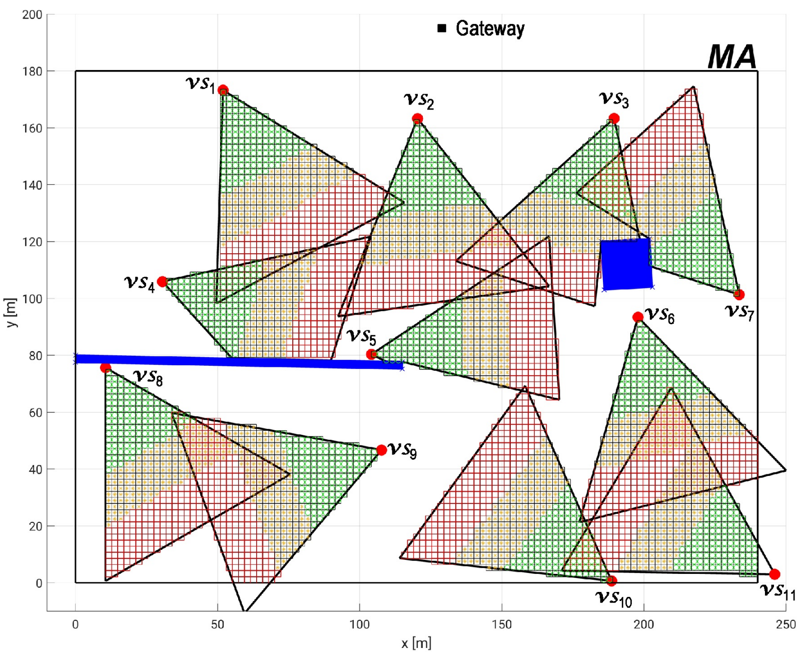


Figure 6.20: Network deployment considering QoM under visual failures.

In order to evaluate the impact of visual failures on dependability, first we perform the system availability assessment considering that $AQM_{min} = 0\%$, varying CA_{min} from 0% to the maximum value of the area able to be covered by the network deployment (50%). In this case, we intend to verify how large is the portion of area of the monitored field that the system can cover considering the proposed deployment. We can observe in Figure 6.21(a) that the system presents a high availability (>99%) for values of CA_{min} from 0% to 33%.

Then, the same analysis is performed considering $CA_{min} = 0\%$ and varying AQM_{min} from 0% to the maximum value of quality of the monitoring provided by the network deployment (28%). Here we intend to verify how well the monitoring field is covered considering the proposed deployment. We can observe in Figure 6.21(b) that the system presents a high availability (>99%) for values of AQM_{min} from 0% to 28%. This analysis implies that, if an application intends to implement the proposed deployment, it will have a considerably high availability if its requirements of coverage area and monitoring area are $CA_{min} \leq 33\%$ and $AQM_{min} \leq 18\%$, respectively.

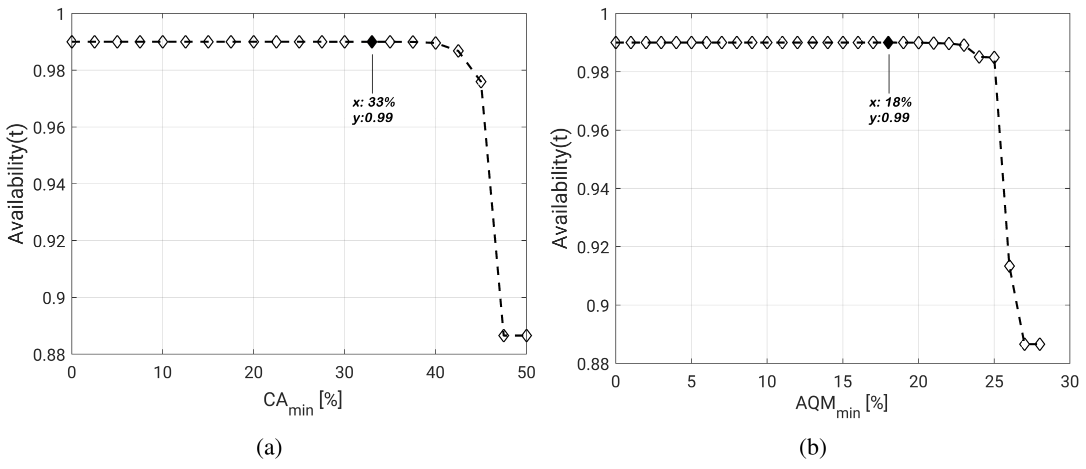


Figure 6.21: Impact of (**a**) CA_{min} and (**b**) AQM_{min} on dependability evaluation.

Next, we evaluate the system availability with the maximum values of CA_{min} and AQM_{min} that guarantee 99% of availability, which are $CA_{min} = 33\%$ and $AQM_{min} = 18\%$, and the maximum possible values, which are the coverage values of network deployment ($CA_{min} = 50\%$ and $AQM_{min} = 28\%$). With this, it is possible to verify, respectively, the maximum and minimum availability response which the system can present for the considered deployment. The Figure 6.22 shows the respective steady availability values, 88.66866% and 99.00779%.

