Doctoral Thesis
博 士 論 文

Developing a System for Constructing Digital Twins of Machining Phenomena Based on Semantic Annotation and Time-Delayed Sensor Signals

セマンティックアノテーションおよび時間遅れ型センサー信号を規範とする機械加工現象のデジタルツイン作成システムの開発

March 2022

Angkush Kumar Ghosh

Kitami Institute of Technology
Kitami, Japan

Abstract (In English)

Manufacturing has rapidly been transforming under the umbrella of the fourth industrial revolution, known as Industry 4.0 or smart manufacturing, which diligently utilizes information and communication technology. In smart manufacturing, cyber-physical systems host Internet-of-Things (IoT)-based networks and digital twins. The networks integrate manufacturing enablers such as computer numerical control machine tools, robots, numerous process and resource planning systems, and human resources. Digital twins are computable virtual abstractions of real-world entities exhibiting real-time responsive capacities. The twins work as the brains of the enablers; that is, the twins supply the required knowledge and help enablers solve problems autonomously, responding to various sensor signals in real-time.

Remarkably, three types of digital twins (object, process, and phenomenon twins) must populate the cyber-physical systems. Compared to other twins, phenomena twins have not yet been researched elaborately. This thesis fills this gap. The issues underlying semantic annotation and time latency (or delay) are significant for a phenomenon twin. Time latency or delay occurs when sensor signals are exchanged through the abovementioned embedded systems. As a result, the signal at its origin (e.g., machine tools) and signal received at the receiver end (e.g., digital twin) differ. Moreover, many datasets of heterogeneous sensor signals are exchanged through IoT-based networks. Hence, acquiring the right signals for a twin is difficult and time-consuming. Semantic annotation-based representation of sensor signals can solve this problem. Thus, a phenomenon twin must machine-learn the required knowledge to emulate the phenomenon from the relevant historical sensor signal datasets, seamlessly interact with the real-time sensor signals, handle the semantically annotated datasets stored in clouds, and accommodate the transmission delay or latency.

Accordingly, this thesis presents two systems denoted as Digital Twin Construction System (DTCS) and Digital Twin Adaptation System (DTAS). The first system constructs a phenomenon twin, and the other adapts the constructed twin into a cyber-physical system. Both systems are developed using a Java™-based platform. The modular architectures of the systems are presented in detail. In addition, real-life machining torque signals are used to demonstrate the efficacy of DTCS and DTAS.

DTCS consists of five modules denoted as Input, Modeling, Simulation, Validation, and Output Modules. The Input Module can make sense of the semantically annotated datasets and helps users select the right ones. It ensures fast and effective data mining using a human-machine-comprehensible semantic annotation mechanism (concept map and Extensible Markup Language (XML) driven). The Modeling Module can extract the required knowledge to emulate a phenomenon from the information supplied by the Input Module. This module uses a Markov chain-based machine learning method and accommodates data transmission delay-related arrangements. The Simulation Module can operate on the knowledge extracted by the Modeling Module and simulate the signals of the phenomenon using a discrete event-based stochastic simulation method. The Validation Module can validate the simulated signals of the phenomenon against the real signals using quantitative measures (e.g., fuzzy numbers). Finally, the Output Module transfers the selected Modules of DTCS to DTAS. DTAS, in turn, can adapt the constructed phenomenon twin into the cyber-physical system for monitoring and troubleshooting.

The thesis is organized as follows. Chapter 1 presents the introduction of this study. Chapter 2 provides a literature review on the role of cyber-physical systems and digital twins in smart manufacturing or Industry 4.0. Chapter 3 describes a semantic annotation-based representation mechanism of data and knowledge. Chapter 4 describes the role of the delay domain in mitigating the effect of time delay or latency of signal transmission. Chapter 5 presents the proposed DTCS and DTAS. Chapter 6 demonstrates the efficacy of DTCS and DTAS using a real-life case of intelligent monitoring of machining (milling). Chapter 7 discusses the implications of this study and highlights future research directions. Finally, Chapter 8 provides the concluding remarks of this thesis.

Since the digital twins of the machining phenomena are needed to make the machine tools and other programmable devices more intelligent and autonomous, the presented DTCS and DTAS contribute to the befitting advancement of Industry 4.0 or smart manufacturing.

Abstract (In Japanese)

製造業は、インダストリー4.0またはスマートマニュファクチャリングとして知られる第4次産業革命のもとで急速に変化している。この革命は積極的な情報通信技術の利用によってサイバーフィジカルシステムを明示し、IoTを規範とするネットワークおよびデジタルツインで構成される。IoTを規範とするネットワークは、コンピューター数値制御型工作機械、ロボット、多数のプロセスおよびリソース計画システム、人材などの製造イネーブラーを統合する。デジタルツイン（リアルタイムの応答能力を持つ実世界のエンティティの計算可能な仮想モデル化）は製造イネーブラーの頭脳として機能している。つまり、デジタルツインは製造イネーブラーが問題を自律的に解決するために必要な知識の獲得やさまざまなセンサー信号のリアルタイムによる応答を支援する。

注目すべき点はサイバーフィジカルシステムには3種類のデジタルツイン（物体、工程、および現象ツイン）を用意しなければならない点である。しかし物体及び工程ツインの研究は進んでいるが、現象ツインにおいてはまだ研究は不十分である。本論文はこのギャップを埋めることを目的にしている。

現象ツインは、サイバースペースで与えられた現象（切削抵抗、トルク、表面粗さなど）をエミュレートし、製造イネーブラー（工作機械など）に必要な知識を与えることでそのイネーブラーをより効率的に活躍させる。しかし現象ツインの作成に当たり、セマンティックアノテーション及び時間遅れという問題を考慮した対策を導入する必要がある。

時間遅れは、センサー信号が上記の組み込みシステムを介して交換されるときに発生する。その結果、発信元の信号（工作機械側）と受信側で獲得される信号（デジタルツイン側）が異なる。さらに、異種センサー信号の多くのデータセットは、IoTを規範とするネットワークを介

して交換される。従って、現象ツインには次の機能が求められる。

1. 与えられたセンサー信号データセットから必要な知識を機械学習によって獲得すること。
2. リアルタイムセンサー信号とシームレスに相互作用すること。
3. クラウドに保存されているセマンティックアノテーションデータセットを処理すること。
4. センサー信号の時間遅れに対応すること。

本論文ではデジタルツイン構築システム（DTCS）およびデジタルツイン適応システム（DTAS）について述べる。DTCSは現象ツインを構築し、DTASは構築されたツインをサイバーフィジカルシステムに適応させる。本システムはJava™プラットフォームによって開発する。各システムの詳細について述べるとともに機械加工のとき発生するトルク信号を用いて開発されたシステムの有効性を実証する。DTCSは、入力、モデリング、シミュレーション、バリデーション、および出力という5つのモジュールで構成される。入力モジュールは、セマンティックアノテーションされた信号データセットを理解し、ユーザーが選択した適切なデータセットを獲得することができる。このモジュールは、セマンティックアノテーションメカニズムによって用意されたデータ（概念マップおよびExtensible Markup Language（XML）型データ）を高速かつ効果的にマイニングすることができる。モデリングモジュールは、入力モジュールによって提供されるデータセットから現象をエミュレートするための知識を獲得することができる。その際、マルコフ連鎖型機械学習の実施および時間遅れの影響に対応することができる。シミュレーションモジュールは、モデリングモジュールによって獲得された知識に基づいて動作し、離散事象シミュレーションを用いて現象の信号をシミュレートすることができる。バリデーションモジュールは、定量的手法によって（例：ファジー数）現象のシミュレーションされた信号を実際の信号に対して検証することができる。出力モジュールは、DTCSから選択されたモジュールをDTASに転送することができる。最後に、モニタリングやトラブルシューティングのため、DTASはDTCSから転送された現象ツインの各モジュールをサイバーフィジカルシステムに導入することができる。

本論文は次のように構成する。 第1章では、第4次産業革命および関連研究分野の概要を説明する。第2章では、インダストリー4.0におけるサイバーフィジカルシステムやデジタルツインの役割に関する文献をレビューする。第3章では、セマンティックアノテーションに関するメカニズムを述べる。第4章では、時間遅れとそのセンサー信号の性質への影響および時間遅れドメイン型信号処理の有効性について述べる。

第5章では、DTCSならびにDTASの構成や開発について述べる。第6章では、DTCSならびにDTASの有効性を実例によって示す。第7章では、本論文の含意および今後の研究課題について述べる。最後に第8章では、本論文の結論を述べる。

機械加工現象のデジタルツインは、工作機械の知能化および自律化に有用であるため本論文で示したシステムはスマートマニュファクチャリングに欠かせない。また工作機械以外のデバイスにおいても同様のことが言える。従って、本論文で示したDTCSおよびDTASはインダストリー4.0またはスマートマニュファクチャリングの進歩に貢献する。

Contents

Chapter 1

Introduction

This chapter addresses the introduction of this study. The introduction outlines the research background (Section 1.1), the context (Section 1.2), the scope (Section 1.3), the aim and objectives (Section 1.4), and the contribution and significance (Section 1.5) of this study. Finally, this chapter presents the structure of this thesis (Section 1.6).

1.1 Research Background

Manufacturing sector has been evolving due to several industrial revolutions throughout time. In the first industrial revolution (also known as Industry 1.0), the main theme was to utilize steam engine-based devices. In the second industrial revolution (also known as Industry 2.0), the main theme was to enhance productivity by using mass production assembly lines. In the third industrial revolution (also known as Industry 3.0), the main theme was to automate manufacturing tasks by using numerically controlled devices. Nowadays, the fourth industrial revolution, popularly known as Industry 4.0 [1] or smart manufacturing [2], has been fostering profound transformations in the traditional manufacturing landscape. This evolution of manufacturing sector is schematically illustrated in Figure 1.1.

Industry 4.0 or smart manufacturing diligently utilizes the Information and Communication Technologies (ICT) to bring about automation and autonomy among the manufacturing enablers (e.g., machine tools, robots, CAD/CAM systems, process planning systems, resource planning systems

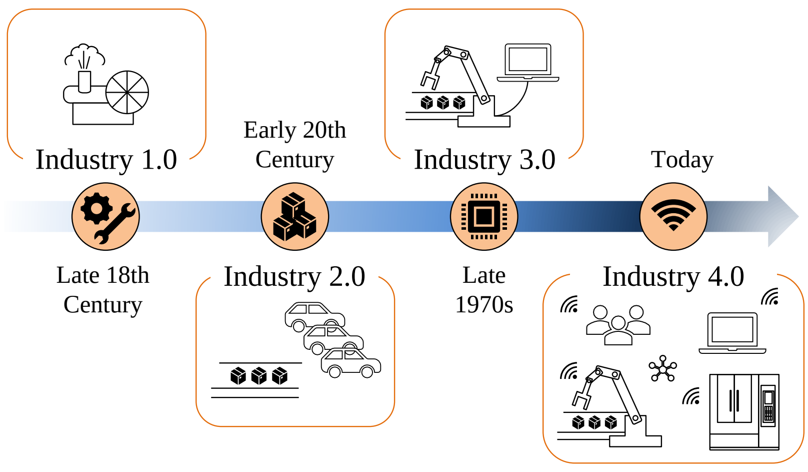


Figure 1.1: Evolution of manufacturing.

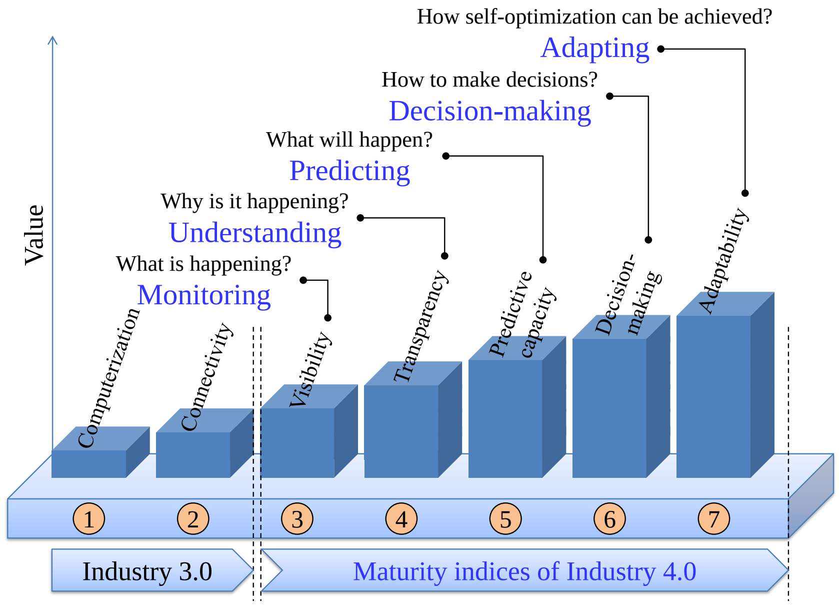


Figure 1.2: Maturity indices of Industry 4.0 (re-arranged from the work in [3]).

(ERP), human resources, and alike). As such, the enablers must perform some high-level cognitive tasks such as monitoring, understanding, predicting, decision-making, and adapting in real-time [3, 4]. Here, monitoring means what is happening in a given manufacturing context. Understanding means why it is happening. Predicting means what will happen. Decision-making means setting the right course of actions based on understanding and prediction. Adapting means to adapt the made decisions and selfoptimization. These cognitive tasks are considered the maturity indices for transforming a traditional automation-centric manufacturing environment (Industry 3.0) to an Industry 4.0 environment, as schematically illustrated in Figure 1.2.

Nevertheless, embedded systems such as Cyber-Physical Systems (CPS) [4, 5, 6] and the Internet of Things (IoT) [6, 7] empower the enablers toward performing the abovementioned cognitive tasks. The CPS is a seamless merger among the physical manufacturing environment and its cyber counterpart. As seen in Figure 1.3, CPS hosts four basic constituents: (1) IoT-embedded manufacturing enablers in the physical layer, (2) cloud-based data storage facility, data management and analytic systems in the cyber layer, (3) an ever-growing knowledge base in the cyber layer, and (4) arrangements regarding Digital Twins (DTs). These constituents interact with each other for performing the abovementioned cognitive tasks, whenever needed.

1.2 Context

The remarkable thing is that different types of DTs supply the knowledge for the ever-growing knowledge-base (see Figure 1.3). By definition, a DT is

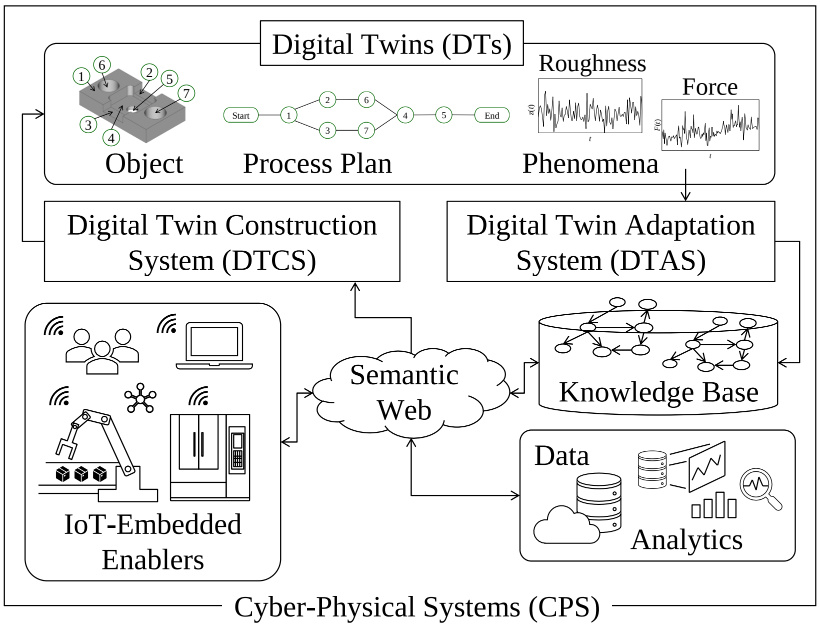


Figure 1.3: Role of digital twins in cyber-physical systems.

a computable virtual abstraction of a real-world entity, which must respond in real-time [8, 9]. Nevertheless, DTs can be classified into three types: (1) object twin, (2) process twin, and (3) phenomenon twin. An object twin means the DT of a real-world object (e.g., workpiece, cutting tool, machine tool, sensor, and alike) used in a given manufacturing environment. A process twin means the DT of a process sequence or process plan in a given manufacturing environment. A phenomenon twin means the DT of a machining phenomenon (e.g., surface roughness, cutting force, tool wear, cutting torque, and alike) in a given manufacturing environment. These twins collectively recreate the real-world manufacturing environment in the cyber layer, supply the relevant pieces of knowledge, and drive the abovementioned cognitive tasks (monitoring, understanding, predicting, decision-making, and adapting).

Now, DTs are not readily available. They must be developed. As such, two systems denoted as Digital Twin Construction System (DTCS) and Digital Twin Adaptation System (DTAS) must exist (see Figure 1.3). The DTCS constructs the relevant DTs and the DTAS uses the constructed DTs for functionalizing the cognitive tasks. Although many researchers have embarked on developing the object and process twins, phenomenon twin has not been studied in detail yet. This study fills this gap. One immediate question arises that how the DTs of manufacturing phenomena can be developed.

1.3 Scope

Manufacturing phenomena (e.g., surface roughness, cutting force, tool wear, cutting torque, and alike) are complex, non-linear, and exhibit stochastic features [10]. As such, it is a cumbersome task to develop the relevant DT analytically. One alternative way might be extracting the knowledge underlying the phenomena-relevant historic sensor signals and using the extracted knowledge. For this reason, the relevant DTCS and DTAS (see Figure 1.3) must construct the DT from the historic sensor signals and adapt the constructed DT for monitoring, respectively.

Now, the issues underlying semantic annotation and time latency (or delay) are significant for developing a sensor signal-based phenomenon twin. Time latency or delay [11, 12] occurs when sensor signals are exchanged through the CPS. As a result, the signal at its origin (e.g., machine tools) and signal received at the receiver-end (e.g., DT) differ. Moreover, many datasets of heterogeneous sensor signals are exchanged through IoT-based networks.

Hence, acquiring the right signals for constructing a twin is difficult and time-consuming. Semantic annotation-based representation [13] of sensor signals can solve this problem.

Thus, a sensor signal-based phenomenon twin must machine-learn the required knowledge to emulate the phenomenon from the relevant historical sensor signal datasets, seamlessly interact with the real-time sensor signals, handle the semantically annotated datasets stored in clouds, and accommodate the transmission delay or latency.

Based on the abovementioned requirements, this study presents the relevant systems (DTCS and DTAS) for constructing a phenomenon twin and adapting the constructed twin into a CPS. Both systems are developed using a Java^{TM}-based platform. The modular architectures of the systems are presented in detail. In addition, real-life machining torque signals are used to demonstrate the efficacy of the systems (DTCS and DTAS).

Therefore, the following questions are emphasized for developing the systems and demonstrating their efficacy.

1. How can sensor signals be annotated using user-defined semantics?
2. How does time latency/delay affect sensor signals' nature?
3. Which signal processing method can accommodate the effect of delay effectively?
4. What should be the architecture of DTCS to construct the DTs based on semantically annotated and time-delayed sensor signals?
5. What should be the architecture of DTAS to use the constructed DTs for real-time in-process monitoring?
6. How can the proposed DTCS and DTAS be developed?
7. Does DTCS construct reliable DTs from the sensor signals?
8. Does DTAS functionalize effective monitoring?

1.4 Aim and Objectives

The aim of this study is to develop two computerized systems (DTCS and DTAS) for constructing a DT of machining phenomenon (or phenomenon twin) based on semantically annotated and time-delayed sensor signals, and using the constructed DT for real-time monitoring.

As such, the underlying objectives are as follows.

1. To develop a semantic annotation-based representation mechanism,
2. To investigate the impact of delay on sensor signals' nature,
3. To determine the requirements and structure of the systems (DTCS and DTAS),
4. To develop the systems, and
5. To test the efficacy of the systems using a real-life machining process.

1.5 Contribution and Significance

The main contribution of this study is the systems, i.e., DTCS and DTAS, for constructing and adapting DTs of machining phenomena (e.g., cutting torque, cutting force, surface roughness, and alike). As described in Section 1.2, in the CPS of Industry 4.0, different types of DTs (object, process, and phenomenon twins) are needed for virtualizing a real-life manufacturing environment and functionalizing high-level cognitive tasks (monitoring, understanding, predicting, decision-making, and adapting). Although several researchers have been embarked on object and process twins, the phenomena twins (DTs of machining phenomena) have not been studied in detail yet. There is a lack of a steadfast procedure of constructing and adapting such DTs. This study fills this gap by presenting the required systems (DTCS and DTAS).

The systems are developed using a Java™-based platform. The efficacy of the systems is tested using a real-life machining process (milling) and milling torque signals. The systems are found effective and fast while constructing the relevant DT and performing real-time monitoring. Since the DTs of the machining phenomena are needed to make the machine tools and other programmable devices more intelligent and autonomous, the presented DTCS and DTAS provide a steadfast procedure for materializing the vision.

Having said that, the presented systems are capable of handling data transmission delay and semantic annotation-related issues, which are significant from the context of Industry 4.0 (as mentioned in Section 1.3). This study also elucidates these issues in detail.

It is seen that the delay affects sensor signals' nature. When delay is considered, the conventional signal processing methods (time and frequency domain-based processing) are inadequate for unfolding signals' nature. In such case, an alternative method, i.e., delay domain is found more effective, revealing the fact that computational arrangements in the CPS must accompany delay domain-based processing apart from others.

On the other hand, the presented semantic annotation-based representation mechanism ensures fast and effective data mining. It also ensures a scalable way for representing contents and creating linked data over the web. This means the mechanism ensures socialization among heterogeneous environments. This is significant from the viewpoint of Semantic Web (Web 3.0/4.0) because Industry 4.0 is expected to face the era of Semantic Web in the upcoming days.

Thus, this study contributes to the befitting advancement of Industry 4.0 or smart manufacturing.

1.6 Thesis Structure

This thesis is structured into eight (8) chapters as follows.

Chapter 1 presents the introduction of this study, describing the research background, context, scope, aim and objectives, and contribution and significance of this study.

Chapter 2 provides an extant literature review on the role of CPS, IoT, and DTs in Industry 4.0 or smart manufacturing.

Chapter 3 describes a semantic annotation-based representation mechanism of data and knowledge.

Chapter 4 describes the role of the delay domain in mitigating the effect of time delay or latency of signal transmission.

Chapter 5 presents the proposed DTCS and DTAS, describing the systems' context and modular architectures.

Chapter 6 demonstrates the efficacy of the DTCS and DTAS, using a real-life machining process (end milling and milling torque signals).

Chapter 7 discusses the implications of this study. This chapter also highlights some future research directions.

Finally, Chapter 8 provides the concluding remarks of this thesis.

Chapter 2

Literature Review

As described in Chapter 1, embedded systems called Cyber-Physical Systems (CPS) and Internet of Things (IoT) are two mainsprings for merging the physical and cyber worlds underlying a manufacturing environment. The goal is to functionalize high-level cognitive tasks, i.e., monitoring, understanding, predicting, decision-making, and adapting, for a better manufacturing experience. In this context, different types of Digital Twins (DTs) supply the required knowledge for achieving the abovementioned tasks.

Consequently, CPS, IoT, and DTs have created remarkable research directions in manufacturing research groups. This chapter presents an extant literature review on CPS, IoT, and DTs, as follows. Section 2.1 describes some of the recent articles on the role of CPS and IoT in Industry 4.0 or smart manufacturing. Section 2.2 describes some of the recent articles on the role of DTs in Industry 4.0 or smart manufacturing. Section 2.3 summarizes this chapter.

2.1 CPS and IoT in Industry 4.0

Many authors have embarked on CPS and IoT's implications in Industry 4.0 or smart manufacturing. Some of the recent articles are briefly described below.

Zhou et al. [4] described the conceptual framework of new generation intelligent manufacturing systems, in terms of a Human-Cyber-Physical Systems (HCPS). It (HCPS) hosts a self-growing knowledge base that integrates expert knowledge and machine intelligence through new generation artificial intelligence. Morteza [6] reviewed the extant literature of the Industry 4.0 ecosystem. The entities involved in the ecosystem are simulation and modeling, CPS, semantic technologies, IoT, Internet of Service (IoS), Internet of Data (IoD), cloud computing, big data analytics, block chain, cyber security, augmented reality, automation and industrial robotics, and additive manufacturing. The proximal and distal relationships among these entities achieve personalized product, corporate social responsibilities, smart factory, smart product, interoperability, modularization, decentralization, virtualization, real-time capacity, vertical integration, and horizontal integration. Rossit et al. [14] presented a dynamic scheduling architecture based on data-driven procedures in smart manufacturing environments. Both vertical and horizontal data integration in the CPS to make the dynamic scheduling achievable. The vertical allows establishing direct contact between the physical and decision-making levels and horizontal integration of CPS-driven business functions traditionally carried out independently. Big data and other relevant technologies become the backbone of dynamic scheduling architecture due to its data dependency. Lu [15] presented a state-of-the-art survey of the ongoing research on Industry 4.0, elucidating its content, scope, and findings. This study emphasizes that the symbiosis of machine-to-machine communication and machine-to-human collaboration will emerge in Industry 4.0. The integration of engineering processes, business processes, and service processes creates a new enterprise information system architecture for Industry 4.0. Oztemel and Gursev [16] provided a literature review on Industry 4.0 and related technologies. They have emphasized the harmonized roles of humans and machines in Industry 4.0, where machines perform monotonous and recurrent tasks, opening more opportunities for humans to perform creative tasks. Developing a taxonomy is a pressing need in Industry 4.0. Monostori et al. [5] elucidated the structure, functions, and roles of the CPS in manufacturing science and technology. They emphasized that CPS evolve due to the advent of computer science and information and communication technologies. It will bring all stakeholders and aiding manufacturing systems into a networked type of integration, breaking the hierarchies. Further research, development, and implementation activities are needed to give a concrete shape to manufacturing CPS where the socio-ethical aspects must not be neglected. Yao et al. [17] described the eight-tuple structure of CPS with the characteristics of real-time data access, reconfiguration, inter-operation, decentralized decision-making, intelligence, and pro-activity to overcome the limitations of existing integrated manufacturing systems. They have also emphasized that the eight-tuple structure must be upgraded to a nine-tuple structure, adding the concept of wisdom manufacturing, incorporating the functionalities of social computing, community, crowd-sourcing, customization/personalization, innovation, and sustainability. Lee et al. [18] provided a pragmatic 5C architecture of the CPS in manufacturing consisting of hierarchically organized five layers called connection, conversion, cyber, cognition, and configuration. This architecture helps develop more intelligent and resilient manufacturing equipment and helps achieve better product quality and system reliability. Lu and Cecil [7] outlined the IoT-based collaborative framework, which serves as the foundation for cyber-physical interactions and collaborations in Industry 4.0. They have introduced a language called engineering enterprise modeling language to materialize the exchange of data and information among the IoT-embedded devices. Pang et al. [19] developed a data-driven intelligent decision-making method for facilitating quality control in CPS. Different machine learning techniques (e.g., artificial neural network) must be incorporated to perform the quality control tasks in a predictive way. Nakayama et al. [20] described how Industry 4.0 evolves its predecessor (Industry 3.0). They have emphasized that achieving vertical and horizontal integration with the aid of Industrial Internet of Things (IIoT), service systems, semantic web, artificial intelligence, and collaborative networks can transform Industry 3.0 to Industry 4.0. Morgan et al. [21] revisited re-configurable manufacturing from the context of Industry 4.0. They developed a three layered modularity framework of cyber-physical systems and IIoT framework, consisting of physical control, cyber control and artificial intelligence. Physical control (edge) and cyber control (fog) layers are connected by industrial network wherein DTs play their roles. Cyber control and artificial intelligence (cloud) are connected by enterprise network where services play their roles.

In sum, CPS hosts IoT-embedded manufacturing enablers, cloud-based data storage and management system, and an ever-growing knowledge-base. And, the DTs supply the knowledge for the ever-growing knowledge-base (can also be seen in Figure 1.3).

2.2 Role of DTs in Industry 4.0

As mentioned above, DTs supply the knowledge for the ever-growing knowledge base resided in the CPS. This eventually functionalizes the high-level cognitive tasks (monitoring, understanding, predicting, decision-making, and adapting) in an Industry 4.0 environment. Many authors have been embarked on DTs' conceptualization, systematization, construction, and adaptation from the context of Industry 4.0 and beyond. Some of the recent articles are described below.

Aheleroff et al. [8] described DT with respect to digital model and shadow. The digital model has no real-time connectivity with its physical counterpart, but the digital shadow has limited or one-directional connectivity with its physical counterpart. On the other hand, DT has bi-directional real-time connectivity with its physical counterpart. It has been stressed that DT needs semantically annotated content for fulfilling its bidirectional real-time connectivity. Note that, there is another relevant concept known as a digital thread. This concept is synonymous with the digital model or digital shadow where the model is product-life-cycle-aware, that is, the model is semantically annotated in terms of its role in the product-life-cycle. Fuller et al. [22] reviewed the recent developments regarding DTs from different viewpoints (manufacturing, healthcare, and smart cities). They described that a DT incorporates data analytics, artificial intelligence (AI)-based modeling, simulation, and visualization for decision-making. They suggested that the DT might include validation measures to ensure the trustworthiness of the AI-generated models before putting them into practice. Kaur et al. [23] mentioned that a DT machine learns from real-time sensory data for forecasting the future of the corresponding physical counterparts. Jiang et al. [24] described that the DTs drive value-added services (intelligent operation and maintenance, monitoring, and optimization of assets) in IoT-embedded industrial environment. For this, they proposed a Data-Model-Service-based framework (DMS framework) for developing the DTs. The Industrial Internet Consortium (IIC) [25] articulated that the DTs can be designed discretely (relevant to a single entity in a real-life manufacturing environment) for monitoring purposes. The discrete DTs can also be composed to create a composite DT for realizing the overall manufacturing environment. Nevertheless, the DTs acquire data (physical objects' data, time-series data, historical data, and alike) for modeling physical entities. The models represent the behavior of those entities by incorporating machine learning and simulation techniques. The IIC also highlighted that the DTs might contain a set of human-comprehensive interfaces for putting them (DTs) into practice in industrial applications and making them available to other stakeholders whenever required. Tao et al. [26] have presented a DT-driven approach for product development. The approach incorporates information related to developing a product (design, design requirements, process, historical data in the form of big data, and other associated factors such as environmental factors, market review, and customer feedback). It improves the product design and optimizes relevant production and process plans. Ruppert et al. [27] described that the real-time sensory data-driven DTs can be integrated with the CPS for monitoring the status of the enablers and relevant decision-making. They also introduced a DT framework called Real-Time Locating Systems (RTLS)-based DT for production scheduling. Aheleroff et al. [28] stressed the role of DT for Mass Personalization Manufacturing (MPM). They described that DT promises a high degree of personalization with the aid of IoT, artificial intelligence (AI), additive manufacturing (AM), and cloud-based technologies in the smart manufacturing paradigm. They also proposed a three-dimensional DT reference model consisting of DT architecture, agile product development, and DT integration hierarchy.

However, DTs can make a machine tool intelligent [29, 30, 31]. As a result, DT-embedded machine tools monitor and troubleshoot their activities autonomously. For this purpose, sensor signal-based DTs (phenomena twins) must be developed and adapted into the CPS apart from other types of DTs: object and process twins (see Figure 1.3).

2.3 Summary

In sum, sensor signal-based DTs or phenomena twins are needed to make machine tools intelligent, and functionalize monitoring and autonomous troubleshooting. However, such DTs have not been studied in detail yet. This study fills this gap by addressing its development from historic sensor signal datasets.

Since the development requires the right piece of sensor signal dataset from a storage facility (e.g., cloud)—containing heterogeneous contents from heterogeneous sources, and Industry 4.0 is embracing semantic web (web 3.0/4.0)-based technologies, the effective representation of sensor signals is a relevant research question. This study first answers this question by presenting a semantic representation of data and knowledge, as follows.

Chapter 3

Semantic Annotation

This chapter describes semantic annotation-based representation for sharing and reusing manufacturing contents (e.g., sensor signal datasets, cutting conditions, and alike) over the web. For better understanding, the chapter is organized as follows. Section 3.1 briefly describes the significance of semantic annotation. Section 3.2 describes some of the recent studies on manufacturing knowledge representation mechanisms. Section 3.3 presents a semantic annotation-based representation mechanism of data and knowledge. Section 3.4 presents an example, where experimental results underlying a machining experiment is represented using the presented mechanism in Section 3.3. Finally, Section 3.5 summarizes this chapter.

