Embedded software development for a real-time locating system

Tero Liukko

School of Electrical Engineering

Thesis submitted for examination for the degree of Master of Science in Technology.

Espoo 16.5.2017

Thesis supervisor:

Prof. Ville Kyrki

Thesis advisor:

M.Sc. (Tech.) Antti Renko

Aalto University
School of Electrical Engineering

AALTO UNIVERSITY
SCHOOL OF ELECTRICAL ENGINEERING

ABSTRACT OF THE
MASTER'S THESIS

Author: Tero Liukko

Title: Embedded software development for a real-time locating system

Date: 16.5.2017 Language: English Number of pages: 14+86

Department of Electrical Engineering and Automation

Professorship: Automation technology

Supervisor: Prof. Ville Kyrki

Advisor: M.Sc. (Tech.) Antti Renko

Asset tracking often necessitates wireless, radio-frequency identification (RFID). In practice, situations often arise where plain inventory operations are not sufficient, and methods to estimate movement trajectory are needed for making reliable observations, classification and report generation.

In this thesis, an embedded software application for an industrial, resource-constrained off-the-shelf RFID reader device in the UHF frequency range is designed and implemented. The software is used to configure the reader and its air-interface operations, accumulate read reports and generate events to be reported over network connections. Integrating location estimation methods to the application facilitates the possibility to make deploying middleware RFID solutions more streamlined and robust while reducing network bandwidth requirements.

The result of this thesis is a functional embedded software application running on top of an embedded Linux distribution on an ARM processor. The reader software is used commercially in industrial and logistics applications. Non-linear state estimation features are applied, and their performance is evaluated in empirical experiments.

Keywords: radio-frequency identification, received signal strength, inverse synthetic aperture radar, Unscented Kalman filter, real-time locating system, indoor positioning, embedded software

AALTO-YLIOPISTO
SÄHKÖTEKNIIKAN KORKEAKOULU

DIPLOMITYÖN
TIIVISTELMÄ

Tekijä: Tero Liukko

Työn nimi: Sulautettu ohjelmistototeutus reaaliaikaiseen paikannusjärjestelmään

Päivämäärä: 16.5.2017 Kieli: Englanti Sivumäärä: 14+86

Sähkötekniikan ja automaation laitos

Professuuri: Automaatiotekniikka

Työn valvoja: Prof. Ville Kyrki

Työn ohjaaja: DI Antti Renko

Tavaroiden seuranta edellyttää usein langatonta radiotaajuustunnistustekniikkaa (RFID). Käytännön sovelluksissa tulee monesti tilanteita joissa pelkkä inventointi ei riitä, vaan tarvitaan menetelmiä liikeradan estimointiin luotettavien havaintojen ja luokittelun tekemiseksi sekä raporttien generoimiseksi.

Tässä työssä on suunniteltu ja toteutettu sulautettu ohjelmistosovellus teolliseen, resursseiltaan rajoitettuun ja kaupallisesti saatavaan UHF-taajuusalueen RFID-lukijalaitteeseen. Ohjelmistoa käytetään lukijalaitteen ja sen ilmarajapinnan toimintojen konfigurointiin, lukutapahtumien keräämiseen ja raporttien lähettämiseen verkkoyhteyksiä pitkin. Paikkatiedon estimointimenetelmien integroiminen ohjelmistoon mahdollistaa välitason RFID-sovellusten toteuttamisen aiempaa suoraviivaisemin ja luotettavammin, vähentäen samalla vaatimuksia tietoverkon kaistanleveydelle.

Työn tuloksena on toimiva sulautettu ohjelmistosovellus, jota ajetaan sulautetussa Linux-käyttöjärjestelmässä ARM-arkkitehtuurilla. Lukijaohjelmistoa käytetään kaupallisesti teollisuuden ja logistiikan sovelluskohteissa. Epälineaarisia estimointiominaisuuksia hyödynnetään, ja niiden toimivuutta arvioidaan empiirisin kokein.

Avainsanat: radiotaajuustunnistus, ISAR-tutka, Unscented Kalman –suodin, reaaliaikainen paikannus, sisätilapaikannus, sulautettu ohjelmisto

Preface

I have had the priviledge of having been able to choose my own path in education, and it has led me getting involved with UHF RFID technology all the way since upper secondary level studies. I have also had the chance to continue practicing my expertise in the field for the duration of my studies in Aalto. Getting a subject for this thesis was not as straightforward a process as I would have hoped, and initially I preferred something other than RFID related study for a change, but now that I look back, I am more contented about the choice than I thought I could ever be. I now see how important the time I have spent with the people and the code had become for me, and being able to finally document some of the results for others to see and read in the form of a thesis makes me happy.

The writing process itself was smooth enough in the beginning to eventually lead to deadline-induced panic. Although I have not yet let my perfectionism to completely get the best of me, there are still issues left with this thesis I wish I had had the time for to make right. There would have been much more to be said, but the limits for the scope have to be set somewhere. Also who would have thought that peer-reviewed research articles could contain so many errors in mathematical notation or discussion? Some of the most fascinating ones did not end up being discussed here in the end for that very reason.

Nevertheless, the journey has now come to its next waypoint and it is time to acknowledge the people without whom this achievement would be void. I want to thank my instructor Antti Renko and my excellent professor Ville Kyrki for their effort in giving feedback for this thesis, and Ville for his mentoring over the past year. I would like to express my gratitude to Suvi Räisänen for giving invaluable help in proofreading the whole thesis on a tight schedule. I would also like to thank Asko Puoliväli and his team for the opportunity I have had these past years in working with them in an effort to develop the revolutionary on-reader application.

My dearest thanks go to my family and friends, to my cousin and roommate Aapo, for keeping me sane, and to my beloved parents who have always supported and encouraged me, and been there for me since time immemorial.

Tero Kristian Liukko

Espoo, 16.5.2017

Contents

List of Tables

List of Figures

List of Symbols

Radio signals

Ω	Ohm (unit of electrical impedance)
λ	Wavelength
A	Antenna aperture
c	Speed of light
f	Frequency
G	Antenna gain
g	Antenna gain
P	Power
v	Velocity
r	Distance from an antenna
Γ	Reflection coefficient
n	Loss exponent
PL	Path loss
φ	Phase angle
t	Time
ω	Angular frequency
$\mathbb{N}$	Set of natural numbers
$\mathbf{p}$	Target position
$\mathbf{a}$	Antenna position
b	Bias in pseudo-distance

ISAR

$\mathbf{r}$	Position vector
$\mathbf{d}$	Relative position vector
$\mathbf{x}$	Target trajectory
P	Probability density function

Stochastic estimation

$\mathbf{0}$	Zero-vector
I	Identity matrix
$\mathbb{E}$	Expectation operator
$\mathcal{N}$	Normal distribution
n	State dimension
λ	Sigma-point scaling parameter
β	State distribution parameter
α	Sample distribution parameter
κ	Secondary parameter
W	Sigma-point weight
f	Process model
h	Observation model

$\mathbf{w}$	Process noise
$\mathbf{v}$	Measurement noise
$\mathbf{y}$	Measurement vector
$\hat{\mathbf{y}}$	Predicted measurement vector
$\mathbf{x}$	State vector
$\hat{\mathbf{x}}$	State estimate
$\mathbf{P}$	State covariance matrix
$\mathbf{Q}$	Process noise covariance matrix
$\mathbf{R}$	Measurement noise covariance matrix
$\mathbf{K}$	Kalman gain
$\boldsymbol{\chi}$	Matrix of sigma-points as column vectors

List of Abbreviations

ABI	Application Binary Interface
ABNF	Augmented Backus–Naur form
AC	Access control
AJAX	Asynchronous JavaScript and XML
AOA	Angle of arrival
API	Application Programming Interface
ASLR	Address space layout randomization
COM	Communication port
COTS	Commercial off-the-shelf
CPU	Central Processing Unit
CSS	Cascading Style Sheets
CWNA	Continuous white noise acceleration
DRM	Dense Reader Mode
EAN	Electronic Article Number
EKF	Extended Kalman Filter
EMI	Electromagnetic interference
EPC	Electronic product code
EPCIS	EPC Information System
e.r.p.	Effective Radiated Power
ERP	Enterprise Resource Planning
ETSI	European Telecommunications Standards Institute
FCC	Federal Communications Commission
FPGA	Field Programmable Gate Array
FOV	Field of View
FTP	File Transfer Protocol
Gen2	EPCglobal Class 1 Generation 2 specification
GPI	General purpose input
GPIO	General purpose input/output
GPS	Global Positioning System
GSM	Global System for Mobile communications
GUI	Graphical user interface
HF	High-frequency
HID	Human interface device
HMI	Human-machine interface
HTML	Hypertext Markup Language
HTTP	Hypertext Transfer Protocol
HTTPS	HTTP over TLS
I/O	Input-Output
IEC	International Electrotechnical Commission
IEEE	Institute of Electrical and Electronics Engineers
ILBAC	Indoor-location based access control
IP	Internet Protocol
IPS	Indoor positioning system

ISAR	Inverse synthetic-aperture radar
ISM	Industrial Scientific Medical
ISO	International Organization for Standardization
IT	Information technology
JIT	Just-in-time
KF	Kalman Filter
LBS	Location based service
LBT	Listen Before Talk
LF	Low frequency
LLRP	Low-level reader protocol
LOC	Lines of code
LOS	Line-of-sight
LRU	Least-recently used
LTKC	LLRP Toolkit C library
MISAS	Multilateral inverse synthetic aperture secondary radar
NFC	Near-field communication
MAC	Media access control
MCU	Microcontroller unit
MES	Manufacturing Execution System
MTB	Message/Transport Binding
MVVM	Model-view-viewmodel
NLOS	Non-line-of-sight
OS	Operating system
PCB	Printed circuit board
pdf	probability density function
PDOA	Phase difference of arrival
PoE	Power over Ethernet
POSIX	Portable Operating System Interface for uniX
QA	Quality assurance
QUPER	Quality performance model
RAII	Resource acquisition is initialization
RAM	Random Access Memory
RF	Radio frequency
RFID	Radio-frequency identification
RMSE	Root mean squared error
RP	Reader Protocol
RPC	Remote procedure call
RSSI	Received signal strength indicator
RTLS	Real-time locating system
SAR	Synthetic aperture radar
SMA	SubMiniature version A
SNR	Signal-to-Noise Ratio
SSE	Server-Sent Events
TCP	Transport Control Protocol
TDOA	Time difference of arrival

TLS	Transport Layer Security
TOA	Time of arrival
TOF	Time of flight
UHF	Ultra-high frequency
UKF	Unscented Kalman Filter
ULA	Uniform linear array
UML	Unified Modeling Language
URI	Uniform resource identifier
USB	Universal Serial Bus
UT	Unscented Transform
UWB	Ultra-wide band
VM	Virtual Machine
WiFi	Wireless Fidelity
WLAN	Wireless local area network
WPAN	Wireless personal area network
WSN	Wireless sensor network
XML	Extensible Markup Language

1 Introduction

Two major topics are discussed in this thesis. The first of the two is about wireless, passive radio-frequency identification (RFID) based asset tracking solutions that employ low-level sensory information, such as received signal strength and phase information. Another topic of interest is about embedded RFID reader software application to process that information and generate reports based on that. In this introductory section, an overview of the research problem is presented and the scope of the thesis is defined, motivation for doing this research and software engineering is stated, and the way this thesis is organized to sections is elaborated.

1.1 Motivation

Industrial and logistics processes mandate supply-chain management, asset tracking, and often full traceability. Management of people and physical assets essentially build on the information about identity, events, time and location; how can we know where is the unit of interest currently, what is it, where has it come from, where is it going, or when? Typical examples include tracking items or tools inside a workshop, packages and containers on a conveyor system inside a warehouse, as well as people entering and leaving facilities.

Automating material-handling, identification, and tracking of assets, as well as enabling integration with automation systems are used to bring down logistics costs that occur with manual labor. Optical solutions such as barcodes, and wireless radio protocols like RFID are used for automatic identification of labels or tags attached to assets. Electronic transponders, or *tags*, have similar features as barcodes, and they may be used in parallel with or as an alternative solution to barcodes in many applications. RFID does have some unique use cases though, as the technology does not require optical line of sight, data written to the tag can be altered multiple times, and assets with a life-cycle measured in years can be tagged and identified with the same tag [1]. Covering the whole lifetime of products is necessary in fields that require full traceability like in pharmaceutical or food industries [2, 3].

More recent use cases for RFID originating from experiences in the field include tracking human-wearable accessories and portable devices. In addition to physical access control (AC) solutions that have been commonly implemented with smart door locks and key tokens based on inductive coupling or near-field communications (NFC), a need for more unobtrusive and transparent approaches from the viewpoint of the user are sometimes preferred. Examples of such an application include indoor-location based access control (ILBAC) for authorizing a user to operate machinery using portable human-machine interface (HMI) device such as a tablet computer only within line of sight and in the immediate vicinity of the operated equipment, or for monitoring and mandating people to wear safety equipment when entering an industrial area or a construction site.

Such applications present more challenges to the identification and tracking process than in traditional use cases, where the targets are either stationary or move in a controlled manner, and where it is usually sufficient to perform an inventory

round to observe the presence of assets currently in the field of view (FOV) of the reader device. Coarse location information is always available in granularity of the area covered by a single read point. However, when there are several transponders in the interrogation zone simultaneously, and either their direction of movement or relative position or order has to be known, the currently employed methods are insufficient.

1.2 Research problem

The objectives of this thesis can be summarized in two steps. First objective is to review some of the previous work done in radiolocating methods that could be implemented in an RFID reader operating in the ultra-high frequency band (UHF), 865–868 MHz specifically. The second goal is to come up with an implementation of an on-board embedded software application for a commercial off-the-shelf (COTS) reader device that is capable of operating in advanced RFID applications requiring real-time information on movements of passive RFID tags.

Ideally, the reader software shall be capable of determining the spatial location and the movement vector of a passive transponder in the interrogation zone with accuracy in the decimeter scale, as demonstrated in [4]. Localization methods based on propagation of electromagnetic waves are implemented either in terms of distance measurement (*ranging*, or *lateration*) or perceived angles of arriving signals (*angulation*). All practical localization methods presented in the literature that do not require custom RF circuit design, resort to using the received signal strength indicator (RSSI), signal phase angle rotation, or the Doppler frequency shift as inputs. These are all quantities available in the reader device for which the software is developed. Thus, the scope of this thesis is limited to methods that can be used to make spatial estimations from these variables.

Another aspect of this research is the discipline of embedded software engineering. The targeted device is an industrial Impinj Speedway Revolution R420 UHF RFID reader running an embedded distribution of the Linux operating system on ARMv5 processor, with a maximum of 16 megabytes of RAM and 8 megabytes of Flash storage space available for the custom application.

Currently, there exists only few pieces of commercially available RFID reader control software outside the case-by-case tailored solutions provided by RFID middleware vendors. Even these pieces of software are mostly available only for authorized dealers of reader hardware and middleware vendors themselves, and it is assumed that providing such products separately is not in their business interests. Therefore, the demand for creating a competing RFID reader software product stems from the current low availability of feature-wise satisfying applications, vendor lock-ins in using any available software, as well as from the added value of actually owning the source code and intellectual property.

Discernible research problems are then to determine which methods are feasible for locating passive UHF RFID tags, what are their limits and advantages, as well as the possible assumptions made in their application, and secondly, how to design and implement an embedded software for a resource-constrained embedded platform,

suitable for use in a real-time locating system (RTLS) as introduced above.

The first problem may be further refined to encompass finding the computational and real-time requirements imposed by any of the methods, as the resources available on the targeted device are severely limited.

Several sub-problems also arise from the set of software requirements succinctly proposed here: For the software to be of practical value, it must be configurable by the user for various use cases. For example in some cases a customer's needs are met by inventorying a shelf every few minutes, and sometimes multiple read points and real-time events and movement information are required. Remote procedure calls (RPC) for configuration and notification channels for reporting necessitate support for networking protocols. Still, the reader should be easily deployable in the field, and most frequent use cases should be simple for the user to apply. This design philosophy could be summarized as famously put to words by Alan Kay; *simple things should be simple and complex things should be possible.* Lastly, a notable sub-problem following any system with network connectivity is making the system secure and keeping it patched against unintentional and – to a degree – intentional misuse.

1.3 Structure of the thesis

The rest of this thesis is organized as follows. In Section 2, a glimpse into real-time locating systems and radiolocating is provided. In addition to radio-frequency identification technology which is in the main focus of this thesis, three other wireless technologies, namely ZigBee, Bluetooth, and WiFi are acknowledged. An in-depth look at the passive UHF RFID technology and its technical details is then given.

A selection of locating methods applicable for the purposes of this thesis are discussed in Section 3. Methods based on received signal strength and Doppler frequency shift are covered, but the focus is on phase-based methods as they provide opportunity for better accuracy over the common uses of RSSI.

In Section 4, an original contribution is made in the form of documentation of the major aspects of the embedded software development process. The context for the demand for such a product is suggested, an overview and details of the embedded target are disclosed, the main requirements and governing specifications are elaborated and selection of major parts in the software design and implementation is laid out. Finally an overview of the final software product is presented.

The performance of the system is considered in two experimental setups, covered in Section 5. The first experiment simulates a typical scenario with two antennas and a tag with a known trajectory, which is often the case in e.g. conveyor systems. The goal is to observe the estimated 2D trajectory and compare it against the known path. Likewise, the second case is concerned with estimating the trajectory of a tag, but with a more complex configuration of four reader antennas and in three spatial dimensions. For the purposes of the experiments, the trajectory of the tag is constrained to linear motion with constant velocity.

The results are presented and discussed in Sections 6 and 7, respectively. Final conclusions about the work presented in this thesis are drawn in Section 8.

2 Background

In this section the basic concepts of real-time locating systems are covered, along with the most commonly studied radio communication technologies applicable for such purposes. The core matter of the section is UHF RFID technology, which will be presented in more detail.

2.1 Real-time locating systems

Real-time locating systems are wireless systems where an item can be located anywhere in some defined space in real-time or close to real-time, using measurements made from the physical properties of the radio link, as defined in ISO/IEC 24730 standard concerning RTLS [5]. The conceptual classifications of RTLS consists of four cases, determined by the area and accuracy of the system, ranging from satellite positioning from 10 meter accuracy to more confined spaces with accuracies down to three meters or tens of centimeters. Although to be compliant with the standard a system must implement at least one of its 2.4 GHz or UWB air interface protocols – none of which will be carried out here – the definitions do give a useful distinction between plain RFID and RTLS. Particularly, methods for locating an asset by a hand-held device, or by mere virtue of the fact that asset passes some point at a certain time, are better classified as RFID than RTLS.

Real-time locating systems have many applications. In addition to asset management, location information is valuable in many monitoring and notification systems, location-based services (LBS), as well as data mining and optimization. Examples could include minimizing the time it takes to locate an item or to provide routing information in a warehouse, monitoring occupancy levels or capacities for safety or security purposes, notifying on misplacements or unauthorized exits of people or assets, as well as providing users targeted information, advertisements and services based on their location.

A high-level illustration of RTLS infrastructure is represented in Figure 1. The system is comprised minimally of a location engine, fixed infrastructure of sensors or beacons, as well as mobile tags. The tags communicate with the rest of the network over an air interface, i.e. a radio protocol. Transmitting and receiving radio signals requires RF circuitry and antennas on both ends of the link. The tags can be active

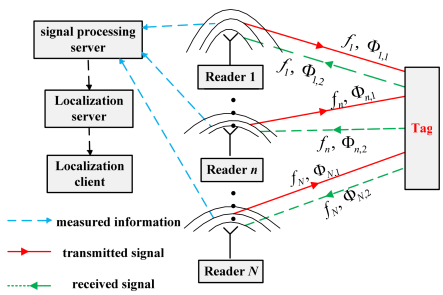


Figure 1: Real-time locating system diagram [6]

in the sense that they have their own transmitter and are able to probe frequencies for receivers, like mobile phones with Bluetooth support, or they can be passive and activate only when receiving power from carrier wave after which they backscatter their memory contents, like Gen2-type UHF RFID transponders do. Active tags may determine their own location by communicating with fixed beacons with known locations. Similarly, a portable RFID reader mounted e.g. on a forklift can be used to determine the location of the vehicle given high enough concentration of passive tags distributed in the environment with known locations. Determining the position in either end is possible with these so called *monostatic* configurations where the transmitting and receiving antennas are the same, but some methods use *bistatic* antenna configurations with different antennas used for transmission and reception, and are thus usually better suited for locating mobile tags with respect to fixed base stations [7].

Controlling the propagation of electromagnetic waves is not trivial. It is well-known that a GPS signal is mainly available outdoors with a line of sight to the satellites in the orbit, and non-line-of-sight (NLOS) conditions give rise to the need for indoor positioning systems (IPS). Several other factors stem from the laws of physics governing radio wave propagation that affect the measurements and pose challenges to RTLS design. These environmental factors include attenuation, reflection, diffraction, refraction and absorption, or more generally, multipath propagation [8]. Real-world signals fade, bounce, get blocked, and get absorbed. Many noise sources exists too, internally in the circuits and in the form of external radio signal emissions and electromagnetic interference (EMI).

The transponders are implicitly identified by some unique identifier, such as a MAC address of the network interface, globally unique identifier code, an electronic product code (EPC), or similar. The location is determined from measurements made from signal properties, usually by measuring round-trip time or time-of-flight (TOF), time difference of arrival (TDOA), angle of arrival (AOA), phase difference of arrival (PDOA), or received signal strength (RSSI) [9]. One-way propagation time, or time of arrival, (TOA) could also be used to measure distances to multiple sensors, but that would require precise synchronization of all transmitters and receivers, and is often infeasible.

The accuracy of many IPS solutions with *active transponders* presented in the literature lies in the range of 1–5 meters for WLAN, Bluetooth, RFID and GSM, and under one meter for UWB. Response times range from 0.1 seconds (real-time) to 30 seconds. [9]

2.2 Wireless technologies

The availability of inexpensive hardware as well as the pervasiveness of certain radio protocols that has followed make a good fit for the technologies to be used for locating systems, even if not their original purpose. RFID, as well as a few other common wireless technologies that have been of interest in the RTLS research – namely Bluetooth, ZigBee, and Wifi – are introduced and shortly discussed below in terms of their applicability for RTLS. Other technologies that have have been studied (like

UWB, for example) but which are not comparable e.g. in market penetration are excluded for brevity.

2.2.1 Radio-frequency identification

RFID is a general term for identifying, reading from and writing to integrated circuits via radio communications in the interrogation zone of an RFID reader. RFID has its roots in military radar technology and aircraft transponders, but has since been used increasingly for industrial, logistics, retail, access control, and mobile payment applications. [10]

A typical RFID system consists of a population of tags, one or more interrogator devices, and a background IT system consisting of a server, a database, and a network [11]. Reader devices may have integrated antennas or external ports with coaxial connectors, cables and antennas of varying designs. Readers – despite their name – can also be used to commission new tags and to write custom data to tag memory banks.

RFID tags are generally categorized into three groups by their use of a power source: *Active* tags have their own battery and a transmitter that enables independent transmissions and long ranges. *Passive* tags do not have their own power source nor transmitter per se, but what they do instead is that they accumulate energy from the RF carrier wave sent by the reader and scatter back their responses by modulating the wave. *Semi-passive* tags have their own battery but no active transmitters. Instead, semi-passive tags use *battery-assisted backscatter* which means a similar mechanism to that of the passive tags, but with a longer range due to the additional power.

The properties of an RFID system also vary according to the frequency used. As the signal wavelength depends on the frequency, the method of coupling between interrogators and a transponder depends also on the operating frequencies. For HF and LF systems with long wavelengths, the coupling is usually inductive, the communication happens in the near-field of the antennas, and the radiators are often constructed as magnetic loops. On the other hand, systems operating in UHF and microwave bands have wavelengths in the centimeter scale, and with them the communication takes place in the radiative far-field, where electromagnetic plane waves have been fully formed [10, p. 112]. The method of coupling, along with the type of tag power source, are the main factors affecting the read range of a system. Typical characteristics of RFID systems are presented in Table 1. RFID systems operating at different frequencies are not all applicable to every case because of the restrictions in their reading range, data transfer rate or noise properties, or because there exists liquids or metals in the operating environment [12, 13].

RFID air interface protocols are governed by ISO/IEC 18000 family of standards, among others. The ISO 18000-6C is of special interest, because it specifies the air interface for RFID applications in the 865–868 MHz ISM band, and also because it was previously known as the EPCGlobal Class 1 Generation 2 specification that had gained wide adoption [15, 16]. ISM frequencies are internationally reserved for *industrial, scientific, and medical* uses without special licenses required. Still, local radio regulations may apply, as later discussed in Section 2.3.

Table 1: Properties of RFID frequency bands [14, p. 41]

Band	LF	HF	UHF	Microwave
Frequencies	30–300 kHz	3–30 MHz	0.3–3 GHz	2–30 GHz
Coupling	Near-field	Near-field	Far-field	Far-field
RFID band	125–134 kHz	13.56 MHz	433 MHz, 865–956 MHz	2.45 GHz
Read range	< 0.5 m	< 1.5 m	< 100 m, < 5 m	< 10 m
Data rate	~ 1 kbps	~ 100 kbps	640 kbps	~ 100 kbps
Applications	Smart tokens, identification of animals	Access control and security	Logistics	Highway tolls

Various studies have been conducted in the last decade on the applicability of RFID for local positioning – invariably using only UHF frequencies of 433 MHz or 865 MHz, and above – see e.g. [17–26] for references. Typically three categories of techniques exist for RFID localization; ranging, scene analysis (also known as RSSI fingerprinting), and proximity.

Distance estimation is often based on measurements made from the direction of arrival of the signal, time parameters, received signal strength, or phase difference between transmitted and received signals, as mentioned in Section 2.1. They can provide accurate estimates under LOS conditions, but often require special hardware.

2.2.2 ZigBee

ZigBee is a radio communication protocol by the ZigBee Alliance, intended for low-cost and low-rate wireless personal area networks (WPAN) and wireless sensor networks (WSN). The protocol is based on IEEE 802.15.4 international standard [27]. It operates on the same unlicensed ISM frequencies as RFID.

Wireless sensor network comprises a network of cooperative devices, nodes that sense and control the environment, and enable interactions between users, computers and the surrounding environment [28]. WPANs, and ZigBee WSNs, are implemented with either of two topologies (Figure 2); a star topology with a central communication node or mesh networks with peer-to-peer topology [27, p. 46].

WSNs and ZigBee have applications in environmental monitoring, healthcare, positioning and tracking, as well as logistics. Almost all of the applications can be classified into two categories; event detection and spatial process estimation [28].

ZigBee is designed for low-power, battery-operated use to enable unattended wireless operation for minimal power and wiring costs, and maximal flexibility; instead of being always on, ZigBee "sleeps" most of the time and in many cases has years of battery life [29]. ZigBee transceivers are widely available as printed circuit

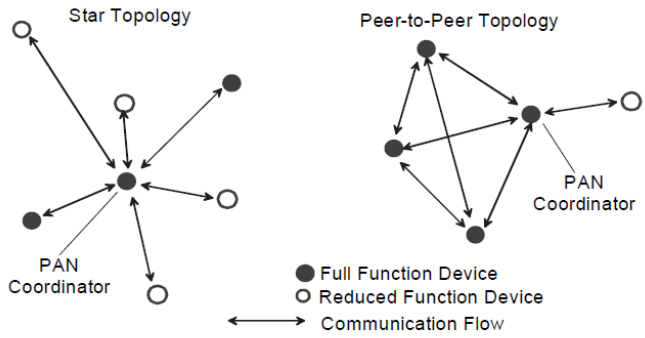


Figure 2: Star and peer-to-peer topology examples [27, p. 46]

board modules with integrated antennas, and as single-chip microcontroller solutions from large MCU manufacturers like Microchip. ZigBee has typical signal range of 10-100 meters, and under special conditions even up to 400 meters, and has a data rate of 20–250 kbps [29].

The applicability of using ZigBee for location estimation has been tested in various studies, most of which opt to use the traditional approach of RSSI ranging. Some examples of time-difference of arrival and angle-of-arrival methods exists, but they require special arrangements like precise synchronization or rotating antennas, which is to be expected. For the reason that ZigBee is constituted of nodes forming a mesh network, it can be built to tolerate communication failures and reliably re-route messages.

2.2.3 Bluetooth

Bluetooth is a short-range radio communication technology designed to replace serial RS-232 cables in personal area networks. Bluetooth is standardized as IEEE 802.15.1, but the future development is left in the hands of the Bluetooth Special Interest Group. Bluetooth operates in the 2.4 GHz ISM frequency band, has a range of up to 100 meters, and a bit rate in the order of 1 Mbps [29, p. 23].

Bluetooth has many applications in consumer electronics, such as handsfree headsets and wireless speakers, computer mice and keyboards. It can also be found in places that used to adopt infrared communications. The technology has actually become ubiquitous together with mobile phones, every one of which now has the support for Bluetooth.

To provide an answer for the growth of personal electronic devices and the Internet of Things, a low-energy version of Bluetooth (BLE) has been adopted into the standard. Available hardware solutions make Bluetooth relatively ubiquitous technology, as it is included in practically all modern mobile devices. Also, the rise of battery-powered Bluetooth beacons may foreshadow the increase in the number of location-based services available in the future.