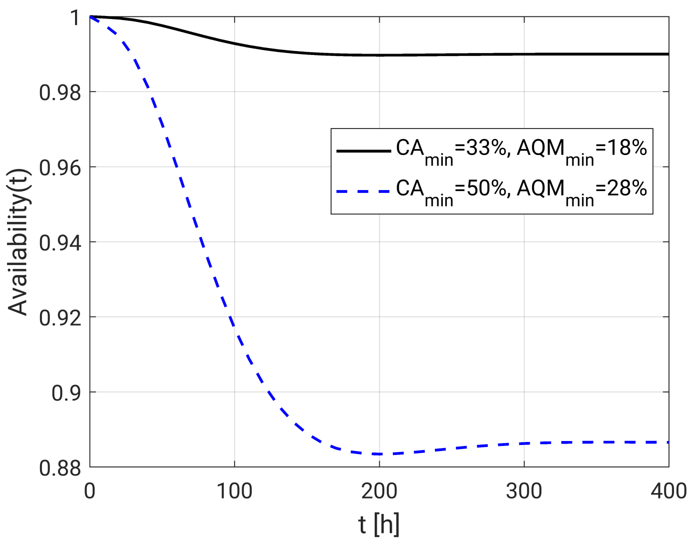


Figure 6.22: Availability at minimum and maximum acceptable visual failures.

If we consider that the nodes are supplied by a power line instead of a battery, we can re-evaluate the system dependability without the effect of batteries. This can be achieved by removing the battery Markov Chain from the model or simply by setting the battery discharge rates (λ_{bt}) to zero, avoiding to change the fault tree model. In this case, the system availability increases to 99.97261% and 99.67175%, respectively, achieving expected results of a well formed WirelessHART network (Muller et al., 2011).

Finally, it is analyzed the impact of transforming sensors with low coverage ($OFoV_{vs} < FoV_{min}$) into relay sensors. Figure 6.23 shows an example where the sensor vs_7 is occluded and its OFoV is less than 5%. In this case we consider that this sensor cannot provide enough visual information and then its camera is turned off to save battery, but the node keeps re-transmitting messages. We analyze the worst case scenario, that is, the maximum possible values of CA_{min} and AQM_{min}, which is the coverage values of network deployment ($CA_{min} = 47\%$ and $AQM_{min} = 26\%$). In this case, the application availability considering vs_7 as a visual sensor node is 89.56148%. When the sensor is considered as a relay node, the assessed availability is 89.56149%. This is a very small increase in terms of the absolute values, although, in terms of dependability, it is very significant, which shows that the proper management of power consumption turning some visual nodes into relay nodes compensates somehow the visual failure.

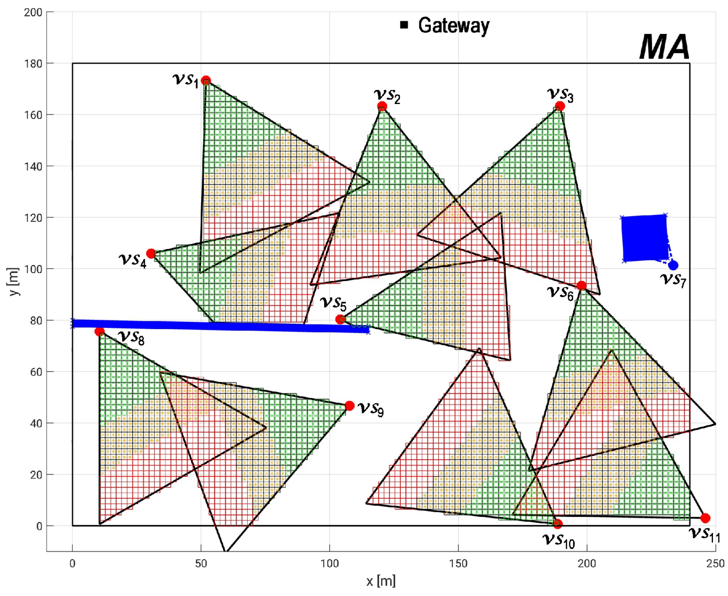


Figure 6.23: Network deployment with considerable FoV occlusion.

6.5 Conclusion

In this chapter, visual aspects related to system dependability have been discussed. First it was proposed a network redeployment according to the operating mechanisms of the proposed methodology, aiming dependability maximization. In that case, the methodology was also used to assess the dependability of the resulting network in order to assure the optimization.

Then, the methodology was extended to cope with some new issues, being able now to take into account visual coverage failures related to quality of monitoring with respect to the distance of view, as well as VCF related to the lost of coverage caused by occlusion. For this, the modelling of these VCF was detailed and integrated into the methodology.

Finally, practical aspects were approached based on the example of a realistic industrial application. We showed how to perform the methodology parameter settings for experimental purposes, with the concern to make the methodology feasible to use.

Part III

Conclusions and Future Work

Chapter 7

Conclusions and Future Work

This chapter states some conclusions of the work presented in this thesis and provides insights related to possible future works to be developed.

7.1 Thesis Validation and Contributions

This work is based on the hypothesis that " *is possible to evaluate quantitatively the dependability of wireless visual sensor networks for area coverage under visual coverage failures based on models that are realistic, simple to build and computationally feasible*". Here this hypothesis has been proven and we have provided a series of novel contributions. Thus, in this section we briefly discuss the validation of the stated thesis and the main contributions of this work.

In this thesis, dependability was approached in terms of reliability and availability, which are two attributes that are quantitative by definition and that characterize the successful operational behavior of the system over time. Reliability can be computed by the probability that a system or component will not fail, expressing the ability of a system or component to continuously perform its required functions, without interruption. On the other hand, availability is the probability of a system or component to be operating satisfactorily at any given point in time, considering its operating and repair times.

In order to evaluate these attributes of a WVSN, a methodology was proposed to deal specifically with area coverage applications, analyzing automatically simple models that describe the whole network together. The developed models are combined in a hierarchical structure which captures different aspects of the application, from the static topology of the network to the dynamic behavior of the operational nodes. Using these models and knowing some input data provided from the user (including the application requirements), the methodology is capable to compute individual area coverage of each visual sensor node and to merge it with the coverage of the neighbor visual nodes. The network topology (links and communication routes) and the hardware well function (generic hardware and battery) are analyzed in order to guarantee that it is possible to deliver the gathered visual data to the sink node.

