DOCTORAL THESIS NO. 42

Collaborative Predictive Maintenance for Smart Manufacturing

From Wireless Control to Federated Learning

Ali Bemani

Gävle University Press

Dissertation for the Degree of Doctor of Philosophy in Electrical Engineering, to be publicly defended on Thursday, 4th April 2024 at 13:00 in 12:108, University of Gävle.

External reviewer: Janet Lin, Associate Professor at Luleå University of Technology and Guest Professor at Mälardalen University

Cover illustration: Ali Bemani

Gävle University Press
ISBN 978-91-89593-23-7
ISBN 978-91-89593-24-4 (pdf)
urn:nbn:se:hig:diva-43564

Distribution:
University of Gävle
Faculty of Engineering and Sustainable Development
Department of Electrical Engineering, Mathematics and Science
SE-801 76 Gävle, Sweden
+46 26 64 85 00
www.hig.se

Abstract

Industry 4.0 represents a significant shift in the industrial landscape, aimed at improving efficiency, productivity, and competitiveness. This shift involves the digitalization of industries, impacting manufacturing and maintenance processes. A pivotal element of this transformation is the development of Cyber-Physical Systems (CPS) that seamlessly connect the physical factory floor with the digital realm. These systems monitor real-time data from the physical world and prepare feedback from the digital space, necessitating the harmonious integration of computation and communication, especially through wireless technology. Simultaneously, Machine Learning (ML) methods are advancing across various domains. The proliferation of wireless sensors and the Internet of Things, particularly within the CPS framework, generates substantial data. To address challenges such as latency, device resource limitations, and privacy concerns associated with centralized cloud processing, there is a shift towards edge computing, enabling distributed learning algorithms.

This dissertation tackles these challenges with four innovative methods that combine wireless technology, control systems, and distributed ML in the context of Industry 4.0. These methods aim to harness the potential of this digital transformation, making Predictive Maintenance (PdM) in industries smarter and more efficient. The first method, parallel event-triggering, is designed for multi-agent systems in industrial environments. It utilizes distributed event-based state estimation to enhance control performance and reduce network resource consumption. The second and third methods are developed for collaborative PdM using wireless communication in a federated approach. The second method focuses on real-time anomaly detection while preserving asset privacy at the edge level, and the third method optimizes remaining useful life prediction from sequential data within a federated learning framework. Both federated approaches enhance efficiency, simplify communication, and improve local model convergence. The fourth method introduces an innovative approach to collaborative PdM, utilizing over-the-air computation at the edge level. This approach offers low latency and improved spectral efficiency. The optimization challenges at the edge level are addressed by using a modified gradient descent approach, which effectively handles noisy communication channels and improves the convergence of ML algorithms. All four methods proposed in this thesis underwent a comprehensive evaluation, and the experimental findings demonstrate their effectiveness in achieving their intended objectives.

Sammanfattning

Industri 4.0 representerar en betydande förändring i det industriella landskapet i syfte att förbättra effektivitet, produktivitet och konkurrenskraft. Denna förändring innebär en digital transformation av industrier, vilket påverkar tillverknings- och underhållsprocesser. En central del av denna transformation är utvecklingen av Cyber-fysiska system (CFS) som sömlöst kopplar samman det fysiska fabriksgolvet med den digitala världen. Dessa system övervakar realtidsdata från den fysiska världen och analyser från den digitala sfären, vilket kräver harmonisk integration av beräkning och kommunikation, särskilt genom trådlös teknik. Samtidigt utvecklas maskininlärningsmetoder inom olika områden. Spridningen av trådlösa sensorer och Internet of Things, särskilt inom CFS-ramverket, genererar betydande data. För att hantera utmaningar som latens, begränsade resurser hos lokala enheter och integritetsbekymmer kopplade till centraliserade molntjänster sker en övergång till beräkningar på nätverkets kant, även kallat edge computing, vilket möjliggör distribuerade inlärningsalgoritmer.

Denna avhandling tacklar dessa utmaningar med fyra innovativa metoder som kombinerar trådlös teknik, styrsystem och distribuerad maskininlärning inom ramen för Industri 4.0. Dessa metoder syftar till att utnyttja potentialen i denna digitala transformation och göra prediktivt underhåll i industrier smartare och effektivare. Den första metoden, parallell händelseutlösning, är utformad för fleragentssystem i industriella miljöer. Den använder distribuerad händelsebaserad tillståndsestimering för att förbättra reglerprestanda och minska förbrukning av nätverksresurser. De andra och tredje metoderna är utvecklade för att prediktivt underhåll och trådlös kommunikation skall samarbeta med ett federativt tillvägagångssätt. Den andra metoden fokuserar på avvikelsedetektering i realtids samtidigt som dataintegriteten bevaras på edge-nivå, och den tredje metoden optimerar förutsägelse av kvarvarande användbar livslängd från sekventiella data inom en federativ inlärningsram. Båda federativa tillvägagångssätten förbättrar effektiviteten, förenklar kommunikationen och förbättrar lokalt konvergensen av modeller. Den fjärde metoden introducerar ett innovativt tillvägagångssätt för samarbetsbaserat prediktivt underhåll, där over-the-air beräkningar används på edge-nivån. Denna metod erbjuder låg latens och effektivare utnyttjande av frekvensband. De utmaningar som finns på edge-nivån hanteras genom att använda en modifierad gradient descent metod, som effektivt hanterar brusiga kommunikationskanaler och förbättrar konvergensen av ML-algoritmer. Alla fyra metoder som föreslås i denna avhandling genomgick en omfattande utvärdering, och de experimentella resultaten visar deras effektivitet för att uppnå sina avsedda mål.

Acknowledgments

I would like to express my deepest gratitude to Professor Niclas Björsell, my esteemed supervisor, for his unwavering belief in my capabilities and his invaluable support in granting me the autonomy to pursue my research ideas. Working under Professor Björsell's guidance has been an exceptionally fortunate and enriching experience. His patience, encouragement, enthusiasm, and constructive criticism created an environment where I was able to think freely and tackle challenges with confidence. Professor Björsell is not only an exceptionally generous collaborator but also a true friend and insightful mentor who has significantly influenced my perspective, not only in technical discussions but in my overall approach to the world. I would also like to thank my co-supervisor, Professor Daniel Rönnow, for his invaluable guidance, constructive feedback, and meaningful discussions on both research and teaching.

I would like to express my gratitude to Dr. Mikael Cronhjort, head of the Department of Electrical Engineering, Mathematics, and Natural Sciences. Additionally, I extend my appreciation to Professor Magnus Isaksson, who previously held this position, for their continuous support and invaluable advice over the years.

I would like to express my heartfelt gratitude to all my colleagues at the University of Gävle, particularly those at the Department of Electrical Engineering, for fostering a positive, friendly, and supportive atmosphere throughout my academic journey. Special thanks to Dr. Per Ängskog, Dr. Håkan Hugosson, Dr. Jose Chilo, and Dr. Smruti Panigrahi for their guidance. I am equally thankful to my friends Dr. Hadi Amin, and Dr. Arvid Jahedi, and fellow PhD students, including Amirhossein Hosseinzadeh, Vipin Choudhary, Oscar Bautista Gonzalez, Muhammad Hassan, Arash Jouybari, and all other PhD students from the Faculty of Engineering, for engaging seminars, insightful discussions, and meaningful conversations that enriched my experience.

I am profoundly grateful to my beloved wife, Niloofar, for her unwavering support and understanding, which have been my strength throughout this academic journey. Her love is my greatest inspiration. As we eagerly await the birth of our daughter, I dedicate this thesis to Niloofar and our beloved daughter, with boundless love and deep appreciation for the joy and strength they bring to my life.

List of Papers

This thesis is based on the following papers, which are referred to in the text by Roman numerals.

Paper I

Bemani, A., Björsell, N. (2020, June). Cyber-Physical Control of Indoor Multi-vehicle Testbed for Cooperative Driving. In 2020 *IEEE Conference on Industrial Cyberphysical Systems (ICPS)* (Vol. 1, pp. 371-377). IEEE. https://doi.org/10.1109/ICPS48405.2020.9274688

Paper II

Bemani, A., Björsell, N. (2021). Distributed event triggering algorithm for multi-agent system over a packet dropping network. *Sensors*, 21(14), 4835. https://doi.org/10.3390/s21144835

Paper III

Bemani, A., Björsell, N. (2021, June). Distributed Event-Triggered Control of Vehicular Networked System with Bursty Packet Drops. In *2021 7th International Conference on Event-Based Control, Communication, and Signal Processing (EBCCSP)* (pp. 1-7). IEEE.
https://doi.org/10.1109/EBCCSP53293.2021.9502365

Paper IV

Bemani, A., Björsell, N. (2022). Aggregation strategy on federated machine learning algorithm for collaborative predictive maintenance. *Sensors*, 22(16), 6252. https://doi.org/10.3390/s22166252

Paper V

Bemani, A., Björsell, N. (2023). Low-Latency Collaborative Predictive Maintenance: Over-the-Air Federated Learning in Noisy Industrial Environments. *Sensors*, 23(18), 7840.
https://doi.org/10.3390/s23187840

Author's Contribution to the Papers

In the following, the author's contribution to the appended papers is summarized.

Paper I

- Developed the research project and conceptualized the study with co-authors,
- developed the theory and performed the computation and programming,
- carried out the hardware and software implementation,
- analyzed and discussed the results jointly with the co-author,
- took the lead in writing the manuscript, and
- discussed the results and commented on the manuscript.

Paper II

- Devised the main conceptual idea and the outline for the evidence,
- performed all the numerical experiments and simulations, and preparing all the results,
- analyzed and discussed the results jointly with the co-author,
- took the lead in writing the manuscript, and
- based on the comments of the co-authors, finalized the paper.

Paper III

- Conceptualized the study with the feedback from co-author,
- performed all the numerical experiments, simulations, and the results,
- analyzed and discussed the results jointly with the co-author,
- took the lead in writing the manuscript, and
- based on the comments of the co-author, finalized the paper.

Paper IV

- Designed the main idea and conceptualized the study,
- performed the analytic calculations and did programming for numerical experiments,
- analyzed and discussed the results jointly with the co-author,
- took the lead in writing the manuscript, and
- based on the comments of the co-author, finalized the paper.

Paper V

- Initiated and conceptualized the idea of the study,
- performed the analytic calculations and did programming for numerical experiments,
- analyzed and discussed the results jointly with the co-author,
- took the lead in writing the manuscript, and
- based on the comments of the co-author, finalized the paper.

Nomenclature

Abbreviation	
AWGN	Additive White Gaussian Noise
CMAPSS	Commercial Modular Aero-Propulsion System Simulation
CPS	Cyber-Physical Systems
DEBSE	Distributed Event-Based State Estimation
EBSE	Event-Based State Estimation
EKF	Extended Kalman Filter
ET	Event-Triggering
ETC	Event-Triggered Control
ETSE	Event-Triggered State Estimation
FedAvg	Federated Averaging
FL	Federated Learning
FSVRG	Federated Stochastic Variance Reduced Gradient
FSVRG-OACC	Federated Stochastic Variance Reduced Gradient Over-the-Air Communication and Computation
GD	Gradient Descent
KF	Kalman Filter
LSTM	Long Short-Term Memory
ML	Machine Learning
MPC	Model Predictive Control
non-iid	Non-Independent and Identically Distributed
OACC	Over-the-Air Analog Computation and Communication
OFDM	Orthogonal Frequency Division Multiplexing
PdM	Predictive Maintenance
PET	Parallel Event-Triggering
RUL	Remaining Useful Life
SGD	Stochastic Gradient Descent
SNR	Signal-to-Noise Ratio
SVM	Support Vector Machine
SVRG	Stochastic Variance Reduced Gradient
TTC	Time-Triggered Control
WNCS	Wireless Networked Control Systems

Table of Contents

1 Introduction

In the rapidly evolving landscape of today's industrial sector, the convergence of digital technology and manufacturing processes has ushered in a transformative era known as Industry 4.0. This paradigm shift, driven by the relentless pursuit of efficiency, productivity, and competitiveness, has given rise to the digitalization of industries across the globe. Industry digitalization encompasses a multifaceted approach that affects not only manufacturing but also maintenance, ultimately redefining how industrial businesses operate in the modern world. As part of these efforts, digital models of physical components and machines are developed, commonly known as Cyber-Physical Systems (CPS) (Stark et al., 2017). These systems monitor and synchronize information from various aspects, bridging the gap between the physical factory floor and the digital computational space. In general, a CPS encompasses the real-time acquisition of data from the physical world and the feedback of information from cyberspace. This necessitates the seamless integration of computation and communication in the digital realm with actions in the physical world. In accordance with the CPS framework, wireless communication will be employed, particularly impacting the real-time dimension. The incorporation of this dimension into the system model leads to the designation of Wireless Networked Control Systems (WNCS) (Park et al., 2017). In parallel with the recent progress in wireless communications, contemporary Machine Learning (ML) methods have led into remarkable advancements across various domains of science and technology. The growing number of wireless sensors and Internet of Things (IoT), especially in the CPS paradigm, is generating vast amounts of data. Current centralized approaches to process this data in the cloud face challenges like latency, limited device resources, and privacy concerns. To address these issues, there's a shift towards bringing intelligence to the network edge, where devices can collaborate and implement distributed learning algorithms (Eldar et al., 2022).

In this chapter, the exploration of contextual information and illustrative application examples takes place. Additionally, emphasis is given to the relevance of CPS and WNCS to these examples. Unlocking the full capabilities of future CPS and WNCS requires addressing various challenges, which will be explored in Section 1.2. Subsequently, an examination of related works that have attempted to address these problems is conducted. The research objectives, contributions, and methodology will be discussed in the following sections: Section 1.4, Section 1.5, and Section 1.6, respectively. Finally in Section 1.7, we provide an overview of the thesis outline.

1.1 Background

Digitalization holds a central position within the industrial strategies of various nations, notably exemplified by Germany's Industry 4.0 initiative. While the early achievements within these initiatives are promising, it is important to note that many successful projects are conducted within recently established manufacturing facilities, a context that deviates from the prevailing conditions in the broader industrial landscape. Established industrial facilities often encompass a blend of both contemporary and legacy technologies, thereby introducing novel complexities and challenges in the context of digitalization.

Recent advancements in the manufacturing industry have paved the way for the systematic deployment of CPS. These systems facilitate the close monitoring and synchronization of information from various perspectives, bridging the gap between the physical factory floor and the digital computational domain (Lee et al., 2015). Furthermore, through the application of advanced information analytics, interconnected machines can operate with enhanced efficiency, collaboration, and resilience. In essence, a CPS comprises two principal functional components: (i) advanced connectivity, which ensures real-time data acquisition from the physical environment and information exchange with the digital realm, and (ii) intelligent data management, analytics, and computational capabilities, which collectively form the foundation of the digital space. Hence, in the future, CPS are expected to independently function in the physical world, using embedded computers and networks for both computation and communication (Ma et al., 2018). Herein, we highlight two instances of CPS that are anticipated to exert significant influence.

Autonomous vehicle. Today, the futuristic concepts of autonomous vehicles, such as self-driving capabilities, automated parking, and intelligent obstacle detection and avoidance, have materialized into reality. Autonomous driving, when integrated with WNCS and CPS, represents a paradigm shift in transportation. It leverages the power of connectivity, real-time data exchange, and intelligent decision-making to create a safer and more efficient transportation ecosystem. As depicted in Figure 1a, the realization of the full potential of autonomous driving hinges solely on vehicles' ability to share information with each other. This convergence paves the way for more efficient and intelligent mobility solutions with the potential to reduce accidents, reduce fuel consumption, alleviate traffic congestion, and usher in new forms of transportation services (Elefteriadou, 2020). However, it also raises challenges related to network reliability and cybersecurity that must be addressed to realize its full potential.

Smart Manufacturing. It is a prominent application area for CPS, which

(a) Autonomous vehicles [MQTT, Inc]. (b) Smart manufacturing [ScaleOut Software, Inc].

Figure 1. Examples of CPS

promises a transformative change in industrial production. Unlike traditional manufacturing setups where machines operate in isolation, often supervised by human operators, smart manufacturing represents a vision where machines seamlessly communicate with each other, enabling production optimization based on dynamic demand. This shift from mass production to highly customized manufacturing aligns with the growing demand for customized products. As smart manufacturing continues to gain traction, it is likely to be early adopters of CPS technologies that will change the future of manufacturing by increasing efficiency, agility, and product customization while presenting new challenges in automation, data management, and cybersecurity (Rawat et al., 2017).

Smart manufacturing, in conjunction with predictive maintenance (PdM), represents a revolutionary approach to industrial operations. This synergy leverages advanced technologies, such as IoT sensors, machine learning, WNCS, and AI, to monitor the real-time performance of machinery and equipment in manufacturing plants. Predictive maintenance systems can anticipate potential equipment failures by analyzing data streams from these sensors, enabling timely repairs and preventive measures. This proactive approach not only reduces downtime and maintenance costs but also extends the lifespan of critical assets, ensuring uninterrupted production and ultimately improving overall efficiency and competitiveness in the manufacturing sector (Chien and Chen, 2020).

1.2 Problems Statement

The preceding discussion highlights essential application areas of wireless CPS, but existing technology faces significant challenges in realizing their full potential. The fundamental requirement for all CPS applications is the acquisition of accurate and reliable data from machines, requiring wireless communication between devices. However, wireless communication introduces inherent challenges, including transmission delays, unreliable transmission,

and limited bandwidth. Wireless systems, often designed to operate under additive white Gaussian noise (AWGN), face distinct difficulties when exposed to impulsive interference, resulting in different communication effects. In the context of CPS, real-time data acquisition from the physical world is paramount, and this thesis focuses on the impact of wireless communication on real-time aspects, particularly in applications involving devices with complex dynamics and challenging tasks.

One approach to address the complexity of both device dynamics and task requirements is through AI and learning-based techniques, which can facilitate machine analysis and intelligence for various applications. However, the conventional approach of centralizing data processing on cloud servers encounters hurdles such as communication latency, device resource constraints, privacy issues, and low information density in collected data. To address these challenges, this thesis investigates a promising solution: the decentralization of intelligence to the network edge, with a particular focus on its application to PdM within smart manufacturing. This approach empowers wireless devices to implement distributed and collaborative learning algorithms, enhancing data processing efficiency and effectively addressing the specific challenges posed by time-sensitive applications such as PdM. In the following, we will provide a more detailed and tangible description of the specific problems that have been tackled within this thesis.

1.2.1 Control Over Wireless Network

WNCS are a subset of networked control systems in which sensors, controllers, and actuators exchange information through a wireless digital communication network. In wireless CPS, the primary focus is often on ensuring the safety of critical devices, such as autonomous vehicles, where maintaining system reliability is of utmost importance. For instance, the ability to transmit sudden braking information to following vehicles can help prevent accidents. To address this, leveraging wireless communication is a potential solution. However, it is important to note that wireless channels inherently carry the risk of transmission delays, packet loss, and network congestion, which can significantly affect the real-time performance of critical systems (Park et al., 2017). Traditional control theory assumes flawless communication to guarantee system stability, but this assumption does not hold in the realm of wireless CPS. Ensuring system stability in wireless CPS poses a challenge due to imperfect communication. This challenge becomes even more complex when we encounter correlated message loss, especially in cyberattack situations.

In this section, our primary aim is to create a resilient and flexible testing

environment that can accurately emulate real-world scenarios. This is crucial as it provides the opportunity to gather valuable insights into the impact of wireless communication on real-time processes. These insights will serve as the foundation for a deeper comprehension of these dynamics, facilitating the development of strategies aimed at enhancing the effectiveness of time-sensitive applications in wireless environments. Therefore, **the first problem statement** addressed in this thesis pertains to understanding the influence of wireless communication on real-time aspects and implementing stable control methods over wireless networks.

In the future wireless CPS, we envision a landscape where numerous agents are interconnected through a common network. Recent years have witnessed significant research attention directed toward addressing consensus challenges in the context of controlling multi-agent systems within WNCSs (Xu et al., 2021; Guinaldo et al., 2014; Li et al., 2022). This surge in interest is driven by the proliferation of diverse systems across engineering and scientific domains, including drone swarms, autonomous vehicles, and hierarchical production processes such as steel manufacturing and building automation, among others. The communication networks embedded within these systems assume a pivotal role in effectively coordinating the agents, thereby facilitating the attainment of shared objectives or enhancing overall system performance.

The core challenge in multi-agent systems lies in structuring communication among the agents to ensure their eventual convergence to a common state, tracking a synchronized trajectory, or accomplishing collaborative tasks. Our primary focus centers on multi-agent systems characterized by interdependencies among the agents, necessitating the periodic update of their states during various processes. However, owing to the nature of wireless network communication and the shared bandwidth among nodes, it becomes imperative to judiciously utilize communication resources. Consequently, each agent must employ these resources sparingly, reserving communication only when it is essential. Moreover, wireless communication introduces variables such as packet loss, communication latency, and packet disorder due to environmental factors, all of which demand consideration during controller design.

Hence, **the second problem statement** revolves around the development of a control algorithm capable of accommodating communication imperfections where periodic information exchange is not achievable among all agents.

1.2.2 Collaborative PdM for Smart Manufacturing

The challenges mentioned earlier become even more intricate when dealing with high-dimensional systems, which are frequently encountered in fields such as smart manufacturing and PdM, the primary focus of this thesis. Given the dynamic nature of many components within these systems, there is a critical need to automatically construct dynamic models for these components using sensor data. On the other hand, machine learning (ML) has made significant strides in recent years, providing valuable advancements that can be applied in these domains.

In the pursuit of effective PdM, a substantial volume of data is collected from machines, undergoes processing, and is subjected to analysis employing various ML algorithms. In the context of developing global PdM models for assets, the machines involved are typically regarded as edge devices. These devices transmit their data wirelessly to cloud infrastructure for processing and modeling, ultimately culminating in the creation of a unified PdM model. However, the transmission of substantial data volumes between edge devices and the cloud entails notable drawbacks, including high costs, increased delays, and privacy concerns. These issues hold particular significance within the context of future smart manufacturing and PdM within large-scale enterprises.

Consequently, **the third problem statement** central to this thesis revolves around the development of a comprehensive PdM model for assets and the strategic application of edge computing and collaborative PdM. This endeavor seeks to address data transmission cost reduction on wireless networks, enhance processing speed, and establish a global PdM model tailored for smart factories on a global scale. To achieve this, we intend to employ federated learning (FL), a collaborative machine learning approach that trains a global model using data from various devices without centralizing the data, thereby safeguarding data privacy and conserving network bandwidth (Konečnỳ et al., 2015, 2016; Yang et al., 2019b). In FL, models often perform less effectively compared to those trained in a centralized fashion, particularly when the training data differ significantly across local devices. This is a frequent occurrence in PdM applications due to the diverse anomalies present at edge devices and the variability introduced by human involvement in maintenance and reporting. Resolving this challenge is crucial in the context of FL for collaborative PdM.

1.2.3 Fast-Edge Learning for Collaborative PdM

In the field of PdM applications, the volume of data obtained from online sensors is substantial. As a result, it becomes imperative to consider this significant data volume when implementing FL techniques for PdM applications. Additionally,

time sensitivity and temporal awareness are crucial attributes for the effective execution of PdM activities. Therefore, it is essential to take into account time latency in PdM applications.

Migrating the learning process from centralized clouds to the edge allows edge servers to rapidly acquire real-time data generated by edge devices, facilitating the swift training of AI models. Consequently, distributing these models from servers to nearby devices enables these devices to effectively respond to real-time events, making them well-suited for PdM applications. However, despite the rapid advancement of computing speeds, wireless transmission of large data volumes by any device faces limitations due to restricted radio resources and the challenging conditions of wireless channels. This limitation creates a communication bottleneck that hampers fast-edge learning (Liu et al., 2021; Peng et al., 2022).

Communication schemes for FL can be categorized as digital or analog. Digital communication, though burdensome for wireless networks, assigns communication resources to each client's ML model parameters. Analog communication reduces overhead by allowing shared resources for transmitting FL models. However, incorporating the impact of noisy channels in analog communication complicates convergence analysis due to noise propagation during each communication round. Moreover, the collective impact of these noisy communications in the FL model aggregation on the final learning performance necessitates a comprehensive design and analysis approach.

Hence, **the fourth problem statement** revolves around understanding the impact of communication-induced noise during FL training on the convergence and accuracy performance of the ML model and how we can mitigate these effects and optimize client resources concurrently. In essence, we need to study and gain a better understanding of how noisy channels impact cyber twins in fast-edge analog FL.