3.1 Significance of Semantic Annotation

As summarized in Chapter 2, acquiring the right piece of sensor signal dataset among many from a storage facility (e.g., cloud) is the first-most concern to develop a DT of a machining phenomenon. For this, an effective and meaningful representation mechanism is needed so that enablers (e.g., human, monitoring systems, DTs, and alike) can share and reuse the experimental results (e.g., sensor signal datasets, cutting conditions, and alike) over the web (web 2.0, or web 3.0/4.0), whenever required. One way to achieve this is using user-defined semantics to link all the relevant entities while sharing a piece of content. This will create linked data of all essential entities and help other stakeholders understand the overall context before reusing the shared content. Thus, semantic annotations can facilitate effective data mining.

Moreover, as seen in Figure 3.1, Industry 4.0 is expected to face the era of semantic web (web 3.0/4.0) [32, 13] in the upcoming days. Therefore, the representation mechanism must create human and machine-comprehensible linked data by putting semantic annotations on top of syntax [33] for the sake of socialization. Here, syntax means the codified contents for enabling Machine-to-Machine (M2M) communications. On the other hand, semantics mean the meaning of the contents. The semantics associated with the syntax provides a meaningful way to adapt the contents (experimental results underlying a manufacturing activity) whenever required. However, in reality, representation mechanisms are mostly domain-specific, strict ontology- and query language-dependent (e.g., SOSA ontology and SPARQL/C-SPARQL queries), and thus, esoteric by nature.

Nevertheless, several researchers have been embarked on representation mechanisms and the importance of semantic annotations. Some of the recent articles are briefly described in the following section.

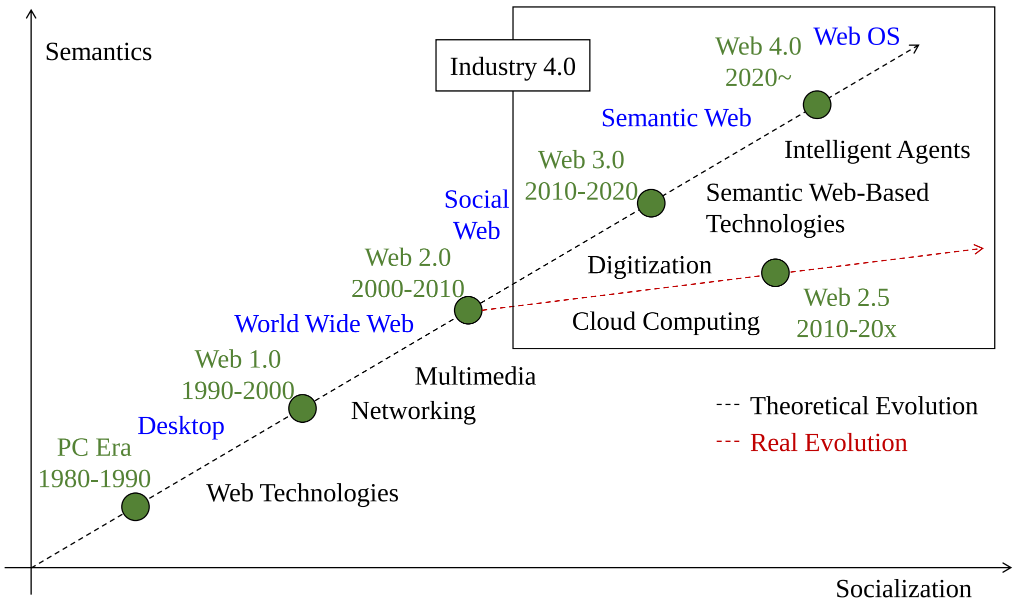


Figure 3.1: Evolution of web and its context in Industry 4.0 (rearranged from the work in [34]).

3.2 Studies on Representation Mechanism

Jaskó et al. [35] articulated that Manufacturing Execution Systems (MESs) must represent knowledge in the form of semantic metadata-embedded linked data for exchanging contextual information among all the enablers in the Cyber-Physical Systems (CPS). Mostafa et al. [36] stressed the importance of metadata for acquiring contextual information from big data. Souri et al. [37] described that knowledge management frameworks must handle the discrete nature of data from heterogeneous sources. The frameworks must also integrate the multi-source data for easy access and reuse. For this, they proposed a framework that semantically classifies engineering design and manufacturing knowledge. Wang et al. [38] introduced a holistic tool that acquires, organizes, and fuses knowledge from heterogeneous sources for improving process innovation. For this, it uses a semantic relationship-based knowledge fusion technique. Liu et al. [39] stressed that semantic information associated with multi-source data drives the development of intelligent monitoring systems (e.g., digital twins). Ullah [10] presented a semantic model of sensor signal-based digital twins. The author also represented the underlying knowledge from the viewpoint of the semantic web (Web 3.0/4.0) [32], which is most likely to dominate in the upcoming days and functionalize the CPS. Likewise, Fenza et al. [40] described manufacturing knowledge representation from the viewpoint of the semantic web. They articulated that the knowledge must be semantically enriched and machine-readable for the sake of knowledge-sharing among heterogeneous independent industrial environments. They also proposed a method that collects, processes, and represents the semantic data, using the SOSA ontology and C-SPARQL queries. Ullah [41] argued that knowledge representation must be both human-comprehensible and machine-readable, incorporating syntax and user-defined semantics. Pomp et al. [42] stressed the importance of semantics for reducing data-analytic-time. They also introduced a semantic data platform called Evolving Semantic Knowledge Aggregation and Processing Engine (ESKAPE) to semantically annotate raw data (collected from heterogeneous sources), resulting in a continuously evolving knowledge graph. Fill [43] proposed a semantic annotation-driven enterprise modeling architecture for the sake of risk management. Ullah [13] emphasized the importance of interpreting manufacturing knowledge before its digitization and representation. Accordingly, the author interpreted manufacturing knowledge by three elements called knowledge claim, provenance, and inference. The author also articulated that knowledge representation mechanisms must be scalable and user-friendly instead of domain-specific and esoteric.

In sum, knowledge must be both human-comprehensible and machine-readable. For this, its representation must incorporate semantics on top of syntax. The representation must follow a scalable ontology for integrating contents from heterogeneous sources, handling the discrete nature of data, creating linked data, and sharing it among heterogeneous independent manufacturing environments. However, in reality, the representation mechanisms are mostly domain-, strict ontology-, and query language-dependent (e.g., SOSA ontology, SPARQL/C-SPARQL queries, and alike). Thus, the representation mechanisms are esoteric by nature. For this, a flexible and user-friendly human and machine comprehensible representation mechanism is presented in this study, as follows.

3.3 Proposed Representation Mechanism

As seen in Figure 3.2, whenever a manufacturing activity (e.g., a machining experiment) happens, it entails different entities (e.g., sensors, cutting tool, machining tool, machining condition, workpiece material, signal datasets, and alike). The entities collectively build up the knowledge underlying the

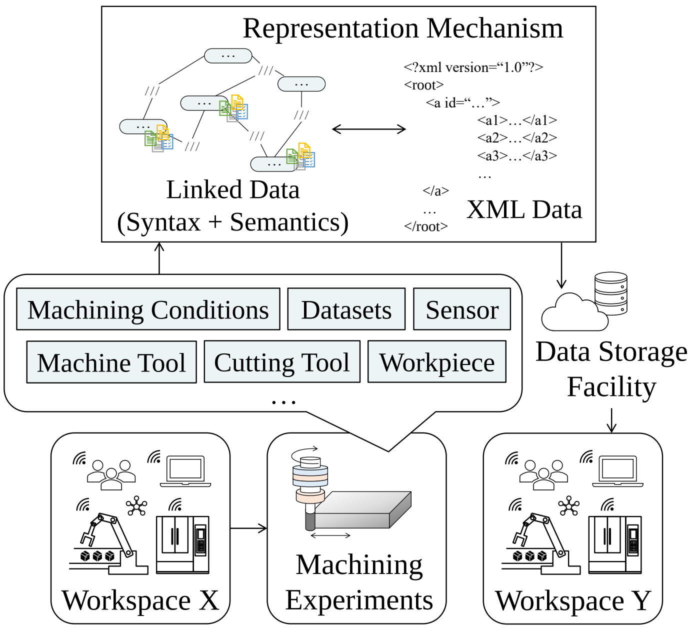


Figure 3.2: Outline of semantic annotation-based knowledge representation mechanism.

experiment. Hence, the knowledge representation mechanism must reflect rational relationship among the entities.

One pragmatic way to achieve this is to create linked data by incorporating user-defined semantics among the entities' syntax. As such, the semantic annotation process must be flexible and easy-to-comprehend considering the heterogeneous structure of Industry 4.0. Afterwards, the semantically annotated linked data can be shared in the form of XML (Extensible Markup Language) data in a data storage facility (e.g., cloud) so that other stakeholders can access, understand, reuse a piece of experimental results (e.g., sensor signal datasets), whenever required. This scenario is also schematically illustrated in Figure 3.2. For the sake of better understanding, consider an example as described below.

3.4 Use of the Proposed Mechanism

3.4.1 Machining Experiment

As seen in Figure 3.3, consider a machining experiment, where a bimetallic workpiece is machined for understanding the cutting torque behavior. The workpiece is made of commercially pure Titanium and aluminum (Japanese Industrial Standards or JIS: A1070), denoted as 'Ti' and 'Al', respectively. The corresponding cutting conditions (e.g., cutting velocity, spindle speed, feed rate, axial depth of cut, and alike) are also mentioned in Figure 3.3 in detail. The corresponding machining equipment (e.g., cutting tool, machine tool, sensor, and alike) and cutting directions are summarized in Table 3.1. Here, the cutting conditions are pieces of definitional knowledge [13], because these are well-defined in the manufacturing domain. On the other hand, the numerical values of the conditions and the machining equipment are pieces of inductive knowledge [13], because one may formulate these in different ways based on experience or requirements.

As mentioned in Table 3.1 (can also be seen in Figure 3.3), the workpiece is machined from two directions: hard-to-soft material direction, i.e., Ti to Al, and soft-to-hard material direction, i.e., Al to Ti. Say, the former direction is denoted as 'direction-1' and the latter one is denoted as 'direction-2'. As such, a rotary dynamometer (see Table 3.1 for details) is used for measuring cutting torque signals, while machining the workpiece from both directions. Figure 3.4 shows the time series plots of the measured torque signals.

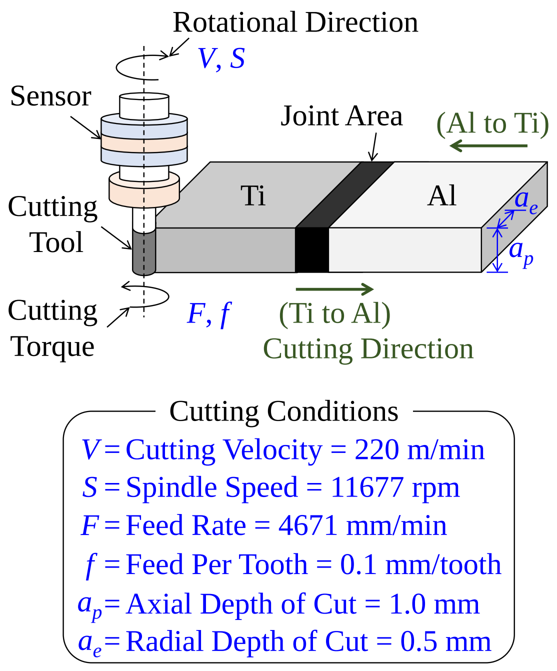
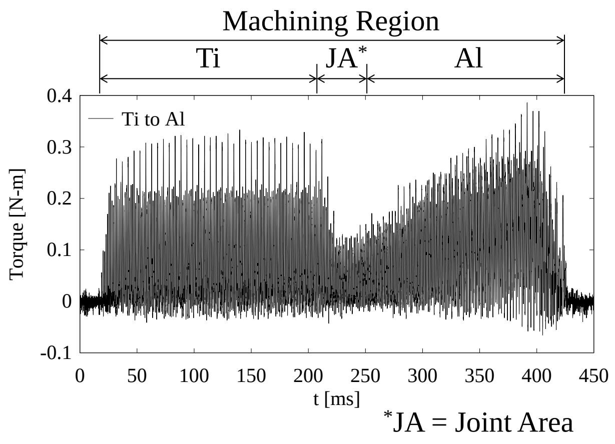

Rotational Direction
V, S
Sensor
Joint Area
(Al to Ti)
Ti
Al
Cutting Tool
a_e
a_p
Cutting Torque
F, f
(Ti to Al)
Cutting Direction
Cutting Conditions
V = Cutting Velocity = 220 m/min
S = Spindle Speed = 11677 rpm
F = Feed Rate = 4671 mm/min
f = Feed Per Tooth = 0.1 mm/tooth
a_p = Axial Depth of Cut = 1.0 mm
a_e = Radial Depth of Cut = 0.5 mm

Figure 3.3: A machining experiment and underlying cutting conditions.

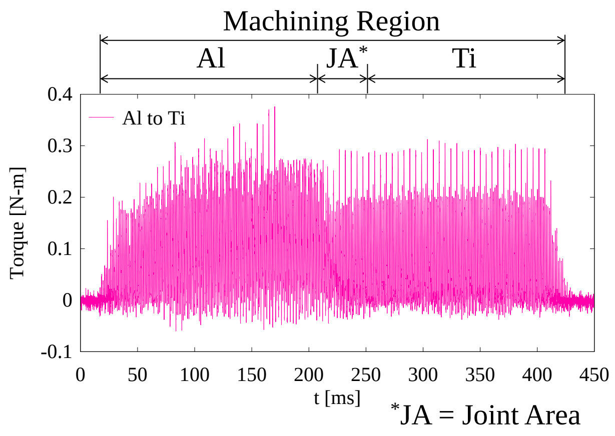
Table 3.1: Conditions for the machining experiment.

Item	**Description**
Machine tool	Vertical machining center Make: Mori Seiki Model: NV5000
Cutting tool	Carbide $\Phi 6$ solid end mill Make: Mitsubishi Hitachi Model: EPP4060-P-CS
Sensor	Rotary dynamometer Make: Kistler Type: 9170A
Workpiece material	Bimetallic material made of commercially pure Titanium (Ti) and Aluminum, JIS: A1070 (Al)
Cutting directions	Ti to Al, Al to Ti

As seen in Figure 3.4a, for direction-1 (Ti to Al), the torque signals first remain stable, then go downward, and finally increase continuously while machining the Ti, joint area (heat-affected area while joining Ti and Al, see Figure 5), and Al, respectively. On the other hand, as seen in Figure 3.4b, for direction-2 (Al to Ti), the torque signals remain somewhat stable all through the machining region. Since unstable cutting torque is undesired in machining, it can be contemplated that the machining of the given workpiece must follow direction-2 (Al to Ti) instead of direction-1 (Ti to Al), considering the given cutting conditions. As such, the abovementioned contemplation is a piece of inductive knowledge [13], because it is experimental data-dependent and made by analyzing the obtained results.



(a)



(b)

Figure 3.4: Measured cutting torque signals. (a) cutting direction: Ti to Al, (b) cutting direction: Al to Ti.

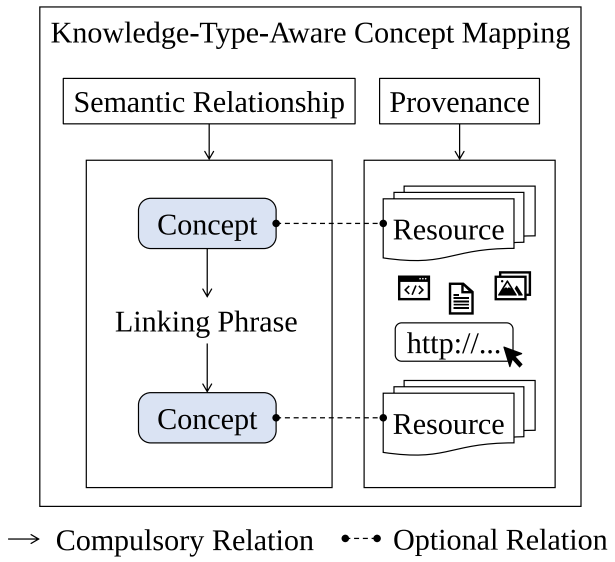


Figure 3.5: Outline of knowledge-type-aware concept mapping.

Now, the abovementioned knowledge underlying the experiment are represented in the form of linked data incorporating all the entities (e.g., cutting conditions, sensor signal datasets, observations, and alike) associated with some user-defined semantics. One straightforward way to do this is knowledge-type-aware-concept mapping [10, 13, 41], as follows.

3.4.2 Knowledge-Type-Aware Concept Mapping

A concept map consists of at least one focus question Q, a set of semantic relationships or propositions $P = \{P_i \,|\, i = 1, 2, ...\}$, a set of concepts $C = \{C_j \,|\, j = 1, 2, ...\}$, a set of linking phrases $L = \{L_k \,|\, k = 1, 2, ...\}$, and a set of resources $R = \{R_l \,|\, l = 1, 2, ...\}$. A semantic relationship is constructed by relating at least two concepts via one linking phrase. Figure 3.5 schematically illustrates this scenario. As seen in Figure 3.5, a concept may or may not be resource-embedded. Here, resource means datasets, documents, images, URL(s), and alike. The resources provide the provenance (a guide to the truthiness) for the constructed relationships. As such, the sets P, C, and L can never be empty, but R can.

Now, recall the abovementioned machining experiment (see Section 3.4.1). A concept map is constructed (using CmapTools, developed by Florida Institute for Human & Machine Cognition (IHMC), available from the following

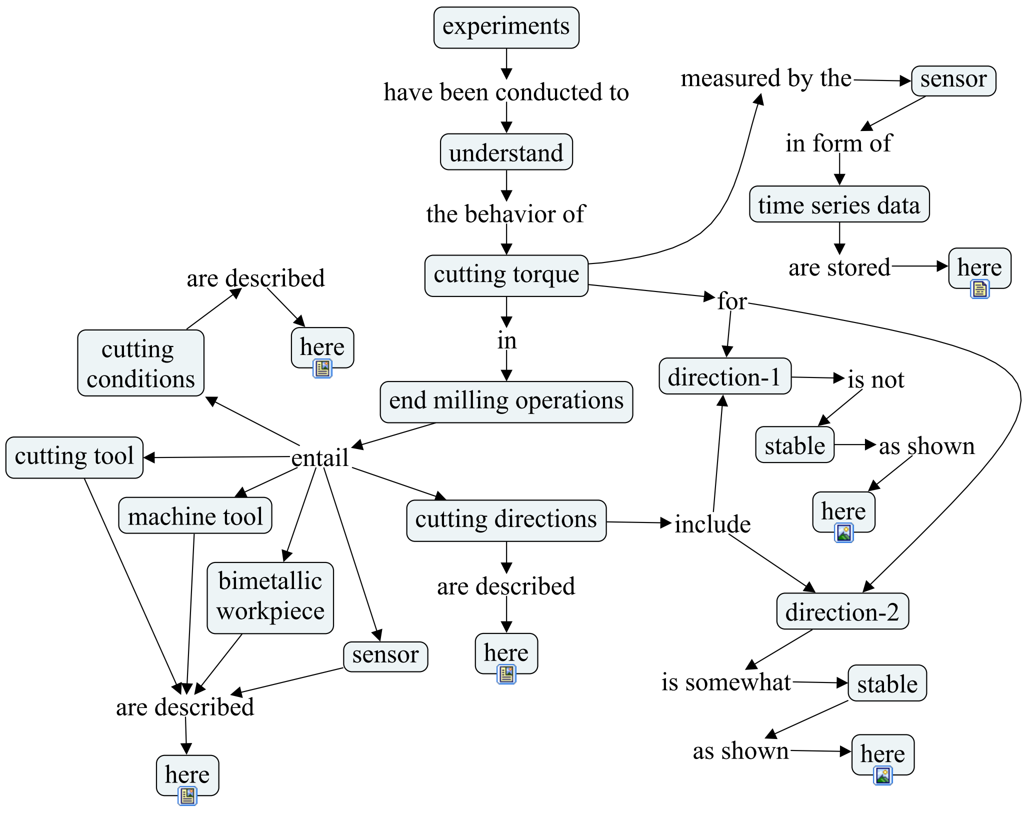


Figure 3.6: A concept map for the machining experiment.

url: https://cmap.ihmc.us/) for representing the knowledge underlying the experiment, as shown in Figure 3.6.

As seen in Figure 3.6, the constituents of the concept map are as follows: Q = "Behavior of cutting torque in end milling operations", C = {experiments, understand, cutting torque, end milling operations, cutting directions, here, cutting conditions, time series data, direction-1, direction-2, cutting tool, machine tool, sensor, bimetallic workpiece, stable}, L = {have been conducted to, the behavior of, in, entail, are described, as shown, include, measured by the, in form of, are stored, is not, is somewhat, is described, for}, and $R = \{R_1, ..., R_6\}$.

As such, the concept map shown in Figure 3.6 boils down to 9 (nine) propositions, $P = \{P_1, ..., P_9\}$, as follows:

(P_1) experiments have been conducted to understand the behavior of cutting torque in end milling operations.

(P_2) end milling operations entail cutting conditions, cutting tool, machine tool, bimetallic workpiece, sensor and cutting directions.

(P_3) cutting conditions are described here.

(P_4) cutting tool, machine tool, bimetallic workpiece and sensor are described here.

(P_5) cutting directions are described here.

(P_6) cutting directions include direction-1 and direction-2.

(P_7) cutting torque for direction-1 is not stable as shown here.

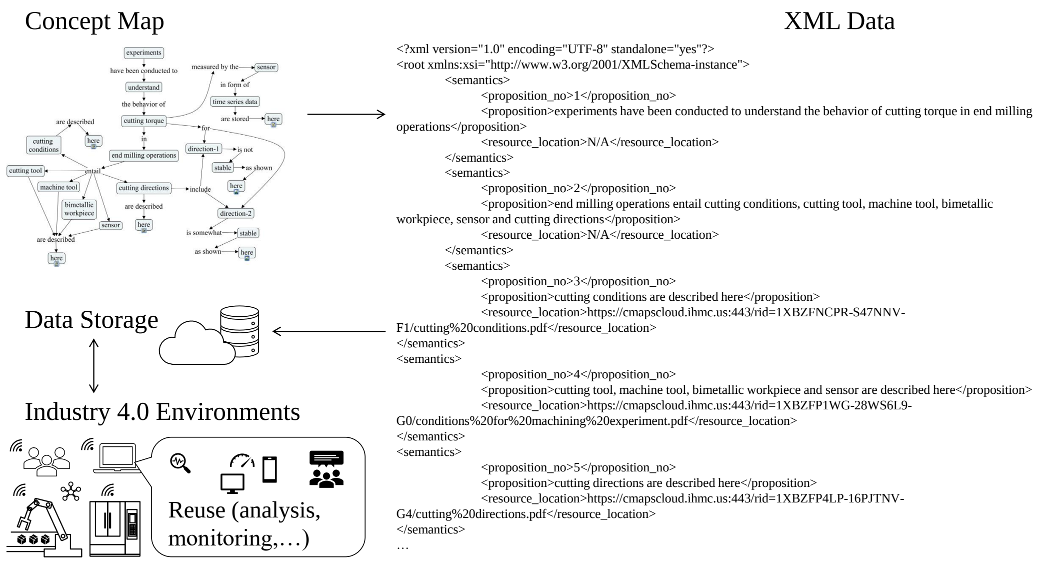
(P_8) cutting torque for direction-2 is somewhat stable as shown here.

(P_9) cutting torque measured by the sensor in form of time series data are stored here.

In P_3, P_4, P_5, P_7, P_8, and P_9, the concept called 'here' is embedded with $R_1, ..., R_6$, respectively. Here, R_1 refers to a document describing the cutting conditions (see Figure 3.3). R_2 refers to a document describing the machining equipment (see Table 3.1). R_3 refers to a document describing the cutting directions (see Figure 3.3 and Table 3.1). R_4 refers to the time series plot of torque signals for direction-1 (see Figure 3.4a). R_5 refers to the time series plot of torque signals for direction-2 (see Figure 3.4b). R_6 refers to a document (format: text document, .txt) storing the numerical datasets of the measured torque signals. One may view the concept map from the following url: `https://cmapscloud.ihmc.us/viewer/cmap/1XBZFP7CV-NCSQ86-G9`.

One remarkable thing about concept mapping is that the sets P, C, and L are user-defined. For example, one can use different sets of P, C, and L for constructing the same concept map shown in Figure 3.6. This makes knowledge representation using a concept map highly flexible and user-friendly. In addition, one can embed R with P for the sake of user comprehensibility and meaningful representation.

Now, this concept map (see Figure 3.6) is essentially a high-level description of a machining experiment (see Figure 3.3), which is human comprehensible. However, in Industry 4.0 environments, other enablers and stakeholders (machines and systems) are also expected to access the concept map and make sense of it. For this, the concept map can be shared among all the stakeholders in various forms over the web. One straightforward way is sharing in the form of extensible Markup Language or XML, as shown in Figure 3.7. This functionalizes other stakeholders to understand the overall context underlying a manufacturing experiment in a more effective manner,



```xml
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<root xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
        <semantics>
                <proposition_no>1</proposition_no>
                <proposition>experiments have been conducted to understand the behavior of cutting torque in end milling
operations</proposition>
                <resource_location>N/A</resource_location>
        </semantics>
        <semantics>
                <proposition_no>2</proposition_no>
                <proposition>end milling operations entail cutting conditions, cutting tool, machine tool, bimetallic
workpiece, sensor and cutting directions</proposition>
                <resource_location>N/A</resource_location>
        </semantics>
        <semantics>
                <proposition_no>3</proposition_no>
                <proposition>cutting conditions are described here</proposition>
                <resource_location>https://cmapscloud.ihmc.us:443/rid=1XBZFNCPR-S47NNV-
F1/cutting%20conditions.pdf</resource_location>
</semantics>
<semantics>
                <proposition_no>4</proposition_no>
                <proposition>cutting tool, machine tool, bimetallic workpiece and sensor are described here</proposition>
                <resource_location>https://cmapscloud.ihmc.us:443/rid=1XBZFP1WG-28WS6L9-
G0/conditions%20for%20machining%20experiment.pdf</resource_location>
</semantics>
<semantics>
                <proposition_no>5</proposition_no>
                <proposition>cutting directions are described here</proposition>
                <resource_location>https://cmapscloud.ihmc.us:443/rid=1XBZFP4LP-16PJTNV-
G4/cutting%20directions.pdf</resource_location>
</semantics>
…
```

Figure 3.7: XML data of the concept map.

before reusing a piece of the experimental results (here, numerical datasets of cutting torque signals).

3.5 Summary

This chapter presents a semantic annotation-based knowledge representation mechanism for a machining experiment. It (mechanism) semantically annotates the relevant entities (e.g., cutting conditions, machine tool, cutting tool, sensor, workpiece, sensor signal datasets, and alike) for representing different knowledge-types underlying the experiment, and results in human and machine comprehensible linked data (concept map and XML data). The mechanism is highly flexible and user-friendly compared to the conventional domain-specific and esoteric representation mechanisms. Its use is also demonstrated in detail considering a real-life machining experiment.

As such, the findings suggest that the semantic annotation-based representation helps other stakeholders (e.g., human resources and monitoring systems) understand the overall context underlying the experiment, before reusing a piece of experimental results (e.g., sensor signal datasets). This is particularly significant for developing DTs of machining phenomena (e.g., cutting force, cutting torque, surface roughness, and alike). The DTs need phenomenon-relevant historical sensor signal datasets for modeling the phenomenon and generating simulated signals. As such, the DTs first need to understand the usability of the available signal datasets for acquiring the right piece of signal dataset among many from the web. The proposed representation mechanism solves this issue in an effective way.

Nevertheless, after acquiring the appropriate sensor signal dataset from a data storage facility, the dynamics underlying the signal dataset must be encapsulated for developing the DT. This means an appropriate sensor signal processing method must be incorporated in the DTCS. Since Industry 4.0-relevant communication networks face time latency or delay while transmitting sensor signals among different systems, its (time latency or delay) effect on sensor signals needs detailed investigation for identifying the appropriate signal processing method. The following chapter describes this issue in detail.

Chapter 4

Delay and Its Significance

As described in Chapter 1, sensor signals are subjected to time latency or delay [11, 12] while being transmitted among different systems in the CPS. As such, the interplay of delay must be considered and investigated in detail for developing systems (DTCS and DTAS), required for constructing and adapting a sensor signal-based DT of machining phenomenon in the CPS.

When delay is considered, the signal analyses (e.g., time and frequency domains-based analyses) may not adequately capture the dynamics of the underlying phenomenon. Instead, a domain called the delay domain [44, 45] can be considered for capturing the dynamics of the underlying phenomenon. Based on this consideration, this chapter presents the effect of delay on sensor signal processing in detail, proving some guidelines showing how to accommodate delay in developing sensor signal-based DTs.

For better understanding, the rest of this chapter is organized as follows. Sections 4.1 and 4.2 describe some state-of-the-art studies on delay and sensor signal processing methods, respectively. Section 4.3 describes delay domain-based signal processing. Section 4.4 presents the implications of delay using an arbitrary signal and also some real-life machining signals, where delay domain-based processing is deployed to make sense of the signals. Section 4.5 presents the efficacy of delay domain-based processing where it is shown that the delay domain can distinguish different machining situations more effectively than the frequency domain. Finally, Section 4.6 summarizes this chapter.

4.1 Studies on Time Latency or Delay

Many researchers active in communication and control engineering have been working on delay from the context of Industry 4.0-relevant communication networks, focusing on its impacts and compensation mechanisms. Some of the recent articles are briefly described below.

Raptis et al. [46] reviewed the data delivery delay and its compensation for the sake of data management (coordination and computation of data) in Industry 4.0. Zoppi et al. [47] developed a Quality of Service (QoS) framework for mitigating end-to-end delay in industrial Wired/Wireless Sensor Networks (WSN). Mo et al. [48] focused on the performance of the Networked Control Systems (NCS) due to delay. They showed how delay impedes the reliability of NCS. Zhang et al. [49] proposed a delay compensation algorithm for the NCS using a method called Controller Area Network (CAN)-buses. Guck et al. [50] introduced a Software-Defined Networking (SDN)-based function split framework for reducing delay. It (SDN-based framework) meets the end-to-end delay requirements, guaranteeing the real-time QoS of data exchange. Yagi and Sawada [51] articulated that delay underlying a communication network is random. It causes instability in the feedback systems. They also designed a Kalman filter to reduce the effect of the random delay. Fan et al. [52] articulated that a random delay underlying the NCS degrades the control performance and makes the system unstable. In this regard, they presented a control scheme called Networked Predictive Control (NPC). It mitigates the effects of transmission delay and achieves desired control performance using two components: Network Delay Compensator (NDC) and Control Prediction Generator (CPG), respectively. Wu et al. [53] described sensor-to-actuator and controller-to-actuator delays using Markov chains. They also proposed a control scheme that compensates these delays and makes the NCS stable. Sun and Huo [54] developed a switching control approach called Markovian Jump Linear System (MJLS). It models the random delays as a Markov process for making the NCS stable. Guo and Gu [55] described that random time delay is the cause of instability and poor performance in the NCS. They also suggested a model to improve stability. Baillieul and Antsaklis [56] described that delay is unavoidable and one of the challenges of modern NCS composed of heterogeneous systems and applications. They also mentioned that the sensors might fail to transmit data immediately, and data loss might occur due to communication delays. They considered that distributed control systems perform better in overcoming delay-related issues compared to the centralized ones. Bijami and Farsangi [57] described that communication delay—underlying distributed networked systems composed of heterogeneous sub-systems—causes random data loss and makes the system unstable. They also developed a control framework for stabilizing the networked systems. Zunino et al. [58] mentioned that delay compensation is needed to transmit data at a higher update rate for materializing the real-time behavior of Industry 4.0. They also emphasized that systems and applications in Industry 4.0 need to be delay-aware for addressing the time-related performance and reliability requirements. Ferrari [12] identified different types of delays in an IoT-based environment, such as end-to-end delay, round-trip-time, and jitter. These delays cause low data delivery rate, data loss, and failure of the control systems. The author provided a hardware-independent method for measuring delays in an industrial IoT-based environment. Jhaveri et al. [11] presented a delay-aware framework to measure end-to-end delay in real-time SDN-based networks. Kontogiannis and Kokkonis [59] proposed a protocol called Fuzzy Real-Time haPticS Protocol (FRTPS) for alleviating the impacts of delay in real-time applications. It (FRTPS) alleviates data loss and low data delivery for achieving better synchronization and reliability of the systems. Xia et al. [60] proposed an algorithm called Mixed-Criticality Relative-execution Deadline (MCRD) for managing delay, scheduling of data transmission, and reducing data loss in the Industrial Internet of Things (IIoT)-based environments. Basir et al. [61] mentioned that fog computing performs better than cloud computing for compensating delay-related issues. It also achieves low latency data delivery and reliable real-time communication among heterogeneous systems in the IIoT-based environments. They also mentioned that the systems and applications in the IIoT need to be delay-aware to limit different types of delays (e.g., processing, propagation, transmission, and computation delays) while communicating in real-time. Wang et al. [62] emphasized that communication delay needs to be considered while designing controllers to monitor and control in the NCS.