The area coverage computation considers the occurrence of Visual Coverage Failures (VCF), which are impairments to the network to perform a sufficient area coverage. In this thesis, the modelled causes for coverage degradation are the reduction of the field of view of some cameras by occlusion and the loss of quality of monitoring from the network. These two types of the modelled causes for coverage degradation are the reduction of the field of view of some cameras by occlusion and the loss of quality of monitoring from the network. These two types of VCF are modelled and integrated to the methodology, providing a more comprehensive dependability assessment.

Computationally wise, the proposed models are simple, describing elementary network components and are integrated by a fault tree. Besides that, we focused on the usage of approximation and reductions on the algorithms for area coverage computation and the cut sets identification. These features create a conducive scenario to develop a tractable methodology that can be applied to a large range of applications. This computational advantage leverages the usage of the methodology as a tool for dependability optimization processes. For these cases, the dependability evaluation can be the main metric or can compose an objective function that determines a grading score of the current network deployment.

This thesis also discussed practical aspects of parameters setting, providing directions about how to use the methodology in real applications, potentially supporting more effective dependability evaluations. Moreover, it was considered the study case of a realistic industrial application of a paper mill where is applied a WVSN. This example can be easily replicated when evaluating other applications.

Summarizing, the proposed methodology is a useful tool for visual applications, being possible to applied in the design, operation and maintenance phases of the system, providing a basis of comparison for different scenarios. We believe that wireless visual sensor network applications can significantly benefit from the proposed approaches discussed in this thesis. In fact, we expect not only a more efficient planning of new monitoring applications, but also a continued understanding of how failures may impact the final coverage and quality and how and where improvements can be performed.

7.2 Future Work

This thesis discussed important issues related to dependability evaluation of wireless visual sensor networks, providing solutions specifically for area coverage applications. However some problems are still open, while some other appear as consequence of this work. Next, we discuss some issues that should receive more attention in future works.

First, the entire methodology depends on the way of the network is represented by the developed models, which entails some hypothesis and approximations. The methodology can use more detailed models to be more realistic, and also include more elements, such as more routing strategies, different battery failures, common cause failures (nodes and links), etc. Of course, this action needs to be carefully balanced with the produced benefits, since more complex models imply more parameters to be estimated, measured and set, which may be a prohibitive task.

Since it was noticed that dependability evaluation is more sensitive to hardware failures from battery discharge, a more realistic model could be designed, taking into account other faults, such as short circuit, dead cell, overcharge and overdischarge. Another way to achieve a more realistic model is to integrate the battery modelling directly to the routing protocol modelling, in order to associate the average discharging current according to the selected routing strategy.

As important as the described models, the application coverage can also be enhanced. Since the monitored area is a continuous space, it is hard to determine what area has been covered by which visual sensors, which makes the definition of the minimum sensors set necessary to perform the area coverage a challenging task. However, this issue should be discussed in order to enhance the usage of resources to provide a high quality area coverage with the minimum effort.

Furthermore, dealing with area coverage implies the definition of the best position and orientation of the visual nodes to cover a wider area. This is intrinsically related to the reduction of the coverage redundancy among the visual nodes. On the other hand, in the sense of generating higher dependability, redundancy can be increased. This can be done through the usage of spare nodes, even whether this measure means that resources are underused or wasted. At this point, it is necessary to consider the trade-off between increasing redundancy and saving power, as well as between increasing redundancy and increasing area coverage.

Even when the application performs an expected coverage, external factors can jeopardize it. In this thesis, it was only considered coverage failures by loss of quality or occlusion. Other aspects can decrease the effective amount and quality of area coverage, and thus they should be proper modelled and integrated into the methodology. For outdoor monitoring, visual sensors may also become faulty during the night or under certain weather conditions. When regular cameras are employed, the ambient light may determine if a certain visual sensor is under a coverage failure, since the retrieved visual data may become useless under low ambient light. On the other hand, visual data processing algorithms may be used to identify if a camera's lens is dirty or with too many water drops. Alternatively, additional sensors may identify adverse weather conditions that can be mathematically computed when processing coverage failures.

Another possible future work would encompass the extension of the coverage model in order to better describe the monitoring applications. For this purpose, the cameras' field of view could consider a 3D coverage instead of a 2D. This implies into the development of computational efficient volumetric coverage methods. Also, all models related to coverage (occlusion, QoM) should be revisited.

The quality of monitoring can also be improved. In this thesis, the sharpness in an image (or video) is described by discrete ranges of distance within the field of view of the cameras, with different quality levels. In a different and more realistic way, the sharpness could be described considering a continuous variation of quality of image associated to a continuous variation of distance from the camera. This description can be even better whether we add in the quality assessment the effect of the angle of view that a camera faces a region. The angle of view represents the difference between the direction of view of the visual sensor node and the frontal face of a region. A small angle of view implies in a frontal perception of the region of interest, which results in a high quality of monitoring.

Finally, although barrier coverage applications are not a issue in the scope of this thesis, they are a related topic that was under our attention during the development of this work. The performed literature review showed the lack of works approaching dependability and coverage failures in such applications. These are very important issues in order to be able to classify and identify an object or an intruder violating a barrier, as well as to determine the duration of the expected barrier monitoring and successful operational application behavior. Dependability metrics should be proposed to assess specifically barrier coverage applications or, at least, should be developed feasibility studies about the adaptation and application of metrics used for area and target coverage to barrier coverage applications. More than that, redundancy should be considered as determining factor for dependability assessment in barrier coverage applications, since a visual sensor failure can create a hole in the barrier. This assessment can help to design and schedule preventive or contingency measures.