1.3 Related Work

CPS has been gaining more and more attention in academic and industrial circles because they are believed to offer benefits for society, the economy, and the environment (Shi et al., 2011; Kim and Kumar, 2012; Liu et al., 2017). One of the key areas where CPS is applied is in autonomous driving (Lu et al., 2014; Parkinson et al., 2017) and smart manufacturing (Tao et al., 2019; Zhou et al., 2021). To realized these applications, several problems need be addressed. Therefore, this section lays the groundwork for the topic and briefly discusses the current state of knowledge. It focuses on the most important methods in WNCSs, which aims to tackle wireless communication imperfections and

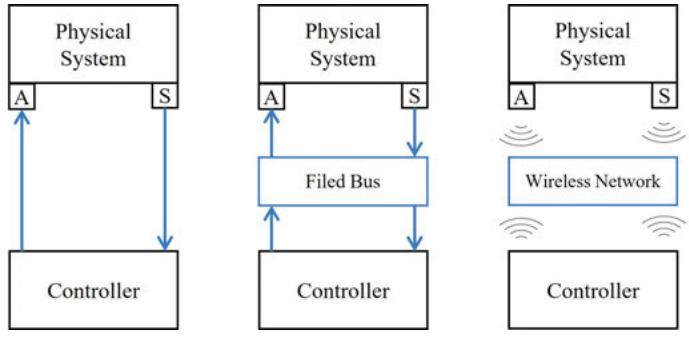


Figure 2. Development of Control System Structures. Physical system with sensor (S) and actuator (A) is connected to a controller, a traditional wired connection (left), a filed bus connection (middle), or a wireless network connection (right)

reduce bandwidth usage. We examine existing literature that discusses the challenge of controlling systems over wireless channels characterized by unreliable communication. It then goes on to describe potential approaches for implementing collaborative PdM on smart manufacturing using FL algorithms. Finally, it explores the use of FL for low-latency applications like PdM over the air.

1.3.1 Wireless Networked Control System

Before we dive into the research about WNCSs, let's briefly look back at how wireless control systems came into existence, even though they come with significant challenges when compared to traditional control setups (Antsaklis and Baillieul, 2007; Baillieul and Antsaklis, 2007). In the past, sensors and actuators were linked to a controller using direct wires (Figure 2, left). Later, a bus network system was introduced, which worked well for distributed systems and is still commonly used in automation and control (Figure 2, middle). In the present era, modern communication technology facilitates information sharing among system components, Actuators, sensors, and controllers, thereby enhancing overall system performance. Wireless connections enable the exchange of data from anywhere and facilitate advanced control technology for mobile objects (Figure 2, right). Traditional control theory makes the assumption that a system's components are connected through a perfect communication channel, assuming things like consistent data sampling, minimal communication delays, and no loss of information. In contrast, wireless networked control theory looks at how data is sent over imperfect channels, where transmission delays can change, and data loss can happen. In the following discussion, we will talk about how experts in control and communication have dealt with these issues and introduce methods that aim to

design control and communication systems together to overcome these challenges.

Numerous studies have previously investigated the design and stability analysis of various architectures, including delay models and processes involving message loss. These studies aimed to tackle the challenges arising from unreliable communication channels in sensor networks, with the ultimate objective of enhancing navigation and tracking applications (Sinopoli et al., 2004; Zhang et al., 2001; Xiong and Lam, 2007; Donkers et al., 2011). In the realm of real-time applications, delays play a crucial role. Numerous studies have been carried out to develop new protocols specifically designed to overcome the challenges of ensuring the reliable delivery of data packets within tight time constraints in low-power wireless networks (Zimmerling et al., 2017; Lu et al., 2015).

Research into the combined design of control and communication has been conducted through simulations in two separate studies. The first study (Li et al., 2016) investigates how routing choices affect control performance, while the second study (Ma et al., 2018) focuses on adapting the network protocol in real-time to respond to changes in the physical system's condition. This part of the thesis primarily emphasizes the design of a robust controller to withstand the impact of communication on control. In a section of Chapter 3, we will delve further into this topic. In today's industrial context, a major concern is the reliability of wireless communication. Building trust in wireless communication requires a combination of theoretical analysis and practical experiments conducted on a real-world CPS testbed. Therefore, practical initiatives related to wireless control will be elaborated upon in the subsequent sections.

We place our emphasis on the existing testbeds from both a structural perspective and in terms of execution techniques. These testbeds generally consist of three main components: simulation-based, hardware-based, and hybrid platforms. In this study, we particularly concentrate on hybrid platform testbeds, which incorporate physical components in the real world and link them with cyber components to facilitate communication within a virtual environment. This method has been adopted by several research groups, such as in the case of multi-agent vehicle system testbeds (Zhou et al., 2014; Jiang et al., 2015), automation robot testbeds in industrial settings (Damgrave and Lutters, 2019; Zhao et al., 2019; Kaczmarczyk et al., 2018), and Unmanned Aerial Vehicle (UAV) system testbeds (Saeed et al., 2014; Jamshidi et al., 2011).

In (Zhou et al., 2014), a platform was created involving several smart agents

with vehicle-to-vehicle (V2V) and vehicle-to-infrastructure (V2I) communication to assess a real connected vehicle testbed. However, this testbed has a drawback as it lacks a digital replica of the system for virtual environment testing. Another approach, in (Jiang et al., 2015), introduced a system of connected vehicles resembling a complex CPS. They used hardware-in-loop simulation technology, incorporating ZigBee and Wi-Fi radio modules for V2V and V2I communication. They also used a wireless network simulator to build a multi-vehicle testbed for cross-layer cooperative communication simulation. This testbed, developed by the University of Waterloo, aimed to validate various cooperative scenarios and collaborative interactions in indoor laboratory settings. Nonetheless, it also revealed limitations, such as the need to create a virtual link between the virtual and physical testbeds and issues with the accuracy of indoor positioning systems.

The testbed described in (Saeed et al., 2014) is created to simplify the control of physical components in UAV-related CPS experiments, making them more manageable and cost-effective. This testbed uses a single camera positioned at the center of the setup, which tracks drones with unique color tags to determine their positions. The UAV controller's job is to move the drones to specific x, y coordinates based on feedback from the positioning system. In (Jamshidi et al., 2011), a cyber-physical control system for UAVs offers a testbed for evaluating multi-vehicle cooperative control algorithms and studying the impact of time delays in network communication. Depending on the architecture of the CPS, this testbed can be either centralized if the algorithm runs on a base station or decentralized if the algorithm operates on each UAV. At the simulation level of this testbed, UAVs can maintain a predefined altitude using fuzzy rules and adaptive control.

In a recent analysis of the Swedish-German Testbed for Smart Production, the development of a testbed designed to facilitate smart manufacturing and model-based analytics through cloud-based applications is examined. This testbed functions as a validation platform specifically tailored to the powertrain manufacturing of heavy vehicles, thereby addressing the demand for cross-location development in the realm of Industry 4.0 and smart production. It comprises an integrated hardware and software interface that connects a network of interlinked machine tools to data generated by sensors and connectors. This data is subsequently stored within a digital twin, allowing for real-time visualization of the production process.

In the context of future wireless CPS, we have interconnected systems that work together. Consequently, there's a clear need for a specialized testbed that simulates a network of these wireless CPS systems. This testbed should mimic

complex systems with rapid dynamic changes and effectively integrate wireless communication and control methods. Its importance lies in its capacity to allow for testing different situations, enabling a comprehensive evaluation of wireless CPS, including both network and control elements. Creating such a testbed is essential for improving our understanding and capabilities in this critical field.

1.3.2 Control system Under Limited Bandwidth

Ongoing research in WNCSs explores the use of event-triggered control (ETC) as an alternative to traditional time-triggered control (TTC). In TTC, data is periodically gathered from sensors and transmitted to the controller. In contrast, ETC relies on event-triggered systems, where data transmission occurs when specific event conditions related to system states or control parameters are met. This approach, which is not reliant on a fixed schedule, transmits data only when it contains valuable information, reducing the load on the communication network. ETC and event-triggered state estimation (ETSE) algorithms have gained significance due to their potential to deliver high-performance control in resource-limited systems. For comprehensive insights, you can refer to related literature (Pan et al., 2022; Cao et al., 2022; Xu et al., 2022; Cao et al., 2023) for control and (Shi et al., 2016; Trimpe and Campi, 2015; Schmitt et al., 2022; Zhong et al., 2023) for state estimation.

Many researchers have recently emphasized the significance of a network control system and the attainment of high-performance control on resource-constrained systems with ETC. This is evident in various studies, such as (Guinaldo et al., 2015; Rong and Wang, 2021; Tan et al., 2020), which address the control of multi-agent systems through event-based communication. Additionally, (Trimpe and Baumann, 2019; Zhu et al., 2023; Liang et al., 2019) discusses the use of Kalman filtering as an estimator in the design of event-based state estimation systems, while (Schenato, 2008; Leong et al., 2016; Li and Peng, 2018) explores optimal state estimation in the context of networked control systems, particularly in the presence of packet drops.

State estimations and predictions play a pivotal role in an event-based triggering system, primarily centered around calculating states with a minimum mean square error (Wu et al., 2012; Martínez-Rey et al., 2015). Researchers have introduced various approaches to adapt Kalman filtering under conditions of wireless communication imperfections, such as data packet drops (Zhong and Liu, 2021; Feng et al., 2019; Nie et al., 2021). Given the distributed nature of WNCSs, recent investigations have concentrated on distributed Kalman filtering. In this context, each agent within a WNCS computes local state estimations and predictions using Kalman filtering based on its own sensor data

and the information exchanged with other agents (Qin et al., 2020; Zhang et al., 2019a; Yang et al., 2020). These studies have put forward several algorithms addressing consensus issues, and many of them are rooted in different forms of Kalman filtering, aimed at devising novel methods for event-based triggering in WNCSs.

The ETC method traditionally requires continuous agent sampling to assess whether the ET condition has been met. To overcome these constraints, the concept of self triggered control has been introduced (Chen et al., 2020a; Cui et al., 2023). With this approach, it becomes possible to predict the need for future sampling and determine the next triggering time at the previous trigger event. However, both approaches necessitate setting a lower bound on the trigger interval to prevent Zeno behavior which is a phenomenon in hybrid systems where an infinite number of discrete transitions occur within a finite time interval. To address these issues, periodic event-triggered control has been proposed for synchronizing discrete-time linear stochastic dynamic systems. Several research studies have explored the application of periodic event triggering to consensus problems in multi-agent systems, which can be found in (Garcia et al., 2016; Liu et al., 2020a).

Numerous control design approaches have been developed for multi-agent systems that rely on wireless communication for coordination and addressing consensus challenges. For instance, in (Pan et al., 2017; Xiong et al., 2022), there is a focus on state feedback control in multi-agent systems when dealing with data packet drops. Additionally, in (Zheng et al., 2016; Qiang et al., 2022), a distributed model predictive control algorithm is introduced, tailored for diverse multi-agent systems featuring directional and unidirectional network topologies. Furthermore, (Wang et al., 2019) presents an event-triggered consensus strategy that incorporates state feedback for linear multi-agent systems, accounting for the impact of random packet losses.

1.3.3 Federated Learning for Anomaly Detection and PdM

To achieve effective PdM, substantial volumes of data need to be collected, processed, and subsequently analyzed by a ML algorithm. Edge and fog computing can process this data through the utilization of distributed algorithms, offering opportunities to reduce data transfer costs and enhance processing speed, particularly in PdM applications (Teoh et al., 2021). In (De Donno et al., 2019), three primary techniques have been introduced, which employ distributed machine learning algorithms and data processing on intermediary nodes. These techniques are classified based on where the data is processed: Edge, Fog, and Cloud. As edge computing has become a vital

paradigm for IoT-based systems, endeavors are being directed towards utilizing devices situated at the edge of the network for performing computations whenever feasible, rather than relying solely on the cloud for data processing (Zeng et al., 2020).

FL offers a collaborative approach to PdM at the edge level, utilizing cross-company data from various manufacturing sites or across multiple organizations. This method is designed to share failure pattern data related to a specific asset without revealing sensitive raw data. In reference (Mohr et al., 2021), a novel distributed PdM algorithm integrating FL and blockchain mechanisms is proposed. One challenge with implementing FL in PdM applications is that certain edge devices may lack sufficient computational resources to train the global model promptly. These resource-constrained clients can cause delays in model aggregation and may even disconnect during training iterations, hindering efficient collaborative learning among edge devices. To address this issue, the Split Pred framework for collaborative PdM is introduced in (Bharti and Mcgibney, 2021), enabling cross-device FL to facilitate reliable model training on edge devices.

Reference (Park et al., 2018) introduces a real-time fault detection system for edge computing, using a two-layer architecture with a real-time fault detector based on Long Short-Term Memory (LSTM) recurrent neural networks. A system architecture for edge-based PdM applications in IoT-based manufacturing has been presented in (Chen et al., 2018). It emphasizes the advantages of distributed learning in edge computing, particularly regarding low latency response for edge control and bandwidth optimization. The cooperation among edge, fog, and cloud computing resources is discussed to illustrate their functionality. An empirical study on failure prediction in the production line using FL has been conducted in (Ge et al., 2022). They develop federated Support Vector Machine (SVM) and random forest algorithms for horizontal and vertical FL scenarios. Their analysis demonstrates that the distributed FL algorithm can effectively replace centralized methods for failure and maintenance prediction.

An important aspect of PdM is anomaly detection, and some research has focused on deploying FL algorithms in this context. For instance, in (Sater and Hamza, 2021), a novel FL algorithm for the LSTM framework is introduced and evaluated for anomaly detection in sensor behavior in smart building applications. This approach involves a local LSTM model on sensor edge devices and a global model on fog that aggregates weights, updates parameters, and distributes them among the sensor edge devices. Results indicate that this method converges twice as fast as the centralized LSTM model during training.

Similarly, in (Liu et al., 2020c), authors propose a communication-efficient FL algorithm for sensing time-series data in distributed anomaly detection applications. They introduce an attention mechanism-based Convolutional Neural Network Long Short-Term Memory (AMCNN-LSTM) model to accurately detect anomalies by capturing essential features using CNN and predicting future time-series data using LSTM.

1.3.4 Federated Learning Over the Air Communication and Computation

In light of the importance of addressing wireless channel effects in FL, recent research has focused on methods aimed at mitigating these effects (Zhu et al., 2019; Yang et al., 2019a; Chen et al., 2020b). These studies delve into the intricacies of analog aggregation techniques designed for over-the-air transmission, harnessing the inherent property of signal superposition in the wireless multiple-access channel. Analog over-the-air aggregation stands out as a highly promising technique that significantly enhances spectral efficiency and reduces multi-access latency. Its widespread use in Federated Averaging (FedAvg) is due to its ability to enable participating devices to share only the sum of their local gradients or model parameters. These methods have been extensively explored in various investigations, as evidenced in works such as (Amiri and Gündüz, 2020; Krouka et al., 2021).

However, it is crucial to acknowledge that noisy channels can have a detrimental impact on analog over-the-air communication and computation, especially when working with analog over-the-air aggregation. In (Ang et al., 2020), the primary focus of the research was on addressing noise in wireless communications for federated learning. The approach involved the creation of an expectation-based model and a worst-case model. The study introduced a sampling-based successive convex approximation algorithm to tackle the problem effectively, by incorporating noise as a regularizer in the loss function during the training process. The outcomes of this approach were evident in improved prediction accuracy and reduced loss values, showcasing its effectiveness in mitigating the impact of noise.

Furthermore, the influence of a noisy channel on the performance of the FedAvg algorithm was examined in (Amiri et al., 2021). The study's findings revealed that, particularly in the case of noisy downlink transmission, noise could not be effectively mitigated solely through step-size design, making it challenging to ensure precise convergence. Consequently, addressing this fundamental issue necessitates the imposition of strict requirements on the estimation of noise associated with aggregated model weights. These insights

emphasize the significance of accounting for and addressing the challenges posed by noisy channels in the context of analog over-the-air aggregation within the realm of FL.

1.4 Research Objectives and Questions

In response to the research gaps identified and comprehensively examined in Sections 1.2 and 1.3, this dissertation outlines four specific research objectives. These objectives are pursued through the formulation of pivotal research questions, which are outlined below:

Objective I: *To investigate the impact of wireless communication on real-time system performance and aiming to develop strategies for mitigating these effects.* The following questions are addressed to accomplish this objective:

- **RQ1:** In what way can communication imperfections be investigated within a WNCS?
- **RQ2:** In what way can cyber twins be developed in order to be effective and applicable for industrial use?

Objective II: *To enhance control algorithms in order to make them resilient to communication imperfections and disruptions, ensuring robust system operation in wireless communication environments.* The following questions are addressed to achieve this objective:

- **RQ3:** In what way can WCNS jointly optimize communication and control algorithms?
- **RQ4:** In what ways can WNCS be designed to address the challenges posed by harsh environmental conditions, particularly burst packet drops?

Objective III: *To design and develop user-friendly cyber twins specifically tailored for PdM operations, facilitating PdM tasks in industrial settings.* The following questions are addressed to attain this objective:

- **RQ5:** In what way a global FL model for PdM applications can be designed?
- **RQ6:** How do imperfections in wireless communications impact cyber twins' performance, especially in global PdM applications?

Objective IV: *To conduct a comprehensive study on the influence of communication loss on PdM cyber twins and explore methods for further*

improving their performance and reliability in the presence of noisy channel. The following questions are addressed to achieve this objective:

- **RQ7:** How can communication delays be minimized in time-sensitive applications like global PdM??
- **RQ8:** How a system that minimizes the effects of communication imperfections can be constructed?

1.5 Research Contributions

Four new methods have been developed and proposed for this PhD thesis. One method addresses objectives one and two, two methods pertain to objective three, and the final method relates to objective four. These newly developed methods have been published in five scientific papers, which are listed at the beginning of this thesis. In the following sections, we present the contributions of these methods. Figure 3 illustrates the connections between the scientific articles, research objectives, contributions, and associated questions.

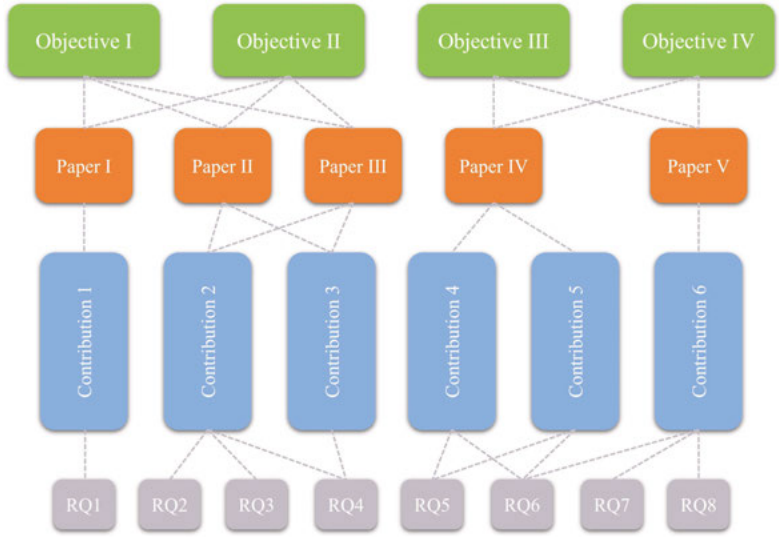


Figure 3. Mapping between the scientific articles, research objectives, contributions, and research questions

- **Contribution 1:** A multi-vehicle testbed for cooperative driving scenarios within a Wireless Cyber-Physical System of Systems (Wireless CPSoS) has been proposed. The testbed, comprising four robot vehicles equipped with Raspberry Pi microprocessors and sensory components, facilitates the verification and validation of cooperative driving

algorithms involving WNCS. To address the challenge of wireless communication imperfections, the method employed for achieving the research objective involves Model Predictive Control (MPC) with a state observer based on a Kalman Filter (KF). This testbed offers a practical platform for testing and evaluating advanced cooperative driving scenarios, laying the foundation for future research and development in the realm of wireless technology and control systems.

- **Contribution 2:** A novel Event-Triggering (ET) mechanism has been proposed which is tailored for distributed multi-agent systems operating in the presence of packet drop imperfections within WNCS. Our event-triggering mechanism adopts a parallel approach and utilizes Distributed Event-Based State Estimation (DEBSE) to design a parallel event-triggering algorithm capable of handling high packet drop probability conditions typically encountered in industrial environments. The primary goal is to sustain control performance at the desired level in consensus problems within multi-agent systems. By incorporating this ET algorithm into WNCS, we achieve a significant reduction in network resource usage. Although our proposed Parallel Event-Triggering (PET) algorithm slightly increases network resource utilization compared to ET, it consistently maintains satisfactory control performance in multi-agent consensus problems, even under challenging packet drop conditions.
- **Contribution 3:** Explores the application of distributed event-triggered control techniques to address leader-follower consensus problems in vehicular platoons, particularly in challenging environments with a high probability of burst packet drops in communication channels.
- **Contribution 4:** A new federated model called FedSVM has been introduced. It is a federated SVM model designed for PdM applications within a FL framework. FedSVM specializes in anomaly detection while safeguarding the privacy of individual assets at the edge level. It includes a preprocessing step that employs a moving average strategy on sensor data, optimizing memory usage and making it well-suited for real-time PdM applications. This method substantially enhances the efficiency of online PdM operations.
- **Contribution 5:** A new federated model called FedLSTM has been proposed. It is a federated LSTM model designed for PdM applications within a FL framework. FedLSTM specializes in predicting the remaining useful life (RUL) of assets by effectively learning from sequential data at the edge level. The incorporation of a moving average strategy reduces consecutive data blocks, significantly enhancing the training efficiency

of the model at the fog level. This approach contributes to streamlined communication and more rapid local model convergence in the context of collaborative PdM.

- **Contribution 6:** A hierarchical approach to PdM has been proposed, building upon contributions four and five. The key feature of our approach is the utilization of Over-the-Air Analog Computation and Communication (OACC) for the FL algorithm at the edge level. This choice is motivated by the advantages of low latency, making it well-suited for PdM applications while also enhancing spectral efficiency. Our proposal includes Federated Stochastic Variance Reduced Gradient Over-the-Air Communication and Computation (FSVRG-OACC), which is a distributed approach to address the optimization challenge for PdM at the edge level, leveraging OACC. FSVRG-OACC employs analog over-the-air aggregation, enabling effective handling of highly noisy communication channels and contributing to improved convergence in minimizing the cost function associated with the ML algorithm.

1.6 Research Methodology

The flowchart in Figure 4 shows the system overview of the methodology used in this thesis. After conducting a literature review and formulating research questions, the primary focus of this thesis, which is the development of Wireless CPS in the presence of wireless imperfections, was executed. Initially, we set up a testbed for a Wireless CPSoS by deploying four robot vehicles capable of implementing cooperative driving algorithms within a WNCS. To control the platooning of these vehicles in the presence of wireless imperfections, we implemented an MPC with a state observer based on the KF. The results from these implementations successfully verify the feasibility of our approach.

The second aspect of this thesis is dedicated to the development of a distributed ET algorithm for multi-agent systems operating in wireless networks characterized by a high likelihood of packet drops. This ET approach adopts a parallel framework and integrates the DEBSE technique to formulate a PET algorithm. To assess its efficacy, the algorithm has been subjected to simulation testing, where it was employed to control a mathematical model of vehicle platooning, resembling the conditions of the wireless CPSoS testbed, operating over a wireless network with a significant rate of message drops.

The third dimension of this thesis revolves around developing a distributed ML solution for collaborative PdM. The new methods were approached through a combination of theoretical evaluation using mathematical and

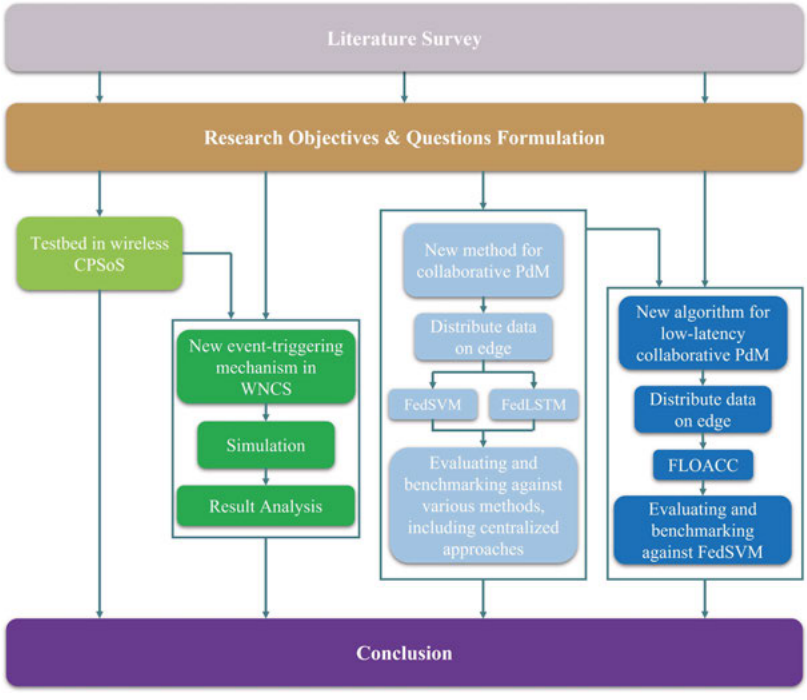


Figure 4. Methodology followed in the thesis

statistical models, as well as practical testing using a renowned dataset. In the context of RUL prediction, we chose to work with the widely recognized and benchmark CMAPSS dataset from NASA, which simulates a commercial modular aero-propulsion system. To address the collaborative PdM, we deployed mathematical models in a distributed manner and implemented the FedSVM and FedLSTM algorithms. The distributed RUL prediction and anomaly detection outcomes were thoroughly scrutinized and compared against other centralized approaches.