In sum, delay causes data loss and system instability in a given communication and data transmission network. This is true for the networks in CPS. For this, its effect on sensor signal processing needs to be understood. In this respect, first some of the recent works on sensor signal processing are studied, as follows.

4.2 Studies on Signal Processing Methods

Sensor signal processing is a mainspring for machine- and process-condition monitoring in manufacturing. Its role has been intensified due to the advent of Industry 4.0-centric embedded systems (CPS and DTs). Numerous researchers have been working on implementing the existing signal processing techniques or even developing new ones. For the sake of better understanding, some of the recent articles on signal processing are briefly described below.

First, consider the techniques used in sensor signal processing for manufacturing. Some of the recent works are summarized in Table 4.1. As seen in Table 4.1, the signals are commonly processed in the time, frequency, and time-frequency (e.g., WPT and WTMM) domains for extracting domain-relevant features (attributes of signal data). Since this provides an overwhelming number of features, the most relevant ones are selected using some computational arrangements (e.g., Pearson's correlation coefficient [70], Spearman's correlation coefficient [74], and alike). The enablers (human and intelligent systems) use the relevant features for machine learning and monitoring purposes (e.g., chatter monitoring, tool condition monitoring, roughness assessment, and alike). Nevertheless, numerous authors have also studied the technical problems and limitations of the abovementioned sensor signal processing techniques. Some of the noteworthy studies are briefly described as follows.

Mahata et al. [67] articulated that time or frequency domain-based signal processing is not effective for analyzing non-linear signals (e.g., signals underlying grinding wheel wear). In this respect, they proposed a modified method defined as Hilbert-Huang Transform (HHT). Espinosa et al. [75] articulated that traditional frequency analysis is not effective for unfolding the characteristics of non-linear signals, compared to the alternative analytics such as Approximate Entropy (ApEn) and Sampling Entropy (SampEn). Bayma et al. [76] proposed a Non-linear Output Frequency Response Functions (NOFRFs)-based approach for analyzing non-linear systems from the contexts of condition monitoring, fault diagnosis, and non-linear modal analysis. Bernard et al. [77] articulated that in the machine and process-condition monitoring research areas, the methods used or introduced highly depend on high data acquisition rates. They emphasized the need for alternative methods that can perform with low data acquisition rates to meet some challenges of intelligent manufacturing, such as fast computation and low data storage. They also proposed a data-driven KDE function-based method for

Table 4.1: A summary of recent studies on signal processing methods in manufacturing.

Article	Process	Purpose	Signal	Method
[63]	Milling	Tool Condition Monitoring (TCM)	Cutting force, Vibration	Frequency, Time-Frequency
[64]	Milling	TCM	Sound	Wavelet Transform Modulus Maxima (WTMM)
[65]	Milling	Chatter monitoring	Cutting force	Time, Time-Frequency
[66]	Grinding	Grinding wheel condition monitoring	Acoustic Emission (AE)	Time-Frequency
[67]	Grinding	Grinding wheel wear identification	Vibration, Power	Frequency, Time-Frequency
[68]	Milling	TCM	Cutting force, Vibration, AE	Time, Frequency, Time-Frequency
[69]	Milling	Chatter identification	Cutting force, Acceleration, Image ripple distance	Time, Frequency
[70]	Turning	Tool wear estimation	Cutting force, AE, Vibration acceleration	Wavelet Packet Transform (WPT)
[71]	Milling	Tool failure detection	Current, Vibration, AE	Time, Frequency
[72]	Grinding	Roughness prediction	Grinding force, Vibration, AE	Time, Frequency
[73]	Polishing	Roughness assessment	AE, Strain, Current	Time, WPT
[74]	Drilling	Tool health monitoring	Thrust force, Torque	Time, Frequency, Fractal

tool condition monitoring (TCM). Cerna and Harvey [78] demonstrated that processing a signal dataset—associated with low sampling frequency—in the frequency domain results in a misleading representation due to aliasing. Du et al. [79] demonstrated that time latency (or time delay) and sampling rate underlying the sensor signals must be synchronously optimized for improving the control performance in a feedback control system. Lalouani et al. [80] described that acquiring data at higher sampling rates and its (data) transmission cause significant energy dissipation from a sensor system. They also articulated that suppressing energy dissipation is a prime requirement for the sustainable operation of a sensor system. They developed an energy optimization method, which deploys in-network processing to reduce the number of data transmissions. Halgamuge et al. [81] presented an overview of the sources of sensor power consumption. Some sources are: sensing signals at a sampling rate, signal conditioning, Analog to Digital Conversion (ADC), reading/writing the sensed data in memory, and data transmitting. They also developed an energy model for estimating the life expectancy of WSN. Wang and Chandrakasan [82] articulated that both the communication and computing energy need to be suppressed for prolonging the lifetimes of the wireless sensors in a multi-sensory network. For this, they stressed the need for an efficient signal processing method to extract meaningful information from the sensed data. McIntire et al. [83] described the requirements of Embedded Networked Sensor (ENS) systems from the viewpoint of critical environment monitoring. They articulated that the ENS must consume low energy. At the same time, the ENS must satisfy complex information processing to select proper sensor sampling. Marinkovic and Popovici [84] developed a method for suppressing the communication energy dissipation in a Wireless Body Area Network (WBAN) sensor node. The method adapts wireless wake-up functionality enabled by a Wake-Up Receiver (WUR). Brunelli et al. [85] articulated that computational tasks in a monitoring environment should consume less energy to guarantee an energy-efficient sensor network and data center.

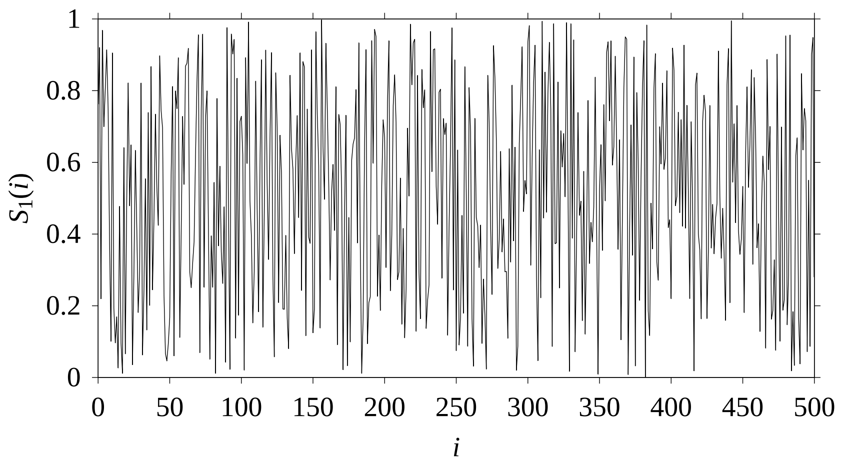
However, in reality, sensor signals underlying manufacturing phenomena are mostly non-linear and stochastic. Also, a low data acquisition rate is a possible outcome due to time latency (or delay) in complex communication networks underlying smart manufacturing systems [56, 57, 58]. As described above, the conventional signal processing methods (time, frequency, and time-frequency-based methods) do not perform well under the abovementioned restraints. In particular, the methods are inadequate for understanding the nature underlying non-linear and stochastic signals. They (methods) depend on a higher data acquisition rate and complex computational arrangements, which create difficulties in storing the sensed data, consume more time to process the data, and cause energy dissipation from the sensor network. Therefore, apart from the conventional methods, alternative methods must be investigated and incorporated for signal processing and handling the abovementioned difficulties from the smart manufacturing context. One alternative method might be delay domain-based signal processing, as presented in the following section.

4.3 Delay Domain-Based Signal Processing

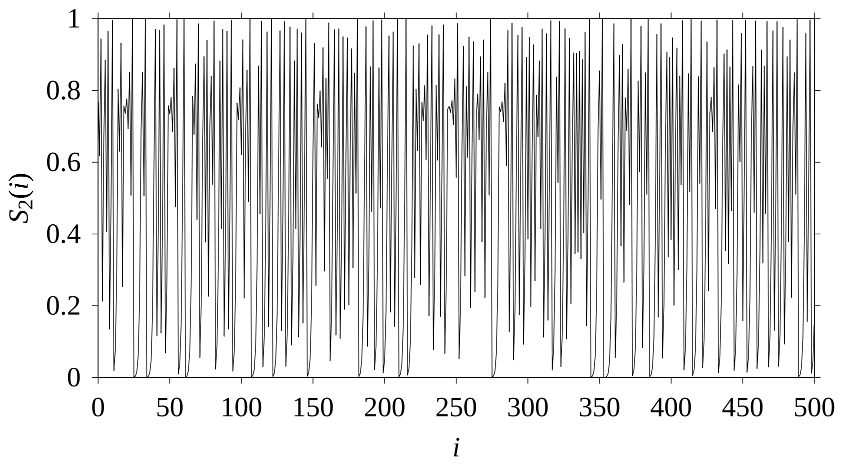
As mentioned in the previous section (Section 4.2), alternative signal processing methods are needed for unfolding the effect of time latency. For this, one straightforward way might be to adapt delay domain-based signal processing [44, 45]. Few authors have embarked on this issue—delay domain-based signal processing and its implication in smart manufacturing. For example, Ullah [86] used delay domain-based processing for unfolding the dynamics underlying surface roughness signal datasets. Ullah and Harib [87] proposed a rule-based knowledge extraction process for simulating surface roughness where the rules are derived from the delay domain-based processing of historical roughness datasets. Wang and Li [88] used delay domain-based processing for the correlation analysis while developing a chaotic image encryption algorithm.

By definition, delay domain-based signal processing means transferring a dataset (time series) to a space called delay map [44]. The map considers a series of forward or backward values from the dataset based on a forward or backward delay parameter (a non-zero integer), respectively. For example, let $\{A(t) \in \Re \,|\, t = 0, \Delta t, 2\Delta t, ...\}$ be a signal dataset, where Δt is the sampling interval. Thus, $\big(t, A(t)\big)$ is the corresponding time series. Now, let d be a non-zero integer to define the value of delay or time latency, i.e., $d \in \mathbb{Z}^+$. As such, the dataset of ordered pairs $\big\{\big(A(t), A(t \pm D)\big) \,|\, D = d \times \Delta t, t = 0, \Delta t, 2\Delta t, ...\big\}$ becomes the corresponding delay map. The time series can be represented by indexing its elements using a pointer, if preferred. In this case, $A(t)$ is replaced by $A(i)\big(= A(t)\big)$, where $t = i \times \Delta t$ and i is the pointer, $i = 0, 1, ...$ As such, $\big(A(i), A(i \pm d)\big)$ becomes the corresponding delay map.

For better understanding, consider an example to illustrate its (delay domain) significance as follows. Let S_1 and S_2 be the time series of two arbitrary signals, where the signal values are $S_1(i), S_2(i) \in [0, 1], i = 0, 1, ...,$ as shown in Figure 4.1. Thus, the plots shown in Figure 4.1 are the time



(a)



(b)

Figure 4.1: Time series. (a) S_1, (b) S_2.

domain representations of the signals. Eight quantifiers (features) $(F_0, ..., F_7)$ = (Mean, Root Mean Square (RMS), Standard Deviation (SD), Peak, Crest Factor (CF), Shape Factor (SF), Impulse Factor (IF), Peak to Peak (PTP)) are used to quantify S_1 and S_2 in time domain. The values of the features are plotted in Figure 4.2 for both signals.

As seen in Figure 4.2, the values of all features of S_1 and S_2 are almost the same. Therefore, as far as the time domain is concerned, S_1 and S_2 are two very similar signals. Let us quantify the signals in frequency domain. For this, the Fast Fourier Transform (FFT) is performed on the datasets shown in Figure 4.1 and the amplitude-frequency diagrams of S_1 and S_2 are

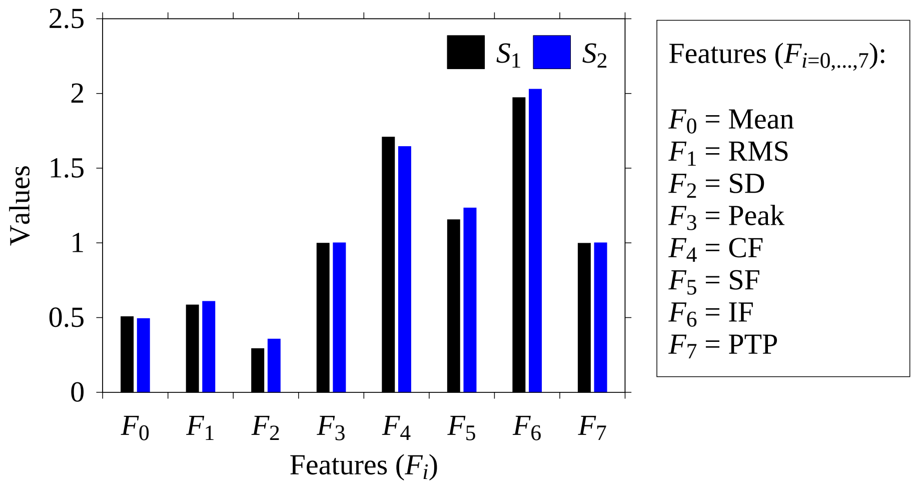


Figure 4.2: Quantifying S_1 and S_2 in time domain.

constructed, as shown in Figure 4.3. As seen in Figure 4.3, the frequencies underlying S_1 (Figure 4.3a) and S_2 (Figure 4.3b) also exhibit a very similar pattern.

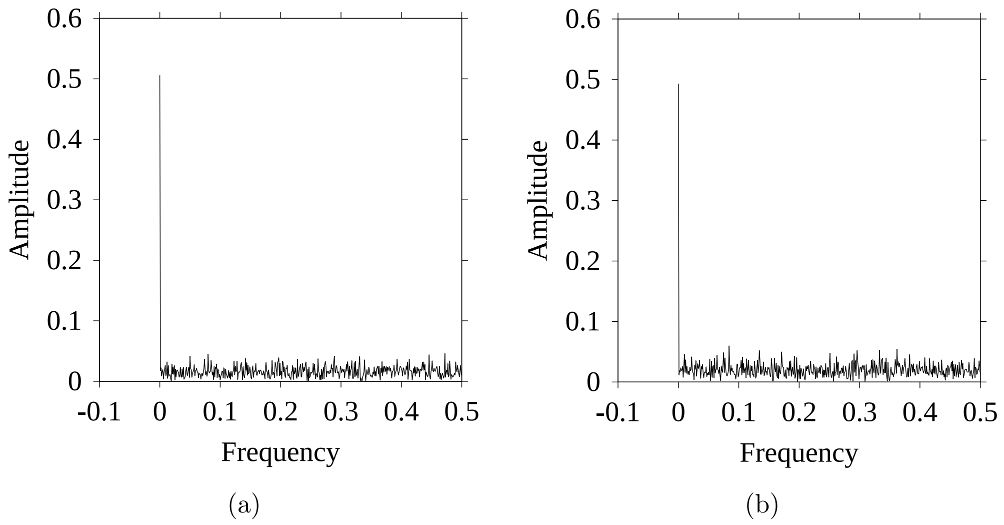


Figure 4.3: Frequency domain representations (FFT). (a) S_1, (b) S_2.

Finally, consider delay domain-based analysis of S_1 and S_2. For this, two delay maps as shown in Figures 4.4a-4.4b for S_1 and S_2, respectively, are constructed. The d (delay parameter) is set to be 1. Consider the delay map of S_1 (Figure 4.4a). In this case, the points are randomly distributed on the delay map revealing the fact that the value of signal at a point of time,

$S_1(i)$, can change to a value taken from the interval [0,1] randomly in the next point of time, $\big(S_1(i+1)\big)$. On the other hand, consider the delay map shown in Figure 4.4b. A systematic pattern (the points are organized on parabolic curve) is shown in the delay map. This means that value of signal at a point of time, $S_2(i)$, cannot change to a value taken from the interval [0,1] randomly in the next point of time, $\big(S_2(i+1)\big)$; it follows an order.

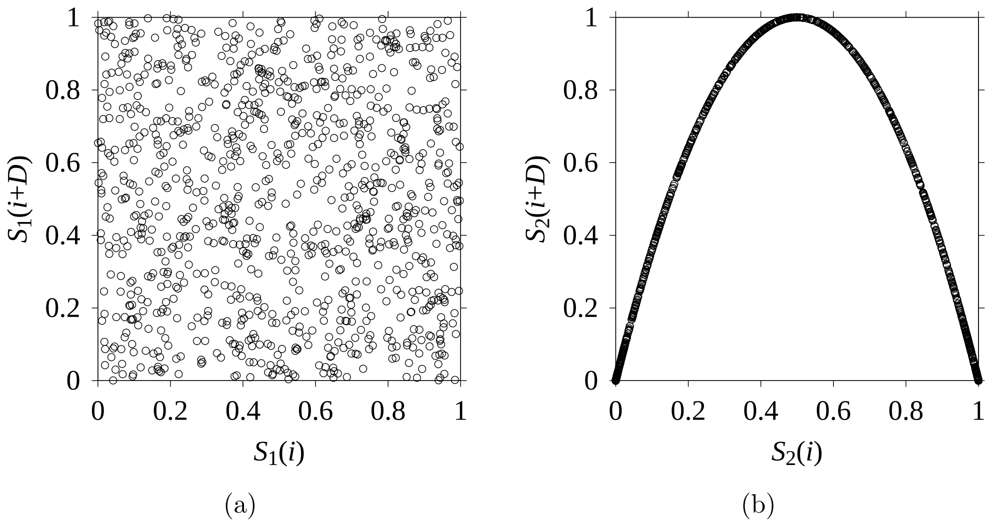


Figure 4.4: Delay maps (for $d = 1$). (a) S_1, (b) S_2.

From the above time domain, frequency domain, and delay domain analyses, it is clear that delay domain-based analysis is more informative compared to time domain and frequency domain-based analyses for understanding the dynamics underlying S_1 and S_2. Thus, the delay domain is a powerful means to process signals. However, let us describe the implication of time latency on signal processing using an arbitrary signal and some real-life machining signals, as follows.

4.4 Implications of Delay

4.4.1 Arbitrary Signal

First, consider an arbitrary signal denoted as $S(i) \mid i = 0, 1, ...,$ that follows Equation 4.1. In Equation 4.1, $a_1, ..., a_4$ and $f_1, ..., f_4$ are the amplitude and frequency components, respectively, where $a_1 = 15, a_2 = 10, a_3 = 5, a_4 =$

$10, f_1 = 0.08, f_2 = 0.03, f_3 = 0.06$, and $f_4 = 0.045$. As seen in Figure 4.5, $S(i)$ (denoted as A) is the source signal. When $S(i)$ is transmitted from the source and received at a receiver-end, how delay impacts it ($S(i)$) is the research question here. Say, a constant delay is occurred in this process, denoted as c, where $c = 5$. As a result, the receiver-end receives a constant delay-driven signal denoted as $T(j)$, where $T(j) = S(i + c) \,|\, j = 0, 1, ...$, as shown in Figure 4.5 (denoted as B). On the other hand, say, a random delay is occurred, denoted as r_l, where $r_{l=1,...,5} \in \{1, 2, 3, 4, 5\}$. As a result, the receiver-end receives a random delay-driven signal denoted as $U(k)$, where $U(k) = S(i + r_l) \,|\, k = 0, 1, ...$, as shown in Figure 4.5 (denoted as C).

$$S(i) = a_1 \sin(2\pi f_1) + a_2 \cos(3 \times 2\pi f_2) + a_3 \sin(2\pi f_3) + a_4 \cos(3 \times 2\pi f_4) \quad (4.1)$$

As seen in Figure 4.5, when a delay occurs whether it is constant or random, the characteristic of the source signal (here $S(i)$) gets affected. However, from visual inspection, the time series of the signals ($S(i)$, $T(j)$ and $U(k)$) are not so insightful for understanding the underlying dynamics. For this reason, the abovementioned signals ($S(i)$, $T(j)$ and $U(k)$) are processed in the frequency domain (using Fast Fourier Transformation (FFT)). The corresponding results are shown in Figures 4.6a-4.6c, respectively.

As seen in Figure 4.6a, the frequency domain represents the frequencies underlying $S(i)$ as expected. As seen in Figure 4.6b, the frequencies underlying $T(j)$ are not the same compared to that of $S(i)$ (see Figure 4.6a). This means that the constant delay can change the frequency information associated with the source signal, $S(i)$. As seen in Figure 4.6c, the frequencies underlying $U(k)$ are jumbled. This means that the random delay can change the frequency information associated with the source signal, $S(i)$ (see Figure 4.6a), significantly.

In addition, the abovementioned signals ($S(i)$, $T(j)$ and $U(k)$) are transferred to the delay domain, which is often recommended for chaotic signals. The corresponding results are shown in Figure 4.7a-4.7c, respectively. Figure 4.7a shows the delay map of $S(i)$ consisting of points $\big(S(i), S(i+1)\big)$. It is seen that the returns from one point to another follow a very systematic pattern. This means $S(i)$ is not chaotic. As seen in Figure 4.7b, the delay map of $T(j)$ consisting of points $\big(T(j), T(j + 1)\big)$, is somewhat distorted compared to that of $S(i)$ (see Figure 4.7a). This means that the constant delay has affected the systematic behavior of the $S(i)$. As seen in Figure 4.7c, the delay map of $U(k)$ consisting of points $\big(U(k), U(k + 1)\big)$, is dis-

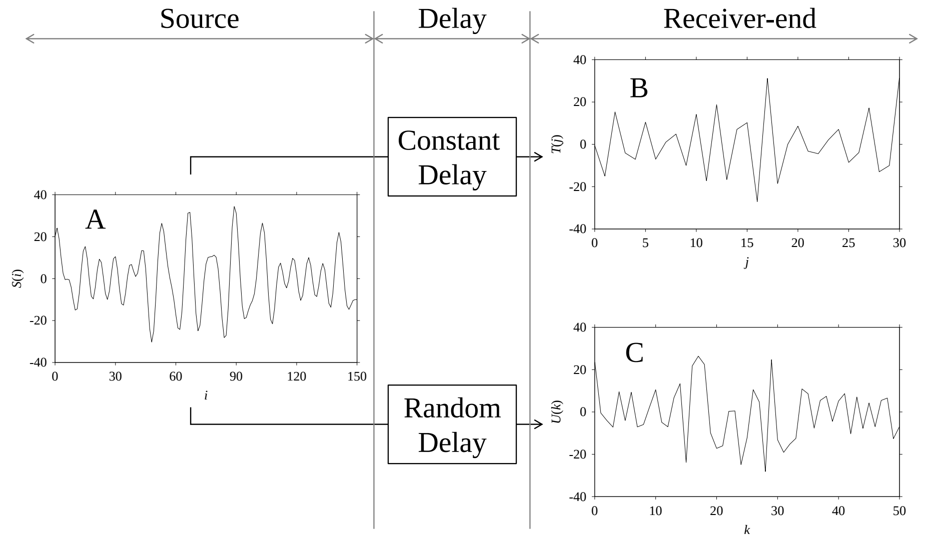


Figure 4.5: Implications of delay on $S(i)$.

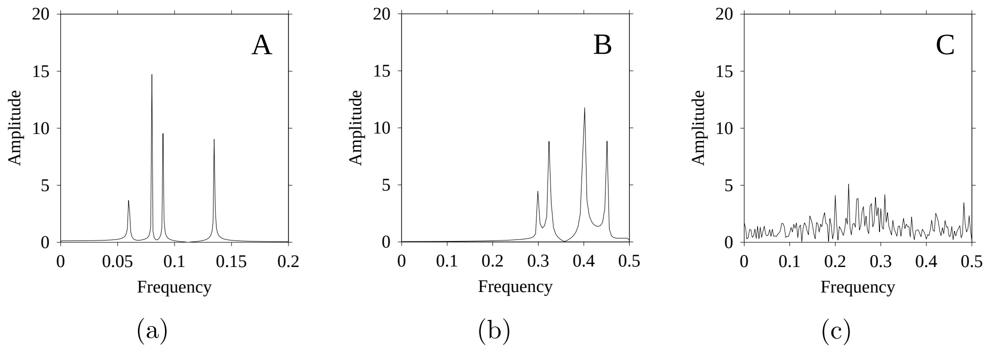


Figure 4.6: Frequency domain representations (FFT). (a) $S(i)$, (b) $T(j)$, (c) $U(k)$.

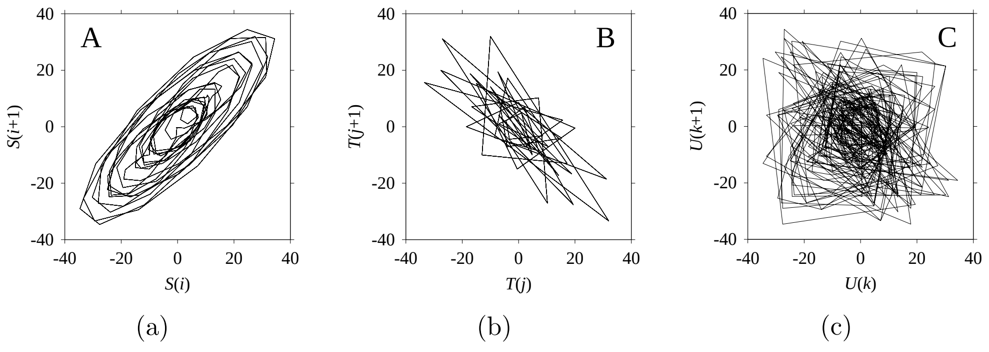


Figure 4.7: Delay maps. (a) $S(i)$, (b) $T(j)$, (c) $U(k)$.

torted compared to that of $S(i)$ (see Figure 4.7a) and $T(j)$ (see Figure 4.7b). The returns from one point to another does not follow a systematic pattern anymore. This means that the random delay makes the signal chaotic.

4.4.2 Real-Life Machining Signals

The previous sub-section (Section 4.4.1) shows the significance of signal processing in the delay domain using an arbitrary signal. Now, let us consider some real-life machining signals to see whether or not the delay domain remains significant in real-life settings. In particular, cutting force signals obtained by performing machining experiments according to the settings shown in Table 4.2 are considered.

Table 4.2: Conditions for the machining experiment.

Item	**Description**
Machine tool	Vertical machining center Make: Mori Seiki Model: NV5000
Cutting tool	Carbide Φ6 solid end mill Make: Mitsubishi Hitachi Model: EPP4060-P-CS
Sensor	Rotary dynamometer Make: Kistler Type: 9170A
Workpiece material	Stainless Steel (JIS: SUS304) Mild Steel (JIS: S15CK) Ductile cast iron (JIS: FCD)
Cutting velocity (V_c)	220 m/min
Spindle speed (N)	11677 rpm
Feed per tooth (f)	0.1 mm/tooth
Feed rate (V_f)	4671 mm/min
Depth of cut (a_p)	1.0 mm
Width of cut (a_e)	0.5 mm

Consider a machining experiment, where a set of workpiece specimens denoted as $W, \forall W \in \{W_1, W_2, W_3\}$ undergo end milling. Here, W_1, W_2, and W_3 refer to the workpiece specimens made of Stainless Steel (JIS: SUS304), Mild Steel (JIS: S15CK), and Ductile Cast Iron (JIS: FCD), respectively. Table 4.2 summarizes corresponding machining conditions (e.g., spindle speed N (rpm), feed per tooth f (mm/tooth), depth of cut a_p (mm), width of cut a_e (mm), and alike) in detail. Figure 4.8 schematically illustrates the outline of the experiment and relative conditions (see the segment denoted as A).

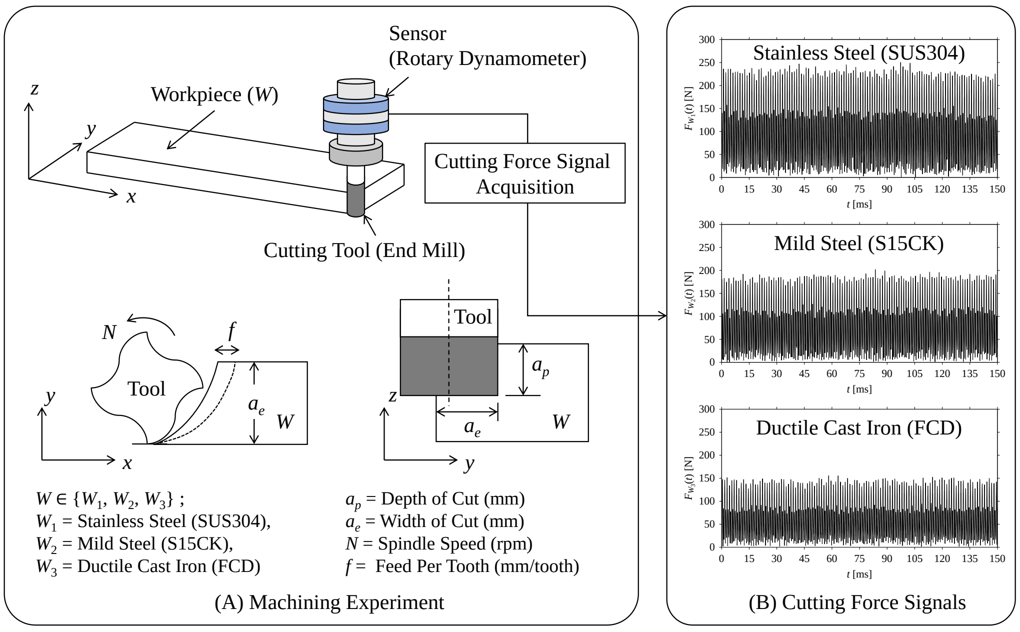

Sensor
(Rotary Dynamometer)
Workpiece (W)
Cutting Force Signal Acquisition
Cutting Tool (End Mill)
z
y
x
N
f
Tool
a_e
W
a_p
$W \in \{W_1, W_2, W_3\}$;
W_1 = Stainless Steel (SUS304),
W_2 = Mild Steel (S15CK),
W_3 = Ductile Cast Iron (FCD)
a_p = Depth of Cut (mm)
a_e = Width of Cut (mm)
N = Spindle Speed (rpm)
f = Feed Per Tooth (mm/tooth)
(A) Machining Experiment
Stainless Steel (SUS304)
Mild Steel (S15CK)
Ductile Cast Iron (FCD)
$F_{W_i}(t)$ [N]
t [ms]
(B) Cutting Force Signals

Figure 4.8: Outline of the machining experiment and signal acquisition.

As seen in Figure 4.8, the corresponding cutting force signals are obtained using a sensor called rotary dynamometer (see Table 4.2 for details) while machining W. Let, $F_W(t), t = 0, \Delta t, 2\Delta t, ..., m \times \Delta t$, be the force signals while machining W, as shown in the segment 'B' in Figure 4.8. Here, Δt denotes a sampling period of 0.02 ms. The goal is to analyze the F_W under time latency or delay. For this, two cases are considered, (1) a short sampling window and (2) a low sampling rate due to varying delay. The F_W is analyzed considering both the abovementioned cases, as described below.

Analyzing Sensor Signals Under Different Sampling Windows

In Industry 4.0-centric systems, the sensor signal sampling windows may vary due to physical limitations and suppressing energy dissipation . Therefore, signals sampled using different sampling windows must be considered. Each piece of sampled signals can then be analyzed in the time, frequency, and delay domains, respectively.

Consider the cutting force signal obtained while machining Stainless Steel (JIS: SUS304), i.e., F_{W_1}. Figure 4.9 shows its time series. As seen in Figure 4.9, the sample size of F_{W_1} is denoted as L_{W_1}, where $L_{W_1} = 7501$. F_{W_1} is sampled four times using four different sampling windows (light blue colored regions in the time series of F_{W_1} shown in Figure 4.9). This results in four new signals denoted as $F_{W_1S_1}, ..., F_{W_1S_4}$. The corresponding sample sizes are denoted as $L_{W_1S_1}, ..., L_{W_1S_4}$, where $L_{W_1S_1} = 5001, L_{W_1S_2} = 2501, L_{W_1S_3} = 1501$, and $L_{W_1S_4} = 501$.