Studies concerning the use of Bluetooth as a localization technology are discussed e.g. in [9]. Using distance estimates based on the strength of received signals,

localization accuracy up to 2–4 meters can be achieved, in some implementations with a considerable delay.

2.2.4 WiFi

WiFi is a wireless local area network technology based on IEEE 802.11 standards, operating in the 2,4 GHz ISM frequecies. IEEE 802.11 is currently the dominating wireless network standard, which makes it an attractive target for indoor localization [9].

WiFi is a familiar technology to many, used for connecting wirelessly to the internet with portable devices. WLAN connectivity and public hotspots are common things to encounter in urban areas, and location estimation with WiFi opens up the possibility for tracking different types of assets indoors.

WiFi compatible hardware consists of network bridges, routers, adapters, access points and endpoint devices. The technology is also provided for embedding it in the form of modules and development boards, like the ESP8266 and its successor ESP32 that have gained huge popularity in powering low-energy internet-enabled embedded devices.

The applicability of WiFi for locating systems has been studied perhaps more extensively than the alternative ones. Artificial neural networks, k-nearest neighbors algorithm and other classifiers have been proposed together with probabilistic methods, as well as the usual RSSI-based distance estimation. [9]

2.3 UHF RFID

In this section, a more in-depth look at UHF RFID technology is given in the context specific to EPCglobal Class 1 Generation 2 compliant [15] mechanisms. The term Gen2 will be used below meaning the same thing. Especially, hardware and air interface aspects are considered, together with the implications they might have regarding the assumptions made in this thesis.

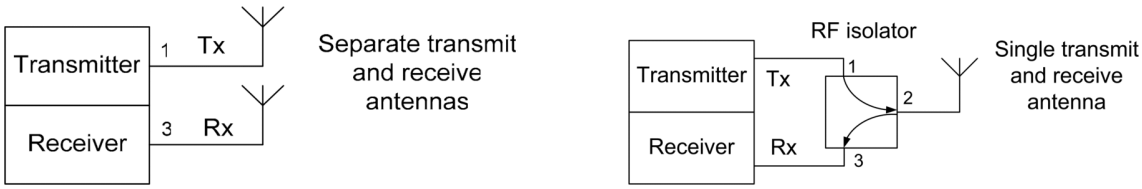
EPCglobal aims to establish standards and adoption for the Electronic Product Code (EPC), which is written to memory banks of Gen2 tags and is used to identify them. EPCglobal is part of GS1, an organization providing standards for EAN barcodes. EPC is designed to be a form of a universally unique identifier for every physical object, it is defined in the open EPCglobal Tag Data Standard [30], and is compatible with the widely used EAN barcodes, as they can be mapped to GS1 identifiers. The adoption of Gen2 as the dominating UHF RFID standard, followed by the ratification as ISO 18000-6C standard, led to world-wide growth in availability and lower prices for UHF RFID technology. EPCglobal also holds several standards on EPC information systems.

Gen2 RFID applications are typically used in supply-chain management and warehouse control systems. Its advantages are a read range of several meters, high throughput, and low cost – as already seen in Section 2.2.1. RFID middleware systems are often integrated with larger backends, like ERP or MES.

2.3.1 Hardware

Several kinds of RFID reader devices exists. They vary in their purpose, form and size, as well as features; hand-held readers for manual inventory tasks, standalone readers with processing power and network connectivity, and reader modules to be used as an embedded component, for example. The primary properties that are affected by different use cases - in addition to the air interface protocol and regulatory region – are choice between integrated antennas and the number of external antenna ports, maximum transmit power and receiver sensitivity level, physical dimensions, input/output interfaces, networking, and price.

Design choice that affects isolation and signal-to-noise ratio (SNR) in transmit and receive channels of the reader, as well as radiolocating methods, is the antenna configuration (Figure 3). Most available COTS devices have monostatic RF frontend, but custom bistatic configurations appear in the literature on local positioning systems.



(a) Bistatic (b) Monostatic

Figure 3: Reader antenna configurations [31]

Antenna selection is important; the edge of the interrogation zone in the UHF frequencies is not sharp, especially in multipath environments. The geometry of an antenna determines the directivity properties and together with the feed point they affect the form of the antenna lobe, i.e. the radiation pattern. There are a wide spectrum of antenna designs, most common ones probably being patch, (meandering) dipole, and planar PCB antennas like the inverted-F model. Different types of antennas can be used with RFID readers.

Antennas can be either linearly polarized, or circularly polarized in either direction. The polarization of the tag must match the reader antenna polarization to maximize transferred power and read range. Polarization mismatch loss due to misaligned antennas is $20\log(\cos\theta)$ dB, where θ is the angle between the tag and the reader antenna [32]. Antennas in Gen2 transponders are mostly based on the dipole with linear polarization. Due to non-ideal antennas even the polarizations are not ideal, and antennas may radiate with polarizations in more than one direction.

Gen2 tags come in many shapes and sizes. A tag has an antenna (the most prominently visible element) and a circuit chip. All Gen2 tags are passive tags and send their response to interrogators by backscattering modulated signal. Figure 4 represents an illustration of the power levels of the modulation sidebands of the transponder in a logarithmic scale.

There are four memory banks in a Gen2 compliant tag, as seen in Figure 5, and they may comprise zero or memory words. The contents of the *EPC* memory

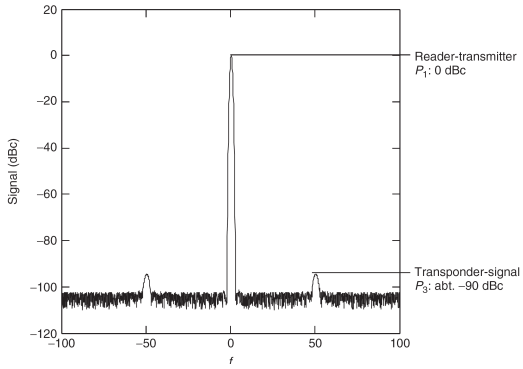


Figure 4: Tag response sideband power level [10, p. 142]

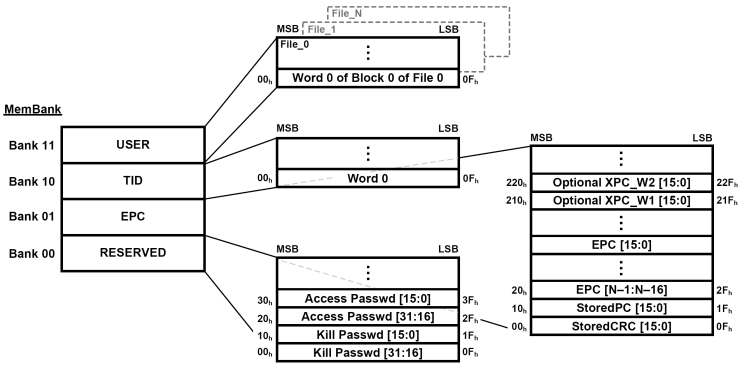


Figure 5: Logical memory map of a Gen2-compliant tag [15, p. 44]

bank include the code used to identify the tag, which is usually an EPC identifier encoded following the GS1 specifications. The *TID* memory bank contains ”sufficient identifying information” about the tag model and commands or features the tag supports [15, p. 44], is written by the tag manufacturer and in general is not writable. The contents of the *EPC* and *User* banks can be modified, providing the correct password is known if implemented and the tag is not locked or killed. During a Gen2 protocol inventory round, the tags respond with the contents of their *EPC* memory bank. Additional *access commands* can be sent to the tag to read the *TID* or *User* contents.

The throughput for inventorying tag populations obviously depends on the number of additional commands and amount of data transferred. A high read rate is generally favorable in RTLS applications, but there exists a special case in using the Doppler frequency shift signal parameter, where the the accuracy increases by slowing down the data transfer – even by using slower, dense-reader modes (DRM) that use more symbols for encoding the binary data [33, 34].

2.3.2 Air interface

The approach in this thesis is focused and limited to the air interface in 865–868 MHz frequency range in the ETSI regulatory region. ETSI EN 302 208 is a specification

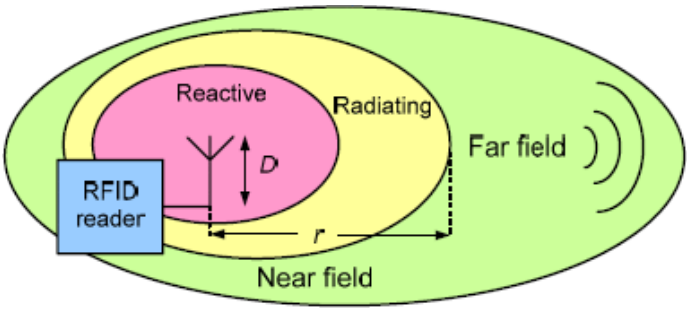


Figure 6: Antenna near and far field regions [36]

that covers the minimum characteristics considered necessary to make the best use of the said frequencies [35]. It governs the use of RFID devices using up to 2 Watts of *effective radiated power* (e.r.p.) in the European Union. It should be noted though, that this limitation of scope is done for practical reasons and without the loss of generality – in fact, many radiolocating methods would benefit having a wider bandwidth that is available e.g. in the 900 MHz band in the FCC governed region.

The interrogation zone, or the read field of an RFID reader is illustrated in Fig. 6, with the distinction to near and far fields. In addition to the transmit power and antenna gain, the factors affecting the actual interrogation zone include the losses happening in the cables and connectors, as well as tag sensitivity. The RF connectors, cables, and antennas are traditionally matched to have a real impedance of 50 Ω, but discontinuities and mismatches in transmission paths result in reflections and standing waves, which reduce the amount of power transferred to the propagation medium (air).

When the gain of a circularly polarized antenna with respect to an *isotropic radiator* is known, as well as the cable losses (documented in datasheets) and the transmit power, the greatest allowed conducted power can be computed according to the ETSI specifications [35, p. 34] by using the formula

$$P_C = P_{\text{erp}} - G_{IC} + 5,15 + C_L \quad \text{dBm}, \tag{1}$$

where P_{erp} is the effective radiated transmit power, G_{IC} antenna gain, and C_L total cable loss.

The primary aspect ruling the extents of the interrogation zone is limited by the downlink, i.e. the transmit power of the reader device, as the transponder necessitates a minimum electric field strength to be able to power up. Once a tag has accumulated enough energy from the carrier, it is able to retain its state for some time - the exact persistence time depends on the tag model, but also on the protocol parameters.

Four independent sessions are defined in the Gen2 protocol to be used in inventory rounds. The selection of protocol session, together with the inventoried status flag of the tag define how often the tag responds when energized as well as how long the state persists when not energized. Figure 7 depicts the tag reads and persistence in different sessions together with the reader inventory search mode selecting only tags in certain states.

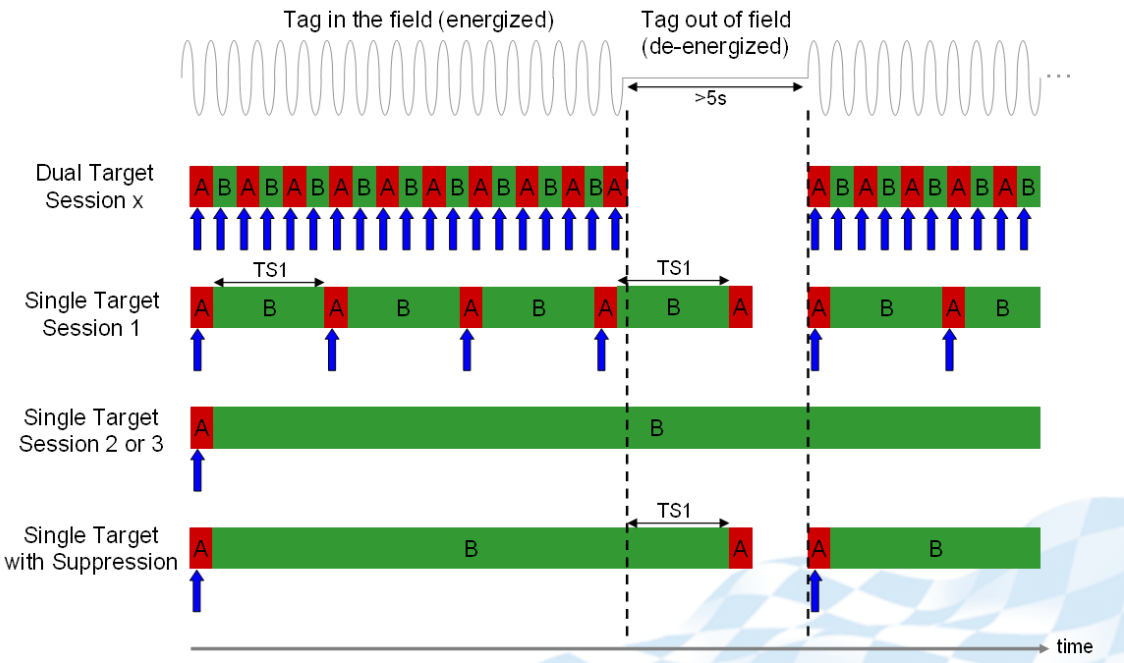


Figure 7: Sessions, persistence, and search modes when tag is in the field [37]

The inventory search mode that targets both tag states results in the highest possible read rate for a single tag, irrespective of the persistence parameters. This *dual target search mode* is then the most likely favorable configuration for locating tags in cases where high sampling is needed and the number of tags simultaneously in FOV is controlled – a high number of tags all responding at once results in a number of collisions that degrade the performance. The Gen2 protocol is *half-duplex*, meaning there are no simultaneous transmissions to both directions and only one party may transmit at a time. Tags are neither required to demodulate reader commands while backscattering [15, p. 22].

The reader can try to mitigate tag-to-reader collisions using random-slotted collision arbitration, where the interrogator commands a tag to load a Q-bit random counter to slot counter that is decremented on reader command and enables tag response when counter reaches zero [15, p. 17]. Gen2 separates readers' transmissions from the transmissions of the tags in the frequency spectrum so that tags collide only with tags and readers collide only with readers [17].

Readers operating near each other can also interfere with one another; to reduce reader-to-reader collisions, different channels and session may be selected, and readers may opt to use special dense-reader modes and signaling parameters. Low-duty cycle modes and triggered RF operations may also be used to limit the amount on interference caused to neighboring devices. *Listen Before Talk* procedure (LBT) that mandated listening the channel for transmissions before transmitting used to be required by ETSI EN 302 208-1, but the requirement was removed before version 1.2.1 was released.

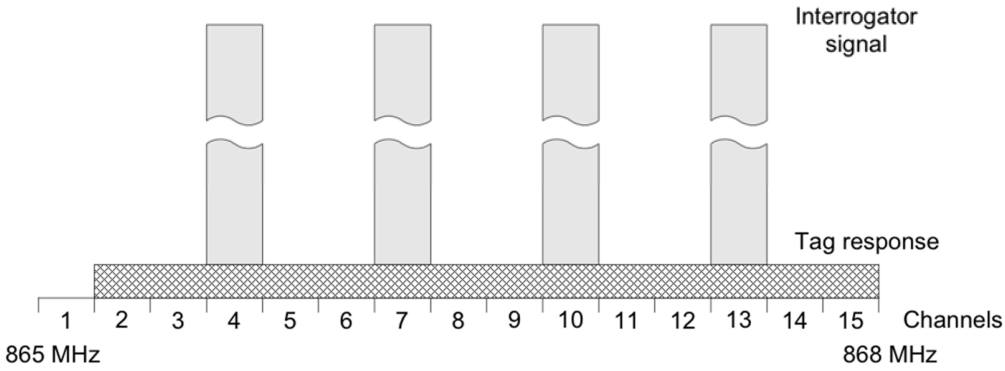


Figure 8: ETSI channel band for 865–868 MHz range [35, p. 13]

2.3.3 Performance considerations

Performance of an UHF RFID system is considered here as a function of the extent of the FOV of the reader, i.e. its read range, as well as the rate of operations a reader can apply on a tag. The desired parameters may vary depending on the application, but in general it is favorable to have as large a FOV and high read rate as possible – from there it is trivial to alter the RF protocol parameters and transmit power of the reader to achieve lower figures when needed.

The operations or transactions carried out by the reader include reading, writing, locking and killing tags. Reading a tag is the fastest operation as the interrogator only queries tags present in its field. Other operations take longer as they require multiple commands and tags changing their memory contents. Reading tags often results in a number of events where tags backscatter their EPCs several times. The rate of these events can rise up to as high as hundreds of read events per second. These events, their rate and other quality indicators can be processed in order to provide finer information to the middleware systems and business applications. If the read reports were to be delivered unfiltered, a substantial load could be imposed on the network and software systems. With filtering more abstract events can be generated, such as tag entering and leaving the field. High read rates are still associated with reliability as it gets less probable for a tag intended to be read to not get inventoried. A complementary objective is to maximize the rate of unique tags inventoried, which also increases the probability of a single tag to get inventoried, and is achieved by proper selection of air interface protocol parameters.

The performance of passive UHF RFID systems is limited by a number of factors, including tag characteristics, propagation environment, and reader parameters. Tag chip sensitivity, antenna gain and antenna polarization determine the minimum power required to power on the transponder, the direction of the maximum read range, and power transfer loss caused by the mismatch of signal polarization. [38]

Path loss describes the power density of the electromagnetic waves as function of distance, and is strongly affected by the environment; the reflection, refraction and diffraction of signals, or more generally, multipath propagation. Receiver sensitivity

of the reader is not in many cases the dominating factor affecting the read range, as the sensitivity is usually high enough to make the system limited by the tag characteristics instead. Radio antennas are tuned to have the best performance on some selected frequencies, and they perform poorly in other parts of the spectrum. The performance can be further degraded, when an antenna is detuned from its optimal frequency by the surroundings affecting the impedance and thus the ratio of power radiating from the antenna.

An aspect that may be sometimes overlooked is the expert knowledge in choosing the right antennas for specific applications, deciding their placement and orientation, and in general controlling the interrogation zone. High gain equals high directivity, and antennas with high gain have a narrow beam. The type of the tag and how it is attached to its carrier have an effect on how reader antennas are best positioned. Line of sight is more reliable than NLOS and should usually be preferred if possible [39,40]. The interrogation zone can also be divided into smaller sections with proper antenna placement where the presence or absence of a tag observed with some combination of antennas can be converted to information about the coarse location of a tag.

3 Location estimation

Methods for determining location using RFID technology can be in general classified into two groups according to the target being located; tags moving in the interrogation zone of a static reader, or a mobile reader estimating its position based on reference tags with known positions. Ways to compute location estimates from radio signals include ranging, angulation, scene analysis and proximity techniques. The methods can be further categorized by the signal parameters used to derive the information, return signal strength and phase angle difference between the transmitted and received signals being the most common ones. Timing parameters and instantaneous frequency shift may be exploited as well.

The discussion in this section covers some of the radiolocating methods applied together with UHF RFID technology in research literature in recent years. A traditional approach is first taken here by introducing the methods together with each of the signal parameters available in COTS reader devices. A more detailed look is then given into two promising techniques. As the hardware sets limits for making estimations based on RSSI, PDOA, and Doppler frequency shift, all methods using time difference of arrival or time-of-flight parameters are out of the scope of this thesis.

3.1 RSSI-based methods

The relationship between wavelength λ and frequency f is $v = f\lambda$, where v is the speed of light c in free space. With a known frequency, the equation for the received power using the famous *Friis transmission formula* [41] and the area A of an effective antenna aperture [10, p. 122] can be derived as

$$\frac{P_\mathrm{r}}{P_\mathrm{t}} = \frac{A_\mathrm{r} A_\mathrm{t}}{r^2 \lambda^2} \tag{2}$$

$$A = \frac{\lambda^2}{4\pi} G \tag{3}$$

$$P_\mathrm{r} = G_\mathrm{r} G_\mathrm{t} \left(\frac{\lambda}{4\pi r} \right)^2 P_\mathrm{t}. \tag{4}$$

The symbols used in Equations 2–4 are as follows. P_t and P_r are powers for transmission and reception, λ is the wavelength, r corresponds to distance, and transmitting and receiving antennas have gains G_t and G_r. The subscripts r and t correspond to the quantities of the receiveing and transmitting antennas, respectively. Unlike in Friis' original notation, a radius r is used in Equation 2 to denote the distance, because it is illustrative to think of a radio wave propagating from its source as a spherical wave front (cf. Figure 6). Equation 2 only holds in the far field, where $r \geq 2a^2/\lambda$ and a the largest of the linear dimensions of either antenna [41]. A lossless signal path in free space is also assumed. Equation 4 is used to derive the *free-space link budget* in generalized bistatic form using the reflection coefficient of

the transponder [25], and is given as

$$\frac{P_\mathrm{r}}{P_\mathrm{t}} = G_\mathrm{t} G_\mathrm{r} (g_\mathrm{t} \Gamma g_\mathrm{r}) \left(\frac{\lambda}{4\pi r} \right)^4, \tag{5}$$

where g_t, g_t and Γ are the two gains and reflection coefficient of the transponder, respectively.

To compute the distance between a receiver and a transmitter, a logarithmic path-loss model (Eq. 6) has been proposed in the literature [17,20], where PL_0 equals the free-space link budget (Eq. 5) at reference distance r_0. Moreover, n is the loss exponent and X_σ a white noise term.

$$\mathrm{PL}(r) = \mathrm{PL}_0 + 10n \log \left(\frac{r}{r_0} \right) + X_\sigma. \tag{6}$$

Using that model, the value for received signal strength indicator (RSSI) can be obtained using the transmit power of the reader (P_t) as

$$\mathrm{RSSI}(r) = P_\mathrm{t} - \mathrm{PL}(r). \tag{7}$$

The equations presented above are primarily used to model a LOS environment. In many cases the only solution for modeling NLOS conditions and other artefacts is numerical simulation, but for short distances a closed-form solutions for multipath propagation can be presented. For a completely deterministic multipath channel the path gain can be presented [7] in the form

$$G_\mathrm{path} = \left(\frac{\lambda}{4\pi r_0} \right)^2 |H|^2, \quad \text{where} \tag{8}$$

$$H = 1 + \sum_{i=1}^{N} g_\mathrm{r}^i g_\mathrm{t}^i \Gamma_i \frac{r_0}{r_i} e^{-\mathrm{j}k(r_i - r_0)}. \tag{9}$$

Indices i in the latter equation refer to each of the individual reflecting objects in the propagation path. The normalized signal gain terms g^i and reflection coefficients Γ_i depend on the angle of incidence. For a monostatic reader in such an environment an equation

$$\mathrm{RSSI} = P_\mathrm{r} G_\mathrm{r}^2 G_\mathrm{path}^2 K \tag{10}$$

is given for the power of the tag signal received by the reader, with a backscatter gain K defining how much incident power is converted to the backscattered modulated power [7].

Ideally, RSSI varies monotonically as a function of distance given constant transmit power – in practice this is not the case and multipath propagation makes measurements unreliable. Nevertheless, RSSI is used in multilateration and scene analysis in attempt to make location estimates.

Multilateration, or ranging, is a family of methods used to determine location from a set of distance measurements. The most simple methods use averages of

known locations weighted by the received RSSI values [17, p. 211], or model the expected RSSI responses from tags and try to minimize some error function [20]. To account for imperfections in antenna radiation patterns, a standard least-squares approach can be used after the distances have been estimated [17, p. 212].

Scene analysis, or RSSI fingerprinting relies on offline measurements for creating probabilistic maps to which online readings can be compared. Several algorithms have been used for the classification task, including *k-nearest neighbors*, *Kalman filters*, and *artificial neural networks* [19, 21, 25, 42, 43].

3.2 PDOA-based methods

The next thing to consider is the relationship between the phase difference of arrival (PDOA) and the distance from the reader antenna to a tag. The following model is used for a modulated signal

$$s(t) = \alpha e^{-j\varphi(t)} + \epsilon(t), \tag{11}$$

where α is a damping factor, $\varphi(t)$ the phase of the signal at time t. The term $\epsilon(t)$ models additive zero-mean white noise.

The round-trip distance traveled by the signal can be expressed as a sum of a multiple of wavelength and a residual:

$$2d = n\lambda + \frac{\varphi}{2\pi}\lambda, \quad \varphi \in [0, 2\pi[, \tag{12}$$

from which the total accumulated phase can be solved as

$$n2\pi + \varphi = 2\pi\left(\frac{2d}{\lambda}\right). \tag{13}$$

In practice, the number of full wavelengths n is not known, and the only measurable quantity is the phase difference between transmitted and received signals. The measurement includes an unknown phase bias caused by the RF frontend, cables, and the transponder. The actual measured phase can thus be presented as function of distance according to the equation

$$\begin{aligned} \varphi &= 2\pi\left(\frac{2d}{\lambda}\right) + \varphi_{\text{offset}} \mod 2\pi \\ &= \frac{2\omega}{c}d + \varphi_{\text{offset}} \mod 2\pi. \end{aligned} \tag{14}$$

From the above equation it can be observed that the measured phase angle repeats its value every $\lambda/2$ distance travelled. Moreover, the angle and distance have a linear dependency, and using Equations 11 and 14 the observed signal can be given in the form

$$s(t) = \alpha_i e^{-j(2\omega/c)|\mathbf{p}(t)-\mathbf{a}_i|}, \tag{15}$$

in which $|\mathbf{p}(t) - \mathbf{a}_i|$ equals the distance from some point $\mathbf{p}$ and ith antenna $\mathbf{a}_i$.

Due to the ambiguity of the measured phase difference, several readings are always required for estimation, and in general, a moving target. An exception to the latter would be to use stepped frequency read-out, i.e. frequency-domain phase difference of arrival (FD-PDOA). Using phase information measured on two or more separate frequencies the distance can be estimated [7, 44, 45] as

$$|\mathbf{p}(t) - \mathbf{a}_i| = \frac{c}{4\pi} \cdot \frac{\varphi_1 - \varphi_2}{f_1 - f_2} + \epsilon = \frac{c}{2\pi\Delta f} \cdot \Delta\varphi + \epsilon. \tag{16}$$

For example, with $\Delta f = 1$ MHz the whole phase ambiguity cycle corresponds to a distance $c/2\Delta f \approx 150$ m [45], which in theory provides unique solutions to distance estimates in the read ranges of UHF RFID. Using custom RF hardware and low subcarrier frequencies the effect can be applied on a single baseband carrier frequency as is done in [6], and without the limitations incurring from the use of multiple measurements.

Methods using phase measurements obtained at multiple frequencies are limited by the assumptions they have to make. As expected, the tag is not allowed to move between consecutive measurements too much. On top of that, the tag phase offset and tag backscatter phase should not change with frequency or they should be calibrated out otherwise [7].

Leveraging the sensitivity threshold at which a tag is powered up as well as the stepped frequency read-out has been proposed in [46] to estimate dispersive properties of the reflection coefficient of the tag. The method requires adjustable transmit power and phase difference measurements.

Angulation, or the process of determining the angle of arrival (AOA) is an alternative technique to direct ranging. Distances to the target are computed using trigonometry and angles to two or more known locations of spatially distributed antennas. Arrays of antennas with inter-element distances less than half of wavelength are often used for AOA estimation. The bearing can then be calculated e.g. using spatial-domain of phase difference of arrival (SD-PDOA) [22, 24, 47], but the range of allowed angles will be limited by the distance of the antenna elements as well as the regularity of phase characteristics of the antennas in different directions [48].

A less complicated process for handling phase measurements but of great industrial value and thus of great interest, is a technique for relative localization of tags [49]. The method is used to determine the spatial ordering of tags based on the extreme values of unwrapped phase. An important application for that is e.g. identification and ordering of tags moving on a conveyor. In industrial settings the interrogation zones are not usually sharp or well-defined, and first-seen or last-seen timestamps cannot be trusted when imposing order on the set of observed tags.

Using bare phase values for absolute positioning is possible using methods that take into account a whole series of measurements to compute statistical estimates. Two such methods are later described in greater detail.

3.3 Doppler frequency shift based methods

The Doppler effect is a perceived shift in the frequency of the received signal caused by relative motion of the target. Assuming the relative velocity v between the reader

and a tag satisfies $v \ll c$, the Doppler frequency shift on the baseband signal is

$$f_D = \frac{2v}{\lambda} \cdot \cos\alpha, \tag{17}$$

where α is the angle between the tag velocity vector and the radial velocity vector pointing towards the reader [33].

If takes time ΔT for the reader to transmit a data packet, the phase angle rotation introduced by the Doppler effect over that time period is

$$\Delta\varphi = 2\pi \cdot (2 f_D \Delta T), \tag{18}$$

from which the Doppler shift experienced by the reader is obtained as

$$f_D = \frac{\Delta\varphi}{4\pi\Delta T}. \tag{19}$$

Time-domain phase difference of arrival (TD-PDOA) can be used to simulate the time differential of phase angle to compute an estimate of the radial velocity of a tag (Fig. 9). From Equation 17 the angle-dependent velocity is deduced using the Doppler frequency shift estimate as

$$v = \frac{f_D \lambda}{2\cos\alpha} \tag{20}$$

and is only valid for $\alpha \neq n\pi/2, n \in \mathbb{N}$.