The fourth facet of this study focuses on leveraging OACC for the FL algorithm at the edge level within collaborative PdM. FSVRG-OACC has been proposed and implemented. We adopted a hierarchical approach to PdM, integrating FLOACC over the CMAPSS dataset. To assess its performance, we compared the results with those obtained using the FedSVM algorithm. Figure 5 shows how the proposed methods are connected to the scientific papers where the related results have been published.

In the final phase of the research methodology, we summarized the major discoveries and conclusions drawn from this study.

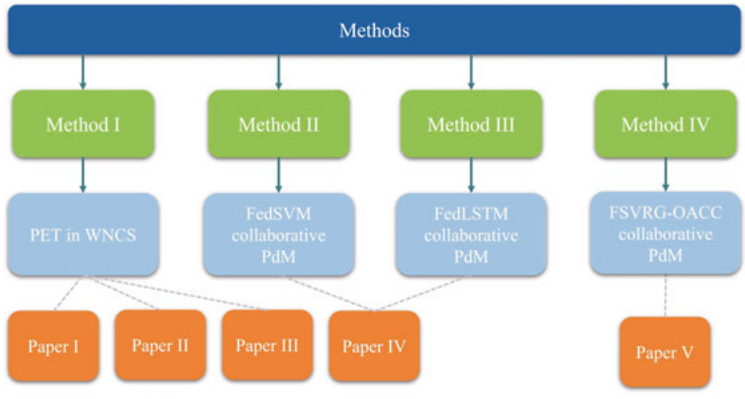


Figure 5. Relation between the proposed methods and the scientific papers

1.7 Thesis Outline

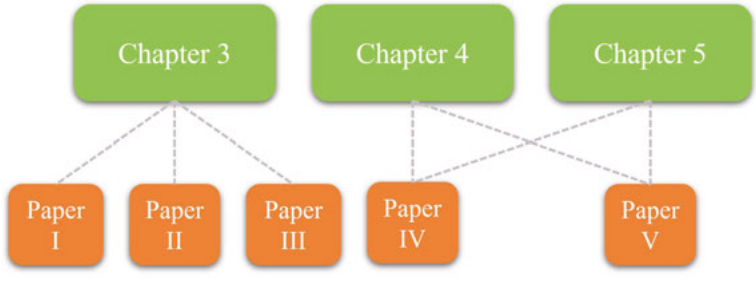


Figure 6. Relationship between the scientific papers and Chapters 3 to 5.

This thesis addresses the fundamental research problems outlined in Section 1.2, with contributions encompassing wireless control, ML over wireless networks, and stochastic distributed models for collaborative PdM applications. The thesis is structured into two main parts. Part I summarizes the study and the remainder is organized as follows. Chapter 2 briefly overviews the fundamental concepts and algorithms that form the foundation of this research. This chapter delves into WNCS, exploring topics such as the control implications in wireless networks, ET mechanisms, and predictive triggering. It continues to focus on ML over wireless networks, FL in PdM applications, OACC, and Over-the-Air Distributed ML. Chapters 3 to 5 elucidate the developed methods and their performance, presented in the form of five scientific papers. In particular, Chapter 3 describes the implemented Wireless CPSoS testbed and explains the distributed ET algorithm for Multi-Agent Systems over a wireless network, Chapter 4 details the FedSVM and FedLSTM for collaborative PdM, while Chapter 5 delves into Over-the-Air FL in a noisy industrial environment, emphasizing its application for low-latency collaborative PdM. Figure 6

illustrates the link between Chapters 3 to 5 and the scientific papers. Finally, Chapter 6 provides a conclusion and offers insights into potential avenues for future research. Part II comprises the compilation of the five publications.

2 Theoretical Background

This chapter serves as an essential foundation for understanding the subsequent sections of the thesis. It is divided into third main parts. The first part introduces the concept of WNCS and discusses the key challenges addressed by control theory in the context of wireless networked systems. Within this part, we delve into the MPC algorithm, which is capable of handling time delays and packet loss in control systems. Additionally, we explore the Event-Based Control System, a method designed to reduce the communication bandwidth usage in control systems. This section further narrows down to the introduction of Event-Based State Estimation (EBSE). In the second part, we provide an overview of ML and delve into the ML models commonly employed in PdM applications, including tasks like RUL estimation and anomaly detection. Lastly, this section introduces the FL algorithm as a prominent approach for dealing with data distributed across devices over wireless network, particularly in the context of edge computing. The third part delves into the details of OACC. It includes a mathematical description of the system model and explores the concept of analog over-the-air aggregation, which pertains to distributed ML. Within this context, issues related to fading and additive noise on wireless channels emerge as critical considerations.

2.1 Wireless Networked Control System

WNCS represent a category of control systems in which sensors, controllers, and actuators communicate through a wireless digital network (Figure 7), which may introduce delays and potential message losses. WNCS utilize wireless devices, offering several advantages over their wired counterparts, including ease of deployment, flexible architecture, cost-effective installation and maintenance, absence of cabling, and enhanced mobility.

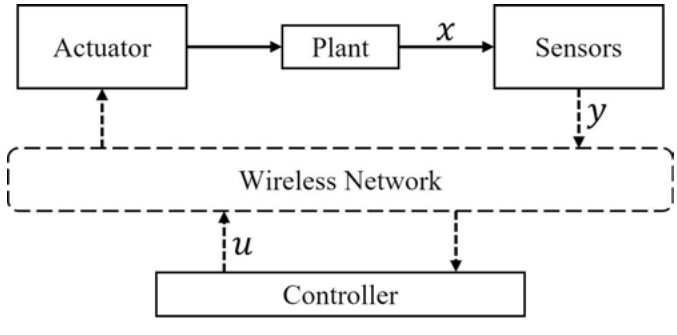


Figure 7. Schematic of a WNCS.

The interdisciplinary field of WNCS combines control theory, communication technology, and computer science to develop a dynamic setting for real-time control systems. In this environment, there are limitations on the resources available, and the workload can change, which creates uncertainties about the availability of computing resources and communication resources, like bandwidth (Figure 8).

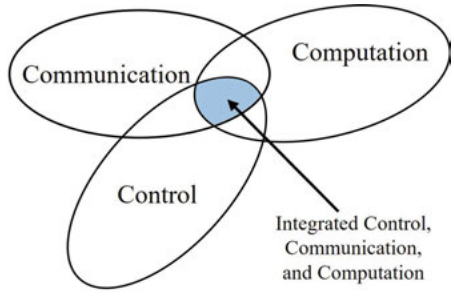


Figure 8. WNCS lies at the intersection of control, communication, and computation.

In traditional control theory, it deals with dynamic systems that are connected through perfect channels, assuming that data is collected at regular intervals, communication delays are very short or constant, and no data is lost. In contrast, communication theory focuses on how information is transmitted over imperfect channels, which can have different delays and might lose some data along the way. Furthermore, when systems engineers design things, they often assume that the control and communication tasks have fixed schedules that repeat at regular intervals, come with strict deadlines, and have well-understood worst-case execution times (WCET). Nevertheless, these assumptions may not always be valid, particularly in scenarios where resources change over time. To design and put into action control methods and communication rules for real-time networked control systems, it is essential to incorporate ideas and techniques from computer science and communication technology into the process of designing control systems. This approach leads to a combined design for control, communication, and computing.

The linear physical system in Figure 7 corresponds to a differential equation of the form

$$\mathrm{d}x(t) = \Phi x(t)\mathrm{d}t + \Gamma u(t)\mathrm{d}t + Q\,\mathrm{d}W(t), \tag{2.1}$$

where $x(t) \in \mathbb{R}^n$ is the plant state, $u(t) \in \mathbb{R}^m$ is the control input, $W(t) \in \mathbb{R}^n$ is a multi dimensional Wiener process capturing process noise, $\Phi \in \mathbb{R}^{n\times n}$ is the state transition matrix, $\Gamma \in \mathbb{R}^{n\times m}$ is the input matrix, and $Q \in \mathbb{R}^n$ is the covariance matrix. Because of digital communication and the integration of CPS in embedded devices, control laws are primarily implemented in a digital

format. Wireless networks pose significant considerations in control system design and performance. Some of these considerations, which are addressed in this thesis, are outlined below.

- **Reliability:** Ensuring that sensor data and control commands are not lost is crucial for control algorithms. However, improving reliability through retransmissions may consume more energy, so a balance between reliability and energy usage must be found.
- **Latency:** Timely delivery of sensor data and control commands is essential for control system performance and stability. Retransmissions for reliability can introduce delays, so a tradeoff between reliability and latency is necessary.
- **Data Rate:** Advanced control strategies may require more computation and data transmission, and the choice of control strategy affects the data volume. Scheduled control mandates timely data transmission, while event-based control focuses on access when events occur.

2.1.1 Control Implications of Wireless Networks

In the context of WNCS, the data acquired by the sensor and the control signal generated by the controller are communicated through a network. This transmission process introduces potential challenges, such as packet loss, significant delays, jitter, quantization issues, packet fragmentation, and other related concerns. One prevalent approach to address these imperfections involves employing MPC, which can effectively compensate for both delays and packet loss. Think about sending not only the initial control sequence value but also larger segments of this sequence. This offers a backup input in case of packet loss during subsequent transmissions. MPC, as a predictive control method, provides a sequence of input values that can compensate for network delays and losses (Findeisen and Varutti, 2009; Grüne et al., 2012).

This approach addresses imperfections in wireless communication between the controller and actuator. However, there is a possibility of dropped measurements between sensors and the controller. Hence, an estimator is required to reconstruct plant states that are missing on the controller side as well. Figure 10 depicts the WNCS seamlessly integrated with an MPC and an KF serving as a state observer. τ_{ca} signifies the time delay, and p_{ca} indicates the probability of message drop between the controller and the actuator. On the sensor side, τ_{so} denotes the time delay, and p_{so} represents the probability of message drop between the sensor measurements and the observer.

By sampling the plant with a time-varying sampling period, denoted as $h_k = t_{k+1} - t_k$, and expressing this period as the sum of a base sampling time

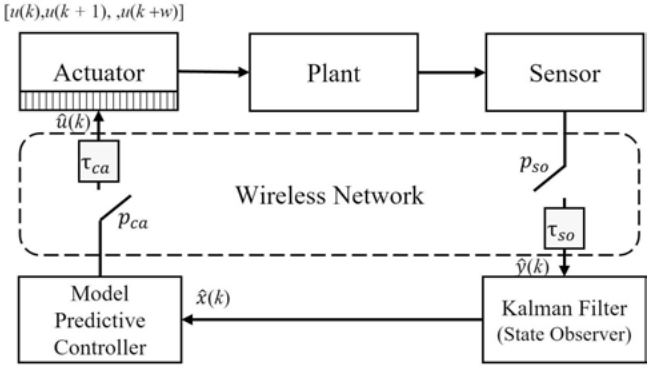


Figure 9. Integration of MPC and State Observation in a WNCS

h_0 and a term τ_{so}^k, the discrete-time formulation of the continuous-time plant dynamics (Equation 2.1) is derived as follows, assuming a zero-order hold.

$$x_{k+1} = Ax_k + Bu_k + w_k, \tag{2.2}$$

$$y_k = Cx_k + v_k, \tag{2.3}$$

here, h_0 is a constant in $h_k = h_0 + \tau_{so}^k$, τ_{so} satisfies $0 < \tau_{so}^k < h_k$, and the sampling period h_k is bounded by $0 < h_{min} \leq h_k \leq h_{max}$. The matrices A and B are defined as $A = e^{\Phi h_k}$ and $B = \int_0^{h_k} e^{\Phi s} ds B$, respectively. Additionally, we make the assumption that the random sequences $\{\tau_{so}^k\}_0^\infty$ and $\{h_k\}_0^\infty$ are independent and have known distributions.

As illustrated in Figure 10, measurements $y(k)$ and control inputs $u(k)$ are transmitted via the wireless network. It is assumed that upon transmission, both reach the observer and the system with specific delays and probabilities, each governed by independent Bernoulli processes. The Bernoulli processes are denoted by β_k and α_k, which are independent and identically distributed (i.i.d.) binary variables. These variables indicate whether messages were lost ($\beta_k = 1, \alpha_k = 1$) or successfully received ($\beta_k = 0, \alpha_k = 0$). Modeling the packet loss sequence results in the subsequent expression for the sensor measurements in the observer and the control value in the actuator.

$$\hat{y}_k = (1 - \beta_k)(Cx_k + v_k), \tag{2.4}$$

$$\hat{u}_k = (1 - \alpha_k)u_k, \tag{2.5}$$

β_k and α_k represent Bernoulli random variables, where the probabilities of packet loss are $\mathrm{P}(\beta_k = 1) = p_{so}$ and $\mathrm{P}(\alpha_k = 1) = p_{ca}$. These variables model the packet loss between the sensor and the observer and between the controller and the actuator, respectively. Ultimately, the subsequent linear difference

equations are derived, accounting for intermittent sensor measurements and control commands through WNCS.

$$x_{k+1} = Ax_k + B\hat{u}_k + w_k, \tag{2.6}$$

$$\hat{y}_k = (1-\beta_k)y_k, \tag{2.7}$$

where (x_0, w_k, v_k) are Gaussian variables that are uncorrelated with mean $(\bar{x_0}, 0, 0)$ and covariance (P_0, Q, R). If there is packet loss between the sensor and the observer, the KF state estimator should be employed to ensure accurate updates to the controller input. Therefore, the use of an estimator becomes necessary in WNCS to reconstruct the missing states. In cases of packet loss between sensors and the observer, the utilized KF state estimator is represented as follows

1. Time update:

$$x_{k+1|k} = A\hat{x}_{k|k} + B\hat{u}_k, \tag{2.8}$$

$$e_{k+1|k} = Ae_{k|k} + w_k, \tag{2.9}$$

$$P_{k+1|k} = AP_{k|k}A^{\mathrm{T}} + Q, \tag{2.10}$$

2. Measurement update:

$$\hat{x}_{k+1|k+1} = \hat{x}_{k+1|k} + (1-\beta_{k+1})K_{k+1}\left[y_{k+1} - C\hat{x}_{k+1|k}\right], \tag{2.11}$$

$$e_{k+1|k+1} = \left[I - (1-\beta_{k+1})K_{k+1}C\right]e_{k+1|k} - (1-\beta_{k+1})K_{k+1}w_{k+1}, \tag{2.12}$$

$$P_{k+1|k+1} = P_{k+1|k} - (1-\beta_{k+1})K_{k+1}CP_{k+1|k}, \tag{2.13}$$

$$K_{k+1} = P_{k+1|k}C^{\mathrm{T}}\left(CP_{k+1|k}C^{\mathrm{T}} + R\right)^{-1}, \tag{2.14}$$

here, y_k represents the actual sensor measurement, $C\hat{x}_{k|k-1}$ denotes the measurement prediction, K_k is the time-varying Kalman filter gain matrix, and $P_{k|k-1}$ is the error covariance matrix, which depends on the occurrence of packet loss between sensors and observer. If sensor measurements are dropped in one or several consecutive sampling periods, the estimator only runs the time update part in an open loop until the time when all sensor packets will be received successfully. A threshold exists for sensor measurement loss, crucial for the stability of the state estimator. As outlined in (Sinopoli et al., 2004), critical probabilities p_{so}^{crti} and p_{ca}^{crti} determine stability limits, with the estimated mean state covariance remaining bounded below these values. Beyond the thresholds, the covariance can diverge for some initial conditions.

At the controller side, the goal is to create an application-level solution capable of compensating for delays and message dropouts. The MPC controller serves as a solution that can anticipate the control horizon, sending an array of inputs to the actuators in the WNCS. This ensures that the actuator always has sufficient information to feed into the plant, even in the event of delays or message dropouts. MPC employs an optimization based control law, where the performance measure typically involves minimizing a quadratic cost. By specifying positive definite matrices as the performance weights (Q and R), the primary objective is to determine the optimal control input that minimizes the performance cost over an infinite horizon.

$$J_k(\hat{x},u) = \sum_{j=k}^{\infty} \hat{x}_{j|k}^T Q \hat{x}_{j|k} + u_{j|k}^T R u_{j|k}. \tag{2.15}$$

For a full-state feedback system, the cost function can be rewritten as follows to track the reference point (r).

$$J_k(r,\hat{x},u) = \sum_{j=k}^{\infty} (r_{j|k} - \hat{x}_{j|k})^T Q (r_{j|k} - \hat{x}_{j|k}) + \Delta u_{j|k}^T R \Delta u_{j|k}. \tag{2.16}$$

The cost function in Equation 2.16 is unconstrained, and the optimal solution to minimize it is determined by the linear quadratic (LQ) controller. In cases where constraints are present, there is no analytical solution available. However, in MPC, a strategy is employed where a prediction horizon, denoted as w, is defined. The goal is to solve a minimization problem with a finite-horizon cost. This approach is a key method in WNCS, and it enables the MPC to calculate an array of inputs for each time step $k \in 0,1,...,w-1$. This proactive strategy helps mitigate challenges such as delays and message drops.

$$J_k(r,\hat{x},u) = \sum_{j=k}^{k+w-1} (r_{j|k} - \hat{x}_{j|k})^T Q (r_{j|k} - \hat{x}_{j|k}) + \Delta u_{j|k}^T R \Delta u_{j|k}. \tag{2.17}$$

MPC is an approach that iteratively addresses a series of optimal control problems within a finite prediction horizon. By utilizing the model (Equation 2.2) as a constraint, MPC algorithm can be defined as follows.

Algorithm 2.1: MPC algorithm for WNCS

Init: Set k to 0, with the provided model of the plant.

1: Address the optimal control problem within a reduced time horizon, utilizing the most recent state measurement supplied by the Kalman Filter, denoted as $\hat{x}$. This involves determining a permissible array of control inputs, denoted as u, by minimizing the cost function while adhering to the model constraints.

$$\min_{u} \sum_{j=k}^{k+w-1} (r_{j|k} - \hat{x}_{j|k})^T Q (r_{j|k} - \hat{x}_{j|k}) + \Delta u_{j|k}^T R \Delta u_{j|k}$$

$$\text{subject to } u(k+j \mid k) \in \mathscr{U}$$

$$\hat{x}(k+j \mid k) = A\hat{x}(k+j-1 \mid k) + Bu(k+j-1 \mid k)$$

2: Define the static state feedback.

Static state feedback is defined as the first element of the computed optimal input sequence. This sequence is stored in the buffer and fed to the actuator of the plant one element at a time for implementation.

3: Move the prediction horizon in time by incrementing k to $k+1$, and then repeat this process, starting from step 1.

The iterative application of this algorithm establishes a feedback control law over an infinite time horizon, even though utilizing only a finite prediction horizon at each time instant. It is crucial to highlight that MPC relies on a plant model to predict the future behavior of the system and consequently determine an appropriate control sequence.

As discussed, we employed MPC and Kalman Filter state estimation to address imperfections in WNCS. However, designing WNCS involves balancing various trade-offs, particularly those involving control performance, communication performance, and network lifetime. Therefore, it is crucial to design control and communication jointly, considering these factors.

2.1.2 Control and Communication Co-design

The principal challenge within WNCS is effectively combining communication and control system designs because they are closely connected. This integrated approach aims to enhance both the control performance and the network lifetime. However, control theory makes assumptions about factors like sampling instances, performance, and reliability, which can be challenging to fulfill using standard or specialized communication technologies. Conversely,

computer networks may possess limitations that are suboptimal for control systems. To reconcile this disparity, the proposal of a joint design for both the control and communication systems has been proposed (Cao et al., 2012; Chamaken et al., 2009).

From an architectural point of view, WNCS can be divided into two layers, as illustrated in Figure 10. The control layer encompasses the application functions of controllers, sensors, and actuators, while the communication layer manages various networking functions. Consequently, the optimization problem needs to be addressed in both aspects. This concept imposes constraints on the complexity of control algorithms and communication protocols, as well as considerations for throughput and reaction times.

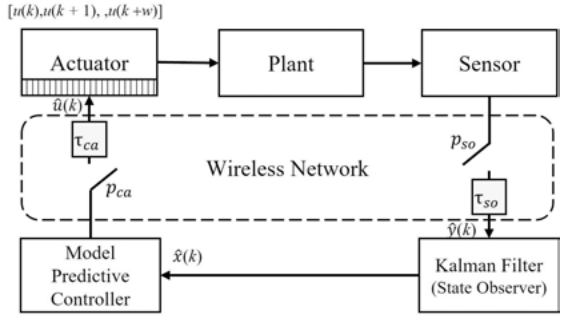


Figure 10. Joint design architecture of WNCS

Quality of control (QoC) and quality of service (QoS) are terms used to describe the control and communication requirements of a system. QoS focuses on efficiency aspects like latency, bandwidth, and loss rate for data flow between sensors, controllers, and actuators. To adapt to changing control setpoints and resource availability, the system defines a minimum QoS to ensure a basic level of QoC, as well as a preferred QoS for dynamic disturbances (Vashisht and Jain, 2019). Hence, the goal is to provide robust assurances for meeting the minimum QoS requirements and to fulfill the preferred QoS requirements to the greatest extent possible.

Another way to jointly design the communication and control of WNCS is through the event-based control schemes, like event and predictive triggering. The design of communication systems for event-triggered sampling has predominantly focused on the MAC layer. In this approach, sensing and actuating happen only when the system requires attention. Unlike periodic systems, the traffic pattern of event triggering is asynchronous. Event-triggered control can significantly reduce network traffic with little to no impact on control performance, as traffic is generated only when the signal changes by a specific amount (Araújo et al., 2013). Various studies have explored the

trade-off between the threshold crossing level in control systems and packet losses in communication systems (Ramesh et al., 2016; Vilgelm et al., 2016; Mamduhi et al., 2014). However, there is a gap in research when it comes to analyzing event triggering mechanisms for distributed multi-agent systems in the presence of message drop imperfections. We will address this gap in Chapter 3, but before doing so, let us delve into the fundamental theory of event-based control and triggering, focusing particularly on multi-agent distributed systems.

2.1.3 Event based control

In event-based control, the feedback loop closes when an event shows that the control error goes beyond an acceptable limit, prompting data transmission from sensors to controllers and controllers to actuators. This makes event-based control a key method for cutting down on communication load in a WNCS. In event-based control, data is not collected at regular intervals. Instead, it is determined by an event-triggered system.

This approach differ fundamentally from time-triggered control systems, where data is periodically collected from the sensors and sent simultaneously to the controller. In periodic sampling, communication occurs regardless of the control error size, even when it is small and feedback is not needed for performance requirements. This can lead to unnecessary use of communication and computing resources.

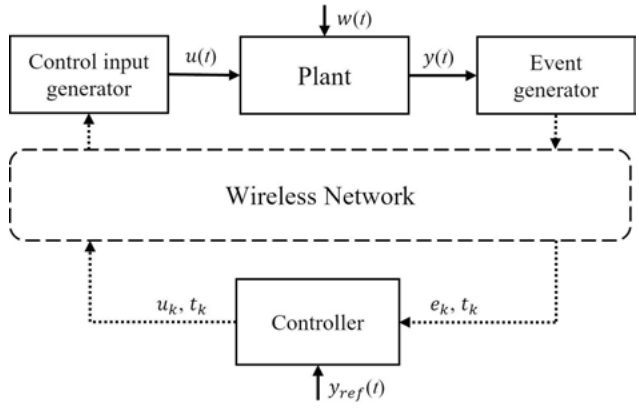


Figure 11. Event-based control of a WNCS

Figure 11 depicts the essential elements of an event-based control system for a WNCS. In this system, a continuous-time input signal $u(t)$ is generated by a control input generator. Simultaneously, an event condition is continuously assessed by an event generator. When the event generator detects that the

control error surpasses a predefined limit at a specific time t_k, an event e_k is triggered through the dashed lines, which represent the communication links in the figure. This event could be either the current state $x(t_k)$ or the current output $y(t_k)$, which is then transmitted to the controller. Upon receiving the event information, the controller calculates a new input u_k. The control input generator utilizes this updated input to calculate the continuous input signal $u(t)$ within the time interval $[t_k, t_{k+1})$ until the occurrence of the next event at time t_{k+1}. This cyclic process helps in adapting the control input based on the system's behavior, ensuring effective control in response to relevant events and decrease the communication load.

Various methods have been proposed for event-based control, taking into account the definitions and functions of the event generator and control input generator. Some of these methods include event-driven control, event-triggered control, and send-on-delta control (Aranda-Escolastico et al., 2020; Lunze and Lehmann, 2010). Additionally, there are conceptually different approaches such as predictive-triggered control and self-triggered control. In predictive-triggered and self-triggered control, the plant state is not continuously monitored by the event generator. Instead, the next event time t_{k+1} is predicted by the event generator at the current event time t_k (Heemels et al., 2012; Yi et al., 2018; Trimpe and Baumann, 2019). This allows the system to enter an idle mode until the predicted next sampling time, which is advantageous for low-power wireless nodes.