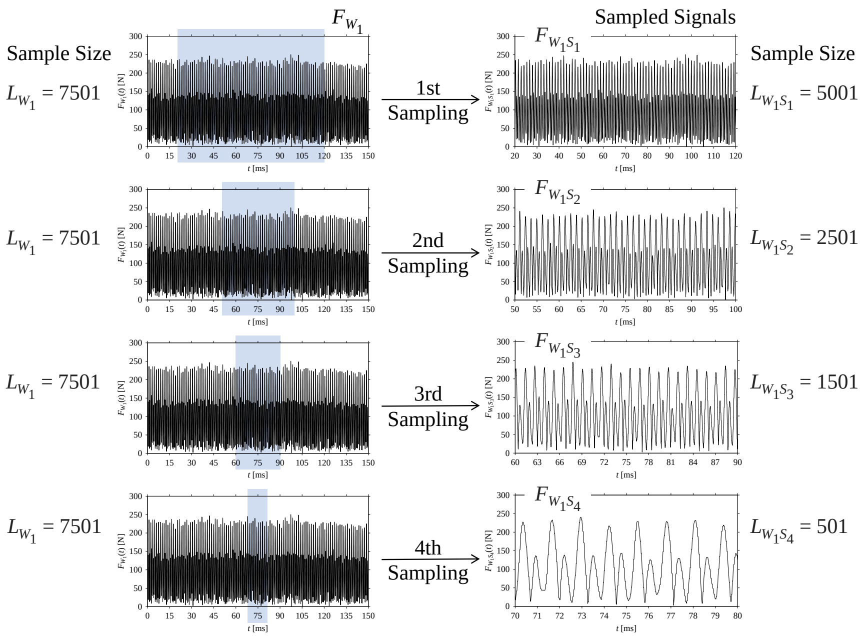
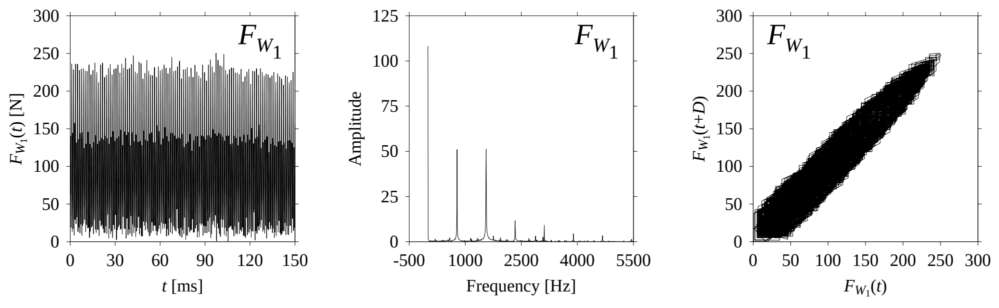


Figure 4.9: Sampling the cutting force signal for machining Stainless Steel.

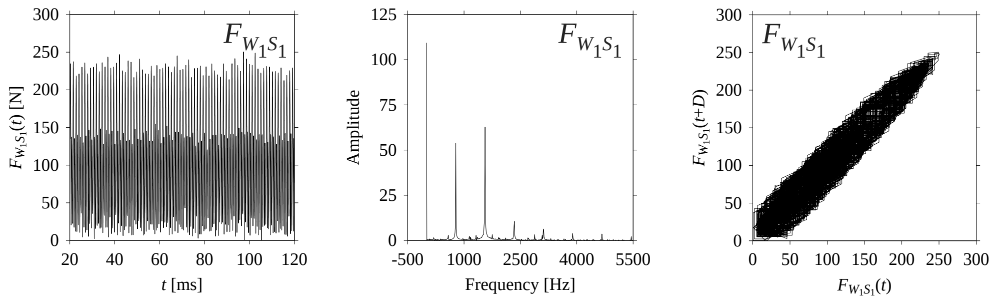
For the sake of analysis, the signals F_{W_1} and $F_{W_1S_1}, ..., F_{W_1S_4}$ are transferred to the frequency domain (using FFT) and delay domain (using $d = 1$, as described in Section 4.3). As such, Figure 4.10 shows the time series, FFT, and delay map for F_{W_1}. Figures 4.11-4.14 show the time series, FFT, and delay map for $F_{W_1S_1}, ..., F_{W_1S_4}$, respectively.

As seen in Figure 4.10b, the prominent frequencies underlying F_{W_1} are 0 Hz, 780 Hz, 1560 Hz, 2333.333 Hz, 3113.333 Hz, 3893.333 Hz, and 4673.333 Hz. As seen in Figure 4.11b, the prominent frequencies underlying $F_{W_1S_1}$ are 0 Hz, 780 Hz, 1560 Hz, 2340 Hz, 3110 Hz, 3890 Hz, and 4670 Hz. As seen in Figure 4.12b, the prominent frequencies underlying $F_{W_1S_2}$ are 0 Hz, 780 Hz, 1560 Hz, 2340 Hz, and 3120 Hz. As seen in Figure 4.13b, the prominent frequencies underlying $F_{W_1S_3}$ are 0 Hz, 766.6667 Hz, 1566.667 Hz, 2333.333 Hz, and 3133.333 Hz. As seen in Figure 4.14b, the prominent frequencies



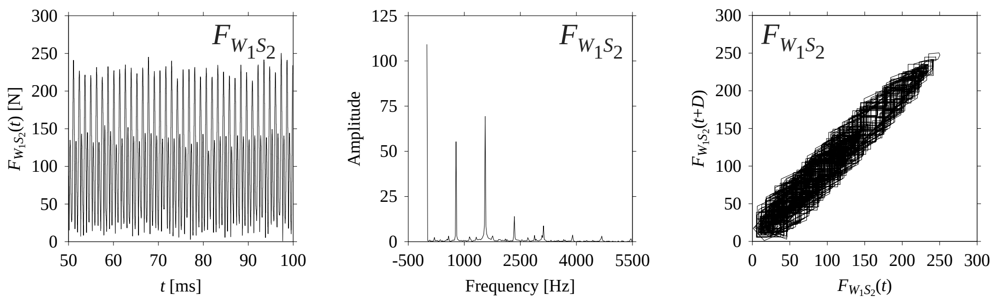
(a) (b) (c)

Figure 4.10: F_{W_1}. (a) Time series, (b) Fast Fourier Transformation (FFT), (c) Delay map ($d = 1$).



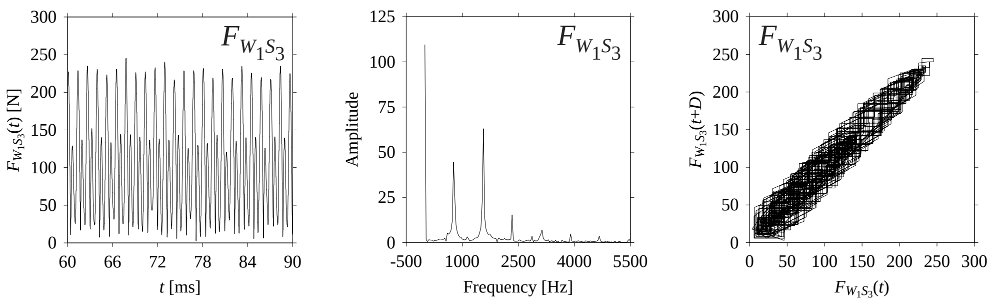
(a) (b) (c)

Figure 4.11: $F_{W_1S_1}$. (a) Time series, (b) Fast Fourier Transformation (FFT), (c) Delay map ($d = 1$).



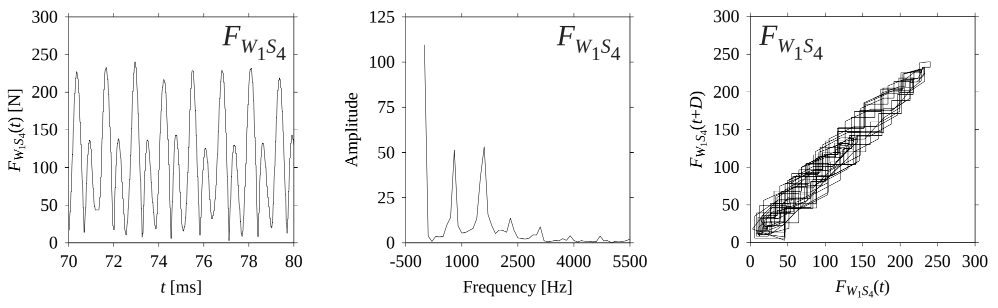
(a) (b) (c)

Figure 4.12: $F_{W_1S_2}$. (a) Time series, (b) Fast Fourier Transformation (FFT), (c) Delay map ($d = 1$).



(a) (b) (c)

Figure 4.13: $F_{W_1S_3}$. (a) Time series, (b) Fast Fourier Transformation (FFT), (c) Delay map ($d = 1$).



(a) (b) (c)

Figure 4.14: $F_{W_1S_4}$. (a) Time series, (b) Fast Fourier Transformation (FFT), (c) Delay map ($d = 1$).

underlying $F_{W_1S_4}$ are 0 Hz, 800 Hz, 1600 Hz, 2300 Hz, and 3100 Hz. This means that the frequency information varies with the sampling window. In addition, the FFT pattern gets affected when the sampling window is shorter (see Figure 4.14b compared to Figure 4.10b).

On the other hand, the delay maps shown in Figure 4.10c and Figures 4.11c-4.14c exhibit similar characteristics under different sampling windows. In particular, the returns of points from one to another are identical. This means that the underlying nature of the F_{W_1} and $F_{W_1S_1}, ..., F_{W_1S_4}$ are the same, regardless of the sample size. In addition, the density of points underlying the delay maps provides meaningful insight into the sample size of the signal. For example, the density of the delay map shown in Figure 4.14c is lighter compared to that of in Figure 4.10c, which mean the corresponding signal $F_{W_1S_4}$ (see Figure 4.14a) undergoes a low sampling window compared to the F_{W_1} (see Figure 4.10a).

Nevertheless, similar outcomes are observed for the other two workpiece specimens, i.e., W_2 = Mild Steel (JIS: S15CK) and W_3 = Ductile Cast Iron (JIS: FCD), as described in Appendix A.1 and Appendix B.1, respectively.

Analyzing Sensor Signals Under Different Delays

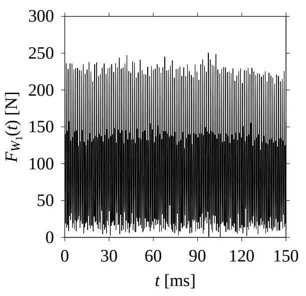
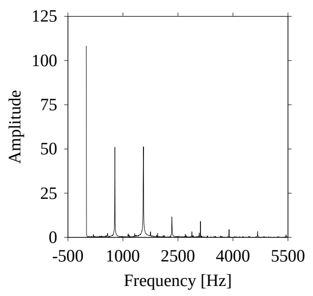
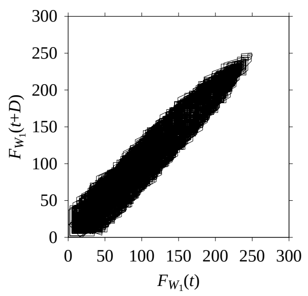
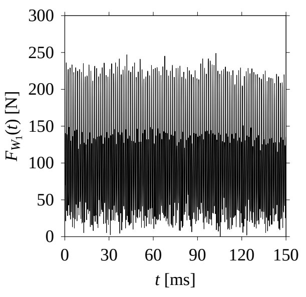
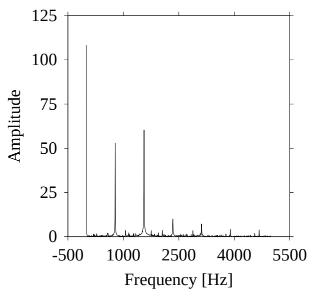
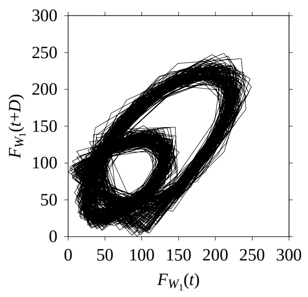
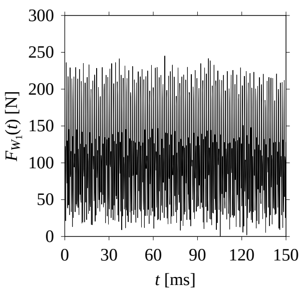
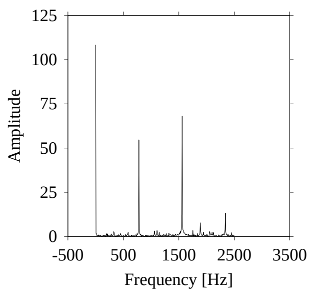
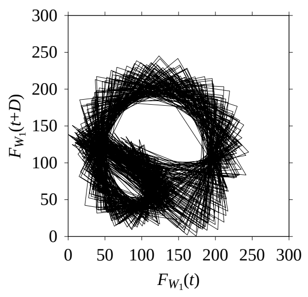
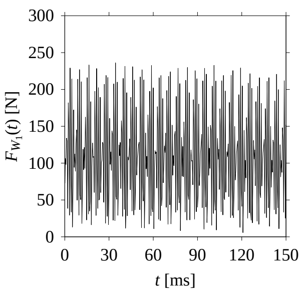
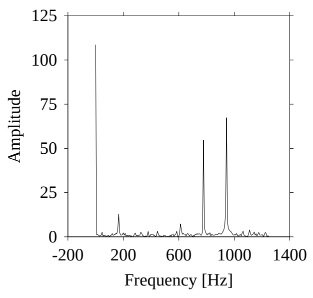
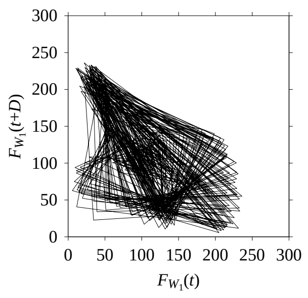
Again, consider the cutting force signal obtained while machining Stainless Steel (JIS: SUS304), i.e., $F_{W_1}(t), t = 0, \Delta t, 2\Delta t, ..., m \times \Delta t$ (can also be seen in Figure 4.8). As mentioned before, here Δt is the sampling period of 0.02 ms. To incorporate time latency or delay, Δt is increased using the delay parameter d (non-zero integer), such as $d \times \Delta t$. For example, for $d = 1$, the sampling period remains $1 \times 0.02 = 0.02$ ms; for $d = 2$, the sampling period becomes $2 \times 0.02 = 0.04$ ms; and alike. As mentioned in Section 4.3, $d \times \Delta t$ is simplified using D, where $D = d \times \Delta t$. As such, a set of time series is generated using D where $d = 1, 2, ...$.

The goal here is to understand the dynamics underlying the cutting force signals due to varying delay. For this, the time series datasets are transferred to the frequency domains (using FFT) and corresponding delay domains. Table 4.3 shows the outcomes for some of the delays, i.e., $d = 1, 5, 10, 20, 30, 40, 50$, and 60.

As seen in Table 4.3, when d increases, the frequency information underlying the F_{W_1} gets affected. The prominent frequencies (see the FFT diagram for $d = 1$) are gradually lost due to aliasing [78] when $d > 5$ (see the FFT diagrams for $d = 10, 20, 30, 40, 50$, and 60).

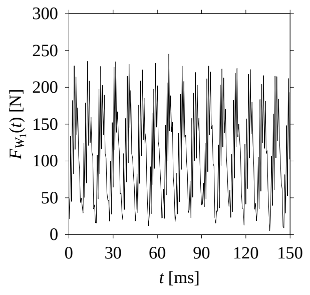
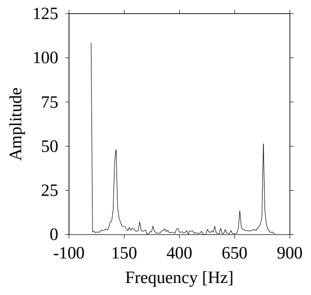
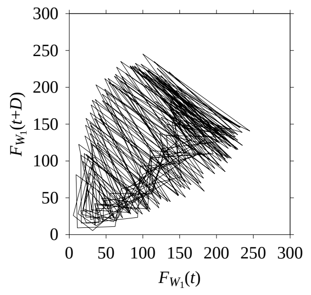
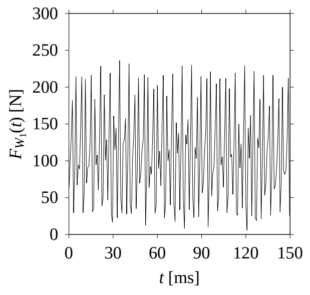
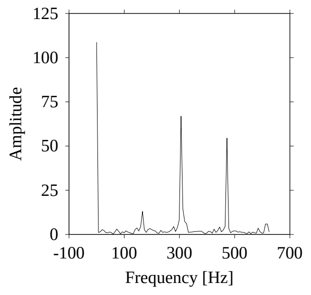
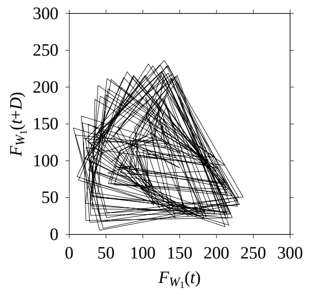
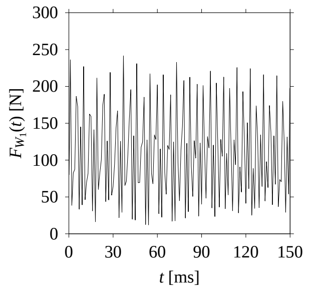
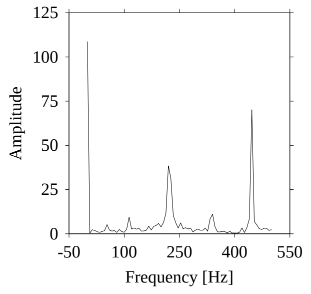
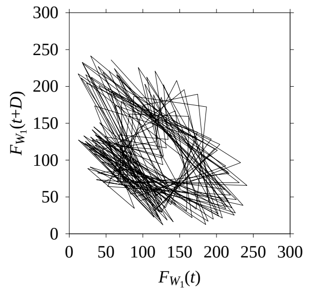
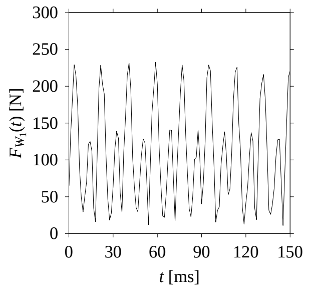
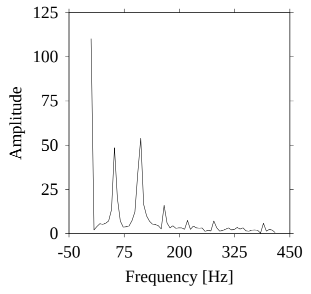
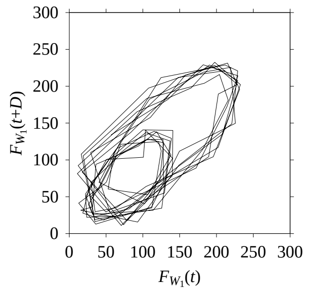
On the other hand, consider the delay maps consisting of the points $\big(F_{W_1}(t), F_{W_1}(t + D)\big)$ shown in Table 4.3. When $d = 1$, the delay map exhibits a very systematic pattern. When d increases, the delay maps get more and more scattered (see the delay maps for $d = 5, 10, 20, 30, 40$, and 50). This means that the signal gets more and more chaotic due to the presence of delay. However, when $d = 60$, the corresponding delay map is somewhat systematic and similar to that of $d = 5$. This means that the underlying natures of these two signals are similar, regardless of the difference in the sampling rate. It is worth mentioning that the corresponding FFTs (see the FFTs for $d = 5$ and for $d = 60$ shown in Table 4.3) are different and do not preserve the nature of the source signal. As such, delay domain-based representation is more informative for understanding the underlying nature of F_{W_1} under a low data acquisition rate due to time latency or delay.

Table 4.3: Cutting force signal (F_{W_1}) in the form of time series, FFT, and delay map under varying delay (d).

Time series(s)	**FFT(s)**	**Delay maps**
	$d = 1$	
		
	$d = 5$	
		
	$d = 10$	
		
	$d = 20$	
		

(continued on next page)

Table 4.3: Cutting force signal (F_{W_1}) in the form of time series, FFT, and delay map under varying delay (d) (continued from previous page).

Time series(s)	FFT(s)	Delay maps
	$d = 30$	
		
	$d = 40$	
		
	$d = 50$	
		
	$d = 60$	
		

Nevertheless, similar outcomes are observed for the other two workpiece specimens, i.e., W_2 = Mild Steel (JIS: S15CK) and W_3 = Ductile Cast Iron (JIS: FCD), as described in Appendix A.2 and Appendix B.2, respectively.

To summarize, as seen in Table 4.4, real-life cutting force signals collected while machining different materials (Stainless Steel, Mild Steel, and Ductile Cast Iron) are analyzed in the frequency and delay domains. This time, both the signal window and the amount of delay are varied to see their effect on signal processing.

Table 4.4: Summary of observations from real-life signal processing.

Analyzing the signals under different sampling windows		
Sample size	**Observations (O)**	
	Frequency domain	**Delay domain**
Larger	O1. Meaningful frequency information is prominent.	O1. Point clouds' density is higher.
	O2. Effective for unfolding signals' original nature.	O2. The density signifies the sample size.
		O3. Effective for unfolding signals' original nature.
Smaller	O1. Frequency information varies with the sample size.	O1. Point clouds' density becomes lower.
	O2. Frequency information gets lost or distorted.	O2. The density signifies the sample size.
	O3. The FFT exhibits a different pattern than the FFT for larger sample.	O3. Regardless of the sample size, the delay maps retain the dynamics.
	O4. Inadequate for unfolding signals' original nature.	O4. Effective for unfolding signals' original nature.
Analyzing the signals under different delays		
Delay, $d = \mathbf{1, ..., 100}$	**Observations (O)**	
	Frequency domain	**Delay domain**
$d < 5$	O1. Meaningful frequency information is prominent.	O1. Delay map exhibits very systematic patterns.
	O2. Effective for unfolding signals' original nature.	O2. Effective for unfolding signals' original nature.

(continued on next page)

Table 4.4: Summary of observations from real-life signal processing (continued from previous page).

Analyzing the signals under different delays		
Delay,	**Observations (O)**	
$d = \mathbf{1, ..., 100}$	**Frequency domain**	**Delay domain**
$d = 5$	O1. Meaningful frequency information is somewhat prominent.	O1. Delay map exhibits somewhat systematic patterns.
	O2. Effective for unfolding signals' original nature.	O2. Effective for unfolding signals' original nature.
		O3. Delay map starts getting scattered.
$d > 5$	O1. Frequency information is gradually lost due to aliasing.	O1. Delay map gets more and more scattered.
	O2. Inadequate for unfolding signals' original nature.	O2. The signal starts to get more and more chaotic.
		O3. The chaotic delay map signifies the presence of delay.
$d = 60$	Same as above.	O1. Delay maps' pattern is found similar to the pattern for $d = 5$.
		O2. This signifies that delay domain guarantees signals' original nature for some critical values of delay (e.g., here 5 and 60).
		O3. This also signifies that delay domain is more robust in unfolding the nature of a signal subjected to a high delay.
$60 < d \leq 100$	Same as above.	Delay map gets scattered again.

In the case of the signal window, it is found that both frequency and delay domains are effective for understanding the signals' original nature for a larger window. On the other hand, the delay domain is more effective for smaller windows than the frequency domain. This is because frequency information gets lost or distorted when the sample size is smaller, whereas the delay domain retains the dynamics associated with the signals regardless of the sample size. On the other hand, in the case of varying delay, it is found that when delay increases, the frequency spectrum gets affected. The prominent frequencies of the original signal are gradually lost due to aliasing when the delay exceeds a critical value.

On the other hand, when the delay increases, the delay domain gets more and more scattered. Furthermore, for some critical values of delay (one may be very high and the other may be very low), the delay domains exhibit similar characteristics, which is not the case for the frequency domains. Thus, when a very short window or low sampling rate (high delay) is used to analyze a signal, delay domains guarantee its (signal's) original nature. This means that the delay domain-based representation is more robust in understating the nature of a sensor signal subjected to high time latency, a common phenomenon in Industry 4.0-relevant manufacturing environments.

However, let us consider another example, where it is shown that the delay domain can distinguish different machining situations more effectively than the frequency domain, as described in the following section.

4.5 Distinguishing Machining Situations

When machining is performed, the whole process undergoes different stages or situations. For example, at the onset and completion of machining, the cutting tool remains idle. Between the onset and completion, the tool machines the workpiece using a set of predefined cutting conditions. During cutting, the cutting conditions can be changed depending on the geometry or material of the workpiece. This creates the following situations: idle, idle to cutting, cutting with different cutting conditions/materials, cutting to idle, and idle. During this situation, an abnormality may happen (e.g., breakage of a tool, chatter vibration, and so on), adding other situations in the whole process.

The Industry 4.0-centric systems must understand these situations on a real-time basis from sensor signals. Furthermore, the systems must coordinate among all machines under consideration and decide the right courses of action. Otherwise, the safety, economy, and quality of the other planning activities (e.g., maintenance scheduling) cannot be ensured [89]. Since delay maps effectively understand a signal's hidden characteristics even though the signals are subjected to time latency and the signal sampling window is a short one (as shown in Section 4.4.2), these maps can be employed to distinguish different machining situations. This section explores this possibility showing the efficacy of the delay domain.

For the sake of better understanding, Section 4.5.1 describes the machining experiment and sensor signal acquisition. Section 4.5.2 describes the pre-processing of the acquired signal. Finally, Section 4.5.3 describes the frequency and delay domain-based signal processing, and the obtained results.

4.5.1 Sensor Signal Acquisition

As seen in Figure 4.15, a multi-material workpiece made of Stainless Steel (JIS: SUS304) and Mild Steel (JIS: S15CK) is machined following an end milling process. The machining conditions are summarized in Table 4.5. The reasons for choosing a multi-material workpiece over a mono-material workpiece are as follows: (1) The usage of multi-material objects is increasing because these perform better in terms of material efficiency compared to their mono-material counterparts [90, 91], and (2) Machining a multi-material object underlies more complex machining situations compared to machining a mono-material object.

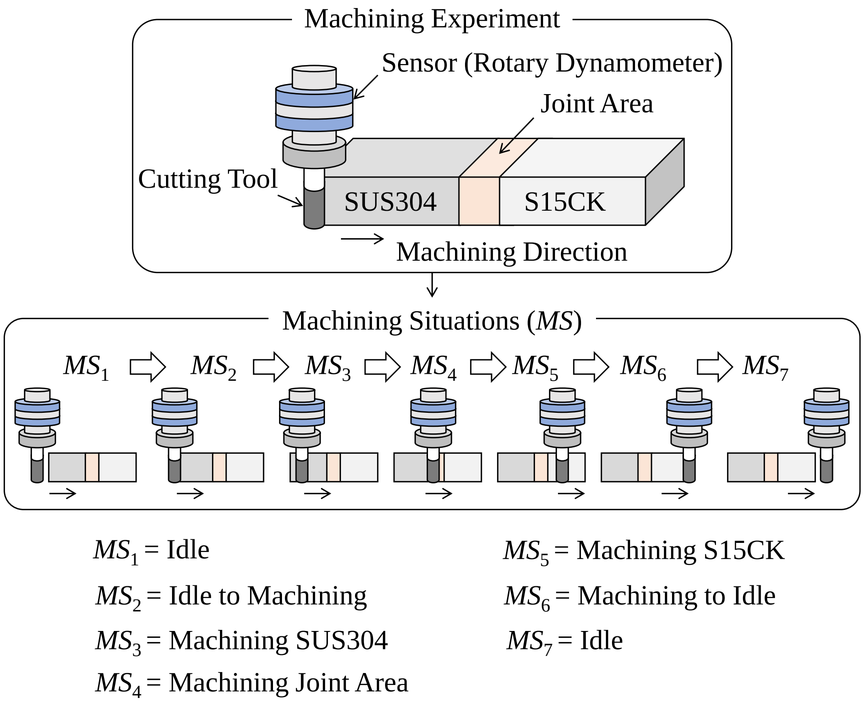


Figure 4.15: Machining situations underlying the machining experiment.

However, the multi-material workpiece is machined from the hard-to-soft material direction (i.e., SUS304 to S15CK), as shown in Figure 4.15 (also reported in Table 4.5). As such, the machining involves a set of machining situations, denoted as $MS, \forall MS \in \{MS_1, ..., MS_7\}$. Here, MS_1 denotes the situation when the cutting tool is idle just before the machining. MS_2 denotes the transition from an idle state to a machining state. MS_3 denotes the machining of SUS304 segment. MS_4 denotes the machining of the joint area (heat-affected area while joining SUS304 and S15CK). MS_5 denotes the machining of S15CK segment. MS_6 denotes the transition from a machining state to an idle state. Finally, MS_7 denotes the situation when the tool is idle after completing the machining.

Table 4.5: Conditions for machining a multi-material workpiece.

Item	**Description**
Machine tool	Vertical machining center Make: Mori Seiki Model: NV5000
Cutting tool	Carbide Φ6 solid end mill Make: Mitsubishi Hitachi Model: EPP4060-P-CS
Sensor	Rotary dynamometer Make: Kistler Type: 9170A
Workpiece material	Bimetallic material made of Stainless Steel (JIS: SUS304) and Mild Steel (JIS: S15CK)
Cutting velocity (V_c)	220 m/min
Spindle speed (N)	11677 rpm
Feed per tooth (f)	0.2 mm/tooth
Feed rate (V_f)	9341 mm/min
Depth of cut (a_p)	2.0 mm
Width of cut (a_e)	0.5 mm
Machining direction	SUS304 to S15CK

While the cutting tool is passed through the abovementioned situations, the corresponding machining force signals are recorded using a rotary dynamometer (as reported in Table 4.5). Let F_o be the raw force signals, $\forall o \in \{x, y, z\}$. Here, F_x, F_y, and F_z refers to the force signals along the x-axis, y-axis, and z-axis, respectively. The force signal in the x-axis, i.e.,

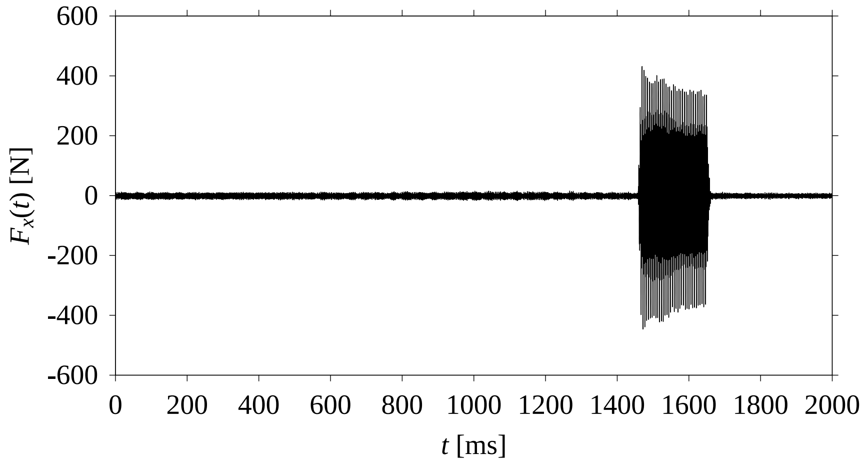


Figure 4.16: Time series of F_x.

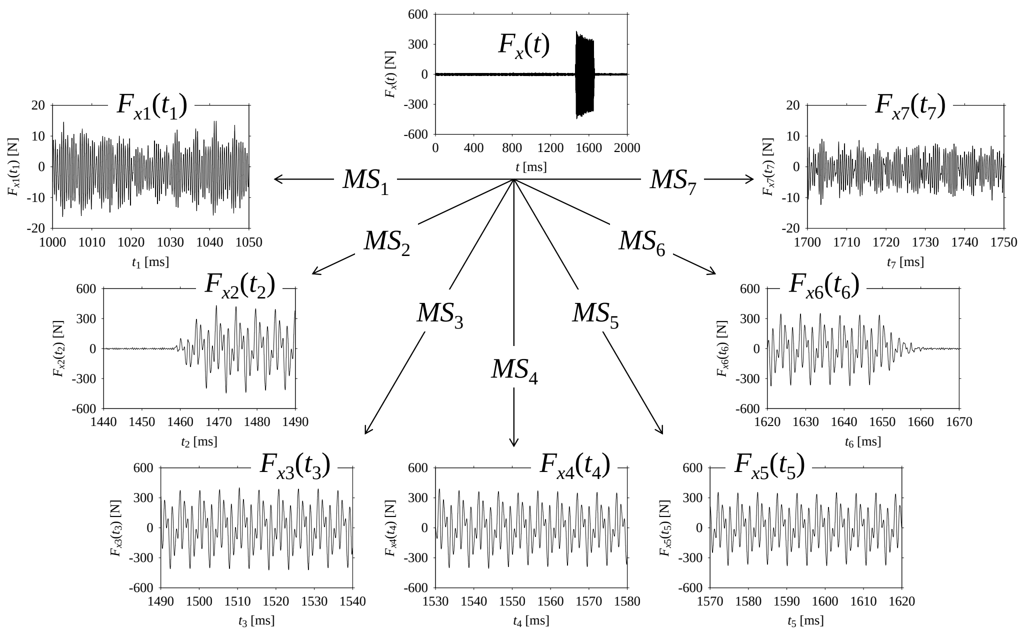


Figure 4.17: Sampling $F_x(t)$ based on machining situations (MS).