Bibliography

Ahlen, A., Akerberg, J., Eriksson, M., Isaksson, A. J., Iwaki, T., Johansson, K. H., Knorn, S., Lindh, T., and Sandberg, H. (2019). Toward wireless control in industrial process automation: A case study at a paper mill. *IEEE Control Systems Magazine*, 39(5):36–57.

Ahmad, W., Hasan, O., and Tahar, S. (2016). *Formal Dependability Modeling and Analysis: A Survey*, pages 132–147. Springer International Publishing, Cham.

Ahmed, W., Hasan, O., Pervez, U., and Qadir, J. (2017). Reliability modeling and analysis of communication networks. *Journal of Network and Computer Applications*, 78:191 – 215.

Akyildiz, I. F., Melodia, T., and Chowdhury, K. R. (2007). A survey on wireless multimedia sensor networks. *Computer Networks*, 51:921–960.

Al-Karaki, J. N. and Gawanmeh, A. (2017). The optimal deployment, coverage, and connectivity problems in wireless sensor networks: Revisited. *IEEE Access*, 5:18051–18065.

Almalkawi, I. T., Guerrero Zapata, M., Al-Karaki, J. N., and Morillo-Pozo, J. (2010). Wireless multimedia sensor networks: Current trends and future directions. *Sensors*, 10(7):6662–6717.

Andrade, E. and Nogueira, B. (2020). Dependability evaluation of a disaster recovery solution for iot infrastructures. *The Journal of Supercomputing*, 76(3):1828–1849.

Andreescu, T. and Feng, Z. (2003). *A Path to Combinatorics for Undergraduates: Counting Strategies*. Birkhäuser Boston.

Avizienis, A., Laprie, J. C., Randell, B., and Landwehr, C. (2004). Basic concepts and taxonomy of dependable and secure computing. *IEEE Transactions on Dependable and Secure Computing*, 1(1):11–33.

Bai, H. and Atiquzzaman, M. (2003). Error modeling schemes for fading channels in wireless communications: A survey. *IEEE Communications Surveys Tutorials*, 5(2):2–9.

Banfi, J., Basilico, N., and Carpin, S. (2018). Optimal redeployment of multirobot teams for communication maintenance. In *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 3757–3764.

Bause, F. and Kritzinger, P. (2002). *Stochastic Petri Nets: An Introduction to the Theory*. Vieweg+Teubner Verlag.

Bernardi, S., Merseguer, J., and Petriu, D. C. (2012). Dependability modeling and analysis of software systems specified with uml. *ACM Comput. Surv.*, 45(1).

Billinton, R. and Allan, R. N. (1992). *Reliability Evaluation of Engineering Systems: Concepts and Techniques*. Springer US, 2nd edition.

Bondavalli, A., Ceccarelli, A., Falai, L., and Vadursi, M. (2010). A new approach and a related tool for dependability measurements on distributed systems. *IEEE Transactions on Instrumentation and Measurement*, 59(4):820–831.

Brualdi, R. (1977). *Introductory combinatorics*. North-Holland.

Bruneo, D., Distefano, S., Longo, F., Puliafito, A., and Scarpa, M. (2012). Evaluating wireless sensor node longevity through markovian techniques. *Computer Networks*, 56(2):521 – 532.

Bruneo, D., Puliafito, A., and Scarpa, M. (2010). Dependability evaluation of wireless sensor networks: Redundancy and topological aspects. In *2010 IEEE Sensors*, pages 1827–1831.

Bujorianu, L. M. (2012). *Stochastic Reachability Analysis of Hybrid Systems*, pages E1–E3. Springer London, London.

Charfi, Y., Wakamiya, N., and Murata, M. (2009). Challenging issues in visual sensor networks. *IEEE Wireless Communications*, 16(2):44–49.

Cinque, M., Cotroneo, D., and Martino, C. D. (2012). Automated generation of performance and dependability models for the assessment of wireless sensor networks. *IEEE Transactions on Computers*, 61(6):870–884.

Coronato, A. and Testa, A. (2013). Approaches of wireless sensor network dependability assessment. In *2013 Federated Conference on Computer Science and Information Systems*, pages 881–888.

Costa, D. G. (2020). Visual sensors hardware platforms: A review. *IEEE Sensors Journal*, 20(8):4025–4033.

Costa, D. G. and Duran-Faundez, C. (2016). Assessing availability in wireless visual sensor networks based on targets' perimeters coverage. *Journal of Electrical and Computer Engineering*, 2016:14.

Costa, D. G., Duran-Faundez, C., and Bittencourt, J. C. N. (2017a). Availability issues for relevant area coverage in wireless visual sensor networks. In *2017 CHILEAN Conf. on Electrical, Electronics Engineering, Information and Communication Technologies*, pages 1–6.

Costa, D. G. and Guedes, L. A. (2010). The coverage problem in video-based wireless sensor networks: A survey. *Sensors*, 10(9):8215–8247.

Costa, D. G., Rangel, E., Peixoto, J. P. J., and Jesus, T. C. (2019). An availability metric and optimization algorithms for simultaneous coverage of targets and areas by wireless visual sensor networks. In *2019 IEEE 17th International Conference on Industrial Informatics (INDIN)*, pages 617–622.