Readers capable of measuring the Doppler frequency shift avoid the limitation that would follow the use of several phase measurements from multiple inventory rounds, such as maximum velocity for the target. At the same time, being able to accurately estimate small Doppler shift frequencies necessitates longer sampling times, i.e. slower data transfer rates and thus lower read rate.

Doppler frequency shift can be used for localization as demonstrated in [34]. In practice, the Doppler frequency shift readings provided by COTS UHF RFID readers tend to have noise levels that are too high to be meaningfully used as an input.

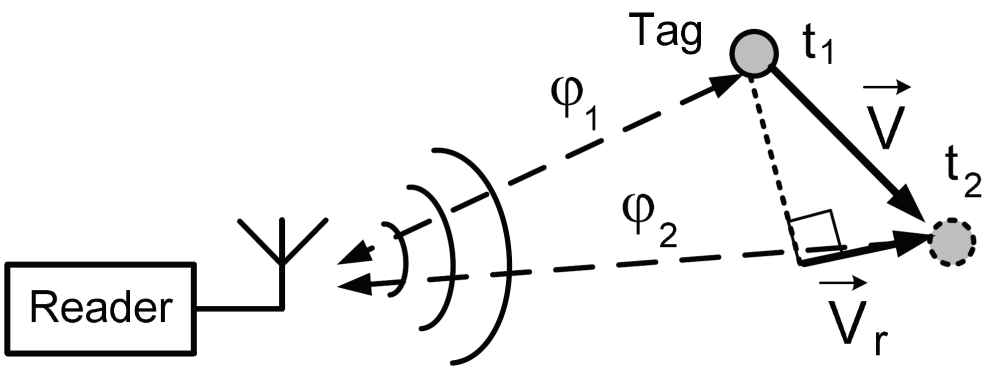


Figure 9: Radial velocity measurement [7]

3.4 Inverse Synthetic Aperture Radar

SAR and ISAR are synthetic aperture radar concepts used for generating radar images of targets by creating a virtual aperture that is larger than the physical antenna dimensions would dictate, by incorporating *a priori* information about movement trajectory. The secondary radar concept relies on cooperative reply signals from transponders – as opposed to primary radars known from military applications used to locate even uncooperative targets.

Instead of imaging stationary targets with a moving SAR platform, ISAR uses a stationary antenna to locate moving targets with known dynamics. These *multilateral inverse synthetic aperture secondary radars* (MISAS) have been recently applied for precise absolute localization of RFID tags. The ISAR methods presented in the literature [50–57] form the image using phase measurements. The methods are sometimes dubbed as *holographic* localization.

What is excellent about these approaches is that they do not require any modifications of tag, reader or antenna hardware where phase measurements are available. The localization can be done on a single narrowband frequency, and with stationary antennas. Compared to using RSSI measurements, phase-based holographic techniques are shown to have better accuracy, sensitivity, and robustness [55]. Even the requirement of knowing the trajectory can be relaxed with some assumptions [55]. Precise localization can be achieved in severe multipath environments [52, 53].

ISAR images are usually formed using computationally expensive grid search, and without further guidance or advancements in the field the method is unlikely to be feasible to be computed on embedded hardware. The method is applicable to problems with any number of dimensions, but the practicality is often limited by computational resources available. Due to the novelty of the principle and accuracy of the results, the holographic localization principle is still relevant for possible future production uses, and although being able to perform computations on the reader would be a nice-to-have feature, it is assumed not to be a decisive factor in the adoption of the method.

Expected typical application for the method in industrial context could be e.g. locating tags on conveyor system, and that aspect is in fact commonly simulated in research experiments using linear guidings [51, 54, 57]. Fixed antennas next to conveyors provide predictable environment where tag trajectories can be modeled with very high precision, and is the reason they make up for an attractive demonstration. The holographic method is applied in practice in Section 5.

Inverse synthetic aperture radar using phase measurements is described here in its simplest form as it is given in [54] to establish an ISAR solution for tracking of UHF RFID tags. The method can be formulated to take into account the additional phase-ambiguity of π radians introduced by the signal processing algorithms of an RFID reader.

Consider a case of RFID tags on a conveyor belt (Figure 10), where tags move with known dynamics i.e. constant velocity past the reader antenna with each on their own distance from the antenna. The trajectories are split into an unknown

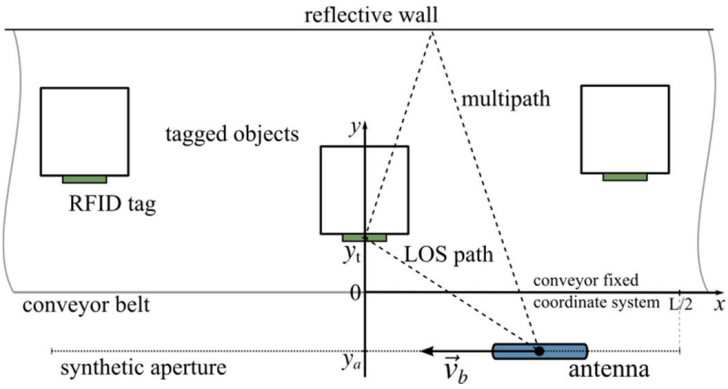


Figure 10: Tags localized with a static antenna and synthetic aperture [54]

starting position $\mathbf{r}_{\mathrm{t},1}$ and a known path $\mathbf{x}$ relative to the starting position

$$\mathbf{r}_{\mathrm{t}}(t) = \mathbf{r}_{\mathrm{t},1} + \mathbf{x}_{\mathrm{t}}(t). \tag{21}$$

Distances to tags at any point in time are computed from m time-dependent antenna-to-tag vectors

$$\mathbf{d}_{\mathrm{at},k}(t) = \mathbf{r}_{\mathrm{t},1} + \mathbf{x}_{\mathrm{t}}(t) - \mathbf{r}_{\mathrm{a},k} \tag{22}$$

where $\mathbf{r}_{\mathrm{a},k}$ is the fixed position of kth antenna. To represent expected phase values that would become measured for each vector $\mathbf{d}$, a hypothesis for the measurement is formulated as

$$\varphi_{\mathrm{hypo},k}(t) = \frac{4\pi}{\lambda}|\mathbf{d}_{\mathrm{at},k}(t)| + \varphi_{\mathrm{a},k}(\mathbf{d}_{\mathrm{at},k}(t)) + \varphi_{\mathrm{off},k} \mod 2\pi \tag{23}$$

with direction dependent phase characteristic $\varphi_{\mathrm{a},k}(\mathbf{d}_{\mathrm{at},k}(t))$ of kth antenna and offset introduced by the RF frontend. It is assumed that the tag follows the given trajectory – deviating from the trajectory would cause the ISAR image to de-focus resulting in inaccurate or useless location estimates. Discrete values are computed for the set of hypothesized phase measurements by assigning $\varphi_{\mathrm{hypo},k,i} = \varphi_{\mathrm{hypo},k}(t_{k,i_k})$ using timestamps t_{k,i_k} where, $i_k = 1, 2, \ldots, n_k$ and n_k is the number of phase measurements of the kth antenna.

Given the hypothetical phase values, a probability density function (pdf) is formulated to assign a probability for the hypothesis and the position vector in question to be correct. The pdf is derived by positing that the likelihood for a tag to be at the assumed position follows from summation of the signals so that they are superimosed constructively near the correct location [51]. For a system with calibrated phase offsets all the signals can be summed together [54], which results in a pdf

$$\tilde{P}_{\mathrm{c}}(\mathbf{r}_{\mathrm{t},1}) = \left| \sum_{k=1}^{m} \sum_{i_k=1}^{n_k} e^{j(\varphi_{\mathrm{hypo},k,i_k} - \varphi_{\mathrm{meas},k,i_k})} \right|. \tag{24}$$

The pdf for a non-coherent system with uncalibrated errors between the different antennas can be formed by interpreting the values as n_k virtual antenna elements in m separate, coherent sub-arrays of antennas, i.e.

$$\tilde{P}_{\mathrm{nc}}(\mathbf{r}_{\mathrm{t},1}) = \sum_{k=1}^{m} \left| \sum_{i_k=1}^{n_k} e^{j(\varphi_{\mathrm{hypo},k,i_k} - \varphi_{\mathrm{meas},k,i_k})} \right|. \tag{25}$$

The phase uncertainty introduced by the reader hardware is accounted by making the signal model have the complex angle wrap around every π radians, i.e.

$$\tilde{P}'_{\mathrm{nc}}(\mathbf{r}_{\mathrm{t},1}) = \sum_{k=1}^{m} \left| \sum_{i_k=1}^{n_k} e^{j2(\varphi_{\mathrm{hypo},k,i_k} - \varphi_{\mathrm{meas},k,i_k})} \right| \tag{26}$$

It is convenient to have the pdf scaled to a logarithmic scale and have it normalized so that the theoretically attainable maximum value equals 0 dB, and can be done as simply as

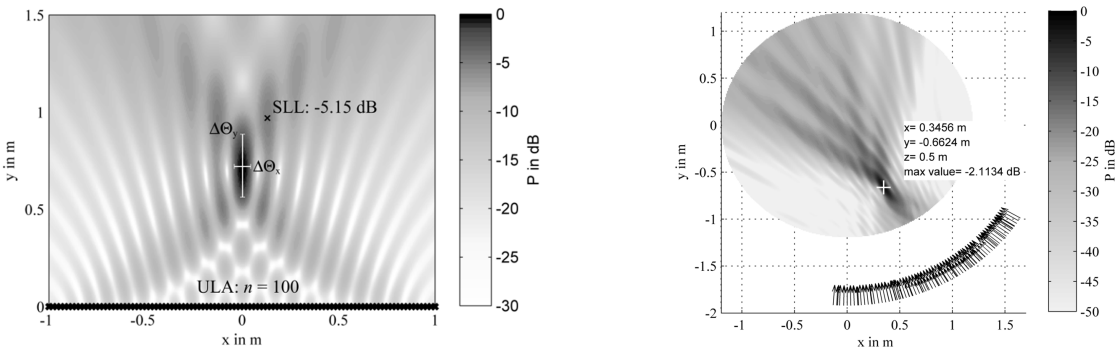
$$P(\mathbf{r}_{\mathrm{t},1}) = 20 \cdot \log_{10} \left(\frac{\tilde{P}_{\mathrm{nc}}(\mathbf{r}_{\mathrm{t},1})}{\sum_{k=1}^{m} n_k} \right). \tag{27}$$

Equation 27 differs slightly from the one given in [54] due to an error in the original notation. The starting point $\mathbf{r}_{\mathrm{t},1}$ can be found by maximizing the likelihood function;

$$\mathbf{r}^*_{\mathrm{t},1} = \arg\max_{\mathbf{r}_{\mathrm{t},1}} P(\mathbf{r}_{\mathrm{t},1}). \tag{28}$$

The position of the tag at any given time can then be computed using the known dynamics in Equation 21.

In the case with a conveyor belt and linear motion, evaluating the pdf over the spatial domain of the interrogation zone results in a 2D ISAR image like in Figure 11a, with the maximum value indicating the computed location of the tag. Similarly the image could be computed for a turntable with a circular tag motion, as is illustrated in Figure 11b.



(a) Uniform linear array (ULA)

(b) Uniform circular array

Figure 11: ISAR directivity patterns [54]

3.5 Random walk and nonlinear filtering

Bayesian filtering schemes have been applied for RFID localization, e.g. in [4, 58, 59]. Nonlinear estimation methods such as *Extended Kalman Filter* (EKF), *Unscented Kalman Filter* (UKF), and *particle filters* are examples of algorithms in the family of online, recursive methods for estimating unknown state variables from noisy measurements.

With a process model describing the dynamics of a moving target and a measurement model for predicting future measurements, these nonlinear filters can be applied to estimate the set of unknown state variables. A simple model for the dynamics of an RFID tag is introduced below. The model could be considered as a textbook example in the context of localization methods – a random walk, also known as *continuous white noise acceleration* and *Wiener velocity model*. In its essence, the model describes the movement as a *random walk*, with constantly jittering velocity affecting the position. The maximal accelerations experienced by the target are characterized by white noise and its covariance. A brief formulation of the model is given below, which is later used together with the UKF to locate RFID tags. The model reproduced following the formulation and ideas proposed in [59], but in this thesis UKF is applied instead of the EKF the researchers used.

3.5.1 Wiener velocity model

The distance r_i between the reader antenna and a tag can be computed from a series of signal phase measurements using equations 15 and 14;

$$r_i(t_n) = |\mathbf{p}(t_n) - \mathbf{a}_i| = -\frac{c}{2\omega}\arg(s_i(t_n)) + n_i\pi\frac{c}{\omega}, \tag{29}$$

where $\mathbf{p}(t_n)$ is the position of the tag at time t_n, $\mathbf{a}_i$ is the position of ith antenna, n_i an unknown multiplier for the number of wavelengths, and ω the angular frequency of the signal s_i from Equation 15. Assuming that the phase angle will not wrap around between two subsequent samples, the change in position can be recursively estimated with

$$\hat{r}_i(t_n) = \hat{r}_i(t_{n-1}) - \frac{c}{2\omega}\arg\left(\frac{s_i(t_n)}{s_i(t_{n-1})}\right). \tag{30}$$

In other words, the tag is allowed to move no more than a half of the wavelength between two inventory rounds. This sets a lower limit on the read rate and conversely an upper limit for the velocity of the tag.

Let us model the spatial dynamics of the tag using continuous white noise acceleration. A simple model only entails the position and velocity as the state variables. The model can be established in the form of a stochastic state-space model as follows:

$$\frac{\mathrm{d}}{\mathrm{d}t}\begin{bmatrix}\mathbf{p}(t)\\ \mathbf{v}(t)\end{bmatrix} = \begin{bmatrix}\mathbf{0} & \mathrm{I}\\ \mathbf{0} & \mathbf{0}\end{bmatrix}\begin{bmatrix}\mathbf{p}(t)\\ \mathbf{v}(t)\end{bmatrix} + \begin{bmatrix}\mathbf{0}\\ \mathrm{I}\end{bmatrix}\mathbf{w}_{\mathcal{V}}(t), \quad \mathbf{w}_{\mathcal{V}}(t) \sim \mathcal{N}(\mathbf{0}, \mathbf{Q}_{\mathcal{V}}). \tag{31}$$

The change in velocity is modeled as a zero-mean white noise $\mathbf{w}_{\mathcal{V}}(t)$, and the expected squared acceleration as its variance.

For localization, the state vector is further augmented with Gaussian random variables b_i, one for each of the antennas. They are used to model the unknown bias from the real distance, which is a unique constant for every antenna, but which are modeled as continuous random variables to enable the filter to unwrap the phase. The measurement model is established using the distances to antennas $h_i(\mathbf{x}(t_n)) = |\mathbf{p} - \mathbf{a}_i| + b_i$. After discretizing the stochastic differential equation, the UKF introduced below can be applied [59].

3.5.2 Unscented Kalman Filter

Kalman filter (KF) is a well-known recursive state estimator for filtering noisy measurements. In addition to the classical linear formulation, there exists the Extended Kalman filter for smooth and differentiable non-linear systems, as well as UKF which is based on *Unscented Transform* (UT), i.e. sampling of sigma-points and propagating them through the system model. The advantages of the UKF over the EKF include more accurate approximations of nonlinearities as well as avoiding the derivation and computation of the Jacobian matrix at each iteration. After propagating the sigma-points through the non-linear system model, the mean and covariance of the state estimate can be obtained. Figure 12 illustrates the propagation of statistics through the Unscented Transform.

Computing the estimate using UKF is somewhat more convoluted than with the linear KF. For a non-linear system

$$\begin{cases} \mathbf{x}_k = f(\mathbf{x}_{k-1}) + \mathbf{w}_{k-1} \\ \mathbf{y}_k = h(\mathbf{x}_k) + \mathbf{v}_k \\ \mathbf{w}_k \backsim \mathcal{N}(\mathbf{0}, \mathbf{Q}_k) \\ \mathbf{v}_k \backsim \mathcal{N}(\mathbf{0}, \mathbf{R}_k). \end{cases} \tag{32}$$

with process and observation models $f(\mathbf{x}_{k-1})$ and $h(\mathbf{x}_k)$, and their additive white noise components $\mathbf{w}_k$ and $\mathbf{v}_k$, respectively, the state estimate $\hat{\mathbf{x}}$ is computed using the UKF as in [61]. The initial state is denoted with $\mathbf{x}_0$, and the initial state estimate and initial covariance are defined using Equations 33–34. Minus and plus signs are used in superscript to denote prior and posterior estimates, respectively.

$$\hat{\mathbf{x}}_0 = \mathbb{E}[\mathbf{x}_0] \tag{33}$$

$$\mathbf{P}_0^+ = \mathbb{E}[(\mathbf{x}_0 - \hat{\mathbf{x}}_0^+)(\mathbf{x}_0 - \hat{\mathbf{x}}_0^+)^\intercal] \tag{34}$$

A set of sigma-points are drawn to approximate the distribution by sampling the neighborhood around the previous posterior estimate:

$$\boldsymbol{\chi}_{k-1} = \left[\hat{\mathbf{x}}_{k-1}^+ \quad \hat{\mathbf{x}}_{k-1}^+ + \sqrt{(n+\lambda)\mathbf{P}_{k-1}^+} \quad \hat{\mathbf{x}}_{k-1}^+ - \sqrt{(n+\lambda)\mathbf{P}_{k-1}^+}\right], \tag{35}$$

where n equals the state dimension and λ is a scaling parameter

$$\lambda = \alpha^2(n+\kappa) - n. \tag{36}$$

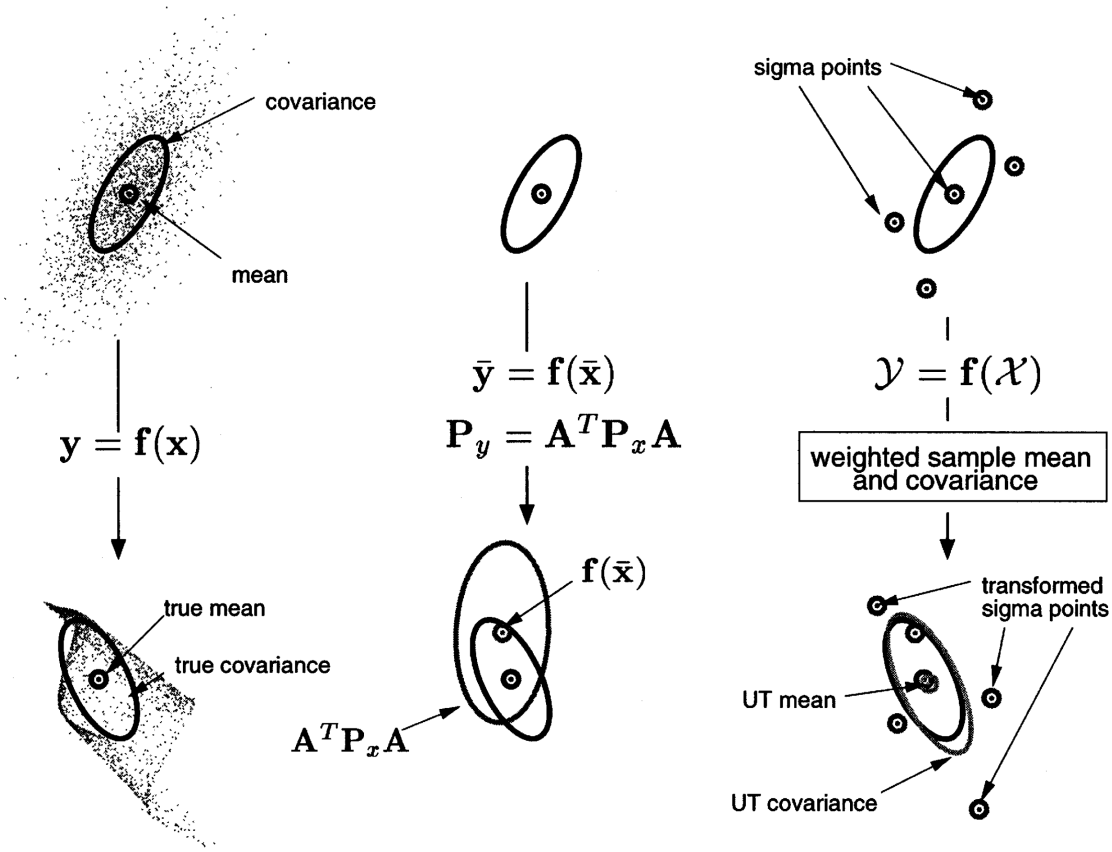


Figure 12: Propagation of mean and covariance in the system (left), EKF (middle), and Unscented Transform (right) [60, p. 231]

The constant α affects how the samples are distributed around the mean, and κ is a secondary parameter and usually chosen to be zero. The prediction step is done by computing the output of the process model with each of the column vectors $\boldsymbol{\chi}_{k-1}^{i}$ of the matrix $\boldsymbol{\chi}_{k-1}$:

$$\boldsymbol{\chi}_{k-1}^{*,i} = f(\boldsymbol{\chi}_{k-1}^{i}), \quad i = 0, 1, \ldots, 2n. \tag{37}$$

Initial state estimate and its covariance is computed from weighted sums of the sigma-points as

$$\hat{\mathbf{x}}_k^- = \sum_{i=0}^{2n} W_i^{(\mathrm{m})} \boldsymbol{\chi}_{k-1}^{*,i} \tag{38}$$

$$\mathbf{P}_k^- = \sum_{i=0}^{2n} W_i^{(\mathrm{c})} (\boldsymbol{\chi}_{k-1}^{*,i} - \hat{\mathbf{x}}_k^-)(\boldsymbol{\chi}_{k-1}^{*,i} - \hat{\mathbf{x}}_k^-)^\intercal + \mathbf{Q}_{k-1}, \tag{39}$$

where $\mathbf{Q}_{k-1}$ equals the process noise covariance at time step $k-1$, and $W_i^{(\mathrm{m})}$ as well

as $W_i^{(\mathrm{c})}$ are the weights of the mean (m) and its covariance (c) as follows

$$W_0^{(\mathrm{m})} = \frac{\lambda}{\lambda + n} \tag{40}$$

$$W_0^{(\mathrm{c})} = \frac{\lambda}{\lambda + n} + (1 - \alpha^2 + \beta) \tag{41}$$

$$W_i^{(\mathrm{m})} = W_i^{(\mathrm{c})} = \frac{1}{2(\lambda + n)}, \quad i = 1, \ldots, 2n. \tag{42}$$

Parameter β expresses information about the distribution of state variables, and for Gaussian variables $\beta = 2$. The sigma-points are updated using equation

$$\boldsymbol{\chi}_{k|k-1} = \begin{bmatrix} \hat{\mathbf{x}}_k^- & \hat{\mathbf{x}}_k^- + \sqrt{(n+\lambda)\mathbf{P}_k^-} & \hat{\mathbf{x}}_k^- - \sqrt{(n+\lambda)\mathbf{P}_k^-} \end{bmatrix}. \tag{43}$$

Predictions for observations are obtained by passing each of the updated sigma-point column vectors through the observation model

$$\hat{\mathbf{y}}_k^{(i)} = h(\boldsymbol{\chi}_{k|k-1}^i) \tag{44}$$

and computing the weighted sum of the output

$$\hat{\mathbf{y}}_k = \sum_{i=0}^{2n} W_i^{(\mathrm{m})} \hat{\mathbf{y}}_k^{(i)}. \tag{45}$$

The updated state estimate is obtained by computing the the difference between prediction and observation weighted by the Kalman gain $\mathbf{K}_k$:

$$\hat{\mathbf{x}}_k^+ = \hat{\mathbf{x}}_k^- + \mathbf{K}_k(\mathbf{y}_k - \hat{\mathbf{y}}_k) \tag{46}$$

$$\mathbf{K}_k = \mathbf{P}_{xy}\mathbf{P}_y^{-1}, \tag{47}$$

where

$$\mathbf{P}_{xy} = \sum_{i=0}^{2n} W_i^{(c)} (\boldsymbol{\chi}_{k|k-1}^i - \hat{\mathbf{x}}_k^-)(\hat{\mathbf{y}}_k^{(i)} - \hat{\mathbf{y}}_k)^\intercal \tag{48}$$

$$\mathbf{P}_y = \sum_{i=0}^{2n} W_i^{(c)} (\hat{\mathbf{y}}_k^{(i)} - \hat{\mathbf{y}}_k)(\hat{\mathbf{y}}_k^{(i)} - \hat{\mathbf{y}}_k)^\intercal + \mathbf{R}_k. \tag{49}$$

The measurement noise covariance $\mathbf{v}_k$ indicates the amount of noise that is included in the auto-covariance of the measurement $\mathbf{P}_y$. Finally, the updated covariance matrix is computed as

$$\mathbf{P}_k^+ = \mathbf{P}_k^- - \mathbf{K}_k \mathbf{P}_y \mathbf{K}_k^\intercal. \tag{50}$$

4 Embedded software development

In this section, the development process and software life-cycle of the embedded on-reader application is described. The context for development, relevant specifications, software architecture and design choices are presented, as well as the most significant implementation aspects. As detailed below, the targeted embedded system is industrial Impinj Speedway Revolution R420 UHF RFID reader device. In addition to the air interface protocol implemented by the reader, the operation of the embedded software is mainly governed by two protocol standards; the *Low-Level Reader Protocol* (LLRP) and EPCglobal *Reader Protocol* (RP). Commands are given to the reader over (local) LLRP connection, and the operations implemented by the reader – as seen from the perspective of the host or a user – conform to the Reader Protocol model.

Several software engineering process models exist, the most traditional ones being the waterfall model and, an extension of the former, the V-model. The development of the reader software broadly falls in to the waterfall model, and is thus well suited to be presented as a part of a thesis. The development starts from software requirements specifications, followed by design, implementation, verification and maintenance. Figure 13 depicts the waterfall model for software engineering.

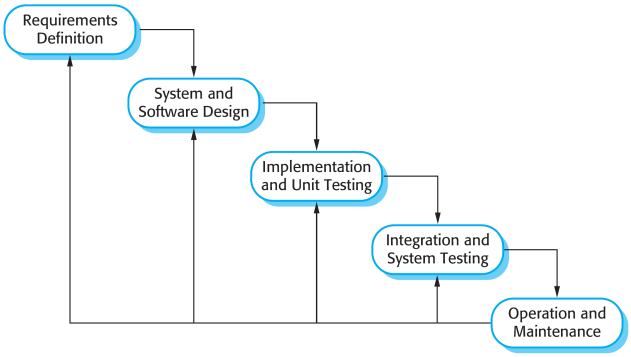


Figure 13: Waterfall model for software development process [62, p. 30]

Compared to the usual software development for desktop computers with x86 family of processors, the embedded systems aspect brings about peculiarities of its own, namely RAM, CPU and storage space constraints, and more elaborated builds with cross-compilation of the source code, including any 3rd party libraries. As a (firm) real-time system, execution times, overheads and CPU usage are also to be kept in mind.

This section is continued as follows. First, the context for the need and the importance of a reader software is introduced, followed by the description of the embedded target in question. The two protocols, LLRP and RP, are introduced shortly to gain understanding of the main source of the software requirements. Several software design aspects are considered, followed by implementation details related to architecturally significant decisions. The section is brought to conclusion by giving an overview of the finished software product in its state at the time of writing.

4.1 Overview of the target system

The context and needs driving this thesis stem from the industrial uses of UHF RFID. Typical RFID solutions targeted by this research are logistics tracking solutions with a database and server application in the backend. The RFID readers are deployed in the field to provide reliable and automatic identification of assets in deterministic points of material flow. The reader devices are connected to a network for receiving configuration and sending read reports. Readers may also receive input triggers from connected devices via GPI peripheral ports to start RF or reporting operations – see Fig. 14.

Modern UHF RFID devices intended to be interoperable or compliant with GS1 information system specifications [63] can be communicated with using LLRP. It is a compact binary protocol for sending commands to the reader and receiving read reports and other reader events over TCP/IP connection [64]. The protocol has specified message types for configuring reader RF operations, triggers, tag memory accesses and Gen2 protocol parameters. The author has found the Low-Level Reader Protocol being too low-level for many practical use cases most often encountered by an RFID middleware vendor, and that is one of the reasons a separate *Reader Protocol* layer [65] is implemented for day-to-day use on top of the LLRP.

