IARIA

UBICOMM 2023

The Seventeenth International Conference on Mobile Ubiquitous Computing,
Systems, Services and Technologies

ISBN: 978-1-68558-106-0

September 25 - 29, 2023

Porto, Portugal

UBICOMM 2023 Editors

Jaime Lloret Mauri, Polytechnic University of Valencia, Spain

UBICOMM 2023

Forward

The Seventeenth International Conference on Mobile Ubiquitous Computing, Systems, Services and Technologies (UBICOMM 2023), held between September 25th and September 29th, 2023, continued a series of international events meant to bring together researchers from the academia and practitioners from the industry in order to address fundamentals of ubiquitous systems and the new applications related to them.

The rapid advances in ubiquitous technologies make fruition of more than 35 years of research in distributed computing systems, and more than two decades of mobile computing. The ubiquity vision is becoming a reality. Hardware and software components evolved to deliver functionality under failure-prone environments with limited resources. The advent of web services and the progress on wearable devices, ambient components, user-generated content, mobile communications, and new business models generated new applications and services. The conference makes a bridge between issues with software and hardware challenges through mobile communications.

Advances in web services technologies along with their integration into mobility, online and new business models provide a technical infrastructure that enables the progress of mobile services and applications. These include dynamic and on-demand service, context-aware services, and mobile web services. While driving new business models and new online services, particular techniques must be developed for web service composition, web service-driven system design methodology, creation of web services, and on-demand web services.

As mobile and ubiquitous computing becomes a reality, more formal and informal learning will take pace out of the confines of the traditional classroom. Two trends converge to make this possible: increasingly powerful cell phones and PDAs, and improved access to wireless broadband. At the same time, due to the increasing complexity, modern learners will need tools that operate in an intuitive manner and are flexibly integrated in the surrounding learning environment.

Educational services will become more customized and personalized, and more frequently subjected to changes. Learning and teaching are now becoming less tied to physical locations, co- located members of a group, and co-presence in time. Learning and teaching increasingly take place in fluid combinations of virtual and "real" contexts, and fluid combinations of presence in time, space and participation in community. To the learner full access and abundance in communicative opportunities and information retrieval represents new challenges and affordances. Consequently, the educational challenges are numerous in the intersection of technology development, curriculum development, content development and educational infrastructure.

We take here the opportunity to warmly thank all the members of the UBICOMM 2023 technical program committee, as well as all the reviewers. The creation of such a high-quality conference program would not have been possible without their involvement. We also kindly thank all the authors who dedicated much of their time and effort to contribute to UBICOMM 2023. We truly believe that, thanks to all these efforts, the final conference program consisted of top-quality contributions. We also thank the members of the UBICOMM 2023 organizing committee for their help in handling the logistics of this event.

We hope that UBICOMM 2023 was a successful international forum for the exchange of ideas and results between academia and industry and for the promotion of progress related to mobile ubiquitous computing, systems, services, and technologies.

UBICOMM 2023 Chairs

UBICOMM 2023 Steering Committee Chair
Jaime Lloret Mauri, Polytechnic University of Valencia, Spain

UBICOMM 2023 Steering Committee
Stéphane Galland, Belfort-Montbéliard University of Technology, France
Wladyslaw Homenda, Warsaw University of Technology, Poland
Chunguo Li, Southeast University, China
Dmitry Korzun, Petrozavodsk State University, Russia

UBICOMM 2023 Publicity Chairs
Laura Garcia, Universitat Politecnica de Valencia, Spain
Lorena Parra Boronat, Universitat Politecnica de Valencia, Spain

UBICOMM 2023
Committee

UBICOMM 2023 Steering Committee Chair

Jaime Lloret Mauri, Polytechnic University of Valencia, Spain

UBICOMM 2023 Steering Committee

Stéphane Galland, Belfort-Montbéliard University of Technology, France
Wladyslaw Homenda, Warsaw University of Technology, Poland
Chunguo Li, Southeast University, China
Dmitry Korzun, Petrozavodsk State University, Russia

UBICOMM 2023 Publicity Chairs

Laura Garcia, Universitat Politecnica de Valencia, Spain
Lorena Parra Boronat, Universitat Politecnica de Valencia, Spain

UBICOMM 2023 Technical Program Committee

Afrand Agah, West Chester University of Pennsylvania, USA
Wafaa Ait-Cheik-Bihi, Schneider Electric, France
Mehmet Akşit, TOBB ET University, Ankara, Turkey / University of Twente, The Netherlands
A. B. M. Alim Al Islam, Bangladesh University of Engineering and Technology, Bangladesh
Mrim Alnfiai, Dalhousie University, Canada
Tahssin Altabbaa, Istanbul Gelisim University / Huawei Istanbul, Turkey
Nafisa Anzum, University of Waterloo, Canada
Paramasiven Appavoo, University of Mauritius, Mauritius
Mehran Asadi, Lincoln University, USA
F. Mzee Awuor, Kisii University, Kenya
Muhammed Ali Aydin, Istanbul University - Cerrahpasa, Turkey
Nebojsa Bacanin, Singidunum University, Serbia
Chiara Bachechi, University of Modena and Reggio Emilia, Italy
Matthias Baldauf, FHS St.Gallen, Switzerland
Anoud I Bani-hani, Zayed University, Dubai, UAE
Luca Bedogni, University of Modena and Reggio Emilia, Italy
Oladayo Bello, New Mexico State University, Las Cruces, USA
Imed Ben Dhaou, University of Turku, Finland
Imen Ben Lahmar, ISIM Sfax | ReDCAD laboratory | University of Sfax, Tunisia
Djamal Benslimane, Université Claude Bernard Lyon 1, France
Aurelio Bermúdez, Universidad de Castilla-La Mancha, Spain
Javier Berrocal, University of Extremadura, Spain
Nik Bessis, Edge Hill University, UK
Robert Bestak, Czech Technical University in Prague, Czech Republic
Sourav Kumar Bhoi, Parala Maharaja Engineering College, India
Lucas Botoni De Souza, Federal University of Technology - Paraná, Brazil

Nadia Bouassida, Higher Institute of computer science and Multimedia, Sfax, Tunisia
Chérifa Boucetta, University of Reims Champagne-Ardenne, France
Yassine Boujelben, National School of Electronics and Telecommunications of Sfax | University of Sfax, Tunisia
Azedine Boulmakoul, Université Hassan II de Casablanca, Morocco
Maurizio Bozzi, University of Pavia, Italy
Erik Buchmann, Hochschule für Telekommunikation Leipzig, Germany
Joseph Bugeja, Malmö University, Sweden
Christian Cabrera, Trinity College Dublin, Ireland
Diego Leoenel Cadette Dutra, Federal University of Rio de Janeiro, Brazil
Juan Vicente Capella Hernández, Universitat Politècnica de València, Spain
Beenish Moalla Chaudhry, The University of Louisiana, Lafayette, USA
Chao Chen, Purdue University Fort Wayne, USA
Radu-Ioan Ciobanu, University Politehnica of Bucharest, Romania
Michael Collins, Technological University Dublin, Ireland
André Constantino da Silva, IFSP & NIED/UNICAMP, Brazil
Giuseppe D'Aniello, University of Salerno, Italy
Roland Dodd, Central Queensland University, Australia
Ivanna Dronyuk, Lviv Polytechnic National University, Ukraine
Jalel Dziri, National Engineering School of Tunis | University Tunis El Manar, Tunisia
Wael M. El-Medany, University of Bahrain, Bahrain
Francisco Falcone, ISC-UPNA, Spain
Przemyslaw Falkowski-Gilski, Gdansk University of Technology, Poland
Andras Farago, University of Texas at Dallas, USA
Olga Fedevych, Lviv Polytechnic National University, Ukraine
Niroshinie Fernando, Deakin University, Australia
Renato Ferrero, Politecnico di Torino, Italy
Olivier Flauzac, University of Reims, France
Franco Frattolillo, University of Sannio, Benevento, Italy
Stéphane Galland, Belfort-Montbéliard University of Technology, France
Crescenzio Gallo, University of Foggia, Italy
Jose Garcia-Alonso, University of Extremadura, Spain
Vassilis C. Gerogiannis, University of Thessaly, Greece
Seyed Ali Ghorashi, University of East London, UK
Mikhail Gofman, California State University, Fullerton, USA
Javier Gozalvez, Universidad Miguel Hernandez de Elche, Spain
Clementine Gritti, University of Canterbury, New Zealand
Weixi Gu, UCBerkeley, USA
Zhichun Guo, University of Notre Dame, USA
Mesut Güneş, Institute for Intelligent Cooperating Systems | Otto-von-Guericke-University Magdeburg, Germany
Cornelia Aurora Győrödi, University of Oradea, Romania
Qiang (Nathan) He, Swinburne University of Technology, Australia
Songlin He, New Jersey Institute of Technology (NJIT), USA
Wladyslaw Homenda, Warsaw University of Technology, Poland
Tzung-Pei Hong, National University of Kaohsiung, Taiwan
Sergio Ilarri, University of Zaragoza, Spain
Yasser Ismail, Southern University and A&M College, USA

Ivan Pires, University of Beira Interior, Portugal
Laura Po, University of Modena and Reggio Emilia, Italy
Christian Prehofer, DENSO Automotive, Germany
Rashed Rahman, Georgia State University, USA
Tomasz Rak, Rzeszow University of Technology, Poland
Ann Ramirez, University of Florida, USA
Luca Reggiani, Politecnico di Milano, Italy
Elena Renda, IIT - CNR, Italy
André Restivo, University of Porto, Portugal
Amine Rghioui, EMI - Mohamed V University, Morocco
Ana Patrícia Rocha, University of Aveiro, Portugal
Federica Rollo, University of Modena and Reggio Emilia, Italy
Michele Ruta, Politecnico di Bari, Italy
Khair Eddin Sabri, The University of Jordan, Jordan
Prasan Kumar Sahoo, Chang Gung University, Taiwan
Zaineb Sakhrawi, University of Sfax, Tunisia
Josep Maria Salanova Grau, Center for Research and Technology Hellas, Greece
Mohsen Amini Salehi, University of Louisiana at Lafayette, USA
Moid Sandhu, University of Queensland | Data61 - Commonweath Scientific and Research Organization (CSIRO), Australia
José Santa, Technical University of Cartagena, Spain
Peter Schneider-Kamp, University of Southern Denmark, Denmark
Floriano Scioscia, Polytechnic University of Bari, Italy
Luca Sciullo, University of Bologna, Italy
Hugo Sereno Ferreira, Universityof Porto, Portugal
Alireza Shahrabi, Glasgow Caledonian University, Scotland, UK
Jianchen Shan, Hofstra University, USA
Ahmed S. Shatnawi, Jordan University of Science and Technology, Jordan
Shih-Lung Shaw, University of Tennessee, Knoxville, USA
Haichen Shen, Amazon Web Services, USA
Michael Sheng, Macquarie University, Australia
Shouqian Shi, University of California, Santa Cruz, USA
Matteo Signorini, Nokia Bell Labs, France
Sandeep Singh Sandha, University of California-Los Angeles, USA
Rute C. Sofia, fortiss GmbH, Munich, Germany
Francesco Soldovieri, Institute for Electromagnetic Sensing of the Environment | CNR, Italy
Christoph Stach, University of Stuttgart, Germany
Álvaro Suárez Sarmiento, University of Las Palmas de Gran Canaria, Spain
K. Subramani, West Virginia University, USA
Apostolos Syropoulos, Greek Molecular Computing Group, Xanthi, Greece
Violet R. Syrotiuk, Arizona State University, USA
Yoshiaki Taniguchi, Kindai University, Japan
Sudeep Tanwar, Institute of Technology | Nirma University, India
Adrian Tarniceriu, Securecell, Switzerland
Angelo Trotta, University of Bologna, Italy
Takeshi Tsuchiya, Suwa University of Science, Japan
Sudhanshu Tyagi, Thapar Institute of Engineering & Technology, India / Jan Wyykowski University Polkowice, Poland

Copyright Information

For your reference, this is the text governing the copyright release for material published by IARIA.

The copyright release is a transfer of publication rights, which allows IARIA and its partners to drive the dissemination of the published material. This allows IARIA to give articles increased visibility via distribution, inclusion in libraries, and arrangements for submission to indexes.

I, the undersigned, declare that the article is original, and that I represent the authors of this article in the copyright release matters. If this work has been done as work-for-hire, I have obtained all necessary clearances to execute a copyright release. I hereby irrevocably transfer exclusive copyright for this material to IARIA. I give IARIA permission or reproduce the work in any media format such as, but not limited to, print, digital, or electronic. I give IARIA permission to distribute the materials without restriction to any institutions or individuals. I give IARIA permission to submit the work for inclusion in article repositories as IARIA sees fit.

I, the undersigned, declare that to the best of my knowledge, the article is does not contain libelous or otherwise unlawful contents or invading the right of privacy or infringing on a proprietary right.

Following the copyright release, any circulated version of the article must bear the copyright notice and any header and footer information that IARIA applies to the published article.

IARIA grants royalty-free permission to the authors to disseminate the work, under the above provisions, for any academic, commercial, or industrial use. IARIA grants royalty-free permission to any individuals or institutions to make the article available electronically, online, or in print.

IARIA acknowledges that rights to any algorithm, process, procedure, apparatus, or articles of manufacture remain with the authors and their employers.

I, the undersigned, understand that IARIA will not be liable, in contract, tort (including, without limitation, negligence), pre-contract or other representations (other than fraudulent misrepresentations) or otherwise in connection with the publication of my work.

Exception to the above is made for work-for-hire performed while employed by the government. In that case, copyright to the material remains with the said government. The rightful owners (authors and government entity) grant unlimited and unrestricted permission to IARIA, IARIA's contractors, and IARIA's partners to further distribute the work.

Table of Contents

LTE-M Communication for Low-Powered IIoT: An Experimental Performance Study 1
Eddy Bajic, Kais Mekki, and Clement Rup

Using Environmental Contexts to Model Restrictions on Sensor Capabilities 7
Martin Richter, Christine Jakobs, Theresa Werner, and Matthias Werner

CRDT-based Collaborative Editing in OppNets: a Practical Experiment 13
Yves Maheo, Frederic Guidec, and Camille Nous

Deep Learning based Indoor Positioning Approach Using WiFi CSI/RSSI Fingerprints Technique 22
Marco Muhl and Wiem Fekih Hassen

Threat Detection based on System Credibility by Logging Analysis and Visualization 28
Wei Qiao, Youjun Bu, Yao Chen, Xiaoxiao Jiang, and Bingbing Jiang

Location and Object-Based Mobile Applications - Development Based on Use Case Templates and Visual Programming 34
Martin Zimmermann

WiFi CSI/RSSI Fingerprints Positioning based on Data Augmentation Technique 40
Mohamed Amin Elaoud and Wiem Fekih Hassen

LTE-M Communication for Low-Powered IIoT: An Experimental Performance Study

Eddy Bajic
Research Centre for Automatic
Control of Nancy, CNRS 7039
Campus Sciences, 54506 Vandoeuvre-lès-Nancy, France
e-mail: eddy.bajic@univ-lorraine.fr

Kais Mekki
OKKO SAS, 107 rue Saint Jean,
57510 Remering-lès-Puttelange,
France
e-mail: kais.mekki@okko-france.com

Clement Rup
Research Centre for Automatic
Control of Nancy, CNRS 7039
Campus Sciences, 54506 Vandoeuvre-lès-Nancy, France
e-mail: clement.rup@univ-lorraine.fr

***Abstract*—This paper presents an experimental study of the communication performance of the Long-Term Evolution for Machines (LTE-M) network for the Industrial Internet of Things (IIoT). We conducted an in-depth literature review on the communication networks used by IIoT devices, based on criteria such as transportable data size, throughput, connectivity, latency, transmission energy budget and cost. Next, we experimented with LTE-M communication for a prototype energy-constrained IoT device. Using the microcontroller and modem, we were able to establish TCP requests (send and receive), as well as HTTP requests (POST and GET). The results of our study show that LTE-M communication offers significant advantages in terms of throughput, latency and energy performance compared with other existing Low Power Wide Area Networks (LPWAN), making it particularly well suited to the needs of IIoT applications. This experimental work opens a wide range of prospects for setting up efficient, high-performance IIoT solutions in industrial environments.**

Keywords—Industrial Internet of Things; Low Power Wide Area Network; LTE-M; Experimental study.

I. Introduction

The Industrial Internet of Things (IIoT) is a constantly evolving field, with rapid growth in the number of IoT devices integrated into multi-tier communication architectures to collect and process mass industrial data. Communication networks play a crucial role in this evolution, offering solutions to meet the needs of different industrial scenarios.

Low Power Wide Area Networks (LPWAN) are key communication technologies that are chosen based on specific criteria such as the size of transportable data, throughput, connectivity, latency, the energy budget for transmissions and cost [1]. Sigfox, LoRaWAN, LTE-M, and NB-IoT are the main existing LPWAN networks. Currently in France, the Sigfox and LoRaWAN networks have demonstrated their inefficiency in terms of connectivity/coverage for large-scale deployment, particularly in rural areas [2][3]. For example, Objenious, Bouygues Telecom's IoT subsidiary, has shut down its LoRaWAN network [4]. The LTE-M network, on the other hand, is currently being deployed and marketed on a larger scale than the NB-IoT network [5].

In this context, our experimental study focuses on the communication performance of the LTE-M network. To do this, we first carried out an in-depth literature review on LPWAN for IoT communications, based on the available scientific literature. Then, we set up an experiment to test and evaluate the performance of LTE-M communication for an energy-constrained prototype IoT terminal.

In view of our study of the existing literature, we consider that this work represents an initial contribution detailing the implementation of this type of experiment. It details the sequence of AT commands to be used to operate an LTE-M modem. This sequence is not detailed anywhere on the Internet, even in the modem manufacturers' documentation. Hence, this work represents a tutorial and a reference document for future projects and opens new prospects for the implementation of effective, high-performance IoT solutions in industrial environments.

The rest of the paper is organized as follows. First, the state of the art is presented. Then, the test carried out of LTE-M network is detailed. Finally, experimental results are discussed.

II. State of the Art

In the following, we present the Industrial Internet of Things domain, its emerging technologies, and the technical aspects of its wireless communication solutions.

A. Internet of Things

The Internet of Things (IoT) describes the network of physical terminals, or "objects", that embed sensors, actuators, software, and other technologies to connect to servers on the Internet (Cloud servers) and exchange data with them. These terminals range from simple domestic devices to highly complex industrial tools [6]. It is thanks to several different technologies that the IoT was able to emerge. Technologies, such as advances in connectivity, with the proliferation of network protocols that have made it easier to connect sensors to the Cloud and to other 'objects', making data transfer more efficient; low-cost, low-power sensor technologies that have made the field more accessible; advances in Machine Learning and analytics; and access to vast quantities of diverse data stored in the Cloud, enabling businesses to obtain information more quickly [7].

B. Industrial Internet of Things

The Industrial Internet of Things (IIoT) refers to the application of IoT technology in an industrial context, especially for the instrumentation and control of sensors and terminals using Cloud technologies. To transfer all this data between objects, networks have had to be set up that are adapted not only to the objects themselves, but also to the quantity of data to be sent and the industrial environment in which the objects operate. The IIoT generally uses a two- or three-tier Device-Edge-Cloud network communication architecture to collect and process massive industrial data [8][9]. Moreover, LPWAN are increasingly used for industrial IoT applications because of their ability to provide low-cost, low-power connectivity. The characteristics of these networks are therefore very important when choosing the appropriate network for an industrial IoT application.

C. IoT networks

With the rapid advance of wireless communication technologies, choosing the right network for IoT devices can seem complicated. However, for an IoT system architect, it is essential to consider the specifications and constraints associated with the application when choosing the most suitable network. The choice of network for an IoT device must be based on a careful assessment of these specifications and constraints. There are four main criteria for choosing a network:

- The scope
- Data transmission rate
- Energy consumption
- The cost of deployment

Range is determined by the maximum distance data can be transmitted between devices and access points, while data transmission rate is the speed at which data can be transmitted. Power consumption is crucial for battery powered IoT devices, as it determines the autonomy of the devices. Also, the cost of deployment is an important factor to consider when deciding to use a specific network.

D. LPWAN

Low Power Wide Area Networks (LPWAN) characterizes, as its name suggests, all networks with a long range and low energy consumption. By reducing the data rate, data can be sent over greater distances while maintaining low power consumption [10]. When we talk about long distance for this type of network, we are talking about an order of magnitude of a few kilometers, with a device autonomy of up to several years and a bandwidth of around several hundred kbits/sec. The various LPWAN implementations differ in terms of throughput and frequency bands used. Nevertheless, these implementations can be classified into two categories: cellular LPWAN and non-cellular LPWAN.

1) Non-cellular LPWAN:

SigFox: French start-up SigFox was the first to make LPWAN technology popular. In the early 2010s, the company developed the SigFox network, which is based on patented Ultra Narrow Band (UNB) technology and uses unlicensed Industrial, Scientific, and Medical (ISM) frequency bands: 868 MHz in Europe, 915 MHz in North America and 433 MHz in Asia. SigFox antennas have a range of between 3 and 10 kilometers in dense areas and up to 50 kilometers in areas with few obstacles [6]. However, it should be noted that the SigFox network covers the whole of France, with 2,000 antennas deployed throughout the country. Internationally, the network covers 71 other countries in Europe and around the world, including 21 with global coverage.

Long Range Wide Area Network (LoRaWAN): LoRaWAN provides two-way data transmission at very low bit rates and very low energy consumption between objects and gateways. It uses a Chirp Spread Spectrum (CSS) modulation called LoRa. Like SigFox, this modulation is used mainly in frequency bands without an ISM license. The use of CSS modulation for the Internet of Things was patented by Cycléo, a French company that was later acquired by Semtech in 2012 [8]. On average, this modulation allows up to 5 kilometers between a gateway and an object in urban areas and 15 kilometers in rural areas.

2) Cellular LPWAN:

Narrowband Internet of Things (NB-IoT): NB-IoT antennas have a range of up to one kilometer in urban areas and 10 kilometers in rural areas. Within these ranges, NB-IoT provides bidirectional transmission at data rates of between 20 and 250 Kbps. This low data rate has led to a significant reduction in the energy consumption of IoT terminals, making them more suitable for battery-powered remote monitoring applications. This low data rate has also reduced the complexity and therefore the cost of deployment.

Long-Term Evolution for Machines (LTE-M): LTE-M signals have a range of up to 400 meters in urban areas and around 8 kilometers in rural areas. In the same way as for NB-IoT, the reduction in the data transmission rate has made it possible to reduce the energy consumption of the terminals deployed, thus opening the field of IoT applications to use cases that were previously reserved solely for non-cellular LPWAN.

Unlike NB-IoT, LTE-M enables data to be transmitted at lower data rates, in the order of ten bits per second, and at data rates of around 1 Mb per second. As a result, LTE-M covers a wider range of applications than NB-IoT [5] [11].

III. Testing LTE-M network

As discussed before, LTE-M is currently being deployed and marketed on a larger scale than others IoT network. Thus, this section aims to test and evaluate the performance of this technology for an energy-constrained prototype IoT terminal.

A. Hardware

To carry out the implementation and various tests, we used an IoT development board designed by our project partner OKKO. This board incorporates a compact, high-performance ESP32-S2 microcontroller, specially designed for IoT applications. It was a wise choice because of its small size, advanced features, and low power consumption. It also incorporates a Sierra Electronics HL7800 modem, a wireless communication module that uses LTE-M and NB-IoT networks to provide low-speed connectivity and low power consumption, making it an ideal choice for IoT applications [12]. The modem also offers an integrated GPS module for location based IoT applications. This board offers several I2C connectors for connecting ambient sensors. We used an IoT SIM card from the multi-operator Thingsmobile [13]. Figures 1, 2 and 3 illustrate the hardware used.

B. *Problem analysis*

The main objective of our project was to develop a system capable of efficiently transmitting the data collected by various sensors to a remote server. To achieve this, we used the LTE-M modem to establish an Internet connection and send the data via TCP and HTTP requests. The card's operation had to ensure very low energy consumption. We therefore implemented a strategy involving a sequence of actions followed by a prolonged sleep mode, to save energy during periods of inactivity. This approach considerably extended the system's autonomy, while guaranteeing reliable and efficient transmission of the data collected.

The ESP32-S2 microcontroller used for this project is a customized version which presents a few differences from the standard models, particularly in terms of the number and limitation of programmable pins. This particularity meant that our codes had to be adapted to take account of this constraint. In addition, we have found that the command return terminal can sometimes have uncontrollable character inversions or deletions. We had to deal with this problem when implementing certain functionalities, such as setting up a monitoring and connection recovery system in the event of signal loss. Despite these constraints, we succeeded in overcoming these difficulties by carefully analyzing the feedback from the terminal and adapting our codes to take account of it. We thus worked to achieve two key objectives: the transmission of data via TCP requests, then the sending of HTTP requests to an external Cloud server.

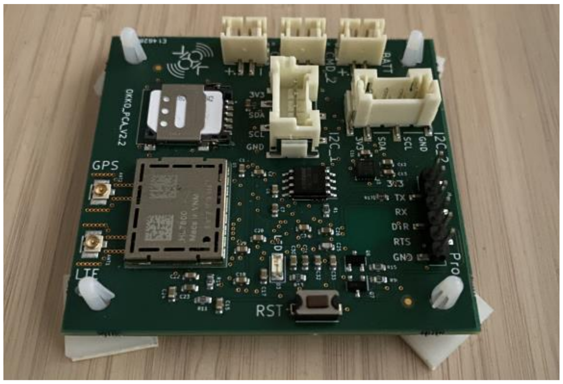


Figure 1. The OKKO board: ESP32-S2 microcontroller + HL7800 LTE-M modem + I2C connectors.

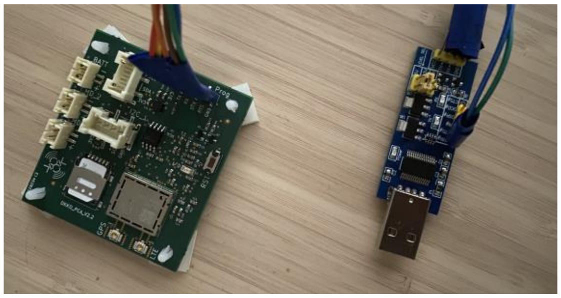


Figure 2. Assembly for uploading Arduino code to the OKKO board.

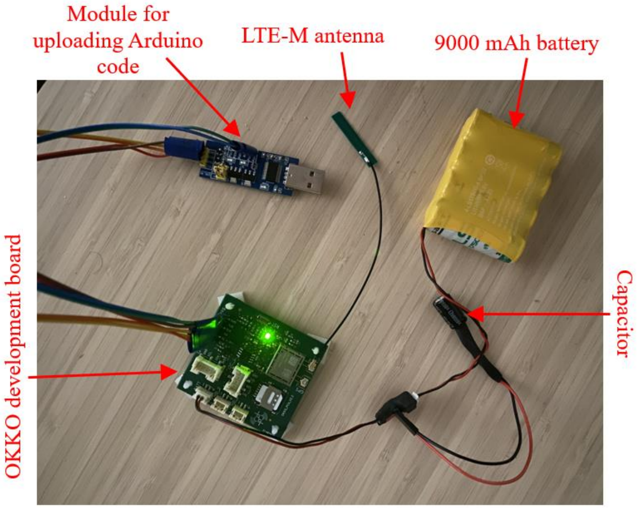


Figure 3. Complete assembly for battery operation.

C. *Study of the manufacturer's documentation*

To understand how the ESP32-S2 microcontroller works, we studied its documentation and carried out tests by modifying the values of the pins on the electronic board. This enabled us to control the flashing of the LED on the board according to our needs. This step was crucial for the rest of the project, as it enabled us to create a reliable visual feedback method for future communication tests with the LTE-M modem. We also explored the different libraries available for communicating with this HL7800 modem [12], testing several of them to determine the one best suited to our needs.

We then consulted the modem's documentation in full. The ESP32-S2 microcontroller can communicate directly with the HL7800 modem using the AT commands listed in the documentation. You can find explanations of the possible commands and even examples of instruction strings to perform certain tasks. As our focus is on TCP and HTTP requests, we have concentrated on these areas in the documentation.

Each of these commands can be used in different ways, and most functions have three versions:

- A read mode specified by adding a ? to the command name, returns the status of parameters related to the command.

- A write mode (adding an = to the command name) which allows these parameters to be modified and the command to be executed.
- A test mode that returns a list of possible values for each parameter.

We have seen that the LTE-M modem has two main modes and, depending on what is sent to it, one or the other will be activated:
- The main mode, known as command mode, is the one the modem should be in when you want to send it an AT command.
- The given mode is the one the modem is in when, for example, it is waiting to receive data to transmit. This can be seen when connecting to a TCP server, where the mode is activated to send data to the server.