Costa, D. G., Silva, I., Guedes, L. A., Portugal, P., and Vasques, F. (2014a). Availability assessment of wireless visual sensor networks for target coverage. In *Proceedings of the 2014 IEEE Emerging Technology and Factory Automation (ETFA)*, pages 1–8.

Costa, D. G., Silva, I., Guedes, L. A., Portugal, P., and Vasques, F. (2014b). Selecting redundant nodes when addressing availability in wireless visual sensor networks. In *2014 12th IEEE International Conference on Industrial Informatics (INDIN)*, pages 130–135.

Costa, D. G., Silva, I., Guedes, L. A., Vasques, F., and Portugal, P. (2014c). Availability issues in wireless visual sensor networks. *Sensors*, 14(2):2795–2821.

Costa, D. G., Vasques, F., and Portugal, P. (2017b). Enhancing the availability of wireless visual sensor networks: Selecting redundant nodes in networks with occlusion. *Applied Mathematical Modelling*, 42:223 – 243.

Dar, K. S., Taherkordi, A., and Eliassen, F. (2016). Enhancing dependability of cloud-based iot services through virtualization. In *2016 IEEE First International Conference on Internet-of-Things Design and Implementation (IoTDI)*, pages 106–116.

Derasevic, S. (2018). *Node fault tolerance for distributed embedded systems based on FTT-Ethernet*. PhD thesis, Universitat de les Illes Balears.

Distefano, S., Longo, F., and Trivedi, K. S. (2012). Investigating dynamic reliability and availability through state–space models. *Computers & Mathematics with Applications*, 64(12):3701 – 3716.

Doerffel, D. and Sharkh, S. A. (2006). A critical review of using the peukert equation for determining the remaining capacity of lead-acid and lithium-ion batteries. *Journal of Power Sources*, 155(2):395 – 400.

Dubrova, E. (2013). *Fundamentals of Dependability*, pages 5–20. Springer New York, New York, NY.

Dâmaso, A., Rosa, N., and Maciel, P. (2014). Reliability of wireless sensor networks. *Sensors*, 14(9):15760–15785.

Dâmaso, A., Rosa, N., and Maciel, P. (2017). Integrated evaluation of reliability and power consumption of wireless sensor networks. *Sensors*, 17(11).

Egeland, G. and Engelstad, P. E. (2009). The availability and reliability of wireless multi-hop networks with stochastic link failures. *IEEE Journal on Selected Areas in Communications*, 27(7):1132–1146.

Elghazel, W., Bahi, J., Guyeux, C., Hakem, M., Medjaher, K., and Zerhouni, N. (2015). Dependability of wireless sensor networks for industrial prognostics and health management. *Computers in Industry*, 68:1 – 15.

Endress+Hauser (2016). Wirelesshart adapter SWA70. Technical Information. (TI00026S/04/EN/21.16 71339584) 2016.

FLIR® Integrated Imaging Solutions Inc (2017). *FLIR BLACKFLY®S BFS-U3-13Y3 Reliability*. `http://softwareservices.flir.com/BFS-U3-13Y3/latest/Quality/MTBF.htm` [Accessed: June 10th, 2020].

Frühwirth, T., Krammer, L., and Kastner, W. (2015). Dependability demands and state of the art in the internet of things. In *2015 IEEE 20th Conference on Emerging Technologies Factory Automation (ETFA)*, pages 1–4.

Ghazalian, R., Aghagolzadeh, A., and Hosseini Andargoli, S. M. (2020). Energy optimization of wireless visual sensor networks with the consideration of the desired target coverage. *IEEE Transactions on Mobile Computing*, pages 1–1.

Giyenko, A. and Cho, Y. I. (2016). Intelligent uav in smart cities using iot. In *2016 16th International Conference on Control, Automation and Systems (ICCAS)*, pages 207–210.

Gokhale, S. S. and Trivedi, K. S. (1998). Analytical modeling. *Encyclopedia of Distributed Systems*.

Gomez, C., Cuevas, A., and Paradells, J. (2006). AHR: A two-state adaptive mechanism for link connectivity maintenance in AODV. In *Proceedings of the 2nd International Workshop on Multi-Hop Ad Hoc Networks: From Theory to Reality*, REALMAN '06, page 98–100, New York, NY, USA. Association for Computing Machinery.

Gross, J. (2008). *Combinatorial methods with computer applications*. Discrete mathematics and its applications. Chapman & Hall/CRC.

Hanoun, S., Bhatti, A., Creighton, D., Nahavandi, S., Crothers, P., and Esparza, C. G. (2016). Target coverage in camera networks for manufacturing workplaces. *Journal of Intelligent Manufacturing*, 27:1221 – 1235.

He, S., Shin, D. H., Zhang, J., Chen, J., and Sun, Y. (2016). Full-view area coverage in camera sensor networks: Dimension reduction and near-optimal solutions. *IEEE Transactions on Vehicular Technology*, 65(9):7448–7461.

Hirel, C., Sahner, R., Zang, X., and Trivedi, K. (2000). Reliability and performability modeling using sharpe 2000. In Haverkort, B. R., Bohnenkamp, H. C., and Smith, C. U., editors, *Computer Performance Evaluation.Modelling Techniques and Tools*, pages 345–349, Berlin, Heidelberg. Springer Berlin Heidelberg.

Høyland, A. and Rausand, M. (1994). *System reliability theory: models and statistical methods*. Wiley series in probability and mathematical statistics: Applied probability and statistics. J. Wiley.

Hsiao, Y. P., Shih, K. P., and Chen, Y. D. (2017). On full-view area coverage by rotatable cameras in wireless camera sensor networks. In *2017 IEEE 31st International Conference on Advanced Information Networking and Applications (AINA)*, pages 260–265.