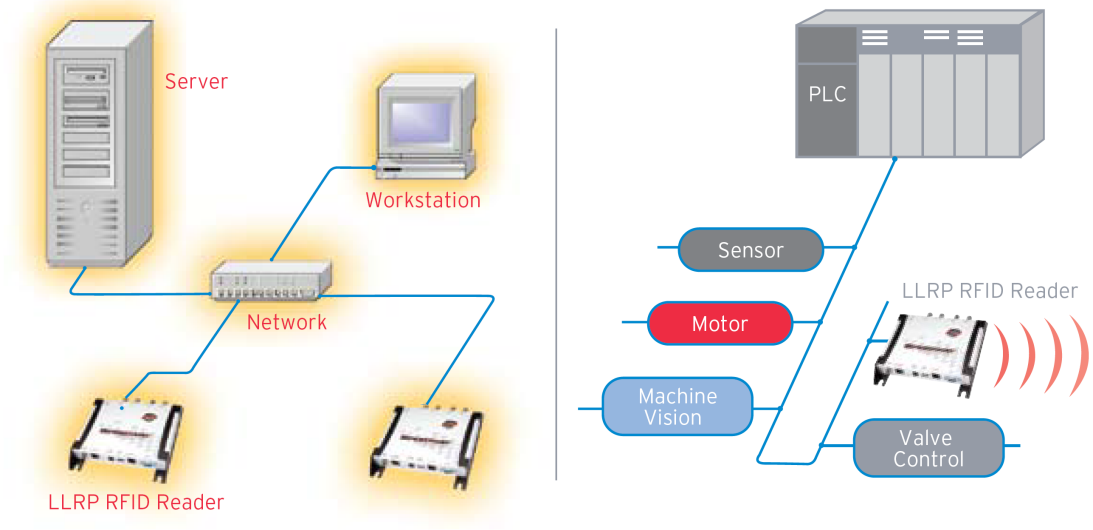


Figure 14: RFID control sources vary [66]

The GS1 EPC Information Systems specification (EPCIS) used to specify the Reader Protocol as the communication protocol between an RFID reader and higher level systems, but has now deprecated RP in favor of LLRP. The figure illustrating the EPCglobal Architecture Framework (Fig. 15) has stayed the same between various versions of the specification, excluding the choice of Reader Protocol. The situation is curious as RP which was also created by EPCglobal, seems to have been abandoned although specification complementary and extension to RP, the *Reader Management specification* [67] continues to have an official ratified status.

The EPCglobal information system architecture is illustrated in Figure 15 to help visualize the place of an RFID reader in an IT system. Nevertheless, the development of the software at hand started initially with the goal to replace another on-reader application that happened to conform to RP, and supporting the Reader Protocol seemed natural in order to have a drop-in replacement.

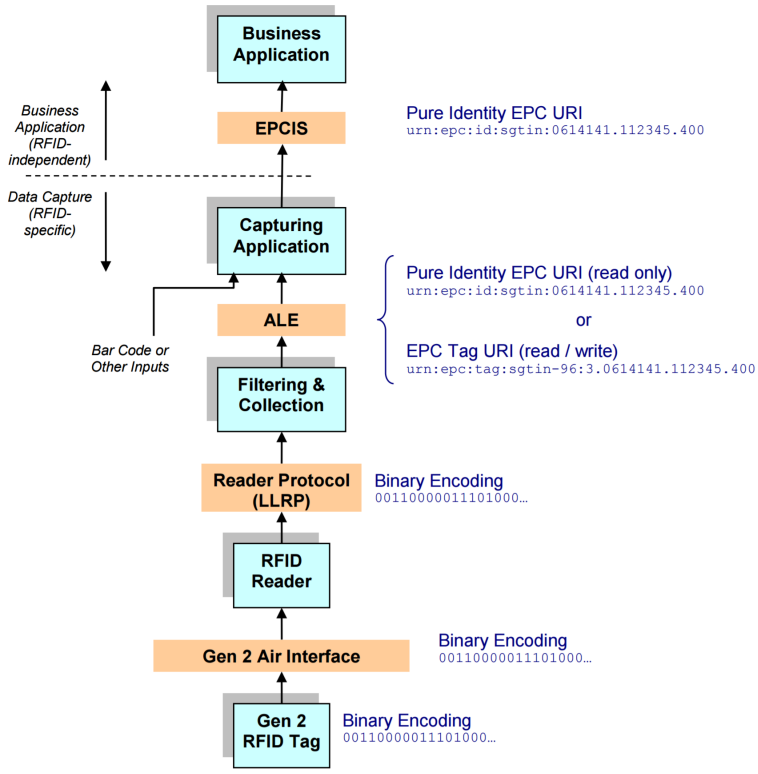


Figure 15: EPCglobal Architecture Framework [68, p. 23]

As mentioned, the device of choice is Speedway Revolution R420 reader (Fig. 16) with four monostatic antenna ports and LLRP connectivity. Developing custom embedded applications for the device is supported by the manufacturer Impinj, Inc. The device has an industrial casing and form-factor, a 24 VDC power connector or PoE (Power over Ethernet) as a power source, 10/100 base-T Ethernet connectivity and a distribution of GNU/Linux operating system tailored for embedded use. The details presented here are summarized from documentation [69] provided by Impinj, Inc. to its authorized partners.

The reader has following features, in addition to those given in Table 2. Antenna ports are equipped with reverse-polarity TNC coaxial connectors. The device has a RS-232 serial port using DE15 connector with four optoisolated inputs and outputs, and configurable de-bounce interval for the input pins. It also features an unconventional RS-232 serial console port over a RJ45 cable. USB device port with a HID driver, as well as USB host port for external Flash drive or a WiFi adapter are also included. Air-protocol and time-critical functions are implemented within an FPGA. The hardware block diagram is given in Figure 17.

Figure 16: Impinj Speedway Revolution R420 UHF RFID reader

Table 2: Reader device features [69]

Component	PCBA <v4.x.x	PCBA >v5.x.x
MCU	AT91SAM9260	AT91SAM9G29
Clock speed	200 MHz	400 MHz
Flash	128 MB	256 MB
RAM	64 MB	64 MB
SOP	16 MB	28 MB
SPP	8 MB	16 MB
CAP	8 MB	32 MB
UFS	32 MB	63 MB

The reader runs a Debian GNU/Linux 8 (Jessie) with kernel version 2.6.39.4. The Linux kernel manages available RAM, which is shared between the OS, reader applications, custom applications, and the RAM filesystem. No swap space exists. The Flash device is partitioned in a way to contain a dual firmware image architecture and an upgrade filesystem partition (UFS) so that minimal downtime during upgrades, as well as the possibility to fall back to a previous firmware image can be provided. Firmware image is organized to system operating partition (SOP), system persistent partition (SPP) and custom application partition (CAP), which contain read-only root filesystem, logs and configurations, as well as storage mounted read-write for the custom on-reader application, respectively. The partition sizes are given in Table 2 and the filesystem structure in Figure 18.

The CAP partition is freely usable for embedded software development, and may be used to contain the custom application software, application logs, and any libraries, tools and configuration files required by the application. The RFID operations of the reader can be commanded over an LLRP connection the on-reader application that can be made using a local TCP connection. Establishing an LLRP connection

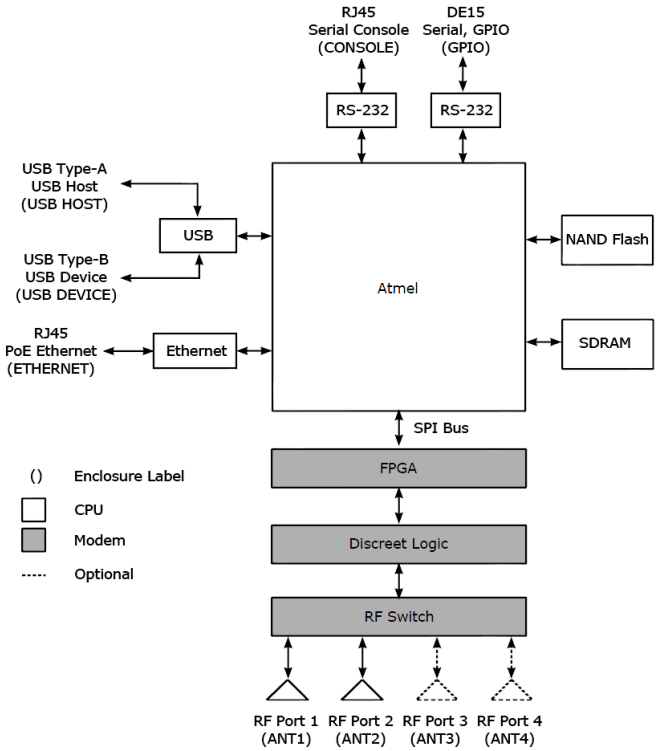


Figure 17: Speedway Hardware Block Design [69, p. 19]

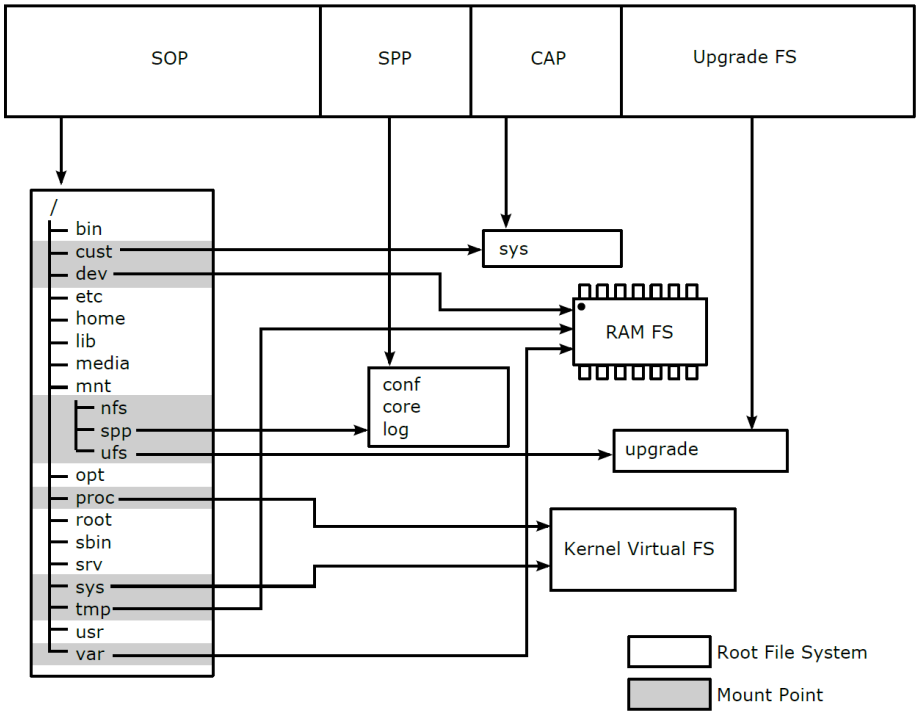


Figure 18: RFID reader file system layout [69, p. 25]

prevents other LLRP clients from connecting to the reader at the same time. The GPIO pins are controlled with LLRP message extensions implemented by Impinj.

Having an on-reader application is important for the following reason. Although there exists an open-source toolkit for LLRP clients with libraries for C, C++, C#, Perl, and Java languages, requiring business applications or any client software to have intimate knowledge of low-level protocols is not always meaningful. Instead, an interface with a higher level of abstraction is useful for having commands that are understandable to both humans and machines. So far all of the use cases the author has been involved in can be characterized requiring a configure-and-forget method for deploying RFID readers, which are then capable of asynchronously reporting RFID events to database applications, and possibly reacting to triggers on GPI or network ports. These needs can be met by writing an on-reader application that hides most of the complexity from outside actors.

4.2 Protocol specifications

The central communication protocols the embedded software interacts with are introduced below, and include Low-Level Reader Protocol and Reader Protocol. LLRP is reviewed for completeness and to understand the type of mapping that has to be done between the protocols, although the discussion is kept brief due to the fact that the application will be only interfacing with it and not implementing it. RP on the other hand is covered in more detail to give an understanding of its object model, its advantages and limitations. It should be noted that there are no existing libraries or toolkits available for RP the author is aware of, and it is assumed that devices that conform to the specification have their own proprietary implementations.

4.2.1 Low-Level Reader Protocol

LLRP is a binary network protocol from 2007 by EPCglobal, designed to act as an interface between RFID readers and clients. LLRP is a message-oriented protocol carried over a reliable stream transport, with TCP/IP as the only specified transport medium. The specification is freely available [64] – as is usually the case with standards published by GS1 EPCglobal. LLRP is, unlike RP, air-protocol aware, providing coupling with Gen2-dependent tag access commands, RF power levels and spectrum utilization.

Interfacing with LLRP-enabled devices can be done in software using an open-source, Apache 2.0 licensed library using various programming languages [70] . For the purposes of this thesis the only interesting binding is the LTKC, a C language API for the toolkit, due to reasons explained below in Section 4.4.

LLRP is an application layer protocol without message re-transmission and reordering facilities, but acknowledgments for messages sent from client to the reader are mandatory. A client can get and set reader configuration, discover reader capabilities, and manage air-protocol inventory and access operations. Reader can update the client about its status and access reports resulting from inventory and tag access operations.

Typical flow of protocol message transactions is depicted in Figure 19. Upon connecting to a reader, the client typically queries the device to discover its capabilities, resets the reader to its default configuration if not specifically designed to reliably work with disconnected operation, sets configuration values and adds antenna inventory and tag access specifications. The client may then listen for access reports the reader sends asynchronously, or if disconnected, the reader will accumulate reports to its internal buffer up to some implementation defined memory limit.

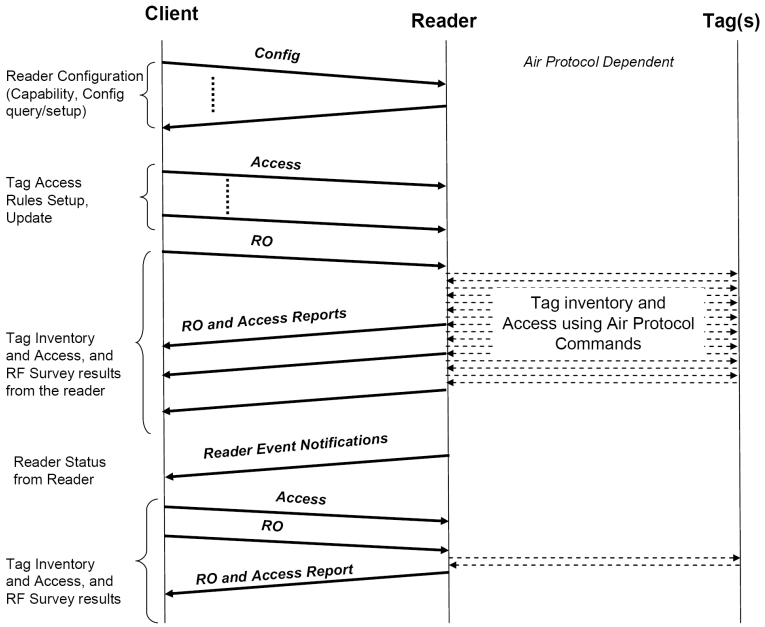


Figure 19: Typical flow of LLRP messages [64, p. 22]

An LLRP message may consist of several parameters and sub-parameters; to have a reader begin air protocol transmissions and read tags in the FOV, several parameters, or *specs*, have to be defined. The main spec that describes the operations the reader executes is the reader operations spec, or *ROSpec*. A ROSpec may contain *AccessSpecs* that are used to read or write tag memory contents in custom memory locations, as well as lock or kill tags. In addition, the ROSpec includes antenna inventory specs (*AISpec*) that bind antennas and *InventoryParameterSpecs* to configure air protocol commands used for inventorying tags in the interrogation zone of specific antennas. *BoundarySpecs* in a ROSpec define the start and stop triggers which determine the temporal extents of the operations. Triggering can be done for example on changing signal edge on a GPI port, or with a timer. Some of the messages have extension points, where vendors may provide their own commands and parameters – Impinj, Inc. among others provides its own build of the LLRP toolkit that is distributed with the vendor extensions built-in and ready to be used with reader devices of their own make. In conclusion, complex combinations of operations can be configured with proper selection of parameters in various number of specs and sub-parameters, and covering them further is not in the scope of this thesis.

4.2.2 EPCglobal Reader Protocol

The Reader Protocol standard (RP) is a specification by EPCglobal that defines the interface by which readers interact with middleware software applications. These readers may include RFID tag readers or optical barcode scanners, they may use any combination of RF protocols, they can be fixed or hand-held devices, and may also be capable of writing data into tags. The standard specifically insulates the higher application layers from knowing what RF protocols are in use, and specifies only the interaction between an EPCglobal-compliant application (*host*) and the device (*reader*), although the protocol itself does not necessitate any particular host application to be used. [65, p. 17]

The RP consists of three layers that specify the abstract interface and commands of the protocol (*reader layer*), serialization format (*messaging layer*), as well as well network facilities (*transport layer*). The protocol layers are depicted in Figure 20. The *reader layer* effectively defines the operations the reader is able to perform. The message/transport binding (MTB) in the Reader Protocol terminology defines a pair of message format and its transport; the last ratified version only defines two message formats (text and XML) and three transport mediums (TCP, HTTP, and serial COM). Other MTBs were left to be defined in the future.

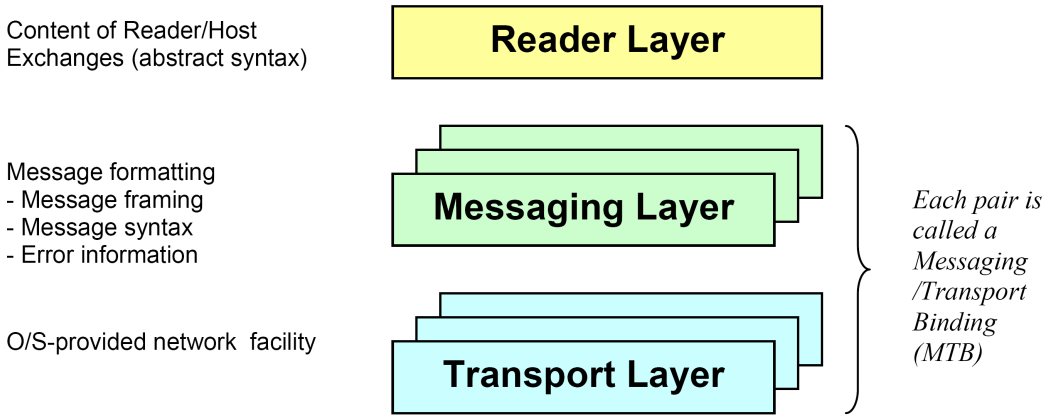


Figure 20: Layers of the Reader Protocol [65, p. 18]

The communication between the host and the reader takes place over message channels. A *control channel* and a *notification channel* is defined in the RP. The control channel is used to issue commands from the host to the reader following the request-response pattern. The reader may support more than one control channel, but is then required to respond to commands on the channel from which the command was received [65, p. 165]. Notification channels are defined separately as to enable asynchronous messaging from the reader to one or more hosts that may or may not be issuing commands to the reader.

The reader layer is a collection features and abstract commands the conforming device implements. Not all readers are required to implement all features, and there are only a small number of methods that *shall* be implemented. When available,

they can be accessed in a standard manner, and a standard set of error codes are used otherwise. Readers may even place restrictions on their implementation of the methods specified in the standard, e.g. by restricting the set of valid parameters for some specified call, or limit the number of supported objects of some type. Thus, even conforming readers can vary a great deal in features and functionality they provide. Reader Protocol, too, is open for vendors to provide their extensions which can be additional parameters to existing commands, or even totally new custom commands.

The messages exchanged at the reader layer follow the familiar request-response pattern; the host sends a request to which the reader responds with a normal response or an error response. Asynchronous messaging from reader to host may happen via notification channels, but some commands sent from the host may trigger the delivery of notification messages.

There are three types of commands in the reader layer, namely *do*, *set*, and *get* commands, which cause the reader to take action, change internal state or return data to the host as a response to the command, respectively. The standard introduces the abstract command format in an UML-like notation, the exact format of which is determined by the MTB used in the control channel. The format is given in [65, p. 48] as

```
[static] OBJECT.COMMAND ([
                    PARAMETER: DataType,
                    PARAMETER: DataType,
                    ...]): DataType
```

which basically represents familiar notation adopted by many object-oriented programming languages. Data types for parameters and return values are further defined in the document.

What makes the format elaborate is its encoding in one of the supported message formats. Using either the text or XML messages, or both, would result in implementing a custom parsing logic for them either way. What is then unfortunate is that the Reader Protocol specification has some inconsistencies, and arguably even mistakes in its XML Schemas, which can be observed when trying to define extensions to the command set, and even validating standard messages against the provided schemas. In the case of simple inconsistencies, the standard actually states a section of the document that is the overriding definition of requirements [65, p. 47] – a fact the author considers interesting for a standards document.

The dataflow between different subsystems given in Figure 21 is a non-normative illustration that does not constrain the design of a reader – the Reader Protocol does not care about the order in which the device carries out its computations, e.g. filtering tags participating in the data acquisition may be done on the air-protocol level or later in the report buffer before delivering read reports in a notification.

Even if not implemented as-is, the subsystems of the reading pipeline are still useful illustrations and mental models, capturing basically the essence of RP: how to get tags read, events generated, and reports delivered to a host. The extensibility of the protocol model makes the pipeline a very good starting point when reacting to

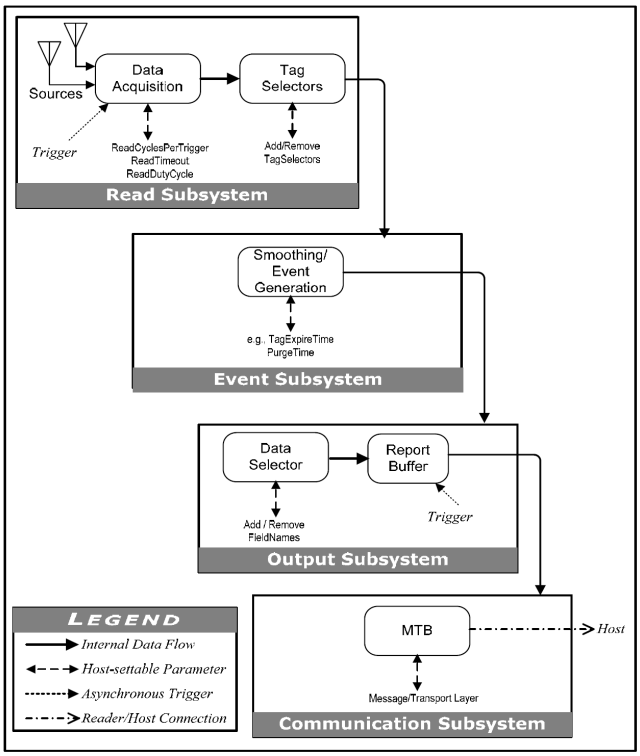


Figure 21: Conceptual reading pipeline [65, p. 21]

the growing list of features required from a reader software. Specifically, even though the standard presents a state machine for event generation (Fig. 22), additional events may be generated on each state transition, and technically even more states or an independent and parallel event generation scheme can be added. This provides a convenient model for e.g. location-based event generation of inventoried tags, as well as adding other custom event types on-demand.

The read subsystem is the input or data acquisition stage of the pipeline. Read points, i.e. antennas, are assigned to one or more *source* objects. Sources are responsible for keeping track of their own inventory of tags and generate events only for them. Data begins to flow as read operations are executed using read triggers. Various types of triggers can be defined. Filtering of tags is defined in terms of *tag selector* objects, which are used to reduce the volume of data by including only tags of interest.

The event subsystem is a stage where data may be smoothed and its volume reduced over time. As the amount of data coming from readings of RFID tags can be substantial, this stage is introduced to generate events only when something interesting happens from the point of view of the application. Most obvious types of events are tags appearing and leaving the interrogation zone of the reader. Several types of events are modeled with their timing parameters, as seen in Fig. 22, some of which are used to generate events only when a tag has been continuously observed for some time and other ones in cases where the tag is but glimpsed without being reliably determined to be present in the field. The frequency of the generated events

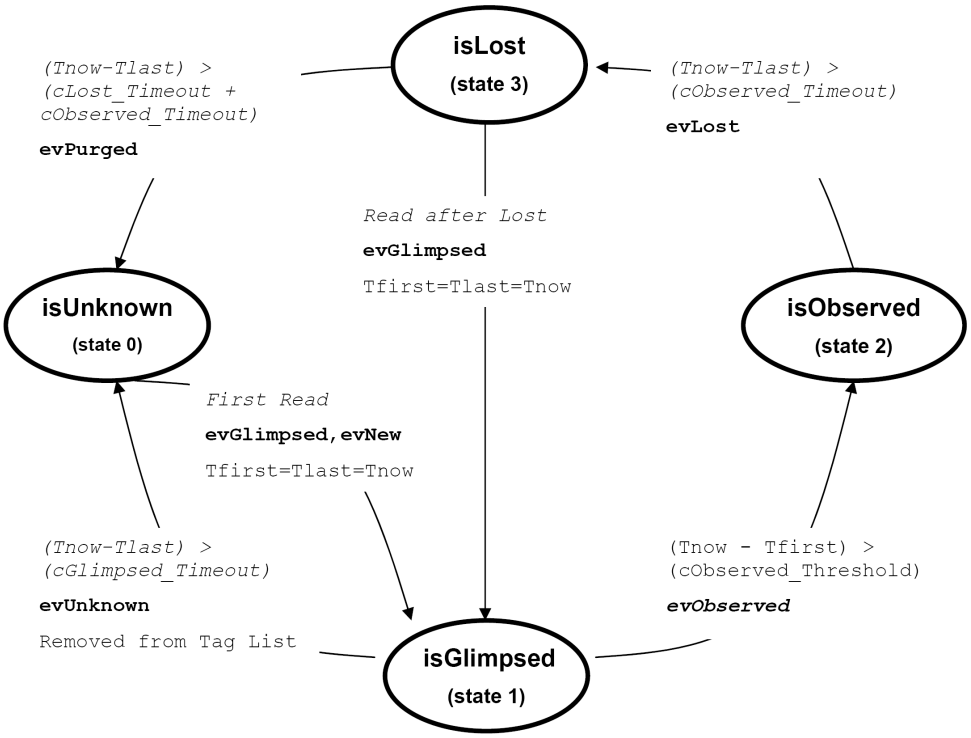


Figure 22: Event life cycle [65, p. 27]

can be configured by altering the threshold and timeout parameters on a per-source basis.

The output subsystem is a more of an implicit stage built into the rest of the pipeline; it consists of report buffers for each of the notification channels in the device, as well as a *data selector* object for each channel that acts as an output filter selecting only the specific set of fields to be reported which that have been requested by the host. The contents of the report buffer are delivered asynchronously when a trigger fires, or the host can synchronously request the contents of the buffer by sending a command. The communication subsystem is also implicitly defined as a part of notification channel objects, being responsible for the message-transport binding and message delivery over the transport medium.

The MTBs define the message serialization formats, handshaking and transport layer connection handling, as well as framing the message if needed by the transport layer. Both text and XML message formats can be used over any of the defined transports (COM, TCP, HTTP). The text message format for commands from host to reader follow one-to-one mapping from the Reader Protocol UML and are defined in terms of augmented Backus–Naur form (ABNF), a syntax for specifying grammars used to define syntax of programming languages, formats, or communication protocols. The read report text message encoding consists of lines of text with fields said to be delimited by whitespace, although the ABNF definition and examples specify the format to be delimited by commas [65, p. 197]. The XML messages ought to follow the XML Schema documents contained in the standard.

The Reader Protocol proposes an object model (Fig. 23) to overcome the simplifications made in the pipeline model. The object model is shared with the EPCglobal Reader Management specification [65, p. 31] [67, p. 14]. In short, the model consists of a *ReaderDevice* singleton, which manages all the other objects added to the list of objects with static functions. A device may have an arbitrary number of any of the objects depicted in the figure, and the rest of the cardinality constraints can be seen as well.

In addition to the sources, notification channels and triggers having associations instead of composition – which is not as elegant as it could, and turns out to cause implementation issues – the relationship between notification channels and data selectors is also confusing, as the operation of notification channels under the condition of having zero data selectors created is not defined anywhere in the standard. Error definitions for responses to commands under such conditions do exist, but that does not in fact solve the undefined behavior. The problem can be overridden due to the fact that most of the functionality in the specification is optional and in practice the only restriction defined for vendor-specific commands is to not duplicate any mandatory or optional command with a different method.