Once these tasks had been completed, we concluded that it was necessary to have routines that could be executed after the microcontroller and modem had been initialized. The SIM card in the LTE-M modem must first be setup to establish a connection with the cellular network. AT commands are used for this initialization step. These commands allow the modem to be configured and controlled via a serial command interface. When the SIM card is initialized, it is also possible to configure the cellular network parameters such as the Access Point Name (APN), username and password, which are required to connect to the cellular network. Once the SIM card has been initialized, the modem can then be configured to establish a TCP or HTTP connection with a remote server using the relevant AT commands. To test the capabilities of our system, we set up and programmed two servers in Python, as shown in Figure 4: one capable of receiving HTTP requests and the other TCP requests. This set-up enabled us to test the two modes of communication and validate that they worked properly. According to the modem's documentation, sending HTTP requires an extra step than sending TCP, because the fields needed for the HTTP request must be placed. We therefore started with the TCP request using the following AT commands: AT-KTCPCFG, AT-KTCPSND, ATKTCPRCV and AT-KTCPCLOSE. For HTTP requests, we used the following AT commands: AT-KHTTPCFG, AT-KHTTPGET, AT-KHTTPPOST, ATKHTTPHEADER and AT-KHTTPCLOSE.

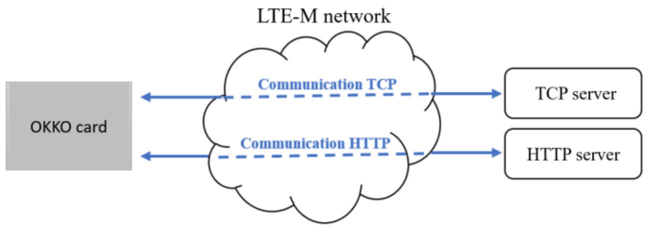
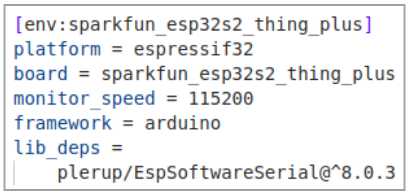


Figure 4. Communication between the card and the servers via the LTE-M network.

D. Implementation

To implement our project, we used the Visual Studio Code editor with the Arduino PlatformIO extension to compile and send our code to our microcontroller, which enabled us to work efficiently and save time in the development process. As with any PlatformIO project, we defined the specifications for the board and the Framework in a "platform.ini" file, as shown in Figure 5. We set the data transmission speed from the serial port to the monitor, which helps display the modem responses, to 115200 baud. The choice of this speed depends on the microcontroller used, in our case an ESP32-S2 derivative. The EspSoftwareSerial library is included in PlatformIO and provides the various instructions for output to the monitor. It was necessary to import it when working on the Visual Studio Code environment.

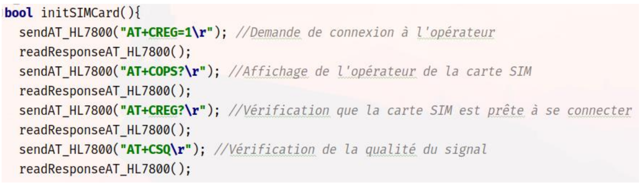


```
[env:sparkfun_esp32s2_thing_plus]
platform = espressif32
board = sparkfun_esp32s2_thing_plus
monitor_speed = 115200
framework = arduino
lib_deps =
    plerup/EspSoftwareSerial@^8.0.3
```

Figure 5. Extract from the platform.ini file.

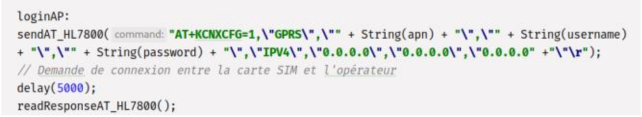
To initialize the SIM card, several AT commands are required. Firstly, the AT+CREG= command is used to start the operator search and connection to the chosen operator. Two checks are then carried out:
- AT+COPS? command ensures that the modem has chosen the right operator.
- AT+CREG? command shows the status of the operator search and ensures that it has been assigned to the correct operator.

The quality of the data rate (in dB) is then read using AT+CSQ to check that the antenna is operating correctly. This sequence of commands is illustrated in Figure 6. Then, as shown in Figure 7, the AT+KCNXCFG= command tells the modem which parameters to use to connect to the operator's network.



```
bool initSIMCard(){
  sendAT_HL7800("AT+CREG=1\r"); //Demande de connexion à l'opérateur
  readResponseAT_HL7800();
  sendAT_HL7800("AT+COPS?\r"); //Affichage de l'opérateur de la carte SIM
  readResponseAT_HL7800();
  sendAT_HL7800("AT+CREG?\r"); //Vérification que la carte SIM est prête à se connecter
  readResponseAT_HL7800();
  sendAT_HL7800("AT+CSQ\r"); //Vérification de la qualité du signal
  readResponseAT_HL7800();
```

Figure 6. initSIM function - part 1.



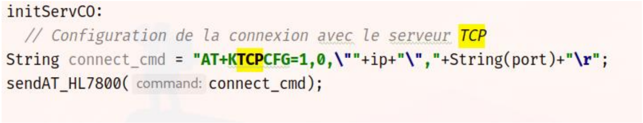
```
loginAP:
sendAT_HL7800( command: "AT+KCNXCFG=1,\"GPRS\",\"" + String(apn) + "\",\"" + String(username)
+ "\",\"" + String(password) + "\",\"IPV4\",\"0.0.0.0\",\"0.0.0.0\",\"0.0.0.0" +"\"\r");
// Demande de connexion entre la carte SIM et l'opérateur
delay(5000);
readResponseAT_HL7800();
```

Figure 7. initSIM function - part 2.

ISBN: 978-1-68558-106-0

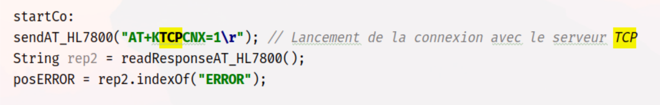
1) Sending a TCP message:

As shown in Figure 8, the AT+KTCPCFG= command allows the modem to specify the IP address and port of the TCP server to connect to. Then, in Figure 9, the AT+KTCPCNX= command enables the LTE-M modem to launch the TCP connection with the server.



```
initServCO:
  // Configuration de la connexion avec le serveur TCP
String connect_cmd = "AT+KTCPCFG=1,0,\""+ip+"\","+String(port)+"\r";
sendAT_HL7800( command: connect_cmd);
```

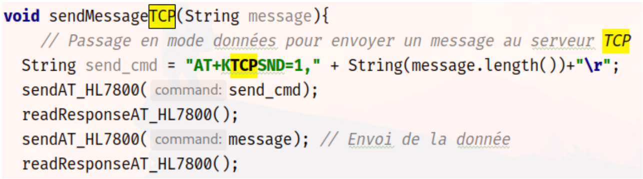
Figure 8. initSIM function - part 3.



```
startCo:
sendAT_HL7800("AT+KTCPCNX=1\r"); // Lancement de la connexion avec le serveur TCP
String rep2 = readResponseAT_HL7800();
posERROR = rep2.indexOf("ERROR");
```

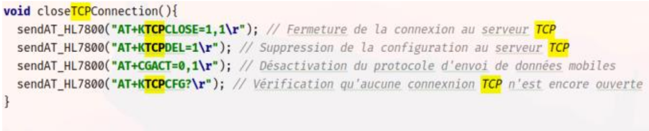
Figure 9. Extract from the connectTCP function.

The AT+KTCPSND= command switches the modem to data mode and transmits everything it receives until it has reached the maximum size specified in the parameter, or until it returns to command mode. Finally, the TCP connection is closed. These procedures are illustrated in Figures 10 and 11, respectively. This sequence of AT commands allows the connection to the TCP server to be fully executed.



```
void sendMessageTCP(String message){
   // Passage en mode données pour envoyer un message au serveur TCP
  String send_cmd = "AT+KTCPSND=1," + String(message.length())+"\r";
  sendAT_HL7800( command: send_cmd);
  readResponseAT_HL7800();
  sendAT_HL7800( command: message); // Envoi de la donnée
  readResponseAT_HL7800();
```

Figure 10. Extract from the sendMessageTCP function.

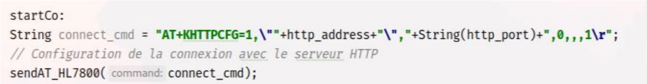


```
void closeTCPConnection(){
  sendAT_HL7800("AT+KTCPCLOSE=1,1\r"); // Fermeture de la connexion au serveur TCP
  sendAT_HL7800("AT+KTCPDEL=1\r"); // Suppression de la configuration au serveur TCP
  sendAT_HL7800("AT+CGACT=0,1\r"); // Désactivation du protocole d'envoi de données mobiles
  sendAT_HL7800("AT+KTCPCFG?\r"); // Vérification qu'aucune connexnion TCP n'est encore ouverte
}
```

Figure 11. Extract from the closeTCPConnection function.

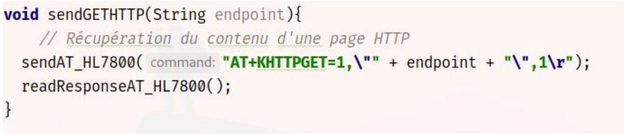
2) HTTP GET and HTTP POST exchanges between the card and the server:

As shown in Figure 12, the AT+KHTTPCFG= command allows the modem to specify the IP address and port of the HTTP server to connect to. Then, in Figure 13, the AT+KHTTPGET= command is used to make a GET request to the specified address of the HTTP server to which the modem is connected. This retrieves the data from the server.



```
startCo:
String connect_cmd = "AT+KHTTPCFG=1,\""+http_address+"\","+String(http_port)+",0,,,1\r";
// Configuration de la connexion avec le serveur HTTP
sendAT_HL7800( command: connect_cmd);
```

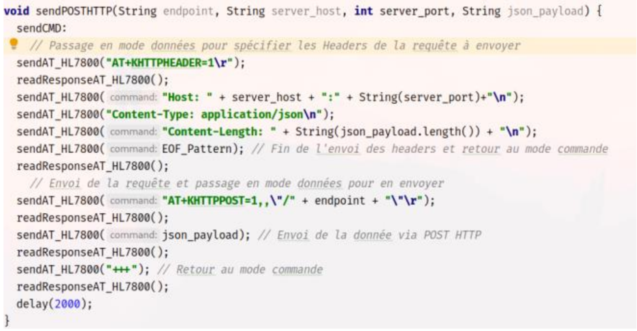
Figure 12. Extract from the connectHTTP function.



```
void sendGETHTTP(String endpoint){
    // Récupération du contenu d'une page HTTP
  sendAT_HL7800( command: "AT+KHTTPGET=1,\"" + endpoint + "\",1\r");
  readResponseAT_HL7800();
}
```

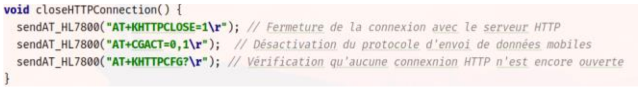
Figure 13. Extract from the sendGETHTTP function.

The AT+KHTTPHEADER= command is used to specify the headers of the HTTP POST request to be sent. For example, we specify the type of data we are sending and its size (in our case the data type is JSON). Then we use AT+KHTTPPOST= to send the HTTP POST message to the specified server address. As a result, the modem switches to data mode. We give it the data to send to the server. The "+++" then allows the modem to return to command mode, as shown in Figure 14. Finally, we close the HTTP connection (see Figure 15). This sequence of AT commands enables the HTTP GET and POST connection to the server to be fully executed.



```
void sendPOSTHTTP(String endpoint, String server_host, int server_port, String json_payload) {
  sendCMD:
   // Passage en mode données pour spécifier les Headers de la requête à envoyer
  sendAT_HL7800("AT+KHTTPHEADER=1\r");
  readResponseAT_HL7800();
  sendAT_HL7800( command: "Host: " + server_host + ":" + String(server_port)+"\n");
  sendAT_HL7800("Content-Type: application/json\n");
  sendAT_HL7800( command: "Content-Length: " + String(json_payload.length()) + "\n");
  sendAT_HL7800( command: EOF_Pattern); // Fin de l'envoi des headers et retour au mode commande
  readResponseAT_HL7800();
   // Envoi de la requête et passage en mode données pour en envoyer
  sendAT_HL7800( command: "AT+KHTTPPOST=1,,\"/" + endpoint + "\"\r");
  readResponseAT_HL7800();
  sendAT_HL7800( command: json_payload); // Envoi de la donnée via POST HTTP
  readResponseAT_HL7800();
  sendAT_HL7800("+++"); // Retour au mode commande
  readResponseAT_HL7800();
  delay(2000);
}
```

Figure 14. Extract from the sendPOSTHTTP function.



```
void closeHTTPConnection() {
  sendAT_HL7800("AT+KHTTPCLOSE=1\r"); // Fermeture de la connexion avec le serveur HTTP
  sendAT_HL7800("AT+CGACT=0,1\r");  // Désactivation du protocole d'envoi de données mobiles
  sendAT_HL7800("AT+KHTTPCFG?\r"); // Vérification qu'aucune connexnion HTTP n'est encore ouverte
}
```

Figure 15. Extract from the closeHTTPConnection function.

E. Study of energy consumption

Once the program has been uploaded to the OKKO card, the microcontroller initializes the modem. The modem then starts the TCP and HTTP communication described in the previous section. Once transmission is complete, the card goes into Deep Sleep mode until the next transmission request. As discussed earlier in this article, this mode saves battery power during periods of card inactivity.

The energy consumption of this behavior was measured using the OTII tool [14]. Table 1 shows the power consumption of the different execution phases of the card. As this table shows, the power consumption of the OKKO card in TCP mode is lower than in HTTP mode because the latter has more protocol load (header and connection establishment) to establish the connection between the modem and the server. We therefore recommend using the TCP protocol to send data from the card to the server via the LTE-M network. The table also shows the card's consumption in sleep mode. This consumption is equal to 32 micro-amperes, which guarantees long battery life.

Calculations and tests carried out at OKKO indicate a battery life of 5 years for a card that sends a single TCP message every 24 hours (with a 9000 mAh battery).

TABLE I. ENERGY CONSUMPTION OF THE OKKO CARD.

	Modem Initialization		**Communication**		**Sleeping mode**
	Average duration	**Average consumption**	**Average duration**	**Average consumption**	**Average consumption**
TCP	15 s	88 mA	22 s	96.2 mA	32 uA
HTTP	27 s	91 mA	106 s	128 mA	

IV. Conclusion

In this paper, we studied the performance of the LTE-M network for the Industrial Internet of Things, based on an in-depth literature review and a practical implementation on a prototype IoT terminal with energy constraints from the OKKO company. Thanks to the ESP32-S2 microcontroller and the LTE-M modem, we have succeeded in establishing TCP communications as well as HTTP communications (POST and GET), which contributes to the implementation of efficient, high-performance IoT solutions in industrial environments. The results show that the LTE-M network is perfectly suited to applications requiring low-speed transmission over a wide deployment area. These results open up a wide range of possibilities for the implementation of autonomous IoT systems in industry.

Acknowledgment

This paper is an exploratory study on IoT as part of the I2RM (21-SIOM-0007-02) project funded by the French National Research Agency.

References

[1] E. U. Ogbodo, A. M. Abu-Mahfouz, and A. M. Kurien, “A Survey on 5G and LPWAN-IoT for Improved Smart Cities and Remote Area Applications: From the Aspect of Architecture and Security,” Sensors, vol. 22, pp. 32132-32149, August 2022, doi:10.3390/s22166313.

[2] P. Levchenko, D. Bankov, E. Khorov, and A. Lyakhov, “Performance Comparison of NB-Fi, Sigfox, and LoRaWAN,” Sensors, vol. 22, pp. 1-21, December 2022, doi:10.3390/s22249633.

[3] M. A. M. Almuhaya, W. A. Jabbar, N. Sulaiman, and S. Abdulmalek, “A Survey on LoRaWAN Technology: Recent Trends, Opportunities, Simulation Tools and Future Directions,” Electronics, vol. 11, pp. 1-32, January 2022, doi:10.3390/electronics11010164.

[4] Bouygues Telecom LoRaWAN network shutdown: https://objenious.com/blog/technologie/arret-du-reseau¬lorawan-de-bouygues-telecom/. [retrieved: September, 2023].

[5] S. He, K. Shi, C. Liu, and B. Guo, “Collaborative Sensing in Internet of Things: A Comprehensive Survey,” IEEE Communications Surveys & Tutorials, vol. 24, pp. 1435-1474, June 2022, doi: 10.1109/COMST.2022.3187138.

[6] K. Mekki, E. Bajic, F. Chaxel, and F. Meyer, “A comparative study of LPWAN technologies for large-scale IoT deployment,” ICT Express, vol. 5, pp. 1-7, March 2019, doi:10.1016/j.icte.2017.12.005.

[7] J. Wang, M. K. Lim, C. Wang, and M. L. Tseng, “The evolution of the Internet of Things (IoT) over the past 20 years,” Computers & Industrial Engineering, vol. 155, pp. 1395-1413, May 2021, doi:10.1016/j.cie.2021.107174.

[8] K. Mekki, E. Bajic, F. Chaxel, and F. Meyer, “Overview of Cellular LPWAN Technologies for IoT Deployment: Sigfox, LoRaWAN, and NB-IoT,” The IEEE International Conference on Pervasive Computing and Communications Workshops (PerCom Workshops), March 2018, pp. 197-202, ISBN: 978-1-5386-3227-7.

[9] K. Nahrstedt, H. Li, P. Nguyen, S. Chang, and L. H. Vu, “Internet of Mobile Things: Mobility-Driven Challenges, Designs and Implementations,” The First IEEE International Conference on Internet-of-Things Design and Implementation (IoTDI), April 2016, pp. 25-36, ISBN: 978-1-4673-9948-7.

[10] L. Zemko and P. Čičák, “IoT and LPWAN Networks: Increasing Efficiency by Communication Planning,” The 45th IEEE International Conference on Telecommunications and Signal Processing (TSP), July 2022, pp. 78-85, ISBN: 978-1-6654-6948-7.

[11] C. Westerkamp, A. Grunwald, and M. Schaarschmidt, “LoRaWAN, NB IoT and other radio networks for agricultural applications,” The 26th IEEE ITG-Symposium Mobile Communication - Technologies and Applications, May 2022, pp. 61-67, ISBN:978-3-8007-5873-9.

[12] AT Commands Interface Guide for AirPrime HL78xx: https://source.sierrawireless.com/resources/airprime/software/hl78xx_at_commands_interface_guide/#sthash.2HvUJG26.dpbs. [retrieved: September, 2023].

[13] Thingsmobile, www.thingsmobile.com. [retrieved: September, 2023].

[14] Otii by QOITECH, www.qoitech.com. [retrieved: September, 2023].

Using Environmental Contexts to Model Restrictions on Sensor Capabilities

Martin Richter, Christine Jakobs, Theresa Werner, Matthias Werner
Operating Systems Group
Chemnitz University of Technology
09111 Chemnitz, Germany
email: {martin.richter, christine.jakobs, theresa.werner, matthias.werner}@informatik.tu-chemnitz.de

***Abstract*—Cyber-Physical Systems (CPS) incorporate the physical and digital worlds via sensors and actuators. The system devices may be heterogeneous, distributed in space, and mobile. This leads to new challenges in the design of applications that should utilize the full potential of the system. As devices are unreliable and may be arbitrarily moving away from locations of interest, there is a continuous necessity to replace them with alternatives during runtime. The handling of this task by the application programmer is error-prone and complex because the system might be heterogeneous and different sensors and actuators possibly possess varying means to observe and influence the environment. Therefore, a capability model has to be employed on the operating system level which provides a holistic view of the different devices. In this regard, an environmental context model has to be supplied as the devices' capabilities may be restricted depending on the contexts they are located in. This paper presents an environmental context model based on an abstract sensor capability model. In conjunction, the models provide a description of sensors concerning their ability to observe properties of physical objects of interest within their particular environmental contexts. The effect of contexts on the spatial interpretability of sensor measurements is therefore made explicit. Corresponding inferences are made with respect to the localization of physical objects or phenomena of interest.**

***Keywords*—cyber-physical systems; context awareness; heterogeneity; sensor virtualization.**

I. Introduction

Through emerging trends like the Internet of Things, Industry 4.0 or Smart Home, devices like sensors and actuators are of increasing importance in our daily lives. This leads to the emergence of Cyber-Physical Systems (CPS) that create a link between the physical and digital worlds. Such systems consist of devices that may be large in number, distributed in space, heterogeneous, unreliable, and mobile. As the target of CPS is the observation and influence of the physical world, the respective applications are often bound to certain devices situated in locations of interest. This severely impairs the portability of applications between systems. Additionally, the unpredictable motion and failure of sensors as well as actuators become a challenge for properly executing the applications. Furthermore, the system's full potential may go unrecognized as certain tasks may only be executable when taking the aggregated capabilities of multiple devices into account instead of considering them individually. The coordination of such groups of devices heavily depends on their environmental contexts as they may limit the devices' abilities to cooperate (e.g., robots being close to each other but being located on opposite sides of a wall). The described challenges lead to a demand for new device capability models that abstract the heterogeneous hardware such that applications are made portable and the full potential of CPS is utilized.

This paper provides an abstract sensor capability model. It incorporates the influence of environmental contexts on the ability of sensors to observe their surroundings. Additionally, it allows to describe which combinations of the available devices are suitable for performing a task with regard to their respective locations and environmental contexts. This enables the virtualization of hardware resources, i.e., the transparent utilization of a common set of devices by multiple applications.

Existing approaches to sensor virtualization (such as [1] and [2]) focus on concurrently executing multiple applications on a fixed set of sensors. In our view, the dynamics of the different devices have to be recognized. Due to the motion and failure of sensors, it is necessary to alternately execute applications on a changing set of devices such that they are able to continuously observe and influence the environment as desired. Our model enables the Operating System (OS) to decide which sensor measurements are required for a given task based on the programmer's task description, the sensors' capabilities, as well as the devices' environmental contexts. As the task description is detached from controlling the sensors, the execution of an application on transparently alternating sets of devices is accomplished. This leads to new possibilities for the task planning of sensors. Their mobility and heterogeneity can be exploited such that a leaving or failing device may be replaced by one or possibly multiple other devices that provide similar types of measurements. In that way, the application's possibilities for sensing and influencing the environment remain the same or may even be enhanced. Therefore, it is possible to unlock the full potential of CPS and to port applications to other systems that employ unrelated devices.

As a running example, we use a street surveillance system near a highway. Its goal is the detection of environmental hazards close to the road, such as wildfires. It consists of immobile cameras on the roadside as well as cars driving down the street which incorporate a dashboard camera as well as two temperature sensors for measuring interior and exterior temperatures respectively. The back seat windows of the cars shall be tinted such that cameras from the outside are not able to observe the back seat area of the car. The front seats of the car are observable through untinted windows. We assume that all sensors provide information on their respective locations and are able to communicate wirelessly.

The article is further structured as follows. Section II depicts the concept for an abstract sensor and environmental context model. Section III discusses existing approaches for virtualizing sensors and compares them to our model with respect to modeling the influence of environmental contexts on the capabilities of devices. Section IV presents the conclusion and possible directions for future work.

II. CONCEPT

The following sections present the programming model on which our sensor capability model is based, followed by the capability model, as well as the environmental context model.

A. Programming Model

The presented capability model is created in the context of the programming model introduced in [3]. The programming model provides sensor and actuator virtualization, as well as transparency with respect to distribution, location, and motion at the application level while allowing location- and motion-aware control of devices at the OS level. This is achieved by reversing the view of the CPS developer such that he or she takes a physical perspective. In his application, the programmer describes the properties of a physical object and its behavior depending on internal dynamics as well as potential actuator actions. The set of properties and their values form the state of the object, which may change over time. Additionally, the developer specifies a target state for the object via constraints. These constraints are then solved based on the state and behavioral descriptions of the object. The capability and context model we propose in this work is used to determine whether the available sensor capabilities with respect to the sensor contexts are sufficient for observing the object of interest and if so, in which locations it is present.

As mentioned before, the programmer provides a property description for the physical object of interest. For the object o it takes the form of a state description vector $\vec{z}_o$.

$$\vec{z}_o = \left[(\tau_1, r_1), \dots, (\tau_n, r_n)\right]^T \tag{1}$$

Each element of the vector consists of a tuple (τ_i, r_i). The first element denotes an object property type τ_i (e.g., color, shape, or temperature), which represents the domain of possible values for the corresponding property. The second element is a rule for specifying a range of values for the property (e.g., for a fire the color has to be red or yellow, and the temperature has to exceed 300 degrees Celsius). Each rule represents a logic formula that evaluates either to `true` or `false` for a given location based on the available sensor measurements. If all the rules for an object evaluate to `true` at a given location, the object is present there, otherwise it is assumed not to be.

B. Sensor Model

The provided programming model allows the developer to exclusively focus on the creation of the state description vector $\vec{z}_o$. The OS is responsible for utilizing the required sensors transparently. To achieve this, it requires knowledge of the sensors' capabilities as each of them may have different measurands and may observe varying physical phenomena. The following paragraphs provide an abstract model for the capabilities of a sensor.

A possibly mobile sensor at location $\vec{x} = \vec{x}(t)$ observes a physical quantity q (e.g., electromagnetic radiation or temperature). The measurement of the physical quantity is then transformed by the sensor into a digital signal $v = v(t)$ (e.g., an array of pixels) through a measuring process μ. The resulting signal is interpretable for different locations in space $X \subseteq X_\Sigma$, where X_Σ denotes all locations relevant to the system. The set of interpretable locations depends on the sensor's location $\vec{x}$ and possibly other sensor-specific parameters $\vec{p} = \vec{p}(t)$, which may also change over time (i.e., $X = X(\vec{x}, \vec{p})$). In conclusion, a sensor s is characterized by the following six-tuple.

$$s = (q, \vec{x}, v, \mu, \vec{p}, X(\vec{x}, \vec{p})) \tag{2}$$

To select sensors for given tasks efficiently it is necessary to group them into classes. The knowledge about the class of a sensor allows the OS to decide how to interpret the outputs of sensors (i.e., which actions to perform on them). It can then transform the results into instantiated physical object properties (e.g., shape or form). Sensors within a class measure the same physical quantity and their results can be utilized similarly. Therefore, the class of a sensor depends on its observed quantity q and its measurement process μ. We denote a sensor class γ as a boolean function that returns `true` for a sensor s_j belonging to the class and `false` otherwise.

$$\gamma_i(s_j) = \begin{cases} true, & \text{sensor } s_j \text{ belongs to class } \gamma_i \\ false, & \text{otherwise} \end{cases} \tag{3}$$

For example, digital dashboard cameras measure electromagnetic radiation within a certain wavelength and transform it into an array of pixels through their measurement process. On these arrays, methods like image recognition can be performed to extract desired object properties such as shape or color. The set of interpretable locations for a camera is described by the following formula for a cone, where $\vec{x} = \vec{x}(t)$ denotes the location of the sensor at time point t, $\vec{u}$ is the location vector for which the equation has to be solved, and the parameter vector $\vec{p}$ consists of the orientation of the camera $\vec{o} = \vec{o}(t)$ as well as its viewing angle ϕ.

$$X\left(\vec{x}, \begin{bmatrix}\vec{o} & \phi\end{bmatrix}^\top\right) = \left\{\vec{u} \in X_\Sigma : \frac{(\vec{u} - \vec{x}) \cdot \vec{o}}{|\vec{u} - \vec{x}||\vec{o}|} \leq \cos\phi\right\} \tag{4}$$

This equation could also be further constrained by considering the maximum range of the camera. A brightness sensor, in contrast, also measures electromagnetic radiation but through its measurement process its output can not be utilized for similar methods as the camera. Additionally, its measurement is solely interpretable for a small radius around its location.

C. Sensor Identification

Depending on the object description presented in Section II-A, multiple different sensors may have to be utilized to gather data on an object of interest. The information on which methods may be used to extract object properties from sensor outputs and which classes of sensors are required to utilize them is managed in a dictionary $\mathcal{D}$. The dictionary takes as inputs:

1) the set of all sensors $S(t)$, which may change over time due to motion or failure,
2) the set of all available methods M, which may be applied to different classes of sensors, and
3) the type τ of the physical property z_i.

As a result of these inputs, the dictionary returns a vector of different methods $\vec{m}$. They can be applied to the outputs of sensors of the corresponding classes to calculate a value for the physical property.

$$\mathcal{D}(S(t), M, \tau) = \vec{m} \tag{5}$$

For example, for determining the shape of an object image recognition methods may be used on the outputs of multiple cameras or sophisticated laser scanning systems may be utilized.

Each method m_j in $\vec{m}$ is a tuple consisting of a set of sensor classes Γ_j^τ and a function ψ_j for calculating the physical property's value from the outputs of sensors of the corresponding classes.

$$m_j = (\Gamma_j^\tau, \psi_j) \tag{6}$$

For each of the sensor classes in Γ_j^τ an individual sensor has to be chosen for providing the inputs to the corresponding calculation function ψ_j. All possible sets of input sensors are determined by a function g. It maps the currently available sensors $S(t)$ and the required classes of sensors Γ_j^τ to the corresponding sets of sensors Θ_j, which are a subset of the power set of $S(t)$.

$$g(S(t), \Gamma_j^\tau) = \Theta_j, \Theta_j \subset \mathcal{P}(S(t)) \tag{7}$$

The dimensions of each set of sensors $\theta_i \in \Theta_j$ depend on how many sensors are required by the method m_j. For example, for the calculation of the position of an object either one distance sensor with known location and orientation is required or several cameras may be utilized through the method of triangulation.