F_x, is reported here. Figure 4.16 shows its time series, such as $F_x(t), t = 0, \Delta t, 2\Delta t, ...$ Here, Δt is a sampling period of 0.02 ms.

4.5.2 Signal Pre-Processing

As seen in Figure 4.17, $F_x(t)$ is sampled to seven (7) fragments based on the abovementioned machining situations (MS). The fragments are denoted as $F_{x1}(t_1), ..., F_{x7}(t_7)$ corresponding to $MS_1, ..., MS_7$, respectively. The sampling spans for $F_{x1}(t_1), ..., F_{x7}(t_7)$ are as follows: $t_1 = [t_{\text{start}},\text{t}_{\text{end}}]$ = [1000,1050], $t_2 = [t_{\text{start}},\text{t}_{\text{end}}]$ = [1440,1490], $t_3 = [t_{\text{start}},\text{t}_{\text{end}}]$ = [1490,1540], $t_4 = [t_{\text{start}},\text{t}_{\text{end}}]$ = [1530,1580], $t_5 = [t_{\text{start}},\text{t}_{\text{end}}]$ = [1570,1620], $t_6 = [t_{\text{start}},\text{t}_{\text{end}}]$ = [1620,1670], and $t_7 = [t_{\text{start}},\text{t}_{\text{end}}]$ = [1700,1750]. Note that each fragment consists of 2501 samples.

Nevertheless, the sampled fragments $(F_{x1}(t_1), ..., F_{x7}(t_7))$ are analyzed in the frequency and delay domains for distinguishing the corresponding machining situations $(MS_1, ..., MS_7)$, as described in the following sub-section.

4.5.3 Analysis in Frequency and Delay Domains

First, consider the frequency domain-based analysis. Figures 4.18a-4.18g show the frequency domain representations (FFT) of $F_{x1}(t_1), ..., F_{x7}(t_7)$ corresponds to $MS_1, ..., MS_7$, respectively.

As seen in Figures 4.18a and 4.18g, the frequency components underlying $F_{x1}(t_1)$ and $F_{x7}(t_7)$ are same, respectively. These frequency components are different from the frequency components underlying $F_{x2}(t_2), ..., F_{x6}(t_6)$, as shown in Figures 4.18b-4.18f, respectively. In addition, the frequency components underlying $F_{x2}(t_2), ..., F_{x6}(t_6)$ are the same (see Figures 4.18b-4.18f, respectively). This means that the frequency domain-based analysis is effective only for distinguishing the situations MS_1 (underlying $F_{x1}(t_1)$) and MS_7 (underlying $F_{x7}(t_7)$) from others ($MS_2, ..., MS_6$ underlying $F_{x2}(t_2), ..., F_{x6}(t_6)$, respectively). It (frequency domain-based analysis) is not effective for distinguishing the situations $MS_2, ..., MS_6$ (underlying $F_{x2}(t_2), ..., F_{x6}(t_6)$, respectively) from each other.

Now, consider the delay domain-based analysis. For this, a set of delay maps are constructed for each signals $F_{x1}(t_1), ..., F_{x7}(t_7)$ by varying the delay parameter $d = 1, ..., 100$. Note that the delay maps are generated following

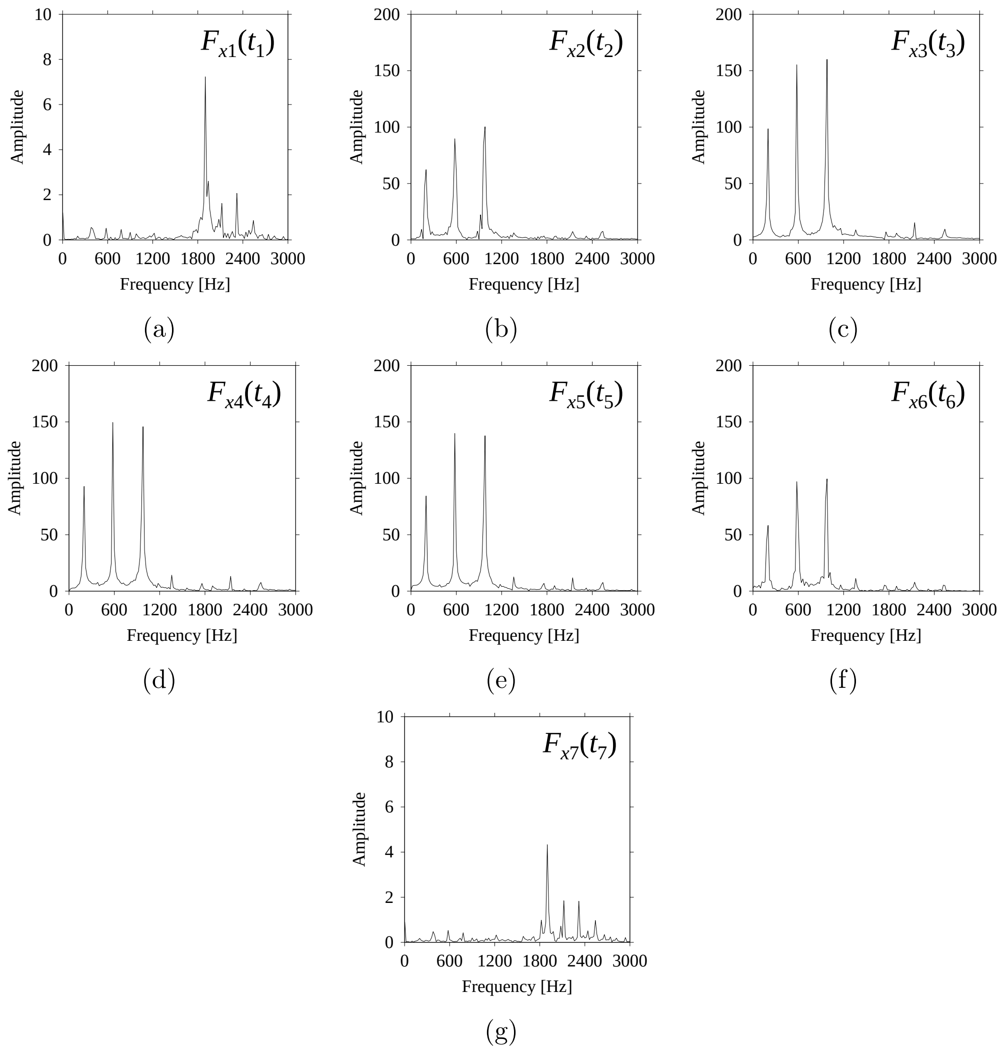


Figure 4.18: Frequency domain representations (FFTs) of the sampled signals. (a) $F_{x1}(t_1)$, (b) $F_{x2}(t_2)$, (c) $F_{x3}(t_3)$, (d) $F_{x4}(t_4)$, (e) $F_{x5}(t_5)$, (f) $F_{x6}(t_6)$, (g) $F_{x7}(t_7)$.

the methodology described in Section 4.3 (can also be seen in Section 4.4.2). Table 4.6 shows the delay maps corresponding to $d = 1, 2, 5, 50$, and 100.

As seen in Table 4.6, the delay maps of $F_{x1}(t_1), ..., F_{x7}(t_7)$ exhibit similar patterns when d is very small ($d = 1$ and 2). The maps become more and more chaotic with the increase in d, as observed in Table 4.6. For some specific delay parameter values, a delay map corresponding to a given machining situation exhibits a distinct pattern, though it is not the case in most cases. Nevertheless, the specific values of delay for which each delay

Table 4.6: Delay maps for $F_{x1}(t_1), ..., F_{x7}(t_7)$ (corresponds to $MS_1, ..., MS_7$) when $d = 1, 2, 5, 50$, and 100.

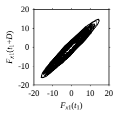
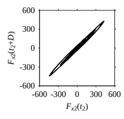
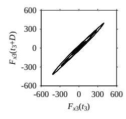
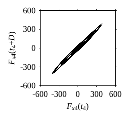
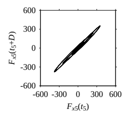
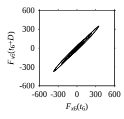
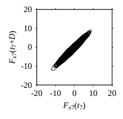
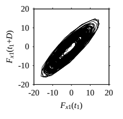
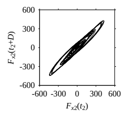
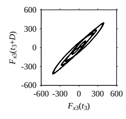
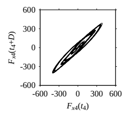
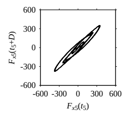
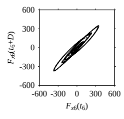
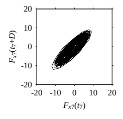
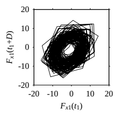
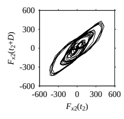
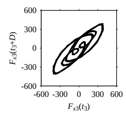
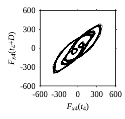
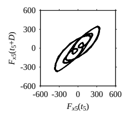
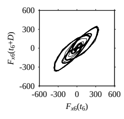
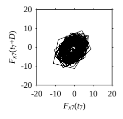
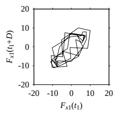
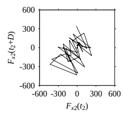
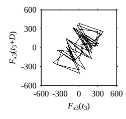
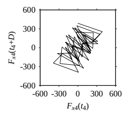
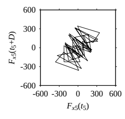
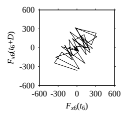
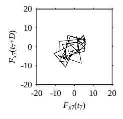
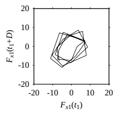
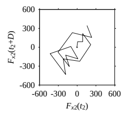
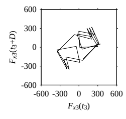
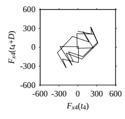
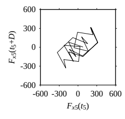
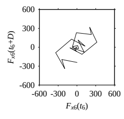
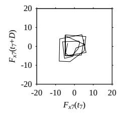
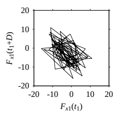
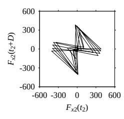
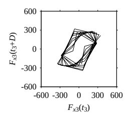
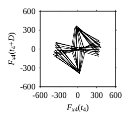
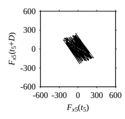
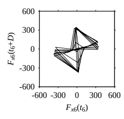
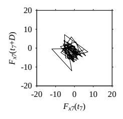
Sampled signals						
$F_{x1}(t_1)$	$F_{x2}(t_2)$	$F_{x3}(t_3)$	$F_{x4}(t_4)$	$F_{x5}(t_5)$	$F_{x6}(t_6)$	$F_{x7}(t_7)$
Delay maps for varying d						
$d = 1$						
						
$d = 2$						
						
$d = 5$						
						
$d = 50$						
						
$d = 100$						
						

Table 4.7: Delay maps for $F_{x1}(t_1), ..., F_{x7}(t_7)$ (corresponds to $MS_1, ..., MS_7$) when $d = 43$.

Sampled signals						
$F_{x1}(t_1)$	$F_{x2}(t_2)$	$F_{x3}(t_3)$	$F_{x4}(t_4)$	$F_{x5}(t_5)$	$F_{x6}(t_6)$	$F_{x7}(t_7)$
Delay maps for $d = 43$						
						

map exhibits a distinct pattern are valuable for Industry 4.0-centric systems. In this case, the systems make one-to-one correspondence between a given machining situation and its delay map, which is ideal for pattern recognition. This means that a delay map can be a signature of a machining situation. For example, for the signals reported in Figure 4.17, the delay maps exhibit unique patterns when $d = 43$, as shown in Table 4.7. In other words, the delay maps corresponding to $d = 43$ are the signatures of the machining situations $MS_1, ..., MS_7$.

4.6 Summary

In this chapter, as a first step, two arbitrary chaotic signals are studied that look alike in the time and frequency domains, but their differences can be clearly understood in the delay domain. This means that the delay domain must be incorporated to make sense of chaotic sensor signals.

In the next step, another arbitrary and real-life sensor signals relevant to manufacturing, particularly cutting force signals collected while machining three different materials (Stainless Steel, Mild Steel, and Ductile Cast Iron), are analyzed in frequency and delay domains. It is found that when the delay increases, the frequency spectrum gets affected. The prominent frequencies of the original signal are gradually lost due to aliasing when the delay exceeds a critical value. On the other hand, when delay increases, the delay domain gets more and more scattered. For some critical values of delay (one may be very high and the other may be very low), the delay domains exhibit similar characteristics, which is not the case for the frequency domains. Thus, when a very short window or low sampling rate (high delay) is used to analyze a signal, the delay domains guarantee its (signal's) original nature. This means that the delay domain-based representation is more robust in understating the nature of a sensor signal subjected to high time latency or delay.

Lastly, the potential of the delay domain becoming a signature of a machining situation is studied using real-life signals. For this, sensor signals are sampled exhibiting seven different machining situations representing the onset and completion of machining, machining different materials, and the transitions among the aforementioned situations. For some specific delays, the delay maps make one-to-one correspondence with machining situations. This means that a delay domain of a machine situation is different from the delay domains of other machining situations for a critical delay. As a result, a delay domain can be used as a signature of a machining situation.

The abovementioned examples show that the dynamics underlying a signal becomes complex due to delay. In such cases, the delay domain is more informative than the time and frequency domains. However, in CPS, a random delay is most likely to be associated with the sensor signals. As a result, while constructing and adapting the sensor signal-based DTs (phenomena twins), the relevant systems (DTCS and DTAS, as shown in Figure 1.3) must accommodate delay domain-based signal processing technique. Based on this consideration, this study proposes an architecture of the sensor signal-based DTs where the sensor signal is processed in the delay domain subjected to a random delay, as follows.

Chapter 5

Sensor Signal-Based Digital Twin

This chapter presents the proposed systems called Digital Twin Construction System (DTCS) and Digital Twin Adaptation System (DTAS) for developing Digital Twins (DTs) of machining phenomena based on semantic annotation (see Chapter 3 for details) and time-delayed (see Chapter 4 for details) sensor signals. For better understanding, this chapter is organized as follows. Section 5.1 describes the interplay of DTCS and DTAS in a Cyber-Physical System (CPS). Section 5.2 presents the requirements underlying the systems (DTCS and DTAS). Section 5.3 presents the modular architectures of the systems. Finally, Section 5.4 summarizes this chapter.

5.1 Systems' Context

As seen in Figure 5.1, the interplay between the systems (DTCS and DTAS) undergoes two phases. In the first phase, the DTCS constructs the DT. In the other phase, the DTAS injects the constructed DT into the knowledge base. As seen in Figure 5.1, the injected DT produces outcomes (a simulated sensor signal) whenever necessary. On the other hand, the machine tool produces a sensor signal while operating. These two signals (simulated and real) are compared to decide the course of action while performing the intelligent machine tool's monitoring and troubleshooting activities.

As described before, when the DT is in the use phase (in this case monitoring phase, as shown on the left-hand side in Figure 5.1), it is subjected to delay as denoted by "d" in Figure 5.1. Therefore, the DT must be con-

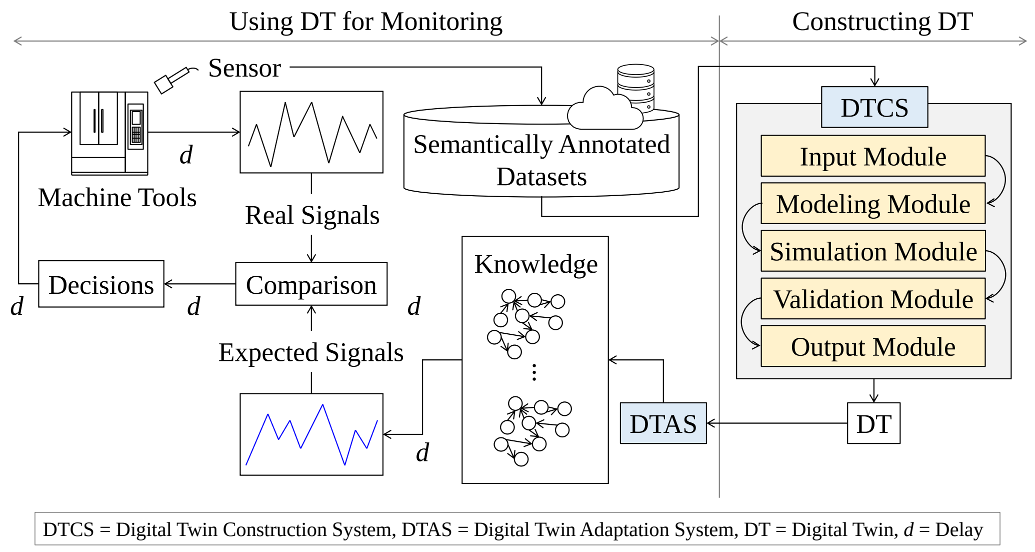


Figure 5.1: Interplay between the Digital Twin Construction System (DTCS) and Digital Twin Adaptation System (DTAS).

structed based on a delay-embedded signal processing method. Otherwise, the comparison does not make any sense. Based on this contemplation, this study proposes a DTCS and DTAS, as described in the following sections.

5.2 Systems' Requirements

As seen on the right-hand side in Figure 5.1, the DTCS consists of five modules: (1) Input Module, (2) Modeling Module, (3) Simulation Module, (4) Validation Module, and (5) Output Module. The modules are intertwined and they collectively constitute the DT. The fundamental purposes of these modules are: (1) Acquiring the right piece of sensor signal data from a given data repository, (2) Extracting knowledge underlying the acquired sensor signal, (3) Simulating sensor signal(s) relevant to the acquired one based on the extracted knowledge, (4) Comparing the characteristics between the real (acquired) and simulated signals, and (5) Transferring the constructed DT to the DTAS, respectively.

For this, these modules collectively carry out the functional requirements called Data Management, Ontology, Modeling, Machine Learning, Simulation, Validation, Real-Time Response, and Semantic Web Compatibility. Here, the function called Data Management means to collect and manage data from a data storage system (e.g., cloud-based data storage system, can be seen in Figures 1.3 and 5.1). Ontology means the high-level descriptions of the datasets and associated entities (e.g., machining process, machining conditions, sensor, sensor signal, work-piece, cutting tool, machines, and alike) in the form of user-defined semantic annotations, making the contents compatible with semantic web and concept maps. Modeling means to model the relevant phenomenon, making the machine learning from sensor signal possible. Machine Learning means to learn rules for simulation using a suitable machine learning approach (e.g., neural networks, deep learning, DNA-based computing, Markov chain, and alike). Simulation means to simulate a signal on demand using a suitable approach (e.g., discrete event-based Monte Carlo simulation and deterministic simulation). Validation means to validate the simulation outcomes using a suitable approach (e.g., possibility distribution [92], entropy [86], DNA-based computing [93], Decisional DNA [94], and alike), ensuring the trustworthiness of the simulation process. Real-Time Response means re-configuring the relevant modules responding to real-time signal data updates [8, 29]. Semantic Web Compatibility means to make the

Table 5.1: Functional requirements and fundamental purposes of the modules underlying the Digital Twin Construction System (DTCS).

	Modules underlying the DTCS				
Functional requirements	**Input Module**	**Modeling Module**	**Simulation Module**	**Validation Module**	**Output Module**
Ontology	•				•
Semantic Web Compatibility	•				•
Real-Time Response	•	•	•	•	•
Data Management	•				•
Modeling		•			
Machine Learning		•			
Simulation			•		
Validation				•	
	Fundamental purposes of the modules				
	Signal acquisition	Knowledge extraction	Signal recreation	Comparison	DT transfer

whole process of digital twining compatible with the semantic web. The relationship among the abovementioned functional requirements and the modules of the DTCS are shown in Table 5.1.

As seen in Table 5.1, Real-Time Response is associated with all the modules. Apart from it (Real-Time Response), the Input Module is associated with the Ontology, Semantic Web Compatibility, and Data Management functional requirements. Similarly, the Modeling Module is associated with the Modeling and Machine Learning functional requirements. The Simulation Module is associated with the Simulation functional requirement. The Validation Module is associated with the Validation functional requirement. The Output Module is associated with the Ontology, Semantic Web Compatibility, and Data Management functional requirements. If the modules can execute the abovementioned functional requirements, then the DT is said to be constructed properly. This means that the relationships shown in Table 5.1 are the design guidelines for constructing a sensor signal-based DT of a machining phenomenon.

Recall the functional requirement called Real-Time Response, which is involved with all the modules. In particular, this functional requirement is subjected to time latency or delay (see Chapter 4 for details). Therefore, it (delay) needs careful consideration. Otherwise, when the DT is in use (see the left-hand side scenario shown in Figure 5.1), it may not produce the desired outcome.

Nevertheless, based on the abovementioned requirements, the modular architectures of the systems (DTCS and DTAS) are presented as follows.

5.3 Modular Architectures

The modular architectures of the abovementioned systems, i.e., DTCS and DTAS, are presented in the following sub-sections (Sections 5.3.1 and 5.3.2, respectively).

5.3.1 Digital Twin Construction System (DTCS)

First, consider the computerized system called DTCS. As mentioned in the previous section (Section 5.2), the proposed DTCS consists of five modules: Input Module, Modeling Module, Simulation Module, Validation Module, and Output Module. These modules collectively construct the DT. The modular architectures of these modules are presented below.

Input Module

The Input Module is the first module of the proposed DT. The associated functional requirements are Ontology, Semantic Web Compatibility, Data Management, and Real-Time Response. Therefore, the Input Module must deal with semantically annotated contents. The ideal case would be to link it to a semantic web-embedded data repository. In reality, this may not be the case. As such, the Input Module must be linked to an ontology-embedded repository. Besides, it must respond to any content update in real-time. Since the goal here is to create a DT based on sensor signal data, the Input Module must provide a facility to search and select the relevant sensor signal data.

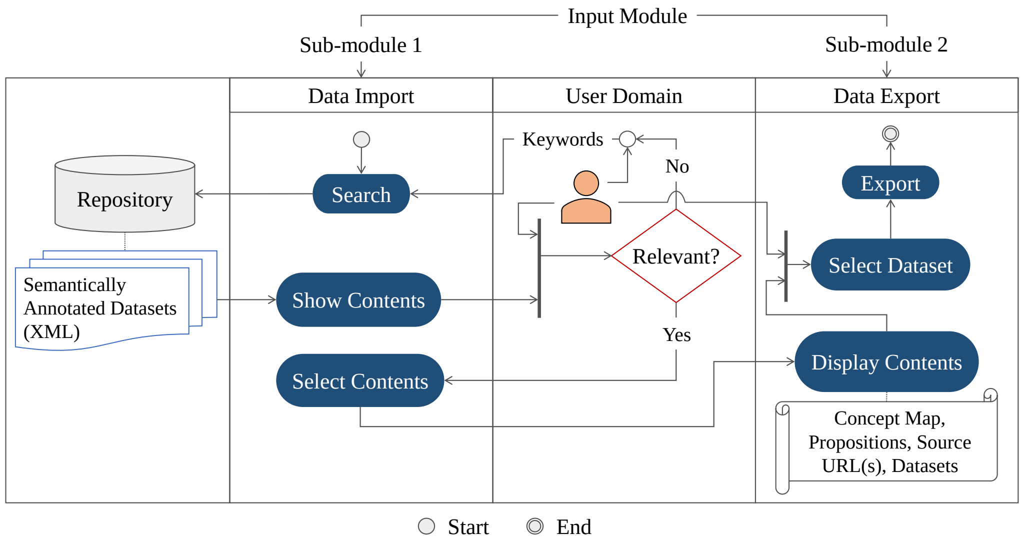


Figure 5.2: Modular architecture of the Input Module.

Based on the abovementioned consideration, a repository is created where signal data of different machining phenomena (e.g., cutting force, cutting torque, surface roughness, and alike) are stored using the Extensible Markup Language (XML) file format. The contents are semantically annotated where the relevant concepts (e.g., machining process, cutting conditions, sensor, sensor signal datasets, cutting tool, and alike) and their relationships are defined. Thus, the content takes the form of concept maps. The relevant node of each concept map contains the numerical datasets of a phenomenon. For better understanding, see the semantic annotation-based knowledge representation mechanism described in Chapter 3. However, the Input Module interacts with the repository using two sub-modules, denoted as Data Import and Data Export sub-modules. Figure 5.2 schematically illustrates the modular architecture of this module.

As seen in Figure 5.2, the Data Import sub-module imports semantically annotated contents from the repository based on a user-defined keyword and delivers the contents to the Data Export sub-module. For this, it operates based on 'search-show-select' functions. The Data Export sub-module exports the imported contents to the desired location (e.g., Modeling Module). For this, it operates based on 'display-select-export' functions.

Modeling Module

The Modeling Module is the second module of the proposed DT. The associated functional requirements are Modeling, Machine Learning, and Real-Time Response. It (Modeling Module) first acknowledges the signal data (exported by the Input Module) and then models the underlying phenomenon using a predefined machine learning approach. One of the approaches (e.g., neural networks, deep learning, semantic modeling, DNA-based computing, or Markov chain) can be used to create a model. Besides, the module must rebuild the model responding to any update in the exported signal in real-time. Since the delay is an inherent characteristic of the systems where the DT is supposed to work (see Figure 5.1), a model created from a given delay map of a signal is perhaps the best option. As such, the Modeling Module uses Markov chain-based modeling approach since it highly correlates with delay maps [95, 96].

Based on the abovementioned consideration, a modular architecture of the Modeling Module is proposed. Figure 5.3 schematically illustrates the modular architecture. As seen in Figure 5.3, the Modeling Module consists of three sub-modules, denoted as Data Import, Delay Settings, and Markov Chain-Based Modeling sub-modules. The Data Import sub-module imports the dataset exported by the Input Module, and displays its (dataset) time series and delay map. The Delay Settings sub-module allows the user to set a delay out of three choices (default, constant, random), and thereby, constructs delay-driven dataset. It also displays the delay-driven dataset (time series and delay map) for the sake of user comprehensibility. As such,

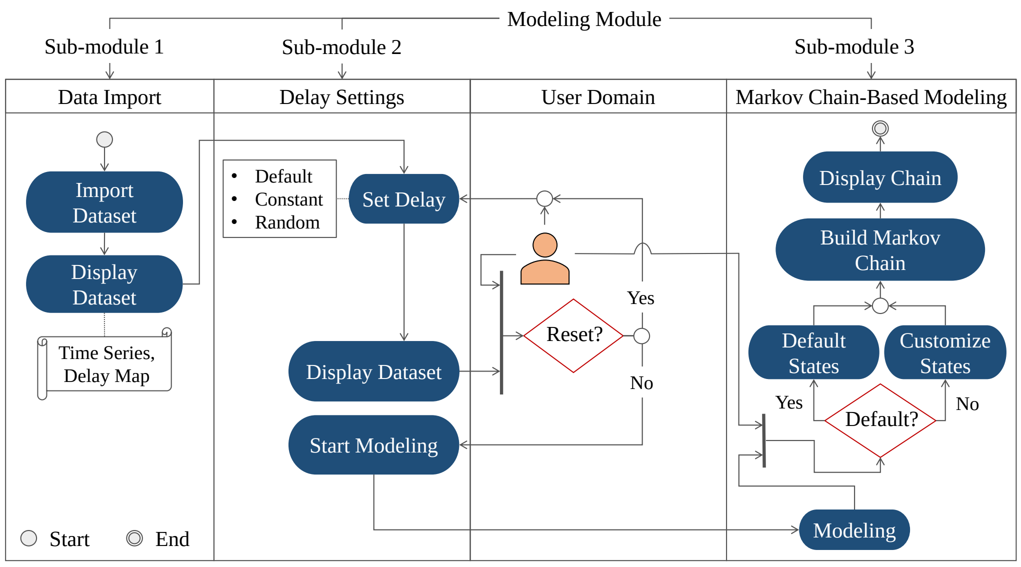


Figure 5.3: Modular architecture of the Modeling Module.

the user may reset the delay if required. The Markov Chain-Based Modeling sub-module extracts knowledge underlying the delay map (obtained from the Delay Settings sub-module) in the form of a Markov chain. For this, the user may use a set of default or customized latent states. Note that the Markov chain-based modelling approach and relevant mathematical formulations are described in Appendix C.

Simulation Module

The Simulation Module is the third module of the proposed DT. The associated functional requirements are Simulation and Real-Time Response. It (Simulation Module) first acknowledges the model created by the Modeling Module and then simulates the phenomenon using a predefined simulation approach (discrete event-based Monte Carlo simulation or deterministic simulation). Besides, the module must respond to the model update on a real-time basis. Since the Modeling Module uses the Markov chain, the Simulation Module uses a discrete event-based Monte Carlo simulation approach. The module first simulates the latent states (states defined in the Modeling Module) and translates the states into their numerical counterpart based on a predefined probability distribution. Note that the relevant simulation algorithm is described in Appendix D.

Based on the abovementioned consideration, a modular architecture of

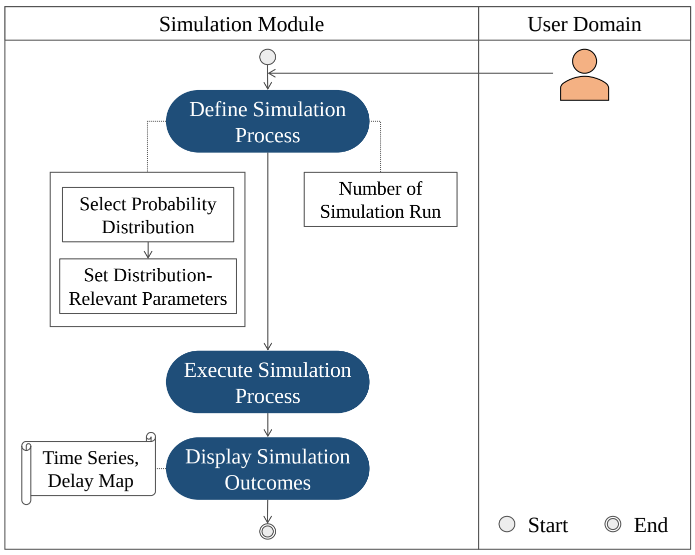

Simulation Module
User Domain
Define Simulation Process
Select Probability Distribution
Set Distribution-Relevant Parameters
Number of Simulation Run
Execute Simulation Process
Time Series, Delay Map
Display Simulation Outcomes
Start
End

Figure 5.4: Modular architecture of the Simulation Module.

the Simulation Module is proposed. Figure 5.4 schematically illustrates the modular architecture. As seen in Figure 5.4, the module operates based on ‘define-execute-display’ functions. The user defines the Monte Carlo simulation approach by selecting a probability distribution and setting the distribution-relevant parameters. The user also decides how many times the same simulation process should run. Based on these, the module executes the simulation process and displays the simulation outcomes in the form of time series and delay maps.

Validation Module

The Validation Module is the fourth module of the proposed DT. The associated functional requirements are Validation and Real-Time Response. It (Validation Module) verifies the simulation outcomes’ appropriateness, using one or more predefined validation approaches (e.g., possibility distribution [92], entropy [86], DNA-based computing [93], Decisional DNA [94], and alike). Note that the module can also adapt phenomenon-dependent parameters (e.g., average surface height or *Ra*, a commonly used surface roughness parameter) for the sake of verification, if required. However, it (module) then selects and stores some of the validated simulation outcomes for reuse. Besides, the module must respond to any update in the simulation outcomes on a real-time basis.