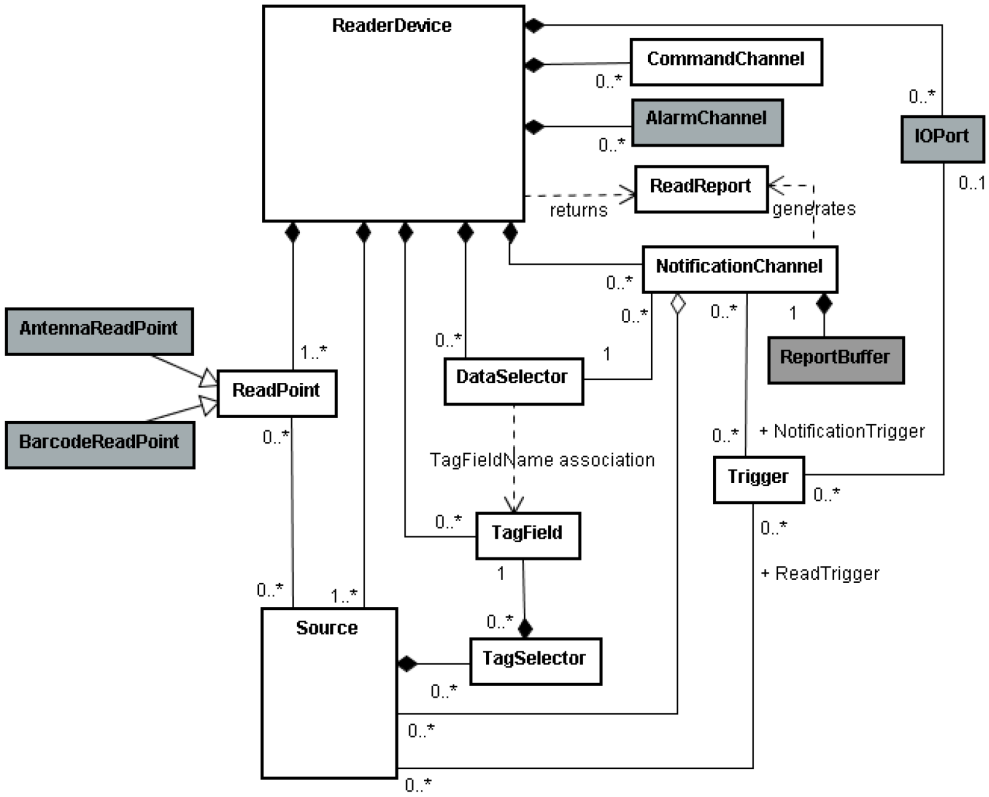


Figure 23: Reader Protocol object model [65, p. 31]

4.3 Requirements

Software requirements describe what a system should do; what functionality it provides and what are its constraints. Software requirement as a term is not used consistently in the industry, and is sometimes used to make an abstract statement about a service the system provides and for some it is a detailed definition of a function [62, p. 83]. Requirements are typically divided to functional and non-functional, or performance requirements.

For the RFID reader software most of the functional requirements follow from the Reader Protocol and Low-Level Reader Protocol interfaces, and some from the use cases encountered in the field. In addition to the protocol specifications introduced above, the limitations of the embedded target set constraints on the implementation. Constraints limit the developer's options, including but not limited to regulatory compliance, hardware limits, interfaces, reliability, usability, and security.

For example, the network hardware interface is limited to a standard RJ-45 based Ethernet. Networking support in software is available using POSIX sockets. Also, commanding the reader happens over a local LLRP connection handled by the LTKC library. Other network-related constraints clearly exist.

Another set of constraints govern the real-time operations. The RFID event generation model necessitates concurrent – but not necessarily parallel – operations to enable real-time events. In practice this means that deviations in the order of few milliseconds in the event generation stage are acceptable as long as the original timestamps are preserved throughout the reading pipeline. Disruption of real-time program flow is not considered harmful during user interaction and reconfiguration.

The software is bound to use a maximum of 16 MB of RAM and 8 MB of storage space. The most noticeable implication following the low amount of RAM is that the maximum number of native threads has to have a limit determined at compile-time. When using pooled threads, the caller is required to be prepared to handle situations where there are no worker threads available. Library functions that use any form of default thread pooling under the hood cannot be used when the exceptions have to be handled in a global error handler.

The non-volatile storage in Impinj Speedway Revolution R420 uses NAND Flash technology and suffers from Flash cell wearing every time they are written to, and thus no read report data or any other high-volume or fast changing data should be written to the internal storage. In addition to keeping the number of third-party libraries to a minimum, the libraries and program code alike are compiled and linked with priority given to size-optimization options. In practice, libraries are compiled and linked as static libraries with linker flags to enable a whole-program optimization at link-time.

All software built for industrial production use has to be reliable. For the reader software this means that no read report data should be lost in the case of network failure. The software should be able to tolerate and recover from network failures, and re-connect automatically if connection had been reader-initiated. The software has to support continuous operation without failing with uptime measured in months. Power-supply is assumed to be uninterruptible and no mitigations against data-loss

during an unexpected power outage is required.

Software and information security is also associated with reliability. For all the purposes of outgoing network connections from the reader, a secure, encrypted option should be available. For all non-local inbound control channel connections only encrypted connections are accepted, and all non-local control channel connections are authenticated. All functionality and services provided to network have to be secured against unintentional misuse, and – to a reasonable extent based on risk assessment – against intentional misuse. Only reputable cryptographic libraries and primitives can be used for any purpose – one of the most important and widely-known aspects of data security is to never roll your own cryptographic algorithms.

4.4 Implementation

The implementation of the embedded application is mainly built using C++11 programming language. The architecture can be considered concurrent as well as event-driven. Implementation aspects of some of the dominating features are presented below.

4.4.1 Dependencies

The C++ Standard Library is part of the C++ language standard and provides templated classes for generic containers, algorithms and stream manipulation. It is used by a large portion of the desktop software written in C++, and is extensively relied upon by many third-party C++ libraries. Along with the C++11 language standard, native support for multithreading, atomics and proper smart pointers was added. Most often used items are container classes `std::vector` and `std::deque`. `std::function` is used as generic functor adapter, often together with C++11 lambda functions. All pointers are wrapped in `std::unique_ptr` where applicable. `std::string` is used as the *de facto* string library.

LTKC is the C language version of the LLRP toolkit. A custom binary release with vendor-specific extensions is available from Impinj to its authorized partners, along with the company's other embedded development tools. The C library is preferred over the C++ version for a couple of reasons. As C++ does not have a stable ABI due to the *name mangling* built into the language, using a pre-built binary release of any C++ library implies all of the C++ code has to have been built using the same compiler version. Updating the library would require updating the compiler at the same time, which then again would mean re-compiling all the other libraries as well. On the other hand, changing the compiler version would not be possible if one were to keep on depending on the vendor extensions of the LLRP toolkit and not build it from a publicly available source code repository. C binaries are known to have relatively stable *application binary interface* (ABI) across compiler versions.

Another reason for choosing the C library was that it was found to be actually easier to use, and more lightweight than its C++ counterpart. This stems from the way the toolkit is built; it uses XML declarations to generate the library code, which

also happens to result in a very intricate class system. The C++ library requires every message, parameter and sub-parameter to be first dynamically allocated and then added to the list of parameters and sub-parameters of others where needed – for every transaction. This requires a lot of dynamic allocations and passing of pointers, and because of the missing of copy semantics in the library none of the instances can be re-used as they are automatically freed during an LLRP transaction. With the C library none of that takes place, the structures can be statically allocated and initialized with correct type descriptors with one-liners, and adding sub-parameters is as trivial as assigning pointers to member variables. In fact, the C++ library feels just like a wrapper around the C library in that regard as due to being automatically generated, there exists a clear mapping between the types in the two languages. Also, in the C++ version every sub-parameter assignment has its own pair of get and set functions, which in this case is an inconvenience.

OpenSSL [71] is the *de facto* cryptography and TLS library securing the majority of the encrypted connections in the Web. OpenSSL is an open-source library written in C with a permissive Apache-style license. It is shipped with practically every distribution of Linux, and a build of the library is shipped with the Octane firmware for Speedway Revolution reader, but due to its central role in security, the library is built from the most up-to-date source separately for the custom application and shipped within the CAP.

The POCO C++ Libraries [72] are well-engineered [73], cross-platform open-source libraries for networked applications that can be run even on embedded systems with as low as 4 MB of RAM. The libraries are written in standard C++, and they depend on the standard library, some platform-dependent system libraries like sockets that have been encapsulated behind a standard interface and handled by the build system, as well as some optional third-party libraries, like MySQL and OpenSSL. One could well consider the POCO libraries to be the set of the most well-written and usable C++ libraries in existence. Among the provided facilities are framework for application subsystems, logging, multithreading, events and delegates, hash tables, XML parsers, configuration files, and network programming. Even a multithreaded HTTPS server based on the reactor pattern is included.

Lua is a lightweight and dynamic, usually interpreted programming language with low memory footprint running in a register-based VM written in C. Lua has interoperability with C language built in that works by passing values in a stack. Lua is a production-ready language widely used as an embedded scripting language in high-budget video games, commonly known as *AAA games*, and in many software tools like Adobe Photoshop's Lightroom, and FEKO, an electromagnetic simulation and 3D field analysis tool. Lua is amongst the fastest dynamic language implementations, and even more so when using *LuaJIT*, a just-in-time (JIT) compiler for Lua. LuaJIT features a high-speed interpreter hand-written in Assembly languages, and an optimizing JIT compiler [74]. It supports the ARM architecture as well.

Several bindings to various languages for Lua exist. For C++11, there exists an open-source project *kaguya* [75] that uses C++ templates to hide the gritty C interface details behind an idiomatic C++ header library, and makes integrating Lua and LuaJIT to C++ projects a trivial task.

Lock-free data structures are in general some of the hardest things in parallel computing to get right. An open-source header library for lock-free multi-producer and multi-consumer concurrent queues in C++ is available in [76]. The library features a single-header implementation, full thread-safety, C++11 move semantics, portable implementation using only the standard library, and exception safety. The library offers some advantages over better-known alternatives, but has some preconditions that have to be met, namely, the queue is not linearizable nor sequentially consistent. It means that elements will not necessarily come out of the queue in the order the were put in by different producers and that dequeue operations can fail even if the queue is not empty while there are producers enqueuing items.

Although the web server component of the application is multi-threaded, rest of the program logic takes place in a single thread. Overheads that incur from synchronizing execution and data access between multiple pre-emptively scheduled threads can be avoided when the application design permits cooperative multitasking instead. Open-source coroutine library *lthread* written in C is special in the regard that it is designed to allow blocking calls and I/O access inside a coroutine making the calling coroutine yield while waiting for the event to complete. These lthreads use a user-space scheduler that transfers the execution between the lthreads, each of which has their own stack. The library uses the *madvise(2)* system call to save stack space and enable creating even thousands of lthreads while keeping a low memory footprint. [77]

Unit testing has been applied using Google's C++ unit test framework. The framework is used in notable projects like the Chromium web browser, the LLVM compiler, Google's Protocol Buffers data interchange format and the OpenCV computer vision library, in addition to Google's other internal uses [78]. Unit testing with the framework consists of test fixture classes, test case functions and various types of assertions. Test cases are registered and run automatically. The library is even thread safe on platforms where the POSIX thread library (*pthread*) is available.

The web interface is created using HTML, CSS, and several small JavaScript libraries, most of them being built on top of jQuery [79]. In addition, Google's Material Design Lite framework [80] is used exclusively in building the visual design of the user interface.

4.4.2 Subsystems

The embedded software consists of several subsystems. On the top level there is an application class responsible for initializing the subsystems and active objects. Any C++ exception that might be thrown in any thread and not handled elsewhere is caught and logged and – depending on the type the exception – possibly ignored without issues or in the case of a bug – which an unhandled exception often is – the application is terminated. The startup script responsible for spawning the application is also responsible for restarting the application on a non-zero exit value and rebooting the device otherwise. The coroutine responsible for applying the settings from the `ReaderDevice` object of the Reader Protocol into the LLRP client, as well as the coroutine responsible for executing Lua scripts, also live in the scope

of the top level application instance.

The `ReaderDevice` singleton described in RP is implemented as a single composite member of the application. The correspondence between the implementation of the C++ class and the definition in the Reader Protocol object model is pretty straightforward. The `ReaderDevice` instance is bound to Lua as userdata with a metatable providing the public interface callable from Lua. A more detailed description of the binding and its issues is given in Section 4.4.4.

The LLRP client implementation is the workhorse of the application, responsible for configuring the RFID operations and processing and forwarding tag reports to be processed in the event generation stage of Reader Protocol `Source` objects. With clever use of the LTKC library and modern and idiomatic C++, the same operations implemented in over 2000 lines of C++ code in the examples of the LLRP toolkit can be done in under 500 lines of code with a better design and better error handling.

The web server is an independent subsystem of the application in the sense that the only things connecting it to the rest of the process are the shared process memory space, library code, and the observer pattern implemented with event and delegate object pairs to pass read reports and RP events to request handlers streaming the events to a web browser via server-sent events (SSE). All other interaction with the application logic proper is done via a local TCP socket by passing Lua snippets and returning output over the connection. This design was chosen to have a method for inter-process communication done manually from the command line in addition to the web server component. The server uses its own pool of worker threads handling the incoming connections, and the few types of request handlers are implemented in C++. The HTTP request handlers are mainly concerned with responding with static documents, streaming events and passing the Lua scripts from the client and responding with the output.

Logging facilities provided by the POCO libraries are used extensively. Some of the C++ exceptions are logged by the syslog daemon of the reader, some debug information and possible security incidents are written to the Flash storage to keep them non-volatile and retrievable across reboots. All low-level tag read reports can also be logged to a file in a USB storage device attached to the reader if required. The logging options are configurable and enable conveniently printing all logs to console when developing on the host system and have the logs written to the right channels when the application is deployed to the reader.

4.4.3 Object model

The software is implemented using object-oriented design. The Reader Protocol model for `ReaderDevice`, `ReadPoint`, `Source` and `NotificationChannel` objects has been mapped to C++ classes and many of the functions specified for them are implemented.

The event generation and smoothing stage of the RP model is of special interest in the design. Every `Source` object has its own data structure for containing the inventoried tags and information about their history of read operations. As the number of unique tags simultaneously kept in the memory by each `Source` is not

known, the data structure must be chosen so that the performance is predictable even under high loads. In algorithmic terms this means that the container storing the tags should provide operations with constant *O(1)* complexity with respect to the number of tags. This is important because each `Source` object has to be able to asynchronously generate events uniquely for each tag, and individually polling the state of every tag would eventually become infeasible.

The event generation problem is solved by introducing an LRU cache with an expiration timeout for the elements at each of the states in the state machine logic (cf. Fig. 22). The POCO libraries provide a caching framework, but unfortunately the containers have operations of $O(\log n)$ complexity. Instead, `Poco::HashMap` is used to provide amortized $O(1)$ lookups, insertions and deletions. Together with a linked list that has $O(1)$ insertions to the head and end of the list a container can be built that meets the requirements. The design is illustrated in Figure 24. The cache then works as follows: Insertions are made by appending the linked list with a new node containing the tag specific information, and inserting an iterator – or a pointer – to the node into the hash table with the EPC of the tag as a key. If an entry already exists for a given EPC, the node pointed to by the hash table entry is moved to the end of the linked list as the tag just became *most recently used*.

An update coroutine is then scheduled to run to remove the *least recently used* nodes from the head of the list and from the hash table had they been expired, i.e. the time since the entry was last accessed is longer than the specified timeout. After the update, the coroutine can always be set to yield until the next possible expiration happens to reduce CPU usage.

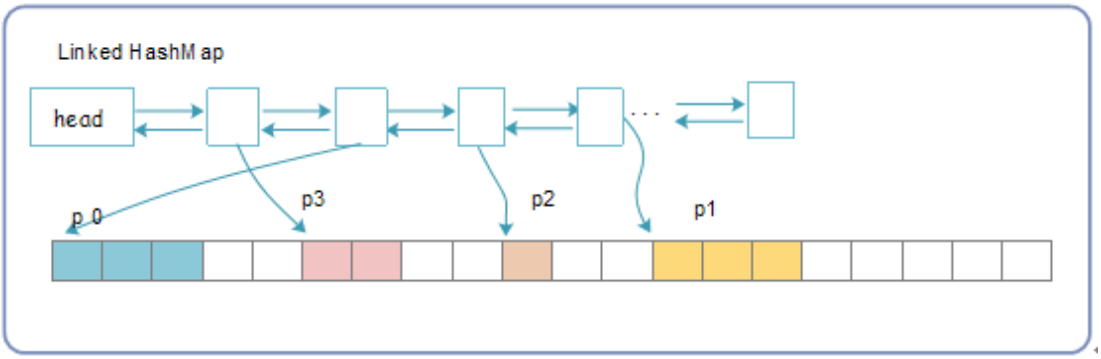


Figure 24: Constant-complexity LRU cache design [81]

Because the EPC codes in Gen2 tags may be of variable widths, but are still most often 96 bits long, a custom class with small-object optimization is implemented for storing the binary data. The LTKC provides two different array types for 96-bit and variable-width EPCs, but it is preferable to have a single interface that does not care about the length. The implemented byte buffer class stores raw byte sequences of 15 bytes and below statically, and for longer sequences makes a dynamic allocation with the required width. The magic number of 15 bytes follows from the need to store the length of the sequence somewhere, and from the fact that these extra bytes on top of the 96 bits do not necessarily cost anything due to memory alignment requirements.

The hash table implementation has to be aware of the type of the key used, and now that a custom byte array class is available, a hash function has to be provided for deriving a hash from the sequence of bytes. Many hash functions suitable for hashing different types of keys exists, but comparing them is outside the scope of this thesis. The hash function currently used is the 32-bit version of the *Murmur3* hash function, the code of which has been originally published in [82] as open-domain.

The RP object model provides a foundation which is easily extendable. The most interesting extension in the context of this work is location-based event generation. Custom event types have already been designed and it is trivial to add more. Especially, the design lends itself for adding custom fields to be reported in the midst of standard notifications, so not only the core object model may be extended, but the reporting formats as well.

4.4.4 Lua binding

The selection of Lua scripts as the technology of choice for RPC is acknowledged to be unconventional. The rationale for this design was essentially to replace the XML message format of the Reader Protocol with a more elegant implementation. Abandoning the Reader Protocol conformance in this regard had the advantage of being able to leave out the XML support library and over 1000 lines of custom parsing logic from the build, reducing the optimized binary size by over a megabyte. LuaJIT when compiled with size optimizations into a shared library for the target platform resulted in a 386 KB binary file.

To make a practical statement, the message format of the two approaches can be compared. Consider the Reader Protocol command

```
NotificationChannel.setAddress(addr: address): void
```

which is invoked as

```
channel1.setAddress("http://myhost.com/epc")
```

and formatted in XML as

```
<command>
    <id>1234</id>
    <targetName>channel1</targetName>
    <notificationChannel>
        <setAddress>
            <addr>http://myhost.com/epc</addr>
        </setAddress>
    </notificationChannel>
</command>
```

Using Lua bindings, the same command can now be given e.g. in the form

```
my_channel = ReaderDevice:getNotificationChannel("channel1")
my_channel:setAddress("http://myhost.com/epc")
```

or event more compactly

```
NotificationChannel("channel1"):setAddress("http://myhost.com/epc")
```

without the need of writing a parser and risk introducing bugs in the process. Of course, there is not that much of a difference between the RP text format of commands and Lua, but outside of supporting the syntax for the sake of conformance – which has never been required nor needed – it does not provide any advantages either and would still require a custom interpreter.

The advantages of embedding Lua accumulate, when users have to deal with the command syntax, when developing a web interface to send commands in the background from a browser, and when the application has to be able to load previously saved configuration from a file. Afterwards the choice has gained even more attraction as it has been trivial to implement a runtime debug console and make the software more open to extensions. A clear merit for Lua is that it has been designed to be embedded in applications, and it provides the means to sandbox untrusted code. Binding C++ classes, functions and variables to be usable from Lua is trivial with the right library. Excluding the few lines of boilerplate code per class, registrations of member functions for a C++ class in Lua are one-liners.

C++ objects are represented in Lua with a *userdata* data type and the data is managed by the Lua garbage collector. Lua userdata supports *metatables* that can be populated with functions that interact with the object as one would normally expect to happen in object-oriented languages. The problem then, is to make the memory management work between garbage-collected Lua and the RAII paradigm of C++. The solution is worth its mention in this thesis because no other safe way exists for enabling such an automatic binding while having the C++ code being responsible for the lifetime of an object.

The memory management issue is solved with an additional wrapper class and a dynamically allocated value wrapped in a `std::shared_ptr` for each of the classes to be exposed in Lua. The way this works is that the wrapper class and its methods are exposed to Lua, and the class contains a `std::weak_ptr` to the shared value of the original class as well as a reference to it. When instantiating an object in Lua, a userdata value with the copy of the `weak_ptr` is managed by Lua, and when calling any of the wrapper functions from Lua, the existence of the referenced object is checked using the `weak_ptr` before calling the member function of the original object. This enables graceful failure in cases where one tries to use an already deleted C++ object from Lua – otherwise a segmentation fault would follow. Consider the simple UML class diagram in Figure 25 for reference. In the figure, a `NotificationChannel` instance is managed by the C++ runtime and instances of `NotificationChannelWrapper` are managed by the Lua VM.

Running code provided outside the system has obvious security risks. Fortunately, Lua provides native facilities for sandboxing by offering *environments* and *debug hooks*, as well as by allowing the execution of code provided only as text, prohibiting the use of Lua bytecode. A sandbox is created by defining a new empty table and assigning it only the methods to be white-listed. By running the code with the newly created table as the current environment, the code running in the sandbox has access

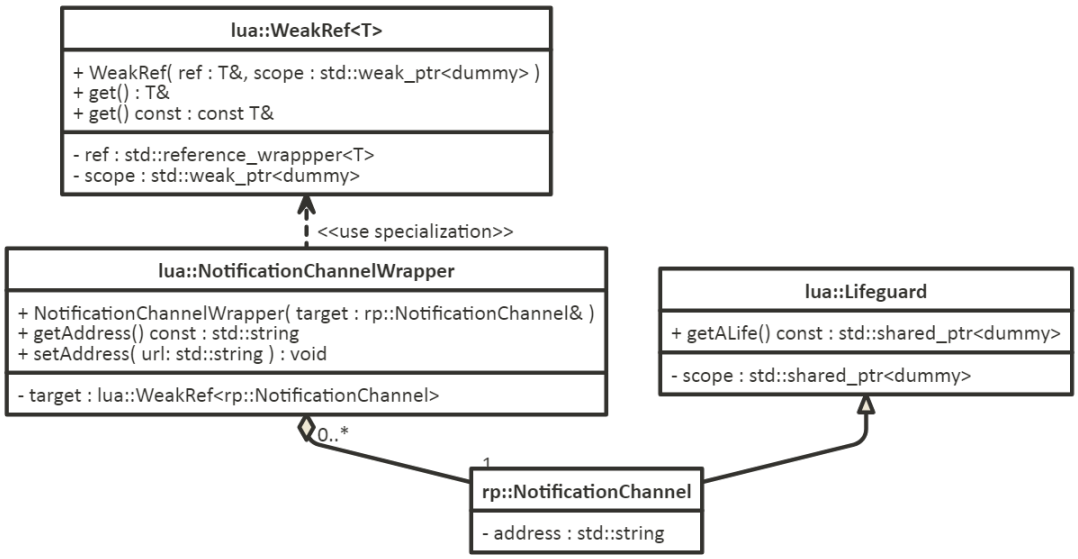


Figure 25: Safe Lua/C++ binding concept

only to the functions explicitly exposed to it. To prevent infinite loops and other forms of denial-of-service threats and resource exhaustion, debug hooks can be set that the interpreter calls after a specified number of instructions, at which point an error can be raised if memory consumption and execution time quotas have been used.

Not that it would matter in the threat model of the reader software, but an interesting thing to note is that *address space layout randomization* (ASLR) can be trivially defeated in Lua, as the default `print` and `tostring` implementations return the actual memory address of the object, say userdata or a function, when given as a parameter for them. The step taken to mitigate the issue is providing custom implementations of the said functions that return just the type information.

4.4.5 Web interface

A web interface for managing the reader is provided within the application. The web technology stack utilizes static HTML, CSS, and JavaScript files for providing a modern and responsive user interface with minimal resources. Many of the fanciest features available have been implemented on the client-side, after providing a control API via HTTP POST requests and Lua, as well as a simple C++ request handler that streams reader-generated events exploiting SSE. The HTTP server class provided by the POCO libraries makes implementing extremely lightweight embedded web server a straightforward process, requiring only a factory class for creating request handlers that respond to HTTP requests sent by the client and a set of request handlers implementing the actual logic.

Request handlers are dispatched to a configurable pool of native threads by the

underlying *reactor* that accepts incoming connections. Request handlers are then provided by a factory class that instantiates the actual handlers based on the content of the incoming request.

All of the request handlers in the reader application inherit from a single base class that makes security checks before passing the handling to subclasses. This way mitigating some of the most important security issues such as checking the response for too long URIs or content size, as well as the HTTP Basic Authentication headers can be concentrated into a single place.

In addition to the stream handler responsible for SSE, and the control handler responsible for mediating between the client and the Lua VM execution unit, a document handler has been implemented to serve static files from a specific web directory on the custom application partition. A simple form of cache control using standard HTTP header fields – including *ETags* – is provided to avoid serving the same files over and over again until possibly updated with the next firmware upgrade.

4.5 Deployment and maintenance

The custom application, along with all of the other contents in the CAP, are deployed on the reader as a single package generated for that specific purpose. The upgrade file can be uploaded to the reader using the HTML form provided by the web server shipped with the system firmware, from a USB storage device, or it can be downloaded from the network. All operations concerning the upgrade filesystem naturally require root user access.

The upgrade file is generated from the contents of a single directory on the host system using manufacturer's file system tools. Building the executable and library dependencies and finally generating the upgrade file can be automated using standard Unix build system tools, such as *GNU make* and *makefiles*. It is customary to integrate running a set of unit tests into the build process.

During development, a secure access to the operating system shell, an FTP server and the GNU Debugger daemon running on the reader are at the developer's disposal. In general, most of the services running on the reader should be disabled for security reasons when the reader is deployed.

Quality assurance (QA) is part of software life-cycle model and covers a set of practices with which the amount of defects in the shipped software product can be lowered. The most notable aspect of QA is software testing. As mentioned above, Google's test framework is the tool of choice in this software project. Although it is often difficult to test networked components, there are always some classes in the object model with low coupling. In fact, arguably the most critical pieces of code used to implement the LRU caching and byte arrays with small-object optimization are targeted with a set of unit tests to achieve a 100% code coverage. Still, some parts of the system remain to be completely untested in terms of unit testing, and an unfortunate fact is that many defects have slipped through into production code for which hotfixes have then been put together.

Proper version control and configuration management are imperative in all software engineering practices. In this work, Git software has been used for version

control during the development. By following the branching model that was later released as the widely used git-flow extension, the history of changes in the codebase was intended to be kept as logical as possible. The workflow is illustrated in Figure 26.

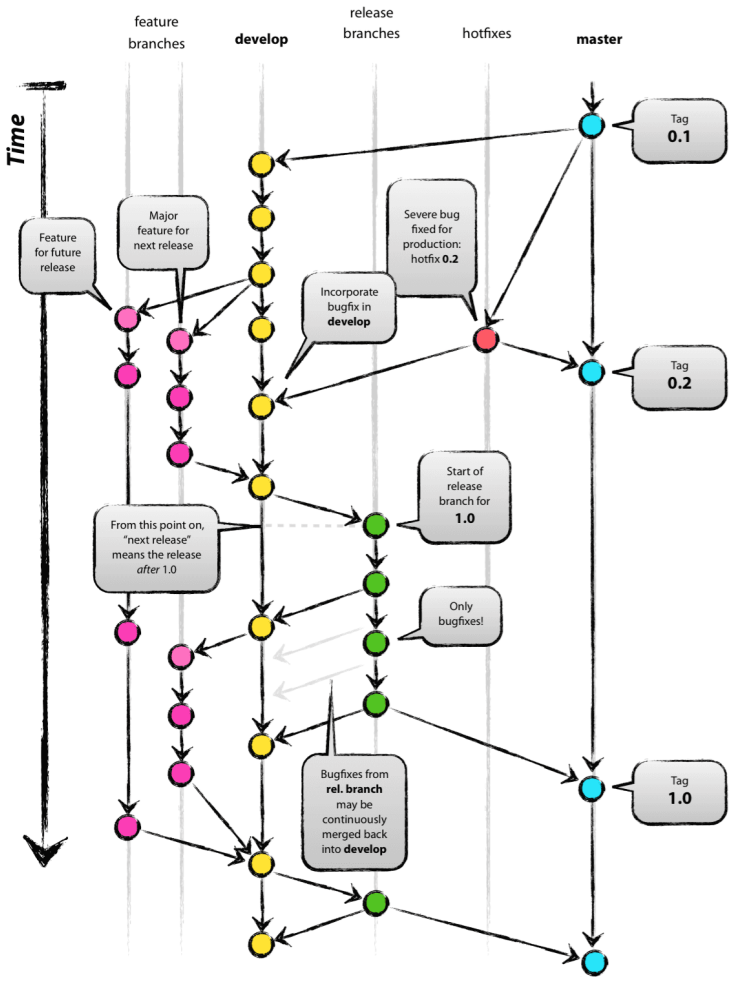


Figure 26: A successful Git branching model [83]

4.6 Final product

The embedded software consists of the main application executable, dynamic libraries, custom utility programs, configuration files and static files for the web interface, packaged into a single firmware upgrade file that can be installed on the Impinj Speedway Revolution R420 RFID reader.

The codebase that constitutes the application proper is around 5000 lines of C++ code, with a peak of over 13k LOC during the development. Constant refactoring and relentless removal of unneeded code helps in bringing down the complexity and line count. Nevertheless, 92% of all of the (mostly optional) functionality specified in Reader Protocol have been implemented. Disregarding strict adherence to the

specification has made it possible to provide all of the same basic functionality and even more with a more minimalistic API.

The software project taught many skills in software engineering, cross-platform development, industrial RFID applications, and perhaps most importantly – problem solving. Value of the software stems from the work invested into its development and the actual uses it has in industrial RFID solutions already delivered.