The plant (2.1), when coupled with state feedback K as given by

$$u(t) = -Kx(t), \tag{2.18}$$

results in the continuous closed-loop system

$$\dot{x}(t) = (\Phi - \Gamma K)x(t) + w(t), \tag{2.19}$$

This occurs when the state $x(t_k)$ is communicated to the control input generator at time t_k. The control input generator can then determine the same control input $u(t_k) = -Kx(t_k)$ as a continuous state-feedback controller. However, for all future time $t > t_k$, the control input generator needs to be aware of the disturbance $w(t)$. By defining $(\Phi - \Gamma K = \bar{\Phi})$, The control input can be expressed as

$$u(t) = -Ke^{\bar{\Phi}(t-t_k)}x(t_k) - \int_{t_k}^{t} Ke^{\bar{\Phi}(t-\tau)}v(\tau)d\tau. \tag{2.20}$$

It has been assumed that the control input generator can appropriately account

for the disturbance beyond the communicated state at time t_k.

The control input generator can be represented by the following model to determine the control input (2.20)

$$\dot{x}_s(t) = \bar{\Phi} x_s(t) + \hat{w}_k, \tag{2.21}$$

$$u(t) = -K x_s(t), \tag{2.22}$$

here, x_s denotes the state of the control input generator, and it is set such that $x_s(t_k^+) = x(t_k)$ for $t \geq t_k$. The solution of this model exactly corresponds to (2.20). Additionally, it is assumed that the disturbance is constant $w(t) = \hat{w}_k$ for $t \geq t_k$, but it should be estimated in the control input generator using an estimator. Figure 12 shows the block diagram of the control input generator.

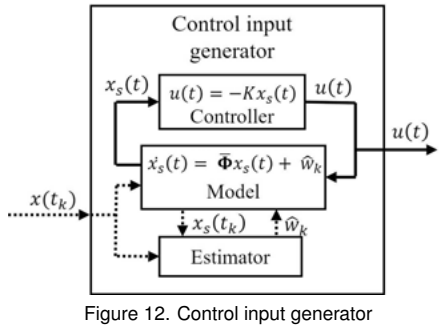


Figure 12. Control input generator

In the event generator block, events are generated by comparing the current state of the system, denoted as $x(t)$, with the estimated state in the feedback loop, represented as $x_s(t)$, considering a constant disturbance $\hat{w}_k$. An event is triggered whenever the difference between the measured plant state $x(t)$ and the estimated reference state $x_s(t)$ satisfies the condition

$$\|x(t) - x_s(t)\|^2 \geq \delta. \tag{2.23}$$

At the event time t_k, when this condition is met, the measured plant state $x(t_k)$ is sent to the control input generator for further processing. Figure 13 shows the block diagram of the event generator.

The event-based state-feedback loop structure is illustrated in Figure 14. It is clear that state estimators play a crucial role in the design of event-based controllers. Consequently, in the next subsection, we will delve into methods for designing event-based state estimators.

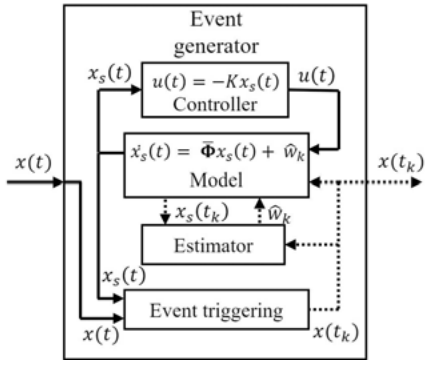


Figure 13. Event generator

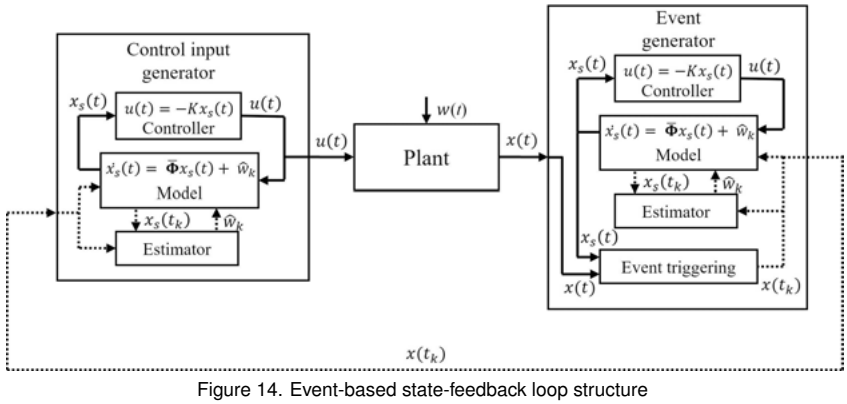


Figure 14. Event-based state-feedback loop structure

2.1.4 Event Based State Estimation

The main challenge in event-based estimation is designing the estimator. Usually, these estimators are designed to be optimal like mean square estimation error. However, because of the event-triggering conditions, the optimal estimators often do not have straightforward recursive structures. This happens because the Gaussian property of the state's conditional distributions, based on the available measurement information, is no longer maintained (Shi et al., 2016).

The most straightforward event-based state estimator involves ignoring information when there is no event and designing the estimator solely based on the received information. The resulting estimator corresponds to the Kalman filter state estimator designed for unreliable communication channels, as introduced in section 2.1.1. The key difference is that the Bernoulli random variable representing packet loss is replaced with the triggered decision variable (γ_k). If an event is triggered ($\gamma_k = 1$), the sensor measurement y_k becomes available to the estimator, and the measurement update in the Kalman filter is

akin to the classic Kalman filter. On the other hand, when the sensor measurement is not available ($\gamma_k = 0$), the filter only carries out the time update step.

2.1.5 Bayesian Filtering and Particle Filter

Bayesian filters are employed to generate precise estimates of the state in a dynamic system, relying on multiple observations despite the presence of noisy measurements. The Kalman filter, predominantly employed in this chapter, serves as a closed form solution to the linear Gaussian filtering problem. As a result of the linear Gaussian model assumptions, the posterior distribution is precisely Gaussian, eliminating the need for any numerical approximations. In many practical applications, it is common for the dynamic and measurement models to be nonlinear, thereby making the Kalman filter inappropriate. Therefore, a modified version of the Kalman filter, known as the Extended Kalman Filter (EKF), is utilized to approximate the nonlinear measurement and dynamic models through linearization. The concept of EKF is closely akin to Kalman filtering, utilizing a Taylor series expansion around the nominal solution for linearization.

An alternative nonlinear estimator is the particle filter, a recursive Bayesian state estimator that employs discrete particles to approximate the posterior distribution of the estimated state. In Bayesian inference, the primary inference challenge frequently involves computing an expectation over the posterior distribution as follows.

$$\mathrm{E}\left[\mathbf{g}(\mathbf{x}) \mid \mathbf{y}_{1:T}\right] = \int \mathbf{g}(\mathbf{x}) p\left(\mathbf{x} \mid \mathbf{y}_{1:T}\right) \mathrm{d}\mathbf{x}, \tag{2.24}$$

where $\mathbf{g} : \mathbb{R}^n \to \mathbb{R}^m$ is an arbitrary function and $p\left(\mathbf{x} \mid \mathbf{y}_{1:T}\right)$ is the posterior probability density of state $\mathbf{x}$ given the measurements $\mathbf{y}_1, \ldots, \mathbf{y}_T$. the challenge lies in the fact that evaluating such an integral in closed form is feasible only in a limited number of special cases and generally, numerical methods must be employed. Monte Carlo methods offer a numerical approach for computing integrals of the form (2.24). Monte Carlo denotes a broad category of methods where the computation of statistical quantities in closed form is replaced by drawing samples from the distribution and estimating these quantities through sample averages.

In the context of the particle filter utilizing the Monte Carlo approximation, N independent random samples $\mathbf{x}^{(i)} \sim p\left(\mathbf{x} \mid \mathbf{y}_{1:T}\right)$ are generated, denoted as $i = 1, \ldots, N$, and the expectation is estimated as

$$\mathrm{E}\left[\mathbf{g}(\mathbf{x}) \mid \mathbf{y}_{1:T}\right] \approx \frac{1}{N} \sum_{i=1}^{N} \mathbf{g}\left(\mathbf{x}^{(i)}\right). \tag{2.25}$$

Each particle has a weight $\{(w_k^{(i)}, \mathbf{x}_k^{(i)}) : i = .\ 1, \ldots, N\}$, for representing the filtering distribution $p(\mathbf{x}_k \mid \mathbf{y}_{1:k})$ such that at every time step k the approximation of an arbitrary function $\mathbf{g}(\cdot)$ can be calculated as the weighted sample average

$$\mathrm{E}\left[\mathbf{g}(\mathbf{x}_k) \mid \mathbf{y}_{1:k}\right] \approx \sum_{i=1}^{N} w_k^{(i)} \mathbf{g}\left(\mathbf{x}_k^{(i)}\right). \tag{2.26}$$

The subsequent step involves resampling, where particles with negligible weights are replaced by new particles in proximity to those with higher weights. This process ensures that only the most likely particles persist into the next iteration of the particle filter, enhancing the accuracy of estimation (Särkkä and Svensson, 2013).

2.2 Edge Computing and Machine Learning

In recent years, there has been a significant increase in the number of IoT devices. This growth can be attributed to rapid advancements in hardware, software, and wireless communication technology. These developments enable the collection of data, allowing for observation and measurement to transition seamlessly from the physical realm to the digital domain. Traditional cloud computing requires sending data from IoT devices to centralized servers for processing. Subsequently, the results are transmitted back to the devices. This process puts strain on the network, increases communication cost and network bandwidth usage, causes delayed system response, and puts data privacy at risk (Alli and Alam, 2020). To address these challenges, fog and edge computing have emerged as a promising paradigm, bringing computation and data storage close to where the data originates (Firouzi et al., 2022). Here, ML and data-driven models stand out as prominent tools for analyzing the data generated by edge devices (Lim et al., 2020).

In PdM applications, machines are outfitted with numerous sensors to collect a substantial amount of data. This data is then processed and analyzed by ML algorithms to predict potential failures in manufacturing equipment. Edge and fog computing play a crucial role in processing this data through distributed algorithms. They offer the advantage of lowering data transfer costs and enhancing processing speed, particularly in PdM applications. These distributed algorithms are classified based on where the data is processed: Edge, Fog, and Cloud (De Donno et al., 2019). The hierarchical and collaborative

edge-fog-cloud architecture, as illustrated in Figure 15, provides the advantage of processing data at the edge device level. Nevertheless, edge learning introduces various challenges, including latency and message loss that complicate ML algorithms. There are also bandwidth limitations for intensive computation processes, and concerns about privacy and security when sharing datasets. Those challenges should be specifically considered for PdM applications dealing with a large volume of sensor data. For instance, in the setup shown in Figure 15, only the model parameters need to be transmitted to the fog level, reducing the overall data transfer.

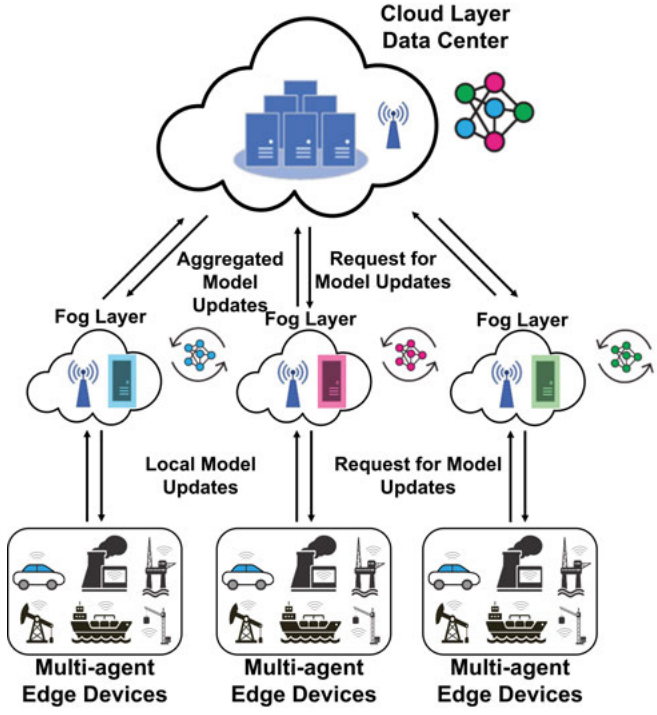


Figure 15. Hierarchical and collaborative edge-fog-cloud architecture

2.2.1 Machine Learning

PdM aims to predict or estimate the time when a machine or its components might fail. This is done using either a physical model or a data-driven model, employing ML techniques to replace faulty components before they fail. Data-driven models utilize methods from statistics, ML, and artificial intelligence to estimate RUL, assess the health of the component, or detect anomalies. This section delves into ML and its goals, covering the mathematical formulations of supervised learning. Examples include SVM and LSTM models, which are

well-known in PdM applications. SVM is effective for anomaly detection, while LSTM is utilized for predicting RUL.

ML is a field of computer science that focuses on learning patterns from data. In supervised ML, models are trained to predict an unobserved quantity using a dataset of observed data points. For instance, one could aim to predict the RUL of a machine using a dataset that includes past sensor data and failure history. In this context, the learning algorithm processes input (x,y), where $x \in \mathbb{R}^d$ is the sample vector, and $y_i \in \mathbb{R}$ is the corresponding label. Specially, the object of a learning algorithm is to find a prediction function $h : \mathscr{X} \to \mathscr{Y}$ mapping from $\mathscr{X}$ to $\mathscr{Y}$. The learning algorithm seeks a suitable h by minimizing a loss function $f : \mathscr{X} \times \mathscr{Y} \to \mathscr{Y}$, measuring the error between the predicted value $\hat{y} = h(x)$ and the true output y.

While the learning algorithm theoretically has the flexibility to choose any prediction function, $h(x)$ is constrained to a specific group of functions parameterized by a real vector w to handle complexity. The optimal value for the learning parameter w^* is then determined by minimizing the loss function $f(h(x,w),y)$.

$$w^* = \arg\min_w f(h(x,w),y). \tag{2.27}$$

For the training data set, consisting of a finite set of tuples (x,y) defined as $\mathscr{D} = (x_i,y_i)_{i=1}^D$, the optimization problem can be rewritten as

$$w^* = \arg\min_w \frac{1}{D}\sum_{i=1}^{D} f(h(x,w),y). \tag{2.28}$$

2.2.2 Model Selection

In this section, SVM and LSTM are explored as ML models in PdM applications. SVMs, in particular, are acknowledged as powerful supervised ML models with a solid mathematical foundation, capable of handling both linear and nonlinear classification (Murphy, 2012). This method can be employed as alarm notifications in PdM applications, detecting abnormal sensor data and notifying factory management when maintenance is needed for an asset close to failure.

The basic concept of SVM is depicted in Figure 16. It displays the decision boundary for classifying a two-class dataset with the maximum distance from the nearest training instance of each class. SVM seeks a hyperplane that effectively separates instances and has the widest margin from the edge of each class. This hyperplane, known as the maximum-margin hyperplane, sits midway between two parallel hyperplanes with the greatest possible separation between

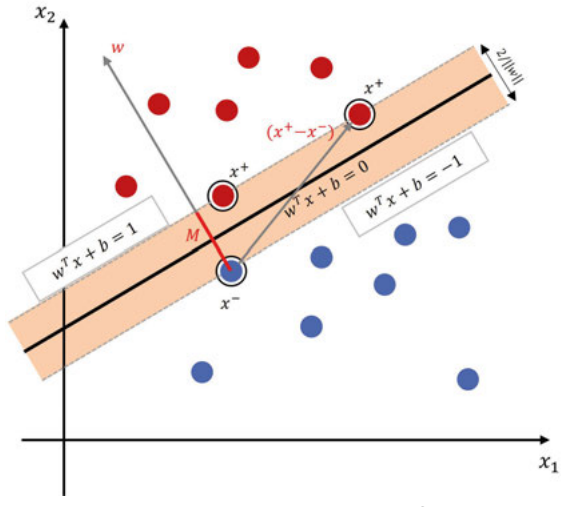


Figure 16. Decision boundaries in SVM

them. Any hyperplane in a d-dimensional space can be defined as $w^T x + b = 0$, where $w \in \mathbb{R}^d$ is the hyperplane normal vector, and b is the bias. The dashed lines representing the two parallel hyperplanes passing along each class can be expressed as $w^T x + b = 1$ and $w^T x + b = -1$. SVM aims to find the maximum-margin hyperplane and prevent instances from falling into the margin. Therefore, the following constraints can be defined for SVM

$$\begin{cases} w^T x_i + b \geq +1, \text{ if } y_i = +1 \\ w^T x_i + b \leq -1, \text{ if } y_i = -1 \end{cases} \tag{2.29}$$

To maximize the distance between these two hyperplanes, the distance, parameterized with w, should be calculated. It's evident that the instance x^+ satisfies the equation $w^T x + b = 1$ and the instance x^- satisfies the equation $w^T x + b = -1$. Therefore, it could be written as

$$w^T x^+ + b = +1, \tag{2.30}$$
$$w^T x^- + b = -1, \tag{2.31}$$

subtracting equation 2.30 from 2.31 can be expressed as

$$w^T (x^+ - x^-) = 2, \tag{2.32}$$

where vector w and vector $(x^+ - x^-)$ are illustrated in Figure 16. Equation 2.32

can be rewritten using the definition of vector multiplication as follows:

$$\|w\| \cdot \|x^+ - x^-\| \cos\alpha = 2, \tag{2.33}$$

Now it's clear that the distance between these two hyperplanes is represented by $\|x^+ - x^-\| \cos\alpha$, denoted as M in figure 16. Therefore it could be written that $\|w\| \cdot M = 2$ so the distance can be calculated as $M = \frac{2}{\|w\|}$, and its maximization corresponds to the minimization of $\|w\|$. To simplify the process of finding the optimum point, the objective function can be defined in a quadratic form as follows:

$$\begin{aligned} &\min \frac{1}{2}\|w\|^2 \\ \text{subject to } &w^T x_i + b \geq +1, \text{ if } y_i = +1 \\ &w^T x_i + b \leq -1, \text{ if } y_i = -1. \end{aligned} \tag{2.34}$$

This optimization problem can be expressed with a single constraint

$$\begin{aligned} &\min \frac{1}{2}\|w\|^2 \\ \text{subject to } &y_i(w^T x_i + b) \geq +1 \end{aligned} \tag{2.35}$$

The solution to this problem, denoted by w and b, yields the maximum-margin hyperplane. By employing the method of Lagrange multipliers, the optimization problem can be transformed into the Lagrangian function, allowing for optimization without the need to explicitly consider the constraints.

$$\min \frac{1}{2}\|w\|^2 - \sum_{i=1}^{D} \alpha_i [y_i(w^T x_i + b) - 1], \tag{2.36}$$

where $\alpha_i \geq 0$ are Lagrangian multipliers. The SVM model can be trained by solving this minimization problem. Another form of this problem, known as the dual form, exists, but for brevity, this thesis does not delve into it. Refer to (Murphy, 2012) for more information.

Another model is LSTM, a type of recurrent neural network. Neural networks are non-linear models that estimate relationships between input and output. In a fully connected neural network, the input layer is linked to hidden layers and an output layer through weights (Du et al., 2016). Let's look at a single neuron with $X \in \mathbb{R}^3$, depicted in Figure 17. The neuron takes a vector input from the

previous layer and computes a scalar output using the activation function $a(z)$

$$y = a(W^T X + b), \tag{2.37}$$

even though this is just a single neuron, in the broader context of a neural network, distinct weights W and biases b apply to neurons in different layers and at various positions within each layer.

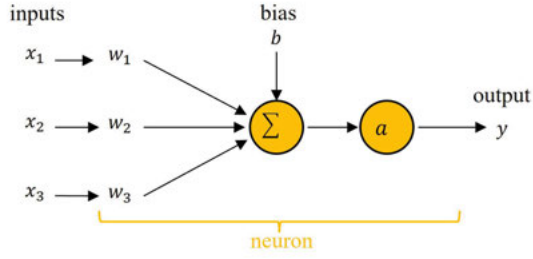


Figure 17. Visualization of a single neuron with $X \in \mathbb{R}^3$

A neural network aims to find values for W and b that effectively describe the observed data and, simultaneously, generalize well for unseen data. This is achieved by minimizing a loss function $f(W,b)$. For a regression problem, a commonly used mean squared error function is given by

$$f(W,b) = \frac{1}{D}\sum_{i=1}^{D}(y_i - \hat{y}_i)^2. \tag{2.38}$$

In the case of a binary classification problem, a frequently used loss function is the cross-entropy loss:

$$f(W,b) = -\frac{1}{D}\sum_{i=1}^{D}\left[y_i \log \hat{y}_i + (1-y_i)\log(1-\hat{y}_i)\right]. \tag{2.39}$$

The LSTM is a type of recurrent neural network (RNN) with the capability to receive feedback from multiple hidden layers, unlike traditional RNNs, which only utilize one hidden layer. Unlike RNNs, LSTM has the ability to manage memory blocks, enabling it to remember input patterns at the beginning of a sequence (Hochreiter and Schmidhuber, 1997). This memory retention is especially advantageous for predicting temporal data, as seen in PdM applications.

The architecture of the LSTM neural network includes several memory blocks, each comprising a memory cell and three types of gateways, illustrated in Figure 18. In this structure, data from the previous layer x_t and the prior time step h_{t-1} serve as inputs. The flow of information is governed by three

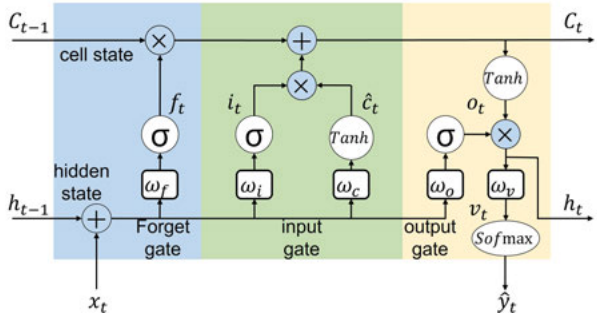


Figure 18. The architecture of LSTM and its memory blocks

gates: forget, input, and output. Each gate utilizes a sigmoid activation function, scaling values between 0 and 1, and performs elementwise multiplication with the input data vector. Values close to 0 indicate disregarded parts of the data, while values close to 1 signify active parts for making predictions. A crucial component is the forget gate in conjunction with the internal state c_t where the current internal state is memorized and multiplied with the forget gate at the next time step. This enables the LSTM to learn when to remember old time information, when to forget, when to use new data, and how to generate output by combining the old memory with new input.

The equations of the LSTM scheme are given as follows.

$$\begin{aligned} f_t &= \sigma(w_f[h_{t-1,x_t}] + b_f) \\ i_t &= \sigma(w_i[h_{t-1,x_t}] + b_i) \\ \hat{c}_t &= \tanh(w_c[h_{t-1,x_t}] + b_c) \\ c_t &= f_t \otimes c_{t-1} + i_t \otimes \hat{c}_t \\ o_t &= \sigma(w_o[h_{t-1,x_t}] + b_o) \\ h_t &= o_t \otimes \tanh(c_t), \end{aligned} \tag{2.40}$$

where f_t, i_t, and o_t present the forget, input, and output gates, respectively, w and b are the corresponding weight and bias parameters for these gates, c is used for cell state, and h is the hidden state. Moreover, σ represents the sigmoid activation function, and $\otimes$ indicates the Hadamard product. The standard LSTM is made by the number of sequential blocks. Each of them is the same as Figure 18 and sequentially connected. At the final block, a softmax function is used as a final activation function, given as follows for the prediction.

$$\hat{y}_t = \frac{1}{D}\sum_{i=1}^{D}\left(-\log\left(\frac{e^{\left(\left(w_v^{(i)}\right)^T h_t^{(i)} + b_v^{(i)}\right)}}{\sum_{j=1}^{s} e^{\left(\left(w_v^{(j)}\right)^T h_t^{(i)} + b_v^{(j)}\right)}}\right)\right). \tag{2.41}$$

2.2.3 Gradient Descent

Gradient descent (GD) is a widely used optimization algorithm for iteratively minimizing loss functions in ML models. This section covers three standard variants, full GD, stochastic gradient descent (SGD), and stochastic variance-reduced gradient (SVRG). GD minimizes $f(h(x,w),y)$ by updating the parameters in the opposite direction of the gradient $\nabla_w f(h(x,w),y)$, using the entire dataset $\mathscr{D} = (x_i,y_i)_{i=1}^{D}$ in each iteration (Ruder, 2016). The steps are computed as

$$w_{n+1} = w_n - \frac{\alpha}{D}\sum_{i=1}^{D}\nabla_w f(h(x_i,w_n),y_i), \tag{2.42}$$

where n is the iteration index, and α is the step size. However, GD becomes computationally expensive when dealing with large neural networks and extensive datasets. To alleviate this issue, SGD substantially reduces the computational cost by processing only one data sample per step

$$w_{n+1} = w_n - \alpha\nabla_w f(h(x_i,w_n),y_i), \tag{2.43}$$

where i is a randomly selected integer with a uniform distribution from 1 to D. Another algorithm in the SGD category is SVRG which operates with two nested loops (Xiao and Zhang, 2014). In the outer loop, the full gradient of the entire function, $\nabla f(w_n^t)$, is calculated. However, this can be computationally expensive and is avoided whenever possible. In the inner loop, the update step is iteratively computed as follows.

$$w_{n+1} = w_n - \alpha\left[\nabla f^i(w_n) - \nabla f^i(w_n^t) + \nabla f(w_n^t)\right], \tag{2.44}$$

where $\nabla f^i(w_n)$ represents the stochastic gradient computed based on a randomly selected data label and $\nabla f^i(w_n^t)$ denotes the stochastic gradient computed over the entire dataset. This iteration is specific to a single device, and its fundamental concept lies in utilizing stochastic gradients to estimate the gradient change from point w^t to w, rather than directly estimating the gradient itself. The pseudocode for SVRG is provided in Algorithm 2.2.