When the function ψ_j is applied to the set of chosen sensors θ_i a value v of type τ is created.

$$\psi_j(\theta_j) = v \tag{8}$$

This value v may correspond to the value of a physical object property depending on whether the corresponding rule r in the state description vector is satisfied or not.

D. Environmental Context Model

Not all sensors that belong to the required classes for performing a method can be utilized for the method. Different sensors may be located in different environmental contexts such that their measurements do not stand in any relationship with each other. An example of this is an interpolation method that can not be used simultaneously for temperature sensors in a car and temperature sensors on the roadside.

An environmental context constrains the area for which the sensors' output can be interpreted. Each context c is defined with respect to its influence on a physical quantity q (e.g., a closed window influences the interpretability of temperature sensors but not of cameras). A context possesses an anchor location $\vec{x}(t)$ which may change over time. This location is continuously updated, e.g., by the use of a positioning sensor. Depending on its anchor location the context additionally consists of a set of surrounding locations $X(\vec{x}(t))$ which describe the spatial extent of the context. A context might inhibit the interpretation of sensor measurements of the corresponding physical quantity q in two ways:

1) It may impede the interpretation of a measurement from outside the context for a location inside the context.
2) It may impede the interpretation of a sensor measurement within the context for a location outside the context.

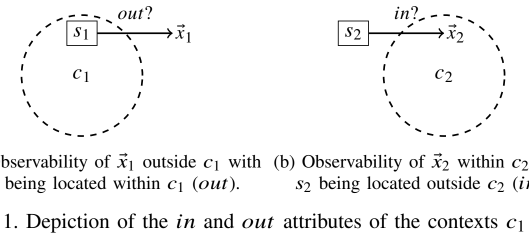
For example, a camera located inside a vehicle may be able to observe locations outside through a tinted window but a camera located outside the vehicle is not able to observe its inside through the same window. This is accounted for by introducing the boolean attributes *in* and *out* for each context. If *in* is `false`, the context constrains the interpretation of sensor measurements from outside the context for locations inside the context. If it is `true` an interpretation is possible. The attribute *out* describes whether the interpretation of a measurement from inside a context is feasible for a location outside in a similar fashion. Figure 1 depicts this situation. In conclusion, a context c related to a physical quantity q is defined by the following quintuplet.

$$c = c(t) = \{q, \vec{x}(t), X(\vec{x}(t)), in, out\}\,,\ X(\vec{x}(t)) \subseteq X_\Sigma \tag{9}$$

The locations for different contexts may overlap arbitrarily. Therefore, a sensor may be influenced by multiple contexts at once. As the sensor's location and the contexts' locations may change over time, the set of contexts that influence a sensor may also change over time.

E. Impact of Contexts on Sensor Measurements

For a given sensor s, the set of interpretable locations $s.X$ is created without taking its environment into account (see Section II-B). The set of locations the sensor is actually able to observe may be smaller because it is constrained by the set of environmental contexts C. The measurement of a sensor can be restricted by a context in two ways, as described in Section II-D. Therefore, the locations X_{obs}^s a



(a) Observability of $\vec{x}_1$ outside c_1 with s_1 being located within c_1 (out).

(b) Observability of $\vec{x}_2$ within c_2 with s_2 being located outside c_2 (in).

Figure 1. Depiction of the in and out attributes of the contexts c_1 and c_2 regarding the sensors s_1 and s_2 that may or may not measure the locations $\vec{x}_1$ and $\vec{x}_2$ respectively.

sensor observes from within its current contexts depend on two sets of locations $X^s_{\neg out}$ and $X^s_{\neg in}$. The set $X^s_{\neg out}$ denotes the locations of contexts a sensor is located in which do not allow the measurement of the sensor to be interpreted for locations outside (i.e., $\neg c.out$).

$$X^s_{\neg out} = \bigcap_{\substack{c \in C, c.q = s.q,\\ \neg c.out, s.\vec{x} \in c.X}} c.X \tag{10}$$

Thus, the locations $s.X$ a sensor may be able to measure have to be intersected with $X^s_{\neg out}$. This removes all locations from $s.X$ which lie outside of the corresponding contexts. From the resulting set the locations $X^s_{\neg in}$ have to be removed. They refer to contexts in which the sensor is not located and which prohibit the interpretation of the sensor's output from outside the context for a location within the context (i.e., $\neg c.in$).

$$X^s_{\neg in} = \bigcup_{\substack{c \in C, c.q = s.q,\\ \neg c.in, s.\vec{x} \notin c.X}} c.X \tag{11}$$

In conclusion, the resulting set of locations X^s_{obs} a sensor is able to observe is defined by:

$$X^s_{obs} = (s.X \cap X^s_{\neg out}) \setminus X^s_{\neg in}. \tag{12}$$

Figure 2 shows an example for this. The camera sensor s is able to measure a cone-shaped area in front of it and is located within the contexts c_1 and c_2. Therefore, the *out* attributes of these contexts are relevant for the spatial interpretation of the output of s. Only c_2 constrains the sensor's measurement locations in this regard. The sensor is not located within c_3 and c_4. Thus, their respective *in* attributes are of relevance as both contexts consist of locations that may intersect with the sensor's observable locations $s.X$. The locations X^s_{obs} for which its measurements are interpretable are represented by diagonal stripes.

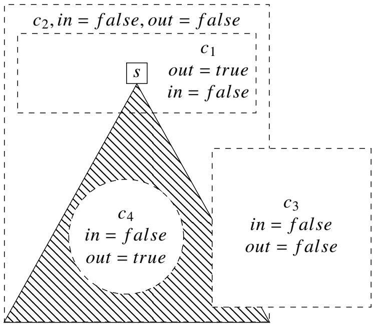


Figure 2. Depiction of observable locations X^s_{obs} for a sensor s with respect to the contexts c_1, c_2, c_3, and c_4 that relate to the same physical quantity s is measuring.

The set of context regions X^s_{con} for a sensor s is defined similarly to X^s_{obs}.

$$X^s_{con} = (X_\Sigma \cap X^s_{\neg out}) \setminus X^s_{\neg in} \tag{13}$$

It denotes the entirety of the sensor's context without taking its observable locations $s.X$ into account. This set allows to analyze whether the measurements of different sensors stand in relation to each other, i.e., whether their context regions overlap.

The set $X^{\theta^{m_i}}_{obs}$ denotes the sets of observable locations for a chosen set of sensors $\theta_{m_i} \in \Theta_i$ utilized by method m_i.

$$X^{\theta^{m_i}}_{obs} = \{s \in \theta_{m_i} : X^s_{obs}\} \tag{14}$$

The set $X^{\theta^{m_i}}_{con}$ depicts the sets of context regions for a chosen set of sensors similarly to $X^{\theta^{m_i}}_{obs}$.

$$X^{\theta^{m_i}}_{con} = \{s \in \theta_{m_i} : X^s_{con}\} \tag{15}$$

The scope $\mathcal{S}_{m_i}$ for the result of a method m_i is determined by a function α.

$$\mathcal{S}_{m_i} = \alpha(X^{\theta_{m_i}}_{obs}, X^{\theta^{\theta_{m_i}}}_{con}, \psi) \tag{16}$$

The scope describes which locations in the system can be covered by the corresponding method by utilizing varying sets of sensors. If the scope is an empty set, the method is not applicable to the available sensors in the given contexts. It depends on the following three parameters:

1) the different sets of observable locations $X^{\theta^{m_i}}_{obs}$ of the utilized sensors as they determine for which locations valid sensor outputs are available,
2) the different sets of context regions $X^{\theta^{m_i}}_{con}$ of the utilized sensors because they describe which sensor measurements stand in relation to each other and may be used for methods like interpolation for example, and
3) the function ψ for calculating the result of the method as varying calculation functions may lead to different results, e.g., interpolation between temperature sensors increases the scope of a method and triangulation between cameras reduces it.

F. *Object Identification*

As described in Section II-A, a physical object property z is computable by a set of different methods. Each of these methods provides a result for a given scope based on which sensors are chosen for its execution. Depending on the programmer's target, it may be feasible to utilize multiple different methods to increase the area in which an object's property can be observed. For a chosen set of methods $\vec{m}_\tau$ for determining the physical object property z of type τ, z is observable in the set of locations X^z which is a union of the utilized methods' scopes.

$$X^z = \bigcup_{m_i \in \vec{m}_\tau} \mathcal{S}_{m_i} \tag{17}$$

The physical object o (i.e., all its properties) is observable at the intersection of the observed locations of all its properties.

$$X^o_{obs} = \bigcap_{z_i \in \vec{z}} X^{z_i} \tag{18}$$

The actual spatial scope X^o of the object is determined by the rules r_i provided by the state description $\vec{z}_o$ (see Section II-A).

$$X^o = \bigcap_{\substack{(\tau_i, r_i) \in \vec{z}_o, \vec{m} = D(S, M, \tau_i),\\ m_j \in \vec{m}, v = \psi_{m_j}(\theta_{m_j}),\\ r_i(v) = true}} X^{z_i} \tag{19}$$

The scope of an object may encompass multiple regions of space which are detached from each other (e.g., multiple fires being localized by the street surveillance system). They are interpreted as distinct instances o_i of the same object type with a set of multiple unconnected regions $X^{o_i} \subseteq X^o$ as a result.

For each of the objects o_i resulting from this interpretation, an instance vector $\vec{v}$ is calculated for the corresponding scope X^{o_i} by applying the according methods, i.e., applying their calculation functions $m_j.\psi$ to a chosen set of sensors θ^{m_j}.

$$\vec{v}_{X^{o_i}}(t,\vec{x}) = \vec{\psi}(\vec{\theta}) = \begin{bmatrix} m_1.\psi(\theta^{m_1}) \\ \dots \\ m_q.\psi(\theta^{m_q}) \end{bmatrix}, \ \vec{x} \in X^{o_i} \tag{20}$$

Each element v_i of this vector provides a value of type τ_i for the corresponding element z_i of the state vector in the object description of an object instance o_i in the scope X^{o_i}. The property value may also change within the scope over time and space. Therefore, it depends on time t as well as space $\vec{x}$. This vector allows monitoring the state of the physical object of interest and changes in its location within its observable scope.

III. Related Work and Discussion

This section discusses related work on incorporating environmental contexts into sensor capability models. Thereafter, the presented approaches are compared to our model.

A. Related Work

As mentioned in [4], the current state of the art is to utilize ontologies for describing the abilities of a sensor and the environmental context it observes (i.e., what it is measuring). In [5] and [6], different ontologies for the semantic specification of sensors are surveyed. They allow interpreting the sensor measurements with respect to their measurands and in which contexts their observations are made. These approaches are static in the sense that the measurands and environmental contexts are directly bound to sensors which does not take the motion of sensors into account. Additionally, the influence of contexts on the sensor capabilities is not made clear as the corresponding measurements are tagged but no adjustments to the specifications of a sensor's capabilities are made. Therefore, the developer has to manually infer the capabilities of the sensors depending on which contexts they are located in. Potential operating systems employing these techniques would therefore be obstructed in their ability to dynamically schedule devices suitably for the tasks at hand.

In [1], an approach to sensor virtualization is described which allows a set of applications to concurrently utilize a common set of sensors. The environmental contexts and capabilities of sensors are not taken into account in this model. Therefore, the choice of which devices to utilize during the execution of the application still lies with the programmer. This introduces room for errors as some sensors may be located in contexts that do not allow to measure the desired environmental entities.

In [7], a sensor virtualization method is presented. It allows the developer to declaratively specify a virtual sensor with respect to which behavior it has to implement. Based on their capabilities, changing physical devices are chosen during runtime for the execution of the application's tasks. This approach allows utilizing possibly changing sets of sensors and also accounts for the motion of devices. The contexts of sensors are not taken into consideration. Thus, an improper selection of devices may lead to an application utilizing data that does not suit the context for which it was developed.

In [2], virtual sensors are created such that each one wraps a physical device. A virtual sensor provides a service that can then be used by multiple different applications. Therefore, the sensors are bound to their respective applications such that taking a changing set of devices transparently into account is not feasible. Additionally, contexts are not considered which leads to similar challenges as described above.

In [8], an ontology is provided which allows the developer to create individual capability models for each sensor. These models are utilized to generate code for the distinct devices which makes their functionalities available to the system. This approach is most suited for systems with a static set of sensors as individual capability models have to be created and the corresponding code needs to be generated for each sensor. Transparently scheduling a changing set of devices is therefore only feasible in systems with an a priori known set of sensors. The importance of taking the sensor contexts into account is mentioned but not further elaborated on.

In [9], an abstraction for sensors and actuators is presented, called resource, which allows the programmer to declarative describe which devices are required by his application. If a device is not available, a new resource may be created by coupling a set of different devices which in combination are able to perform similar actions as the unavailable resource. Therefore, a transparent concurrent utilization of different sensors based on the task at hand is possible. The contexts of the devices are taken into account for the querying process but they are statically bound to sensors and actuators. Thus, the mobility of devices (i.e., changing contexts) is not considered which may lead to an erroneous allocation of mobile sensors or actuators depending on their change of position.

In [10] and [11], two approaches are presented which integrate the notion of contexts into the programming model. The developer binds parts of his application to different contexts based on which devices are available within them. Their models allow to transparently utilize different sets of sensors or actuators based on where they are located. Neither sensor capabilities nor how their environmental contexts influence them are taken into account, which leads to similar drawbacks as described above.

In [12]–[14], varying approaches for modeling contexts of sensor systems are surveyed. The examined propositions' primary focus lies on annotating sensor data such that they can be interpreted with respect to their environment. None of these approaches describe the influence of contexts on the capabilities of sensors. Rather, they enrich the sensor data with context information such that the programmer has to make the correct choice of which contexts are of relevance and which sensor measurements have to be gathered within them. This introduces room for errors as the effect of contexts on the capabilities of sensors is not handled explicitly. Additionally, the developer may not be able to utilize all devices as a sensor located within a given context may still be able to provide measurements for locations outside of a context.

In [15], an approach to defining contexts onthologically is presented. The focus lies on documenting the impact of different contexts (i.e., developmental, behavioral, structural, and functional contexts) on the development of the CPS. This allows the programmers to consider challenges emerging from changes of the system's contexts during runtime. While the presented model is aiding developers in designing the system, it does not provide runtime support for transparently managing devices based on the context descriptions.

B. Discussion

None of the presented approaches discuss the influence of environmental contexts on the capabilities of sensors. Therefore, performing tasks like sensor fusion on measurements that may not relate to each other due to environmental constraints may create erroneous results. The virtualization of devices is affected similarly. The replacement of a failed sensor by one or more other devices is only feasible if their measurements are related.

Our model describes a solution to this challenge by taking the influence of environmental contexts into account. They are defined as regions in space that restrict the interpretable locations of sensor measurements. This allows to decide which measurements are related to each other. A context-aware virtualization of the devices is therefore enabled with respect to which devices possess the capabilities to perform a task.

For performance reasons, it may be required to restrain the definition of contexts in our model to discrete sets of locations in the form of predefined geometric shapes instead of arbitrary continuous regions in space (as in [11] for example). This allows the efficient computation of the required operations on the different sets of locations as described in Section II-D. The approaches discussed in [7], [10], and [11] provide performant implementations which enable sensor virtualization without taking contexts into account. Therefore, they are suitable as a starting point for implementing our model.

IV. Conclusion and Future Work

This paper presents a model for describing the influence of environmental contexts on the capabilities of sensors. This is necessary since a sensor's ability to observe its environment is strongly impacted by its surroundings. Contexts are defined as sets of locations such that they describe regions in space. They are viewed as constraints on the ability to interpret the measurements of a sensor for a given location. These constraints are effective at the edges of contexts, such that they influence the observable locations of sensors within or outside of the context. Our sensor capability and context model makes the influence of environmental contexts on the available sensors explicit.

The model is presented in the context of identifying physical objects based on an object description provided by the programmer. It allows to reason about the influence of choosing different sets of sensors on the coverage of the system space with respect to the object properties to be measured. This allows the OS to choose devices according to their ability to observe locations of interest. Therefore, a virtualization of sensors is achieved. This allows to transparently execute an application on alternating sets of heterogeneous devices while ensuring that locations of interest are observed.

For future work, we intend to enrich the model with further metrics for allowing an optimal choice of sensors based on their capabilities. The adaptation of contexts via exploration during runtime based on available sensor measurements is also a future goal. Additionally, an implementation and integration of the model into the presented programming model is intended.

References

[1] C. Mouradian *et al.*, "Network functions virtualization architecture for gateways for virtualized wireless sensor and actuator networks," *IEEE Network*, vol. 30, no. 3, pp. 72–80, 2016.

[2] P. Evensen and H. Meling, "Sensewrap: A service oriented middleware with sensor virtualization and self-configuration," in *International Conference on Intelligent Sensors, Sensor Networks and Information Processing (ISSNIP)*, 2009, pp. 261–266.

[3] M. Richter, T. Werner, and M. Werner, "A Programming Model for Heterogeneous CPS from the Physical Point of View," in *The Sixteenth International Conference on Mobile Ubiquitous Computing, Systems, Services and Technologies*, 2022, pp. 1–6.

[4] N. Sahlab, N. Jazdi, and M. Weyrich, "Dynamic context modeling for cyber-physical systems applied to a pill dispenser," in *25th IEEE International Conference on Emerging Technologies and Factory Automation (ETFA)*, 2020, pp. 1435–1438.

[5] G. M. Honti and J. Abonyi, "A review of semantic sensor technologies in internet of things architectures," in *Complexity*, 2019, pp. 1–21.

[6] M. Compton, C. A. Henson, L. Lefort, H. Neuhaus, and A. P. Sheth, "A survey of the semantic specification of sensors," in *CEUR Workshop Proceedings*, 2009, pp. 17–32.

[7] S. Kabadayi, A. Pridgen, and C. Julien, "Virtual sensors: abstracting data from physical sensors," in *International Symposium on a World of Wireless, Mobile and Multimedia Networks*, 2006, pp. 586–592.

[8] O. Lemaire, K. Ohba, and S. Hirai, "Dynamic integration of ubiquitous robotic systems through capability model processing," in *SICE-ICASE International Joint Conference*, 2006, pp. 1207 – 1211.

[9] V. Tsiatsis *et al.*, "The sensei real world internet architecture," in *Towards the Future Internet: Emerging Trends from European Research*, 2010, pp. 247–256.

[10] Y. Ni, U. Kremer, and L. Iftode, "Spatial views: Space-aware programming for networks of embedded systems," in *Lang.s and Compilers for Parallel Comput.*, 2004, pp. 258–272.

[11] C. Borcea, C. Intanagonwiwat, P. Kang, U. Kremer, and L. Iftode, "Spatial programming using smart messages: design and implementation," in *24th Int. Conf. on Distrib. Comput. Syst.*, 2004, pp. 690–699.

[12] C. Bettini *et al.*, "A survey of context modelling and reasoning techniques," in *Pervasive and Mobile Computing*, 2010, pp. 161–180.

[13] C. Perera, A. Zaslavsky, P. Christen, and D. Georgakopoulos, "Context aware computing for the internet of things: A survey," in *IEEE Communications Surveys & Tutorials*, 2014, pp. 414–454.

[14] T. Strang and C. Linnhoff-Popien, "A context modeling survey," in *First International Workshop on Advanced Context Modelling, Reasoning And Management at UbiComp*, 2004, pp. 34–41.

[15] M. Daun and B. Tenbergen, "Context modeling for cyber-physical systems," *Journal of Software: Evolution and Process*, vol. 35, no. 7, p. e2451, 2022.

CRDT-based Collaborative Editing in OppNets: a Practical Experiment

Yves Mahéo
Irisa, Université Bretagne Sud
Vannes, France
email: yves.maheo@univ-ubs.fr

Frédéric Guidec
Irisa, Université Bretagne Sud
Vannes, France
email: frederic.guidec@univ-ubs.fr

Camille Noûs
Laboratoire Cogitamus
Vannes, France
email: camille.nous@cogitamus.fr

***Abstract*—Developing distributed applications for Opportunistic Networks (OppNets) has mainly relied on the message passing paradigm so far. Yet, the sharing of distributed data is an alternative worth considering, and Conflict-Free Replicated Datatypes (CRDTs) are interesting candidates for this purpose. A CRDT is a distributed data type that supports optimistic replication, and whose global consistency can be maintained based solely on occasional synchronizations of replicas. In an OppNet, these synchronizations can be driven by the contacts between mobile devices, without any need for network-wide routing. CRDTs can thus serve as interesting software building blocks to develop distributed applications for OppNets. In this paper, we demonstrate the feasibility of using CRDTs in OppNets by presenting an experiment conducted in real conditions, involving a collaborative editing application in which communication relies exclusively on opportunistic contacts between laptops. The software elements used in the experiment consist mainly of a CRDT-based text editor, and a communication module that supports the synchronization between laptops, so as to ensure the eventual consistency of the shared document. Experimentation results are detailed, that confirm the viability and usability of this approach.**

***Keywords*—Opportunistic networking; Experiment; CRDT; Collaborative editing.**

I. INTRODUCTION

Opportunistic Networks (OppNets) are infrastructure-less networks composed of mobile devices that communicate via direct device-to-device radio transmissions. Due to the sparse or irregular distribution of the devices, such networks are often partitioned, and do not provide permanent end-to-end connectivity. Yet, network-wide communication is made possible following the store, carry and forward principle: when a contact occurs between two devices, the opportunity to exchange messages can be seized. The received information can be stored locally, so that the device that carries it can forward it later to another mobile device when the opportunity of a new contact presents itself. Developing applications for OppNets is challenging because of the inherent asynchronism of the communications, with potential large delays needed for reaching certain devices, from one to the next, according to their movements.

A number of use cases have been envisaged for Opportunistic Networks, namely when traditional network infrastructure is not available (e.g., communication for disaster relief or in remote areas) or when it is not desirable to use this infrastructure (e.g., for data offloading, or to avoid censorship).

To support such use cases, the research activity over the last two decades has primarily focused on message routing or dissemination in an OppNet, assuming that network-wide message passing is a de facto requirement of any distributed application, and assuming somehow that a message-oriented API is the best API for developers.

Yet, just like the High Performance Computing community uses alternatively the message-passing paradigm (typically via the Message Passing Interface [1]), or the shared-memory paradigm (via OpenMP [2] for example), depending on application needs, the developers of distributed applications for OppNets should not be bound to rely on a message-passing API, but they should also be offered the means to work with shared data structures.

A promising class of distributed shared data structures, called Conflict-Free Replicated Datatypes (CRDTs), has appeared recently, stemming from research on distributed databases and peer-to-peer networks [3]. CRDTs are distributed data types (counters, sets, maps, etc.) that support optimistic replication: replicas can be updated locally without any coordination, and synchronized asynchronously. CRDT replicas may temporarily diverge, but information is exchanged asynchronously between them thanks to a synchronization algorithm running in the background. This algorithm guarantees that all replicas eventually reach the same final state, provided the synchronization graph is connected, that is, provided the history of successive synchronizations is such that any update is eventually taken into account by every replica.

In the literature, the papers dealing with CRDTs in OppNets present simulation results [4]-[8]. The question whether CRDTs can be of practical use in a real OppNet setting remains unanswered. The objective of this paper is to contribute to answer this question. It describes the different elements of the setting and the results of a real-world experiment carried out in order to assess the possibility to write a document collaboratively, by relying on the implementation of a CRDT deployed in an OppNet. To our knowledge, it is the first time that the use of a real application involving CRDTs in an OppNet is reported. The choice of collaborative editing as a case study is motivated by the fact that it is a demanding distributed application, for respecting the causality of editing operations is not trivial in an OppNet. Besides, several tested off-the-shelf software elements designed for peer-to-peer wired networks can be reused or adapted for OppNets, which makes it easier to develop a robust solution.

Figure 1 illustrates the kind of OppNet we consider: the

ISBN: 978-1-68558-106-0

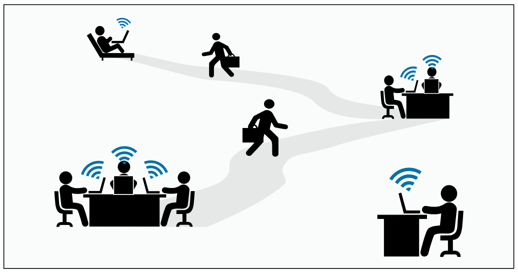


Figure 1. Illustration of collaborative editing in an opportunistic network.

contributors to the shared document use their laptops over a few days to edit the shared document collaboratively. Laptops are considered here as target devices (rather than smartphones or tablets), because they are more convenient to use for text editing.

A contributor may be isolated while editing the document (e.g., at home), in which case no transmission is possible between his/her laptop and those of other contributors. When several contributors are close to one another (e.g., in adjacent rooms at work), their laptops can synchronize by exchanging messages via direct wireless connections. When a user is on the move, his/her laptop is assumed to be switched off.

The remainder of this paper is organized as follows. Section II introduces the concept of CRDT and details the problem of their synchronization, in particular in OppNets. Section III quickly browses the works related to distributed applications targeting OppNets, and in particular on collaborative applications like text editing. In Section IV, we present the different software elements that support the text editing experiment we have set up. The results of this experiment are detailed in Section V. Section VI concludes the paper.

II. Overview of CRDTs

When a data structure must be replicated in a distributed system, there are different ways to maintain the consistency of the replicas of this data structure. Some systems maintain strong consistency at any time by constraining concurrent updates of replicas, or preventing them altogether. Other systems implement optimistic replication, allowing replicas to diverge temporarily, while ensuring that they will eventually reach a common state (eventual consistency).

Conflict-free Replicated Data Types (CRDTs) support optimistic replication: any replica can be updated locally, at any time, without any coordination with the other replicas. Synchronization occurs in the background, usually periodically and between randomly selected pairs of replicas, by exchanging information about past updates. If the synchronization graph is connected (i.e., the consequence of each update is eventually taken into account by each replica), all replicas eventually reach the same state.

A. Concurrent updates and concurrency semantics

The implementation of traditional data types (counters, registers, sets, maps, lists, graphs, etc.) as CRDTs has already been addressed in the literature [3][9]. For each CRDT, the updates it can support are identified, and a concurrency semantics is defined. Note that several alternative concurrency semantics can often be defined for the same abstract data type, as shown below.

In order to illustrate how a CRDT can be used as a shared data structure, let us consider a basic example. A Set CRDT implements a shared set, and it can typically support updates *add()* and *rmv()*. Since these updates cannot commute when they are applied to the same element, a concurrency semantics must be defined to resolve conflicts between concurrent *add(x)* and *rmv(x)* updates. Figure 2 shows an example where a Set CRDT (initially empty) is replicated in two replicas R1 and R2. Element *a* is first added locally to the set in replica R1, while element *b* is added locally in replica R2. The state is thus temporarily different in R1 and R2, but a synchronization occurs between them, after which they agree that the state of the set is now {a, b}. Note that reconciling state {*a*} (from R1) with state {*b*} (from R2) is not an issue, because although *add(a)* and *add(b)* occurred concurrently in R1 and R2, they apply to distinct elements *a* and *b*.

After the first synchronization, element *a* is removed and then added again in R1, while it is only removed in R2. The state is thus different again in R1 and R2, and this time the last *add(a)* on R1 and *rmv(a)* on R2 conflict, as they occurred concurrently *and* apply to the same element *a*. If both replicas synchronize again, the final state depends on the concurrency semantics chosen for this shared set. A possible option is to give *add(a)* priority over *rmv(a)*, so that the final state is {*a, b*} in both replicas. Another option is to give *rmv(a)* priority over *add(a)*, so the final state is {*b*}. A Set CRDT that gives *add()* priority over *rmv()* is called an Add-wins set in the literature, and the opposite is called a Remove-wins set [3].

B. Synchronization of replicas

In the simple example shown in Figure 2, only two replicas are considered, so synchronization is only required between these two replicas. In a system that involves a large number of replicas, synchronization must be addressed with caution in order to guarantee that all replicas eventually converge, while maintaining the cost of synchronization at a reasonable level.

Several methods of synchronization have been considered in the literature, each method requiring a specific implementation of CRDTs. In an operation-based CRDT, whenever an operation (update) is applied to a replica, a description of this operation is embedded in a message, which is sent to all other replicas. This approach tends to produce a large number of small messages (each message carrying information about a single update). Besides it requires a system that supports reliable network-wide broadcast, and even causal broadcast if the updates do not commute [3].