4.6.1 Requirements validation

Verification and validation are integral parts of the software life-cycle. They are processes for verifying that the product meets the requirements set for it. On the highest level, the reader application can be shown to be configurable by the user so that RFID events are being generated in the presence of tags and reports being sent over network connections.

The custom reader software is at least on par in terms of features with the competitor product it was initially designed to replace, and arguably outperforms it. Full quality performance analysis (QUPER) could be performed to make rigorous assertions, but only a hint of such a process is suggested here for brevity.

The QUPER model for roadmapping software quality requirements is originally proposed in [84]. The steps taken in the analysis are documented using templates like tables and views that are illustrated in Figure 27. The model puts quality requirements of the software in the correct perspective with the value that would result in meeting them at different levels.

For example, the time it takes for a user to tweak a configuration value and commit the change could be compared. Using ”our” software the user has to navigate to the settings page of the reader using a web browser on any platform, click a button or select an option from a dropdown widget. The command to the reader is sent asynchronously in the background taking immediate effect, excluding the 0.5 second rate-limiter actually implemented for the user’s convenience. Using the previously available solution provided by the 3rd party vendor, user has to open a Windows-only GUI program for connecting to the reader, open an XML configuration file that lists the Reader Protocol commands to be executed, find the relevant command and modify it in-place or write it from scratch, copy and paste the file into the utility and push a button for sending it to the reader, after which comes the most time-consuming part; the device has to be rebooted which takes very close to 60 seconds on average.

Following the QUPER model for analysis, the competitor product may offer utility value in the context of current market expectations, but the software discussed here certainly far exceeds the differentiation limit, possibly even the saturation limit after which there is no point in optimizing that particular feature.

Obviously one of the major points why the embedded application is considered in this thesis, is the added value that can be achieved with localization algorithms that could be integrated with the product. Clearly there exists little competition in that regard outside research projects and case-by-case tailored solutions. The currently missing features contain mainly implementations of localization methods, which is

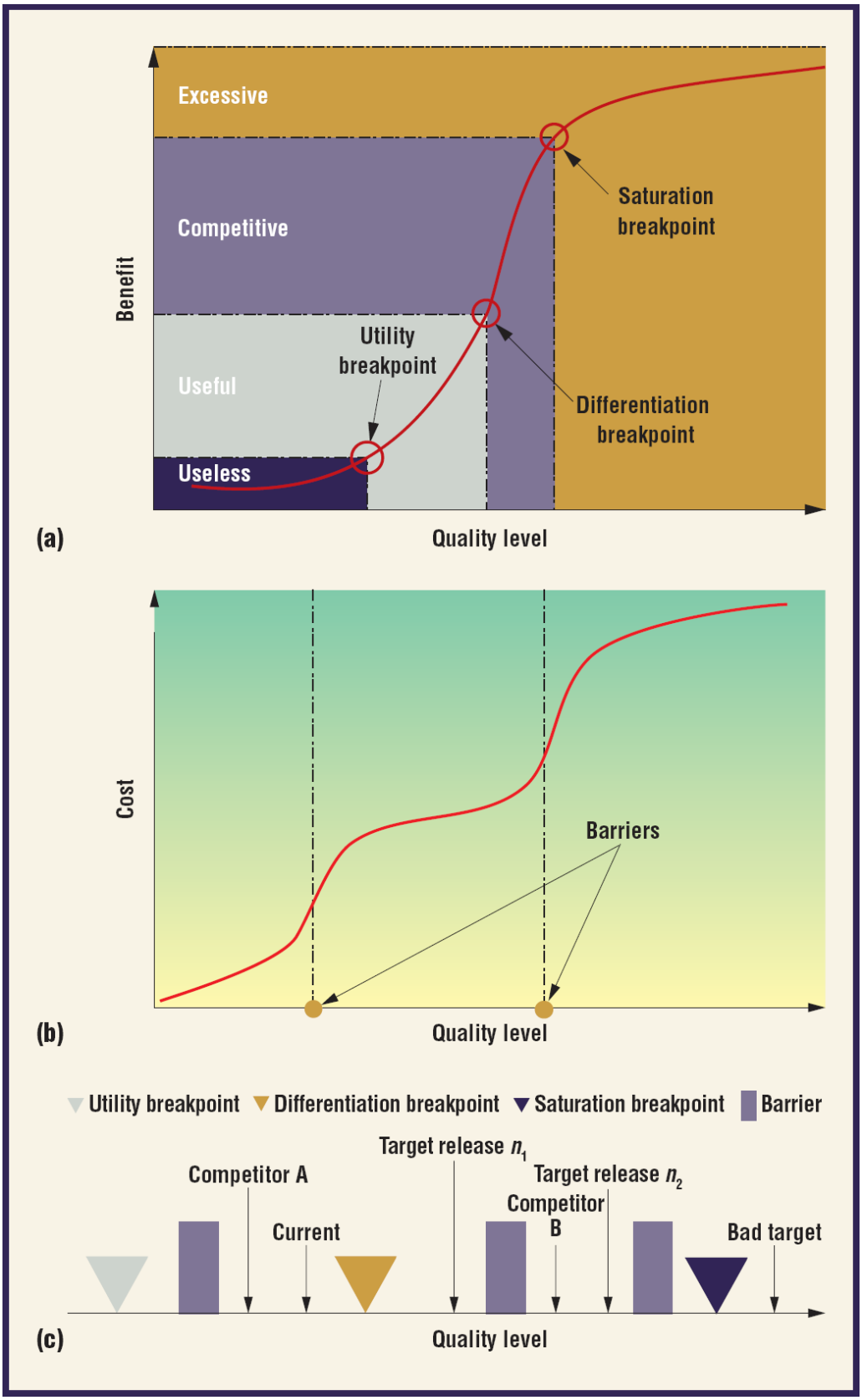


Figure 27: Views of the Quality Performance model [84]

subject to change after the completion of this study.

4.6.2 Extensibility

Partly due to the Reader Protocol object model implementation, the reader software is fairly open for extension. It should be noted that despite its flaws – specifically in the API and command formats – the object model has a reasonably good design. Composing the software from manageably small components makes it possible to add functionality independently from the existing structure.

A notable factor in itself is the integration of LuaJIT. New functionality built on top of the existing framework gets more streamlined using a scripting language. Other aspects that are not discussed in any more detail include the foreign function interface library of LuaJIT that actually ships with a built-in C parser that enables interfacing with C ABI directly, bypassing the Lua stack. This could be potentially used for developing native, dynamically-loaded shared library plugins, a feat which is not trivially done even in C++.

Features most prominently visible to a user can be implemented with client-side JavaScript code. Running scripts in the browser enables transferring some computational burden to the machine of the user, typically an order of magnitude more powerful in terms of computing power than the embedded device. Browser-based demo applications are also quick to develop for exhibitions and similar one-time events, and do not require internet connectivity when the content is shipped with the on-reader application.

4.6.3 User interface

The user interface for the reader application is implemented as a set of web pages served from the embedded HTTPS server. This approach has more advantages than one would care to enumerate. To begin with, the access to the interface is available from anywhere where there exists a routing to the network the reader is connected to, and the connections are secured using modern TLS cipher suites. A graphical user interface is often preferable for occasional reconfiguration and in cases where the user is not technically oriented or familiar with e.g. command line interfaces. The interface design attempts to follow the principle of least surpise – all similar functionality grouped together – no hidden options exist behind additional menus and yet, only the the necessary elements are visible.

Technologies used are standard HTML, CSS and JavaScript. Specifically, the amount of vanilla JavaScript is kept to a minimum and libraries such as jQuery and knockout.js are exploited. The dynamically updating and reactive interface uses a design pattern known as model-view-viewmodel (MVVM) implemented by knockout.js. A viewmodel is constructed in the browser from JSON serialization of the internal object model, which is bound to HTML elements displayed. Interactions with interface elements, like toggling a switch, update the viewmodel which reacts to changes and updates the interface if needed. Also, an asynchronous AJAX call with Lua command as its payload is sent to the reader to update the internal state. This way the changes seen and made by the user are always those that are in effect on

the reader. A screenshot of some of the settings configurable from the web pages are shown in Figure 28.

Actually typing Lua commands on a command line is also possible if needed. The look-and-feel of a traditional command line interface is implemented in HTML and JavaScript, so advanced operations are possible without ever leaving the page.

A novel view for displaying the RFID read events is implemented, which has proved to be convenient for testing RFID installations in the laboratory as well as in the field. The page displays read events as they happen and are being streamed from the reader practically in real-time. A screenshot of the tool is given in Figure 29. Tags are listed by their EPC separately for every antenna, and are being shown with a red background whenever a report arrives, fading back to white over the next half a second if the tag is not observed again. A notification sound is also played when receiving an event, which makes the tool useful in the field when there is necessarily no line of sight to the screen. The table can be filtered and sorted using one or more columns, in real-time.

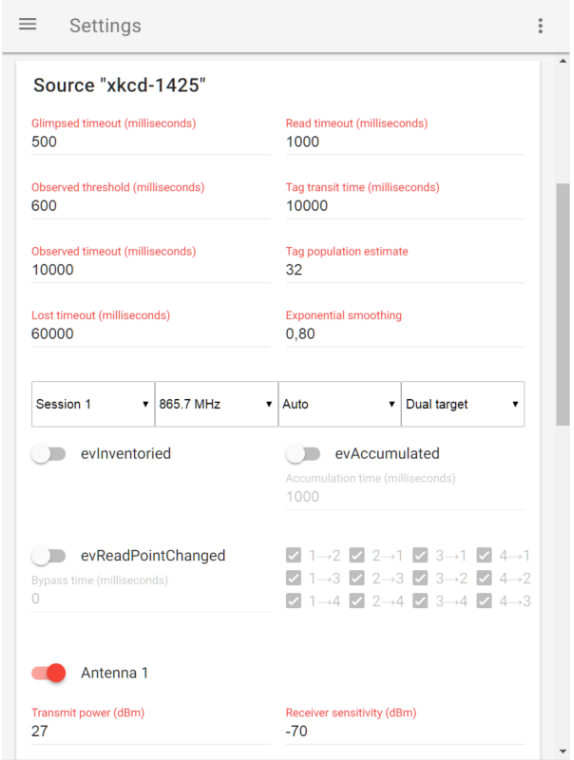


Figure 28: Screenshot of the the responsive web interface

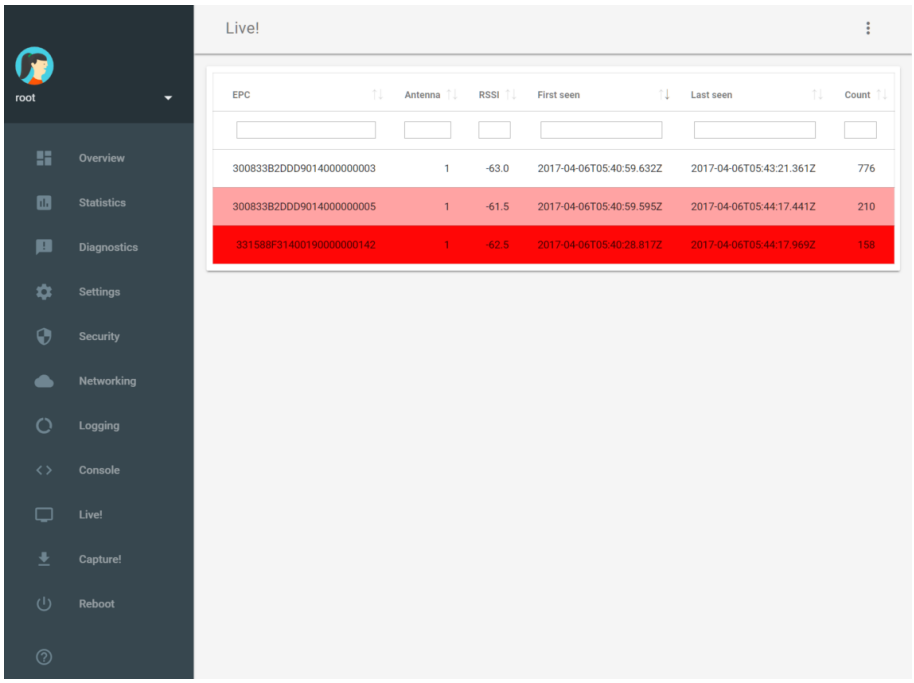


Figure 29: Screenshot of the the live display of read events

5 Empirical validation

The methods presented in Sections 3.4 and 3.5 are applied to practice in this section, with the results presented and discussed in Sections 6–7. The experiments conducted here follow quantitative methodology with an UHF RFID reader used for data acquisition to collect read reports from a Gen2 tag for indoor-localization. The tag is suspended in a way that enables running it with constant velocity in a repeatable manner hanging down from a wire. The experimentation is carried out in an indoor environment clear of obstacles, but without any additional precautions taken to control the propagation environment outside the immediate test setup.

The purpose for these experiments is to 1) show that the selected methods are applicable for tracking Gen2 tags using a commercial off-the-shelf reader, by trying to reproduce similar results as in prior art, 2) implicate the suitability of the developed application for data acquisition, and lastly 3) to be able to draw conclusions on the accuracy of the selected methods or factors limiting their real-world performance and uses in production.

The test setup and methodology are described below, followed by two test cases for 2D and 3D domains, respectively. A calibration step is introduced first to have an idea of the RSSI responses for each of the antennas, which can then be incorporated into a sensor-fusion model used in the tracking procedure.

5.1 Test setup and equipment

The experiments in this thesis were planned to be easily repeatable. An office-like laboratory environment was used to make the test area controlled in way that a line-of-sight from antennas to the target was provided at all times, limited only by the radiation patterns and sensitivity of the antennas. Other than that, no additional steps were taken to make the test area more optimal in terms of ambient noise or multipath propagation. It is also important that an accurate model for the tag dynamics can be provided in order to determine the real location at any given time.

The equipment consisted of Impinj Speedway Revolution R420 UHF RFID reader with four antenna ports, four antennas of varying designs, SMA coaxial connectors and high-quality coaxial cables. All RF hardware had been matched to have an impedance of 50 Ω. The reader had the custom application installed, described in Section 4.

The main idea with this experimentation to have an RFID tag follow a known linear trajectory with a constant velocity across several tests. This simplification was done 1) to simulate use cases the majority of which are expected to fall into the same category, i.e. where the tags move in an automated manner but where localization is required due to variance in their position, 2) to make the testing reliable, and 3) without assuming too much loss of generality due to the fact that one of the dynamics models used was the CWNA introduced in Section 3.5.1, which does not require targets to move linearly.

This was achieved using a tense 0.12 mm diameter fishing line suspended between two stiff springs hooked in the ceiling and the wall. The string was nearly diagonal

with respect to the walls in the room. By fixing the lower end as origin and measuring the location of the other end and all other relevant spatial coordinates relative to that point, the environment can be modeled to a reasonable accuracy. Specifically, the positions of the antennas were measured using a tape measure with one centimeter accuracy.

The transponder used in all of the experiments was a commercially available standard-issue Gen2 tag in a plastic card the size of a credit card, and was picked randomly from the batch. The tag was made to be suspended from the fishing line by two small metal hooks in two of its corners and slid along the line without friction. As the movement is then caused by the gravity of Earth, the tag would experience constant acceleration towards ground if not for the additional step taken. For this reason, a crumpled paper tissue was attached to the rear hook in the tag, i.e. the upper corner the tag was hanging from. The constant velocity is then achieved due to the drag that acts opposite to the movement and is dependent on the velocity.

The velocity the tag would achieve when released from the top end of the string was measured with neon-colored tape markers hanging from the ceiling and placed at one meter intervals next to the string. By capturing the movement on video 30 frames per second, the velocity could be estimated by frame-by-frame comparison. The velocity turned out to be surprisingly constant between repetitions at 1.67 meters per second along the fishing line.

5.2 Preliminary experiment

As a preliminary step for the tracking process, a set of RSSI values were measured over several discrete distances with all four antennas separately. Phase angle values and Doppler frequency shift values were also acquired for completeness. To draw any other conclusions than sample variance from the measured phase values the steps between the subsequent distances should be kept shorter than the 20 centimeter intervals used here. Likewise, the Doppler shift values are mainly of interest due to sample variance, if anything.

Four antennas were used in the experimentation; one commercial off-the-shelf patch antenna, two half-wave dipole antennas and a prototype antenna with extremely robust *metal* case designed for industrial use. The exact specifications are left undisclosed, but it is assumed that more optimal selection of antennas could be made so this is seen as a non-issue.

The measurement process was arranged as follows. The tag was temporarily fixed to maintain the same exact position when hanging from the string for the duration of the test, and the antenna under test was mounted on a wooden fixture with adhesive tape. The fixture was then set at a distance of 60 cm from the tag with the antenna pointed directly towards the tag. The reader was then set to inventory the tag for 5 seconds, after which the fixture was moved to cover distances 60–160 cm in 20 centimeter intervals and the process was repeated at each point. These six distance values were chosen so that a trend could be seen for the RSSI values as a function of distance. The set was limited to keep the preliminary step simple, and in fact only the values acquired at 1 meter reference distance is used later with the simple model

Table 3: Reader parameters used in experiments

Parameter	Value
Frequency	865.7 MHz
Session	0
Inventory search mode	Max Throughput, FM0
Search mode	Dual target
Tag population estimate	1
Tag transit time	10 000 ms
Transmit power	31.5 dBm
Receiver sensitivity	-80 dBm

given in Equations 6–7.

The reader air protocol parameters were configured for maximum throughput given the expected number of tags in the field to be one, and a Gen2 protocol-level filter was set to include only the tag in question. The center frequency in all test cases was set to channel 1 of the ETSI channel plan, i.e. at 865.7 MHz. The rest of the parameters were also kept constant and are given in Table 3.

In summary, RSSI values were measured and averaged for each distance and antenna, which were the only variables altered during the procedure. The experimentation was done offline, meaning that the reader wrote data into a log file, which was later extracted and analyzed. The information reader stores is given in Table 4. The resolution and deviation values are from the manufacturer's documentation [33]. The range and accuracy of the estimated Doppler frequency provided by the reader does not have unique values documented as they depend on many factors, including protocol parameters and signal-to-noise ratio (SNR). This approach does not prevent implementing or integrating the methods to the reader application for online production use in the future unless otherwise stated, but was chosen as the approach guaranteed to work under the schedule set for this thesis.

5.3 2D tracking with two antennas

A case for tracking a tag in two dimensions with two spatially distributed antennas is elaborated here. Motivations for this experiment are the use cases often found in industrial environments consisting of an RFID reader with fixed antennas and tags passing by e.g. when assets are being hauled through a gate, or when being transported on a conveyor system. Four alternative approaches are discussed in this context. For situations where it is adequate to recognize the relative order between tags, a simple comparison of the extreme values of RSSI and phase values

Table 4: Data fields recorded in experiments

Field	Comment
Timestamp	Microsecond accuracy
EPC	Constant
Channel	Constant (1)
RSSI	0.5 dB resolution, 1 dB standard deviation
Phase	0.0015 rad resolution, 0.1 rad standard deviation
Doppler frequency	
Antenna	

Table 5: Test parameters for 2D tracking

Parameter	Value
Repetitions	12
Antenna 1 model	Half-wave dipole
Antenna 2 model	Half-wave dipole
Antenna 1 position	$(0.0, 0.0)$
Antenna 2 position	$(0.8, 0.0)$
Tag start point	$(-1.3, 0.6)$
Tag end point	$(1.3, 0.6)$
Tag velocity	$(-1.67, 0.0)$

will do. On the other hand, more demanding computations have to be performed when information about distance or location is required. For these situations, the recursive distance estimation method together with the Unscented Kalman Filter, as well as an inverse synthetic aperture radar are applied here, both of which are detailed in Section 3.

The measurement process consisted of using the same constant parameters for the reader as in the preliminary step. Rest of the parameters are as given in Table 5. The coordinate system is transformed in a way to have the target trajectory as well as the antennas on the same two-dimensional plane to make up for a cleaner visualization. In reality, the antennas were placed on the same virtual line that was parallel to the string stretched across the room, i.e. in an angle with respect to the floor. From an analysis point of view this effectively simulates a conveyor system delivering the tag with a constant velocity pass-by.

As before, the data was collected on the reader for offline analysis. A single test run consisted of triggering the reader inventory operations, after which the tag was manually released to have it descend along the wire. The test run is repeated several times to introduce reliability for the method. In this setup the tag appeared in the FOV of the first antenna late enough for it to have reached its maximum velocity.

The UKF is the only method applied here requiring parametrization of its own. The introduced parameters are the initial state vector and its covariance matrix, as well as the covariance matrices for the process and measurement noise terms. Two different sets of parameters were used to assess the locating performance with correct and incorrect initial positions. It is customary to have the covariances be diagonal matrices when no empirical data have been acquired in order to decide otherwise. The covariances were chosen heuristically using scaled identities, and they are given as

$$\mathbf{P}_0 = 0.1 \cdot \mathrm{I}_6 \tag{51}$$

$$\mathbf{R} = 0.1 \cdot \mathrm{I}_2 \tag{52}$$

$$\mathbf{Q}_{\mathcal{V}} = 0.1 \cdot \mathrm{I}_4 \tag{53}$$

$$\mathbf{x}_{0,\mathrm{c}} = [x\ y\ b_1\ b_2\ v_x\ v_y]^\intercal = [-1.3\ 0.6\ 0\ 0\ 1.67\ 0]^\intercal \tag{54}$$

$$\mathbf{x}_{0,\mathrm{i}} = [x\ y\ b_1\ b_2\ v_x\ v_y]^\intercal = [-1.3\ 0.4\ 0\ 0\ 1.67\ 0]^\intercal \tag{55}$$

The values given in Eq. 54 and 55 were used for the correct and incorrect initial estimates, respectively. With these choices it was assumed that the velocities were relatively well-known as was the case when applying ISAR, too. The error in the initial position is given based on the reasoning that in the process being modeled the distance in the y-axis is of special interest, i.e. the distance to the tag at the time when it was directly in front of one of the antennas. Without loss of generality, antenna #1 was chosen as the reference. It was also assumed that the initial position varies only in the order of couple decimeters, which is also characterized by the initial covariance chosen above. Clearly there is no rationale for setting the initial biases to a non-zero value.

To describe the system dynamics, a discretized version of Equation 31 is used augmented with the biases b_i for each of the two antennas. The recursive distance

estimation equation $h_i(\mathbf{x}(t_n)) = |\mathbf{p} - \mathbf{a}_i| + b_i$ was used as a measurement model. The logarithmic path-loss model (Eq. 7) was converted for computing distance from a given RSSI measurement, and was used together with phase information to form a sensor-fusion system using the same measurement model for both quantities. The RSSI-based distance estimate was computed separately for each of the antennas using the reference values for 1 meter distance obtained in the calibration step. In the measurement covariance matrix both measurements were weighted equal, although RSSI was then expected to make the estimate contain more noise. This was done nevertheless to help the algorithm converge in general when only phase measurements are used, which was not self-evident given the positions of the antennas and the minimal overlapping in their interrogation zones.

Table 6: Test parameters for 3D tracking

Parameter	Value
Repetitions	9
Antenna 1 model	Half-wave dipole
Antenna 2 model	Prototype
Antenna 3 model	Patch
Antenna 4 model	Half-wave dipole
Antenna 1 position	$(0.0, 0.5, 0.2)$
Antenna 2 position	$(0.35, 1.2, -0.1)$
Antenna 3 position	$(0.47, 2.0, -0.1)$
Antenna 4 position	$(1.44, 0.36, 0.4)$
Tag start point	$(1.57, 2.1, 1.8)$
Tag end point	$(0.0, 0.0, 0.0)$
Tag velocity	$(-0.8244, -1.1027, -0.9452)$

5.4 3D tracking with four antennas

The second experiment follows the same framework as the previous one, although only UKF is considered in this three dimensional tracking problem. Other methods were left out because there is no single meaningful interpretation comparing simple RSSI and phase responses across arbitrarily distributed antennas, and ISAR implemented using grid search is computationally rather expensive and the results hard to visualize.

Motivation for this experiment was to assess the plausibility of locating UHF RFID tags in all three spatial dimensions with COTS hardware, to observe the possible issues and phenomena caused by the asynchronous measurements from different antennas, and to draw conclusions of whether high enough accuracy can be achieved for the method to actually be useful in production software.

The same setup was used for releasing the tag to its predetermined trajectory as described above. The same equipment was used in the same environment, excluding the number of antennas and their differing placement. The antennas were placed rather arbitrarily with no special care taken to optimize the configuration for best possible results, other than providing line-of-sight to the tag. The parameters given in Table 3 also apply here. In addition, a second list of parameters apply, given in Table 6.

Similar methodology was used when applying the CWNA model and the UKF as with the 2D problem domain. The filter parameters were selected following similar reasoning, and are given as

$$\mathbf{P}_0 = \mathrm{I}_{10} \tag{56}$$

$$\mathbf{R} = \mathrm{I}_2 \tag{57}$$

$$\mathbf{Q}_v = 0.1 \cdot \mathrm{I}_7 \tag{58}$$

The elements of covariance matrices have larger magnitudes than before due to the increased uncertainty in this experiment.

$$\mathbf{x}_{0,\mathrm{c}} = [1.57\ 2.1\ 1.8\ 0\ 0\ 0\ 0\ -0.8244\ -1.1027\ -0.9452]^\intercal \tag{59}$$

$$\mathbf{x}_{0,\mathrm{i}} = [2.00\ 2.0\ 2.0\ 0\ 0\ 0\ 0\ 0\ 0\ 0]^\intercal \tag{60}$$

The values given in Eq. 59 and 60 were used for the correct and incorrect initial estimates, respectively. The incorrect initial location was picked arbitrarily within the bounds set by the initial covariance, and is approximately half a meter away from the correct one. To see more clearly the performance under the influence of incorrect initial estimates, the velocity was set to a zero vector.

6 Results

The empirical evidence gathered in the experiments was dissected into batches of read reports which were then fed as an input to the localization algorithms. Position estimates for the RFID tag as a function of time were obtained from the output. The rest of the section is divided by the experiments performed and the method being used, starting with the calibration data. RSSI and phase measurements, ISAR images and estimated trajectories are presented graphically. The results are further discussed in the following section.

6.1 Calibration

In Figure 30 the measured RSSI values can be seen for each of the four antennas. The values vary from around -68 to -47 dBm across the distances ranging from 0.6 meters to 1.6 meters. As expected, the trend is visible as the received power gets lower with increasing distances. What is interesting though, is the rate of change for some of the antennas in the observed range; passive UHF RFID systems have a maximal read range in the order of 10 meters, and receiving as low values as these with such a short distances implies much shorter maximum read ranges. Also, the lines do not decrease monotonically, which is an indication of the presence of sub-optimalities.

These sub-optimal factors could include highly varying radiation patterns where even small changes in antenna orientation affect the power transfer, or there could be reflections and multipath propagation interfering with the signal. Also, the measurements may have been under the influence of the near-field of the antennas with the distances around 0.6 meters. The patch antenna has the highest directivity of the four, and the measurements reflect that. The prototype model with metal casing has apparent trade-offs, as the RSSI response is decreasing more rapidly than with the other three. No other conclusions are drawn from this data, as only the values from 1 meter distance are fitted to the propagation model and the rest of the data points are left for illustration purposes.

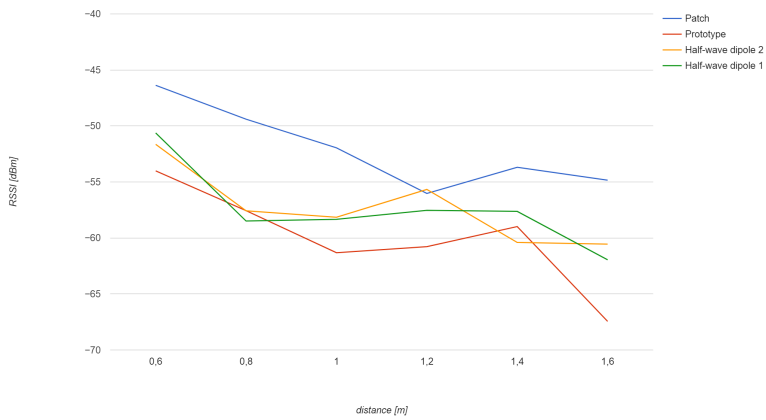


Figure 30: RSSI calibration data

6.2 RSSI and phase responses in 2D tracking

The RSSI and phase values obtained from all of the test runs of the 2D tracking experiment are given in Figures 31 and 32, respectively. In the experiment, the transponder moved along a straight line with a constant speed next to the two antennas while keeping a steady 60 cm distance from the imaginary line at which the antennas were located. In other words, both antennas were 60 centimeters away from the fishing line used to guide the tag. The phase values are unwrapped offline for clearer representation. In that way the values differ from the ones actually stored by the reader, but that does not affect the results otherwise.

The extreme values are clearly seen in the figures for both quantities. An interesting observation is to note that the maximum RSSI was measured before the accumulated phase difference would reach its minimum value in every single repetition of the experiment. The phenomenon is more clearly seen with the first antenna, but applies to both of them. It could have been assumed that the extrema would appear at the same point in time. Explaining factor could yet again be in the radiation patterns, or even in angle-dependent phase characteristics of the antennas. The latter could have been also assumed to be negligible, at least when compared to the variance of RSSI values in general.