Algorithm 2.2: SVRG

Parameters: m= number of stochastic steps, α = stepsize.

for $n = 0, 1, 2 \ldots$ **do**

  Compute and store $\nabla f(w^t) = \frac{1}{D} \sum_{i=1}^{D} \nabla f^i(w^t)$

  Set $w = w^t$

  for $t = 1$ *to* m **do**

    Randomly select $i \in \{1, 2, \ldots, D\}$

    $w = w - \alpha \left[\nabla f^i(w) - \nabla f^i(w^t) + \nabla f(w^t) \right]$

  end

  $w^{t+1} = w$

end

2.2.4 Distributed Learning

Let's look at the collaborative edge-fog-cloud structure in Figure 15. At the edge level, there are K edge devices connected to a parameter server (PS) at the fog level. Each edge device carries its own dataset $\mathscr{D}_k = (x_i, y_i)_{i=1}^{D_k}$, and collectively, they constitute the global dataset $\mathscr{D} = \bigcup_{i \in K} \mathscr{D}_i$. Now, the goal is to train a model that minimizes the empirical loss on this global dataset, which is essentially what distributed ML aims to achieve.

$$w^* = \arg\min_w \frac{1}{D} \sum_{k=1}^{K} \sum_{i=1}^{D_k} f(h(x_i \in \mathscr{D}_k, w), y_i \in \mathscr{D}_k). \tag{2.45}$$

The straightforward way to tackle this minimization problem is by collecting each dataset $\mathscr{D}_k$ at the PS to create the global dataset $\mathscr{D}$ and then training the model with GD (Verbraeken et al., 2020). However, this approach comes with issues like privacy concerns and high communication load. An alternative method is Distributed Gradient Descent, where local training is performed at the edge devices, and the gradients are communicated to the PS for aggregation. The local gradient at each edge device k is computed using an initial model received through broadcasting from the PS.

$$\nabla_k^{(n)} = \sum_{i=1}^{D_k} \nabla f(h(x_i \in \mathscr{D}k, w^n), y_i \in \mathscr{D}k). \tag{2.46}$$

The edge devices then send these gradients to the PS, where the model update takes place

$$w^{n+1} = w^n - \alpha \sum_{k=1}^{K} \nabla_k^{(n)}. \tag{2.47}$$

Another approach to solving distributed optimization problems is using FL algorithms, which reduce the communication load compared to distributed GD. FL has the capability to perform multiple training steps on edge devices before communicating a model update to the PS (McMahan et al., 2017). FL involves two steps: local training and global aggregation. In local training, an edge device downloads the model from the PS through broadcasting and computes an updated model using its local data. The PS then collects these updated models, primarily by averaging. Specifically, after broadcasting the global model $w^{(t)}$, the local devices train new local models $w_k^{(t)}(E)$ by running E iterations of GD as follows.

$$w_k^{(t)}(e+1) = w_k^{(t)}(e) - \alpha \frac{1}{D_k} \sum_{i=1}^{D_k} \nabla f(h(x_i, w_k^{(t)}(e)), y_i). \tag{2.48}$$

The new global model at the PS is computed by averaging all received local models $w_k^{(t)}(E)$.

$$w^{t+1} = \frac{1}{K} \sum_{k=1}^{K} w_k^{(t)}(E). \tag{2.49}$$

There are various FL algorithms for distributed optimization, some of which are evaluated in (Nilsson et al., 2018), including FedAvg, FedProx, and CO-OP. FedAvg runs the training task on edge devices, sharing an overall model with the central server as an average of all parameters. CO-OP introduces an asynchronous approach, merging any received edge model with the global model. Unlike FedAvg, merging is done using a weighting scheme based on the models' age difference. FedProx, similar to FedAvg but with small changes, improves performance and accommodates heterogeneity better, acknowledging the diverse limitations of edge devices (Konečnỳ et al., 2016).

It is crucial to highlight that FedAvg at the PS does not necessitate knowledge of any specific local model. Only the model averaging needs to be calculated and sent back to the edge devices. Instead of computing this average at the PS hardware, it can be done in the wireless channel using over-the-air computation, owing to the waveform superposition property in multi-access communication channels. This method significantly reduces communication latency compared to conventional methods.

2.3 Over-the-Air Distributed Machine Learning

In this section, we provide a detailed explanation of OACC. When multiple edge devices transmit their models simultaneously over the same frequency band in an analog manner, the PS can receive the aggregated model. Consider a network

with K edge devices, each transmitting their model w_k to the PS. The received signal r at the PS is given by the expression

$$r = \sum_{k=1}^{K} h_k w_k + z, \tag{2.50}$$

where h_k represents the channel state information, and $z \sim \mathscr{CN}(0, \sigma_z^2 \mathbf{I})$ is an AWGN. To rebuild the average model at the PS, one solution is to have user devices pre-equalize their channel. Instead of directly transmitting w_k, they send w_k/h_k.

$$r = \sum_{k=1}^{K} h_k(\frac{w_k}{h_k}) + z = \sum_{k=1}^{K} w_k + z. \tag{2.51}$$

This approach relies on accurate channel state estimation for each edge device. One method involves calculating the constant multiplier $c_k = h_k(0)/g_k(0)$, where $h_k(0)$ and $g_k(0)$ represent the uplink and downlink channel gains at time 0 for edge device k respectively. Assuming c_k stays constant, the uplink channel can be determined using the downlink channel gain measured through the PS broadcast. However, this method is suitable only for stationary nodes. In dynamic scenarios like cellular networks and vehicle communication, alternative solutions are needed. This thesis does not delve into the details of channel state estimation. For more information on this topic, refer to (Goldenbaum and Stanczak, 2014).

2.3.1 Effective Noise and Power Control

Power control is a concern in OACC. When an edge device faces a deep fade, the value of h_k becomes very small. Consequently, a significant amount of power is needed for pre-equalization at the edge device. In a practical application, there is also a limit on the peak power available at the edge devices. The received signal at the PS is represented by gain blocks in Figure 3, enabling the PS to estimate the aggregated model through the expression

$$r = \sum_{k=1}^{K} \frac{h_k p_k w_k}{\sqrt{\eta}} + \frac{z}{\sqrt{\eta}}, \tag{2.52}$$

here, η serves as a post-transmission scaling factor. To fulfill the aggregation requirements of OACC distributed learning, edge devices follow the channel inversion rule to determine the instantaneous transmit power

$$p_k = \frac{h_k^H}{|h_k|} \cdot \frac{\sqrt{\eta}}{|h_k|}, \tag{2.53}$$

where H represents the Hermitian transpose. With this definition, the received signal can be expressed as

$$r = \sum_{k=1}^{K} w_k + \frac{z}{\sqrt{\eta}}. \tag{2.54}$$

The received Signal-to-Noise Ratio (SNR) of the PS aggregated signal can be given by

$$\text{SNR} = \mathbb{E}\left\| \frac{1}{K} \sum_{i=1}^{D} \frac{\sqrt{\eta} \sum_{k \in K} w_k^i}{z^i} \right\|^2 = \frac{\eta \mathbb{E} \left\| \sum_{k \in K} w_k \right\|^2}{\sigma_z^2 K^2}. \tag{2.55}$$

It is evident that the received SNR is consistent across all users due to devices with weaker channels compensating by transmitting at higher powers. In various studies, devices with significantly weak channels are often excluded from training due to their inability to pre-equalize their channels.

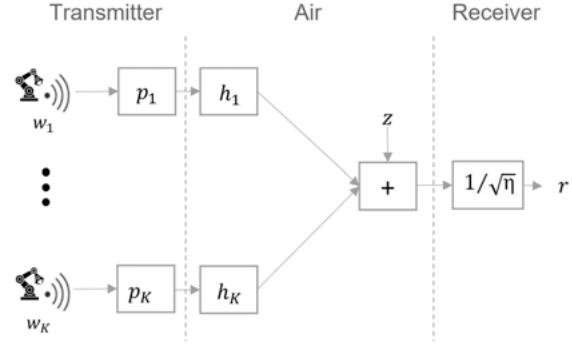


Figure 19. Analog transmission Over-The-Air aggregation

The conventional channel inversion technique proves suboptimal unless all devices possess the capability to invert their channels. Extensive research has been conducted to optimize parameters such as η and p_k with the aim of enhancing model convergence in fading channels within the context of OACC. For a more comprehensive understanding, please consult reference (Cao et al., 2020).

3 Distributed Event Triggering Algorithm in WNCS

This chapter provides a summary of the contributions made in Papers I, II, and III (Bemani and Björsell, 2020, 2021b,a). It successfully achieves *research objectives I* and *II* and addresses *research questions 1*, *2*, *3*, and *4* (*RQ1*, *RQ2*, *RQ3*, and *RQ4*) introduced in Section 1.4.

3.1 Introduction

System of systems (SoS) integrates different systems to enhance capabilities. Systems of systems engineering is a method increasingly used for solving challenges in various fields like industry, transportation, energy, and global communication networks (Ishizaka and Nishimura, 2021). For instance, in transportation, vehicle platooning is a type of SoS. In industry, especially in PdM applications, series machines that rely on each other's output are another example of SoS. The idea is to merge WNCS, CPS, and SoS for real-time control of such systems over wireless networks. To explore this area, the first step is to create an efficient testbed that enables testing, implementation, and troubleshooting of various control algorithms for a wireless CPSoS. A platoon of vehicles serves as the testbed, representing a wireless CPSoS. The vehicles within the testbed possess the capability of V2I and V2V communication.

In order to reduce communication load among these vehicles, an event triggering algorithm is explored. However, the control performance becomes uncertain in an industrial environment with a high probability of packet dropping. Therefore, as the next step, a distributed event triggering algorithm in the state feedback controller for multi-agent systems is proposed. This involves applying distributed event-based state estimation methods to design a new event triggering algorithm named PET for multi-agent systems while maintaining satisfactory control performance, even in conditions with a high probability of packet drops. The subsequent sections provide a detailed explanation of the testbed, PET algorithm, and its evaluation under bursty packet drops.

3.2 Wireless CPSoS testbed

3.2.1 Testbed Description

To illustrate a category of SoS, contemplate the control of a platoon of vehicles, depicted in the Figure 20. The control objective is to synchronize the trajectories of the followers with the leader, ensuring a constant spacing while avoiding collisions.

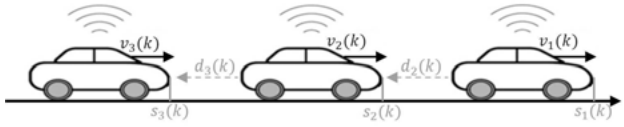


Figure 20. Platooning vehicles in leader-follower configuration with distance d_i

PiCar-S vehicles from SunFounder Company are employed as miniature vehicles for the physical hardware prototype in this testbed. The central component of the PiCar-S is the Raspberry Pi, complemented by features such as the ultrasonic obstacle avoidance module, the line follower module, and the inertial measurement unit for assessing vehicle acceleration. As miniature vehicles proposed for integration into this testbed, they are designed to communicate with the overarching system controller through V2I communication. The Raspberry Pi is assigned four primary functions, regulating DC gear motors for speed control, overseeing servo motor drive for steering angle control, processing sensor data (speed, IMU, ultrasonic, and IR signals), and facilitating Bluetooth and Wi-Fi communication.

The architecture of the testbed is delineated in Figure 21, elucidating its sub-systems and interfaces. Constituting both cyberspace and physical space, the cyberspace component encompasses a Kalman Filter that facilitates state estimation $[v_i, s_i]$ through sensor data. For line tracking, each Raspberry Pi features a decentralized control loop, determining the steering angle set point for individual vehicles. Simultaneously, a centralized upstream control loop in cyberspace leverages an MPC controller to calculate vehicle velocities based on distance, ensuring controlled spacing between the leader and follower cars within a train-vehicle cooperative control algorithm. For clarity, Figure 22 illustrates the control loop of the testbed. Real-time position feedback is facilitated by an Indoor Positioning System (IPS) utilizing Bluetooth beacons and IMU data, the details of which will be explained later.

In each vehicle, real-time sensor data, including position, ultrasonic distance, infrared, and speed, is wirelessly transmitted to a centralized controller through Raspberry Pi. MATLAB processes the received data and executes control algorithms within the Simulink environment. The implementation involves Kalman Filter and Model MPC in MATLAB/Simulink. Given that MPC necessitates a system model, the state space representation for a two-car following system is as follows.

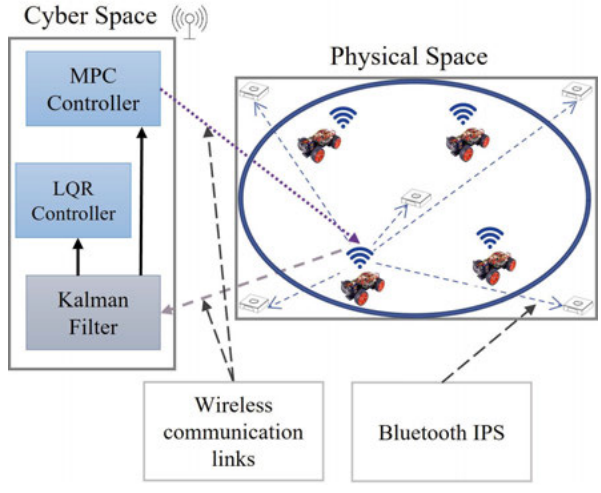


Figure 21. Testbed System Architecture illustrating sub-systems and communications

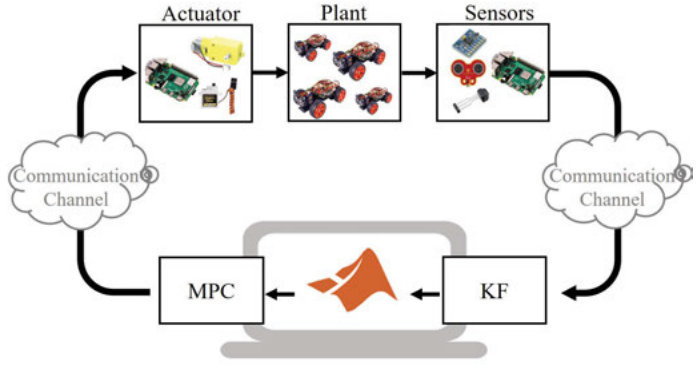


Figure 22. Control loop of the testbed

$$\begin{bmatrix} \dot{v}_1(t) \\ \dot{v}_2(t) \\ \dot{s}_1(t) - \dot{s}_2(t) \end{bmatrix} = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 1 & -1 & 0 \end{bmatrix} \begin{bmatrix} v_1(t) \\ v_2(t) \\ s_1(t) - s_2(t) \end{bmatrix} + \begin{bmatrix} 1 & 0 \\ 0 & 1 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} a_1(t) \\ a_2(t) \end{bmatrix}, \tag{3.1}$$

Here, s_i represents the absolute position, v_i is the velocity, and a_i is the acceleration of vehicle i. It is clear that the control algorithm on the centralized controller must effectively manage communication imperfections. The approach employed aligns with the theory on WNCS discussed in Section 2.1.1.

3.2.2 Cooperative Driving Scenario

The proposed testbed serves as an open system of systems platform for the evaluation of cooperative driving algorithms, spanning from physical entities to cyber components. One such cooperative experimental case implemented on the testbed is the leader-follower scenario. The centralized controller handles four conditions related to the distance between two cars within the platoon configuration

1. If the sensor-measured distance exceeds one meter, the miniature vehicle identifies itself as the leader and begins moving at 0.9 of its full speed.
2. If the distance is less than one meter but greater than 10cm, the LQR controller is activated to maintain a 10cm distance from the lead miniature car.
3. If the distance falls between 10cm and 6cm, the speed follows a linear function of the distance.
4. If the distance becomes less than 6cm, the miniature vehicle comes to a stop.

Figure 23 exhibits a photograph of the testbed, featuring the trajectories of the vehicles and providing a representation of their real-time positions.

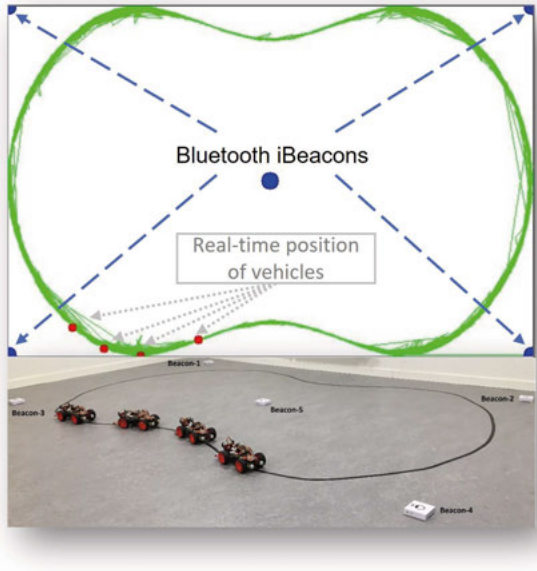


Figure 23. Photograph of the testbed

3.2.3 Indoor Positioning System

In this testbed, we introduce a new IPS utilizing multiple sensing techniques, including IMU sensors and iBeacons built on Bluetooth Low Energy. Five iBeacons are strategically placed as landmarks with precise positions covering the testbed surface. Each vehicle receives the signal strength (RSS) value from the broadcasted iBeacons, and this value is used to calculate the distance between the vehicle and each iBeacon. Position estimation is accomplished using a particle filter. Additionally, IMU data, which measures acceleration and heading direction, is employed for positioning. While IMU accuracy diminishes over time, sensor fusion combines both methods to enhance position estimation accuracy. The block diagram of the proposed IPS is presented in Figure 24.

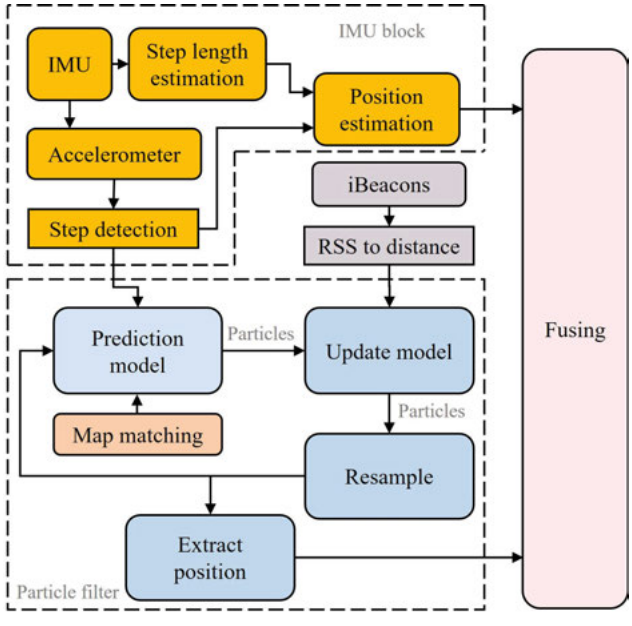


Figure 24. Position tracking system

The entire algorithm, which incorporates particle filtering, map matching, IMU positioning, and distance calculation to iBeacons, has been implemented on Raspberry Pi. The accelerometer samples acceleration data and forwards it to the step detection component. The step detection algorithm triggers the prediction model upon detecting a step. Simultaneously, the RSS to distance component processes RSS values from iBeacons ranging and converts them into distances. The particle filter utilizes the step detection as a trigger for the prediction model and employs the RSS to distance component to update particles in the model. Additionally, the Map Matching component restricts particle movement in the prediction model. The position is determined by averaging these particles, and

this result is fused with the IMU method to improve accuracy. For more details, please refer to *Paper I*.

3.3 Distributed Event Triggering Algorithm

In the designed testbed, the feasibility of feedback control for a fast dynamic process like vehicle platooning over a wireless network is demonstrated. However, communication ran at the control sampling rate to support the control process. To enable resource saving, event-triggered methods for such multi-agent systems will now be presented, where communication occurs only when needed. Vehicles in the testbed had V2I communication, but in this part, it is assumed each vehicle in the platoon has communication with neighbors through V2V communication. Here, we examine the scenario where each vehicle is capable of measuring only its own position and velocity. Therefore, distance information is acquired only through communication. The communication between each vehicle and its neighbors is illustrated in Figure 25.

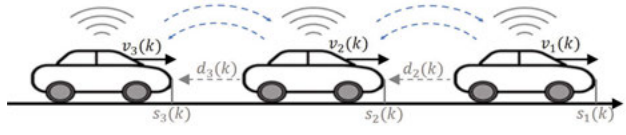


Figure 25. Platooning vehicles with V2V communications

By this assumption, the state space representation of the ith-follower in vehicle platooning, subjected to its interaction with the front vehicle can be obtained as

$$\begin{bmatrix} \dot{v}_i \\ \dot{s}_i - \dot{s}_{i-1} \end{bmatrix} = \begin{bmatrix} 0 & 0 \\ 1 & 0 \end{bmatrix} \begin{bmatrix} v_i \\ s_i - s_{i-1} \end{bmatrix} + \begin{bmatrix} 1 \\ 0 \end{bmatrix} a_i + \begin{bmatrix} 0 & 0 \\ -1 & 0 \end{bmatrix} \begin{bmatrix} v_{i-1} \\ s_{i-1} - s_{i-2} \end{bmatrix}. \tag{3.2}$$

Vehicle platooning with V2V communication is an example of a multiple CPSs connected over a communication network, as illustrated in Figure 26. These interconnected systems are designed for cooperative control. In this setup, the dynamics of agent i's are interconnected with all or a subset of other agents

$$x_k^i = A_i x_{k-1}^i + B_i u_{k-1}^i + w_k^i + \sum_{h \in \mathbb{N}_N, h \neq [i]} N_h \check{x}_{k-1}^h \,, \tag{3.3}$$

where N_h is the interaction matrix between the agents in the multi-agent system, $x_k^i \in \mathbb{R}^{n_x}$ denotes the state, $\check{x}_{k-1}^h \in \mathbb{R}^{n_x}$ denotes the remote state prediction of other agents in agent i, $u_k^i \in \mathbb{R}^{n_u}$ denotes the input, $w_k^i \in \mathbb{R}^{n_x}$ denotes process noise, and A_i, B_i denote the dynamic system parameters for agent i. A discrete-

time linear process with Gaussian noise is considered for each agent and the interaction between them.

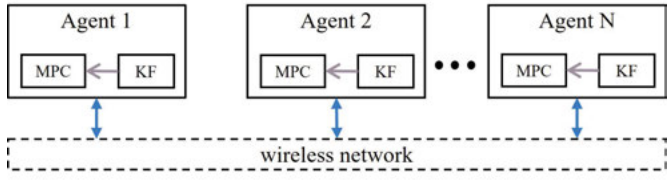


Figure 26. Representation of multiple CPSs interconnected through a communication network

The network bandwidth is shared by multiple agents in such system, each agent should utilize the communication resource only when it is essential. Event triggering algorithms can notably minimize network usage, but they may compromise control performance in industrial environments prone to packet drops. This section introduces a new event-triggering algorithm named PET that incorporates DEBSE, providing control convergence in environments with a high probability of packet drops.

3.3.1 Architecture Design: DEBSE and PET

This section extends a previously developed framework for DEBSE, as outlined in earlier works (Trimpe and Baumann, 2019; Muehlebach and Trimpe, 2017; Trimpe, 2017). The application of this framework is demonstrated in Figure 26. The fundamental concept behind DETSE is to use model-based predictions from other systems, eliminating the necessity for constant data transmission between agents. Updates are transmitted only when these predictions deviate significantly, such as in the presence of disturbances or packet drops. Figure 27 depicts the primary architecture of DEBSE and PET, along with its components for agent i in the overall system.