In a state-based CRDT, each replica must synchronize periodically with other replicas by sending them its entire state. On each receiver the state of the sender is merged with the local state, using a function that deterministically computes the join (least upper bound) of both states. A major advantage of this approach is that is does not require that each update

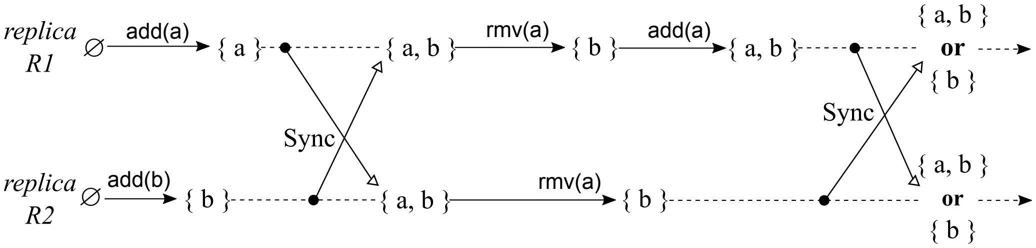


Figure 2. Example of a run involving a Set CRDT replicated in two replicas R1 and R2.

be transmitted to all replicas, so no broadcast is required. A periodic synchronization of each replica with a few other replicas is sufficient to ensure the eventual convergence of all replicas, as long as the synchronization graph is connected. The main drawback of state-based CRDTs is that shipping entire states between replicas can be costly. Delta-state CRDTs reduce this cost by passing only partial information (a *delta*) about the sender's state (typically, only what is required to allow the target receiver to update its own local state) [10]-[12]. In order to determine what is required by another replica, the sender must first receive a digest of this replica's state, and compare this digest with its own local state. The digest of a state can typically take the form of a state vector.

C. CRDTs in OppNets

In the literature, it is commonly assumed that CRDTs are to be deployed in Internet-based peer-to-peer networks. Network-wide message routing or broadcast are thus presumed to be available. In such conditions, each replica can either 1) send each update to all other replicas (in an operation-based CRDT), or 2) select randomly any other replica and synchronize with this replica (in a state-based or delta-state-based CRDT).

In an OppNet, two mobile devices can only communicate as long as they are neighbors. Implementing operation-based CRDTs, which involves sending each update to all other devices, is therefore not trivial as it requires at least epidemic delay-tolerant dissemination. As for state-based and delta-state-based CRDTs, they should rely on contact-driven synchronization, each device synchronizing with its neighbors rather than with randomly selected peers.

Synchronization algorithms for operation-based, state-based, and delta-state-based CRDTs in OppNets have been proposed and evaluated based on simulation scenarios in [4]. The results show that although all synchronization methods ensure the eventual consistency of CRDTs, delta-state-based synchronization clearly outperforms the two other modes of synchronization. Operation-based synchronization is easy to implement on top of an opportunistic communication layer that supports reliable causal broadcast. However, it requires the network-wide dissemination of many small messages (one for each update applied to a replica). This can yield a significant communication overhead in an OppNet, as well as storage issues since each message must be maintained in a local cache on each device. State-based and delta-state-based synchronization can be implemented without any multi-hop routing or network-wide dissemination, using only synchronization between neighbors. The cost of state-based synchronization is significant, though, as it requires exchanging entire states between neighbor devices. Delta-state-based synchronization gets the best of both other methods, as the amount of data transfers required to ensure the synchronization of replicas is kept at a minimum, and as there is no need for message routing or message broadcast.

The work presented in [4] has shown that CRDTs can be deployed in an OppNet, and converge as expected in such an environment ; but the given results have been obtained by running simulations. Whether distributed applications based on CRDTs can be of practical use in a real OppNet setting is still to be demonstrated. The purpose of this paper is to contribute to this task.

III. Related work

Running distributed applications in OppNets has been considered in many papers over the last two decades, but, again, most of these papers only present simulation results. Rare are the papers that present communication systems and applications that have been fully implemented, and tested in real conditions. Among these exceptions are [13] and [14], which present DTN systems aiming at providing Internet-like services in very sparsely populated areas, or in disaster-relief scenarios. Distributed applications for content sharing (files, music, news, software components, etc.) in OppNets have likewise been presented in [15]-[18].

In the abovementioned applications, the content shared over the network is considered as immutable. In contrast, collaborative editing (or, more generally, collaborative work) requires to share content that can change over time.

Although Web-based solutions such as wikis, Google Docs, Etherpad, etc. have been available for a long time now, these solutions usually rely on a client-server architecture, with central servers whose role is to store shared documents and ensure that concurrent editing of the same document does not yield inconsistencies.

Recognizing that any solution involving servers is hardly applicable in OppNets, an early solution for shared content editing in such networks has been proposed in [19]. In this proposal, a revision control mechanisms is used to merge contributions whenever possible, but user intervention is still required to solve conflicting contributions. In [20], the problem

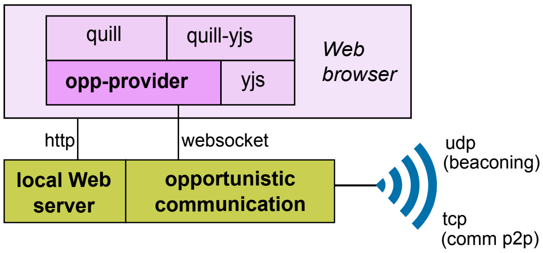
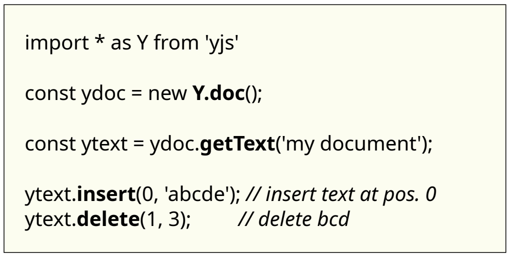


Figure 3. Software elements deployed on each laptop (the elements we have developed are named in boldface).



```
import * as Y from 'yjs'

const ydoc = new Y.doc();

const ytext = ydoc.getText('my document');

ytext.insert(0, 'abcde'); // insert text at pos. 0
ytext.delete(1, 3);        // delete bcd
```

Figure 4. Example of the use of a text CRDT via the Javascript Yjs module.

of ensuring total order in OppNets is considered, as a means to support a variant of the Logoot [5] replication algorithm in such networks. A similar approach is presented in [6], using IBR-DTN [21] for opportunistic communication, and a modified LogootSplit [7] algorithm for ensuring consistency in shared content editing. An approach based on OT (Operational Transformation) is considered in [8], and the convergence of an OT-based collaborative editing framework in opportunistic networking scenarios is investigated. Unfortunately, all these papers only present simulation results, and in most cases it is unlikely that a fully-functional collaborative editing solution exists beyond the simulation code.

In the remainder of this paper we describe the implementation of such a fully-functional solution, and demonstrate that it is viable for actual collaborative editing in opportunistic networking conditions.

IV. Design and implementation

The key feature of our experiment is the use of a Conflict-Free Replicated Datatype to guarantee the consistency of the different contributions to the shared document. In our experimental setup, several software elements have been assembled to allow CRDT-based collaborative editing, forming two main layers (see Figure 3). The upper layer runs in a Web browser. It is thus a Web application that combines the editor itself with the implementation of the CRDT that will ensure the consistency of the shared document. The lower layer enables the opportunistic communication between the contributors' laptops, thus allowing the synchronization of the different replicas of the CRDT. The two layers interact via a websocket. In addition, a Web server is deployed locally to supply the code and data of the Web application. Note that the fact that our editing application is based on Web technologies is not a fundamental requirement. It rather results from the choice of an available, extensible and efficient CRDT-based editor.

A. *Opportunistic communication*

The main functions of the opportunistic communication layer we developed are, on the one hand, to ensure neighbor detection (i.e., to detect laptops located within the radio range), and on the other hand to establish a bidirectional communication channel between each pair of neighbors.

Neighbor detection is trivially based on the periodic broadcasting, by each laptop, of HELLO messages that contain the identity of the emitting device (typically the hostname), its IP (Internet Protocol) address, and the TCP (Transmission Control Protocol) port number it listens to. This broadcast is only performed at one hop (i.e., without any routing), so that each host can only detect its 1-hop neighbors. HELLO messages are embedded in UDP (User Datagram Protocol) datagrams, and addressed to a predefined multicast group.

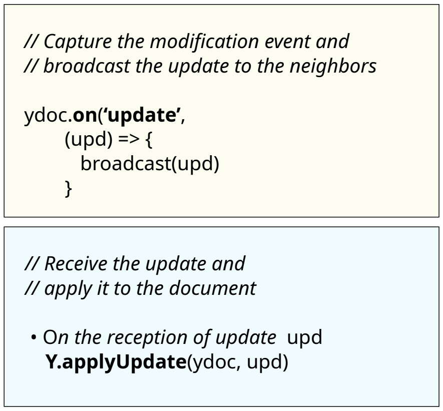
When a laptop receives a HELLO message from a new neighbor with a lower identity (in lexicographic order), it opens a TCP session (with TLS [Transport Layer Security] encryption) with this new neighbor. This session will then serve as a bidirectional channel between the two neighbors, as long as they remain in radio contact. When a contact is lost between two neighbors, it can be reestablished later if they meet again.

B. *CRDT-based editor*

We chose the Quill text editor [22] as the editing software. This editor is written in HTML (HyperText Markup Language)/CSS (Cascading Style Sheets)/Javascript. It can be associated with Yjs [23], through the *quill-yjs* binding module, so that the edited text is maintained internally as a CRDT. Yjs is a Javascript implementation of several types of CRDTs (array, map, text, etc.). It is mainly oriented to linear data structures like text [24]. Figure 4 illustrates the manipulation of a piece of text through the use of the Yjs Javascript library, as could be done by the Quill editor (in this figure, and also in Figures 5 and 6, the functions provided by Yjs are in boldface type).

One of the most interesting characteristics of Yjs is its efficient way of encoding a text CRDT as a double chained list of inserted items (sequences of characters identified by Lamport timestamps) accompanied with a set of deletions (simple set of items), which confers a high efficiency for human-produced text manipulation.

Yjs itself is network agnostic: some extra code is required to ensure the synchronization of replicas. This code must be included in a Yjs *provider*. The providers distributed with Yjs are not suited to opportunistic networking, as they target Internet-based contexts (by using typically WebRTC or centralized servers). We therefore developed our own Yjs provider, called *opp-provider*, which supports delta-state-based synchronization. This task was facilitated by two features of Yjs: *updates* and *state vectors*. An update encodes a series of changes in a document that can serve to modify another document. Updates are commutative, associative, and idempotent. These properties are essential to ensure the eventual convergence of all replicas, whatever the order in which updates are applied to each replica. A state vector characterizes the state of advancement of all replicas, as perceived by one replica. It is essentially a set of Lamport timestamps that captures the causal context. Updates and state vectors are provided to the programmer as opaque structures encoded in a compressed binary format.



```
// Capture the modification event and
// broadcast the update to the neighbors

ydoc.on('update',
      (upd) => {
         broadcast(upd)
      }
```

```
// Receive the update and
// apply it to the document

 • On the reception of update  upd
   Y.applyUpdate(ydoc, upd)
```

Figure 5. Synchronization performed on a set of neighbors when a contributor modifies the text (the code on the top runs on the laptop where the text is modified).

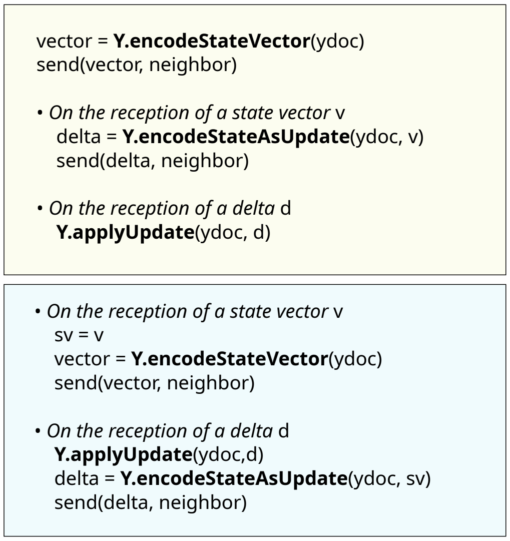
The *opp-provider* is notified of editing events (issued by the Yjs module) and neighbor discovery events (issued by the underlying opportunistic communication layer). When a contributor edits the text (by inserting or deleting a character), the *opp-provider* immediately broadcasts the corresponding update to all current neighbor laptops (if any). The receiving neighbors can then apply this update on their own replica (see Figure 5). From a contributor's perspective, collaborative editing operates in real time between his/her laptop and neighbor laptops. The updates exchanged between these laptops are embedded in small messages, as each update only pertains to the last operation performed on the sender.

In contrast, when two laptops get into radio contact (and become neighbors), their local states must be synchronized, which usually requires exchanging larger messages. The two new neighbors first exchange state vectors, and then exchange only what the other laptop needs to reach a common state (see Figure 6).

V. Experiment

A. Experimental conditions

In order to assess whether collaborative editing based on opportunistic communication can be of practical use, we decided to use this approach to write the latest deliverable of a project our research team is involved in. This project involves six permanent staff members, which all agreed to participate in this experiment. These six participants are not always collocated, though. For example, they all have teaching duties, which do not always occur in the same campus or in the same buildings. Besides, most team members work at home part of the time, so they meet only occasionally. As a general rule, two meetings are organized every week, though, on Tuesday and Friday afternoons, but not all team members attend every meeting.



```
vector = Y.encodeStateVector(ydoc)
send(vector, neighbor)

• On the reception of a state vector v
   delta = Y.encodeStateAsUpdate(ydoc, v)
   send(delta, neighbor)

• On the reception of a delta d
   Y.applyUpdate(ydoc, d)
```

```
• On the reception of a state vector v
   sv = v
   vector = Y.encodeStateVector(ydoc)
   send(vector, neighbor)

• On the reception of a delta d
   Y.applyUpdate(ydoc,d)
   delta = Y.encodeStateAsUpdate(ydoc, sv)
   send(delta, neighbor)
```

Figure 6. Delta-state-based synchronization applied when two neighboring laptops enter in contact (the code on the top is executed by the neighbor with the larger identity).

The appropriate software was installed on each participant's laptop (running Linux), and this laptop was configured so as to support opportunistic communication. More specifically:

- A secondary Wi-Fi interface (small form factor USB dongle) was added to each laptop, and this interface was configured so as to operate continuously in Wi-Fi ad hoc mode. The primary builtin Wi-Fi interface of each laptop therefore remained available for daily activities such as Web browsing, Email, etc. The secondary interface was meant to be used only for opportunistic communication, that is, in that case, for collaborative editing.
- The opportunistic communication layer (cf. Figure 3) was installed on each laptop, configured to use the secondary interface (with self-assigned IPv6 addresses), and run in the background as a *systemd* service.
- The desktop settings on each laptop were configured so that the Web browser opened as soon as the user logged in, and the browser itself was configured so that the Quill/Yjs page was loaded automatically. The participants were asked to keep the browser running as much as possible, and to maintain a window or tab on Quill/Yjs in this browser (note that this did not prevent them to browse other Web sites). The motivation was to ensure that Yjs would keep running in the background, thus ensuring automatically the synchronization between the laptops used in this experiment.
- Each laptop was configured so as to log interesting events, such as the laptop being switched on or off, the discovery of a new neighbor, etc.

Overall, each laptop was configured so that its owner could

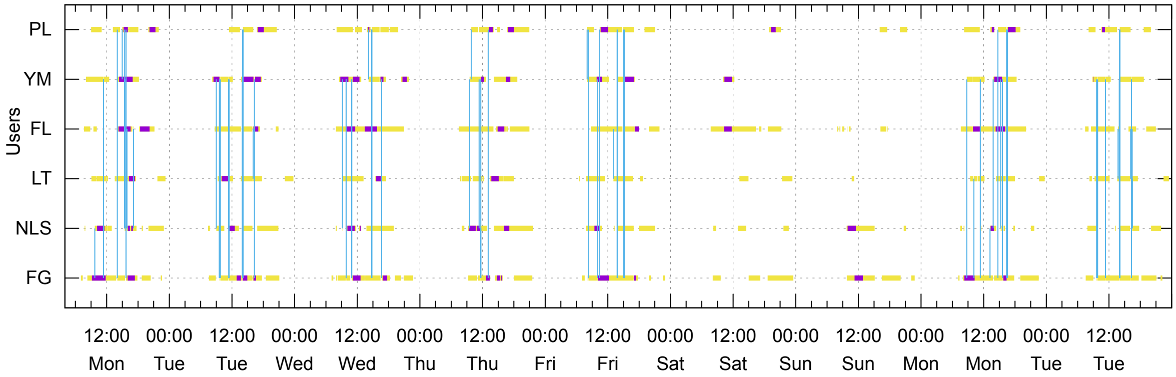


Figure 7. Details of each laptop’s activity during the experiment.

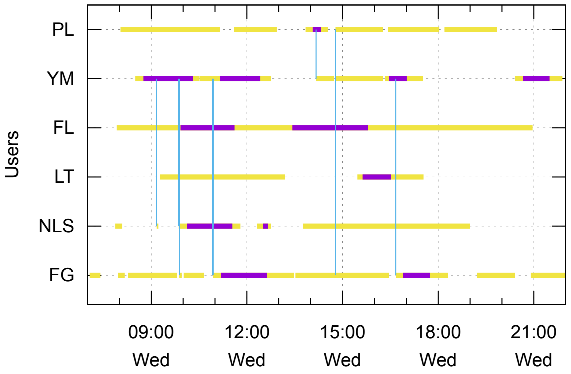


Figure 8. Details of each laptop’s activity on Wednesday.

keep using it as usual (browsing the Web, sending and receiving Email, etc.), while collaborative editing was performed solely based on opportunistic communication. The general idea was to enlist participants in this experiment without overly disrupting their day-to-day activity.

B. Results

The whole experiment lasted 9 days, including one weekend (from Monday to the next Tuesday). Figure 7 shows the timelines of each laptop’s activity during that period. Each laptop is identified by the initials of its owner’s name. In this figure, each horizontal yellow segment corresponds to a period during which the corresponding laptop was up and running. Purple segments mark periods when editing activity was observed on this laptop (i.e., the user was actually editing the shared document). The thin vertical blue lines indicate the beginning of a pairwise radio contact between two laptops, that is, an opportunity for these laptops to synchronize their copies of the shared document.

Figure 8 shows details about the activity observed on a specific day (Wednesday) during the experiment.

Figure 9 shows the evolution of the number of active laptops at any time during the experiment (blue), as well as the number of pairwise connections between these laptops (red). It can be observed that although several laptops were sometimes running at the same time, this does not imply that all these laptops were connected over the wireless ad hoc channel. For example, on Wednesday afternoon all six laptops were up and running most of the time, but only three pairwise connections were observed. This is because, as explained earlier, the participants were not always collocated, so each participant could occasionally use his/her laptop —and possibly edit the shared document— while being disconnected from any other laptop, or while being connected with only one or two other laptops.

Statistical details about the experiment are presented in Tables I and II. They concern the editing events and the transfers of synchronization messages, and the radio contacts and inter-contacts between laptops.

TABLE I. EDITING EVENTS AND SYNCHRONIZATION.

Metrics	Values
Size of the final document	102 651 characters (36 pages)
Nb. of editing events	114 612 “ins”, 6 821 “del”, 104 “cut”, 81 “paste”
Nb. of synchronizations upon radio contact	109
Nb. of updates transferred during radio contacts	102 561

TABLE II. CONTACTS AND INTER-CONTACTS.

Metrics	Values (* = min / max / avg / stdev)
Duration of the experiment	8d16h28’41”
Nb of participants (laptops)	6
Number of contacts	109
Durations of contacts	3’16”/4h19’34”/1h19’53”/41’28”*
Number of inter-contacts	94
Durations of inter-contacts	5’06”/95h29’37”/24h26’51”/30h18’35”*

The shared document produced during this experiment is 102 651-character long (36 pages, in plain text). The numbers of *ins*, *del*, *cut*, and *paste* events triggered to produce this document are detailed in Table I. Overall, 109 synchronizations occurred between pairs of laptops upon radio contact, which is consistent with the number of contacts observed (see Table II). Each of these synchronizations involved the exchange of state

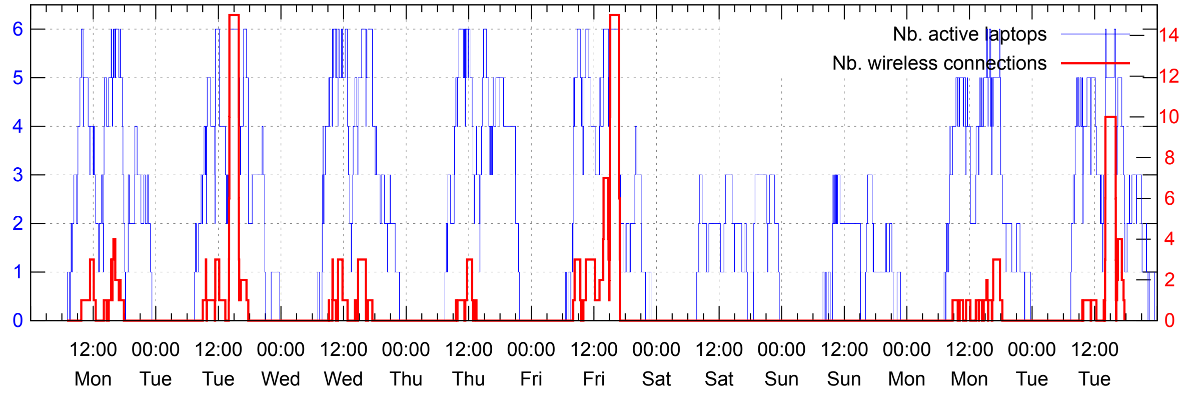


Figure 9. Number of active laptops (blue) and number of pairwise connections between laptops (red).

vectors and delta states between two laptops, as shown in Figure 10. The state vectors (at most 700 bytes) were of course far smaller than the deltas transferred upon contacts.

While laptops were in contact, 102 561 transfers of updates were observed. These transfers of updates are actually transfers of deltas pertaining to the last operation applied on the sender, but we distinguish updates from deltas here because updates are significantly smaller. Figure 11 shows the evolution of the size of the messages carrying these updates over time. A large majority of these messages were indeed quite small (about 130 bytes), for they concerned only the insertion or the removal of a single character. Occasionally, larger messages were observed (up to 700 bytes), when a user pasted a piece of text in the shared document. Note that no attempt was made to compress the data transmitted in the messages in this early version of our editing system.

Figure 12 shows the evolution of the size of the shared document on each laptop. It can be observed that this size sometimes trailed behind significantly on some laptops (see for example laptops FL and LT on Friday), as some participants worked far from any other participant, thus preventing their laptop to synchronize with other laptops. The weekly meetings on Tuesday and Friday afternoons allowed most laptops to fully synchronize, although some participants did not attend all meetings and therefore relied on unplanned contacts to get the latest contributions of every other participant. For example, the shared document was finalized (as planned) during the last Tuesday meeting, even though one of the participants did not attend this meeting. A contact with this participant's laptop occurred shortly after this meeting, which allowed this laptop to get the final version of the document.

C. Outcome of this experiment

The experiment described earlier only involved six participants over a few days. With this small-scale experiment our prime motivation was to verify that collaborative editing based on opportunistic communication is indeed doable and practical. It turns out that editing a shared document in such conditions is actually a near-real-time experience, with synchronizations depending on unpredicted collocation between contributors. At the end of the experimentation period, the participants confirmed that while working on this deliverable they did not perceive opportunistic synchronization as an inconvenience.

Of course opportunistic synchronization requires that all updates eventually reach all replicas. More specifically, the synchronization graph must be connected, which is actually a major requirement for any distributed application involving CRDTs [3]. This experiment shows that this requirement can easily be met in a real-life scenario.

D. Scalability

Scalability is usually a typical concern in distributed applications. Yet, the question whether an experiment similar to that described above could have been performed with hundreds of participants hardly makes sense, since editing a shared document with so many contributors would probably not be practical anyway.

Yjs can however support other kinds of CRDTs (namely arrays or maps), which can serve as building blocks for a large variety of data structures. The software system used in our experiment could therefore be used to support large-scale CRDT-based collaborative applications. In order to determine if this software system would scale up, we ran additional experiments in emulation mode. In these experiments, the LEPTON platform [25] was used to simulate the mobility and opportunistic contacts of a large number of virtual nodes. For each of these virtual nodes, instead of running Quill/Yjs in a Web browser, we used node.js and replaced the real editor Quill by a dummy editor we developed in order to mimic the editing events a real user would generate over time.

With this architecture we ran scenarios involving up to 200 virtual nodes editing the same shared document concurrently, and did not observe any adverse effect on the eventual convergence of all copies of the shared document.

VI. Conclusion

In this paper, we have described the experimental setup and the results of a real-life experiment involving a group of researchers that collaboratively edited a document over nine days, relying exclusively on opportunistic communication to synchronize their contributions to this document. The text

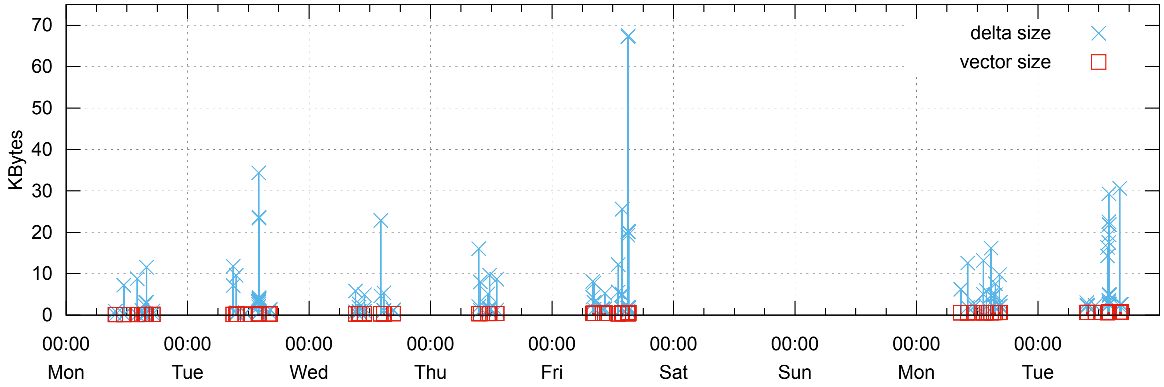


Figure 10. Evolution of the size of the messages carrying state vectors (red) and delta states (blue) transferred between neighbor laptops upon contacts.

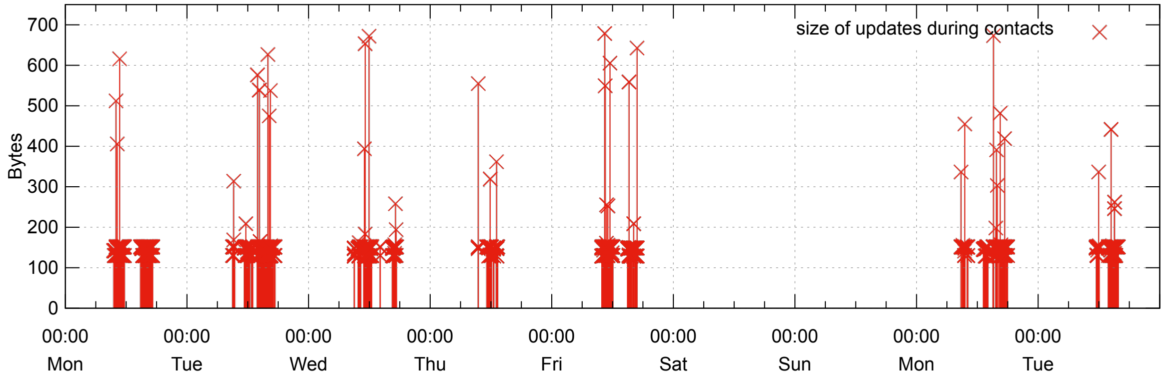


Figure 11. Evolution of the size of the messages carrying updates during contacts.

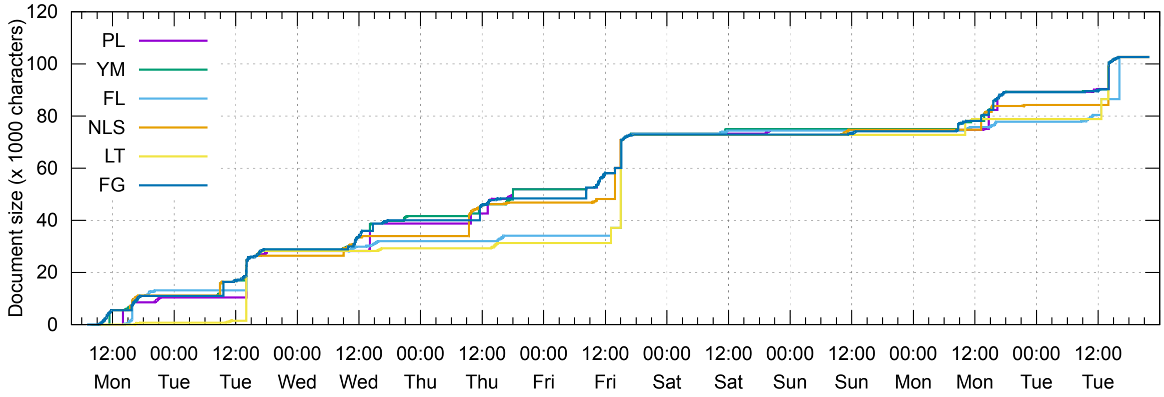


Figure 12. Evolution of the size of the shared document on each laptop.

editor deployed on the users' laptops relied on an off-the-shelf implementation of a text CRDT to store the different replicas of the shared document. A specific module ensured a delta-state-based synchronization of the replicas between neighbor laptops.