Huang, J., Lin, C., Kong, X., Wei, B., and Shen, X. (2014). Modeling and analysis of dependability attributes for services computing systems. *IEEE Transactions on Services Computing*, 7(4):599–613.

Istin, C., Pescaru, D., and Ciocarlie, H. (2010). Performance improvements of video wsn surveillance in case of traffic congestions. In *2010 International Joint Conference on Computational Cybernetics and Technical Informatics*, pages 659–663.

Jesus, T. C., Costa, D. G., and Portugal, P. (2018a). On the computing of area coverage by visual sensor networks: assessing performance of approximate and precise algorithms. In *16th IEEE International Conference on Industrial Informatics (INDIN)*.

Jesus, T. C., Costa, D. G., and Portugal, P. (2019). Wireless visual sensor networks redeployment based on dependability optimization. In *2019 IEEE 17th International Conference on Industrial Informatics (INDIN)*, pages 1111–1116.

Jesus, T. C., Costa, D. G., Portugal, P., and Vasques, F. (2020a). Fov-based quality assessment and optimization for area coverage in wireless visual sensor networks. *IEEE Access*, 8:109568–109580.

Jesus, T. C., Costa, D. G., Portugal, P., Vasques, F., and Aguiar, A. (2020b). Modelling coverage failures caused by mobile obstacles for the selection of faultless visual nodes in wireless sensor networks. *IEEE Access*, 8:41537–41550.

Jesus, T. C., Portugal, P., Costa, D. G., and Vasques, F. (2020). A comprehensive dependability model for qom-aware industrial wsn when performing visual area coverage in occluded scenarios. *Sensors*, 20(22):6542.

Jesus, T. C., Portugal, P., Vasques, F., and Costa, D. G. (2018b). Automated methodology for dependability evaluation of wireless visual sensor networks. *Sensors*, 18(8).

Jia, J., Dong, C., Hong, Y., Guo, L., and Yu, Y. (2019). Maximizing full-view target coverage in camera sensor networks. *Ad Hoc Networks*, 94:101973.

Jiang, F., Zhang, X., Chen, X., and Fang, Y. (2020). Distributed optimization of visual sensor networks for coverage of a large-scale 3-d scene. *IEEE/ASME Transactions on Mechatronics*, pages 1–1.

Kafi, M. A., Othman, J. B., and Badache, N. (2017). A survey on reliability protocols in wireless sensor networks. *ACM Comput. Surv.*, 50(2):31:1–31:47.

Karl, H. and Willig, A. (2005). *Protocols and Architectures for Wireless Sensor Networks*. John Wiley & Sons.

Konda, K. R., Conci, N., and Natale, F. D. (2016). Global coverage maximization in ptz-camera networks based on visual quality assessment. *IEEE Sensors Journal*, 16(16):6317–6332.

Kuawattanaphan, R., Kumrai, T., and Champrasert, P. (2013). Wireless sensor nodes redeployment using a multiobjective optimization evolutionary algorithm. In *2013 IEEE International Conference of IEEE Region 10 (TENCON 2013)*, pages 1–6.

Kularathne, D. and Jayarathne, L. (2018). Point in polygon determination algorithm for 2-d vector graphics applications. In *2018 National Information Technology Conference (NITC)*, pages 1–5.

Kumar, S., Deshpande, A., Ho, S. S., Ku, J. S., and Sarma, S. E. (2016). Urban street lighting infrastructure monitoring using a mobile sensor platform. *IEEE Sensors Journal*, 16(12):4981–4994.

Lanus, M., Yin, L., and Trivedi, K. S. (2003). Hierarchical composition and aggregation of state-based availability and performability models. *IEEE Transactions on Reliability*, 52(1):44–52.

Limnios, N. (2010). *Fault Trees*. ISTE. Wiley.

Macedo, D., Guedes, L. A., and Silva, I. (2014). A dependability evaluation for internet of things incorporating redundancy aspects. In *Proceedings of the 11th IEEE International Conference on Networking, Sensing and Control*, pages 417–422.

Malhotra, M. and Trivedi, K. S. (1994). Power-hierarchy of dependability-model types. *IEEE Transactions on Reliability*, 43(3):493–502.

Malhotra, M. and Trivedi, K. S. (1995). Dependability modeling using petri-nets. *IEEE Transactions on reliability*, 44(3):428–440.

Manno, G. A. (2012). *Reliability modelling of complez systems: an adaptive transition system approach to match accuracy and efficiency*. PhD thesis, Università degli Studi di Catania, Catania, Italy.

Marsan, M. A. (1988). Stochastic petri nets: an elementary introduction. In *European workshop on applications and theory in Petri nets*, pages 1–29. Springer.

Martins, M., Portugal, P., and Vasques, F. (2015). A framework to support dependability evaluation of wsns from aadl models. In *2015 IEEE 20th Conference on Emerging Technologies Factory Automation (ETFA)*, pages 1–6.

Mavrinac, A. and Chen, X. (2013). Modeling coverage in camera networks: A survey. *International Journal of Computer Vision*, 101(1):205–226.

Maza, S. (2013). Observer-based diagnosis modeling using stochastic activity networks for the dependability assessment purpose. In *2013 Conference on Control and Fault-Tolerant Systems (SysTol)*, pages 79–84.

Medvedev, A., Fedchenkov, P., Zaslavsky, A., Anagnostopoulos, T., and Khoruzhnikov, S. (2015). Waste management as an iot-enabled service in smart cities. In Balandin, S., Andreev, S., and Koucheryavy, Y., editors, *Internet of Things, Smart Spaces, and Next Generation Networks and Systems*, pages 104–115, Cham. Springer International Publishing.