The event triggering decision relies on two parallel algorithms. The first algorithm involves comparing the state prediction of agent i with its own state estimation. The second algorithm compares the state prediction with the state estimation of other agents that communicate with agent i (state prediction j and state estimation j). The first event triggering algorithm assesses the accuracy of prediction without knowledge of whether a packet drop has occurred. Conversely, the second event triggering algorithm evaluates the difference between the estimation and prediction of other agents interacting with agent i in their processes. This allows checking if the discrepancy between estimation and prediction has increased, indicating a packet drop in the previous broadcast of agent i. Now, an event trigger is needed to improve the prediction of agent i in other agents. State estimators and predictors, such as those of Kalman filter explained in Section 2.1.3, are employed to forecast the states of all or a subset

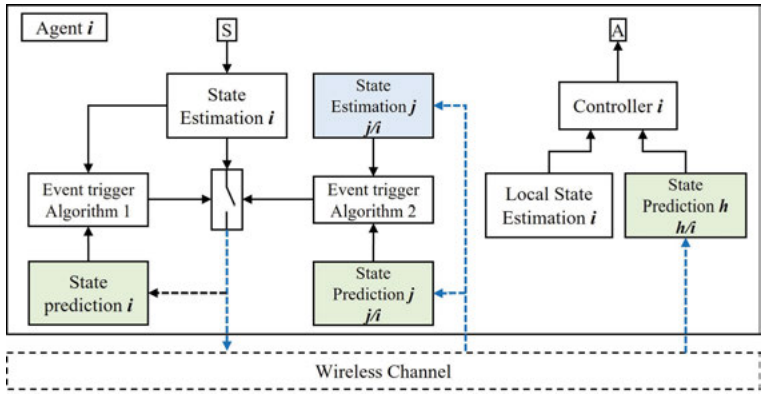


Figure 27. PET based on DEBSE for each agent *i*. The control decision of Agent *i* is determined by the local state estimation and prediction of either all or subset of other systems. Each agent sends an update (Event Trigger) to all other agents when the prediction of its own state (State prediction *i*) deviates significantly from the truth (Local state estimation *i*) or when the predictions of other agents (State prediction *j*) deviate significantly from their estimations (State estimation *j*) in agent *i*.

of agents, relying on the dynamics models of the agents.

- Local State Estimation (Agent *i*): Agent *i* estimates its states using a KF state estimator, incorporating its dynamics model, measured sensor values, and state predictions from interacting agents.
- Local State Prediction (Agent *i*): Agent *i* predicts its states with a KF state predictor, considering potential packet drops. It relies on its dynamics model, the last buffered state values from the previous ET mechanism, and state predictions from interacting agents.
- State Estimation (Agent *j*/*i*): Using a KF state estimator, agent *j*'s states are estimated with consideration for potential packet drops. It relies on its dynamics model and the state values received by agent *i* in the previous ET broadcast of agent *j*. In a distributed multi-agent system, agent *j*'s dynamic model is influenced by the accessible state values of agent *i* in its estimation.
- State Prediction (Agent *j*/*i*): It is similar to the state estimator of agent *j*, but excluding state values of agent *i* due to unclear access caused by potential packet loss, is only based on the dynamic model of agent *j*.

For information about the estimators and predictors in the proposed PET algorithm, as well as the mathematics behind them, please refer to *Paper II*. Additionally, it is important to highlight that the proposed PET algorithm involves a modest increase in communication instances between agents. It is presumed that when an event is triggered in one agent, its states are disseminated

not only to the agents with which it directly interacts but also to those receiving interactions from it. This minimizes additional communications, as sending states functions like group broadcasting. The extra communication load can be estimated based on the multi-agent system's topology and interactions between agents.

3.3.2 Formulate Event Triggering

In this part, PET formulation for each agent is established by defining two estimation costs. The name of predictive state estimation is introduced, where local state estimation is shared among agents through wireless communication. Figure 28 illustrates the configuration of PET for one agent, simplified to its essential components for detailed analysis. In this setup, Agent i, referred to as the sensor agent, transmits its local state estimation wirelessly with a probability of packet drop when a positive triggering decision is made ($\gamma_k^i = 1$). Agent j, known as the remote agent, represents any agent in the multi-agent system and receives these transmissions with a probability of $\lambda_{i/j}$. Agent j requires information from Agent i to address the local control problem. Additionally, Agent j broadcasts its local state estimates when an event occurs, and Agent i receives the data with a probability denoted by $\lambda_{j/i}$. These received data help assess the accuracy of Agent j's state estimation through Agent i in the second event-triggering algorithm. The packet drop model between Agent j and i is characterized by a random Bernoulli variable $v_k^{j/i}$.

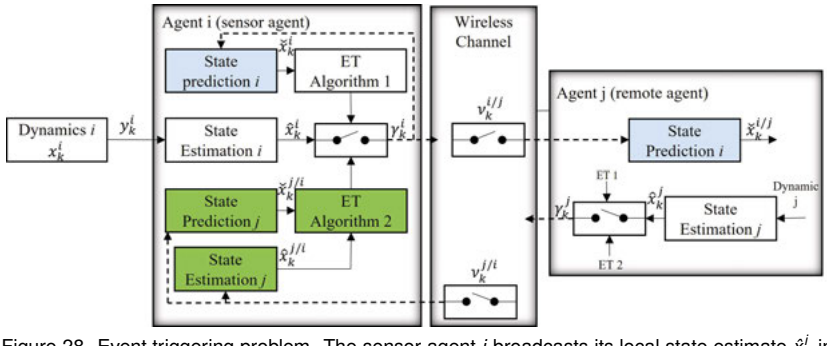


Figure 28. Event triggering problem. The sensor agent i broadcasts its local state estimate $\hat{x}_k^i$ in case of a positive triggering decision ($\gamma_k^i = 1$) and is received by agent j with the probability of $\lambda_{i/j}$.

The two estimation costs in PET are utilized in an optimization problem for the control and communication decision.

The estimation cost E_1^i quantifies the difference between the state estimation and prediction of agent i, expressed as the quadratic norm

$$E_{1|k}^i \triangleq \|\hat{x}_k^i - \breve{x}_k^i\|^2. \tag{3.4}$$

Similarly, the estimation cost E_2^j measures the disparity between the state estimation and prediction of agent j in i, given by

$$E_{2|k}^j \triangleq \|\hat{x}_k^{j/i} - \check{x}_k^{j/i}\|^2. \tag{3.5}$$

The estimation costs, E_1 and E_2, are associated with the first and second event triggering algorithms, respectively, both of which generate the PET algorithm. $E_{1|k}^i$ is the estimation cost for agent i, and $E_{2|k}^i \triangleq \sum_{j\in\mathbb{N}_N\setminus[i]} E_{2|k}^j$ is the total remote estimation cost of other agents through agent i. The detailed expression of the solution to the optimization problem and the mathematical design of these two parallel events are presented in *Paper II*.

3.3.3 Results of Evaluation and Discussion

To assess the effectiveness of the proposed PET algorithm, we apply it to a vehicle platoon control scenario, treating it as a synchronization challenge in a wireless networked dynamical system. In this setup, the initial vehicle serves as the platoon leader, and a group of follower vehicles (two in total) communicate through a network. The control objective is to maintain a desired distance between vehicles, ensuring that the dynamics of the followers converge toward those of the leader. An LQR is designed as a decentralized controller for each vehicle. The architecture of the vehicle platoon with the proposed event triggering algorithm is shown in Figure 29. When an event is triggered in one vehicle, the data is sent to the rear vehicle that interacts with it and the front vehicle that gets interaction from it.

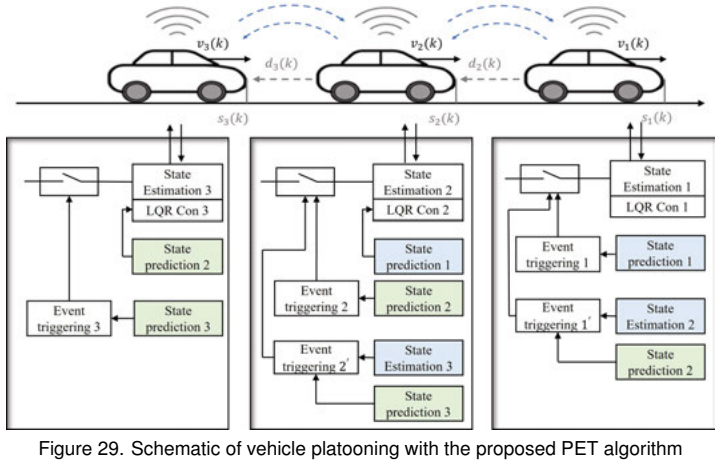


Figure 29. Schematic of vehicle platooning with the proposed PET algorithm

The vehicle platooning, incorporating the PET algorithm, is simulated in the TrueTime simulator, a Matlab/Simulink-based tool, with three vehicles.

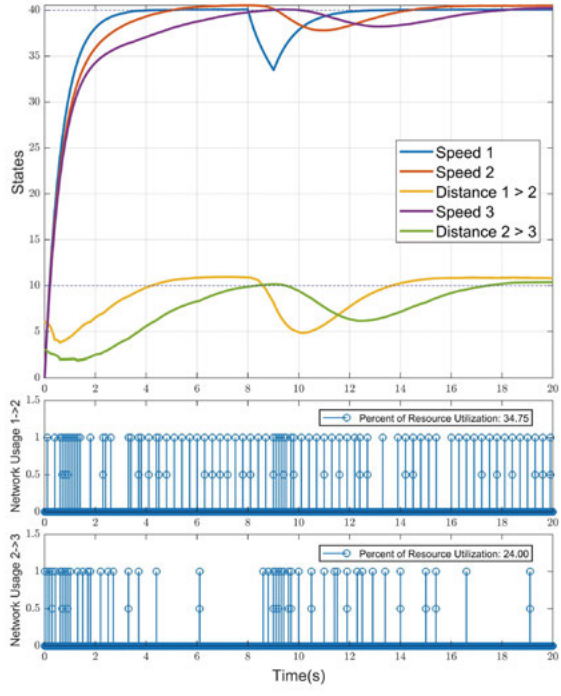


Figure 30. Vehicle platooning with the proposed PET algorithm. From top to bottom: states of vehicles in the platoon (speeds in km/h, relative distance in meters), and network usage between vehicles $1 \rightarrow 2$ and $2 \rightarrow 3$.

Each vehicle's dynamic process is modeled in Simulink, followed by internal communication facilitated by TrueTime's kernel. TrueTime utilizes a modified WLAN (802.11b) protocol without the exchange of acknowledgments during transmission. This part aims to showcase the primary advantage of the PET algorithm in addressing packet drop effects, in contrast to the standard event triggering algorithm, within a multi-agent system.

In this simulation, the desired inter-vehicle spacing is set to 10 meter as a reference. the platoon simulates for 20 seconds. The first vehicle becomes the leader, reaching a constant speed of 40 km/h. An external disturbance, such as reaching a hill, affecting the leader's speed, starting at $t = 8$, and the leader's controller compensates this disturbance, impacting the followers' dynamics. A challenging scenario with a random packet drop rate of 58% is considered, under which conventional ET algorithm fail to ensure system convergence, leading to collisions between vehicles. However, the results obtained with PET (Figure 30) demonstrate that, even in the presence of a 58% packet drop rate, the proposed event triggering algorithm successfully converges the followers to

the leader's motion and maintains the desired inter-vehicle distance.

The PET algorithm is also assessed in a scenario of correlated burst packet drops, simulating the platoon entering a tunnel with a jammer causing a wireless cyber attack. In this situation, the communication channel between vehicles is impacted by a high probability of burst packet drops (68.5%), occurring with a delay from the front vehicle to the tail. Despite correlated burst packet drops in the communication channel, the proposed PET algorithm ensures that each following vehicles follow the leader's speed while maintaining the desired inter-vehicular distance. This underscores the effectiveness of PET in achieving consensus within the platoon, even in the challenging environment of high probability correlated burst packet drops.

The conclusion drawn is that the proposed PET algorithm slightly enhances network resource utilization compared to ET, yet it preserves effective control performance in multi-agent consensus problems, even in challenging packet drop conditions.

4 Collaborative Predictive Maintenance with Federated Learning

This chapter provides a summery of the contribution made in Paper IV (Bemani and Björsell, 2022). It successfully achieves *research objective III* and *IV* and addresses *research questions 5* and *6* (*RQ5* and *RQ6*), introduced in Section 1.4.

4.1 Introduction

Industry 4.0 has revolutionized the way we build industrial assets, transforming them into compact, precise, and interconnected entities. This transformation has also turned modern industrial assets into significant sources of data, providing valuable insights for process optimization specially PdM applications. However, the sheer volume of data collected from machines, often considered as edge devices, presents challenges in terms of cost, latency, and privacy when transmitting to the cloud for processing. To overcome these challenges, there has been a growing emphasis on leveraging edge computing in PdM applications, aiming to reduce data transmission costs and enhance processing speed. FL emerges as a powerful solution, allowing the creation of models from distributed data across edge, fog, and cloud layers without compromising privacy as discussed in Section 2.2.4. Despite its potential, FL encounters challenges in asset management within the industry, particularly in PM applications. Therefore, this study delves into the realm of distributed ML for PdM applications and introduces two federated algorithms, namely FedSVM for distributed anomaly detection and FedLSTM for distributed RUL estimation. These algorithms empower factories at the fog level to optimize the accuracy of their PdM models without sacrificing privacy. Additionally, we propose an aggregation strategy for collaborative PdM at the cloud, fog, and edge levels. The effectiveness of FedSVM and FedLSTM, along with the collaborative PdM aggregation strategy, are evaluated using the CMAPSS dataset distributed on various edge devices. The dataset is utilized for predicting engines' RUL and performing anomaly detection. Each method is comprehensively explained in the subsequent sections.

4.2 Aggregation Strategy for Collaborative PdM

In collaborative PdM scenarios at the edge level, where different devices and failures are involved, some clients may lack enough computing power for training the overall model. This can lead to a decline in model accuracy and aggregation, causing delays in the training process. Such interruptions in the

collaborative training process are costly for PdM. Hence, a hierarchical model is required for effective collaboration in PdM scenarios spanning edge, fog, and cloud levels.

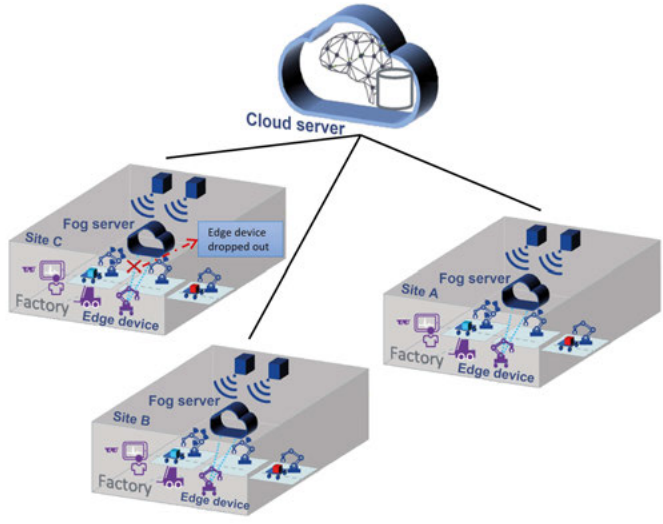


Figure 31. Collaborative PdM scenario at the edge, fog, and cloud level.

The collaborative PdM approach at the edge, fog, and cloud levels is depicted in Figure 31. Edge devices within each factory contribute to creating a local model at the fog level. Subsequently, a global model is constructed from the parameters of these local models in the fogs. This facilitates the utilization of the globally trained model for PdM projects by different companies, even if they are geographically separated. The global model benefits from a broader range of experiences in various failure categories.

Algorithm 4.1 provides a comprehensive description of federation on the edge, fog, and cloud layers. Consider a wireless multi-agent network supported by FL between the fog server and N distributed edge devices at the factory level. The fog server is directly connected to the cloud server through a wireless link with the nearest base station. Each fog server in a factory is equipped with computational resources to offer communication and computation services to the edge devices, which could be similar assets within a factory site. These edge devices communicate with the fog server for FL tasks via a wireless link. The primary analysis in the cloud and fog layers is based on the FedAvg Algorithm. The training process unfolds periodically, comprising an unspecified number of communication rounds. In this context, T_G denotes the count of global communication rounds between cloud and fog servers, while T_l^j signifies the number of local communication rounds between edge devices and the fog server

across various factories. Each local communication round consists of a varying number of iterations on edge devices E, referred to as local epochs.

Algorithm 4.1: Federation on the edge, fog, and coud

Cloud server executes:
initialise w_0
for *each round* $t = 0, 1, \ldots, T_G$ **do**
  for *each factory site* $j \in M$ **do**
    $w_{fog,j}^{t+1} \leftarrow$ Fog server executes(j, w_{cloud}^{t})
  end
  $w_{cloud}^{t+1} \leftarrow \sum_{j=1}^{M} A_j w_{fog,j}^{t+1}$
end

Fog server executes(j, w_{fog})**:**
$h_j \leftarrow$ stepsize
for *each round* $t = 0, 1, \ldots, T_l^j$ **do**
  for *each edge device* $i \in N$ **do**
    $w_{i,j}^{t+1} \leftarrow$ Edge device uppdate(i, w_{fog}^{t})
  end
  $w_{fog,j}^{t+1} \leftarrow \sum_{i=1}^{N_j} \frac{D_i^j}{D^j} w_{i,j}^{t+1}$,
end
$w_{fog,j}^{t+1}$ will be returned to cloud server by request

Edge device uppdate(j, i, w)**:**
$\{\mathscr{D}_i\}_{i=1}^{N} =$ data partition
for *each local epoch k from 1 to E* **do**
  $w_{i,j}^{t+1} = w_{i,j}^{t} - \eta \nabla f_i^j \left(w_{i,j}^{t}\right)$
end
return $w_{i,j}$ to j^{th} fog server

4.3 Distributed Anomaly Detection: FedSVM

4.3.1 Algorithm

FedSVM leverages the SVM algorithm as a supervised machine learning tool for analyzing PdM data in a classification task on the edge, fog, and cloud levels. The FedSVM approach involves initially identifying one or multiple hyperplanes at the fog level to separate and classify failure data from healthy data across all assets within a factory site at the edge level. These hyperplanes are then aggregated at the cloud level using fedAvg. In SVM, the goal is to find

a hyperplane with the largest margin that separates the two groups. However, in FedSVM, the objective is to find the largest margin on a federal hyperplane.

SVM, being a supervised learning method with high time and memory complexity, faces challenges in meeting the fast response requirements of PdM applications. To address this, a lightly modified cost function is introduced for SVM, specifically employing the hinge loss function (equation 4.1). After preparation the dataset from each edge device at a factory site, the faulty and healthy data segments are separated by setting a threshold. For instance, measurement data from a device is assigned a label of $+1$ if its RUL exceeds the threshold, and -1 otherwise. This approach allows the definition of a consistent FedSVM objective function at each level. The hinge loss function used in the modified cost function is defined as

$$\begin{aligned} f(x,w,y) &= \frac{1}{D}\sum_{i=1}^{N} f_i(w) + \lambda ||w||_2^2 \\ f_i(x_i,w,y_i) &= \max(0, 1 - y_i w^T x_i). \end{aligned} \tag{4.1}$$

Here, x_i represents features extracted from the dataset, and y_i is the label associated with the features for a particular edge device at a factory site. Due to the imbalance and non-independent and identically distributed (non-iid) nature of data across edge devices in PdM applications, a regularization term λ has been introduced to the local loss function. For parameter updates, FedSVM is implemented based on federation across edge, fog, and cloud layers (Algorithm 4.1). FedSVM boasts notable advantages, particularly its remarkable speed and swift convergence time. These attributes make it highly suitable and beneficial for online applications.

4.3.2 Dataset Preparation and Distribution

The performance of FedSVM was evaluated using the distributed CMAPSS dataset, which simulates two factory sites, each equipped with 5 edge devices, a fog server, and a global cloud server. CMAPSS, a well-known and benchmarked dataset introduced by NASA (Saxena et al., 2008), records the run-to-failure degradation process of a turbofan engine. The dataset comprises time series data from 21 sensors installed on the turbofan engine, capturing parameters like temperature, pressure, and speed. The dataset is organized into four multivariate time series subsets, namely FD001, FD002, FD003, and FD004, each featuring different conditions and fault modes. These subsets are further divided into training and testing trajectories (Table 1). FD001 is identified as the simplest subset, involving a single condition and fault mode, with 100 training and testing trajectories. In contrast, FD004 presents more intricate scenarios, including six

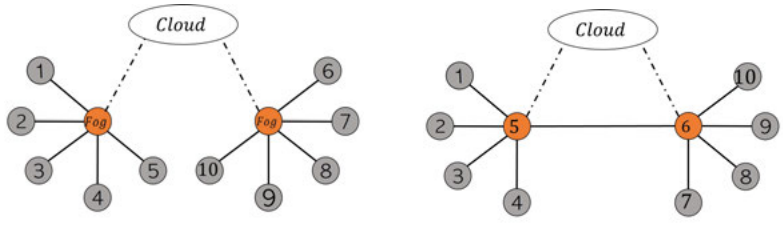
different conditions and two fault modes.

Table 1. Detailed Information of CMAPSS Subsets.

Subsets	Conditions	Fault Modes	Training trajectories	Testing trajectories
FD001	1	1	100	100
FD002	6	1	260	259
FD003	1	2	100	100
FD004	6	2	248	249

The complete CMAPSS dataset is utilized in our experiments to replicate factory site conditions and distribute time-series data among edge devices. Specifically, time series FD001 is partitioned into ten shuffled subsets, distributed across ten edge devices situated in two distinct factory sites, facilitating participation in a collaborative PdM scenario. Each edge device independently trains its FedSVM model using its local time-series data.

The graph connectivity of the edge devices across the two factory sites is illustrated in Figure 32. The FedSVM, based on Algorithm 4.1, operates across different layers of the undirected graph, including edge, fog, and cloud layers. In Figure 32a, a synchronous federated learning method is depicted, where fog and cloud layers wait to collect all parameters from the edge devices. In Figure 32b, an asynchronous configuration is presented, wherein Edge devices 5 and 6 function similarly to fog servers. Each edge device shares its parameters with neighboring edges, acting as fog nodes. The key distinction in the asynchronous configuration lies in the fact that fog agents perform simple federated averaging upon receiving a new parameter, promptly returning the most recent update to the specific edge device from which the last parameter was received.



(a) Synchronous FL. (b) Asynchronous FL

Figure 32. Undirected graph of communication between edge devices and fog servers.

It is assumed that fog servers in different factories can communicate with each other. With this approach, if certain edge devices face resource constraints and cannot participate in fog aggregation, the fog model can continue operating effectively. Parameters are transmitted from fog servers to the cloud upon request from the cloud layer.

4.3.3 Results of Evaluation and Discussion

To assess the performance of the proposed FedSVM, we conducted a benchmark comparison against centralized methods. The FedSVM has linear time complexity positions it as a suitable choice for online PdM applications. Additionally, this characteristic allows its effective use with large datasets and even for a substantial number of edge devices. The final accuracy and overall training runtime serve as quantitative metrics for evaluating the global FedSVM model's performance.

Upon completion of training using the collaborative PdM algorithm, the final model in the cloud is deployed to predict the labeled RUL on the testing subset associated with different time series data. As detailed in the Algorithm section 4.3.1, the RUL prediction is constrained to a binary label, where values less than a specified threshold are labeled as 1, and those exceeding the threshold are labeled as -1. This approach generates two distinct classes, enabling the execution of the FedSVM classifier at the edge devices. The chosen threshold, set at 50 cycles, signifies that maintenance is required for an edge machine when its RUL falls below 50 cycles.

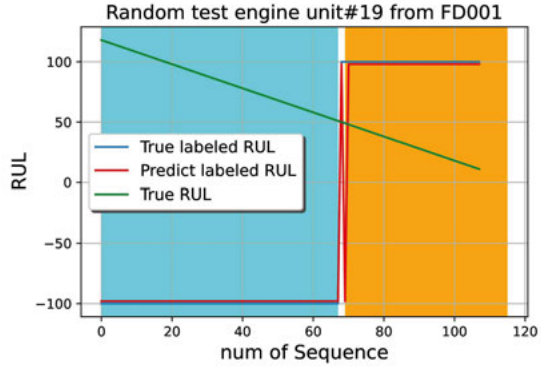


Figure 33. Labeled RUL predictions for the testing engine using the synchronous FedSVM model.

The labeled RUL predictions based on synchronous FedSVM for testing engine unit FD004 is illustrated in Figure 33. The cyan band signifies safe engine operation, while the chrome yellow band indicates the need for maintenance. The green line represents the true RUL, the blue line represents the labeled RUL, and the red line illustrates the predicted labeled RUL. Notably, synchronous FedSVM demonstrates effective detection of maintenance requirements, even for the more complex FD004 engine.

The performance of FedSVM with both synchronous and asynchronous configurations is summarized in Tables 2 and 3, respectively. Remarkably, the proposed FedSVM exhibits an average accuracy of over 85% in both

Table 2. Performance analysis of the synchronous FedSVM.