The analysis of the log files produced during the experiment confirmed the opportunistic nature of the network formed by the laptops, and the ability of the CRDT-based editing system to maintain the consistency of the replicas stored on each laptop.

This small-scale experiment was conducted in an academic setting, avoiding deliberately —and somehow artificially— to rely on the Internet for collaborative editing. It confirms that actual collaborative work in opportunistic networking conditions is indeed viable. Moreover, the mobility of the nodes and the users' behavior are not fundamentally different from those found in other application domains, which let us think that the usefulness of the approach is quite general. Besides, a simulation run involving the same CRDT-based editing system with 200 virtual contributors shows that scalability is not an issue (although having that many contributors edit the same document simultaneously would probably be useless).

We believe this work paves the way for the deployment and use of distributed collaborative applications in situations where opportunistic communication would be the primary and possibly only option, in remote areas or in a disaster-relief situation for example.

REFERENCES

[1] "Message Passing Interface Forum." https://mpi-forum.org/. 2023.08.20.

[2] "The OpenMP API specification for parallel programming." https://www.openmp.org/. 2023.08.20.

[3] N. Preguiça, "Conflict-free Replicated Data Types: an Overview," 2018. Arxiv Preprint https://arxiv.org/abs/1806.10254.

[4] F. Guidec, Y. Mahéo, and C. Noûs, "Supporting conflict-free replicated data types in opportunistic networks," *Peer-to-Peer Networking and Applications*, vol. 16, pp. 395–419, 2023.

[5] S. Weiss, P. Urso, and P. Molli, "Logoot: A Scalable Optimistic Replication Algorithm for Collaborative Editing on P2P Networks," in *29th International Conference on Distributed Computing Systems (ICDCS'09)*, (Montreal, Canada), pp. 404–412, IEEE, June 2009.

[6] C. E. A. Robin and V. M. Romero, "DTNDocs: A delay tolerant peer-to-peer collaborative editing system," in *32nd International Conference on Information Networking (ICOIN 2018)*, (Chiang Mai, Thailand), pp. 92–97, Jan. 2018.

[7] L. André, S. Martin, G. Oster, and C.-L. Ignat, "Supporting adaptable granularity of changes for massive-scale collaborative editing," in *9th International Conference on Collaborative Computing: Networking, Applications and Worksharing (CollaborateCom)*, (Austin, TX, USA), pp. 50–59, IEEE, Oct. 2013.

[8] N. Alsulami and A. Cherif, "Collaborative Editing over Opportunistic Networks: State of the Art and Challenges," *International Journal of Advanced Computer Science and Applications*, vol. 8, no. 11, 2017.

[9] M. Shapiro, N. Preguiça, C. Baquero, and M. Zawirski, "A Comprehensive Study of Convergent and Commutative Replicated Data Types," Tech. Rep. 7506, INRIA, Jan. 2011.

[10] P. S. Almeida, A. Shoker, and C. Baquero, "Efficient State-Based CRDTs by Delta-Mutation," in *International Conference on Networked Systems (NETYS 2015)*, (Agadir, Morrici), pp. 62–76, Springer, May 2015.

[11] A. van der Linde, J. Leitão, and N. Preguiça, "*Delta*-CRDTs: Making *delta*-CRDTs Delta-based," in *2nd Workshop on the Principles and Practice of Consistency for Distributed Data (PaPoC 2016)*, (London, United Kingdom), ACM, Apr. 2016.

[12] P. S. Almeida, A. Shoker, and C. Baquero, "Delta state replicated data types," *Journal of Parallel and Distributed Computing*, vol. 111, pp. 162–173, 2018.

[13] A. Lindgren, A. Doria, J. Lindblom, and M. Ek, "Networking in the Land of Northern Lights: Two Years of Experiences from DTN System Deployments," in *Workshop on Wireless Networks and Systems for Developing Regions (WiNS-DR'08)*, (San Francisco, CA, USA), pp. 1–8, ACM, Sept. 2008.

[14] L. Baumgärtner, P. Gardner-Stephen, P. Graubner, J. Lakeman, J. Höchst, P. Lampe, N. Schmidt, S. Schulz, A. Sterz, and B. Freisleben, "An Experimental Evaluation of Delay-Tolerant Networking with Serval," in *Global Humanitarian Technology Conference (GHTC)*, (Seattle, WA, USA), pp. 70–79, IEEE, Oct. 2016.

[15] P. Tennent, M. Hall, B. Brown, M. Chalmers, and S. Sherwood, "Three applications for mobile epidemic algorithms," in *7th International Conference on Human Computer Interaction with Mobile Devices & services (MobileHCI05)*, (Salzburg, Austria), pp. 223–226, ACM, Sept. 2005.

[16] Z. Chen, E. A. Yavuz, and G. Karlsson, "What a juke! A collaborative music sharing system," in *13th International Symposium on a World of Wireless, Mobile and Multimedia Networks (WoWMoM 2012)*, (San Francisco, CA, USA), pp. 1–6, IEEE, June 2012.

[17] Y. Mahéo, N. Le Sommer, P. Launay, F. Guidec, and M. Dragone, "Beyond Opportunistic Networking Protocols: a Disruption-Tolerant Application Suite for Disconnected MANETs," in *4th Extreme Conference on Communication (ExtremeCom'12)*, (Zürich, Switzerland), pp. 1–6, ACM, Mar. 2012.

[18] F. Guidec, N. Le Sommer, and Y. Mahéo, "Opportunistic Software Deployment in Disconnected Mobile Ad Hoc Networks," *International Journal of Handheld Computing Research*, vol. 1, no. 1, pp. 24–42, 2010.

[19] T. Kärkkäinen and J. Ott, "Shared Content Editing in Opportunistic Networks," in *9th MobiCom Workshop on Challenged Networks (CHANTS'14)*, (Maui, Hawaii, USA), pp. 61–64, ACM, Sept. 2014.

[20] M. Costea, R.-I. Ciobanu, R.-C. Marin, C. Dobre, C. X. Mavromoustakis, G. Mastorakis, and F. Xhafa, "Total Order in Opportunistic Networks," *Concurrency and Computation: Practice and Experience*, vol. 29, no. 10, 2017.

[21] M. Doering, S. Lahde, J. Morgenroth, and L. Wolf, "IBR-DTN: an efficient implementation for embedded systems," in *4th Workshop on Challenged Networks (CHANTS 2008)*, (San Francisco, CA, USA), pp. 117–120, ACM, Sept. 2008.

[22] "Yjs – A CRDT framework with a powerful abstraction of shared data." https://github.com/yjs/yjs/. 2023.08.20.

[23] "Quill Rich Text Editor." https://quilljs.com/. 2023.08.20.

[24] P. Nicolaescu, K. Jahns, M. Derntl, and R. Klamma, "Near Real-Time Peer-to-Peer Shared Editing on Extensible Data Types," in *19th International Conference on Supporting Group Work (GROUP'16)*, (Sanibel Island, FL, USA), pp. 39–49, ACM, Nov. 2016.

[25] A. Sánchez-Carmona, F. Guidec, P. Launay, Y. Mahéo, and S. Robles, "Filling in the missing link between simulation and application in opportunistic networking," *Journal of Systems and Software*, vol. 142, pp. 57–72, Aug. 2018.

Deep Learning based Indoor Positioning Approach Using Wi-Fi CSI/RSSI Fingerprints Technique

Marco Mühl and Wiem Fekih Hassen

Chair of Distributed Information Systems, University of Passau, Innstraße 41, 94032 Passau, Germany
Email: {firstname.lastname}@uni-passau.de

***Abstract*—Indoor Positioning Systems (IPSs) play a vital role in various applications, ranging from asset tracking to location-based services. With different approaches being explored in the last years, Wi-Fi-based IPSs utilizing Channel State Information (CSI) and Received Signal Strength Indicator (RSSI) have gained increased attention. This research aims to develop a Wi-Fi based indoor positioning system using CSI and RSSI measurements, specifically focusing on datasets collected at the University of Passau, since datasets used in related work are private. Additionally, after the acquired data is subjected to preprocessing and data cleaning techniques, the study explores the potential of Machine Learning (ML) techniques, including Support Vector Regression (SVR), Long Short-Term Memory (LSTM), and Convolutional Neural Networks (CNN), to enhance positioning accuracy. These models are trained and evaluated using appropriate performance metrics, including Mean Squared Error (MSE) and distance error. The experimental results, focusing on the prediction of vertical and horizontal coordinates within the laboratory room, demonstrate the effectiveness of the proposed system. For unseen RSSI data, the best distance error based on MSE achieved was 29.5 cm using SVR, while for unseen CSI Amplitude data, the lowest distance error based on MSE was 37.9 cm with a CNN approach. A comparison is conducted within the different methods. All tested models consistently achieve a distance error based on MSE of under 50 cm, proving the high quality of the collected dataset. Future research directions and areas for improvement are also suggested.**

***Keywords*—IPSs, CSI, RSSI, private dataset, Raspberry Pi, SVR, LSTM, CNN.**

I. Introduction

Smart Things and connected devices have become an integral part of people's daily lives as well as various industries, such as healthcare and manufacturing. One of the key areas that is driving this trend is location-based services. While for outdoor applications the Global Positioning System (GPS) is an established standard [1], Indoor Positioning Systems (IPSs) face different challenges [2].

Compared to outdoor positioning, some of these challenges are: Need for higher accuracy, while having higher levels of signal interference, need for low power consumption, and handling environmental changes [3]. Since the demand for IPSs is rising, different approaches to solving these problems have emerged. With some approaches utilizing Radio-Frequency Identification (RFID), Bluetooth, or Ultra-Wideband (UWB) the use of Wi-Fi-signals, which are available in almost every indoor environment, has become a good option to provide accurate indoor localization [4]. They are compatible with almost every mobile device, are low cost, and have wide signal coverage.

Various signal metrics, such as Channel State Information (CSI) and Received Signal Strength Indicator (RSSI) can be employed in Wi-Fi-based IPSs, as well as different positioning techniques, like proximity, multilateration, angulation, or fingerprinting, to determine the location of a device within an indoor environment [5], [6]. In recent years, the application of Machine Learning (ML) algorithms has exhibited considerable improvements in the performance of IPSs [7]. These algorithms have the potential to effectively analyze the complex patterns and relationships present in the collected data, enabling accurate localization and tracking.

Since no CSI dataset for indoor positioning used in related work are publicly available, one primary objective of this work is to create an extensive CSI and RSSI dataset for a laboratory at the University of Passau. Subsequently, a comparative analysis of different Deep Learning (DL) approaches, including Support Vector Regression (SVR), Long Short-Term Memory (LSTM), and Convolutional Neural Networks (CNN), will be conducted to evaluate their effectiveness in achieving accurate indoor localization.

The remainder of this paper is structured as follows: Section II starts by outlining the proposed approach of a Wi-Fi-based indoor positioning system. After that, the data collection steps and the processing techniques for both CSI and RSSI data are detailed. Section III presents the adopted ML algorithms, i.e., SVR, LSTM and CNN. Section IV describes a comparative analysis of the results. Section V concludes the paper and identifies future challenges.

II. Proposed Methodology

A. Overall System Architecture

This subsection presents the proposed system architecture and the methods employed in this work. Our system is composed of three main phases, as depicted in Figure 1:

- *Phase 1*: consists of creating a dataset of RSSI and CSI values based on the specific collection tool Nexmon on a Raspberry Pi.
- *Phase 2*: involves the application of multiple data processing methods to clean up the datasets and prepare the fingerprint dataset.
- *Phase 3*: details the different implemented ML algorithms and their results.

ISBN: 978-1-68558-106-0

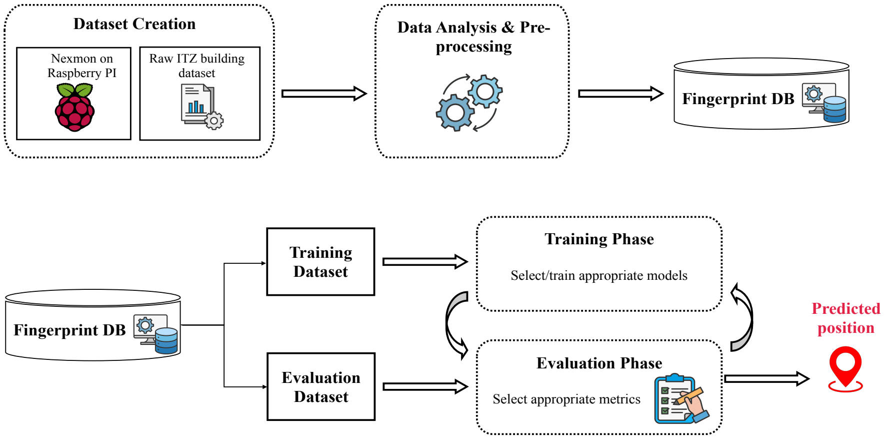


Fig. 1. Overall system architecture.

B. Data Collection

The collection of high-quality data is crucial for subsequent steps of preprocessing and the application of machine learning algorithms. This section provides details on the tool Nexmon CSI extractor, which is used to collect CSI and RSSI data in this work.

The Nexmon CSI Extractor [8], [9] is configured on a Raspberry Pi, allowing for the collection of CSI and RSSI data in a laboratory setting at the University of Passau. Introduced in 2019, it provides a good option to collect CSI data. It employs rooted Broadcom Wi-Fi chips found in multiple devices such as Nexus smartphones, Raspberry Pi boards, and Asus RT-AC86U routers. Custom firmware is employed to enable CSI data collection by listening on a specific User Datagram Protocol (UDP) socket. The Nexmon CSI Extractor offers support for 128 subcarrier groups with the highest resolution of 32 bits. Another significant advantage is its ability to simultaneously collect CSI and RSSI data.

The CSI and RSSI data was collected in a laboratory room located in the ITZ Building at the University of Passau. The room has a pentagon shape with an approximate area of 45 m^2. For each Reference Point (RP) (i.e., A RP is the point learned from the training phase, where the different RSS values were recorded.) inside the room, data was collected in four directions, north, east, south, and west. The RPs are *1 m* apart from each other and most of the outer ones are *0.45 m* apart from the walls. The numbering scheme of the RPs follows a vertical-horizontal structure, with the first (or first two, for three-digit numbers) digits representing the vertical position, and the last digit representing the horizontal position. The vertical axis ranges from 1 to 10 and the horizontal axis ranges from 1 to 5. This gives the room a matrix-like structure, which is useful for further data processing and position predictions.

For each RP, data was collected in four directions. The Raspberry Pi was positioned in front of a person's body, with a collection direction arrow pointing away from them. In each direction at each RP, 100 Wi-Fi Frames were collected. This resulted in a total of 400 Wi-Fi Frames per RP, and with 45 RPs amounting to a total of 18000 collected Wi-Fi Frames. The data is stored in .pcap files on a per direction, per RP basis, meaning that every .pcap file contains 100 Wi-Fi Frames. Data was collected over the course of one day (10h) by two researchers.

TABLE I
EXCERPT OF COLLECTED WI-FI FRAME STRUCTURE

Bytes	Type	Name	Description
1	uint8	RSSI	RSSI value in Two Complement Form
6	uint8[6]	Source Mac	Source Mac ID of the Wi-Fi Frame
variable	int16[]	CSI Data	Each CSI sample is 4 bytes with interleaved Int16 Real/Imaginary.

An excerpt of the structure of a collected Wi-Fi Frame can be found in Table I. For an IPS, the three shown variables in the table are of particular importance. The RSSI and CSI data fields describe the wireless characteristics of the Wi-Fi Frame with a specific Source MAC Address. While the RSSI and Source MAC field contain single values, the CSI data field contains both amplitude and phase information for each Orthogonal Frequency-Division Multiplexing (OFDM) subcarrier in complex form. In the setup of this study, data for 256 subcarriers (Bandwidth (80Mhz) * 3.2) is available. This raw network data is crucial for creating a high-quality fingerprint database. In the next step, this data must be cleaned and preprocessed, to prepare it for the machine learning algorithms.

C. Data Processing

To create the fingerprint database, data processing is a crucial step before training the ML models. In this paper, two fingerprint databases were created, one for the RSSI data and one for the CSI Amplitude data. The CSI Phase was not considered since CSI Amplitude data was more stable and easier to process. For both RSSI and CSI Amplitude data, different preprocessing was performed. This section aims to give an insight into the used techniques and results of creating the datasets. To apply the preprocessing methods, the data was first extracted with the CSIKit library [10]. This library extracts the raw data of the .pcap files into a python environment, where further steps can be executed.

1) The CSI Amplitude dataset: it contains data for 45 RPs, and 1283 columns with information, shaping the dataset to be in a matrix form of 45x1283. With three columns containing position and direction information, the input dataset contains 57600 CSI Amplitude values. The mean CSI Amplitude values of all four directions were calculated for every RP, to make the dataset more robust. Also, the mean direction value of 2.5 was kept in the dataset to make this clearer.

The other columns each represent the CSI Amplitude values of a subcarrier for a specific Source MAC address. Subcarriers, that do not contain CSI Amplitude data like for example null subcarriers [11], are removed. The dataset is filtered using only the Source MAC addresses, which contain a reasonable amount of data. Not a Number (NaN) values are replaced with the minimum value of each column. To further improve the data quality, more preprocessing actions were done for the CSI Amplitude data. CSI data often contains noise and outliers that can distort the essential information. In this study, three filters are used for CSI Amplitude data processing: the running mean filter, a lowpass filter, and a Hampel filter [12]. For the running mean filter, a window size of 10 was chosen, the lowpass filter was configured to isolate frequencies below 10Hz and for the Hampel Filter, a window size of 10 and a significance of 3 was set. Figure 2 shows the effects of preprocessing the CSI Amplitude data for a specific RP.

2) The RSSI dataset: the size of this dataset is 45 RPs, with 8 columns containing data making it a matrix-like shape of 45x8. Three columns contain position and direction information, amounting to a total of 225 RSSI values. The other columns contain the actual RSSI values in dBm, the column names are the Source MAC addresses. Here, again only the Source MAC addresses with a reasonable amount of data were added, in order to reduce complexity of the dataset and therefore improve performance. As discussed earlier, for every direction of each Reference Point, there are 100 collected Wi-Fi Frames. For frames with the same Source MAC address, the mean RSSI values of them is used as value, further improving the robustness of the dataset. Frames with no Source MAC address information are ignored.

If no Wi-Fi Frame was collected for a specific MAC address at a RP, the resulting NaN values were replaced with the minimum possible value for RSSI, -100. With the now clean and preprocessed datasets, it is possible to train the machine learning models. The setup and results will be explored next.

III. ML Algorithms Application

In this section, the effectiveness of SVR, LSTM and CNN algorithms in leveraging both CSI amplitude and RSSI data, is evaluated and compared. With the preprocessed RSSI and CSI datasets of the previous section, the three different models are trained and evaluated. For better comparability, the horizontal and vertical positions are predicted separately, as the SVR approach allows for only one output. For the combined accuracy predicting both horizontal and vertical positions, the mean of the separate values is taken for the SVR approach. The LSTM and CNN implementations allow two outputs. To compare the results, the Mean Squared Error (MSE) is calculated for each model. For all three models, the CSI Amplitude and RSSI datasets are split into train and test datasets. Table II depicts the input dataset sizes for both datasets.

TABLE II
Sizes of input datasets

Value	Train Data (80%)	Test Data (20%)	Total
RSSI	180	45	225
CSI Amplitude	46080	11520	57600

A. Support Vector Regression (SVR)

SVR is a powerful machine learning technique used for solving regression problems [13]. SVR utilizes kernel functions to transform the input data into a higher-dimensional feature space, where linear regression is performed. For the environment of this study, the Radial Basis Function (RBF) kernel was chosen for transforming the input data into a higher-dimensional feature space. It can capture complex non-linear relationships between the input features and the output variable and therefore gave significantly better results than configurations with other kernels. Table III depicts the MSE results for the RSSI and CSI Amplitude datasets.

TABLE III
MSE predicting with SVR model

Value	Vertical		Horizontal		Vertical and Horizontal	
	Train	*Test*	*Train*	*Test*	*Train*	*Test*
RSSI	0.0110	0.0292	0.0531	0.0670	0.032	0.048
CSI Amplitude	0.0076	0.0419	0.0082	0.1156	0.007	0.078

For the RSSI dataset, the MSE was 0.032 for evaluating with the train data, and 0.048 when evaluating with unseen data. With the CSI Amplitude dataset, a the MSE for the train dataset was 0.007 and for the test dataset 0.078.

B. Long Short-Term Memory (LSTM)

In this study, a carefully designed and optimized LSTM model architecture was employed. The selected architecture comprises two layers. The first layer of the model is an LSTM layer. The LSTM layer consists of 50 units for the RSSI dataset, and 8 units for the CSI Amplitude dataset, allowing the model to capture and learn complex patterns in the input data. By leveraging its inherent memory cells, the LSTM layer can retain and utilize essential information from past observations to inform future predictions accurately. An overview of the configuration when predicting the vertical and the horizontal

ISBN: 978-1-68558-106-0

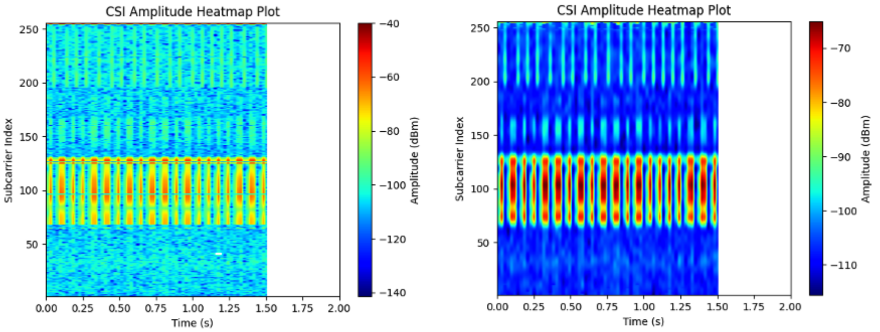


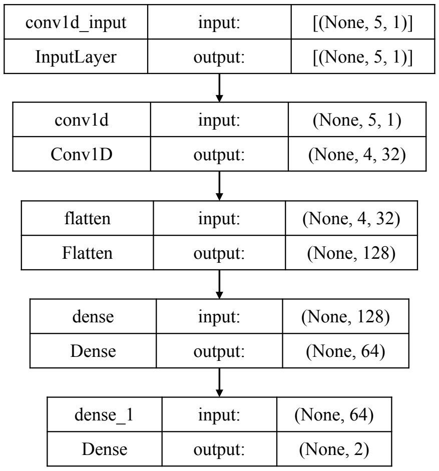
Fig. 2. Heat Maps of CSI Amplitude before (left) and after (right) preprocessing.

position for the LSTM approach is displayed in Table IV. The second layer of the model is a dense layer. The purpose of this layer is to consolidate the information extracted by the LSTM layer and make precise predictions. The number of units in the dense layer is determined based on the desired dimensions of the predicted outputs. For single-dimensional predictions, such as determining the location along a specific axis, a single unit is utilized. However, when predicting both the vertical and horizontal dimensions simultaneously, two units are employed to capture the multi-dimensional nature of the indoor positioning problem.

The MSE for the RSSI dataset was found to be 0.0415 when evaluating it with the training data, and 0.0691 when assessing it with unseen data. Regarding the CSI dataset, the MSE was 0.0008 for the training dataset and 0.0801 for the test dataset, as seen in Table V.

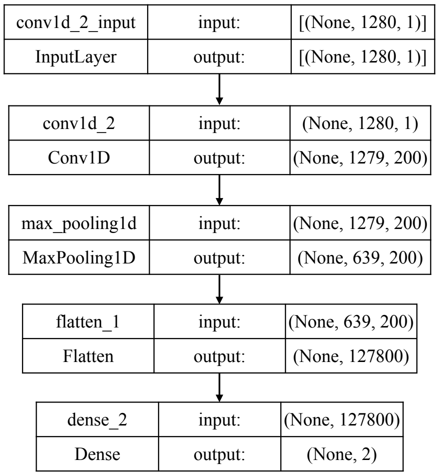
C. Convolutional Neural Networks (CNN)

In this work, a CNN-based ML model is employed for IPSs utilizing the CSI Amplitude and RSSI data. CNNs are particularly well-suited for tasks involving grid-like data, such as images or in this case, CSI Amplitude and RSSI data [14]. Two different architectures for RSSI and CSI Amplitude models were created, since the RSSI data differs a lot from the CSI Amplitude data. An overview of the architectures is depicted in Figure 3 for the RSSI dataset, and Figure 4 for the CSI Amplitude dataset.



conv1d_input	input:	[(None, 5, 1)]
InputLayer	output:	[(None, 5, 1)]
conv1d	input:	(None, 5, 1)
Conv1D	output:	(None, 4, 32)
flatten	input:	(None, 4, 32)
Flatten	output:	(None, 128)
dense	input:	(None, 128)
Dense	output:	(None, 64)
dense_1	input:	(None, 64)
Dense	output:	(None, 2)

Fig. 3. CNN model architecture for RSSI data



conv1d_2_input	input:	[(None, 1280, 1)]
InputLayer	output:	[(None, 1280, 1)]
conv1d_2	input:	(None, 1280, 1)
Conv1D	output:	(None, 1279, 200)
max_pooling1d	input:	(None, 1279, 200)
MaxPooling1D	output:	(None, 639, 200)
flatten_1	input:	(None, 639, 200)
Flatten	output:	(None, 127800)
dense_2	input:	(None, 127800)
Dense	output:	(None, 2)

Fig. 4. CNN model architecture for CSI data

MSE Results for both architectures are shown in Table VI. The RSSI based model achieves a MSE error of 0.031 for the training data, and 0.077 for unseen data. For the model trained with CSI Amplitude data, the MSE for training data is 0.0000007, and 0.066 for the test data. The MSE values were again calculated by getting the mean MSE of 30 trained models, to improve the robustness of the result.

Next, the results are transformed to represent distance error, compared, and discussed.

IV. Results and Discussions

To further compare the results of all three applied algorithms, the approximate average distance errors are computed, by rescaling the MSE to the length of the axes (i.e., 900 cm

TABLE IV
LSTM MODEL CONFIGURATION FOR PREDICTING VERTICAL AND HORIZONTAL POSITION

Value	LSTM-Layer units	Dense-Layer units	Optimizer	Epochs	Batch Size
RSSI	50	2	Adam	1000	64
CSI Amplitude	8	2	Adam	200	32

TABLE V
MSE PREDICTING WITH LSTM MODEL

Value	Vertical		Horizontal		Vertical and Horizontal	
	Train	*Test*	*Train*	*Test*	*Train*	*Test*
RSSI	0.0185	0.0267	0.0496	0.1343	0.0415	0.0691
CSI Amplitude	0.0015	0.0363	0.0062	0.1059	0.0008	0.0801

for the vertical axis, 490 cm for the horizontal axis). This is done for each axis separately, as shown in Table VII and Table VIII, as well as when predicting the position with both axes, shown in Table IX.

A. Vertical Axis

Table VII focuses on predicting the vertical position using RSSI and CSI data. The distance errors are reported for both the training and testing sets.

The SVR algorithm achieves a distance error of 9.91 cm in the training set and 26.30 cm in the testing set when using RSSI data. When using CSI Amplitude data, the distance error reduces to 6.86 cm in the training set but increases to 37.79 cm in the testing set

The LSTM algorithm performs with a distance error of 16.66 cm in the training set and 24.05 cm in the testing set when using the RSSI data. However, when using CSI Amplitude data, the performance improves significantly, resulting in a distance error of only 1.42 cm in the training set and 32.70 cm in the testing set. The CNN algorithm demonstrates superior performance in predicting the vertical position. When using RSSI data, the distance error is 7.51 cm in the training set and 27.05 cm in the testing set. When utilizing CSI Amplitude data, the performance improves even further, achieving a remarkable distance error of only 0.0007 cm in the training set and 32.56 cm in the testing set.

Comparing the algorithms for vertical position prediction, the SVR has reliable results with the RSSI data, but struggles with the more complex CSI Amplitude data. The CNN approach gives comparable results as the SVR approach for the RSSI data, but outperforms both SVR and LSTM, as it consistently achieves lower distance errors with the CSI Amplitude data.

B. Horizontal Axis

Table VIII focuses on predicting the horizontal position using RSSI and CSI data. Like before, the distance errors are reported for the training and testing sets.