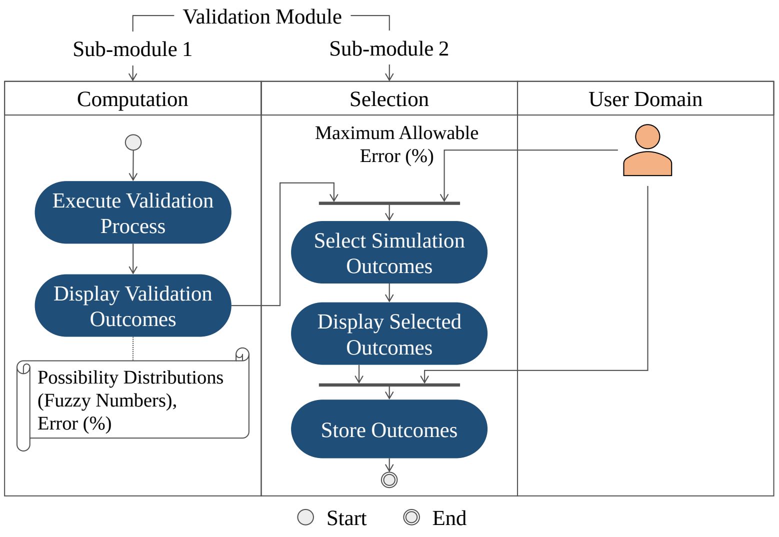

Validation Module
Sub-module 1
Sub-module 2
Computation
Selection
User Domain
Maximum Allowable Error (%)
Execute Validation Process
Display Validation Outcomes
Possibility Distributions (Fuzzy Numbers), Error (%)
Select Simulation Outcomes
Display Selected Outcomes
Store Outcomes
Start
End

Figure 5.5: Modular architecture of the Validation Module.

Based on the abovementioned consideration, a modular architecture of the Validation Module is proposed. Figure 5.5 schematically illustrates the modular architecture. As seen in Figure 5.5, the module consists of two sub-modules, denoted as Computation and Selection sub-modules. The Computation sub-module is responsible for executing the validation process. In this study, it (Computation sub-module) uses possibility distribution (fuzzy numbers) [92] for the sake of validation. Note that other parameters can be integrated if required. On the other hand, the Selection sub-module is responsible for selecting and storing some of the validated outcomes based on some user inputs.

Output Module

The Output Module is the last one. The associated functional requirements are Ontology, Semantic Web Compatibility, Data Management, and Real-Time Response. It (Output Module) acts as the connecting point between the construction (DTCS) and adaptation (DTAS) systems. As such, it must interact with the other modules (Input Module, Modeling Module, Simulation Module, and Validation Module), acknowledge their contents, and export them to the DTAS whenever required. Besides, it must respond to any update in any of the modules mentioned above in real-time.

Figure 5.6 schematically illustrates the modular architecture of this mod-

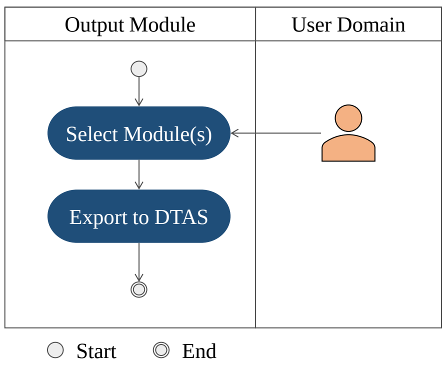


Figure 5.6: Modular architecture of the Output Module.

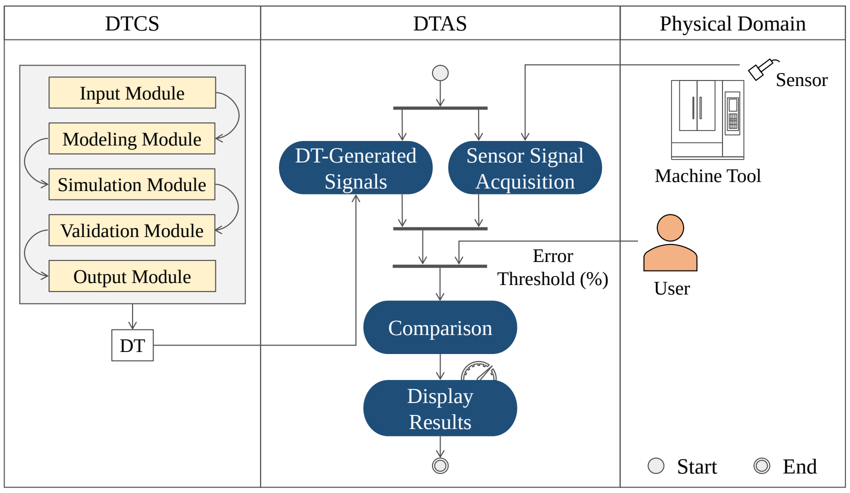


Figure 5.7: Modular architecture of the DTAS.

ule. As seen in Figure 5.6, the module operates based on ‘select-export’ functions. As such, the module transfers the constructed DT to the DTAS whenever required.

5.3.2 Digital Twin Adaptation System (DTAS)

Now, let us consider the other computerized system called DTAS. The DTAS uses the constructed DT for real-time monitoring. As such, Figure 5.7 schematically illustrates the modular architecture of the DTAS.

As seen in Figure 5.7, the DTAS acknowledges the DT-generated signals and uses the signals to monitor a real-life machining process in real-time.

For this, it (DTAS) uses a user-defined error threshold (%). It displays the monitoring results so that the user can decide the right course of action.

5.4 Summary

This chapter describes two computerized systems denoted as Digital Twin Construction System (DTCS) and Digital Twin Adaptation System (DTAS), required for constructing and adapting sensor signal-based DTs (or digital twins of machining phenomena) in a Cyber-Physical System (CPS). The systems' context, requirements, and modular architectures are elucidated in detail.

Based on the abovementioned architectures, the DTCS and DTAS are developed using a Java^{TM}-based platform. The efficacy of the developed systems is also demonstrated considering a real-life machining experiment, as presented in the following chapter.

Chapter 6

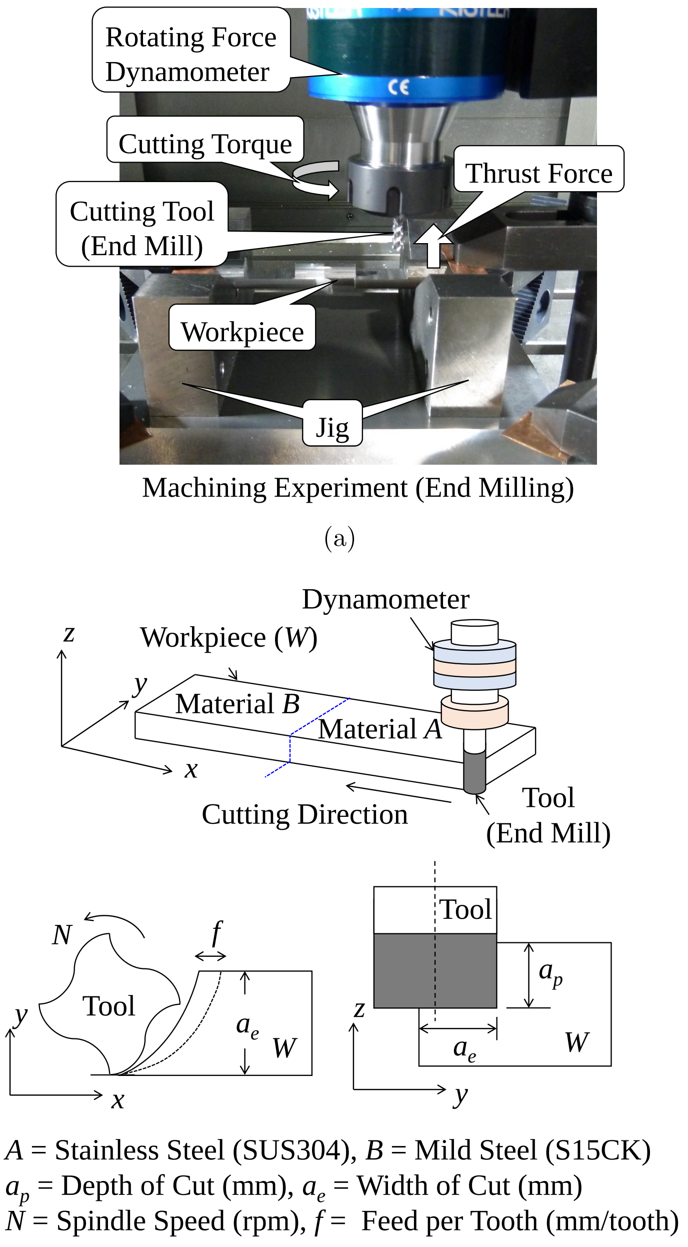
Use of the Digital Twin

This chapter presents the development of the sensor signal-based Digital Twin (DT) of a machining phenomenon. As described in Chapter 5, the DT consists of two systems: DTCS and DTAS. The systems are developed using a Java™-based platform, following the proposed modular architectures of the systems (see Chapter 5 for details). The developed systems (DTCS and DTAS) are tested for demonstrating their efficacy in developing the DTs from semantically annotated and delay-centric sensor signal datasets, and monitoring the real-life machining experiments, respectively.

For better understanding, this chapter is organized as follows. Section 6.1 describes a machining experiment, followed by the findings from the efficacy test described in Section 6.2. Finally, Section 6.3 summarizes this chapter.

6.1 Machining Experiment

Figure 6.1 shows an instance of the machining experiment (see Figure 6.1a) and also schematically illustrates it (see Figure 6.1b). As seen in Figure 6.1b, a bimetallic workpiece made of Stainless Steel (JIS: SUS304) and Mild Steel (JIS: S15CK) is machined, following a material removal process called end milling. The workpiece is machined from hard-to-soft material direction, i.e., from SUS304 to S15CK. The relative cutting conditions for machining, such as depth of cut (a_p), width of cut (a_e), spindle speed (N), and feed per tooth (f), are also depicted in detail in Figure 6.1b. The cutting conditions and machining equipment (e.g., machine tool, cutting tool, sensor, workpiece,


Rotating Force Dynamometer
Cutting Torque
Thrust Force
Cutting Tool (End Mill)
Workpiece
Jig
Machining Experiment (End Milling)
(a)
Dynamometer
z
Workpiece (W)
y
Material B
Material A
x
Cutting Direction
Tool (End Mill)
N
f
Tool
y
a_e
W
x
Tool
a_p
z
a_e
W
y
A = Stainless Steel (SUS304), B = Mild Steel (S15CK)
a_p = Depth of Cut (mm), a_e = Width of Cut (mm)
N = Spindle Speed (rpm), f = Feed per Tooth (mm/tooth)

(b)

Figure 6.1: End milling of a bimetallic workpiece. (a) Real-life machining experiment, (b) Schematic and relative cutting conditions.

and alike) are also reported in Table 6.1.

As seen Table 6.1, the bimetallic workpiece is machined following three sets of cutting conditions. For each set, relevant cutting torque signals are sensed and acquired from the machining environment using a rotary dynamometer (as reported in Table 6.1, can also be seen in Figure 6.1).

As seen in Figure 6.2, the cutting torque signals—acquired from the machining environment—are stored in a cloud-based data repository. For this, the semantic annotation-based representation mechanism (see Chapter 3) is followed. In particular, a concept map is constructed by annotating the signal datasets with other relevant entities (machine tool, cutting tool, workpiece, sensor, cutting conditions, and alike), using some user-defined semantics. Figure 6.3 shows the constructed concept map. One may view the concept map from the following url: `https://cmapscloud.ihmc.us/viewer/cmap/1XNM0FC40-12MXDQ7-DS`. Nevertheless, the concept map boils down to the

Table 6.1: Conditions for end milling.

Item	**Description**		
Machine tool	Vertical machining center Make: Mori Seiki Model: NV5000		
Cutting tool	Carbide Φ6 solid end mill Make: Mitsubishi Hitachi Model: EPP4060-P-CS		
Sensor	Rotary dynamometer Make: Kistler Type: 9170A		
Workpiece material	Bimetallic material made of Stainless Steel (JIS: SUS304) and Mild Steel (JIS: S15CK)		
Cutting direction	SUS304 to S15CK		
Cutting conditions	**Case 1**	**Case 2**	**Case 3**
Cutting velocity (V_c)	180 m/min	180 m/min	180 m/min
Spindle speed (N)	9554 rpm	9554 rpm	9554 rpm
Feed per tooth (f)	0.1 mm/tooth	0.1 mm/tooth	0.2 mm/tooth
Feed rate (V_f)	3822 mm/min	3822 mm/min	3822 mm/min
Depth of cut (a_p)	1.0 mm	2.0 mm	1.0 mm
Width of cut (a_e)	0.5 mm	0.5 mm	0.5mm

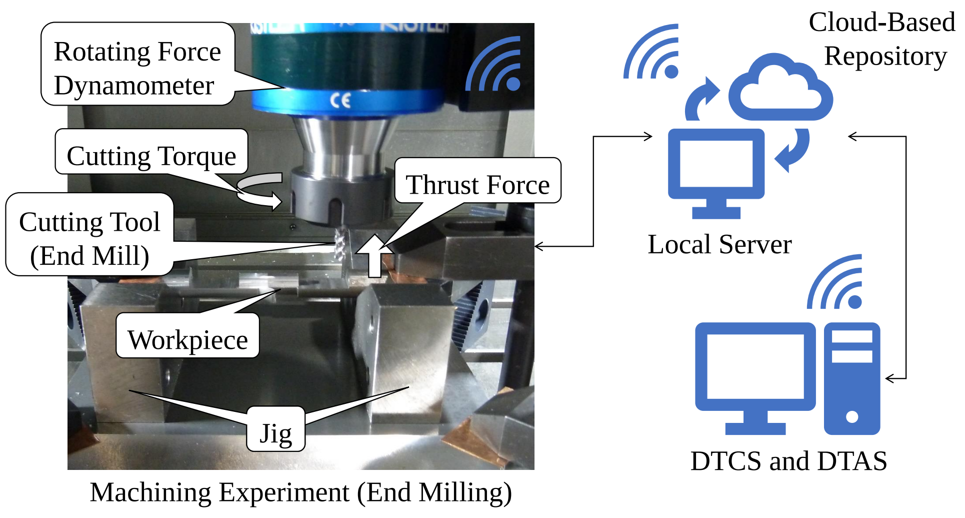


Figure 6.2: Basic framework for the efficacy test of DTCS and DTAS.

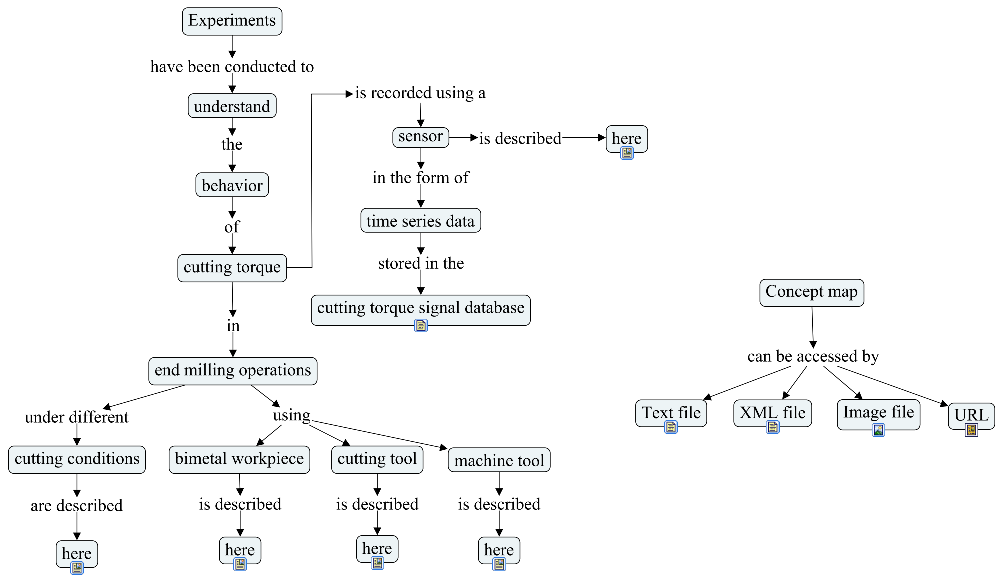


Figure 6.3: A concept map for the machining experiment.

following propositions (P_1, ..., P_{10}).

(P_1) Experiments have been conducted to understand the behavior of cutting torque in end milling operations under different cutting conditions,

(P_2) cutting conditions are described here,

(P_3) Experiments have been conducted to understand the behavior of cutting torque in end milling operations using bimetal workpiece,

(P_4) bimetal workpiece is described here,

(P_5) Experiments have been conducted to understand the behavior of cutting torque in end milling operations using cutting tool,

(P_6) cutting tool is described here,

(P_7) Experiments have been conducted to understand the behavior of cutting torque in end milling operations using machine tool,

(P_8) machine tool is described here,

(P_9) cutting torque is recorded using a sensor in the form of time series data stored in the cutting torque signal database, and

(P_{10}) sensor is described here.

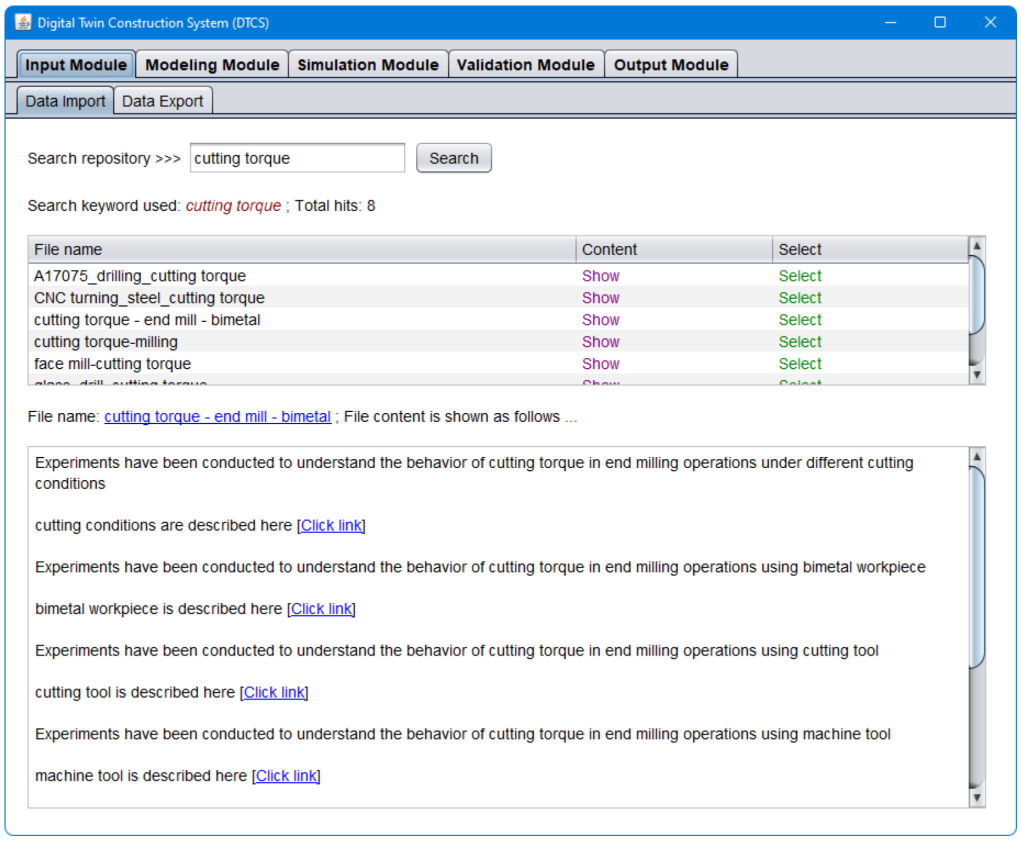
The concept/node called 'here' in the abovementioned propositions (can also be seen in Figure 6.3) is embedded with relevant resources (e.g., documents, datasets, and alike) to support the semantic relationships among all the entities. This way the concept mapping helps other enablers and stakeholders (human resources, machines, and systems) understand the overall context underlying a machining experiment in a more effective manner, before reusing a piece of the signal datasets (here, torque signal datasets). For this, the concept map is shared over the web in the form of XML data as described in Chapter 3.

Now, recall the scenario shown in Figure 6.2, where it is shown that the abovementioned semantically annotated torque signal datasets are stored in a cloud-based data repository in the form of XML data. The DTCS acknowledges the annotated datasets from the repository among others, and constructs the relevant DTs based on delay. On the other hand, the DTAS performs real-time monitoring of a real-life machining experiment (here, end milling) based on the constructed DTs. The following section describes this in detail.

6.2 In-Process Monitoring

6.2.1 Digital Twins' Construction

Consider the abovementioned machining phenomenon called cutting torque created due to end milling (as described in Section 6.1). In this case, the user can use the Data Import sub-module of the Input Module (see Figure 6.4) to search semantically annotated contents based on a keyword (here, cutting torque). The sub-module provides a facility ('Show' button in Figure 31) to visualize the fetched content. It also provides a facility ('Select' button in Figure 6.4) to import the desired content and deliver it to the Data Export sub-module of the Input Module.



Search repository >>> cutting torque

Search keyword used: *cutting torque* ; Total hits: 8

File name	Content	Select
A17075_drilling_cutting torque	Show	Select
CNC turning_steel_cutting torque	Show	Select
cutting torque - end mill - bimetal	Show	Select
cutting torque-milling	Show	Select
face mill-cutting torque	Show	Select
[illegible]	[illegible]	[illegible]

File name: cutting torque - end mill - bimetal ; File content is shown as follows ...

Experiments have been conducted to understand the behavior of cutting torque in end milling operations under different cutting conditions

cutting conditions are described here [Click link]

Experiments have been conducted to understand the behavior of cutting torque in end milling operations using bimetal workpiece

bimetal workpiece is described here [Click link]

Experiments have been conducted to understand the behavior of cutting torque in end milling operations using cutting tool

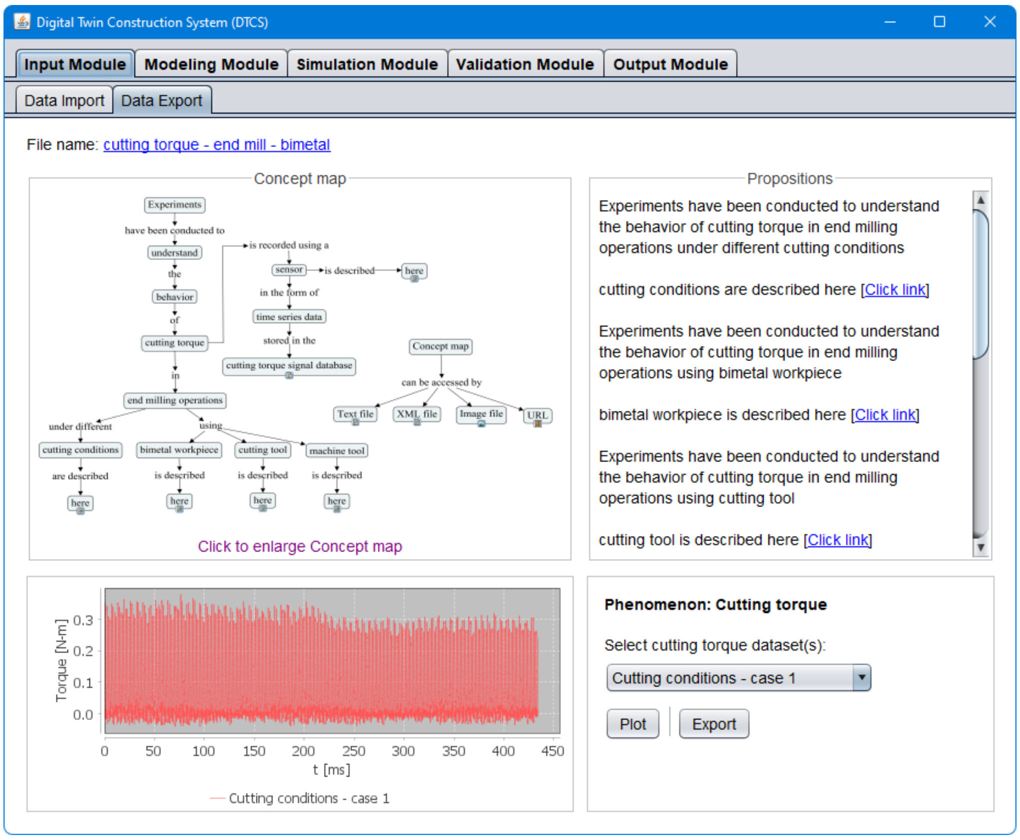
cutting tool is described here [Click link]

Experiments have been conducted to understand the behavior of cutting torque in end milling operations using machine tool

machine tool is described here [Click link]

Figure 6.4: Screen-print of Data Import sub-module of Input Module.

Figure 6.5 shows the user interface of the Data Export sub-module. It first displays the imported content in the form of a concept map, a list of propositions, and graphical plots of numerical datasets. The sub-module then provides a facility ('Plot' button in Figure 6.5) to visualize and select the datasets. It also provides a facility ('Export' button in Figure 6.5) to export the selected dataset to the Modeling Module or any other data storage systems. One of the remarkable aspects of this Input Module is its scalability with user-defined ontology. For example, the contents shown in Figure 6.5 (concept map and underlying propositions) can be represented in various ways in the repository. The module can respond to such variability. This means that the module does not depend on strict ontological formalism. Rather, it provides a flexible way of annotating, and thereby, creating machine and human-readable content.



File name: cutting torque - end mill - bimetal

Experiments have been conducted to understand the behavior of cutting torque in end milling operations under different cutting conditions

cutting conditions are described here [Click link]

Experiments have been conducted to understand the behavior of cutting torque in end milling operations using bimetal workpiece

bimetal workpiece is described here [Click link]

Experiments have been conducted to understand the behavior of cutting torque in end milling operations using cutting tool

cutting tool is described here [Click link]

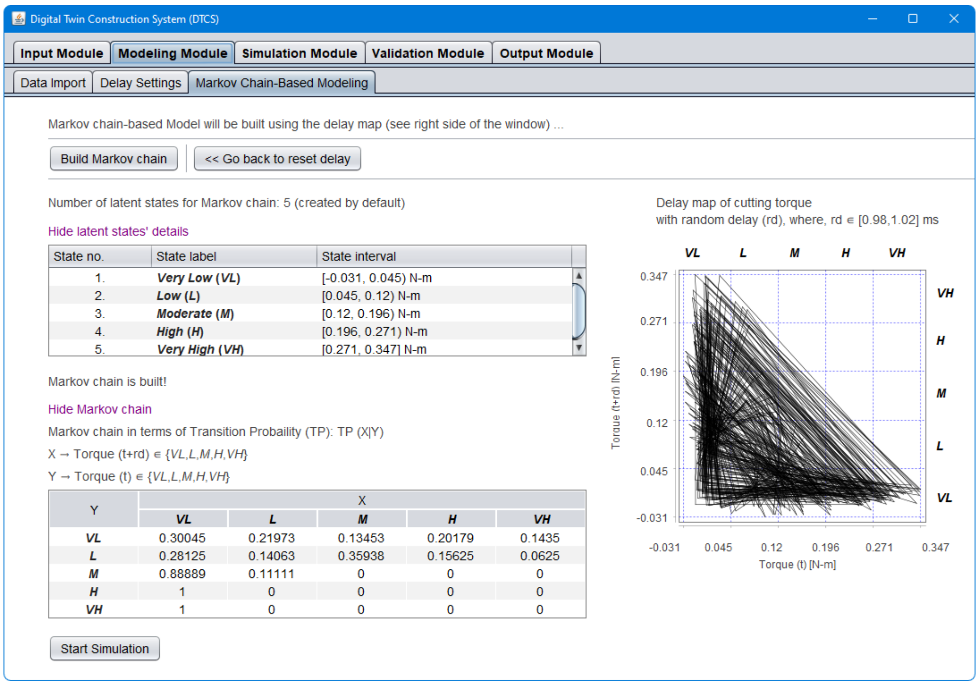
Phenomenon: Cutting torque

Select cutting torque dataset(s):

Figure 6.5: Screen-print of Data Export sub-module of Input Module.

Figure 6.6 shows the user interface of the Modeling Module when a Markov chain is created from a delay map of a cutting torque signal (imported from the Input Module, see Figure 6.5). The map corresponds to a random delay, that is, 1 ± 0.02 ms. Five latent states, namely *Very Low* (*VL*), *Low* (*L*), *Moderate* (*M*), *High* (*H*), and *Very High* (*VH*) are used. The transition probabilities of these states are calculated and displayed. As such, the transition matrix serves as the model (Markov chain) of the given torque signal. If a user prefers, the other labels and the number of latent states can be used for modeling. The preferences can be set using the user interface (not visible in Figure 6.6). Note that the mathematical formulations for Markov chain are described in Appendix C.

Figure 6.7 shows the user interface of the Simulation Module. Here the user sets a probability distribution to simulate the numerical counterparts of



Markov chain-based Model will be built using the delay map (see right side of the window) ...

Number of latent states for Markov chain: 5 (created by default)

State no.	State label	State interval
1.	***Very Low (VL)***	[-0.031, 0.045) N-m
2.	***Low (L)***	[0.045, 0.12) N-m
3.	***Moderate (M)***	[0.12, 0.196) N-m
4.	***High (H)***	[0.196, 0.271) N-m
5.	***Very High (VH)***	[0.271, 0.347] N-m

Markov chain is built!

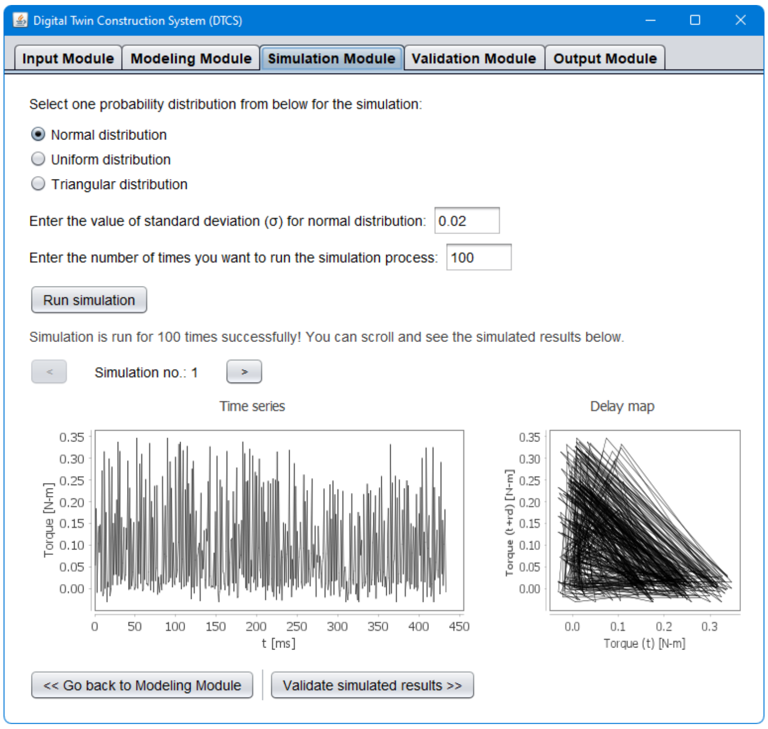
Markov chain in terms of Transition Probaility (TP): TP (X|Y)

X → Torque (t+rd) ∈ {*VL,L,M,H,VH*}

Y → Torque (t) ∈ {*VL,L,M,H,VH*}

Y	X				
	VL	***L***	***M***	***H***	***VH***
VL	0.30045	0.21973	0.13453	0.20179	0.1435
L	0.28125	0.14063	0.35938	0.15625	0.0625
M	0.88889	0.11111	0	0	0
H	1	0	0	0	0
VH	1	0	0	0	0

Figure 6.6: Screen-print of Modeling Module.



Select one probability distribution from below for the simulation:

[✓ Normal distribution]

Uniform distribution

Triangular distribution

Enter the value of standard deviation (σ) for normal distribution: 0.02

Enter the number of times you want to run the simulation process: 100

Simulation is run for 100 times successfully! You can scroll and see the simulated results below.

Simulation no.: 1

Figure 6.7: Screen-print of Simulation Module.

latent states. The user can choose a probability distribution out of normal, uniform, triangular distributions and set the corresponding parameters. In addition, the user can set the number of simulations instances. In the case shown in Figure 6.7, the user has chosen the normal distribution and set the standard deviation equal to 0.02. Other values can be used if preferred. In addition, the user has set the number of simulation instances equal to 100. As such, the system runs the same simulation process 100 times and creates 100 sets of torque signals. The simulation outcomes (time series and delay map) are shown in the user interface. The user can scroll the time series and delay maps to see the consistency of the simulation process. Note that the corresponding simulation algorithm is described in Appendix D.