Optimizer	Evaluation Metrics	Dataset			
		FD001	FD002	FD003	FD004
SGD	Runtime (s)	18.9	43	20.5	45.5
	Final acc (%)	92.5	78.9	92.2	71.7

Table 3. Performance analysis of the asynchronous FedSVM.

Optimizer	Evaluation Metrics	Dataset			
		FD001	FD002	FD003	FD004
SGD	Runtime (s)	19.8	42.5	21.4	43
	Final acc (%)	90	77.5	92.1	83.1

configurations, showcasing its robust performance even when data is distributed.

4.4 Distributed RUL Estimation: FedLSTM

4.4.1 Algorithm

The proposed distributed RUL estimation method, FedLSTM, incorporates modified LSTM models at the edge, fog, and cloud levels. To enable federated learning, model consistency across all levels is essential. Unlike SVM, LSTM involves a substantial number of weight and bias parameters, resulting in increased bandwidth usage during model distribution across levels and introducing computational delays. Therefore, the conventional LSTM model is not well-suited for distributed learning with fully connected neurons in each memory block.

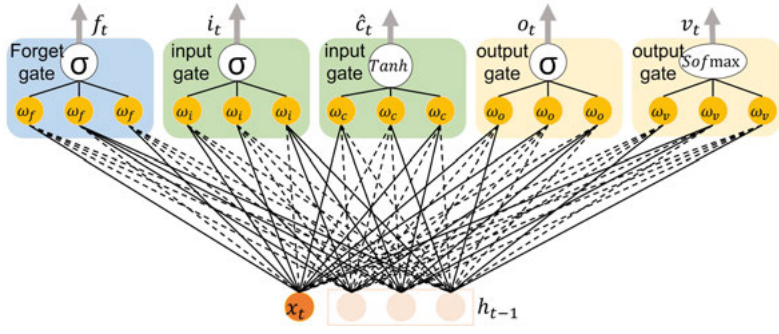


Figure 34. Proposed topology of FedLSTM.

FedLSTM introduces a random topology for synapses at each level, deviating from the traditional fully connected approach. This topology is initially designed in the cloud, and the configuration is then applied to all fog servers and edge devices. Figure 34 illustrates the random FedLSTM topology, where dashed lines indicate removed synapses and solid lines represent retained synapses.

This configured model is subsequently distributed across different layers.

Reducing the number of synapse connections by half results in a modest 30% loss in model accuracy. This reduction operates akin to dropout and regularization techniques, contributing to the model's robustness. Simultaneously, this adjustment leads to decreased bandwidth usage and reduced training time, highlighting a favorable trade-off between accuracy and efficiency.

4.4.2 Dataset Preparation and Distribution

The data preparation process outlined in 4.3.2 is consistently applied to FedLSTM. To emphasize the FedLSTM model's attention on the crucial segment of the data where engine failures are more probable, a clipping operation is implemented on the responses at the 100-cycle threshold. During the training process, inputs with RUL exceeding 100 are adjusted to a fixed value of 100. This ensures a concentrated focus on the critical phase of the data related to potential engine failures. This operation is executed across all edge devices.

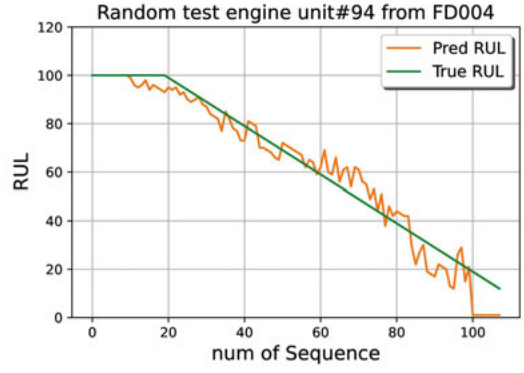


Figure 35. RUL predictions for the testing engine using the synchronous FedLSTM model.

4.4.3 Results of Evaluation and Discussion

FedLSTM was benchmarked on distributed CMAPSS against two representative centralized RUL estimation, namely DCNN (Li et al., 2018) and CNN-XGB (Zhang et al., 2019b). The performance of the global FedLSTM model is assessed using standard root mean square error (RMSE) and scoring factor (SF) as quantitative metrics. SF is an asymmetric function crucial in assessing PdM prediction model, awards higher scores for RUL predictions below the actual value. In PdM applications, prioritizing early predictions over late ones helps mitigate potential adverse outcomes due to delayed predictions.

The RUL prediction result based on the synchronous FedLSTM for testing engine unit FD004 is illustrated in Figure 35. It can be observed that with the FedLSTM model, predictive accuracy is acceptable, especially when edge devices are close to failure. The entire training process for both synchronous and asynchronous FedLSTM has been completed, and the evaluation metrics have been calculated. A comparison with benchmarking results from centralized deep learning algorithms shows that the proposed FedLSTM applications exhibit comparable efficiency to conventional centralized approaches in terms of prediction accuracy. While centralized methods demonstrate higher accuracy, the proposed federated approach is not significantly distant in performance. For more details, please refer to *Paper IV*.

5 Low-Latency Collaborative Predictive Maintenance

This chapter provides a summery of the contribution made in Paper V (Bemani and Björsell, 2023). It successfully achieves *research objectives III* and *IV* and addresses *research questions 6*, *7*, and *8* (*RQ6*, *RQ7*, *RQ8*) introduced in Section 1.5.

5.1 Introduction

In Chapter 4, we focus on the runtime for computation complexity in FedSVM and FedLSTM and the evaluation of the proposed methods was done with this metric. However, despite the rapid advancement of computing speeds, communication latency has emerged as a bottleneck for fast edge learning, especially in time sensitive applications such as PdM. To address this issue, an innovative approach is suggested: analog aggregation over-the-air of model updates transmitted concurrently over wireless channels. However, it is vulnerable to performance degradation due to channel properties like noise and fading. Introducing a method to mitigate the impact of channel noise in FLOACC instead of increasing power of transmission, this approach employs a novel tracking-based stochastic approximation scheme into a standard federated stochastic variance reduced gradient (FSVRG), namely FSVRG-OACC. This effectively averages out channel noise's influence, ensuring robust FLOACC performance without increasing transmission power gain.

5.2 FLOACC: Gradient Distribution

In this section, the focus is on FLOACC, which could be a part of the hierarchical PdM scenario (Figure 31). Here, factory-level agents use analog communication to broadcast updated models over the air, leveraging the advantages of computation through analog communication. As detailed in 2.3, there are two approaches to minimize a global distributed loss function based on FedAVG. The first is model averaging, where each agent minimizes its local loss, transmits model parameters to the PS, and receives an updated model in the next iteration. The second is gradient averaging, centered on transmitting gradients for aggregation, with agents updating their models based on received gradients in subsequent iterations. In FLOACC, only the gradient averaging at the PS is considered. This is because the aggregated gradient is less sensitive in the optimization algorithms compared to the model averaging.

FLOACC provides an efficient multi-access scheme in a low-latency scenario,

which is crucial for applications like PdM that require very fast and real-time task decision making. As discussed in 2.3, the idea of over-the-air distributed ML was inspired by the PS's lack of interest in individual model weight vectors. Instead, The server only requires the average of these weights, conveniently provided by the wireless multiple-access channel as their sum.

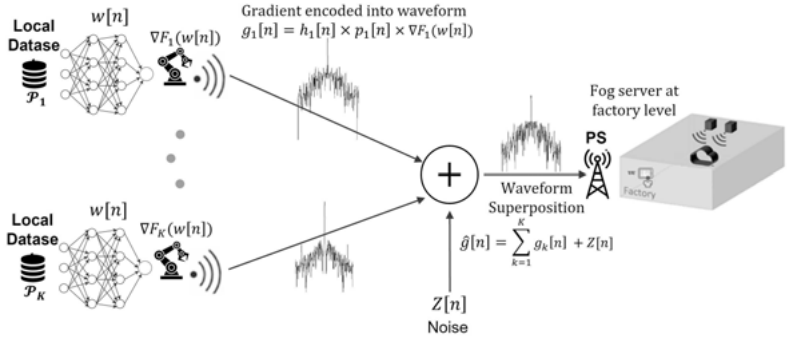


Figure 36. FLOACC scenario at the factory level.

As illustrated in Figure 36, FLOACC facilitates the simultaneous analog transmission of result vectors from all devices and assets at the factory level to the fog PS. Let $\mathbf{w}_k = \left[w_{k,1}, \cdots, w_{k,q}\right]^T$ represent the $(q \times 1)$ local model parameter vector, and $\nabla F_k(\mathbf{w}) = \left[\nabla F_{k,1}, \cdots, \nabla F_{k,q}\right]^T$ denote the local gradient vector of the loss function from the k-th device. In FLOACC, it is assumed that the local gradient of each model is transmitted over an analog medium, with symbols denoted as $\left\{\tilde{\nabla F}_{k,i}\right\}$ and normalized to have zero mean and unit variance. By utilizing Orthogonal Frequency Division Multiplexing (OFDM), each element of the gradient vector can be allocated to a distinct sub-carrier OFDM channel. This approach significantly reduces the learning process latency for FLOACC.

In each round n, all local devices at a factory simultaneously transmit their local gradient based on the distributed loss function. Therefore, the PS obtains the aggregated gradient instead of the aggregated model as specified in equation 2.52. Extensive research has been conducted to optimize transmit power for achieving model convergence and mitigating the effects of existing noise on the aggregated signal (Cao et al., 2020; Liu et al., 2020b). In this context, we investigate the comparison of convergence performance among different algorithms without optimizing transmission power control. In the upcoming section, we delve into the FSVRG-OACC algorithm, which demonstrates superior convergence properties.

5.3 Adaptive FSVRG-OACC Algorithm

5.3.1 Algorithm

FSVR-OACC is derived from the SVRG algorithm, detailed in 2.2.3. SVRG employs two nested loops, updating iteratively in the inner loop as

$$\mathbf{w}[n+1] = \mathbf{w}[n] - \alpha \left[\nabla F^{i}(\mathbf{w}[n]) - \nabla F^{i}(\mathbf{w}^{t}[n]) + \nabla F(\mathbf{w}^{t}[n])\right]. \tag{5.1}$$

This algorithm is naturally suited for centralized implementations as it requires computing the stochastic gradient over the complete dataset. FSVRG, introduced in (Konečnỳ et al., 2016), is designed for distributed optimization. However, it does not exhibit satisfactory convergence in FL for over-the-air analog aggregation. A critical challenge lies in the significant variation in the number of available data points among edge devices, differing from the average. Unlike FSVR, our assumptions consider analog communication as the sole type between local devices and the PS. Consequently, the PS lacks information about the number of data points and data distribution types. In this scenario, local data often clusters around a specific pattern, making it unrepresentative of the overall distribution we aim to learn. Considering aggregation on the entire gradient direction in each iteration could be a promising approach in the concept of analog aggregation.

FSVRG-OACC involves two communication rounds, which increases communication costs but provides advantages in the convergence algorithm. Algorithm 5.1 introduces FSVRG-OACC, a modified version for over-the-air analog aggregation. Practically, $\left[\nabla F^{i_t}(\mathbf{w_k}[n]) - \nabla F_k^{i_t}(\mathbf{w}^{t}[n]) + \nabla F_k(\mathbf{w}^{t}[n])\right]$ is split into two parts. In the first distributed loop, the last two gradients $\nabla F_k^{i_t}(\mathbf{w}^{t}[n])$ and $\nabla F_k(\mathbf{w}^{t}[n])$ are calculated, aggregating their distance across all edge devices. In the second loop, the aggregated value of $\left[\nabla F_k^{i_t}(\mathbf{w}^{t}[n]) - \nabla F_k(\mathbf{w}^{t}[n])\right]$ is accessed. In the first iteration ($n = 1$), the stochastic gradient for each device is computed, subtracted from the aggregated gradient obtained in the first loop, resulting in $\left[\nabla F^{i_t}(\mathbf{w_k}[n]) - \text{aggregation}(\nabla F_k^{i_t}(\mathbf{w}^{t}[n]) - \nabla F_k(\mathbf{w}^{t}[n]))\right]$. This overall gradient update is then aggregated over the air with other devices and then is used to directly update the model parameters of each device. The details of the FSVRG-OACC algorithm are explained in Paper V.

5.3.2 System Model and Dataset

The performance of the proposed algorithm is assessed using FedSVM for anomaly detection in a PdM application. The evaluation involves over-the-air

Algorithm 5.1: FSVRG With Over-the-Air Communication and Computation (FSVRG-OACC)

Parameters: ϕ= number of data points ,ϕ_k= number of data points store on device k, α = stepsize, data partition $\{\mathscr{D}_k\}_{k=1}^{K}$, Randomly initialize $\mathbf{w}_k$ on each device.

for $n = 0, 1, \ldots$ **do**

  for $k = 1$ *to* K ***do in parallel*** *over device* k **do**

    1: Let $\{i_t\}_{t=1}^{\phi_k}$ be random permutation of $\mathscr{D}_k \rightarrow$ Distribution**1**

    2: Compute $\nabla F_k^{i_t}(\mathbf{w}^t[n])$

    3: Compute $\nabla F_k(\mathbf{w}[n]) = \frac{1}{\phi_k}\sum_{i=1}^{\phi_k} \nabla F_k^{i_t}(\mathbf{w}^t[n])$

    4: Over-the-Air Gradient Aggregation: Each device k uploads $g_k = \left[\nabla F_k^{i_t}(\mathbf{w}^t[n]) - \nabla F_k(\mathbf{w}[n])\right]$ Over-the-Air.

  end

  Aggregated signal in PS (Server Side)

  $$\hat{g} = \frac{1}{K}\sum_{k=1}^{K} \frac{h_k[n]p_k[n]g_k}{\sqrt{\eta}} + \frac{z[n]}{\sqrt{\eta}} \rightarrow \text{Aggregation}\mathbf{1}$$

  Aggregated signal $\hat{g}$ is received by all edge devices

  for $k = 1$ *to* K ***do in parallel*** *over device* k **do**

    1: Initialize: $\alpha_k = \alpha/\phi_k$ $\rightarrow$ Distribution**2**

    2: Compute $\nabla F^{i_t}(\mathbf{w_k}[n])$

    3: Aggregated signal $\hat{G}$ is received by the edge device k

    $n = 0 \rightarrow$ randomly initialize $\hat{G}$

    for $t = 1, \ldots, \phi_k$ **do**

      $\mathbf{w}_k[n+1] = \mathbf{w}_k[n] - \alpha_k \hat{G}$

    end

    4: Over-the-Air Gradient Aggregation: Each device k uploads $G_k = \left[\nabla F^{i_t}(\mathbf{w_k}[n]) - \hat{g}\right]$ Over-the-Air.

  end

  Aggregated signal in PS (Server Side)

  $$\hat{G} = \frac{1}{K}\sum_{k=1}^{K} \frac{h'_k[n]p'_k[n]G_k}{\sqrt{\eta'}} + \frac{z'[n]}{\sqrt{\eta'}} \rightarrow \text{Aggregation}\mathbf{2}$$

end

analog aggregation on the CMAPSS dataset. The goal is to explore how four optimization algorithms GD, SGD, FSVRG, and FSVRG-OACC converge in the context of over-the-air analog aggregation. The focus is on examining the performance of these algorithms in terms of model accuracy and convergence rate, crucial metrics in this field.

5.3.3 Results of Evaluation and Discussion

FSVRG-OACC optimizes the FedSVM problem on CMAPSS, comparing results with standard FSRVG, SGD, and GD methods to highlight notable

improvements in convergence rate and accuracy within the context of over-the-air analog aggregation in a noisy environment. The channel inversion policy estimates the transmitted signal in this implementation. However, it's important to note that the main focus of this study is not on the transmission policy.

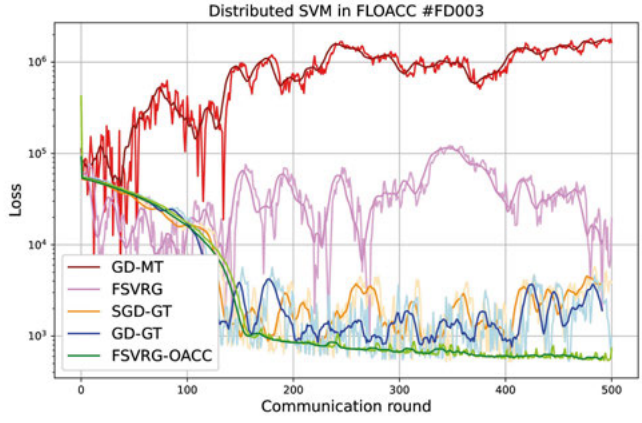


Figure 37. Convergence versus communication round for different algorithms in Over-the-Air analog aggregation ($\sigma_z^2 = 1$, $p = 1$).

Figure 37 illustrates the descending regime of the distributed FedSVM loss function with over-the-air analog aggregation. The lighter graphs depict the signal, while the bolder graphs represent the windowed average of the signal. In this experiment involving 10 devices ($K = 10$) at the factory site, and the transmitting power is set to $P = 300$ mW.

In a distributed setting, there are two approaches: GT, where gradients are transmitted and aggregated for model updating, and MT, where the model is updated locally and then transmitted for aggregation. Both GD-MT and FSVRG algorithms fail to converge under the given noise and power settings, so they are excluded from accuracy analysis. However, GD-GT and SGD-GT converge with significant fluctuations. In contrast, our proposed FSVRG-OACC method shows excellent convergence performance under the same conditions. In the accuracy plot, both SGD-DT and GD-GT algorithms experience a notable drop in accuracy, while our proposed algorithm maintains an average accuracy of 91%, demonstrating higher stability compared to others.

Implementing the proposed approach in real-world industrial settings comes with practical implications and challenges. A significant aspect of this approach involves adjusting the local gradients in large edge devices using an analog waveform and transmitting them through the same wireless channels. The

primary challenge lies in achieving perfect waveform superposition, crucial for the success of FSVRG-OACC algorithm. This task is complicated due to frame timing offset and carrier frequency offset. Overcoming these issues necessitates high-performance devices with substantial computational capabilities. Another significant challenge arises when we only have access to partial information, leading to some edge devices being unable to effectively participate in the FLOACC aggregation process. This limitation results in a deviation in the gradient descent regime, highlighting the importance of robust device selection and strategies to manage situations where certain devices may have limited participation capabilities.

6 Conclusions and Outlook

6.1 Summary

Industry 4.0 is poised to revolutionize industrial manufacturing, aiming to enhance efficiency and productivity. In this context, the maintenance process within manufacturing is evolving into a machine-assisted digital version, focused on monitoring and ensuring asset performance. The communication between assets and their digital counterparts is pivotal for their success, especially leveraging wireless technology. However, challenges arise in achieving control stability through wireless network control systems. Additionally, while data-driven models like machine learning are making significant strides across various domains, but centralized solution is not feasible due to the substantial communication latency it would introduce. As a solution, edge computing emerges, enabling learning algorithms at the edge in a distributed and collaborative manner. This approach addresses the limitations of centralized solutions and holds promise for advancing the efficiency of maintenance applications. This dissertation aims to address such research questions, and five scientific papers have been published to provide insights into these inquiries.

The exploration of this field begins with the establishment of an efficient testbed enabling testing, implementation, and troubleshooting of various control algorithms for a wireless network control system. Consequently, a platoon of vehicles serves as the testbed, representing closed-loop control over a wireless network (*Paper I*). This setup illustrates the difficulties in establishing reliable closed-loop control systems via wireless communication. Particularly in noisy environments, coordinating the smooth movement of a platoon poses a challenging problem under these conditions.

In the context of ongoing connectivity, the constrained bandwidth in wireless communication channels remains a significant challenge. To address this issue and reduce communication load among wireless agents, an event-triggering algorithm is explored. However, control performance becomes uncertain in an industrial environment with a high probability of packet dropping. Therefore, as the first innovative method in this thesis, a distributed event triggering algorithm in the state feedback controller for multi-agent systems is proposed. This involves applying distributed event-based state estimation methods to design a parallel event triggering algorithm for multi-agent systems while maintaining satisfactory control performance, even in conditions with a high probability of packet drops (*Paper II* and *III*). The suggested event-triggering algorithm slightly increases resource usage, but it ensures the stability of multi-agent

systems, even in networks with a high chance of packet drops. While this approach performs effectively, alternative methods like adaptive threshold tuning can also operate well in networks with high packet drops, albeit with an increase in network usage.

In the context of distributed ML for PdM applications and edge computing, the thesis introduces two federated algorithms as the second and third innovative methods. Specifically, FedSVM is presented for distributed anomaly detection, and FedLSTM is proposed for distributed RUL estimation (*Paper IV*). These algorithms provide factories at the fog level with the capability to enhance the accuracy of their PdM models while preserving privacy. These two federated algorithms show effectiveness in PdM applications. However, they face challenges when dealing with non-iid data. Achieving convergence becomes particularly difficult in cases where the data distribution is highly non-iid. Additionally, both algorithms currently focus only on the same assets at the edge level. For future developments, it is recommended to consider a federated model capable of handling various types of assets and different data types. Building upon distributed algorithms, an aggregation strategy is proposed for collaborative PdM applications across the cloud, fog, and edge levels. This facilitates the utilization of a globally trained model for PdM by different companies.

Continuing, the focus shifts to leveraging wireless transmission channels for both communication and computation. Over-the-air analog aggregation is introduced as a method that combines computation and communication simultaneously for model updates in a distributed learning system. However, the learning performance is degraded due to channel noise. Therefore, we propose FSVRG-OACC, a novel tracking-based stochastic approximation scheme adapted as a federated stochastic variance-reduced gradient method for over-the-air computation (*Paper V*). This algorithm effectively averages out channel noise's influence, ensuring robust learning performance without increasing transmission power gain. Nevertheless, while this algorithm ensures convergence, it comes at the cost of doubling the communication rounds. The over-the-air aggregation strategy, though promising, remains a challenging issue that necessitates further practical implementation within this domain.

6.2 Future Research

In the context of the suggested parallel event-triggering algorithm, we applied model-based state estimation and prediction to compare and initiate events. Event-triggered learning is a new idea designed to minimize communication and adjust to changing dynamics in network control systems using learning

algorithms. This learning could involve reinforcement learning to adapt to various environments, including scenarios with a high probability of packet drops. Future research could focus on developing the proposed event-triggering algorithm, incorporating reinforcement learning for handling more complex systems.

In the suggested collaborative PdM scenario, the feasibility of implementing federated learning in PdM applications has been confirmed. With advancements in device and sensor technology, there is an opportunity to integrate IoT sensors into the products of assets, enabling the development of a comprehensive FL algorithm for future research. This federated model takes input data from assets and products and even includes considerations for management and business models. It is important to explore a learning method capable of handling diverse and non-iid data from assets, products, and administrative sources. Such a distributed module could be applicable to various companies with different assets.

In light of our recent proposal, the FSVRG-OACC optimization algorithm aimed at mitigating noise in over-the-air aggregation solutions. Comprehensive theoretical analyses have been conducted in this area. The state of over-the-air analog aggregation has reached a level of maturity, prompting consideration for real-world implementation. Leveraging high-speed devices with advanced computational capabilities on both transmitters and receivers, the next phase of future research involves the practical deployment of over-the-air aggregation. This initiative could be intended to conduct experimental measurements to assess the comparative effectiveness of learning over-the-air aggregation versus offline learning.

References

Alli, A. A. and Alam, M. M. (2020). The fog cloud of things: A survey on concepts, architecture, standards, tools, and applications. *Internet of Things*, 9:100177.

Amiri, M. M. and Gündüz, D. (2020). Machine learning at the wireless edge: Distributed stochastic gradient descent over-the-air. *IEEE Transactions on Signal Processing*, 68:2155–2169.

Amiri, M. M., Gündüz, D., Kulkarni, S. R., and Poor, H. V. (2021). Convergence of federated learning over a noisy downlink. *IEEE Transactions on Wireless Communications*, 21(3):1422–1437.

Ang, F., Chen, L., Zhao, N., Chen, Y., Wang, W., and Yu, F. R. (2020). Robust federated learning with noisy communication. *IEEE Transactions on Communications*, 68(6):3452–3464.

Antsaklis, P. and Baillieul, J. (2007). Special issue on technology of networked control systems. *Proceedings of the IEEE*, 95(1):5–8.

Aranda-Escolastico, E., Guinaldo, M., Heradio, R., Chacon, J., Vargas, H., Sánchez, J., and Dormido, S. (2020). Event-based control: A bibliometric analysis of twenty years of research. *IEEE Access*, 8:47188–47208.

Araújo, J., Mazo, M., Anta, A., Tabuada, P., and Johansson, K. H. (2013). System architectures, protocols and algorithms for aperiodic wireless control systems. *IEEE Transactions on Industrial Informatics*, 10(1):175–184.

Baillieul, J. and Antsaklis, P. J. (2007). Control and communication challenges in networked real-time systems. *Proceedings of the IEEE*, 95(1):9–28.

Bemani, A. and Björsell, N. (2020). Cyber-physical control of indoor multi-vehicle testbed for cooperative driving. In *2020 IEEE Conference on Industrial Cyberphysical Systems (ICPS)*, volume 1, pages 371–377. IEEE.