The trails of approximately constant values visible in the graphs are due to the tag reaching the end of the line and stopping still. This would affect the results later, but the data is truncated to include only the portion where the tag is provably in motion. This course of action makes it easier to compute estimation errors and spares from including the discontinuity in the dynamics model of the tag when computing location estimates with ISAR.

The two antennas used here are of the same model. This fact is consistent with the measurements, as both of the antennas can be seen to have very similar distributions and equal levels in their RSSI values.

6.3 ISAR for 2D tracking

The inverse synthetic aperture radar using phase measurements detailed in Section 3.4 was applied here with a grid search method to obtain a location estimate for the tag at the time instant it was 60 centimeters away and directly in front of the first antenna. Because the trajectory is assumed to be known, modeled with a simple equation of constant velocity motion, the location can be determined afterwards at any given time by computing the offset from the estimated point using the time difference and known velocity. The point at which the tag was directly in front of the antenna was assumed to be the time at which the phase has its minimum value, and was taken into account when computing the errors between estimated and modeled (known) position.

One image with two spatial dimensions and one dimension for the computed logarithmic likelihood was generated for each of the test runs. The images were computed using a brute-force grid search method with one centimeter spatial resolution. The results can be illustrated as grayscale bitmap images with the pixel

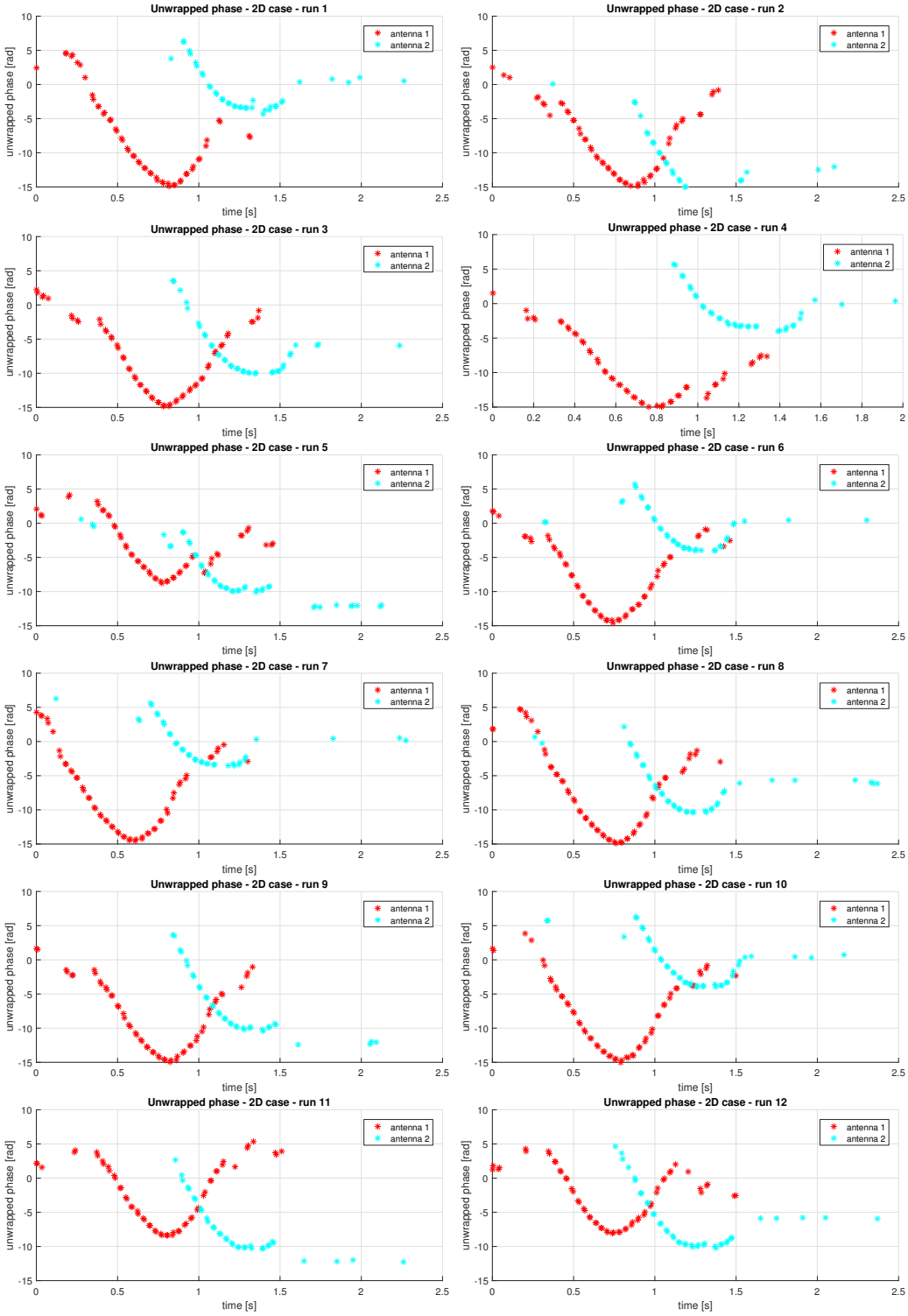


Figure 31: Phase measurements for the 2D case

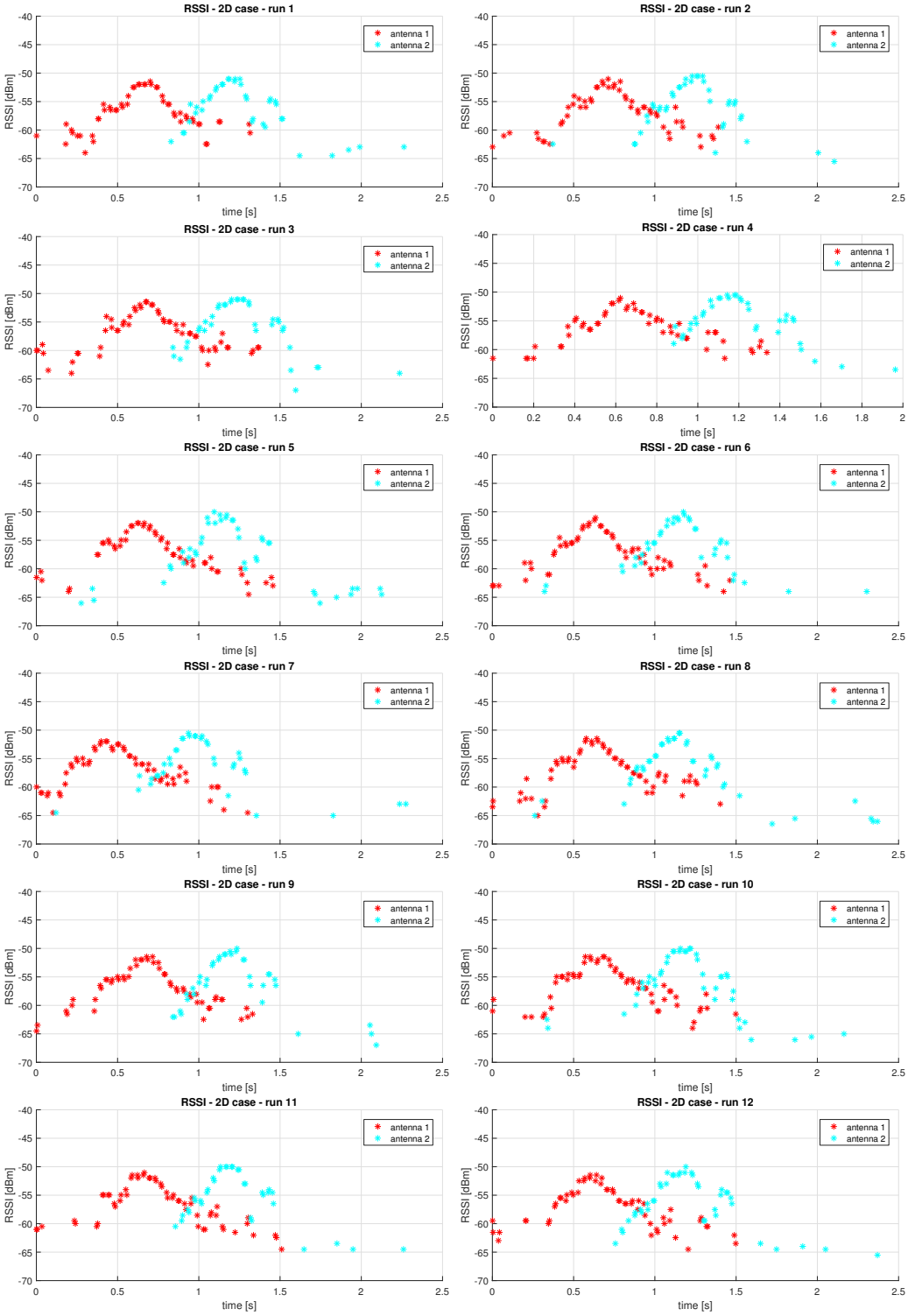


Figure 32: RSSI measurements for the 2D case

intensity illustrating the likelihood of the estimate (Figure 33) or as 3D heightmaps (Figure 34), with color-coded intensity for visualization purposes as well. The ISAR bitmap images for all of the test runs are given in Appendix A for the sake of completeness.

The image dimensions correspond to the two spatial dimensions of the test area with the antennas located at the bottom with spatial coordinates $(0.0, 0.0)$ and $(0.0, 0.8)$. The image covers an area with the x-axis as the horizontal dimension ranging from -0.5 to 1.5 meters, and y-axis as the vertical dimension ranging from 0 to 1 meter. The point of reference in front of the first antenna has coordinates $(0, 0.6)$ and is visible in the images, indicated with a red marker. The estimated location is located at the maximum intesity value and is marked with a black square, the center corresponding precisely with the estimated location.

As a measure of performance, the root mean squared error (RMSE) value is used, computed from the errors of all of the test runs. The location estimation error of a single test run is defined to be the distance between estimated location and the reference point. The estimated locations and the resulting errors are given in Table 7. The table presents a clear view of estimation accuracy in the order of 7 centimeters, which is a fairly decent result, although the main lobe visible in Figure 33 indicates the optimum might be highly sensitive to changes along the formation.

Table 7: Location estimation results with ISAR

n	x	y	error
1	-0.02	0.67	0.0728
2	-0.02	0.65	0.0539
3	-0.05	0.55	0.0707
4	-0.01	0.67	0.0707
5	-0.05	0.58	0.0539
6	-0.01	0.52	0.0806
7	-0.02	0.71	0.1118
8	-0.03	0.63	0.0424
9	0.04	0.51	0.0985
10	-0.02	0.66	0.0632
11	-0.02	0.65	0.0539
12	-0.03	0.54	0.0671
average	-0.02	0.61	0.0700
reference	0.00	0.60	
RMSE			0.0725

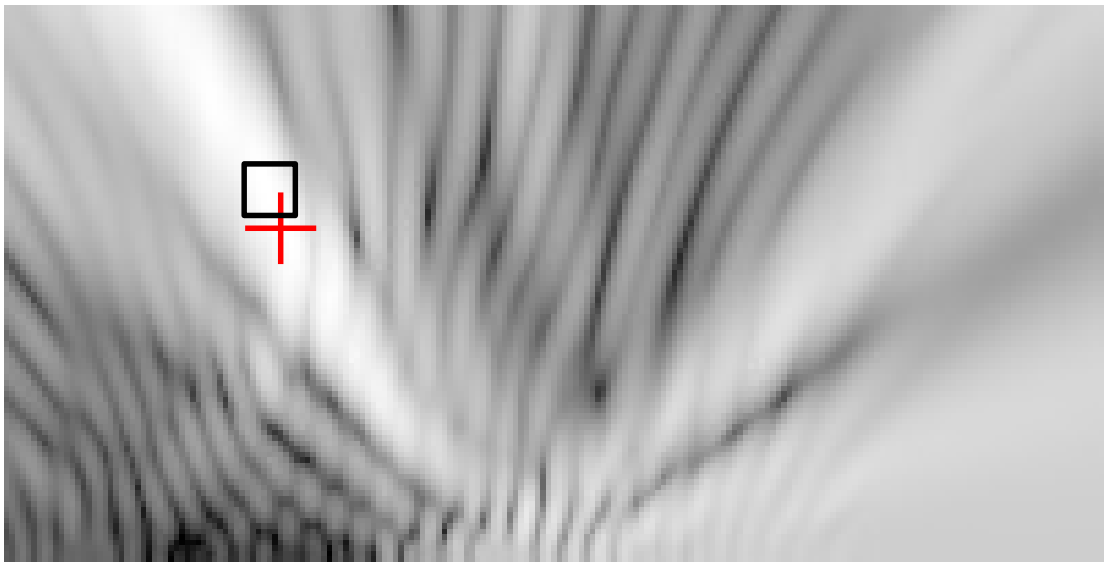


Figure 33: 2D tracking with ISAR

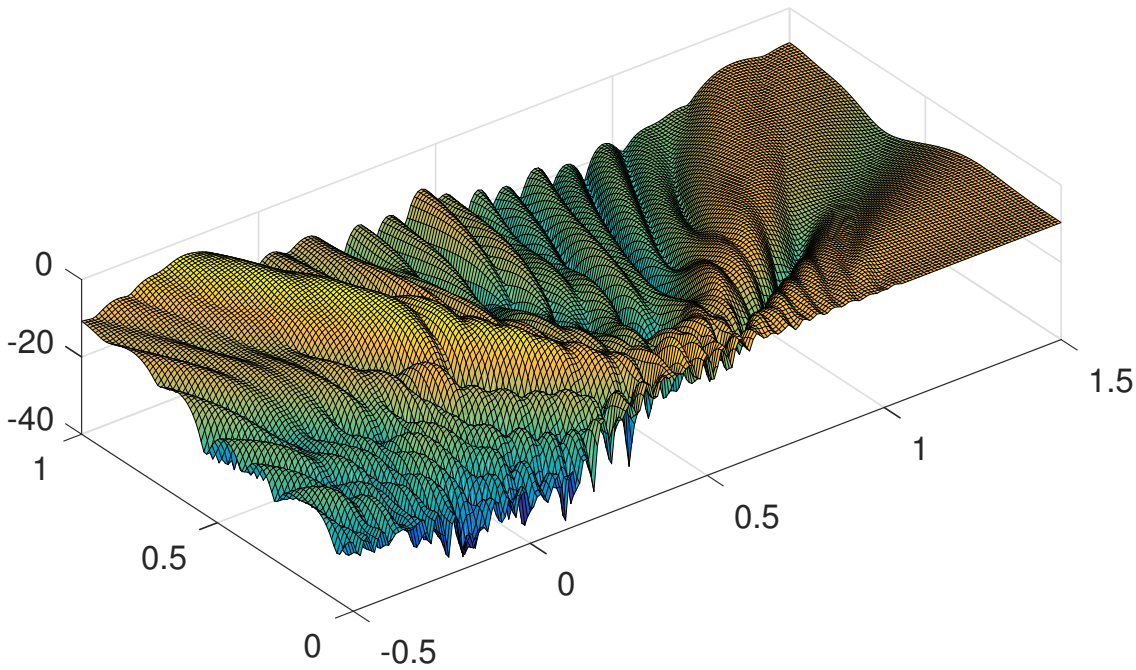


Figure 34: ISAR image as heightmap

6.4 UKF for 2D tracking

Results from the application of UKF for the nonlinear filtering problem in two dimensions are presented here. Using the CWNA as the process model of choice and RSSI and phase measurements as inputs, the filter was applied to obtain recursive location estimates as a function of time. Only results from a single (first) test run are given in graphical form to save paper – all the resulting figures would be very comparable to the ones included here. RMSE values for all of the test runs are given.

The trajectory given by the successive states of the filter using the correct and incorrect initial location estimates are plotted in Figures 37 and 38, respectively. The performance is again evaluated using RMSE, but this time the error is computed for each state. The error for the location is defined here as the shortest distance from the estimated location to the line through which the tag moves.

An alternative way for calculating the error could be to use the actual location of the tag at the time step as the reference, but that would result in quickly accumulating errors from early states if the filter were to converge slowly. Computing the shortest distance to a line is also more straightforward, and serves well for the purposes of this experiment with the distance to the tag in front of the first antenna being mainly of interest. The RMSE values for both correct and incorrect initial state are given in Figures 35–36.

The estimated location is marked in Figures 37–38 corresponding to the time instances the tag was inventoried. The color of the marker corresponds to the antenna responsible for that particular read event. A set of lines emanating from the antenna locations are drawn with similar colors as before, the length of which correspond to the distance estimated using RSSI measurements. The rays are drawn in a way that they intersect with the real trajectory at the time of inventory. Again, the real trajectory is determined by the time the unwrapped phase angle has its minimum value with the tag in front of the first antenna.

From the figures it can be observed that the reader has optimized its inventory performance so that the tag is inventoried twice in a row using the same antenna and the gap between pairs of closely positioned lines is the result of reader switching antennas. This observation is more evident when looking at the recorded events, but the amount of raw data is too large to be included here.

An aspect the error values in Figure 36 are not able to provide is how well the filter converges from the initially incorrect state towards the end. The filter can be thought of as having more information available when reports from the second antenna start coming in, and the filter improved its performance towards the end, as seen in Figure 38. The additional information can be thought of as being partly attributed to noise by the filter as the amount of correction is not enough to make the estimate match the real value precisely.

It should be noted that the tracking problem using phase measurements $\mod 2\pi$ does not have unique solutions using only one antenna, which is the case for the first half of each test run – the interrogation zones of the antennas appear to have only little overlap. It should also be remembered that the initial velocity was assumed to be known in every configuration, which limits the generalizability of these results.

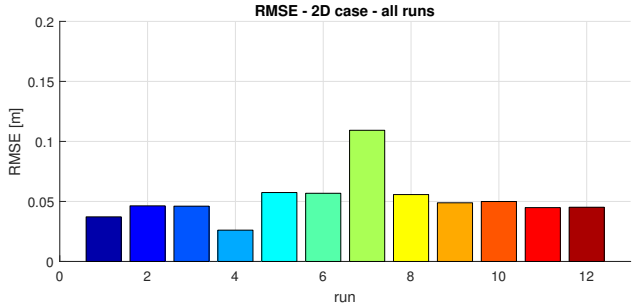


Figure 35: RMSE of UKF with correct initial estimate in 2D

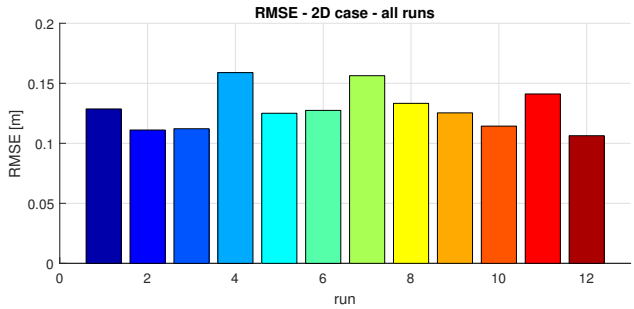


Figure 36: RMSE of UKF with incorrect initial estimate in 2D

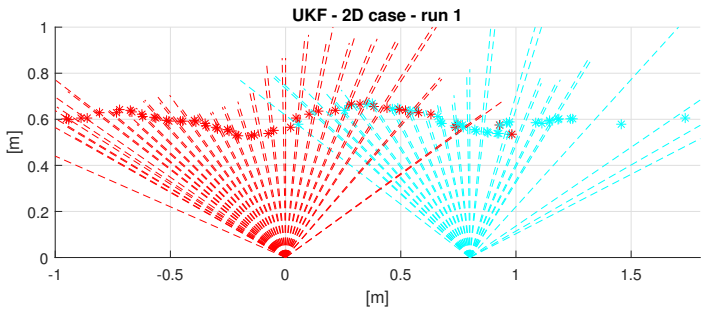


Figure 37: CWNA, UKF and correct initial estimate in 2D

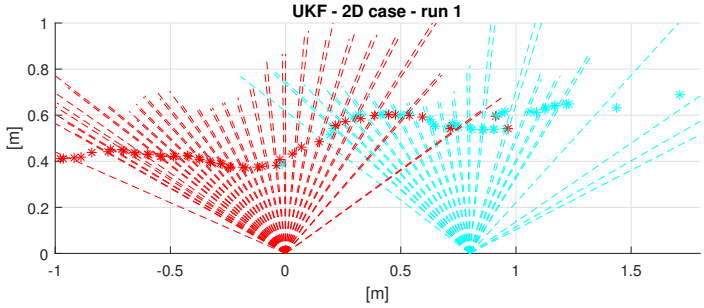


Figure 38: CWNA, UKF and incorrect initial estimate in 2D

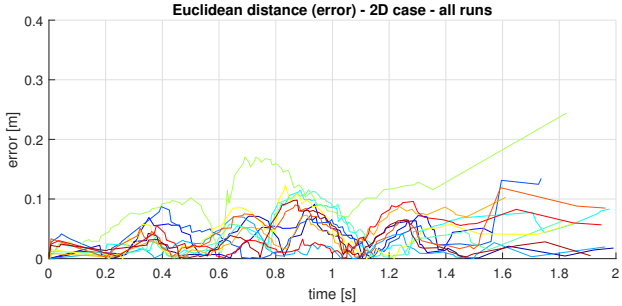


Figure 39: Errors of UKF with correct initial estimate in 2D

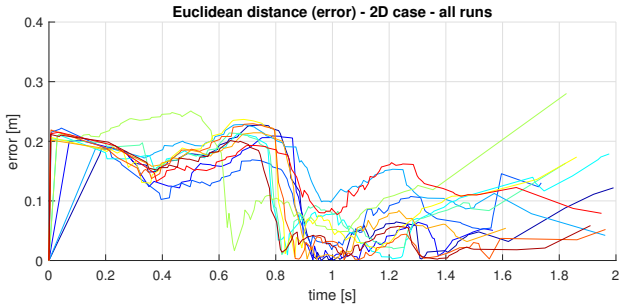


Figure 40: Errors of UKF with incorrect initial estimate in 2D

6.5 UKF for 3D tracking

The results for the 3D tracking experiment with Unscented Kalman Filter are presented below, using the same approach as in the two-dimensional setup. Estimated trajectories for correct and incorrect initial states are plotted in Figures 42 and 43, respectively. Errors are computed using Euclidean distances and RMSE as before, and are displayed if Figures 44–47.

Locations of the antennas are illustrated in Figure 41 with the RSSI-based distance estimates. Clearly the antenna placement is adequate to keep the tag in FOV of the reader at all times, but antennas #1 and #2 (red and green rays in the figure) are located poorly and do not provide much information until very late.

Considering the fact that in this experiment the incorrect value for the initial state included a velocity of zero, the final state is remarkably close to to origin. Both sets of parameters result in a very similar performance: the mean RMSE values are 0.382 and 0.396 for correct and incorrect initial values, respectively.

Judging by the error graphs in Figures 46–47, the accuracy gets better with increasing number of antennas providing measurements to the filter, but not as efficiently as in the two-dimensional case. The most likely reason for that is the way the antennas were placed. More notable improvements in localization accuracy could be expected with proper antenna placement and by guaranteeing a constant stream of measurements from all four antennas.

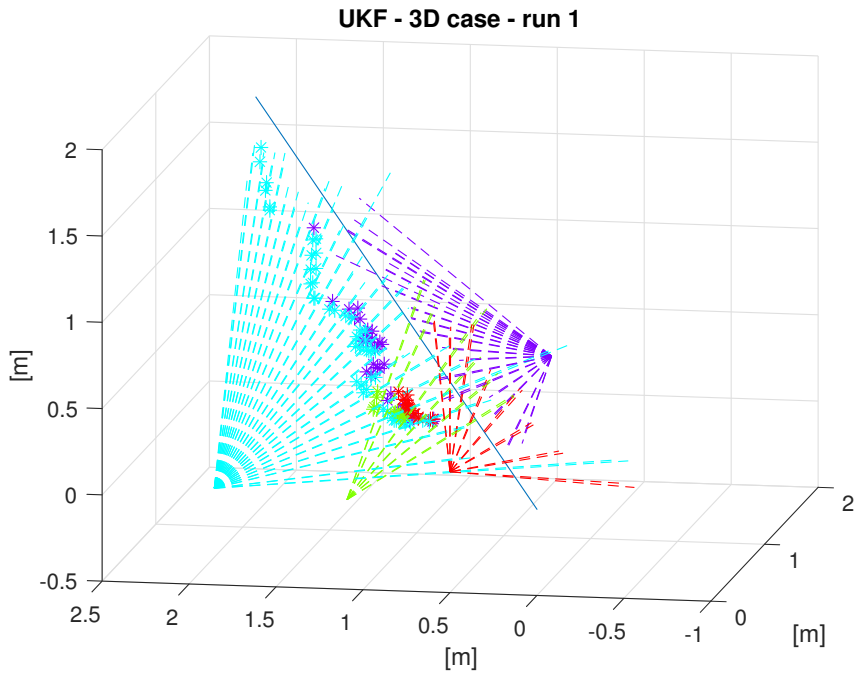


Figure 41: Antenna positions and RSSI distance estimates

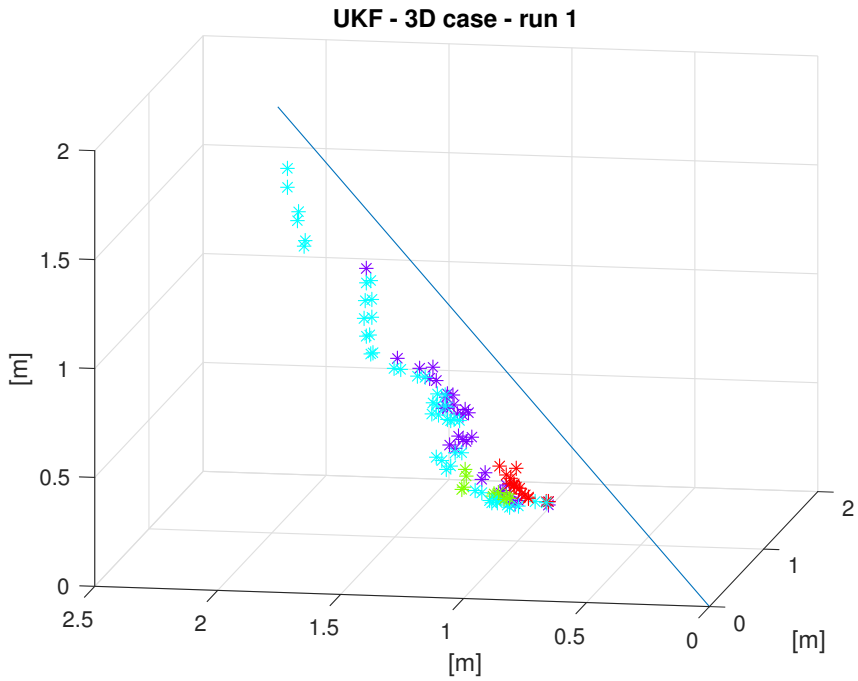


Figure 42: CWNA, UKF and correct initial estimate in 3D

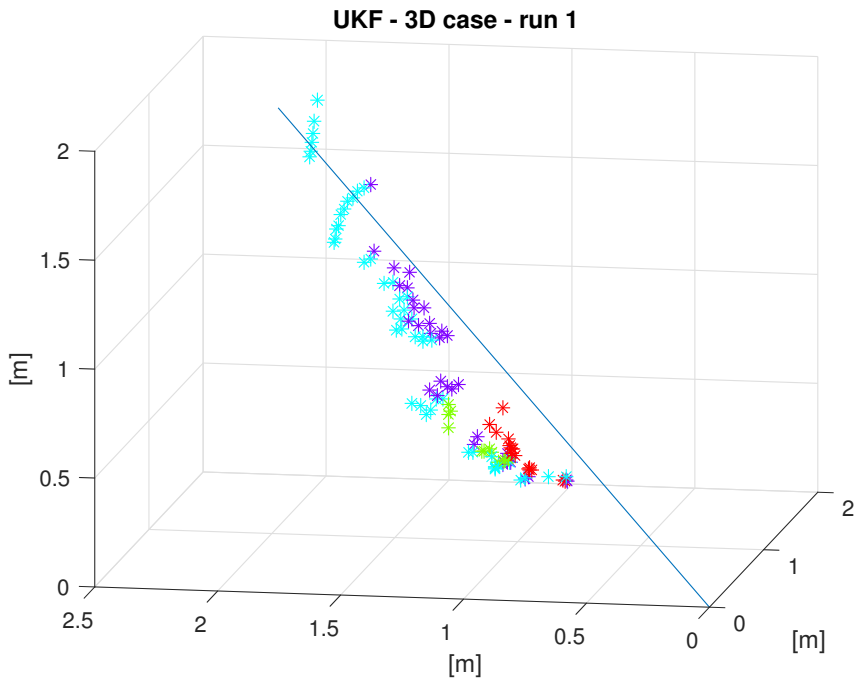


Figure 43: CWNA, UKF and incorrect initial estimate in 3D

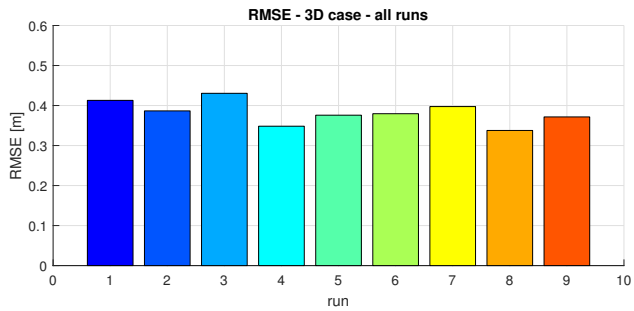


Figure 44: RMSE of UKF with correct initial estimate in 3D

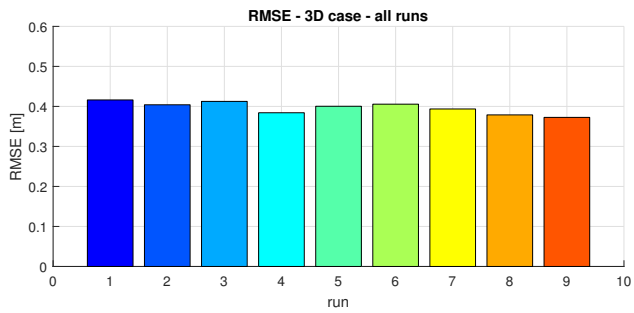


Figure 45: RMSE of UKF with incorrect initial estimate in 3D

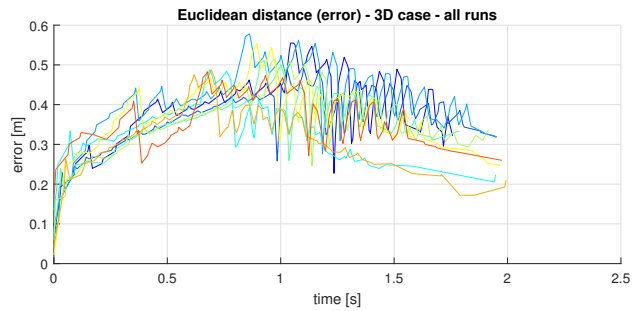


Figure 46: Errors of UKF with correct initial estimate in 3D

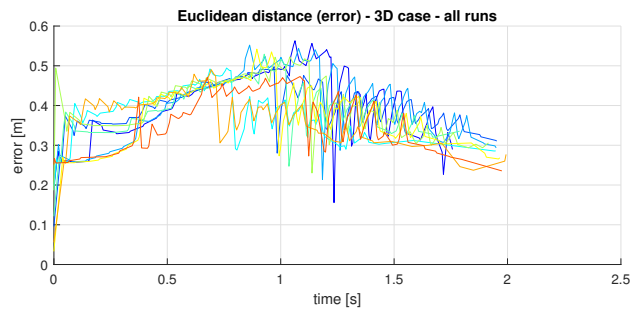


Figure 47: Errors of UKF with incorrect initial estimate in 3D

7 Discussion

The results indicate that the experimentation was conducted in a repeatable manner and no anomalies of unclear origin were observed. Most of the findings can be regarded informative at least in giving directions for future development.