When using RSSI data, SVR achieves a distance error of 26.02 cm in the training set and 32.86 cm in the testing set. With CSI Amplitude data, the distance error decreases to 4.03 cm in the training set but increases to 56.65 cm in the testing set. LSTM performs with a distance error of 24.30 cm in the training set and 65.82 cm in the testing set when using RSSI data. When using CSI Amplitude data, the distance error improves slightly to 3.05 cm in the training set and to 51.93 cm in the testing set. CNN shows consistent performance in predicting the horizontal position. When using RSSI data, the distance error is 23.79 cm in the training set and 57.10 cm in the testing set. With CSI Amplitude data, the performance improves significantly, achieving a distance error of only 0.000009 cm in the training set and 43.24 cm in the testing set.

Comparing the algorithms for horizontal position prediction, CNN again outperforms SVR and LSTM in terms of distance errors, especially when utilizing CSI Amplitude data. For unseen RSSI data, SVR gives the best result.

C. Vertical and Horizontal Axis

Table IX presents the overall performance of the algorithms in predicting both vertical and horizontal positions using RSSI and CSI data.

The distance error for SVR when using RSSI data is 17.97 cm in the training set and 29.58 cm in the testing set. When utilizing CSI data, the performance improves with a distance error of 5.44 cm in the training set and 47.22 cm in the testing set. LSTM achieves a distance error of 20.48 cm in the training set and 44.93 cm in the testing set when using RSSI data. With CSI Amplitude data, the distance error improves to 2.23 cm in the training set and 42.31 cm in the testing set. CNN performs consistently well in predicting both vertical and horizontal positions. When using RSSI data, the distance error is 15.65 cm in the training set and 42.07 cm in the testing set. When utilizing CSI Amplitude data, the performance improves further, achieving a distance error of only 0.0003 cm in the training set and 37.90 cm in the testing set.

Comparing the algorithms for predicting both vertical and horizontal positions, CNN again demonstrates superior performance, achieving lower distance errors compared to SVR and LSTM, especially when utilizing CSI Amplitude data. Overall, based on the results of this study, the CNN algorithm consistently outperforms SVR and LSTM in terms of distance errors for predicting both vertical and horizontal positions in an indoor positioning system using CSI Amplitude data by about 20% and 10% respectively. Additionally, the performance of all algorithms, except SVR, generally improves when CSI data is used instead of RSSI data, highlighting the importance of considering CSI Amplitude data for accurate indoor positioning. For RSSI data only, SVR can give reliable

TABLE VI
MSE PREDICTING WITH CNN MODEL

Value	**Vertical**		**Horizontal**		**Vertical and Horizontal**	
	Train	*Test*	*Train*	*Test*	*Train*	*Test*
RSSI	0.0083	0.0300	0.0485	0,1165	0.031	0.077
CSI Amplitude	$7 * 10^{-7}$	0.0361	$2 * 10^{-8}$	0.0882	$7 * 10^{-7}$	0.066

TABLE VII
DISTANCE ERROR (CM) PREDICTING VERTICAL POSITION (MSE-BASED)

	RSSI		**CSI Amplitude**	
Algorithm	**Train**	**Test**	**Train**	**Test**
SVR	9.91	26.30	6.86	37.79
LSTM	16.66	**24.05**	1.42	32.70
CNN	**7.51**	27.05	**0.0007**	**32.56**

TABLE VIII
DISTANCE ERROR (CM) PREDICTING HORIZONTAL POSITION (MSE-BASED)

	RSSI		**CSI Amplitude**	
Algorithm	**Train**	**Test**	**Train**	**Test**
SVR	26.02	**32.86**	4.03	56.65
LSTM	24.30	65.82	3.05	51.93
CNN	**23.79**	57.10	**0.000009**	**43.24**

results as well, but the algorithm has limitations with the larger and more complex CSI Amplitude dataset.

V. CONCLUSIONS

In this study, an extensive dataset of CSI and RSSI data was meticulously collected within a controlled laboratory environment. The dataset serves as a solid foundation for future research endeavors in the field of IPSs. It encompasses crucial information, including the position with direction details, CSI Phase, CSI Amplitude, and RSSI measurements.

To assess the performance of the IPSs, three distinct ML algorithms were applied to the preprocessed datasets: SVR, LSTM and CNN. Notably, the integration of both CSI Amplitude and RSSI data yielded promising results, with all models achieving a mean distance error based on MSE of less than 50 cm, which is superior to all related works [14]–[16]. Among the individual metrics, SVR based solely on RSSI data demonstrated superior performance, attaining an MSE-based accuracy level of approximately 30 cm. Conversely, CNN, utilizing CSI Amplitude data, showcased the best results with an average MSE-based distance error of about 38 cm.

The findings of this work underscore the effectiveness of employing ML techniques, along with comprehensive preprocessing methodologies, to enhance the accuracy and reliability of IPSs. The results pave the way for future research to explore alternative algorithms, feature engineering techniques, and hybrid approaches to further improve the localization accuracy of IPSs in various indoor environments. By refining and expanding upon the methodologies established in this paper, IPSs can be further improved.

TABLE IX
DISTANCE ERROR (CM) PREDICTING VERTICAL AND HORIZONTAL POSITION (MSE-BASED)

	RSSI		**CSI Amplitude**	
Algorithm	**Train**	**Test**	**Train**	**Test**
SVR	17.97	**29.58**	5.44	47.22
LSTM	20.48	44.93	2.23	42.31
CNN	**15.65**	42.07	**0.0003**	**37.90**

REFERENCES

[1] R. Bajaj, S. L. Ranaweera, and D. P. Agrawal, "Gps: location-tracking technology," *Computer*, vol. 35, no. 4, pp. 92–94, 2002.

[2] W. F. Hassen and F. Najjar, "A positioning handoff decision algorithm for ubiquitous pedestrian navigation systems," pp. 487–494, 2016.

[3] Y. Gu, A. Lo, and I. Niemegeers, "A survey of indoor positioning systems for wireless personal networks," *IEEE Communications surveys & tutorials*, vol. 11, no. 1, pp. 13–32, 2009.

[4] C. Basri and A. El Khadimi, "Survey on indoor localization system and recent advances of wifi fingerprinting technique," in *2016 5th international conference on multimedia computing and systems (ICMCS)*. IEEE, 2016, pp. 253–259.

[5] W. F. Hassen, L. Brunie, A. Nechba, and H. Kosch, "Continuous indoor/outdoor pathway display algorithm for pedestrian navigation service," in *Proceedings of the 17th International Conference on Advances in Mobile Computing & Multimedia*, 2019, pp. 66–73.

[6] W. F. Hassen, F. Najjar, L. Brunie, H. Kosch, and Y. Slimani, "Smart pdr integration for ubiquitous pedestrian navigation service," in *2017 13th International Wireless Communications and Mobile Computing Conference (IWCMC)*. IEEE, 2017, pp. 1558–1563.

[7] P. Roy and C. Chowdhury, "A survey of machine learning techniques for indoor localization and navigation systems," *Journal of Intelligent & Robotic Systems*, vol. 101, no. 3, p. 63, 2021.

[8] M. Schulz, D. Wegemer, and M. Hollick, "Nexmon: The c-based firmware patching framework," *Res. Gate*, 2017.

[9] F. Gringoli, M. Schulz, J. Link, and M. Hollick, "Free your csi: A channel state information extraction platform for modern wi-fi chipsets," in *Proceedings of the 13th International Workshop on Wireless Network Testbeds, Experimental Evaluation & Characterization*, 2019, pp. 21–28.

[10] G. Forbes, S. Massie, and S. Craw, "Wifi-based human activity recognition using raspberry pi," in *2020 IEEE 32nd International Conference on Tools with Artificial Intelligence (ICTAI)*. IEEE, 2020, pp. 722–730.

[11] G. L. Stuber, J. R. Barry, S. W. Mclaughlin, Y. Li, M. A. Ingram, and T. G. Pratt, "Broadband mimo-ofdm wireless communications," *Proceedings of the IEEE*, vol. 92, no. 2, pp. 271–294, 2004.

[12] S. B. Kotsiantis, D. Kanellopoulos, and P. E. Pintelas, "Data preprocessing for supervised leaning," *International journal of computer science*, vol. 1, no. 2, pp. 111–117, 2006.

[13] M. Awad and R. Khanna, *Efficient learning machines: theories, concepts, and applications for engineers and system designers*. Springer nature, 2015.

[14] W. F. Hassen and J. Mezghani, "Cnn based approach for indoor positioning services using rssi fingerprinting technique," in *2022 International Wireless Communications and Mobile Computing (IWCMC)*. IEEE, 2022, pp. 778–783.

[15] Y. Zhang, C. Wu, and Y. Chen, "A low-overhead indoor positioning system using csi fingerprint based on transfer learning," *IEEE Sensors Journal*, vol. 21, no. 16, pp. 18 156–18 165, 2021.

[16] J. Wang and J. Park, "An enhanced indoor positioning algorithm based on fingerprint using fine-grained csi and rssi measurements of ieee 802.11 n wlan," *Sensors*, vol. 21, no. 8, p. 2769, 2021.

Threat Detection based on System Credibility by Logging Analysis and Visualization

Wei Qiao
Purple Mountain Laboratories
Nanjing, China
qiaowei@pmlabs.com.cn

Youjun Bu
Information Engineering University
Zhengzhou, China
buyoujun@pmlabs.com.cn

Yao Chen
Purple Mountain Laboratories
Nanjing, China
chenyao@pmlabs.com.cn

Xiaoxiao Jiang
Purple Mountain Laboratories
Nanjing, China
jiangxiaoxiao@pmlabs.com.cn

Bingbing Jiang
Purple Mountain Laboratories
Nanjing, China
jiangbingbing@pmlabs.com.cn

***Abstract*—The novel theory of Endogenous Safety and Security was proposed from a system architecture perspective is striving to address the current complex cybersecurity threats dilemma. It utilizes multiple heterogeneous and functionally equivalent systems (called mimic systems) to detect threats because different implementations have different vulnerabilities and are dynamically scheduled on some feedback strategies, making it impossible for a single attack to simultaneously compromise all of these implementations. The threat detection heavily relies on the adjudication of outputs from multiple heterogeneous and functionally equivalent systems because it is possible to adjust the outcomes of the majority of compromised systems as correct. Therefore, the credibility of the adjudication should be evaluated for verifying the trustworthiness of the mimic system, but no research is currently available on the system credibility in the mimicry environment. In this paper, we propose a logging analysis algorithms to evaluate the credibility of the adjudication which is related to each single system's disturbance event history, disturbance factors, disturbance number and one-time runtime duration. The experiments also prove the positive performance of the proposed algorithm. The lower the credibility, the higher the possibility of the system being compromised.**

***Index Terms*—cybersecurity, threat detection, credibility.**

I. Introduction

To address the current complex cybersecurity threats dilemma, several innovative approaches have been proposed, including Moving Target Defense (MTD) [1], Zero Trust Architecture [2], Cyber Resilience [3], and Endogenous Safety and Security [4] [5]. Unlike the latter two techniques, which are still in the early stages of development, the former techniques offer systematic implementation methods. However, when it comes to MTD, there is a concern regarding potential leaks due to redundancy considerations. With MTD, only one target is active at any given time, regardless of how the targets move. This creates non-negligible opportunities for successful attacks. To address this issue, the technique of Endogenous Safety and Security encompasses dynamic, varied, and redundant properties. Building upon this concept, J. Wu proposed the Dynamic, Heterogeneous and Redundant (DHR) architecture [4] [5], which is capable of defending against ”unknown unknown” threats. Concrete implementations of this approach, such as mimic routers, mimic web servers, and mimic cloud systems [16], have been developed.

In simpler terms, the principle of DHR relies on the idea that it is extremely difficult for a single attack to penetrate multiple implementations that have different functionalities but are equivalent in performance. This is because vulnerabilities in software or hardware from different manufacturers, or in different operating systems like Windows, Linux, or macOS, are often unique to each of them. Leveraging this fact, DHR's collective awareness surpasses the limited perception of individual components, making it more resilient against (unknown) threats. In the current approach, multiple components with equivalent functionality process the same input, and the verdict module compares their outputs based on predefined rules to determine the final result. However, in this process, all the components are treated equally, regardless of their vulnerability. In reality, the more vulnerable a component is, the less trustworthy it should be. Therefore, it is important to evaluate the credibility of each component individually rather than giving them equal trust in the final verdict.

Log analysis proves to be a valuable tool in addressing the aforementioned concerns. In simple terms, it helps in determining the credibility of each component by analyzing the logs generated by the mimic devices and assessing the potential threat perturbations they have experienced. The idea is that, if an executor suffers a larger number of threat perturbations, which are reflected in the logging system, it becomes less trustworthy. Furthermore, by sharing and analyzing threat log information among all the mimic devices, the entire mimic network can achieve a collective threat awareness. This means that the swarm awareness of an individual mimic device can be extended to the entire network through log analysis. By leveraging this approach, the network can effectively detect and respond to potential threats based on the insights gained from the analysis of the logs. Therefore, we assess the credibility of the component participating in the adjudication process by evaluating its history of being attacked, the reasons and frequency of the attacks, and whether its runtime duration is normal. Then, by combining the credibility of each component with the adjudication strategy, we can calculate the trustworthiness of the final output result and determine whether cyber threats are detected.

The contributions of this paper can be summarized as follows:

- Log analysis for credibility assessment: This paper suggests leveraging log analysis as a powerful tool for assessing the credibility of individual executors within the DHR architecture. By analyzing the threat log information recorded by mimic devices, we establish a correlation between the suffered threat perturbations and the credibility of the executors. This contributes to the overall security and reliability of the system.
- Mitigation of unknown threats: This paper highlights the capability of the DHR architecture, especially when combined with log analysis, to defend against ”unknown unknown” threats. By integrating dynamic, varied, and redundant properties within the architecture and utilizing log analysis for credibility assessment, the system becomes more resilient and capable of addressing unforeseen or evolving cyber threats.
- Practical implementations: The experiments for the DHR architecture, such as mimic routers, mimic web servers, and mimic cloud systems demonstrate the feasibility and effectiveness of the proposed approach in enhancing cybersecurity and mitigating threats in various domains.

The rest of the paper is structured as follows. The related work is presented in Section 2. We present the tool of log system of mimic devices for collecting, processing and analyzing threat data In Section 3. The system credibility evaluation algorithm is introduced in Section 4. The experiment is presented in Section 5. Finally, we conclude our work in Section 4.

II. Related Work

Security Information and Event Management (SIEM) is an advanced technology that combines Security Information Management (SIM) and Security Event Management (SEM). It offers real-time monitoring and analysis of events, as well as the tracking and recording of security data for compliance or audit purposes. It encompasses various functions such as log management, event correlation and analysis, event monitoring and security alerts, network visualization, and threat intelligence. Although there are threat log analysis tools like Splunk [6], Elastic stack [7], and Azure Sentinel [9] based on SIEM, they do not incorporate the specific characteristics of mimicry systems, thus lacking the ability to reflect the swarm awareness of threat from the log analysis level. Furthermore, existing research [10]–[14] on threat awareness or hunting through log analysis does not consider mimic systems. A proposed framework [15] for threat analysis in a heterogeneous log environment focuses on different types of information systems, excluding both mimic and heterogeneous systems, making it unsuitable for such environments.

III. Log System of Mimic Devices

Based on the standard for mimic logs, we have designed and developed a cloud-based management system specifically for mimic logs. This system provides visual management and analysis capabilities for mimic logs, allowing for the detection and warning of potential threats. Security personnel can then take prompt and effective defensive actions, such as implementing patches, upgrades, and executing cleanup procedures, to counter cyberattacks.

The mimic log cloud management system effectively reduces service interruptions caused by differential mode disturbances and minimizes the likelihood of common-mode escape. Additionally, it supports accurate and efficient scheduling and decision-making based on data. The system offers several key functionalities related to log data, including centralized collection, unified preprocessing, normalized parsing, taxonomy indexing, centralized storage, real-time querying, multidimensional analysis, and visualization. These features enable comprehensive management and analysis of log data, empowering security personnel to identify and respond to threats in a timely and efficient manner. Here, we mainly describe what capabilities the log system should have, as well as the implementation methods or tools of various capabilities, without involving specific implementation details.

- **Collection**: Our system is designed to simplify the distribution and deployment process of various log collectors, such as filebeat and metricbeat, across numerous mimic devices. It empowers the distributed collection of log data while maintaining centralized reception and supports an array of log collectors, accommodating diverse use cases. Once deployed, it facilitates the collection of log data from these distributed devices. Through centralized reception, the log data from a single location can be conveniently accessed and analyzed. This centralized approach enhances efficiency and provides a comprehensive overlook of the log data generated by the mimic devices. Our system can streamline the distribution and deployment of log collectors, enabling efficient distributed log collection and centralized log data reception.
- **Preprocessing**: This program is designed to handle log data transmitted to the system with a wide array of operations. It offers features such as field filtering, allowing users to extract specific fields from the log data based on their needs. The program also supports field format conversion to transform the format of fields within the log data, ensuring compatibility and consistency. Additionally, it provides functionality for field duplicate removal, eliminating redundant entries and improving data integrity. With these capabilities, users can efficiently manipulate log data to derive meaningful insights and optimize analysis processes.
- **Normalized Parsing**: By adhering to the standard mimic log format, this program simplifies the development of log data parsing programs. It achieves this by implementing a unified field encoding format, ensuring normalized

log data parsing. As a result, developers can save time and effort in handling various log formats and focus on core functionalities. The use of a consistent field encoding format allows for seamless integration with existing parsing tools and libraries. This streamlined approach greatly enhances the efficiency of log data parsing, resulting in faster and more accurate analysis of log data. With this program, developers can optimize their parsing workflows and maximize the value extracted from log data.

- **Categorical Index**: With this program, users have the capability to generate both typal and statistical log-data indexes tailored to their unique requirements. These indexes serve as powerful tools for querying and analyzing log data. Users can create typal indexes to categorize log data based on predefined patterns or structures, enabling efficient searching and filtering. Additionally, statistical indexes enable users to extract meaningful insights by aggregating and analyzing log data based on statistical measures such as counts, averages, and trends. Overall, these customizable indexes empower users to unlock the full potential of their log data for query and analysis purposes.
- **Centralized Repository**: After performing the three steps mentioned above, the log data from diverse mimic devices is centralized into a central storage system. This centralized storage ensures that log data is securely stored and easily accessible for query and analysis purposes. With a centralized storage system, users can conveniently retrieve and analyze log data without the need to navigate through multiple sources or locations. Moreover, this centralized storage system provides reliable data backup capabilities.
- **Real-time retrieval**: The system is equipped to collect real-time logs from a wide range of mimic devices. It can gather logs related to verdicts, schedules, and performance metrics from devices such as routers, switches, WEB, Unified Data Management (UDM), Home Subscriber Server (HSS), and Advanced Driver Assistance Systems (ADAS). By collecting logs from these diverse devices, the system offers a comprehensive view of the network and its components. This allows for a holistic understanding of system performance and operational status. Furthermore, the accumulated logs can be promptly queried. Users can easily access and retrieve specific logs based on their requirements. This real-time log querying capability enables users to monitor system events and performance metrics, aiding in troubleshooting, performance optimization, and overall network management.
- **Comprehensive Analysis**: In addition to statistical analyses, this system offers correlation analysis of log data to address challenges in analyzing mimic component or device logs. Correlation analysis identifies meaningful relationships and patterns, revealing hidden insights not apparent through individual statistical analyses. These correlations provide a deeper understanding of interactions among log events or variables, facilitating comprehensive data analysis. By leveraging deep learning technology, the system efficiently processes and analyzes large log data volumes, extracting complex patterns and empowering users to make informed decisions.
- **Visualization**: This system provides powerful visualization capabilities for mimic logs, allowing users to gain insights and understand the data more intuitively via Kibana [8]. Users can visualize various types of information, including raw logs, current system status, analysis results, numerical distributions, and trend predictions. By representing log data visually, users can easily identify patterns, anomalies, and trends, facilitating quicker and more effective analysis of the system's behavior and performance. These visualizations offer a comprehensive overview of the system's performance and help users identify potential bottlenecks, optimize resource allocation, and make data-driven decisions for improving the overall functionality and efficiency of the mimic devices.

IV. Log Analysis for Credibility

A. Factors Influencing Credibility

In order to enhance the resilience of the DHR architecture against "unknown" threats in real-world engineering, it is crucial to assess the credibility of executors. Executors (referring to heterogeneous and functionally equivalent systems) play a significant role in determining the credibility of the final verdict results, which indicate whether mimic devices recognize and respond to threats. Thus, measuring the credibility of executors becomes essential.

To measure the credibility of executors, the first step is to identify the factors that influence the reliability of their outputs. Based on the mimic defense theory and benchmark function experiments conducted by Wu et al. [16], we have identified the following four key factors related to executors. Other influences can affect the trustworthy of executors' outputs, but the experiments show that the four factors below characterize the executors' credibility.

1) **Disturbance Event History**: If an executor has previously experienced disturbances, it indicates a potential security risk, and therefore, its credibility should be diminished. This is because the ability of an executor to reliably produce accurate results may be compromised due to the past disturbances. Consequently, it is necessary to reduce the credibility of such an executor in order to maintain the overall trustworthiness of the system. Furthermore, it is important to analyze the specific characteristics and patterns of past disturbances encountered by the executor. By studying the nature and extent of these disturbances, one can gain insight into the potential vulnerabilities or weaknesses of the executor, thereby providing strategies to mitigate the associated security risks. This additional analysis helps in enhancing the overall security and reliability of the system by effectively addressing the identified risks.
2) **Disturbance Factors**: When an executor experiences the same disturbed cause as a previous instance, it implies

that the executor possesses a recognized weakness that has been reused. As a result, their credibility should be diminished. This emphasizes the importance of identifying patterns in an executor's behavior and taking necessary precautions to mitigate potential risks associated with their known weaknesses. By acknowledging and addressing these weaknesses, trust in the executor can be maintained or restored.

3) **Disturbance Number**: If an executor has a disturbed times greater than the average of all executors' disturbed times, it indicates that the executor is more susceptible to attacks. Consequently, their credibility should be diminished. This suggests the need for additional scrutiny and security measures to protect the executor and prevent any potential breaches. It is crucial to identify the reasons behind the increased vulnerability and take appropriate actions to strengthen the executor's defenses. By reducing their credibility, it serves as a cautionary measure to ensure that the executor's actions are carefully monitored and their weaknesses are addressed promptly.
4) **One-time Service Runtime Duration**: If an executor runs for an extended period of time, surpassing the predefined threshold for a standard runtime, it is deemed to be potentially disrupted or compromised, and, as a result, its credibility should be diminished. Additional measures may need to be taken to investigate the root causes of the prolonged runtime and ensure the reliability and security of the executor.

B. Credibility of Executors

Based on the previous analysis, the credibility measurement algorithm for executors is designed as follows:

- **Step 1:** Set E as the current evaluated executor, DEH_list as the executors' disturbance event history, DF_map and DN_map as two maps storing executors and their disturbance factors and number respectively, E_cred as the credibility of the executor and the initial value 1.
- **Step 2:** Query the list and determine whether E is in DEH_list. If $DEH\_list.Query(E)$, $E\_cred = E\_cred - \alpha_h$; ($\alpha_h$ is the weight value of the historical disturbed factor).
- **Step 3:** Query the map DF_map and determine whether E has had the current disturbed factor before. If $DF\_map.find(E).Query(dc)$ (dc is the current disturbance factor), $E\_cred = E\_cred - \alpha_f (\alpha_f$ is the weight value of the disturbed cause factor).
- **Step 4:** Query the map DN_map and calculate E's disturbed frequency and all executors' average value. Set $en = DN\_map.find(E)$, $an = DN\_map.avg()$; If $en > an$, $E\_cred = E\_cred - \alpha_n$ (α_n is the weight value of the disturbance number).
- **Step 5:** Calculate E's running time t at this service. If $t > T$ (T is the threshold of the runtime of one normal service), $E\_cred = E\_cred - \alpha_t$ (α_t is the weight value of the runtime factor).
- **Step 6:** Output the executor E's credibility $E\_cred.$

This method involves the collection and documentation of significant indicators that influence the credibility of executors. Through a combination of experience and iterative experimental testing, the algorithm determines the appropriate weights for each of these factors. By assigning weights to the different indicators, the algorithm can effectively measure the relative importance of each factor in determining the overall credibility of an executor. This enables a comprehensive and systematic evaluation of an executor's trustworthiness. The algorithm dynamically calculates the credibility of each executor based on the weighted factors and their corresponding values. This dynamic output reflects the evolving nature of an executor's credibility, as it can be influenced by changes in performance, client feedback, or other relevant factors.

C. Credibility of System

Several methods can be used to measure the credibility of the verdict results based on the credibility measurement of each online executor. One common approach is to calculate the average credibility of the executors, either simply or conditionally. In this paper, we propose a method that takes into account the current verdict information. Typically, in the majority verdicts, the opinions of the minority do not significantly impact the final result. However, to ensure accuracy, adjustments can be made based on the mean confidence of the online executors.

To implement this method, the average credibility of all executors can be calculated, and then adjusted based on the level of confidence expressed by the majority. By weighing the credibility scores of the executors with their corresponding confidence levels, a more refined measurement of credibility can be obtained. This approach allows for a more nuanced assessment of the credibility of verdict results, taking into consideration both the collective opinion of the executors and the level of confidence they exhibit. It enhances the accuracy and reliability of the final verdict by appropriately weighting the influence of each online executor. Assume the number of online executors is $m = 2k + 1$.

- If more than k executors satisfy one of the above four factors, the confidence of the verdict result should be reduced to that of all satisfying the factor.
- If it is less than k, the confidence should be increased to that of none satisfying the factor.
- If exactly k, we also check if these k executors are the current abnormal executors.

The calculation formula is as follows:

$$V\_cred = \frac{1}{m} \sum_{E \in VLR.set} E\_cred + \sum_{i \in \{h,f,n,t\}} \Delta_i(VLR)$$

$$\Delta_i(VLR) = \begin{cases} \frac{Count(VLR,i)}{m} \times \alpha_i & , if\ j < k; \\ \frac{1}{2m} \times Norm(VLR, i) \times \alpha_i & , if\ j = k \\ -\frac{m-j}{m} \times \alpha_i & , otherwise. \end{cases}$$

$$Count(VLR, i) = \sum_{E \in VLR.set} isFactor(E, i)$$

$$Norm(VLR, i) = \begin{cases} 1 & , if\ VLR.abnorm \notin VLR.set(i); \\ 0 & , otherwise. \end{cases}$$

$$isFactor(E, i) = \begin{cases} 1 & , if\ E\ has\ the\ factor\ i; \\ 0 & , otherwise. \end{cases}$$

$$i \in \{h, f, n, t\}.$$

where VLR is the current verdict log record, $VLR.set$ is the set of online executors contained in VLR, $VLR.set(i)$ is the set of online executors in VLR satisfying the factor i, $VLR.abnorm$ is the abnormal executor in VLR, V_cred is the credibility of the verdict result, E_cred is the credibility of the online executor E, $\{h, f, n, t\}$ represents the above four factors, and Δ_i stands for the tuning parameter of the corresponding factor.

V. Experiments

In order to validate the threat awareness of mimic devices and mimic networks, an experiment was conducted in the Network Endogenous Security Testbed (NEST) environment. The experiment utilized a hardware setup consisting of an Intel(R) Xeon(R) Silver 4214R CPU with a clock speed of 2.40GHz and 12 processor cores. The software environment employed was CentOS Linux version 7.4.1708 (Core). To facilitate the experiment, several applications were utilized: Filebeat 8.3.3 (Linux-x86_64), Kafka 2.0, Logstash 8.1.0, Elasticsearch 8.1.0, and Kibana 8.1.0. The threshold parameters in our algorithm were all settled as 0.1 (this number is arrived through the parameter debugging according to expert advice) in the following experiment. Within this experimental setup, the mimic devices and mimic networks were subjected to various scenarios and situations to assess their threat awareness. The aim was to evaluate how well these entities could detect and respond to potential threats in a realistic and controlled environment. By conducting the experiment in the NEST environment and employing the aforementioned hardware and software components, the study aimed to gain insights into the effectiveness of the mimic devices and networks in identifying and mitigating potential security risks. The results obtained from the experiment will contribute to further enhancing the security measures and threat awareness capabilities of these systems.

By utilizing the mimic log system and implementing the proposed algorithm, the credibility scores of the verdict results are computed. This allows for the determination of the success of disturbances in the system. The obtained results are then compared with real-world data, enabling an assessment of the accuracy and precision of our algorithm. The mimic log system aids in generating credibility scores for the verdicts produced. These scores serve as an indication of the reliability and trustworthiness of the results. Based on this information, it becomes possible to determine whether any disturbances or attacks have successfully affected the system. To evaluate the performance of our model, a comparison is made between the results obtained from the threat awareness model and the actual observed data. This analysis allows us to assess the accuracy and precision of our model in detecting and responding to potential threats. By considering the consistency and alignment between the model's predictions and the real-world outcomes, we can gauge the effectiveness of our threat awareness system. Ultimately, this evaluation process provides valuable insights into the capabilities and limitations of our model.