Bemani, A. and Björsell, N. (2021a). Distributed event-triggered control of vehicular networked system with bursty packet drops. In *2021 7th International Conference on Event-Based Control, Communication, and Signal Processing (EBCCSP)*, pages 1–7. IEEE.

Bemani, A. and Björsell, N. (2021b). Distributed event triggering algorithm for multi-agent system over a packet dropping network. *Sensors*, 21(14):4835.

Bemani, A. and Björsell, N. (2022). Aggregation strategy on federated machine learning algorithm for collaborative predictive maintenance. *Sensors*, 22(16):6252.

Bemani, A. and Björsell, N. (2023). Low-latency collaborative predictive maintenance: Over-the-air federated learning in noisy industrial environments. *Sensors*, 23(18):7840.

Bharti, S. and Mcgibney, A. (2021). Privacy-aware resource sharing in cross-device federated model training for collaborative predictive maintenance. *IEEE Access*, 9:120367–120379.

Cao, H., Huang, X., Song, Y., and Lewis, F. L. (2023). Cooperative control of multiagent systems: A quantization feedback-based event-triggered approach. *IEEE Transactions on Cybernetics*.

Cao, L., Pan, Y., Liang, H., and Huang, T. (2022). Observer-based dynamic event-triggered control for multiagent systems with time-varying delay. *IEEE Transactions on Cybernetics*, 53(5):3376–3387.

Cao, X., Cheng, P., Chen, J., and Sun, Y. (2012). An online optimization approach for control and communication codesign in networked cyber-physical systems. *IEEE Transactions on Industrial Informatics*, 9(1):439–450.

Cao, X., Zhu, G., Xu, J., and Huang, K. (2020). Optimized power control for over-the-air computation in fading channels. *IEEE Transactions on Wireless Communications*, 19(11):7498–7513.

Chamaken, A., Litz, L., Krämer, M., and Gotzhein, R. (2009). A new approach to the joint design of control and communication in wireless networked control systems. *Automation 2009, VDI-Berichte/VDI-Tagungsbände*, pages 251–255.

Chen, B., Wan, J., Celesti, A., Li, D., Abbas, H., and Zhang, Q. (2018). Edge computing in iot-based manufacturing. *IEEE Communications Magazine*, 56(9):103–109.

Chen, J., Fan, Y., Zhang, C., and Song, C. (2020a). Sampling-based event-triggered and self-triggered control for linear systems. *International Journal of Control, Automation and Systems*, 18:672–681.

Chen, M., Yang, Z., Saad, W., Yin, C., Poor, H. V., and Cui, S. (2020b). A joint learning and communications framework for federated learning over wireless networks. *IEEE Transactions on Wireless Communications*, 20(1):269–283.

Chien, C.-F. and Chen, C.-C. (2020). Data-driven framework for tool health monitoring and maintenance strategy for smart manufacturing. *IEEE Transactions on Semiconductor Manufacturing*, 33(4):644–652.

Cui, H., Zhao, G., Liu, S., and Li, Z. (2023). A decentralized dynamic self-triggered control approach to consensus of multiagent systems. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*.

Damgrave, R. and Lutters, E. (2019). Smart industry testbed. *Procedia CIRP*, 84:387–392.

De Donno, M., Tange, K., and Dragoni, N. (2019). Foundations and evolution of modern computing paradigms: Cloud, iot, edge, and fog. *Ieee Access*, 7:150936–150948.

Donkers, M., Heemels, W., Van de Wouw, N., and Hetel, L. (2011). Stability analysis of networked control systems using a switched linear systems approach. *IEEE Transactions on Automatic control*, 56(9):2101–2115.

Du, X., Cai, Y., Wang, S., and Zhang, L. (2016). Overview of deep learning. In *2016 31st Youth Academic Annual Conference of Chinese Association of Automation (YAC)*, pages 159–164. IEEE.

Eldar, Y. C., Goldsmith, A., Gündüz, D., and Poor, H. V. (2022). *Machine learning and wireless communications*. Cambridge University Press.

Elefteriadou, L. (2020). Cyber physical systems in transportation: Traffic management with connected and autonomous vehicles. *Cyber-Physical Systems in the Built Environment*, pages 237–254.

Feng, Y., Nie, X., and Chen, X. (2019). Robust optimal filtering over lossy networks. *IEEE Transactions on Automatic Control*, 65(5):2272–2277.

Findeisen, R. and Varutti, P. (2009). Stabilizing nonlinear predictive control over nondeterministic communication networks. *Nonlinear Model Predictive Control: Towards New Challenging Applications*, pages 167–179.

Firouzi, F., Farahani, B., and Marinšek, A. (2022). The convergence and interplay of edge, fog, and cloud in the ai-driven internet of things (iot). *Information Systems*, 107:101840.

Garcia, E., Cao, Y., and Casbeer, D. W. (2016). Periodic event-triggered synchronization of linear multi-agent systems with communication delays. *IEEE Transactions on Automatic Control*, 62(1):366–371.

Ge, N., Li, G., Zhang, L., and Liu, Y. (2022). Failure prediction in production line based on federated learning: an empirical study. *Journal of Intelligent Manufacturing*, 33(8):2277–2294.

Goldenbaum, M. and Stanczak, S. (2014). On the channel estimation effort for analog computation over wireless multiple-access channels. *IEEE Wireless Communications Letters*, 3(3):261–264.

Grüne, L., Pannek, J., and Worthmann, K. (2012). Ensuring stability in networked systems with nonlinear mpc for continuous time systems. In *2012 IEEE 51st IEEE Conference on Decision and Control (CDC)*, pages 14–19. IEEE.

Guinaldo, M., Dimarogonas, D. V., Lehmann, D., and Johansson, K. H. (2015). Distributed event-based control for interconnected linear systems. *Asynchronous Control for Networked Systems*, pages 149–179.

Guinaldo, M., Lehmann, D., Sánchez, J., Dormido, S., and Johansson, K. H. (2014). Distributed event-triggered control for non-reliable networks. *Journal of the Franklin Institute*, 351(12):5250–5273.

Heemels, W. P., Johansson, K. H., and Tabuada, P. (2012). An introduction to event-triggered and self-triggered control. In *2012 ieee 51st ieee conference on decision and control (cdc)*, pages 3270–3285. IEEE.

Hochreiter, S. and Schmidhuber, J. (1997). Long short-term memory. *Neural computation*, 9(8):1735–1780.

Ishizaka, A. and Nishimura, H. (2021). An architecture description of resilience sos engineering process for critical infrastructure. In *2021 16th International Conference of System of Systems Engineering (SoSE)*, pages 25–30. IEEE.

Jamshidi, M., Betancourt, A. J., and Gomez, J. (2011). Cyber-physical control of unmanned aerial vehicles. *Scientia Iranica*, 18(3):663–668.

Jiang, Q., Zhou, Y., Wang, J., Wang, Y., and Wang, X. (2015). A lightweight cross-layer cooperative testbed for evaluation of connected vehicles. In *2015 11th International Conference on Mobile Ad-hoc and Sensor Networks (MSN)*, pages 194–201. IEEE.

Kaczmarczyk, V., Baštán, O., Bradáč, Z., and Arm, J. (2018). An industry 4.0 testbed (self-acting barman): principles and design. *IFAC-PapersOnLine*, 51(6):263–270.

Kim, K.-D. and Kumar, P. R. (2012). Cyber–physical systems: A perspective at the centennial. *Proceedings of the IEEE*, 100(Special Centennial Issue):1287–1308.

Konečnỳ, J., McMahan, B., and Ramage, D. (2015). Federated optimization: Distributed optimization beyond the datacenter. *arXiv preprint arXiv:1511.03575*.

Konečnỳ, J., McMahan, H. B., Ramage, D., and Richtárik, P. (2016). Federated optimization: Distributed machine learning for on-device intelligence. *arXiv preprint arXiv:1610.02527*.

Krouka, M., Elgabli, A., Issaid, C. B., and Bennis, M. (2021). Communication-efficient and federated multi-agent reinforcement learning. *IEEE Transactions on Cognitive Communications and Networking*, 8(1):311–320.

Lee, J., Bagheri, B., and Kao, H.-A. (2015). A cyber-physical systems architecture for industry 4.0-based manufacturing systems. *Manufacturing letters*, 3:18–23.

Leong, A. S., Dey, S., and Quevedo, D. E. (2016). Sensor scheduling in variance based event triggered estimation with packet drops. *IEEE Transactions on Automatic Control*, 62(4):1880–1895.

Li, B., Ma, Y., Westenbroek, T., Wu, C., Gonzalez, H., and Lu, C. (2016). Wireless routing and control: A cyber-physical case study. In *2016 ACM/IEEE 7th International Conference on Cyber-Physical Systems (ICCPS)*, pages 1–10. IEEE.

Li, W., Zhang, H., Zhou, Y., and Wang, Y. (2022). Bipartite formation tracking for multi-agent systems using fully distributed dynamic edge-event-triggered protocol. *IEEE/CAA Journal of Automatica Sinica*, 9(5):847–853.

Li, X., Ding, Q., and Sun, J.-Q. (2018). Remaining useful life estimation in prognostics using deep convolution neural networks. *Reliability Engineering & System Safety*, 172:1–11.

Li, Y. and Peng, L. (2018). Event-triggered fault estimation for stochastic systems over multi-hop relay networks with randomly occurring sensor nonlinearities and packet dropouts. *Sensors*, 18(3):731.

Liang, C., Wen, F., and Wang, Z. (2019). Trust-based distributed kalman filtering for target tracking under malicious cyber attacks. *Information Fusion*, 46:44–50.

Lim, W. Y. B., Luong, N. C., Hoang, D. T., Jiao, Y., Liang, Y.-C., Yang, Q., Niyato, D., and Miao, C. (2020). Federated learning in mobile edge networks: A comprehensive survey. *IEEE Communications Surveys & Tutorials*, 22(3):2031–2063.

Liu, K., Ji, Z., and Zhang, X. (2020a). Periodic event-triggered consensus of multi-agent systems under directed topology. *Neurocomputing*, 385:33–41.

Liu, S., Yu, J., Deng, X., and Wan, S. (2021). Fedcpf: An efficient-communication federated learning approach for vehicular edge computing in 6g communication networks. *IEEE Transactions on Intelligent Transportation Systems*, 23(2):1616–1629.

Liu, W., Zang, X., Li, Y., and Vucetic, B. (2020b). Over-the-air computation systems: Optimization, analysis and scaling laws. *IEEE Transactions on Wireless Communications*, 19(8):5488–5502.

Liu, Y., Garg, S., Nie, J., Zhang, Y., Xiong, Z., Kang, J., and Hossain, M. S. (2020c). Deep anomaly detection for time-series data in industrial iot: A communication-efficient on-device federated learning approach. *IEEE Internet of Things Journal*, 8(8):6348–6358.

Liu, Y., Peng, Y., Wang, B., Yao, S., and Liu, Z. (2017). Review on cyber-physical systems. *IEEE/CAA Journal of Automatica Sinica*, 4(1):27–40.

Lu, C., Saifullah, A., Li, B., Sha, M., Gonzalez, H., Gunatilaka, D., Wu, C., Nie, L., and Chen, Y. (2015). Real-time wireless sensor-actuator networks for industrial cyber-physical systems. *Proceedings of the IEEE*, 104(5):1013–1024.

Lu, N., Cheng, N., Zhang, N., Shen, X., and Mark, J. W. (2014). Connected vehicles: Solutions and challenges. *IEEE internet of things journal*, 1(4):289–299.

Lunze, J. and Lehmann, D. (2010). A state-feedback approach to event-based control. *Automatica*, 46(1):211–215.

Ma, Y., Gunatilaka, D., Li, B., Gonzalez, H., and Lu, C. (2018). Holistic cyber-physical management for dependable wireless control systems. *ACM Transactions on Cyber-Physical Systems*, 3(1):1–25.

Mamduhi, M. H., Tolić, D., Molin, A., and Hirche, S. (2014). Event-triggered scheduling for stochastic multi-loop networked control systems with packet dropouts. In *53rd IEEE Conference on Decision and Control*, pages 2776–2782. IEEE.

Martínez-Rey, M., Espinosa, F., Gardel, A., and Santos, C. (2015). On-board event-based state estimation for trajectory approaching and tracking of a vehicle. *Sensors*, 15(6):14569–14590.

McMahan, B., Moore, E., Ramage, D., Hampson, S., and y Arcas, B. A. (2017). Communication-efficient learning of deep networks from decentralized data. In *Artificial intelligence and statistics*, pages 1273–1282. PMLR.

Mohr, M., Becker, C., Möller, R., and Richter, M. (2021). Towards collaborative predictive maintenance leveraging private cross-company data. *INFORMATIK 2020*.

Muehlebach, M. and Trimpe, S. (2017). Distributed event-based state estimation for networked systems: An lmi approach. *IEEE Transactions on Automatic Control*, 63(1):269–276.

Murphy, K. P. (2012). *Machine learning: a probabilistic perspective*. MIT press.

Nie, T., Deng, Z., Wang, Y., and Qin, X. (2021). A robust unscented kalman filter for intermittent and featureless aircraft sensor faults. *IEEE Access*, 9:28832–28841.

Nilsson, A., Smith, S., Ulm, G., Gustavsson, E., and Jirstrand, M. (2018). A performance evaluation of federated learning algorithms. In *Proceedings of the second workshop on distributed infrastructures for deep learning*, pages 1–8.

Pan, Y., Wu, Y., and Lam, H.-K. (2022). Security-based fuzzy control for nonlinear networked control systems with dos attacks via a resilient event-triggered scheme. *IEEE Transactions on Fuzzy Systems*, 30(10):4359–4368.

Pan, Y.-J., Werner, H., Huang, Z., and Bartels, M. (2017). Distributed cooperative control of leader–follower multi-agent systems under packet dropouts for quadcopters. *Systems & Control Letters*, 106:47–57.

Park, D., Kim, S., An, Y., and Jung, J.-Y. (2018). Lired: A light-weight real-time fault detection system for edge computing using lstm recurrent neural networks. *Sensors*, 18(7):2110.

Park, P., Ergen, S. C., Fischione, C., Lu, C., and Johansson, K. H. (2017). Wireless network design for control systems: A survey. *IEEE Communications Surveys & Tutorials*, 20(2):978–1013.

Parkinson, S., Ward, P., Wilson, K., and Miller, J. (2017). Cyber threats facing autonomous and connected vehicles: Future challenges. *IEEE transactions on intelligent transportation systems*, 18(11):2898–2915.

Peng, C., Hu, Q., Wang, Z., Liu, R. W., and Xiong, Z. (2022). Online-learning-based fast-convergent and energy-efficient device selection in federated edge learning. *IEEE Internet of Things Journal*, 10(6):5571–5582.

Qiang, Z., Dai, L., Chen, B., and Xia, Y. (2022). Distributed model predictive control for heterogeneous vehicle platoon with inter-vehicular spacing constraints. *IEEE Transactions on Intelligent Transportation Systems*, 24(3):3339–3351.

Qin, J., Wang, J., Shi, L., and Kang, Y. (2020). Randomized consensus-based distributed kalman filtering over wireless sensor networks. *IEEE Transactions on Automatic Control*, 66(8):3794–3801.

Ramesh, C., Sandberg, H., and Johansson, K. H. (2016). Performance analysis of a network of event-based systems. *IEEE Transactions on Automatic Control*, 61(11):3568–3573.

Rawat, D., Brecher, C., Song, H., and Jeschke, S. (2017). Industrial internet of things: Cybermanufacturing systems. *Cham, Switzerland: Springer*.

Rong, N. and Wang, Z. (2021). Event-based fixed-time control for interconnected systems with discontinuous interactions. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, 52(8):4925–4936.

Ruder, S. (2016). An overview of gradient descent optimization algorithms. *arXiv preprint arXiv:1609.04747*.

Saeed, A., Neishaboori, A., Mohamed, A., and Harras, K. A. (2014). Up and away: A visually-controlled easy-to-deploy wireless uav cyber-physical testbed. In *2014 IEEE 10th International Conference on Wireless and Mobile Computing, Networking and Communications (WiMob)*, pages 578–584. IEEE.

Särkkä, S. and Svensson, L. (2013). *Bayesian filtering and smoothing*, volume 17. Cambridge university press.

Sater, R. A. and Hamza, A. B. (2021). A federated learning approach to anomaly detection in smart buildings. *ACM Transactions on Internet of Things*, 2(4):1–23.

Saxena, A., Goebel, K., Simon, D., and Eklund, N. (2008). Damage propagation modeling for aircraft engine run-to-failure simulation. In *2008 international conference on prognostics and health management*, pages 1–9. IEEE.

Schenato, L. (2008). Optimal estimation in networked control systems subject to random delay and packet drop. *IEEE Transactions on Automatic Control*, 53(5):1311–1317.

Schmitt, E. J., González, A., and Fettweis, G. P. (2022). Comparison of event-based remote state estimation techniques for uav formation control. In *2022 IEEE/SICE International Symposium on System Integration (SII)*, pages 90–96. IEEE.

Shi, D., Shi, L., and Chen, T. (2016). Event-based state estimation. *Switzerland: Springer*.

Shi, J., Wan, J., Yan, H., and Suo, H. (2011). A survey of cyber-physical systems. In *2011 international conference on wireless communications and signal processing (WCSP)*, pages 1–6. IEEE.

Sinopoli, B., Schenato, L., Franceschetti, M., Poolla, K., Jordan, M. I., and Sastry, S. S. (2004). Kalman filtering with intermittent observations. *IEEE transactions on Automatic Control*, 49(9):1453–1464.

Stark, R., Kind, S., and Neumeyer, S. (2017). Innovations in digital modelling for next generation manufacturing system design. *CIRP annals*, 66(1):169–172.

Tan, X., Cao, M., and Cao, J. (2020). Distributed dynamic event-based control for nonlinear multi-agent systems. *IEEE Transactions on Circuits and Systems II: Express Briefs*, 68(2):687–691.

Tao, F., Qi, Q., Wang, L., and Nee, A. (2019). Digital twins and cyber–physical systems toward smart manufacturing and industry 4.0: Correlation and comparison. *Engineering*, 5(4):653–661.

Teoh, Y. K., Gill, S. S., and Parlikad, A. K. (2021). Iot and fog computing based predictive maintenance model for effective asset management in industry 4.0 using machine learning. *IEEE Internet of Things Journal*.

Trimpe, S. (2017). Event-based state estimation: an emulation-based approach. *IET Control Theory & Applications*, 11(11):1684–1693.

Trimpe, S. and Baumann, D. (2019). Resource-aware iot control: Saving communication through predictive triggering. *IEEE Internet of Things Journal*, 6(3):5013–5028.

Trimpe, S. and Campi, M. C. (2015). On the choice of the event trigger in event-based estimation. In *2015 International Conference on Event-based Control, Communication, and Signal Processing (EBCCSP)*, pages 1–8. IEEE.

Vashisht, S. and Jain, S. (2019). An energy-efficient and location-aware medium access control for quality of service enhancement in unmanned aerial vehicular networks. *Computers & Electrical Engineering*, 75:202–217.

Verbraeken, J., Wolting, M., Katzy, J., Kloppenburg, J., Verbelen, T., and Rellermeyer, J. S. (2020). A survey on distributed machine learning. *Acm computing surveys (csur)*, 53(2):1–33.

Vilgelm, M., Mamduhi, M. H., Kellerer, W., and Hirche, S. (2016). Adaptive decentralized mac for event-triggered networked control systems. In *Proceedings of the 19th International Conference on Hybrid Systems: Computation and Control*, pages 165–174.

Wang, F., Wen, G., Peng, Z., Huang, T., and Yu, Y. (2019). Event-triggered consensus of general linear multiagent systems with data sampling and random packet losses. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, 51(2):1313–1321.

Wu, J., Jia, Q.-S., Johansson, K. H., and Shi, L. (2012). Event-based sensor data scheduling: Trade-off between communication rate and estimation quality. *IEEE Transactions on automatic control*, 58(4):1041–1046.

Xiao, L. and Zhang, T. (2014). A proximal stochastic gradient method with progressive variance reduction. *SIAM Journal on Optimization*, 24(4):2057–2075.

Xiong, J. and Lam, J. (2007). Stabilization of linear systems over networks with bounded packet loss. *Automatica*, 43(1):80–87.

Xiong, S., Wei, Z., and Zhang, L. (2022). Multiple harmonics disturbance-observer-based containment tracking control of multi-agent systems with asymmetric data packet dropout. In *2022 China Automation Congress (CAC)*, pages 2850–2855. IEEE.

Xu, L.-X., Zhao, L.-N., Ma, H., Wang, Y.-L., and Kang, H. (2021). Event-triggered cooperative output regulation of heterogeneous multi-agent systems

with adaptive fault-tolerant control. *IEEE Transactions on Circuits and Systems II: Express Briefs*, 69(3):1149–1153.

Xu, Z., Liu, T., and Jiang, Z.-P. (2022). Nonlinear integral control with event-triggered feedback: Unknown decay rates, zeno-freeness, and asymptotic convergence. *Automatica*, 137:110028.

Yang, H., Li, H., Xia, Y., and Li, L. (2020). Distributed kalman filtering over sensor networks with transmission delays. *IEEE Transactions on Cybernetics*, 51(11):5511–5521.

Yang, H. H., Liu, Z., Quek, T. Q., and Poor, H. V. (2019a). Scheduling policies for federated learning in wireless networks. *IEEE transactions on communications*, 68(1):317–333.

Yang, Q., Liu, Y., Chen, T., and Tong, Y. (2019b). Federated machine learning: Concept and applications. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 10(2):1–19.

Yi, X., Liu, K., Dimarogonas, D. V., and Johansson, K. H. (2018). Dynamic event-triggered and self-triggered control for multi-agent systems. *IEEE Transactions on Automatic Control*, 64(8):3300–3307.

Zeng, Q., Du, Y., Huang, K., and Leung, K. K. (2020). Energy-efficient radio resource allocation for federated edge learning. In *2020 IEEE International Conference on Communications Workshops (ICC Workshops)*, pages 1–6. IEEE.

Zhang, H., Zhou, X., Wang, Z., and Yan, H. (2019a). Maneuvering target tracking with event-based mixture kalman filter in mobile sensor networks. *IEEE Transactions on Cybernetics*, 50(10):4346–4357.

Zhang, W., Branicky, M. S., and Phillips, S. M. (2001). Stability of networked control systems. *IEEE control systems magazine*, 21(1):84–99.

Zhang, X., Xiao, P., Yang, Y., Cheng, Y., Chen, B., Gao, D., Liu, W., and Huang, Z. (2019b). Remaining useful life estimation using cnn-xgb with extended time window. *IEEE Access*, 7:154386–154397.

Zhao, R., Yan, D., Liu, Q., Leng, J., Wan, J., Chen, X., and Zhang, X. (2019). Digital twin-driven cyber-physical system for autonomously controlling of micro punching system. *IEEE Access*, 7:9459–9469.

Zheng, Y., Li, S. E., Li, K., Borrelli, F., and Hedrick, J. K. (2016). Distributed model predictive control for heterogeneous vehicle platoons under unidirectional topologies. *IEEE Transactions on Control Systems Technology*, 25(3):899–910.

Zhong, Y. and Liu, Y. (2021). Flexible optimal kalman filtering in wireless sensor networks with intermittent observations. *Journal of the Franklin Institute*, 358(9):5073–5088.

Zhong, Y., Tang, J., Yang, N., Shi, D., and Shi, L. (2023). Event-triggered sensor scheduling for remote state estimation with error-detecting code. *IEEE Control Systems Letters*.

Zhou, X., Xu, X., Liang, W., Zeng, Z., Shimizu, S., Yang, L. T., and Jin, Q. (2021). Intelligent small object detection for digital twin in smart manufacturing with industrial cyber-physical systems. *IEEE Transactions on Industrial Informatics*, 18(2):1377–1386.

Zhou, Y., Wang, J., Du, H., Li, H., Hu, P., Wang, G., and Shao, R. (2014). The multi-agent based evaluation of connected vehicle systems. In *2014 IEEE Vehicular Networking Conference (VNC)*, pages 131–132. IEEE.

Zhu, G., Wang, Y., and Huang, K. (2019). Broadband analog aggregation for low-latency federated edge learning. *IEEE Transactions on Wireless Communications*, 19(1):491–506.

Zhu, M., Sui, T., and Wang, R. (2023). Distributed kalman filtering over sensor networks with fading measurements and random link failures. *Journal of the Franklin Institute*, 360(4):2457–2475.

Zimmerling, M., Mottola, L., Kumar, P., Ferrari, F., and Thiele, L. (2017). Adaptive real-time communication for wireless cyber-physical systems. *ACM Transactions on Cyber-Physical Systems*, 1(2):1–29.

Papers
Associated papers have been removed in the electronic version of this thesis.
For more details about the papers see:
http://urn.kb.se/resolve?urn=urn:nbn:se:hig:diva-43564