Treating RSSI measurements more rigorously and calibrating the loss exponent as well could improve the localization accuracy in future studies. The actual benefit remains still questionable, as signal parameters and impedance matching vary with the changes in the environment. The main benefit in incorporating RSSI measurements is using them as a proximity information and thus preventing the location estimate from diverging outside the area of interest.

The RSSI and phase measurements were shown to have only little variation in their general patterns. Nothing can be said about the precision with which several tags could be spatially distinguished based on the experiment. In principle having two antennas next to each other simulates a situation with one antenna and two tags placed apart at an equivalent distance, and by altering the distance some information about the resolution could be determined. The information provided by the experiment is also valuable when assessing systems that have been deployed in production. Specifically, a use case already exists in which tags attached to containers have to be identified as they move on a conveyor.

In this particular example, the containers come in two piles on a pallet, and the problem is to determine which tags belong to which pile. The currently employed solution is based on *ad hoc* timestamp comparison logic, but as a matter of fact another reader is installed next to the active read point but does not take part in the decision-making. It is still used for inventorying tags passing by at discrete time intervals. Even at a relatively coarse sampling interval the maximum RSSI value could be used to determine the relative ordering. The point being, that the results of the first experiment give further trust in the method, and propose an idea that has now been implemented in the reader application – a custom field that keeps track of the time of the maximum RSSI value observed for each tag as long as the tag stays in its observed state as per the Reader Protocol event generation model. For use cases that meet the certain assumptions, the new feature can be used directly for the aforementioned purpose without requiring the reader to report multiple events containing RSSI values.

Usage of phase values for the same purpose has been recently studied in [49]. The applicability is again limited by the velocity of the target and the maximum read rate of the reader. Due to the optimized configuration used in the experiment, a read rate of over 100 reports per second can be achieved for a single tag, which should allow velocities up to 4 meters per second.

Approximately 7 centimeter accuracy was achieved using ISAR for 2D localization. The result is not directly comparable to those presented in prior art due to the fact that it is hard to estimate whether a difference in the order of few centimeters in accuracy is significant when the problem domain has dimensions in the order of few meters at maximum. The accuracy achieved in some previous studies vary from 2 to 17 cm [56,57], and in that context the results presented in this thesis are well-founded

and reproducible. The computational requirements for performing unguided grid search to form the results are too high for the method to be implemented in the reader application, as expected. Even using a powerful workstation for computing the solution, it is questionable whether a naive implementation fulfills real-time requirements. Nevertheless, grid search belongs to the category of *embarrassingly parallel* problems, and using a graphics processing unit for the task should provide a significant speedup.

Under the same a priori assumptions that were set for ISAR, the UKF was observed to perform reasonably well in two dimensions, even with a challenging antenna configuration. Great care has to be taken with positioning and orienting the antennas for the method to be usable for general-purpose localization . With the same assumptions and with an accurate process model, the UKF is certainly a feasible choice for localization using RSSI and phase measurements. The computational requirements of UKF are much lower than those of ISAR, and its recursive nature makes it suitable for real-time problems.

Localization accuracy for the combination of CWNA process model, Extended Kalman Filter and Rauch-Tung-Striebel smoother was reported in [59] to be around 2 cm, with a test area of 3 by 3 meters. The results obtained here are not within the same order magnitude, but when accounting for the fact that in [59] the researchers performed the measurements in an anechoic chamber using an RFID reader capable of an average read-out of 400 read reports per seconds, or 100 for each antenna, as well as carefully placing the antennas, the experiment can be still viewed as being successful. Therefore, the method can be tentatively claimed as feasible for production, depending on the requirements set for the localization accuracy.

The usability of UKF together with the proposed measurement model has its limitations. Particularly, using phase measurements limit the maximum velocities and minimum read rates as have been repeated. Fusing RSSI measurements into the model certainly prevents the filter from diverging, but comes with a trade-off regarding the accuracy that can be achieved. Also, radio wave propagation always suffers in real environments and the surroundings always pose limitations when implementing radiolocating systems.

Finally, the reader device together with the embedded software performed well enough for the purposes of these experiments. The device and the software are capable enough for acquiring measurements at a rate that is suitable for locating RFID tags.

8 Conclusions

Real-time locating systems and methods for locating targets using radio signals have been discussed in this Master's thesis. UHF RFID technology was reviewed in detail, and an embedded software for an industrial RFID reader was presented.

The primary objectives of this thesis were to determine methods suitable for locating passive UHF RFID tags and to develop an embedded software that can be used when applying the methods in practice. In conclusion, methods using RSSI and phase measurements for ranging have been used effectively in literature, and results indicating their applicability were obtained in this study.

Commercial off-the-shelf hardware has its limitations, but provides enough measurements so that localization can be made feasible. Recursive estimation algorithms are well-suited for real-time systems. The inverse synthetic aperture radar could be used when computations can be done offline or with a more powerful hardware than what is included in the embedded device.

The software is widely configurable for various use cases and has a modular design based on EPCglobal Reader Protocol object model. Remote control as well as asynchronous notifications are supported over common network protocols like TCP/IP and HTTP. The reader software has been successfully deployed to production multiple times, and it has provided users the means to reconfigure it on-demand. Network and software security issues were accounted with good security practices, including authenticating all inbound requests and taking the necessary steps to provide a safe way to expose an API that could be used by machines and humans alike.

Further study is needed so that multi-frequency measurements for locating RFID tags can be incorporated into the localization process. After implementing the UKF for locating RFID tags on the reader device, some research is needed so that a calibration and configuration procedure can be provided for the user and so that the method can be applied in different scenarios.

References

[1] S. Tyagi, M. Ayoub Khan, and A. Q. Ansari, "RFID Data Management," in *Radio Frequency Identification Fundamentals and Applications Bringing Research to Practice*, Cristina Trucu, Ed. InTech, February 2010, DOI:10.5772/8001.

[2] M. Fera, R. Iannone, V. Mancini, M. M.Schiraldi, and P. Scotti, "Economic Evaluation of RFID Technology in the Production Environment," *International Journal of Engineering Business Management*, vol. 5, no. 39, June 2013, DOI:10.5772/56750.

[3] L. Ruiz-Garcia and L. Lunadei, "Monitoring Cold Chain Logistics by Means of RFID," in *Sustainable Radio Frequency Identification Solutions*, Cristina Turcu, Ed. InTech, February 2010, DOI:10.5772/8006.

[4] S. Särkkä, V. Viikari, and K. Jaakkola, "RFID-based butterfly location sensing system," in *2014 22nd European Signal Processing Conference (EUSIPCO)*, Lisbon, Portugal, September 2014, pp. 2045–2049.

[5] "Information technology – Real-time locating systems (RTLS) – Part 1: Application program interface (API)," ISO/IEC 24730-1:2014, February 2014, 2nd edition.

[6] Y. Ma, L. Zhou, K. Liu, and J. Wang, "Iterative Phase Reconstruction and Weighted Localization Algorithm for Indoor RFID-Based Localization in NLOS Environment," *IEEE Sensors Journal*, vol. 14, no. 2, pp. 597–611, February 2014, DOI:10.1109/JSEN.2013.2286220.

[7] P. Nikitin, R. Martinez, S. Ramamurthy, H. Leland, G. Spiess, and K. V. S. Rao, "Phase based spatial identification of UHF RFID tags," in *IEEE International Conference on RFID*, Orlando, FL, USA, April 2010, pp. 102–109, DOI:10.1109/RFID.2010.5467253.

[8] D. I. Tapia, R. S. Alonso, S. Rodriguez, F. de la Prieta, J. M. Corchado, and J. Bajo, "Implementing a real-time locating system based on wireless sensor networks and artificial neural networks to mitigate the multipath effect," in *14th International Conference on Information Fusion*, Chicago, IL, USA, July 2011, pp. 1–8.

[9] H. Liu, H. Darabi, P. Banerjee, and J. Liu, "Survey of Wireless Indoor Positioning Techniques and Systems," *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, vol. 37, no. 6, pp. 1067–1080, November 2007, DOI:10.1109/TSMCC.2007.905750.

[10] K. Finkenzeller, *RFID Handbook. Fundamentals and Applications in Contactless Smart Cards, Radio Frequency Identification and Near-Field Communication*, 3rd ed. John Wiley & Sons, Ltd., Wiltshire, United Kingdom, 2010.

[11] M.-S. Jian, "RFID System Integration and Application Examples," in *Radio Frequency Identification Fundamentals and Applications Bringing Research to Practice*, Cristina Turcu, Ed. InTech, February 2010, DOI:10.5772/7999.

[12] K. Arora, H. Mallinson, A. Kulkarni, J. Brusey, and D. McFarlane, "The Practical Feasibility of Using RFID in a Metal Environment," in *IEEE Wireless Communications and Networking Conference*, Kowloon, China, March 2007, pp. 1679–1683, DOI:10.1109/WCNC.2007.316.

[13] A. Mercer, R. James, G. Bennett, P. Patel, C. Johnston, and J. Cai, "Performance testing of RFID systems with RF-harsh materials," in *IEEE International Conference on RFID-Technologies and Applications*, Sitges, Spain, September 2011, pp. 537–543, DOI:10.1109/RFID-TA.2011.6068597.

[14] "RFID. Part 1, Guide, introduction to technology," SFS handbook 301-1, [in Finnish], March 2010, 1st edition, Suomen Standardoimisliitto SFS Ry, Helsinki.

[15] "Specification for RFID Air Interface. EPC Radio-Frequency Identity Protocols Class-1 Generation-2 UHF RFID Protocol for Communications at 860 MHz – 960 MHz," GS1 EPCglobal Class-1 Generation-2 Specification, 2004.

[16] M. C. O'Connor, "Gen 2 EPC protocol approved as ISO 18000-6C," *RFID Journal*, vol. 1, no. 11, 2006.

[17] A. Papapostolou and H. Chaouchi, "The Applicability of RFID for Indoor Localization," in *Deploying RFID - Challenges, Solutions, and Open Issues*, Cristina Turcu, Ed. InTech, August 2011, DOI:10.5772/17968.

[18] D. Zhang, L. T. Yang, M. Chen, S. Zhao, M. Guo, and Y. Zhang, "Real-Time Locating Systems Using Active RFID for Internet of Things," *IEEE Systems Journal*, vol. 10, no. 3, pp. 1226–1235, September 2016, DOI:10.1109/JSYST.2014.2346625.

[19] M. Bouet and A. dos Santos, "RFID tags: Positioning principles and localization techniques," in *1st IFIP Wireless Days*, Dubai, United Arab Emirates, November 2008, pp. 1–5, DOI:10.1109/WD.2008.4812905.

[20] S. Shao and R. Burkholder, "Passive UHF RFID tag localization using reader antenna spatial diversity," in *IEEE International Conference on Wireless Information Technology and Systems*, Maui, HI, USA, November 2012, pp. 1–4, DOI:10.1109/ICWITS.2012.6417767.

[21] H. Fukada, T. Mori, H. Noguchi, and T. Sato, "Use of Active RFID and Environment-Embedded Sensors for Indoor Object Location Estimation," in *Deploying RFID - Challenges, Solutions, and Open Issues*, Cristina Turcu, Ed. InTech, August 2011, DOI:10.5772/17498.

[22] Y. Zhang, M. G. Amin, and S. Kaushik, "Localization and tracking of passive RFID tags based on direction estimation," *International Journal of Antennas and Propagation*, vol. 2007, Hindawi Publishing Corporation, DOI:10.1155/2007/17426.

[23] G. Li, D. Arnitz, R. Ebelt, U. Muehlmann, K. Witrisal, and M. Vossiek, "Bandwidth dependence of CW ranging to UHF RFID tags in severe multipath environments," in *IEEE International Conference on RFID*, Orlando, FL, USA, February 2011, pp. 19–25, DOI:10.1109/RFID.2011.5764631.

[24] S. Azzouzi, M. Cremer, U. Dettmar, R. Kronberger, and T. Knie, "New measurement results for the localization of UHF RFID transponders using an Angle of Arrival (AoA) approach," in *IEEE International Conference on RFID*, Orlando, FL, USA, April 2011, pp. 91–97, DOI:10.1109/RFID.2011.5764607.

[25] A. Bekkali, H. Sanson, and M. Matsumoto, "RFID Indoor Positioning Based on Probabilistic RFID Map and Kalman Filtering," in *3rd IEEE International Conference on Wireless and Mobile Computing, Networking and Communications*, White Plains, NY, USA, October 2007, pp. 21–21, DOI:10.1109/WIMOB.2007.4390815.

[26] C. Hekimian-Williams, B. Grant, X. Liu, Z. Zhang, and P. Kumar, "Accurate localization of RFID tags using phase difference," in *IEEE International Conference on RFID*, Orlando, FL, USA, April 2010, pp. 89–96, DOI:10.1109/RFID.2010.5467268.

[27] "IEEE Standard for Low-Rate Wireless Networks," *IEEE Std 802.15.4-2015 (Revision of IEEE Std 802.15.4-2011)*, pp. 1–709, April 2016, DOI:10.1109/IEEESTD.2016.7460875.

[28] Buratti, Chiara and Conti, Andrea and Dardari, Davide and Verdone, Roberto, "An overview on wireless sensor networks technology and evolution," *Sensors*, vol. 9, no. 9, pp. 6869–6896, 2009, DOI:10.3390/s90906869.

[29] N. Baker, "ZigBee and Bluetooth strengths and weaknesses for industrial applications," *Computing Control Engineering Journal*, vol. 16, no. 2, pp. 20–25, April 2005, DOI:10.1049/cce:20050204.

[30] "EPC Tag Data Standard," GS1 EPCglobal, March 2017, Release 1.10, Ratified Standard.

[31] P. V. Nikitin and K. V. S. Rao, "Antennas and Propagation in UHF RFID Systems," in *2008 IEEE International Conference on RFID*, Las Vegas, NV, USA, April 2008, pp. 277–288, DOI:10.1109/RFID.2008.4519368.

[32] X. Huang, M. Barralet, and D. Sharma, "Behaviors of Antenna Polarization for RSSI Location Identification," in *International Conference on Networks Security, Wireless Communications and Trusted Computing*, vol. 1, Wuhan, Hubei, China, February 2009, pp. 151–154, DOI:10.1109/NSWCTC.2009.49.

[33] *Low Level User Data Support*, Impinj, Inc. Seattle, WA, USA, February 26, 2013, revision 3.0.

[34] J. Yu, K. H. Liu, and P. Luo, "A mobile RFID localization algorithm based on instantaneous frequency estimation," in *2011 6th International Conference on Computer Science Education (ICCSE)*, Singapore, Singapore, August 2011, pp. 525–530, DOI:10.1109/ICCSE.2011.6028694.

[35] "Radio Frequency Identification Equipment operating in the band 865 MHz to 868 MHz with power levels up to 2 W and in the band 915 MHz to 921 MHz with power levels up to 4 W; Harmonised Standard covering the essential requirements of article 3.2 of the Directive 2014/53/EU," ETSI EN 302 208 V3.1.1, November 2016, European Telecommunications Standards Institute, Sophia-Antipolis Cedex, France.

[36] P. V. Nikitin, K. V. S. Rao, and S. Lazar, "An Overview of Near Field UHF RFID," in *2007 IEEE International Conference on RFID*, Grapevine, TX, USA, March 2007, pp. 167–174, DOI:10.1109/RFID.2007.346165.

[37] Understanding EPC Gen2 Search Modes and Sessions. Impinj, Inc. Seattle, WA, USA. [Online]. Accessed March 13, 2017. »https://support.impinj.com/hc/en-us/articles/202756158-Understanding-EPC-Gen2-Search-Modes-and-Sessions«.

[38] P. V. Nikitin and K. V. S. Rao, "Performance limitations of passive UHF RFID systems," in *2006 IEEE Antennas and Propagation Society International Symposium*, Albuquerque, NM, USA, July 2006, pp. 1011–1014, DOI:10.1109/APS.2006.1710704.

[39] A. Lazaro, D. Girbau, and D. Salinas, "Radio Link Budgets for UHF RFID on Multipath Environments," *IEEE Transactions on Antennas and Propagation*, vol. 57, no. 4, pp. 1241–1251, April 2009, DOI:10.1109/TAP.2009.2015818.

[40] L. Zhenzhong, M. Nezih, X. George, G. Liu, Y. Ono, and B. Dayan, "Effects of orientation and obstacles on the RFID performance," in *2011 4th Annual Caneus Fly by Wireless Workshop*, Montreal, QC, Canada, June 2011, pp. 1–4, DOI:10.1109/FBW.2011.5965564.

[41] H. T. Friis, "A note on a simple transmission formula," *Proceedings of the I.R.E. and Waves and Electrons*, vol. 34, no. 5, pp. 254–256, May 1946.

[42] B. Rolfe, S. Ekanayake, P. Pathirana, and M. Palaniswami, "Localization with orientation using RSSI measurements: RF map based approach," in *3rd International Conference on Intelligent Sensors, Sensor Networks and Information*, Melbourne, Qld., Australia, December 2007, pp. 311–316, DOI:10.1109/ISSNIP.2007.4496862.

[43] Y. Huang, C. Chen, B. W. Hong, T. C. Kuo, and H. H. Yu, "Fuzzy neural network based RFID indoor location sensing technique," in *The 2010 International Joint Conference on Neural Networks*, Barcelona, Spain, July 2010, pp. 1–5, DOI:10.1109/IJCNN.2010.5596352.

[44] R. Mittra and U. Pujare, "Real time estimation of motion and range of RFID tags," in *Proceedings of the 5th European Conference on Antennas and Propagation (EUCAP)*, Rome, Italy, April 2011, pp. 3804–3808.

[45] L. Qiu, Z. Huang, S. Zhang, and W. Wang, "RFID Tag Ranging Measurement Based on Multi-frequency Carrier Phase Difference," in *2014 Seventh International Symposium on Computational Intelligence and Design*, vol. 1, Hangzhou, China, December 2014, pp. 245–248, DOI:10.1109/ISCID.2014.109.

[46] V. Viikari, P. Pursula, and K. Jaakkola, "Ranging of UHF RFID Tag Using Stepped Frequency Read-Out," *IEEE Sensors Journal*, vol. 10, no. 9, pp. 1535–1539, September 2010, DOI:10.1109/JSEN.2010.2045497.

[47] D. K. Kim, J. H. Ha, P. J. Kim, and K. H. You, "TDOA/AOA localization in RFID system using dual indirect Kalman filter," in *2011 IEEE/SICE International Symposium on System Integration (SII)*, Kyoto, Japan, December 2011, pp. 440–445, DOI:10.1109/SII.2011.6147489.

[48] J. Zhou, H. Zhang, and L. Mo, "Two-dimension localization of passive RFID tags using AOA estimation," in *IEEE Instrumentation and Measurement Technology Conference*, May 2011, pp. 1–5, DOI:10.1109/IMTC.2011.5944170.

[49] M. Wegener, D. Froß, M. Rößler, C. Drechsler, C. Pätz, and U. Heinkel, "Relative localisation of passive UHF-tags by phase tracking," in *2016 13th International Multi-Conference on Systems, Signals Devices (SSD)*, Leipzig, Germany, March 2016, pp. 503–506, DOI:10.1109/SSD.2016.7473683.

[50] M. Vossiek, A. Urban, S. Max, and P. Gulden, "Inverse Synthetic Aperture Secondary Radar Concept for Precise Wireless Positioning," *IEEE Transactions on Microwave Theory and Techniques*, vol. 55, no. 11, pp. 2447–2453, November 2007, DOI:10.1109/TMTT.2007.908668.

[51] R. Miesen, F. Kirsch, and M. Vossiek, "Holographic localization of passive UHF RFID transponders," in *IEEE International Conference on RFID*, Orlando, FL, USA, April 2011, pp. 32–37, DOI:10.1109/RFID.2011.5764633.

[52] G. Li and M. Vossiek, "A multilateral synthetic aperture wireless positioning approach to precise 3D localization of a robot tool center point," in *2011 IEEE Topical Conference on Wireless Sensors and Sensor Networks*, Phoenix, AZ, USA, January 2011, pp. 37–40, 10.1109/WISNET.2011.5725017.

[53] G. Li, R. Ebelt, and M. Vossiek, "Particle filter based synthetic aperture reconstruction approach for real-time 3D wireless local positioning," in *2012 The 7th German Microwave Conference*, Ilmenau, Germany, March 2012, pp. 1–4.

[54] A. Parr, R. Miesen, and M. Vossiek, "Inverse SAR approach for localization of moving RFID tags," in *IEEE International Conference on RFID*, Penang, Malaysia, April 2013, pp. 104–109, DOI:10.1109/RFID.2013.6548142.

[55] M. Scherhäufl, M. Pichler, and A. Stelzer, "Localization of passive UHF RFID tags based on inverse synthetic apertures," in *2014 IEEE International Conference on RFID (IEEE RFID)*, Orlando, FL, USA, April 2014, pp. 82–88, 10.1109/RFID.2014.6810716.

[56] Z. Huang, G. Yang, L. Qiu, and Y. Ren, "Differential Based Hologram RFID Indoor Localization Method," in *2015 10th International Conference on Intelligent Systems and Knowledge Engineering (ISKE)*, Taipei, Taiwan, November 2015, pp. 489–493, DOI:10.1109/ISKE.2015.16.

[57] A. Buffi, P. Nepa, and F. Lombardini, "A Phase-Based Technique for Localization of UHF-RFID Tags Moving on a Conveyor Belt: Performance Analysis and Test-Case Measurements," *IEEE Sensors Journal*, vol. 15, no. 1, pp. 387–396, January 2015, DOI:10.1109/JSEN.2014.2344713.

[58] V. Sović, A. Athalye, M. Bolić, and P. M. Djurić, "Particle filtering for indoor RFID tag tracking," in *2011 IEEE Statistical Signal Processing Workshop (SSP)*, Nice, France, June 2011, pp. 193–196, DOI:10.1109/SSP.2011.5967656.

[59] S. Särkkä, V. Viikari, M. Huusko, and K. Jaakkola, "Phase-Based UHF RFID Tracking With Nonlinear Kalman Filtering and Smoothing," *IEEE Sensors Journal*, vol. 12, no. 5, pp. 904–910, May 2012, DOI:10.1109/JSEN.2011.2164062.

[60] S. Haykin, Ed., *Kalman filtering and neural networks*, March 2002, John Wiley & Sons, Inc., New York, USA. DOI:10.1002/0471221546.

[61] L. Jargani, M. Shahbazian, K. Salahshoor, and V. Fathabadi, "State estimation of nonlinear systems using novel adaptive unscented Kalman filter," in *International Conference on Emerging Technologies*, Islamabad, Pakistan, October 2009, pp. 124–129, DOI:10.1109/ICET.2009.5353190.

[62] I. Sommerville, *Software Engineering*, 9th ed. Harlow, England: Addison-Wesley, 2010.

[63] "Information Services (EPCIS) 1.1 Specification," GS1 EPCglobal, May 2014, GS1 Standard.

[64] *Low Level Reader Protocol, Version 1.1*, EPCglobal Inc., October 13, 2010, Ratified Standard.

[65] *Reader Protocol Standard, Version 1.1*, EPCglobal Inc., June 26, 2006, Ratified Standard.

[66] "LLRP—Reader Control Simplified," Impinj, Inc. Seattle, WA, USA, June 8, 2010.

[67] *Reader Management Standard, Version 1.0.1*, EPCglobal Inc., May 31, 2007, Ratified Standard.

[68] "The GS1 EPCglobal Architecture Framework," GS1 EPCglobal, April 18, 2015, GS1 Version 1.7.

[69] *Reader and Gateway Embedded Developers Guide Version 5.6.2*, Impinj, Inc. Seattle, WA, USA, December 21, 2015.

[70] LLRP Toolkit. [Online]. Accessed April 30, 2017. »http://www.llrp.org».

[71] OpenSSL – Cryptography and SSL/TLS Toolkit. [Online]. Updated March 23, 2017. Accessed May 1, 2017. »https://www.openssl.org/».

[72] POCO C++ Libraries. [Online]. Updated April 18, 2017. Accessed May 1, 2017. »https://pocoproject.org/».

[73] Why POCO is well implemented and designed? [Online]. Updated September 24, 2009. Accessed April 30, 2017. »https://cppdepend.wordpress.com/2009/09/24/why-poco-is-well-implemented-and-designed/».

[74] LuaJIT. [Online]. Updated January 17, 2017. Accessed May 1, 2017. »http://luajit.org/luajit.html».

[75] C++ binding to Lua. [Online]. Updated April 17, 2017. Accessed May 1, 2017. »https://github.com/satoren/kaguya».

[76] A fast multi-producer, multi-consumer lock-free concurrent queue for C++11. [Online]. Updated April 20, 2017. Accessed May 1, 2017. »https://github.com/cameron314/concurrentqueue».

[77] H. Alayli. lthread, a multicore enabled coroutine library written in C. [Online]. Accessed May 1, 2017. »http://lthread.readthedocs.io/en/latest/intro.html».

[78] Google Test – Google's C++ test framework. [Online]. Updated February 26, 2017. Accessed May 1, 2017. »https://github.com/google/googletest».

[79] jQuery JavaScript Library. [Online]. Updated April 29, 2017. Accessed May 1, 2017. »https://github.com/jquery/jquery».

[80] Material Design Components in HTML/CSS/JS . [Online]. Updated April 26, 2017. Accessed May 1, 2017. »https://github.com/google/material-design-lite».

[81] W. Meizhen, S. Yanlei, and T. Yue, "The design and implementation of LRU-based web cache," in *2013 8th International Conference on Communications and Networking in China (CHINACOM)*, August 2013, pp. 400–404, 10.1109/ChinaCom.2013.6694629.

[82] MurmurHash. [Online]. Updated March 1, 2011. Accessed May 14, 2017. »https://sites.google.com/site/murmurhash».

[83] V. Driessen. A successful Git branching model. [Online]. Updated January 5, 2010. Accessed March 13, 2017. »http://nvie.com/posts/a-successful-git-branching-model».

[84] B. Regnell, R. B. Svensson, and T. Olsson, "Supporting Roadmapping of Quality Requirements," *IEEE Software*, vol. 25, no. 2, pp. 42–47, March 2008, DOI:10.1109/MS.2008.48.

Appendix A ISAR images