- **Accuracy**: The proportion of correct forecast quantities to total quantities in both positive and negative cases.

$$Accuracy = \frac{TP + FN}{TP + FP + TN + FN}$$

- **Precision**: Percentage of correct prediction within the sample with positive prediction.

$$Precision = \frac{TP}{TP + FP}$$

- TP (True Positives): The positive result predicted by the model is consistent with the actual result of the disturbance suffered by the mimic system.
- FP (False Positives): The positive result predicted by the model is the opposite of the actual result of the normal operation of the mimic system.
- FN (False Negatives): The negative result predicted by the model is consistent with the actual result of the normal operation of the mimic system.
- TN (True Negatives): The negative result predicted by the model is the opposite of the actual result of the disturbance suffered by the mimic system.

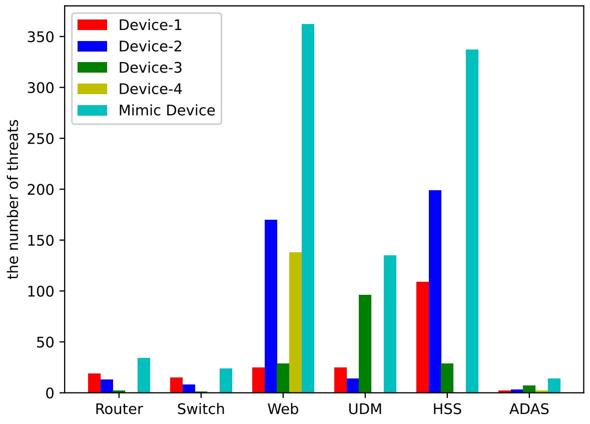


Fig. 1. The number of threats detected by a single-device vs. mimic device.

Figure 1 shows that the greater the number of heterogeneous and functionally equivalent devices, the greater the reliability and confidence in the verdict results, as well as enhanced threat detection capabilities, and Figure 2 illustrates that the threat detection capability of a combination of different types of mimic devices is stronger than that of a single mimic device. Figure 3 presents a comparison between the results of our threat detection algorithm in the experiment and the actual results.

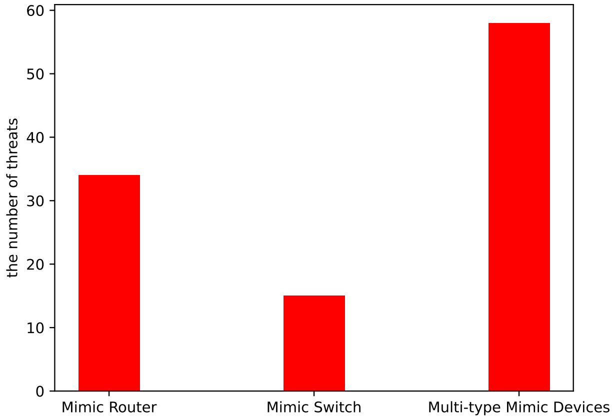


Fig. 2. The number of threats detected by a mimic device vs. multi-type mimic devices.

$$Accuracy = \frac{179 + 93}{179 + 71 + 11 + 93} = 0.768$$

$$Precision = \frac{93}{11 + 93} = 0.894$$

In the experiment, we set the scoring threshold as 0.85. That is, a score of less than or equal to 0.85 indicates a positive case prediction, whereas a score greater than or equal to 0.85 indicates a negative case prediction. We can calculate our model's accuracy as 0.768 and precision as 0.894 which shows it can effectively perceive threats.

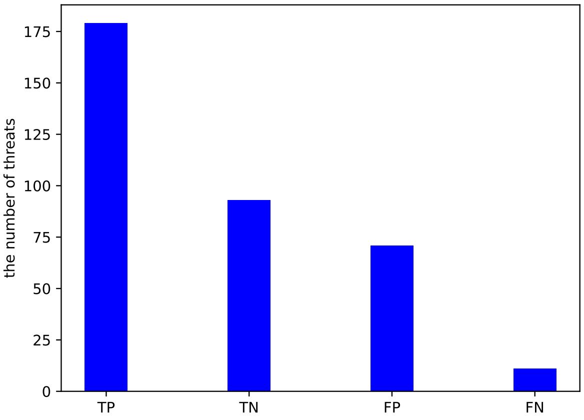


Fig. 3. The result of our threat detection algorithm.

VI. Conclusion

In this paper, a threat detection model based on system credibility is proposed to enhance the network defense capabilities of the entire mimic network. It is specifically designed to handle the log processing and association analysis tasks of mimic devices and serves as a centralized platform for collecting, analyzing, and correlating logs from multiple mimic devices. It also enhances the overall situational awareness by providing a comprehensive view of the network defense status and facilitating prompt detection and response to potential threats. To evaluate the effectiveness of the threat detection model, a verification experiment is conducted in the NEST cyber range. The experimental results demonstrate the capability of the model in effectively perceiving and identifying threat events within the mimic defense context.

Acknowledgment

B. Jiang is supported in part by Jiangsu Province ”Shuangchuang” Plan under Grant JSSCBC20221657. Y. Bu is supported in part by NSFC under Grant 62176264.

References

[1] S. Jajodia, A. K. Ghosh, V. Swarup, C. Wang, and X. S. Wang, “Moving Target Defense - Creating Asymmetric Uncertainty for Cyber Threats,” Advances in Information Security. Springer, August 2011.
[2] S. W. Rose, O. Borchert, S. Mitchell, and S. Connelly, “Zero Trust Architecture,” NIST Special Publication 800-207. August 2020.
[3] R. Ross, P. Viscuso, G. Guissanie, K. Dempsey, and M. Riddle, “Protecting Controlled Unclassified Information in Nonfederal Systems and Organizations,” NIST Special Publication 800-171(R.2). February 2020.
[4] J. X. Wu, “Development paradigms of cyberspace endogenous safety and security,” Sci. China Inf. Sci. vol. 65, pp. 1–3, March 2022.
[5] J. X. Wu, “Problems and solutions regarding generalized function safety in cyberspace,” Security and Safety. vol. 1, pp. 2022001, June 2022.
[6] Splunk Enterprise 9.1.1, Splunk Inc. https://www.splunk.com/en_us/download/splunk-enterprise.html.
[7] Elasticstack, Elastic Security. https://www.elastic.com/security.
[8] Kibana, Elastic Security. https://www.elastic.co/cn/downloads/kibana.
[9] Microsoft Sentinel, Microsoft. https://azure.microsoft.com/en-us/products/microsoft-sentinel/.
[10] R. Yamagishi, T. Katayama, N. Kawaguchi, and T. Shigemeto, “HOUND: Log Analysis Support for Threat Hunting by Log Visualization,” The 12th International Congress on Advanced Applied Informatics (IIAI-AAI), pp. 653–656, 2022.
[11] K. Lamshöft, T. Neubert, J. Hielscher, C. Vielhauer, and J. Dittmann, “Knock, Knock, Log: Threat Analysis, Detection & Mitigation of Covent Channels in Syslog Using Port Scans as Cover,” Forensic Science International: Digital Investigation. vol. 40, no. Supplement, pp. 301335, April 2022.
[12] A. S. Malik, M. K. Shahzad, and M. Hussain, “A Forensic Framework for Webmail Threat Detection Using Log Analysis,” The 14th International Conference on Innovative Security Solutions for Information Technology and Communications (SecITC), pp. 57–69, 2021.
[13] L. Liu, C. Chen, J. Zhang, O. Y. de Vel, and Y. Xiang, “Doc2vec-based Insider Threat Detection through Behaviour Analysis of Multi-source Security Logs,” The 19th IEEE International COnference on Trust, Security and Privacy in Computing and Communications (TrustCom), pp. 301–309, 2021.
[14] T. Qin, Y. Gao, L. Wei, Z. Liu, and C. Wang, “Potential Threats Mining Methods based on Correlation Analysis of Multi-type Logs,” IET Networks, vol. 7, no. 5, pp. 299–305, 2018.
[15] J. Navarro et al., “HuMa: A Multi-layer Framework for Threat Analysis in a Heterogeneous Log Environment,” The 10th International Symposium on Foundations and Practice of Security, vol. 10723, pp. 144–159, 2017.
[16] J. X. Wu, “Principles of Cyberspace Mimicry Defense: Generalized Robust Control and Endogenous Security,” Wireless Networks. Springer, December 2019.

Location and Object-Based Mobile Applications

Development Based on Use Case Templates and Visual Programming

Martin Zimmermann
Department of Economics
Offenburg University
Offenburg, Germany
email: m.zimmermann@hs-offenburg.de

Abstract **- The main advantage of mobile context-aware applications is to provide effective and tailored services by considering the environmental context, such as location, time, nearby objects and other data, and adapting their functionality according to the changing situations in the context information without explicit user interaction. The idea behind Location-Based Services (LBS) and Object-Based Services (OBS) is to offer fully-customizable services for user needs according to the location or the objects in a mobile user's vicinity. However, developing mobile context-aware software applications is considered as one of the most challenging application domains due to the built-in sensors as part of a mobile device. Visual Programming Languages (VPL) and hybrid visual programming languages are considered to be innovative approaches to address the inherent complexity of developing programs. The key contribution of our new development approach for location and object-based mobile applications is a use case driven development approach based on use case templates and visual code templates to enable even programming beginners to create context-aware mobile applications. An example of the use of the development approach is presented and open research challenges and perspectives for further development of our approach are formulated.**

Keywords - Location-Based Services; Object-Based Services; Mobile Applications; Visual Programming.

I. INTRODUCTION

Sensors enable the creation of context-aware mobile applications in which applications can discover and take advantage of contextual information, such as user location, nearby people and objects. As a consequence, context-aware mobile applications can sense clues about the situational environment making mobile devices more intelligent, adaptive, and personalized.

Context has been defined as any required knowledge to identify the current situation of a person or object in order to provide tailormade services. The situational environment of a mobile user becomes more vital in mobile applications where the context, e.g., geo position of a user can change rapidly. For example, depending on its current location, a tourist would like to see relevant tourist attractions on a map together with distance information. Mobile applications can obtain the context information in various ways in order to provide more adaptable, flexible and user-friendly services.

A combination of context-aware applications and mobile devices provides a novel opportunity for both end users and application developers to obtain context and the consequent response to any changes in the context. Hence, the main advantage of mobile context-aware applications is to provide tailored services by considering the environmental context, such as location, time, weather conditions, nearby objects, and adapting their functionality according to the changing situations in the context data without explicit user interaction.

A general definition of context was given by Dey and Abowd [1]: “Any information that can be used to characterize the situation of an entity. An entity is a person, place, or object that is considered relevant to the interaction between a user and an application, including the user and applications themselves.”

Categories of context information that are practically significant include [1]:

- Environmental Context: Includes all the surrounding environmental conditions of current location (like air quality, temperature, humidity, noise level and light condition).
- Activity Context: Defines the user’s current activity including private and professional activities that can be sensed like talking, reading, walking, and running.
- Temporal Context: Consists of temporal factors, such as current time, date, and season of the year.
- Personal (identity) Context: Specifies user’s characteristics and preferences like name, age, sex, contact number, user’s hobbies and interests.
- Spatial Context: Involves any information regarding the position of an entity (person and object), for instance orientation, location, acceleration, speed.
- Vital Signs Context: Covers all information related to health state, such as heart rate, blood pressure, voice tone, and muscle activity.

The combination of spatial, temporal, activity and personal contexts makes the primary context to understand

ISBN: 978-1-68558-106-0

the current situation of entities. These types of contexts can respond to basic questions about when, where, what, and who. The idea behind Location-Based Services (LBS) and Object-Based Services (OBS) as subcategories of context-aware mobile applications is to tailor services according to the location of a user or the objects in a mobile user's vicinity.

In [2], various Model Driven Development (MDD) techniques and methodologies are systematically investigated, i.e., what MDD techniques and methodologies have been used to support mobile app development and how these techniques have been employed, to identify key benefits, limitations, gaps and future research potential. Our approach based on case templates and visual programming is tailored to the needs of mobile programming beginners.

Visual Programming Languages (VPL) and hybrid visual programming languages are considered to be innovative approaches to address the inherent complexity of developing programs [3], [4]. In this work, we introduce an in-depth discussion of a new VPL based method, to enable even programming beginners to create context-aware mobile applications.

The rest of the paper is organized as follows: Section 2 introduces LBS and OBS concepts, especially various technologies to determine the object context of a user. Our proposed development process in terms of use case templates and visual code templates is described in Section 3. Visual programming concepts and the development environment which we use in our projects are introduced in Section 4. Sections 3 and 4 also describe how our approach is applied to an LBS and OBS example in the field of tourism. Finally, the limitations of the VPL approach as well as directions for future research are presented in Section 5.

II. Location and Object Based Services

LBS can be described as applications that are dependent on a certain location. Two broad categories of LBS can be defined as triggered and user-requested [5]. In a user-requested scenario, the user is retrieving the position once and uses it on subsequent requests for location-dependent information. This type of service usually involves either personal location, i.e., finding where you are or services location, e.g., where is the nearest hospital. Examples of this type of LBS are also navigation (usually involving a map) and direction (routing information). A triggered LBS by contrast relies on a condition set up in advance that, once fulfilled, retrieves the position of a given device. An example is in emergency services, where the call to the emergency center triggers an automatic location request from the mobile network.

The idea behind OBS is to tailor a service according to the objects in a mobile user's vicinity. For example, a visitor standing in front of a painting in an art gallery should be provided with additional information about the painting, such as the artist, or even the opportunity to order a print (one button pay). Popular technologies for determining the object context are Quick Response (QR) codes, Near-Field Communication (NFC) tags and beacons.

LBS are typically based on GPS. The position of a person or an object is determined in terms of latitude and longitude. To determine the object context, i.e., the object(s) at which the user is located, various technologies can be used (Table I). An object, e.g., painting in a museum can be provided with one or more of the following elements: Bluetooth Low Energy (BLE) Beacon, NFC tag, and QR code.

Beacons are small wireless, usually battery-powered devices that transmit data at regular intervals using BLE [6], [7]. This mini-radio transmission devices can be 'discovered' and seen by all BLE scanners, e.g., a smartphone within a certain radius. However, beacons do not work by themselves: they require a mobile app. So, in case of an arts gallery or museum, the visitor must have downloaded the mobile application beforehand for the beacon to work.

NFC is a short-range wireless connectivity technology that uses magnetic field induction to enable communication between devices when they're touched together or brought within a few centimeters of each other [8]. NFC builds on the work of the Radio-Frequency Identification (RFID) set of standards and specifications, such as ISO/IEC 14443 and ISO/IEC 15963. By passing a mobile device near an NFC chip, one can read the data it contains and interact with the content. Advantages and disadvantages from a user's point of view of the different technologies for implementing OBS are shown in Table I.

TABLE I. TECHNOLGIES FOR OBJECT BASED SERVICES.

Technology	*Pros*	*Cons*
Beacon	+ no user interactions required	- requires power supply - more expensive (compared with NFC, QR codes)
NFC tag	+ can store the most data + cheap	- user must tap on the item
QR Code	+ well-known technology	- requires most user interactions (open camera app, scan image)

QR codes are a type of matrix bar code that was invented by Denso Wave in 1994 to be used as labels on automotive parts [9]. It allows to store large amount of data (compared to 1D barcodes) and a high-speed decoding process using any handheld device like phones. The popularity of QR code grows rapidly with the growth of mobile users and thus the QR code concept is rapidly arriving at high levels of acceptance worldwide.

III. DEVELOPMENT PROCESS

We propose five steps to perform requirements engineering for location and object-based mobile applications (Figure 1):

- Selection and instantiation of use case templates (input, output, steps)
- Development of the user interface for each use case, e.g., triggering a use case with a button
- Selection and instantiation of visual code templates
- Selection of additional non-visible components, e.g., location sensor, QR code scanner, etc.
- Event based programming (calling methods related to the user interface and the non-visible components).

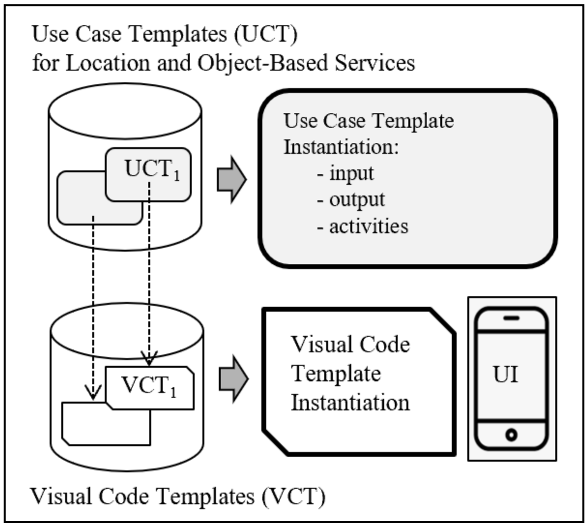


Figure 1. Use Case Templates and Visual Program Code Templates.

Use case templates in Figure 1 are use case specifications that can operate with generic objects and activities. This allows us to create a use case description whose functionality can be adapted to more than one concrete location-based scenarios without repeating the entire description for each scenario. Visual code templates are implementations of use case templates based on visual programming (Section 4). Each use case template is associated with a visual code template.

The use case template in Table II describes an LBS scenario, a search object pattern in terms of input, output and steps to be executed. For example, the template can be applied to visualize some tourist attractions and the current position of a mobile user on a map. Filters are used to display certain tourist attractions, e.g., museums.

Table III illustrates a use case template for an OBS. In this case QR codes are used to identify an object. We defined similar use case templates for OBS using Beacons and NFC tags.

TABLE II. USE CASE TEMPLATE “SEARCH AND SHOW OBJECT(S) ON A MAP”.

Use Case template	*Search and show <object(s)> on a map*
Input	Name/Id/Category of an <object>
Steps	1: Determine the current geo position of the user 2: Show a map with the user's current position as the center point 3: Search the <object(s)> according to the name/id/category (in a list of <object>) if found → Create marker(s) for the <object(s)>
Output	Map with markers: → marker for the current position of the user → marker(s) representing the <object(s)>

The generic part in the use case template in Table III is represented by the activities “Visualize <object property i>”. The concrete visualization is dependent on the scenario to be implemented. E.g., for exhibits, like paintings in a gallery equipped with QR codes, object visualization could mean to represent a link to a video (painter explaining interesting background information) or a link to allow a tourist to buy a print.

TABLE III. USE CASE TEMPLATE “SEARCH AND SHOW OBJECT(S)”.

Use Case Template	*Visualize <object> properties*
Input	QR Code
Steps	1: Scan QR code 2: Search the QR Code in a list of codes if found → Visualize <object property 1> Visualize <object property 2> . . .
Output	Visualized object properties

In the following sections, we describe how our approach is applied to an LBS and OBS example in the field of tourism.

IV. VISUAL PROGRAMMING

Visual programming environments are increasingly used in demanding problem domains, e.g., Internet of Things (IoT) applications [10] or robot applications [11]. For example, Pepper’s popular programming interface is based on visual elements for built-in sensors and actuators [12].

Basic functions of the Pepper platform are provided per sensor and actuator, e.g., open/close hand functions for the actuator hand or detect touch on hand tactile sensor. Flow elements are connected with each other to form business workflows similarly to the visual building blocks of Business Process Model and Notation (BPMN) [13]. BPMN is a graphical representation for specifying business processes in a business process model based on a flowcharting technique very similar to activity diagrams from Unified Modeling Language (UML). BPMN's basic element categories are flow objects (events, activities, gateways), connecting objects (sequence flow, message flow, association), and artifacts (data object, group, annotation).

A. *Development Environment*

We use MIT App Inventor [14] and Thunkable [15], which are both cloud-based visual programming development environments for mobile applications (Android and iOS). The basic concepts are components, events and functions. App Inventor and Thunkable provide the application developer with many different components to use while building a mobile app. Components are chosen on the "Design Screen" and dragged onto the phone (Figure 2).

The properties of these components, such as color, font, speed, etc. can then be changed by the developer. Available component categories are user interface elements, media, storage, location-based services etc. Components can be clicked on and dragged onto the development screen area.

There are two main types of components: visible and non-visible. Visible components, such as buttons, text boxes, labels, etc. are part of the user interface whereas non-visible components, such as the location sensor, QR Code scanner, sound, orientation sensor are not seen and thus not a part of the user interface screen, but they provide access to built-in functions of the mobile device (Figure 2).

Components are based on an object-oriented paradigm, i.e., decomposition of a system (an app) into a number of entities called objects and then ties properties and function to these objects. An object's properties can be accessed only by the functions associated with that object but functions of one object can access the function of other objects in the same cases using access specifiers.

Event handler blocks specify how a program should respond to certain events. After, before, or when the event happens can all call different event handlers. There are two types of events: user-initiated and automatic.

Clicking a button, touching a map, and tilting the phone are user-initiated events. Sprites colliding with each other or with canvas edges are automatic events. Timer events are another type of automatic event. Sensor events function also as user-initiated events. For example, the orientation sensor, the accelerometer, and the location sensor all have events that get called when the user moves the phone in a certain way or to a certain place.

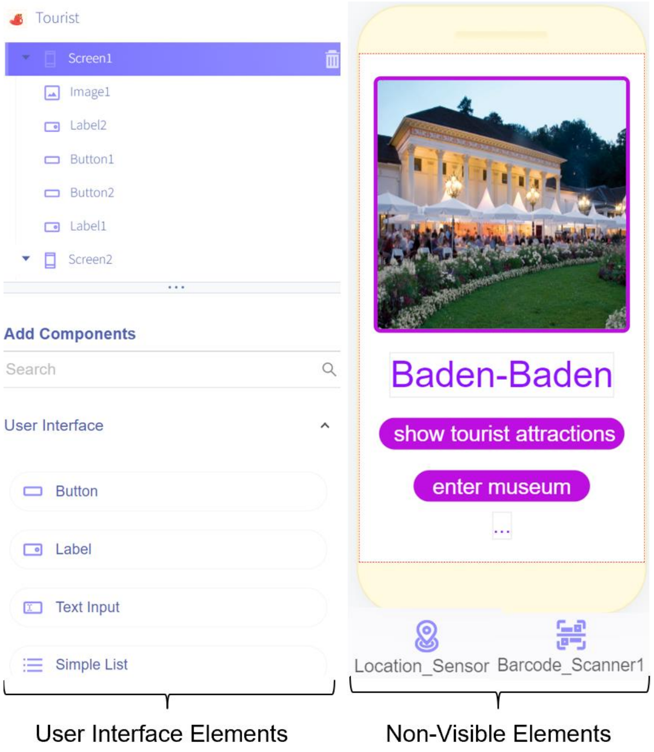
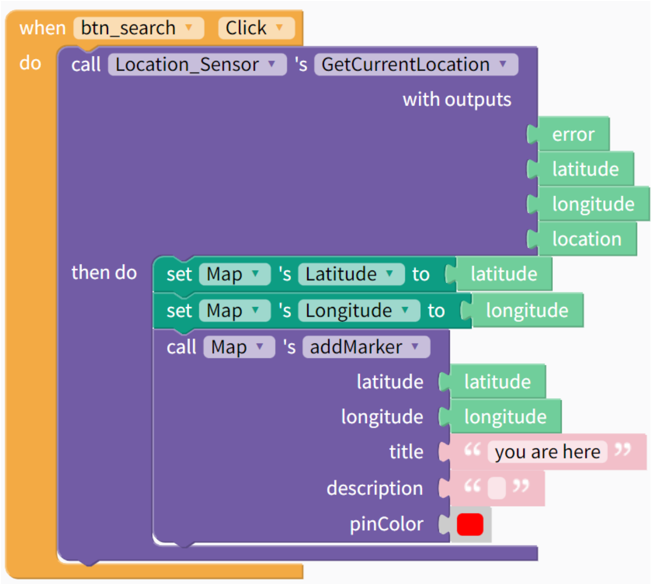


Figure 2. Development Environment.

Figure 3 shows the visual elements of the event-based programming part for a simple LBS.



```
when btn_search Click
do  call Location_Sensor 's GetCurrentLocation
                                    with outputs
                                        error
                                        latitude
                                        longitude
                                        location
    then do  set Map 's Latitude to latitude
             set Map 's Longitude to longitude
             call Map 's addMarker
                            latitude  latitude
                           longitude  longitude
                               title  " you are here "
                         description  " "
                            pinColor  🟥
```

Figure 3. Visual Elements of a Simple LBS.

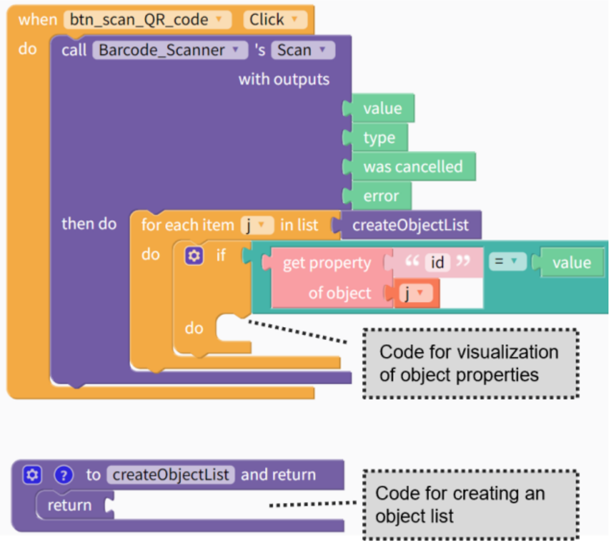
Objects, method calls, arguments and results of method calls are represented by visual elements with different shapes and colors. In the example in Figure 3, first the current position of a user is determined by calling the method

GetCurrentLocation(). The resulting values (latitude and longitude) are used in the next step for the specification of the map center (two set operations). By calling the method addMarker, a marker is created in a third step. The arguments for the last method call are again visual elements (previously calculated values for the current latitude and longitude of the user).

B. *Visual Code Templates*

Each use case template is associated with a visual code template. Figure 4 illustrates the visual code template for the use case template "visualize <object> properties" in Table III. First, the event handler calls the scan method. The resulting id is used to search for the corresponding object in an object list. The instantiation of the template involves

- creation of an object list
- visualization of the object properties.



Code for visualization of object properties

Code for creating an object list

Figure 4. Example Visual Code Template.

The creation of an object list could be based on a local list (as part of a mobile app) or a cloud-based object list. The creation / access to an object list in encapsulated in the function createObjectList(). Finally, the object properties have to be visualized, which is not part of the template, because the concrete visualization are dependent on the scenario to be implemented. E.g., for exhibits, like paintings in a gallery equipped with QR codes, object visualization could mean to create a link to a video (painter explaining interesting background information) or a link to allow a tourist to buy a print.

V. CONCLUSION

The main advantage of mobile context-aware applications is to provide an effective, usable, rapid service by considering the environmental context, such as location, time, nearby objects, and adapting their functionality according to the changing situations in context data. LBS and OBS represent two main categories of context-aware applications. Use case templates and visual code templates are particularly well suited for programming beginners.

Meanwhile, visual programming environments are increasingly used in demanding problem domains, e.g., IoT applications [10]. The development of use cases templates (in the sense of requirements engineering) as the starting point of an app project has proven to be very advantageous. Representing programming language concepts by using visual elements with different shapes and colors fits well with the object-oriented approach.

A main drawback of the used programming environments is the identification and handing of runtime errors due to the lack of integrated debugging functions. However, our use case centered approach leads normally to manageable runtime error because each use case is developed and tested as a separate unit.

Future work will focus on the development of patterns, which are a well-known concept in the traditional software engineering. An architectural pattern is a general, reusable solution to a commonly occurring problem in software architecture. Patterns become reusable solutions for a common set of problems in software development, addressing issues like high availability, performance, and risk minimization. Additionally, we are going to implement additional components (called extensions), e.g., an NFC component offering more powerful and flexible functions and events. Extension components can be used in building projects, just like other built-in components. The difference is that extension components can be distributed on the Web and loaded into the development environment dynamically.

REFERENCES

[1] A. K. Dey and G. D. Abowd, "Towards a Better Understanding of Context and Contextawareness," CHI 2000 Workshop on The What, Who, Where, When, Why and How of Context-awareness, pp. 1–6, 2000.

[2] Md. Shamsujjoha, J. Grundy, L. Li, H. Khalajzadeh, and Q. Lu, "Developing Mobile Applications Via Model Driven Development: A Systematic Literature Review", Information and Software Technology, vol. 140, December 2021.

[3] M. Idrees and F. Aslam, "A Comprehensive Survey and Analysis of Diverse Visual Programming Languages," VFAST Transactions on Software Engineering, vol.10, no. 2, pp. 47–60, 2022.

[4] R. Daskalov, G. Pashev, and S. Gaftandzhieva, "Hybrid Visual Programming Language Environment for Programming Training," TEM Journal, vol. 10, issue 2, pp. 981–986, 2021.

[5] F. F. Chamasemani and L. S. Affendey, "Impact of mobile context-aware applications on human computer interaction," Journal of Theoretical and Applied Information Technology, vol. 62, no.1, pp. 281–287, 2014.

[6] T D'Roza and G Bilchev, "An overview of location-based services," BT Technology Journal, vol. 21, no. 1, pp. 20–27, 2003.