Figure 6.8 shows the user interface of the Computation sub-module of the Validation Module. It displays the time series and delay map used to construct the model in the Modeling Module (black-colored plots). It also displays the simulated time series and delay map created in the Simulation Module (blue-colored plots). (In Figure 6.8, the simulated time series and delay map correspond to simulation number 96 out of 100 simulations.) Whether or not the simulated outcomes correspond to the real one, the Computation sub-module induces possibility distributions (fuzzy numbers) [92], quantifying the uncertainty in the real and simulated outcomes. The sub-module displays the corresponding possibility distributions for the visual inspection. The user can scroll the number of simulations and see the corresponding results. In the graphs of possibility distributions, the induced triangular fuzzy numbers are also displayed (purple-colored curves). Apart from visual inspection, four parameters, denoted as Area, Average, Core, and Support, are used to quantify the similarity between the possibility distributions, as shown on the right-hand side in Figure 6.8. This results in a single error measure, defined as Error (see Appendix E for details). The instance shown in Figure 6.8 corresponds to the Error of 3%.

Figure 6.9 shows the user interface of the Selection sub-module of the Validation Module. It first displays the error summary in the form of a scatter plot between the Error and the number of simulations (shown on the left-hand side in Figure 6.9). It then provides a facility to select some of the simulation outcomes based on a user-defined maximum allowable Error. The case shown in Figure 6.9 corresponds to the maximum allowable Error of 10%. As such, the sub-module retrieves 39 simulation outcomes and displays corresponding results in the form of time series and delay maps (shown on the right-hand side in Figure 6.9). The user can scroll the retrieved outcomes and see the corresponding results. The user can store all or some of the

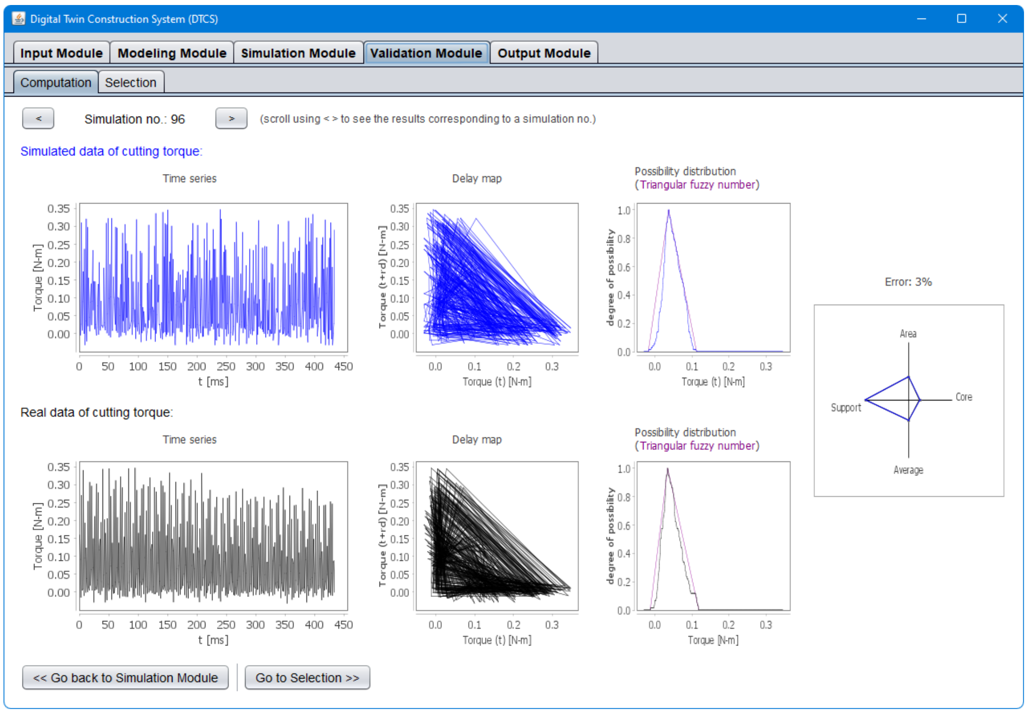
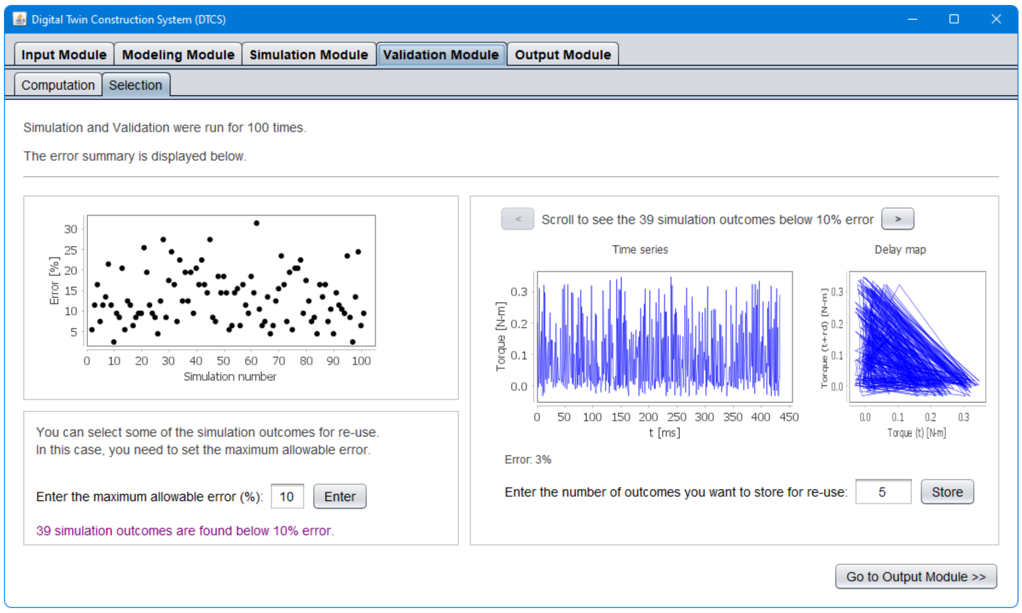


Figure 6.8: Screen-print of Computation sub-module of Validation Module.



Simulation and Validation were run for 100 times.

The error summary is displayed below.

You can select some of the simulation outcomes for re-use.
In this case, you need to set the maximum allowable error.

Enter the maximum allowable error (%): 10

39 simulation outcomes are found below 10% error.

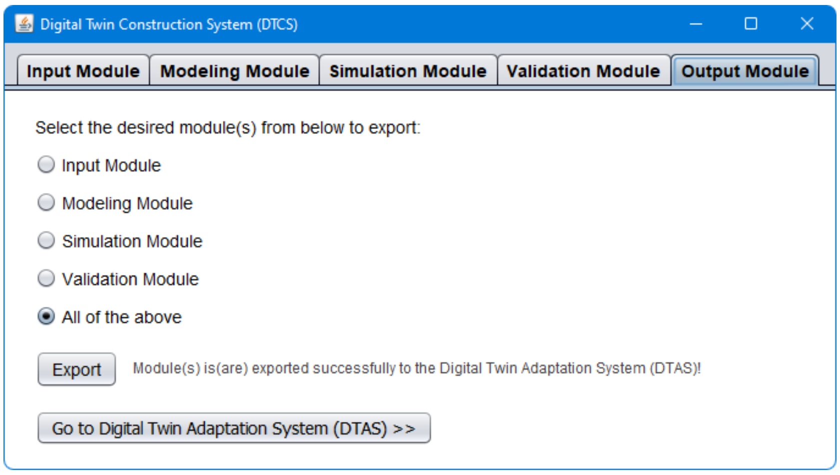
Error: 3%

Enter the number of outcomes you want to store for re-use: 5

Figure 6.9: Screen-print of Selection sub-module of Validation Module.

retrieved outcomes in a repository for reuse. In the case shown in Figure 6.9, five (5) of the retrieved simulation outcomes are stored. As such, these five simulation outcomes can be exported to other stakeholders (e.g., DTAS) on-demand via the Output Module.

Figure 6.10 shows the user interface of the Output Module. The module exports the other module(s) to the DTAS based on user selection. In the case shown in Figure 6.10, all the modules are exported. This means that all the five torque simulation results (available in the Validation Module) and the relevant contents (available in other modules) are exported to the DTAS. Therefore, the DTAS utilizes those simulation results for monitoring a real-life machining process in real-time, as presented in the next sub-section.


Digital Twin Construction System (DTCS)

Input Module | Modeling Module | Simulation Module | Validation Module | Output Module

Select the desired module(s) from below to export:

Input Module

Modeling Module

Simulation Module

Validation Module

[✓ All of the above]

Export

Module(s) is(are) exported successfully to the Digital Twin Adaptation System (DTAS)!

Go to Digital Twin Adaptation System (DTAS) >>

Figure 6.10: Screen-print of Output Module.

6.2.2 Digital Twins' Adaptation

Figure 6.11 shows the user interface of the DTAS. The DTAS acknowledges all the five DT-generated cutting torque signals (visible one signal at a time) and uses them in monitoring. As seen in Figure 6.11, the interface displays the time series of one of the DT-generated torque signals (blue-colored plot) and the torque signal from a machine tool (black-colored plot). The user can scroll to see the other DT-generated torque signals. The system measures the errors between the signal from the machine tool and the DT-generated signals to detect an abnormality. For this, the DTAS considers a user-defined error threshold. In the case shown in Figure 6.11, an error threshold of 15% is used. As seen on the top right-hand side in Figure 6.11, the maximum

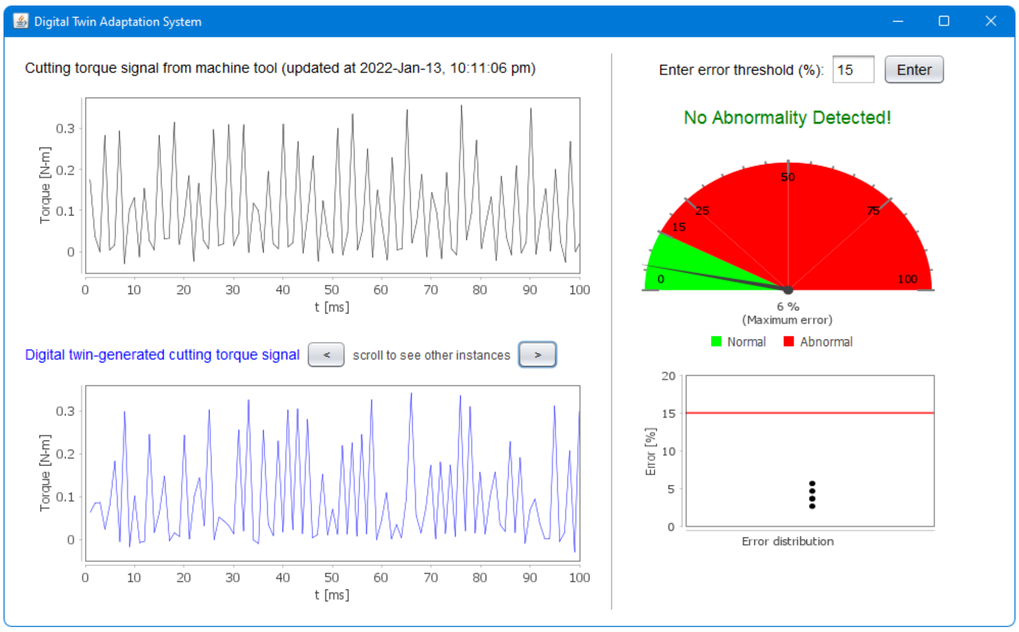

Digital Twin Adaptation System
Cutting torque signal from machine tool (updated at 2022-Jan-13, 10:11:06 pm)
Torque [N-m]
t [ms]
Digital twin-generated cutting torque signal
<
scroll to see other instances
>
Enter error threshold (%): 15
Enter
No Abnormality Detected!
6 %
(Maximum error)
🟩 Normal
🟥 Abnormal
Error [%]
Error distribution

Figure 6.11: Screen-print of DTAS in performing real-time monitoring (when no abnormality detected).

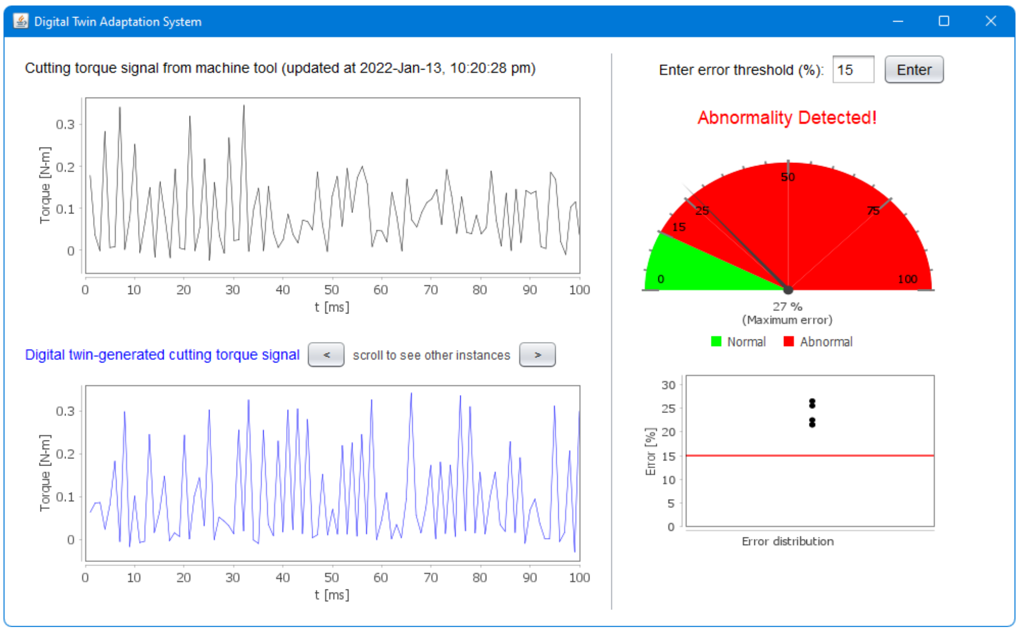

Digital Twin Adaptation System
Cutting torque signal from machine tool (updated at 2022-Jan-13, 10:20:28 pm)
Torque [N-m]
t [ms]
Digital twin-generated cutting torque signal
<
scroll to see other instances
>
Enter error threshold (%): 15
Enter
Abnormality Detected!
27 %
(Maximum error)
🟩 Normal
🟥 Abnormal
Error [%]
Error distribution

Figure 6.12: Screen-print of DTAS in performing real-time monitoring (when abnormality detected).

error (6%) lies under the threshold (green-colored region). This means that the torque signal from the machine tool shows no abnormality. Apart from the maximum error, the error distribution related to all the DT-generated signals can also be seen in the form of a scatter plot, as shown on the bottom right-hand side in Figure 6.11.

On the other hand, the system detects abnormality when any of the error values exceed the threshold. This scenario is shown in Figure 6.12. Note that the error computation follows the mathematical formulations described in Appendix E.

6.3 Summary

This chapter presents the developed sensor signal-based DT and its efficacy based on milling torque signals. Two systems called DTCS and DTAS collectively develop the DT. The systems are developed using a Java™-based platform. The systems (DTCS and DTAS) are deployed to construct sensor signal-based DTs for milling torque signals, and monitor a real-life milling process in real-time, respectively. It is seen that the DTs perform satisfactorily. Note that the systems are also deployed for other types of machining process and relevant phenomena (e.g., cutting force in end milling and surface roughness in grinding). The findings are reported in Appendix F. Nevertheless, the following chapter provides the discussions regarding the outcomes of this study in detail.

Chapter 7

Discussion

Industry 4.0-relevant futuristic manufacturing systems need knowledge-bases for performing high-level cognitive tasks such as monitoring, understanding, predicting, decision-making, and adapting. The goal is to gain better manufacturing experiences. The remarkable thing is different types of DTs (object, process, and phenomenon twins) supply the required knowledge for the abovementioned purposes. As seen in Figure 7.1, the DTs functionalize virtualization of physical manufacturing environments. As such, the DTs functionalize the abovementioned cognitive tasks, and thus, actuate maintenance, planning, effective communication, process optimization, and so forth.

Among different types of DTs, the phenomena twins are the most difficult ones to develop. This is because most manufacturing phenomena (e.g., cutting force, cutting torque, surface roughness, and alike) are complex and exhibit stochastic features. As such, modeling a phenomenon analytically for developing the twin, is a cumbersome task. One alternative way is encapsulating the dynamics underlying the phenomenon-relevant historic sensor signal datasets. Consequently, there are other relevant research questions to deal with for developing such sensor signal-based phenomena twins: (1) How to acquire the right piece of sensor signal dataset from a repository (e.g., cloud), and (2) How to accommodate delay (or time latency) while developing the DT. Also, the DT must be responsive in real-time changes. This study proposes an architecture of the sensor signal-based DTs, answering the abovementioned issues in detail (see Chapter 5). The proposed DT is also developed using a Java™-based platform. The efficacy of the DT is demonstrated for monitoring a real-life machining process (milling). For this, milling torque signals are used as example. The findings suggest that

Figure 7.1: Context of Digital Twins (DTs) in Industry 4.0.

the proposed DT is effective for monitoring the process (see Chapter 6).

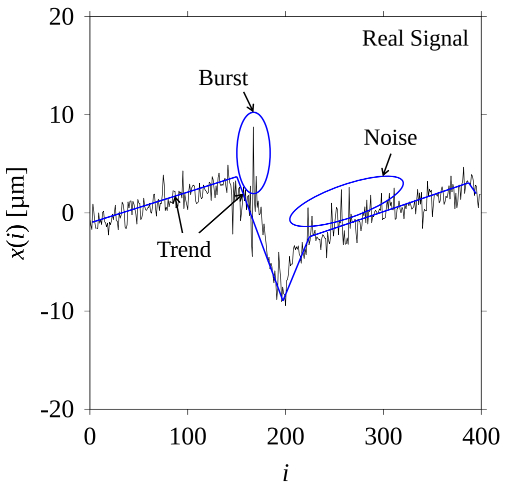
To elaborate, the DT acknowledges semantically annotated signal datasets stored in a repository in the form of XML data. The corresponding semantic annotation-based knowledge representation mechanism is also elucidated in detail (see Chapter 3). This mechanism is user-friendly and highly flexible compared to other conventional mechanisms, which are domain-specific and esoteric by nature [40, 41, 42]. The semantic annotation-based mechanism allows a user to put user-defined semantics (in the form of natural language) for creating the linked data and making it highly comprehensible to the other stakeholders (e.g., human resources and intelligent systems).

However, after acquiring the sensor signal dataset, the DT adapts delay domain-based signal processing and Markov chain-based modeling approach to accommodate delay. In this respect, detail investigation is conducted (see Chapter 4). It is seen that, when delay occurs (random or constant), it impacts sensor signals' characteristic significantly. As such, delay domain-based signal processing (in form of delay maps) is found more informative than other signal processing analyses (time and frequency domain-based analyses). Therefore, a Markov chain-based modeling approach is deployed to encapsulate the characteristic from the delay map. (The mathematical formulations underlying the Markov chain-based modeling approach are described in detail in Appendix C.) This indicates that the delay domain-Markov chain combination is promising to handle sensor signals, especially when low data acquisition rate is concerned because of delay [56, 57, 58, 77]. This is significant from the aspect of sensor data volume and energy-intensive sensor network [80] because conventional signal processing methods and machine learning techniques depend on high data volume and, in turn, occupy an appreciable amount of storage, create difficulties in storing the sensed data, consume more time to process the data, and cause energy dissipation from the sensor network, which is not desirable from the futuristic smart manufacturing context.

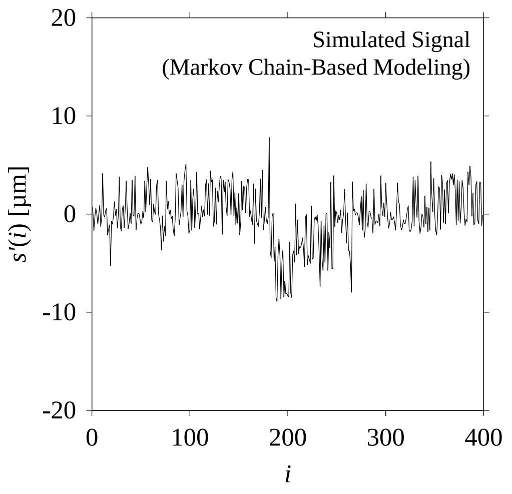
Based on the modeling outcome (here, Markov chain), the DT deploys a stochastic simulation approach (Monte Carlo simulation) (see simulation algorithm in Appendix D) for recreating (or simulating) the phenomenon-relevant sensor signals. To ensure the trustworthiness of the simulated sensor signals, the DT performs validations compared to the real signals (the historical signals used for modeling). For this, this study uses possibility distribution (fuzzy numbers) [92]. Other validation parameters (e.g., DNA-based computing [93]) can also be used, if preferred. Based on the validation outcomes, some (or all) of the simulated outcomes are used for monitoring purpose. It is seen that the DT performs satisfactorily in monitoring different machining processes (see Chapter 6 for details, can also be seen in Appendix F).

Although the proposed DT provide reliable results, there are areas to investigate and improve. Consider the Markov chain-based modeling approach adapted for developing the DT. It performs well when the sensor signals are stochastic by nature. However, it does not perform well when the sensor signals exhibit some exclusive patterns or features. For example, consider a piece of sensor signals associated with a manufacturing phenomenon called surface roughness, denoted as $x(i), i = 0, 1, ...$, as shown in Figure 7.2a. As seen in Figure 7.2a, $x(i)$ entails three stochastic features: (1) Trend, (2) Burst, and (3) Noise. As such, let us deploy Markov chain-based modeling and semantic modeling [10] for encapsulating the dynamics underlying $x(i)$, and thereby, simulating.

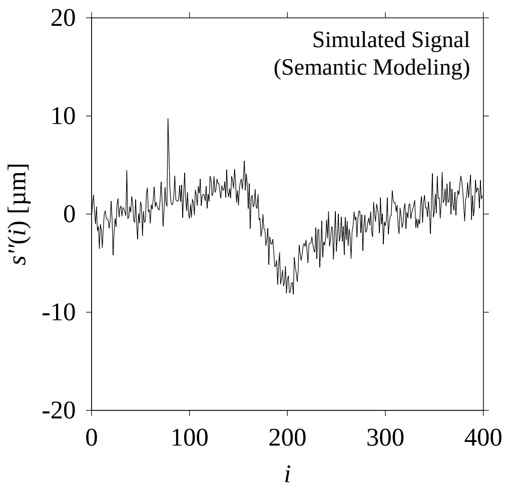
Figure 7.2b shows the simulated roughness based on Markov chain-based modeling, denoted as $s'(i), i = 0, 1, \ldots$. Figure 7.2c shows the simulated roughness based on semantic modeling, denoted as $s''(i), i = 0, 1, \ldots$. From the time series(s) of the real roughness ($x(i)$ in Figure 7.2a) and simulated roughness ($s'(i)$ and $s''(i)$ in Figures 7.2b-7.2c, respectively), it is evident



(a)



(b)



(c)

Figure 7.2: Simulation of a feature-based signal. (a) Real signal ($x(i)$), (b) Simulated signal ($s'(i)$) using Markov chain-based modeling, (c) Simulated signal ($s''(i)$) using semantic modeling.

that Markov chain-based modeling does not model the roughness satisfactorily. On the other hand, another modeling method called semantic modeling performs better.

For better understanding, let us validate or compare the simulation outcomes with the real ones using possibility distribution [92] and DNA-based computation [93, 97]. In this respect, Figures 7.3a-7.3b show the validation outcomes for comparing $x(i)$ and $s'(i)$, using possibility distributions and

DNA-based computations, respectively. It is seen that the possibility distributions (see Figure 7.3a) do not resemble with each other. Similarly, the frequency distributions underlying the DNA-based computations (see Figure 7.3b) do not resemble each other.

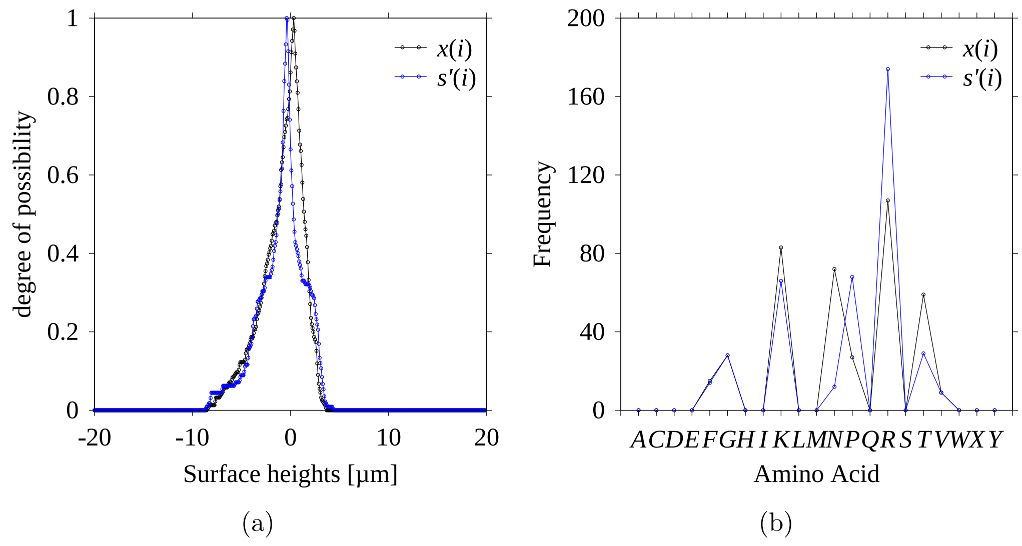


Figure 7.3: Comparison between $x(i)$ and $s'(i)$. (a) possibility distribution, (b) DNA-based computation.

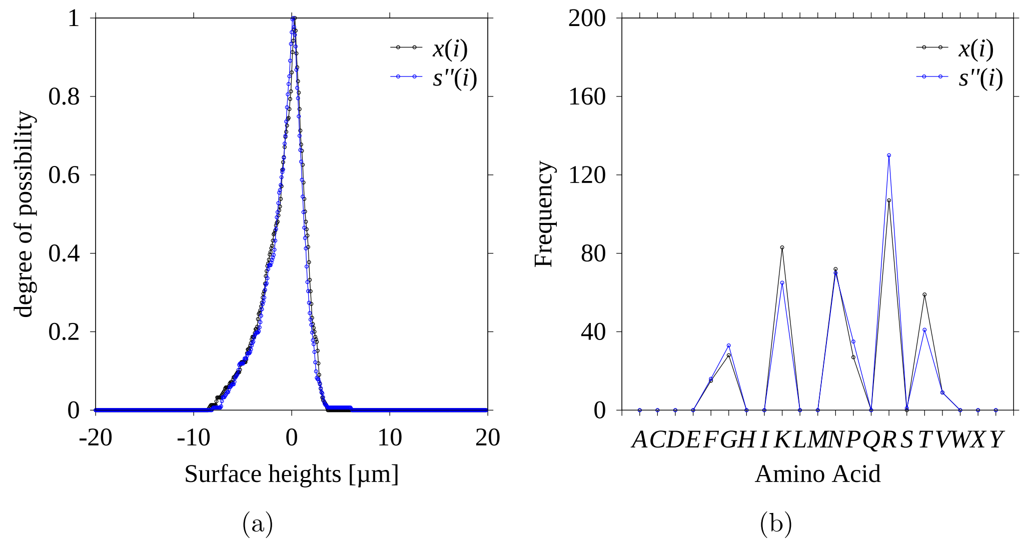


Figure 7.4: Comparison between $x(i)$ and $s''(i)$. (a) possibility distribution, (b) DNA-based computation.

On the other hand, Figures 7.4a-7.4b show the validation outcomes for comparing $x(i)$ and $s''(i)$, using possibility distributions and DNA-based computations, respectively. It is seen that the possibility distributions (see Figure 7.4a) highly resemble with each other. Similarly, the frequency distributions underlying the DNA-based computations (see Figure 7.4b) highly resemble each other.

The abovementioned example illustrates that Markov chain-based modeling is inadequate for encapsulating sensor signals' characteristics when exclusive features (e.g., trend, burst, and noise) are present in signal datasets. To solve this issue, other kind modeling approach such as semantic modeling might be considered. As such, several modeling approaches must be considered for developing a universal DT so that the DT can handle different types of sensor signals more effectively. Further research can be conducted in this direction.

Having said that, based on the findings of this study, some other future research directions can also be contemplated, as described below.

1. The DTs of machining phenomena provide meaningful insight into a machining process in real-time (monitoring, understanding, and predicting) and functionalize decision-making. A relative research question is that how to adapt the made decisions in real-time. This means that what should be the pragmatic approach for setting the right course of actions given that there is anomaly in the process. Further research can be conducted in this direction for making autonomous DT-driven machine tools from the context of Industry 4.0.

2. Since real-time connectivity is a prime spring in Industry 4.0, different sensors perform continuously for sensing and transmitting tremendous amount of signal data. This evolves two issues: (1) need of high data storage capacity and time-consuming computational arrangements [77], and (2) energy-intensive sensor networks [80, 84, 85]. These issues must be taken care of from the context of fast communication and longevity of the sensors. Since the findings of this study suggests that the delay domain-based signal processing is capable of handling less amount of data while unfolding signals' nature, further research can be directed to investigate the abovementioned issues in detail based on the delay domain.

3. Since content sharing and reusing is significant in Industry 4.0, Big Data (BD) of manufacturing must be present in the CPS. The BD supply the relevant content to other stakeholders. Nevertheless, there

is a lack of a steadfast procedure to construct the BD. Further research can be conducted in this direction based on the semantic annotation-based representation mechanism presented in this study.

Chapter 8

Conclusion

The concluding remarks of this study are described as follows:

- The recent literature review on digital twins (DTs) relevant to Industry 4.0 or smart manufacturing suggests that all three types of DTs (object, process, and phenomenon twins) must populate the Cyber-Physical Systems (CPS) to functionalize the futuristic manufacturing systems. This means that the DTs are the contents on which the systems associated with futuristic manufacturing systems (e.g., CPS) act to perform high-level cognitive tasks such as monitoring, understanding, predicting, decision-making, and adapting.

- Although several works have been conducted on the development of DTs, the works mostly concentrate on the development of object and process twins. Significant studies are yet to be conducted on phenomena twins. Phenomena twins assist futuristic machine tools for performing monitoring and troubleshooting tasks autonomously. However, developing a phenomenon twin is a cumbersome task since most manufacturing phenomena (e.g., cutting torque, cutting force, surface finish, and alike) are stochastically non-linear. This study addresses this issue by developing phenomena twins from historical sensor signal datasets associated to the relevant phenomena.

- Since the twin is developed from the extracted knowledge underlying historical sensor signal datasets, the issues related to acquiring the right piece of sensor signal dataset stored in a repository (e.g., cloud) and time latency (or delay) while transmitting signals among different systems are two essential research questions. As such, the sensor signal-based DTs must be able to deal with the abovementioned issues. This study investigates and answers the abovementioned questions in detail, followed by proposing an architecture of the sensor signal-based DTs. The DT is developed by using a Java™-based platform and its (DT) efficacy in monitoring a real-life machining experiment is also elucidated in this study.

- In particular, two systems called Digital Twin Construction System (DTCS) and Digital Twin Adaption System (DTAS) are proposed. DTCS constructs the DTs, while DTAS adapts the constructed DTs for monitoring. The functional retirements and modular architectures underlying the systems (DTCS and DTAS) are elucidated in Chapter 5 in detail.

- The DTCS consists of five modules: Input, Modeling, Simulation, Validation, and Output Modules. These modules collectively construct the DT. These modules' functional requirements are Data management, Ontology, Machine learning, Modeling, Simulation, Validation, Real-time response, and Semantic web compatibility (see Chapter 5).

- The Input Module can search semantically annotated datasets stored in a remote data storage facility and select the appropriate signal datasets. This module handles both human-comprehensible contents (concept maps) and machine-readable contents (XML format). This arrangement ensures effective data mining because signal datasets accompany other relevant information (e.g., machining process, machine tools, machining conditions, sensors, and alike), making it meaningful to different stakeholders. Note that, in reality, most data representation mechanisms are domain-dependent and esoteric by nature. For this, this study proposes a flexible and user-friendly semantically annotated representation mechanism. The proposed mechanism is elucidated in Chapter 3 in detail. The Input Module is developed based on the proposed mechanism.

- The Modeling Module machine learns the dynamics underlying the Input Module-supplied sensor datasets. In this case, numerous machine learning methods can be used. In this article, a Markov chain-based machine learning method. (The method is described in Appendix C in detail). This method works well when the signal datasets are sampled from a delay map, acknowledging the underlying delay (random or deterministic). Note that, delay (or time latency) is unavoidable and affects sensor signals' characteristic significantly (see Chapter 4).

- The Modeling Module supplies the machine-learned model of the signal datasets to the Simulation Module. This module simulates the signal using the modeling information received. In this case, a discrete event stochastic simulation process is recommended. (The simulation algorithm is elucidated in Appendix D).