Meyn, S. and Tweedie, R. L. (2009). *Markov Chains and Stochastic Stability*. Cambridge University Press, USA, 2nd edition.

Mirzazadeh Moallem, M., Aghagolzadeh, A., and Ghazalian, R. (2020). Wireless visual sensor networks energy optimization based on new entropy model. *IEEE Sensors Journal*, 20(2):778–785.

Misra, K. B. (2008). *Dependability Considerations in the Design of a System*, pages 71–80. Springer London, London.

Miyazaki, M., Fujiwara, R., Mizugaki, K., and Kokubo, M. (2012). Adaptive channel diversity method based on isa100.11a standard for wireless industrial monitoring. In *2012 IEEE Radio and Wireless Symposium*, pages 131–134.

Muller, I., Netto, J. C., and Pereira, C. E. (2011). Wirelesshart field devices. *IEEE Instrumentation Measurement Magazine*, 14(6):20–25.

Neishaboori, A., Saeed, A., Harras, K. A., and Mohamed, A. (2014). On target coverage in mobile visual sensor networks. In *Proceedings of the 12th ACM International Symposium on Mobility Management and Wireless Access*, MobiWac '14, page 39–46, New York, NY, USA. Association for Computing Machinery.

Nguyen, T. G. and So-In, C. (2018). Distributed deployment algorithm for barrier coverage in mobile sensor networks. *IEEE Access*, 6:21042–21052.

Omar, N., Bossche, P. V. d., Coosemans, T., and Mierlo, J. V. (2013). Peukert revisited—critical appraisal and need for modification for lithium-ion batteries. *Energies*, 6(11):5625–5641.

O'Connor, A. N., Modarres, M., and Mosleh, A. (2016). *Probability Distributions Used in Reliability Engineering*. Center for Risk and Reliability.

Pirsiavash, H. and Ramanan, D. (2012). Detecting activities of daily living in first-person camera views. In *2012 IEEE Conference on Computer Vision and Pattern Recognition*, pages 2847–2854.

Poledna, S. (2007). *Fault-tolerant real-time systems: The problem of replica determinism*, volume 345. Springer Science & Business Media.

Pretschner, A., Holling, D., Eschbach, R., and Gemmar, M. (2013). A generic fault model for quality assurance. In Moreira, A., Schätz, B., Gray, J., Vallecillo, A., and Clarke, P., editors, *Model-Driven Engineering Languages and Systems*, pages 87–103, Berlin, Heidelberg. Springer Berlin Heidelberg.

Purba, J. H., Lu, J., Zhang, G., and Ruan, D. (2012). An area defuzzification technique to assess nuclear event reliability data from failure possibilities. *International Journal of Computational Intelligence and Applications*, 11(04):1250022.

Pure, R. and Durrani, S. (2015). Computing exact closed-form distance distributions in arbitrarily-shaped polygons with arbitrary reference point. *The Mathematica Journal*, 17:1–27.

Rangel, E. O., Costa, D. G., and Loula, A. (2018). Redundant visual coverage of prioritized targets in iot applications. In *Proceedings of the 24th Brazilian Symposium on Multimedia and the Web*, WebMedia '18, pages 307–314, Salvador, BA, Brazil. ACM.

Raptis, T. P., Passarella, A., and Conti, M. (2018). Performance analysis of latency-aware data management in industrial iot networks. *Sensors*, 18(8).

Rausand, M. and Høyland, A. (2004). *System Reliability Theory: Models, Statistical Methods and Applications*. Wiley-Interscience, Hoboken, NJ.

Rodrigues, L. M., Montez, C., Moraes, R., Portugal, P., and Vasques, F. (2017). A temperature-dependent battery model for wireless sensor networks. *Sensors*, 17(2).

Rodrigues, L. M., Montez, C., Vasques, F., and Portugal, P. (2016). Experimental validation of a battery model for low-power nodes in wireless sensor networks. In *2016 IEEE World Conference on Factory Communication Systems (WFCS)*, pages 1–4.

Sandborn, P. and Myers, J. (2008). *Designing Engineering Systems for Sustainability*, pages 81–103. Springer London, London.

Sanders, W. H. and Meyer, J. F. (2001). *Stochastic Activity Networks: Formal Definitions and Concepts*, pages 315–343. Springer Berlin Heidelberg, Berlin, Heidelberg.

Scott, K., Dai, R., and Kumar, M. (2016). Occlusion-aware coverage for efficient visual sensing in unmanned aerial vehicle networks. In *2016 IEEE Global Communications Conference (GLOBECOM)*, pages 1–6, Washington, DC, USA.

Senouci, M. R. and Abdellaoui, A. (2017). Efficient sensor placement heuristics. In *2017 IEEE International Conference on Communications (ICC)*, pages 1–6.

Senouci, M. R., Mellouk, A., Senouci, H., and Aissani, A. (2012). Performance evaluation of network lifetime spatial-temporal distribution for wsn routing protocols. *Journal of Network and Computer Applications*, 35(4):1317 – 1328. Intelligent Algorithms for Data-Centric Sensor Networks.

Shah, V. R., Maru, S. V., and Jhaveri, R. H. (2016). An obstacle detection scheme for vehicles in an intelligent transportation system. *International Journal of Computer Network and Information Security*, 8(10):23.

Shao, Z., Cai, J., and Wang, Z. (2017). Smart monitoring cameras driven intelligent processing to big surveillance video data. *IEEE Transactions on Big Data*, 4(1):105–116.