[7] R. Faragher and R. Harle, "Location Fingerprinting With Bluetooth Low Energy Beacons," in IEEE Journal on Selected Areas in Communications, vol. 33, no. 11, pp. 2418–2428, 2015.

[8] V. Coskun, B. Ozdenizci, and K. Ok, “A Survey on Near Field Communication (NFC) Technology”, Wireless Pers Commun 71, pp. 2259–2294, 2013.

[9] S. Tiwari, "An Introduction to QR Code Technology," 2016 International Conference on Information Technology (ICIT), Bhubaneswar, India, pp. 39–44, 2016.

[10] M. Silva, J. P. Dias, A. Restivo, and H. S. Ferreira, “A Review on Visual Programming for Distributed Computation in IoT”, Springer Nature Switzerland AG 2021, M. Paszynski et al. (Eds.): ICCS 2021, LNCS 12745, pp. 443–457, 2021.

[11] M. Zimmermann, “Teaching Visual Programming: Humanoid Robot Programming as a Case Study”, 15th International Conference On Education and New Learning Technologies, pp. 6143-6149, 2023.

[12] A. M. Marei et al., “A SLAM-Based Localization and Navigation System for Social Robots: The Pepper Robot Case”, in Machines, vol. 11, issue 2, 2023.

[13] http://www.omg.org, Accessed July 2023.

[14] MIT App Inventor. https://appinventor.mit.edu, Accessed July 2023.

[15] Thunkable. https://www.thunkable.com, Accessed July 2023.

WiFi CSI/RSSI Fingerprints Positioning based on Data Augmentation Technique

Mohamed Amin Elaoud and Wiem Fekih Hassen

Chair of Distributed Information Systems, University of Passau, Innstraße 41, 94032 Passau, Germany
Email: {firstname.lastname}@uni-passau.de

***Abstract*—In today's digitally connected world, Indoor Positioning Systems (IPS) are of paramount importance, especially for applications within enclosed spaces such as buildings. Leveraging the widespread deployment of WiFi technology, this paper presents an IPS that hinges on WiFi signal data specifically, Received Signal Strength Indicator (RSSI) and Channel State Information (CSI), and the powerful generative model, Tabular Generative Adversarial Network (TabGAN). The work entails a meticulous data collection process conducted within a controlled laboratory environment at the University of Passau, in Germany. Subsequently, data augmentation through GAN is employed to enrich the dataset. The augmented data is then evaluated using LightGBM and Convolutional Neural Network (CNN) models, with the Root Mean Square Error (RMSE) as the primary metric and Positioning Error for comprehensive evaluation of the IPS's accuracy and positioning capabilities. The IPS achieved a remarkable result of 0.99 meters for LightGBM and 0.8 meters for CNN, showcasing its high accuracy on unseen data and validating the efficacy of GAN-based data augmentation for enhancing indoor positioning capabilities.**

***Keywords*—IPS, CSI, RSSI, private dataset, Raspberry Pi, GAN, CNN.**

I. Introduction

Localization, the process of determining the locations of entities, devices, and other objects, has become an active research field in recent years. Much of the research focuses on using established technology to determine positions. Depending on the context in which positioning occurs, it can be categorized into two types: outdoor positioning and indoor positioning [1], [2]. While outdoor positioning using Global Navigation Satellite Systems (GNSS) technology, such as the Global Positioning System (GPS), is widely adopted due to the convenience of requiring only one receiver to obtain a position, it fails inside buildings due to signal obstruction by walls and other obstacles. As a result, Indoor Positioning Systems (IPS) have emerged as increasingly essential, particularly for locating people or objects where GPS and other satellite technologies lack precision or fail entirely, such as in hospitals, airports, and underground locations.

Compared to outdoor positioning which usually relies on GPS, indoor positioning doesn't have a one-size-fits-all method. This is mainly due to the unique and varied nature of indoor environments. Instead, any available wireless technique can be utilized to help determine a device's location indoors. Different technologies exist for indoor positioning, and many of them use existing wireless networks, which helps to avoid the need for extra equipment [1].

Various solutions have been proposed for indoor positioning systems. These include technologies that use Bluetooth, WiFi, and Ultra-WideBand (UWB) [3]. One of the most used technologies is WiFi, as it's already found nearly everywhere and can be set up quite easily.

Among different localization technologies, WiFi has gained substantial traction for indoor positioning due to the ubiquitous presence of WiFi-enabled devices and easy access to WiFi Access Points (APs). WiFi-based indoor positioning employs unique mathematical methods or positioning techniques to estimate the location of a device [3]. These techniques include proximity, trilateration, and WiFi fingerprinting [3], [4]. WiFi fingerprinting, in particular, has proven to be an effective and cost-efficient approach for indoor localization [1].

WiFi fingerprinting involves two phases: the offline phase and the online phase. During the offline phase, fingerprints representing Received Signal Strength Indicator (RSSI) or Channel State Information (CSI) measurements are collected at predefined Reference Points (RPs). These fingerprints are stored in a database, called a Radio Map, and are used to train a learning algorithm that maps each fingerprint to its location. In the online phase, the learned model predicts the position of a device based on its RSSI (or CSI) data. However, WiFi fingerprinting faces challenges due to signal fluctuations caused by the multi-path effect and physical obstacles.

Most of Previous research papers have utilized a private dataset [5], [6] and they have achieved a positioning error greater than 1.25 meters. As of the time of writing, there is no publicly available dataset that combines CSI amplitude and phase information with corresponding RSSI values, along with crucial data on collection positions. This comprehensive dataset is essential for training supervised Machine Learning (ML) models effectively. A challenging problem which arises in this domain is the small size of the radio map.

Our research aims at the creation of a CSI (i.e., with amplitude and phase information) and RSSI dataset, the implementation of data augmentation techniques to increase the amount of data and the application of Deep Learning (DL) algorithms for position estimation.

The rest of the paper is structured as follows: Section II presents the comprehensive model pipeline of our WiFi-based indoor positioning system. Section III

ISBN: 978-1-68558-106-0

details the data collection and the processing techniques for both CSI and RSSI data. Section IV describes the application of Tabular Generative Adversarial Network (TabGAN or GAN) for data augmentation and highlights the implementation of DL algorithms, specifically Convolutional Neural Network (CNN) and LightGBM, on the augmented datasets. A thorough comparative analysis of the obtained results is conducted in Section V to assess the performance of each algorithm. Section VI concludes the paper.

II. Methodology

The proposed model pipeline consists of two essential phases: the offline phase and the online phase, as depicted in Figure 1.

In the offline phase, the focus is on training and data augmentation using a deep learning model. The initial step involves collecting real-world data by physically walking around the indoor environment and recording the RSSI at various locations. This dataset serves as the foundation for training the positioning algorithm. To enhance the training dataset, a Tabular GAN is employed for data augmentation. The Tabular GAN utilizes a generator network to learn the underlying distribution of the real data and generate synthetic data points resembling the collected RSSI and CSI measurements. The generated synthetic data, combined with the real data, forms an expanded and more diverse training dataset.

Moving to the online phase, when a user is in motion within the indoor environment, the system follows a series of steps for real-time positioning. First, the access points or beacons emit signals that are detected by the user's device, which measures the RSSI/CSI values. These measured values are then utilized in conjunction with the trained positioning algorithm. The algorithm compares the received RSSI and CSI values with the augmented training dataset, allowing for the estimation of the user's current position in real-time. By leveraging the augmented dataset and the positioning algorithm, which are in our case CNN and LightGBM, accurate and reliable indoor positioning can be achieved.

The combination of the offline phase, featuring deep learning-based data augmentation using Tabular GAN, and the online phase for real-time positioning facilitates dynamic and accurate indoor localization. This pipeline holds potential for a wide range of applications, including indoor navigation, asset tracking, and location-based services, offering improved accuracy and robustness in indoor positioning systems.

III. Dataset Creation

In this section, we will discuss the process of collecting data for the indoor positioning system, including the definition of the Raspberry Pi setup, the definition of the floor plan, and pinning the reference points.

A. *Definition of the floor plan*

In this work, data collection took place in the ITZ building of the University of Passau, in Germany. This indoor environment, spanning an area of approximately 47 square meters, served as the designated area for data collection.

By focusing on a specific location within the university building, the data collection process aimed to capture the unique characteristics and signal propagation patterns present in this particular indoor setting. The selected area provided a controlled environment for gathering data and conducting experiments, ensuring consistency and reproducibility in the collected dataset.

Each RP served as a designated location for data collection with specific coordinates within the room. The positioning of these reference points followed a regular pattern, with a distance of 1 meter between adjacent points and 0.45 meters from the walls. This configuration ensured that the RPs covered the entire area of the room, capturing the signal variations and characteristics at different positions.

To gather comprehensive data and capture signal variations from different directions within each RP, measurements were collected systematically from four cardinal directions: North, South, West, and East. This approach allowed for a more thorough assessment of the signal strength and characteristics in each RP. At each RP, the data collection process involved moving around the point and measuring the signal strength from the four specified directions. By collecting measurements from multiple directions, the dataset encompassed a wider range of signal variations, taking into account potential obstacles, signal blockages, or signal reflections from different angles.

B. *Nexmon Firmware*

For WiFi chips, Nexmon is a framework for firmware modification that makes it possible to enable extra features and capabilities above and beyond what stock firmware generally supports. To explore new possibilities and create cutting-edge applications, it gives researchers and developers the freedom to access and control WiFi chips' low-level features [7].

In the context of collecting RSSI and CSI data for fingerprinting and indoor localization, Nexmon can be used to capture and analyze the wireless signals transmitted by WiFi devices. By modifying the firmware on compatible WiFi chips, Nexmon allows for the extraction of detailed information about the wireless channel, including RSSI and CSI values. Using this information, fingerprints that depict the distinctive qualities of the wireless signals at various points in an indoor area can be made. These radio maps can be used as the foundation for IPS that use fingerprinting to locate a target device based on the characteristics of the received signal.

C. *Data recording*

The data recording process for collecting CSI data for fingerprinting and indoor localization using Nexmon on Raspberry Pi 4 involved several steps. First, the setup and configuration included using Raspberry Pi 4 with Nexmon firmware. Nexmon was configured to capture CSI data on channel 36 with a 80 MHz bandwidth, specifically targeting the first core of the WiFi chip and the first spatial stream. Next, the measurement procedure was conducted at various positions

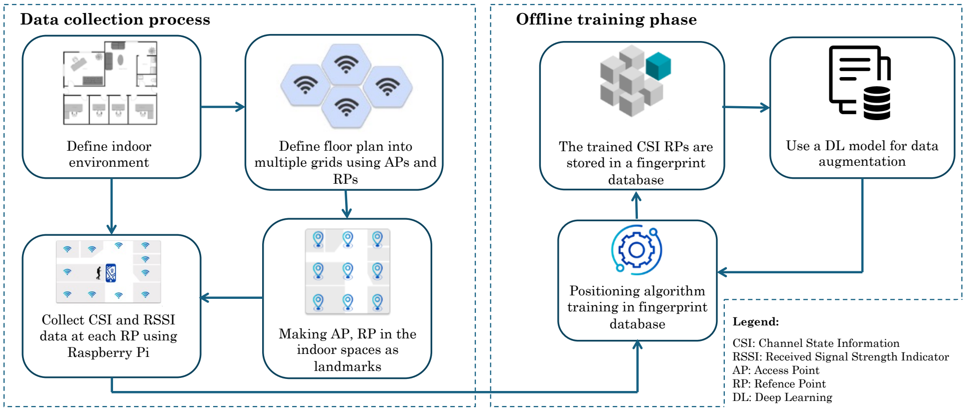

Data collection process
Define indoor environment
Define floor plan into multiple grids using APs and RPs
Making AP, RP in the indoor spaces as landmarks
Collect CSI and RSSI data at each RP using Raspberry Pi
Offline training phase
The trained CSI RPs are stored in a fingerprint database
Use a DL model for data augmentation
Positioning algorithm training in fingerprint database
Legend:
CSI: Channel State Information
RSSI: Received Signal Strength Indicator
AP: Access Point
RP: Refence Point
DL: Deep Learning

Fig. 1: Pipeline of the proposed model

within the target environment. For each position, 100 samples were collected to ensure accuracy and reliability. The Raspberry Pi 4 with Nexmon was carefully placed at each position, maintaining a stable and consistent setup throughout the data collection process. Measurements were taken in multiple directions (North, East, West, and South) to capture a comprehensive view of the wireless signals present. During the data collection phase, Nexmon on Raspberry Pi 4 listened on socket 5500 for User Datagram Protocol (UDP) packets, which contained the captured CSI data. The CSI data was extracted from these UDP packets, providing the necessary information for further analysis. The collected CSI data can be analyzed by opening the PCAP file in Wireshark or parsing it using a script.

1) Fingerprinting dataset: During the data collection phase, data capture was conducted within the ITZ building. The Raspberry Pi, fixed at a fixed height, was placed at various RPs to collect data in four directions. A total of 100 WiFi frames were captured for each reference point in each direction, resulting in 400 WiFi frames in total for all four directions. Since there were 45 reference points, the entire data collection process yielded approximately 18000 WiFi data frames.

The data collection process required two days to set up the RPs around the room and perform the measurements. During this time, two students collaborated to ensure the accurate positioning of the Raspberry Pi in each direction for every RP. The results of these measurements were stored in PCAP files for further analysis and processing.

In the dataset, we have a total of 28 unique MAC addresses. However, it is worth noting that five of these MAC addresses contribute significantly to the data, making up approximately 16,000 rows out of the total 18,000 rows in the dataset. These top five MAC addresses are responsible for a significant portion of the data and play a crucial role in the analysis. We have assigned a unique number to each MAC address, ranging from 1 to 28. This numbering scheme was implemented to facilitate the representation of MAC addresses in the bar chart.

D. Data preprocessing

To extract the CSI data from the PCAP files, the following preprocessing steps were performed using the provided code:

1) *Reading the PCAP file*: This step involved reading the PCAP file containing the captured wireless packets. This step extracted the necessary data from the PCAP file.
2) *Infer Bandwidth*: The bandwidth of the wireless signal was inferred from the length of the packets in the PCAP file. This ensured that the correct bandwidth was used for further processing.
3) *Determine Number of Subcarriers*: The number of Orthogonal Frequency-Division Multiplexing (OFDM) subcarriers was determined based on the inferred bandwidth. This value is required for correctly interpreting the CSI data.
4) *Estimate Maximum Number of Samples*: An estimate for the maximum possible number of samples in the PCAP file was calculated. This estimation is useful for allocating memory for storing the extracted data.
5) *Data Extraction*: The actual extraction of CSI data was performed by iterating over the packets in the PCAP file. For each packet, the relevant information such as RSSI, MAC ID, sequence number, core and spatial stream, and CSI data were extracted and stored.
6) *Conversion to Numpy Arrays*: The extracted CSI data and other relevant information were then converted to NumPy arrays for efficient processing and analysis. This conversion facilitated further manipulation and analysis of the data.
7) *SampleSet Object Creation*: Finally, all the extracted data were encapsulated in a data structure for easy access and analysis. This data structure provides convenient methods to retrieve specific information for a given sample index.

IV. Data Augmentation

In this section, we delve into the key aspect of our research, which is data augmentation using the Tabular GAN model.

A. Tabular Generative Adversarial Network (TabGAN) for Data Augmentation

Tabular GAN is a generative model specifically designed for augmenting tabular data. It leverages the power of GANs to generate synthetic data that closely resembles the original dataset, thereby increasing its size and diversity [8]. The architecture of the Tabular GAN comprises the following components: generator and discriminator.

1) Generator: is responsible for generating synthetic data that captures the distribution and characteristics of the original dataset. Its architecture consists of the following steps:

i. Generating Numerical Variables:
   - Scalar Value Generation: The Generator generates scalar values, such as those representing clusters, using techniques like sampling from a Gaussian Mixture Model (GMM).
   - Cluster Vector Generation: Cluster vectors are generated to represent the probability of each data point belonging to different clusters. These vectors can be obtained from GMM outputs or other methods.
   - Activation Function: Numerical variables are transformed using an activation function, such as the hyperbolic tangent (tanh), to ensure the generated values fall within a desired range.

ii. Generating Categorical Variables:
   - Probability Distribution: Categorical variables, such as labels or classes, are generated as probability distributions over all possible categories. Techniques like softmax activation function are used for this purpose.

iii. Long Short-Term Memory (LSTM) networks:
   - To generate rows effectively, an LSTM network with an attention mechanism is employed. This architecture enables the model to capture temporal dependencies and generate coherent synthetic samples.
   - Inputs to the LSTM include random variables, weighted context vectors, previous hidden states, and embedding vectors.

2) Discriminator: plays a vital role in distinguishing between real and synthetic data. It aims to learn the underlying patterns and characteristics of the original dataset. The Discriminator architecture consists of the following components:

i. Multi-Layer Perceptron (MLP):
   - The Discriminator utilizes an MLP with activation functions like LeakyReLU and techniques such as Batch Normalization. This MLP is responsible for extracting features and discriminating between real and synthetic data.
   - Concatenation: The numerical variables, cluster vectors, and binary variables obtained from the Generator are concatenated to form the input for the Discriminator.

ii. Loss Function:
   - The Discriminator's loss function incorporates components like the Kullback-Leibler (KL) divergence term and the sum ordinal log loss. These components allow the Discriminator to optimize its ability to differentiate between real and synthetic data effectively.

B. Evaluation of TabGAN

To assess the effectiveness of the augmented data generated by the Tabular GAN on the prediction of the position coordinates X and Y, we evaluate the performance using two different models: CNN and LightGBM.

We start by evaluating the performance of the CNN model on the following datasets:

1) *Ground Truth Data*: We assess the performance of CNN model on the original (ground truth) data. This serves as a baseline to compare against the performance on the augmented data. We calculate the Root Mean Square Error (RMSE) between the true values of the X and Y coordinates and the corresponding predictions made by the CNN model.
2) *Augmented Data*: Next, we evaluate the performance of CNN model when trained on the augmented data generated by the Tabular GAN. We use the same evaluation metric to compare the predictions made on the augmented data with the ground truth values. This step allows us to determine how well the Tabular GAN has captured the underlying distribution of the original data and whether the augmented data is useful for improving the prediction accuracy.
3) *Combined Data*: Finally, we assess the performance of CNN model when trained on a combination of the ground truth data and the augmented data. This step aims to investigate the potential benefits of incorporating the augmented data into the training process. We compute the RMSE between the predictions made on the combined dataset and the true values of the X and Y coordinates.

The evaluation process of LightGBM model is similar to the steps described above for CNN model.

C. CNN Architecture

In Figure 2, we present the detailed architecture of our CNN model. The CNN is designed to handle the positioning estimation task efficiently by processing the input data, extracting relevant features, and making accurate predictions.

The architecture comprises several essential components, including convolutional layers, pooling layers, and fully connected layers. These layers work collaboratively to learn hierarchical representations from the input data, enabling the model to capture intricate spatial patterns and relationships present in the RSSI and CSI measurements.

The initial convolutional layers act as feature extractors, convolving over the input data to detect spatial patterns and edges. The pooling layers then downsample the extracted features, reducing the computational complexity and aiding in learning spatial invariance.

Subsequently, the flattened feature maps are passed through fully connected layers, which serve as the decision-making units of the model. These layers combine the learned features

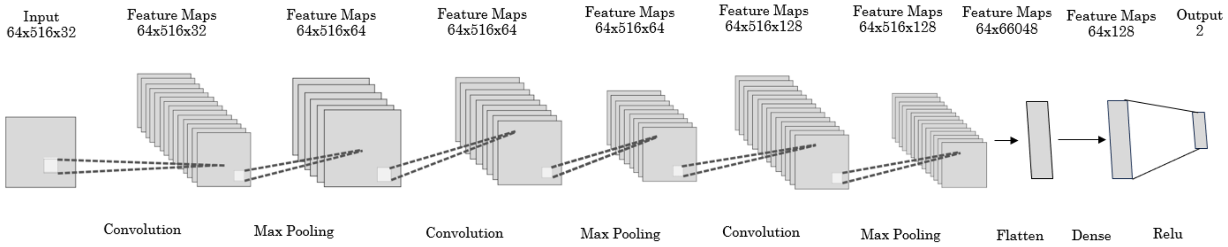


Fig. 2: CNN architecture

and apply non-linear transformations to make accurate positioning predictions.

Additionally, we have employed regularization techniques, such as dropout and batch normalization, to prevent overfitting and enhance model generalization. The CNN model is trained using an appropriate optimization algorithm, whis is Adam in our case, and a suitable loss function for regression tasks, such as RMSE.

The CNN architecture is designed to capture relevant spatial information and learn meaningful representations from the input data. By stacking convolutional and pooling layers, the model can hierarchically extract features and make predictions based on the learned patterns.

V. Results and Discussions

In this section, we present the evaluation and discussion of the system based on the Tabular GAN for data augmentation.

A. *Data augmentation results*

In order to validate the effectiveness of the data augmentation process using GANs, we conducted a comprehensive comparison of the data distribution. Specifically, we randomly selected a column from the original dataset and generated synthetic data using the GAN.

We noticed a striking resemblance between the data distribution of the original dataset and the data distribution of the generated synthetic data, demonstrates the GAN's ability to accurately capture and reproduce the underlying characteristics of the original data. This successful alignment further reinforces the efficacy of the data augmentation technique in preserving data distribution, making the generated data a valuable and reliable resource for enhancing our positioning estimation algorithm.

Through this comparison, we can confidently assert that the data augmentation process using GANs has effectively maintained the integrity of the original data's distribution. Such congruence is essential in ensuring that the generated data contributes meaningfully to the generalization and robustness of our positioning estimation model.

B. *LightGBM for GAN evaluation*

LightGBM, renowned for its efficiency and scalability, is particularly well-suited for tabular data analysis, making it an ideal choice for our positioning estimation problem [9].

First thing, our data was split as follows : 80% for train and 20% for test. After training LightGBM on the augmented dataset, which included the GAN-generated samples, we conducted an extensive evaluation using various regression-specific metrics, such as RMSE and Mean Positioning Error (MPE). RMSE allowed us to measure the average deviation between the predicted positioning coordinates and the ground truth values, providing a comprehensive assessment of the model's accuracy in estimating positions.

In our evaluation, we compared the performance of the LightGBM model on three different types of data: RSSI only, CSI only, and RSSI combined with CSI. We have used both CSI amplitude and phase. The results (see Table I) revealed intriguing insights into the significance of each data type for position estimation.

When training the model on the RSSI only data, we obtained an RMSE of 0.99 for X-coordinate, 2.6 for Y-coordinate and a MPE of 1.58 meters. Incorporating the GAN-generated samples through data augmentation improved the performance to an RMSE of 0.95 for X-coordinate and a MPE of 1.5 meters. Subsequently, when combining both RSSI and CSI data, the RMSE reduced to 0.914 for X-coordinate, 2.433 for Y-coordinate and the MPE to 1.5 meters.

C. *CNN for GAN evaluation*

Based on our feature importance analysis using LightGBM, we found that the combined RSSI and CSI data resulted in a slightly improved performance compared to using CSI data only. This observation highlights the importance of leveraging both RSSI and CSI features for accurate position estimation.

Given the insights from the feature importance analysis, we proceeded with the evaluation of the CNN model on the ground truth data and the augmented data. The CNN model was trained on the ground truth data to establish a baseline performance and assess its inherent capabilities in positioning estimation. Subsequently, we trained the CNN model on the augmented dataset, which included the GAN-generated samples, to evaluate the performance improvements brought about by GAN-based data augmentation.

In the process of training the model solely on the RSSI data, the resulting RMSE values were 0.76 for the X-coordinate and 2.01 for the Y-coordinate, which led to a MPE of 1.45 meters. Upon incorporating GAN-generated samples through data augmentation, the model's performance improved, resulting in an

TABLE I: LightGBM Model Evaluation

	Ground Truth Data		**Augmented Data**		**Combined Data**	
	RMSE (X/Y)	**MPE**	**RMSE (X/Y)**	**MPE**	**RMSE (X/Y)**	**MPE**
RSSI Only	0.99/2.6	1.58 m	0.95/1.555	1.5 m	0.914/2.433	1.5 m
CSI Only	0.795/1.289	1.242 m	0.6888/1.177	1.082 m	0.6487/1.0417	0.99 m
RSSI + CSI	0.796/1.289	1.242 m	0.6888/1.177	1.08 2m	0.6487/1.042	0.99 m

Legend: RMSE - Root Mean Square Error, MPE - Mean Positioning Error, m - meter

TABLE II: CNN Model Evaluation

	Ground Truth Data		**Augmented Data**		**Combined Data**	
	RMSE (X/Y)	**MPE**	**RMSE (X/Y)**	**MPE**	**RMSE (X/Y)**	**MPE**
RSSI Only	0.76/2.01	1.45 m	0.73/1.325	1.27 m	0.69/1,3	1.224 m
CSI Only	0.555/1.102	1.052 m	0.531/0.992	1.082 m	0.504/0.952	0.99 m
RSSI + CSI	0.554/1.1	1.04 m	0.529/0.971	0.97 m	0.507/0.93	0.8 m

Legend: RMSE - Root Mean Square Error, MPE - Mean Positioning Error, m - meter

RMSE of 0.73 for the X-coordinate and a MPE of 1.27 meters. Subsequently, when both RSSI and CSI data were combined, the RMSE reduced to 0.69 for the X-coordinate and 1.3 for the Y-coordinate, with a MPE of 1.224 meters. As seen in Table II, it is evident that the combination of both RSSI and CSI data yielded the most precise localization performance with a MPE of 0.8 meters.

The remarkable reduction in MPE for the augmented data further reinforces the superiority of GAN-based data augmentation in enhancing the model's performance. The significantly lower MPE demonstrates that the CNN, when trained on the augmented data, is better able to estimate the target positions with higher accuracy and precision, making it a compelling choice for positioning estimation tasks in real-world scenarios. The improved learning and generalization capabilities achieved with the augmented data reinforce the efficacy of GAN-based data augmentation in enhancing the CNN's performance for positioning estimation.

VI. Conclusions

In this paper, we embarked on a comprehensive investigation of an indoor localization system, leveraging WiFi data for precise positioning estimation. The data collection process was meticulously executed within a controlled lab environment, utilizing Raspberry Pi and Nexmon firmware to capture WiFi signals. Subsequently, we performed thorough data preprocessing to ensure data quality and consistency.

A key highlight of this work was the application of data augmentation using GAN to enrich the original dataset. The GAN-based augmentation technique effectively generated synthetic data points, enhancing the diversity and volume of the training data, which proved instrumental in improving the accuracy and generalization of our positioning algorithms.

Our evaluation process involved the utilization of two prominent DL algorithms: CNN and LightGBM. The results highlighted the remarkable improvements achieved when training on augmented data, demonstrating the efficacy of GAN-based data augmentation in boosting the precision of the positioning estimation. Our system showed the improvement of at least **25 cm** in positioning error when using GAN. Nonetheless, CSI combined with RSSI has shown a better influence with a positioning error equal to **0.8 meters**.

Future work in this domain could focus on expanding the evaluation to other machine learning algorithms and exploring different GAN architectures for data augmentation.

References

[1] W. F. Hassen and J. Mezghani, "Cnn based approach for indoor positioning services using rssi fingerprinting technique," in *2022 International Wireless Communications and Mobile Computing (IWCMC)*. IEEE, 2022, pp. 778–783.

[2] W. F. Hassen and F. Najjar, "A positioning handoff decision algorithm for ubiquitous pedestrian navigation systems," pp. 487–494, 2016.

[3] W. F. Hassen, L. Brunie, A. Nechba, and H. Kosch, "Continuous indoor/outdoor pathway display algorithm for pedestrian navigation service," in *Proceedings of the 17th International Conference on Advances in Mobile Computing & Multimedia*, 2019, pp. 66–73.

[4] W. F. Hassen, F. Najjar, L. Brunie, H. Kosch, and Y. Slimani, "Smart pdr integration for ubiquitous pedestrian navigation service," in *2017 13th International Wireless Communications and Mobile Computing Conference (IWCMC)*. IEEE, 2017, pp. 1558–1563.

[5] P. Roy and C. Chowdhury, "A survey of machine learning techniques for indoor localization and navigation systems," *Journal of Intelligent & Robotic Systems*, vol. 101, p. 63, 2021.

[6] A. Poulose and D. S. Han, "Hybrid deep learning model based indoor positioning using wi-fi rssi heat maps for autonomous applications," *Electronics*, vol. 10, no. 1, 2021. [Online]. Available: https://www.mdpi.com/2079-9292/10/1/2

[7] Seemoo Lab, "NEXMON: The C-based Firmware Patching Framework," https://github.com/seemoo-lab/nexmon, 2023.

[8] P. F. Moshiri, H. Navidan, R. Shahbazian, S. A. Ghorashi, and D. Windridge, "Using gan to enhance the accuracy of indoor human activity recognition," *arXiv preprint arXiv:2004.11228*, 2020.

[9] L. Yin, P. Ma, and Z. Deng, "Jlgbmloc—a novel high-precision indoor localization method based on lightgbm," *Sensors*, vol. 21, no. 8, p. 2722, 2021.