Shi, H., Sun, G., Wang, Y., and Hwang, K. (2019). Adaptive image-based visual servoing with temporary loss of the visual signal. *IEEE Transactions on Industrial Informatics*, 15(4):1956–1965.

Shooman, M. L. (2002). *Reliability of Computer Systems and Networks: Fault Tolerance, Analysis, and Design*. John Wiley & Sons, Inc., USA.

Shriwastav, S. and Song, Z. (2020). Coordinated coverage and fault tolerance using fixed-wing unmanned aerial vehicles. In *2020 International Conference on Unmanned Aircraft Systems (ICUAS)*, pages 1231–1240, Athens, Greece.

Si, P., Wu, C., Zhang, Y., Jia, Z., Ji, P., and Chu, H. (2017). Barrier coverage for 3d camera sensor networks. *Sensors*, 17(8).

Silva, I., Guedes, L. A., Portugal, P., and Vasques, F. (2012). Reliability and availability evaluation of wireless sensor networks for industrial applications. *Sensors*, 12(1):806–838.

Silva, I., Leandro, R., Macedo, D., and Guedes, L. A. (2013). A dependability evaluation tool for the internet of things. *Computers & Electrical Engineering*, 39(7):2005 – 2018.

Soro, S. and Heinzelman, W. (2009). A survey of visual sensor networks. *Advances in Multimedia*, 2009.

Soro, S. and Heinzelman, W. B. (2005). On the coverage problem in video-based wireless sensor networks. In *2nd International Conference on Broadband Networks, 2005.*, pages 932–939 Vol. 2.

Tanwar, S., Patel, P., Patel, K., Tyagi, S., Kumar, N., and Obaidat, M. S. (2017). An advanced internet of thing based security alert system for smart home. In *2017 International Conference on Computer, Information and Telecommunication Systems (CITS)*, pages 25–29.

Tao, J., Zhai, T., Wu, H., Xu, Y., and Dong, Y. (2017). A quality-enhancing coverage scheme for camera sensor networks. In *43rd Annual Conference of the IEEE Industrial Electronics Society*, pages 8458–8463.

Texas Instruments (2005). CC2420 reliability report. SWRK007 Report. (rev. 1.2) 2005-04-29.

Toth, J. and Gilpin-Jackson, A. (2010). Smart view for a smart grid — unmanned aerial vehicles for transmission lines. In *2010 1st International Conference on Applied Robotics for the Power Industry*, pages 1–6.

Trivedi, K. S. and Bobbio, A. (2017). *Reliability and availability engineering: modeling, analysis, and applications*. Cambridge University Press.

Trivedi, K. S., Grottke, M., and Andrade, E. (2010). Software fault mitigation and availability assurance techniques. *International Journal of System Assurance Engineering and Management*, 1(4):340–350.

Trivedi, K. S., Kim, D. S., Roy, A., and Medhi, D. (2009). Dependability and security models. In *2009 7th International Workshop on Design of Reliable Communication Networks*, pages 11–20. IEEE.

Trivedi, K. S. and Sahner, R. (2009). Sharpe at the age of twenty two. *SIGMETRICS Perform. Eval. Rev.*, 36(4):52–57.

Vesely, W. E., Goldberg, F. F., Roberts, N. H., and Haasl, D. F. (1981). Fault tree handbook. Technical report, Nuclear Regulatory Commission Washington DC.

Wang, Q. and Jiang, J. (2016). Comparative examination on architecture and protocol of industrial wireless sensor network standards. *IEEE Communications Surveys Tutorials*, 18(3):2197–2219.

Wang, Q. and Wang, P. (2010). A finite-state markov model for reliability evaluation of industrial wireless network. In *2010 6th International Conference on Wireless Communications Networking and Mobile Computing (WiCOM)*, pages 1–4.

Wang, Y. and Cao, G. (2011). On full-view coverage in camera sensor networks. In *2011 Proceedings IEEE INFOCOM*, pages 1781–1789.

Wang, Z. and Wang, F. (2019). Wireless visual sensor networks: Applications, challenges, and recent advances. In *2019 SoutheastCon*, pages 1–8.

Warns, T. (2010). *Structural Failure Models for Fault-Tolerant Distributed Computing*. Vieweg+Teubner.

Westhofen, D., Gründler, C., Doll, K., Brunsmann, U., and Zecha, S. (2012). Transponder- and camera-based advanced driver assistance system. In *2012 IEEE Intelligent Vehicles Symposium*, pages 293–298.

Yap, F. G. H. and Yen, H. (2017). Novel visual sensor deployment algorithm in occluded wireless visual sensor networks. *IEEE Systems Journal*, 11(4):2512–2523.

Ye, Y., Guangrui, F., and Shiqi, O. (2013). An algorithm for judging points inside or outside a polygon. In *2013 Seventh International Conference on Image and Graphics*, pages 690–693.

Zannat, H., Akter, T., Tasnim, M., and Rahman, A. (2016). The coverage problem in visual sensor networks: A target oriented approach. *Journal of Network and Computer Applications*, 75:1 – 15.

Zhang, X., Zhang, B., Chen, X., and Fang, Y. (2019). Coverage optimization of visual sensor networks for observing 3-d objects: survey and comparison. *International Journal of Intelligent Robotics and Applications*, 3:342–361.

Zhao, G., Xing, L., Zhang, Q., and Jia, X. (2018). A hierarchical combinatorial reliability model for smart home systems. *Quality and Reliability Engineering International*, 34(1):37–52.

Zonouz, A. E., Xing, L., Vokkarane, V. M., and Sun, Y. L. (2014). A time-dependent link failure model for wireless sensor networks. In *2014 Reliability and Maintainability Symposium*, pages 1–7.