- Whether or not the Simulation Module has simulated the signal datasets faithfully can be tested in the Validation Module. Thus, the Valuation Module is equipped with some quantitative measures. In this article, a possibility distribution-based approach is used for validation. (The possibility distribution-based validation process is described in Appendix E).

- The Output Module transfers the contents generated in other modules for reuse (say, monitoring). The user can select the contents of the Input, Modeling, Simulation, and Validation Modules for transferring them to a data storage facility in the XML format. One of the default locations is DTAS. Thus, the Output Module is the connecting point between the DTCS and DTAS.

- Though the DTAS can adapt all the contents generated in the DTCS, it uses only the simulated signal datasets tested positive by the Validation Module while monitoring a process in real-time. It receives real-time signals from a machine tool for monitoring purposes. Therefore, it is equipped with a user interface showing the monitoring results. Any update in the DTCS will change the contents. Therefore, DTAS also updates itself, acknowledging the changes made in DTCS in real-time. These real-time updating capabilities make these two systems highly coupled.

- In this study, milling torque signals are used as an example for testing the efficacy of the abovementioned systems (DTCS and DTAS), as demonstrated in Chapter 6. Other signals (cutting force and surface roughness) also show promising results as reported in Appendix F. Thus, the DTCS and DTAS can be used to ensure machine tools' ability to perform their monitoring and troubleshooting tasks autonomously. This way, the finding of this study contributes to the advancement of Industry 4.0 or smart manufacturing.

- Nevertheless, a DT can improve the cyber-physical integration of industrial IoT. In most cases, the twin must machine-learn the required knowledge from the historical sensor signal datasets and seamlessly interact with the real-time sensor signals. While doing so, the DTs must handle the semantically annotated datasets stored in clouds and accommodate the data transmission delay. The presented DTCS and DTAS fulfill these requirements of DTs. Therefore, these can be used as general tools for information integration in smart manufacturing.

Appendix A

Signal Analyses (Mild Steel)

A.1 Different Sampling Windows

Consider the cutting force signal obtained while machining Mild Steel (JIS: S15CK), i.e., F_{W_2}. Figure A.1 shows its time series. As seen in Figure A.1, the sample size of F_{W_2} is denoted as L_{W_2}, where $L_{W_2} = 7501$. F_{W_2} is sampled four times using four different sampling windows (light blue colored regions in the time series of F_{W_2} shown in Figure A.1). This results in four new signals denoted as $F_{W_2S_1}, ..., F_{W_2S_4}$. The corresponding sample sizes are denoted as $L_{W_2S_1}, ..., L_{W_2S_4}$, where $L_{W_2S_1} = 5001, L_{W_2S_2} = 2501, L_{W_2S_3} = 1501$, and $L_{W_2S_4} = 501$.

For the sake of analysis, the signals F_{W_2} and $F_{W_2S_1}, ..., F_{W_2S_4}$ are transferred to the frequency domain (using FFT) and delay domain (using $d = 1$, as described in Section 4.3). As such, Figure A.2 shows the time series, FFT, and delay map for F_{W_2}. Figures A.3-A.6 show the time series, FFT, and delay map for $F_{W_2S_1}, ..., F_{W_2S_4}$, respectively.

As seen in Figure A.2b, the prominent frequencies underlying F_{W_2} are 0 Hz, 780 Hz, 1560 Hz, 2333.333 Hz, 3113.333 Hz, 3893.333 Hz, and 4673.333 Hz. As seen in Figure A.3b, the prominent frequencies underlying $F_{W_2S_1}$ are 0 Hz, 780 Hz, 1560 Hz, 2340 Hz, 3110 Hz, 3890 Hz, and 4670 Hz. As seen in Figure A.4b, the prominent frequencies underlying $F_{W_2S_2}$ are 0 Hz, 780 Hz, 1560 Hz, 2340 Hz, and 3120 Hz. As seen in Figure A.5b, the prominent frequencies underlying $F_{W_2S_3}$ are 0 Hz, 766.6667 Hz, 1566.6667 Hz, 2333.333 Hz, and 3100 Hz. As seen in Figure A.6b, the prominent frequencies underly-

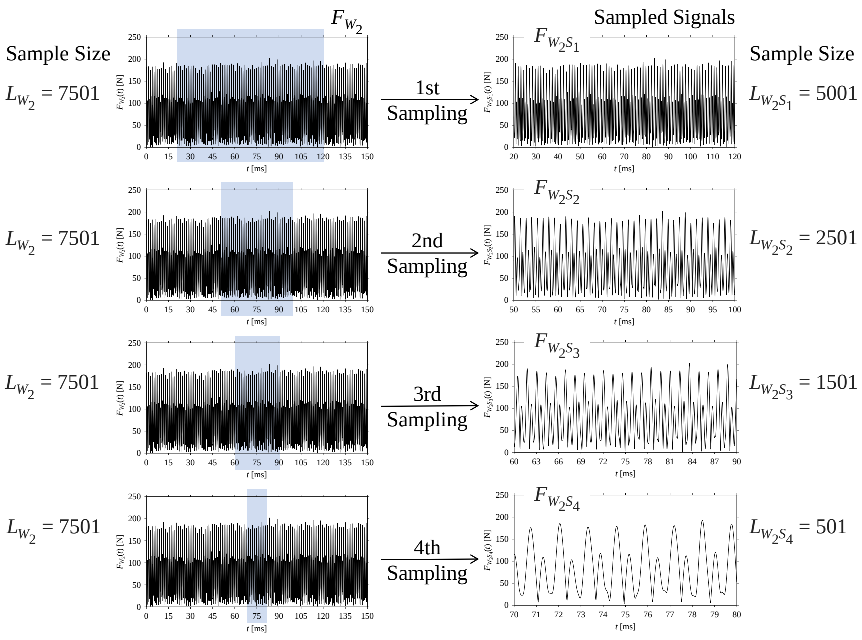


Figure A.1: Sampling the cutting force signal for machining Mild Steel.

ing $F_{W_2S_4}$ are 0 Hz, 800 Hz, 1600 Hz, 2300 Hz, and 3100 Hz. This means that the frequency information varies with the sampling window. In addition, the FFT pattern gets affected when the sampling window is shorter (see Figure A.6b compared to Figure A.2b).

On the other hand, the delay maps shown in Figure A.2c and Figures A.3c-A.6c exhibit similar characteristics under different sampling windows. In particular, the returns of points from one to another are identical. This

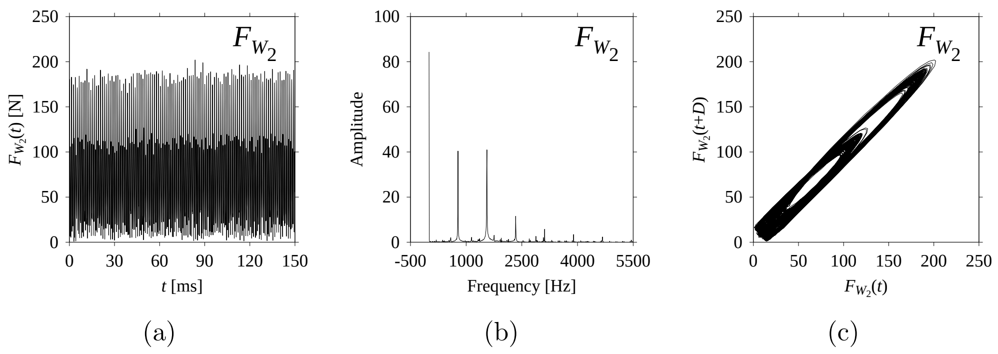
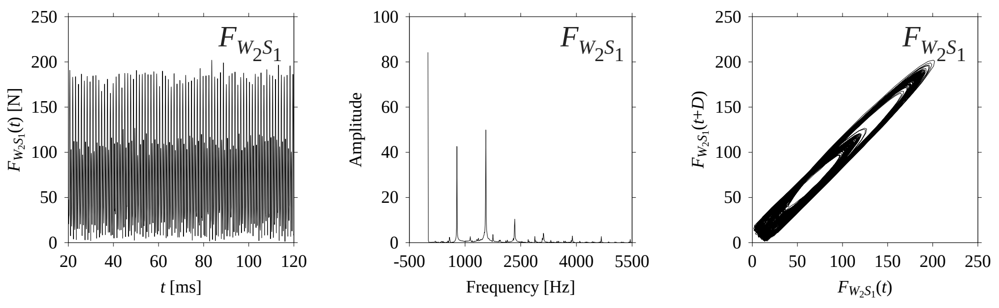
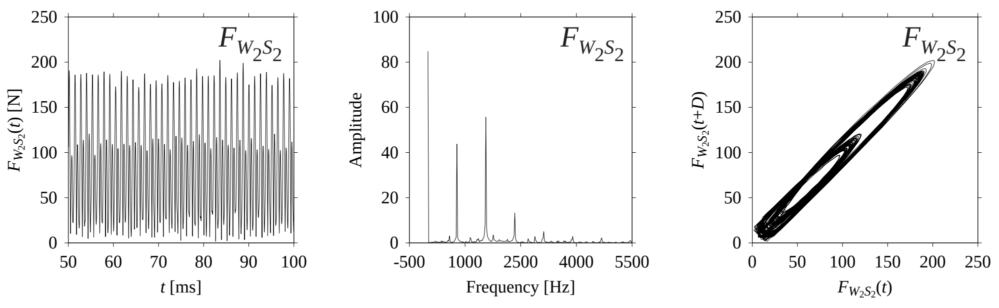


Figure A.2: F_{W_2}. (a) Time series, (b) Fast Fourier Transformation (FFT), (c) Delay map ($d = 1$).



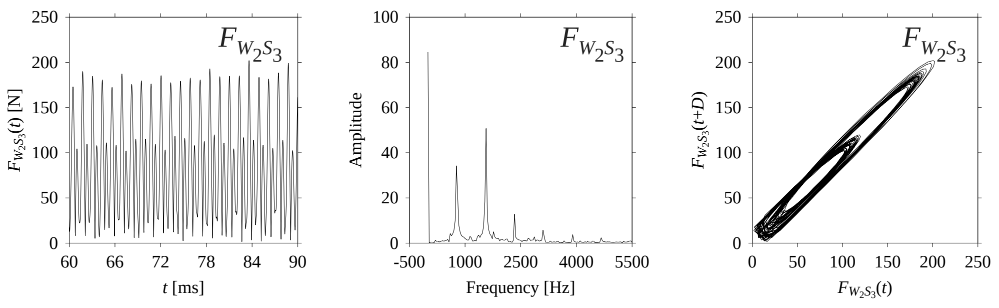
(a) (b) (c)

Figure A.3: $F_{W_2S_1}$. (a) Time series, (b) Fast Fourier Transformation (FFT), (c) Delay map ($d = 1$).



(a) (b) (c)

Figure A.4: $F_{W_2S_2}$. (a) Time series, (b) Fast Fourier Transformation (FFT), (c) Delay map ($d = 1$).



(a) (b) (c)

Figure A.5: $F_{W_2S_3}$. (a) Time series, (b) Fast Fourier Transformation (FFT), (c) Delay map ($d = 1$).

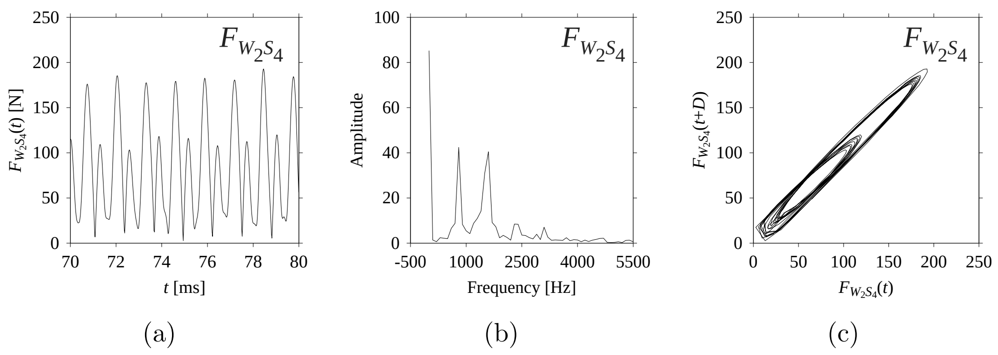


Figure A.6: $F_{W_2S_4}$. (a) Time series, (b) Fast Fourier Transformation (FFT), (c) Delay map ($d = 1$).

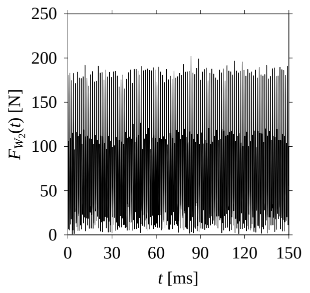
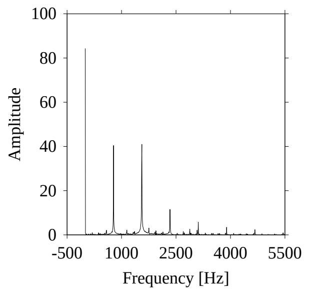
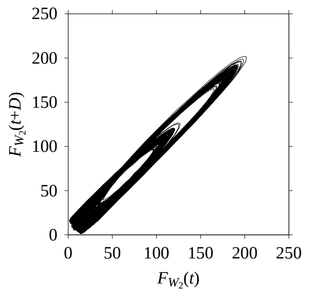
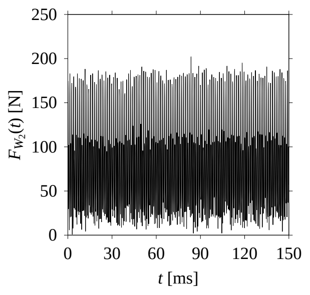
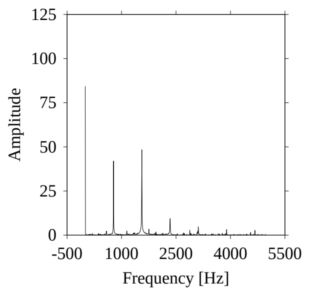
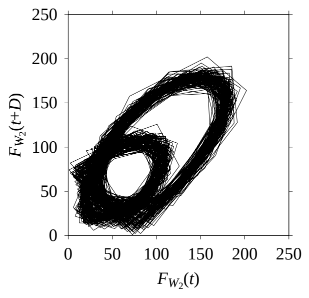
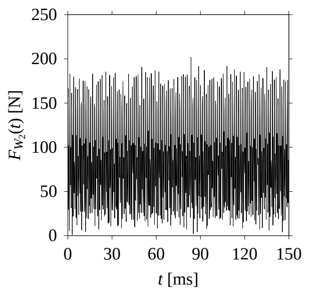
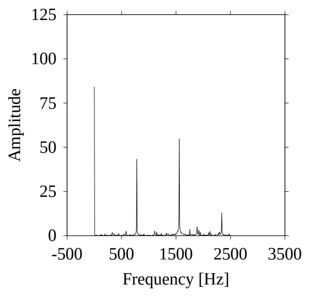
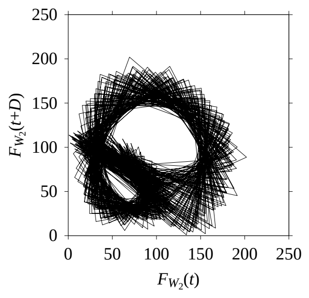
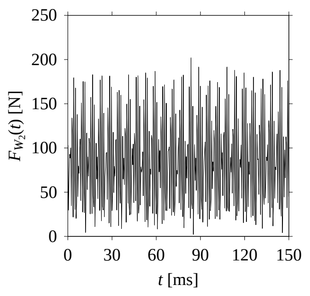
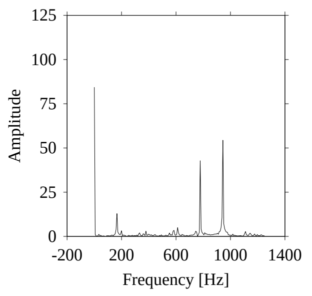
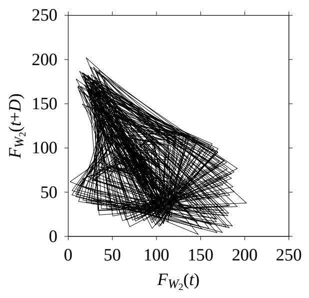
means that the underlying nature of the F_{W_2} and $F_{W_2S_1}, ..., F_{W_2S_4}$ are the same, regardless of the sample size. In addition, the density of points underlying the delay maps provides meaningful insight into the sample size of the signal. For example, the density of the delay map shown in Figure A.6c is lighter compared to that of in Figure A.2c, which mean the corresponding signal $F_{W_2S_4}$ (see Figure A.6a) undergoes a low sampling window compared to the F_{W_2} (see Figure A.2a).

A.2 Different Delays

Again, consider the cutting force signal obtained while machining Mild Steel (JIS: S15CK), i.e., $F_{W_2}(t), t = 0, \Delta t, 2\Delta t, ..., m \times \Delta t$ (can also be seen in Figure 4.8). As mentioned before, here Δt is the sampling period of 0.02 ms. To incorporate time latency or delay, Δt is increased using the delay parameter d (non-zero integer), such as $d \times \Delta t$. For example, for $d = 1$, the sampling period remains $1 \times 0.02 = 0.02$ ms; for $d = 2$, the sampling period becomes $2 \times 0.02 = 0.04$ ms; and alike. As mentioned in Section 4.3, $d \times \Delta t$ is simplified using D, where $D = d \times \Delta t$. As such, a set of time series is generated using D where $d = 1, 2, ...$.

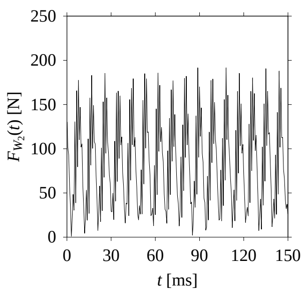
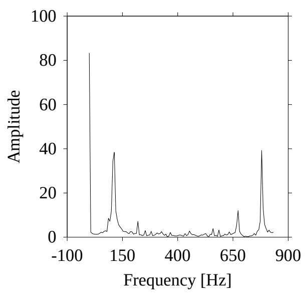
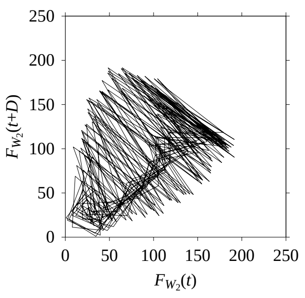
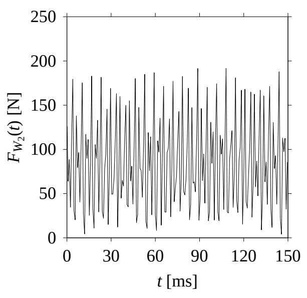
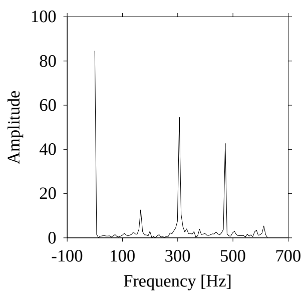
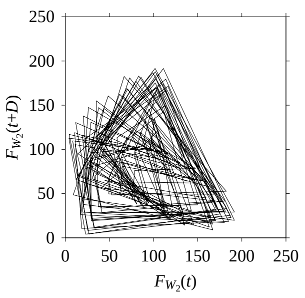
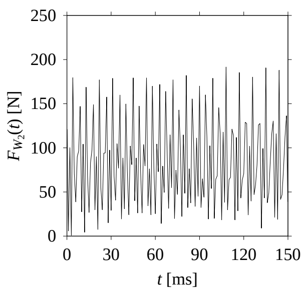
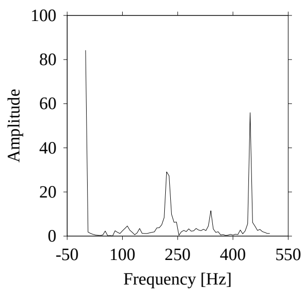
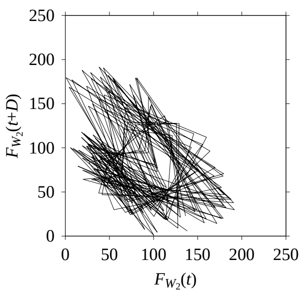
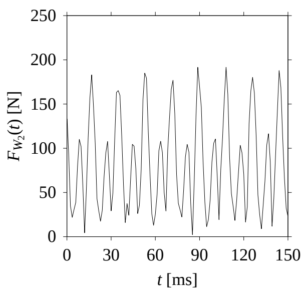
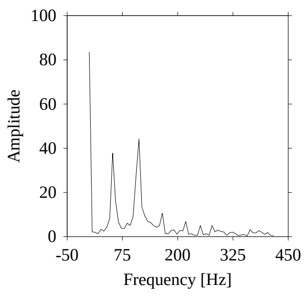
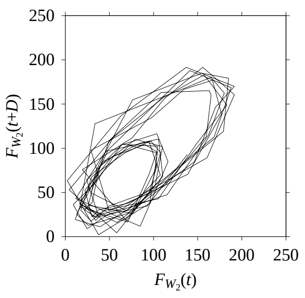
The goal here is to understand the dynamics underlying the cutting force signals due to varying delay. For this, the time series datasets are transferred to the frequency domains (using FFT) and corresponding delay domains. Table A.1 shows the outcomes for some of the delays, i.e., $d = 1, 5, 10, 20, 30, 40, 50,$ and 60.

Table A.1: Cutting force signal (F_{W_2}) in the form of time series, FFT, and delay map under varying delay (d).

Time series(s)	**FFT(s)**	**Delay maps**
	$d = 1$	
		
	$d = 5$	
		
	$d = 10$	
		
	$d = 20$	
		

(continued on next page)

Table A.1: Cutting force signal (F_{W_2}) in the form of time series, FFT, and delay map under varying delay (d) (continued from previous page).

Time series(s)	**FFT(s)**	**Delay maps**
	$d = 30$	
		
	$d = 40$	
		
	$d = 50$	
		
	$d = 60$	
		

As seen in Table A.1, when d increases, the frequency information underlying the F_{W_2} gets affected. The prominent frequencies (see the FFT diagram for $d = 1$) are gradually lost due to aliasing [78] when $d > 5$ (see the FFT diagrams for $d = 10, 20, 30, 40, 50$, and 60).

On the other hand, consider the delay maps consisting of the points $\big(F_{W_2}(t), F_{W_2}(t + D)\big)$ shown in Table A.1. When $d = 1$, the delay map exhibits a very systematic pattern. When d increases, the delay maps get more and more scattered (see the delay maps for $d = 5, 10, 20, 30, 40$, and 50). This means that the signal gets more and more chaotic due to the presence of delay. However, when $d = 60$, the corresponding delay map is somewhat systematic and similar to that of $d = 5$. This means that the underlying natures of these two signals are similar, regardless of the difference in the sampling rate. It is worth mentioning that the corresponding FFTs (see the FFTs for $d = 5$ and for $d = 60$ shown in Table A.1) are different and do not preserve the nature of the source signal. As such, delay domain-based representation is more informative for understanding the underlying nature of F_{W_2} under a low data acquisition rate due to time latency or delay.

Appendix B

Signal Analyses (Ductile Cast Iron)

B.1 Different Sampling Windows

Consider the cutting force signal obtained while machining Ductile Cast Iron (JIS: FCD), i.e., F_{W_3}. Figure B.1 shows its time series. As seen in Figure B.1, the sample size of F_{W_3} is denoted as L_{W_3}, where $L_{W_3} = 7501$. F_{W_3} is sampled four times using four different sampling windows (light blue colored regions in the time series of F_{W_3} shown in Figure B.1). This results in four new signals denoted as $F_{W_3S_1}, ..., F_{W_3S_4}$. The corresponding sample sizes are denoted as $L_{W_3S_1}, ..., L_{W_3S_4}$, where $L_{W_3S_1} = 5001, L_{W_3S_2} = 2501, L_{W_3S_3} = 1501$, and $L_{W_3S_4} = 501$.

For the sake of analysis, the signals F_{W_3} and $F_{W_3S_1}, ..., F_{W_3S_4}$ are transferred to the frequency domain (using FFT) and delay domain (using $d = 1$, as described in Section 4.3). As such, Figure B.2 shows the time series, FFT, and delay map for F_{W_3}. Figures B.3-B.6 show the time series, FFT, and delay map for $F_{W_3S_1}, ..., F_{W_3S_4}$, respectively.

As seen in Figure B.2b, the prominent frequencies underlying F_{W_3} are 0 Hz, 780 Hz, 1560 Hz, 2333.333 Hz, 3113.333 Hz, 3893.333 Hz, and 4673.333 Hz. As seen in Figure B.3b, the prominent frequencies underlying $F_{W_3S_1}$ are 0 Hz, 780 Hz, 1560 Hz, 2340 Hz, 3110 Hz, 3890 Hz, and 4670 Hz. As seen in Figure B.4b, the prominent frequencies underlying $F_{W_3S_2}$ are 0 Hz, 780 Hz, 1560 Hz, 2340 Hz, and 3120 Hz. As seen in Figure B.5b, the prominent frequencies underlying $F_{W_3S_3}$ are 0 Hz, 766.6667 Hz, 1566.6667 Hz, 2333.333 Hz, and 3100 Hz. As seen in Figure B.6b, the prominent frequencies underly-

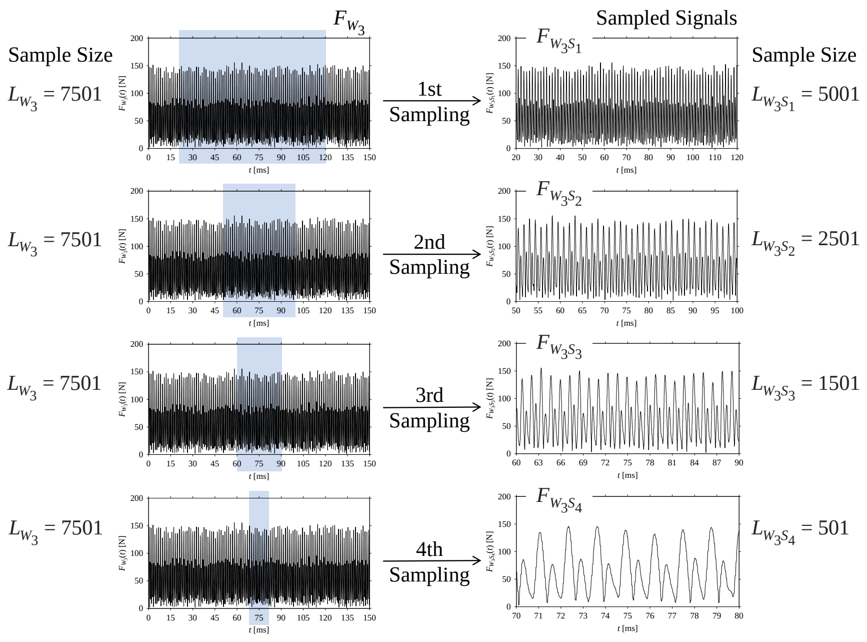


Figure B.1: Sampling the cutting force signal for machining Ductile Cast Iron.

ing $F_{W_3S_4}$ are 0 Hz, 800 Hz, 1600 Hz, 2400 Hz, and 3100 Hz. This means that the frequency information varies with the sampling window. In addition, the FFT pattern gets affected when the sampling window is shorter (see Figure B.6b compared to Figure B.2b).

On the other hand, the delay maps shown in Figure B.2c and Figures B.3c-B.6c exhibit similar characteristics under different sampling windows. In particular, the returns of points from one to another are identical. This means

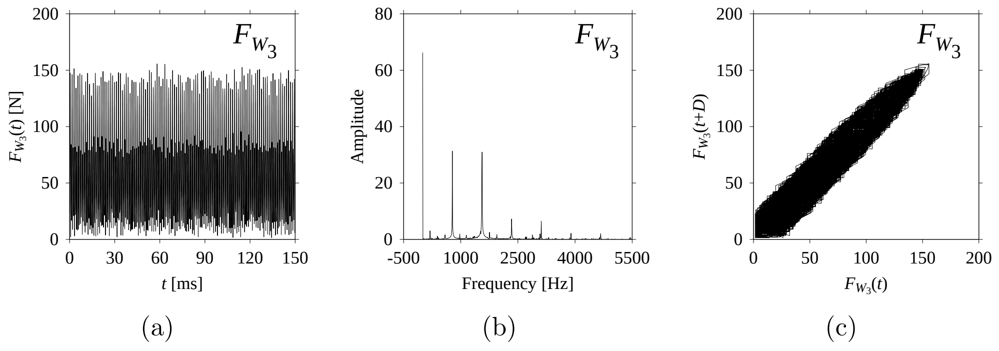


Figure B.2: F_{W_3}. (a) Time series, (b) Fast Fourier Transformation (FFT), (c) Delay map ($d = 1$).

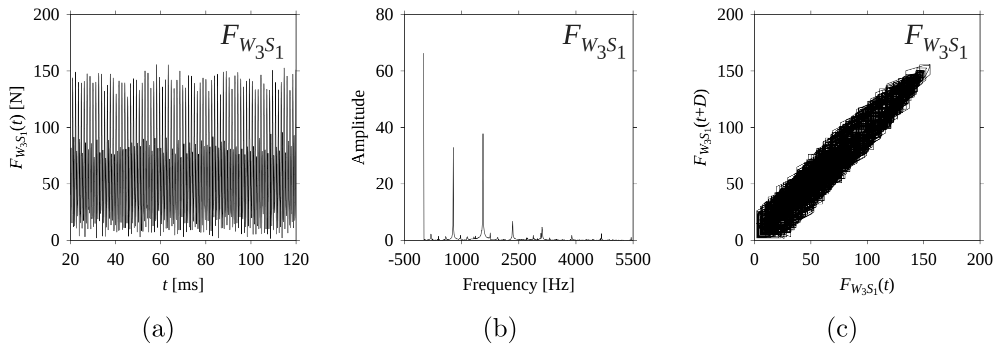


Figure B.3: $F_{W_3S_1}$. (a) Time series, (b) Fast Fourier Transformation (FFT), (c) Delay map ($d = 1$).

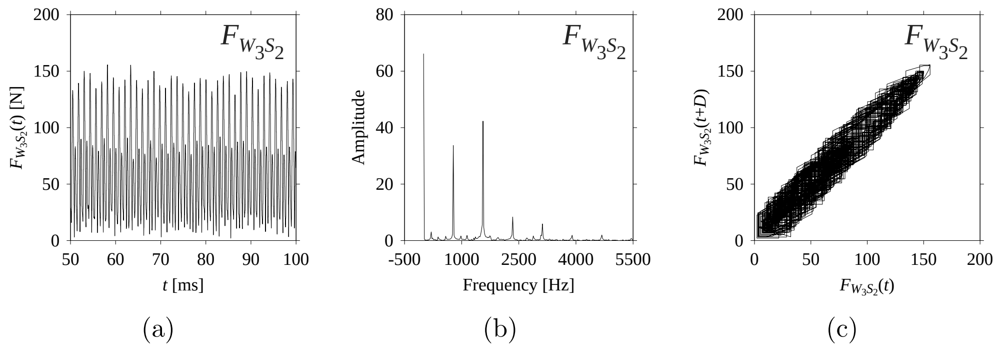


Figure B.4: $F_{W_3S_2}$. (a) Time series, (b) Fast Fourier Transformation (FFT), (c) Delay map ($d = 1$).

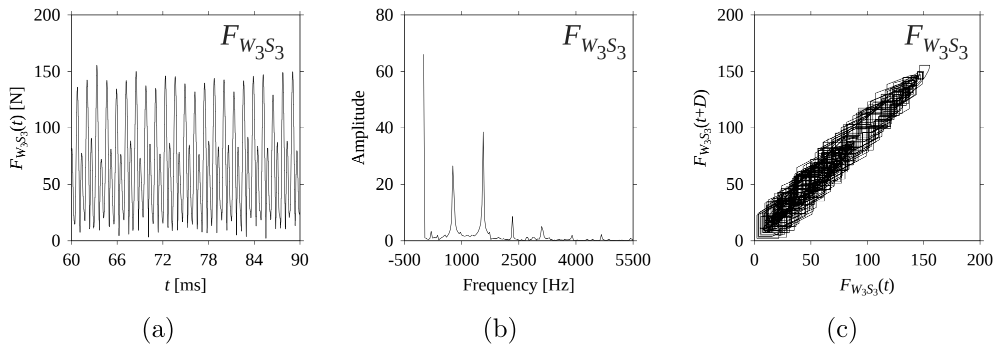


Figure B.5: $F_{W_3S_3}$. (a) Time series, (b) Fast Fourier Transformation (FFT), (c) Delay map ($d = 1$).

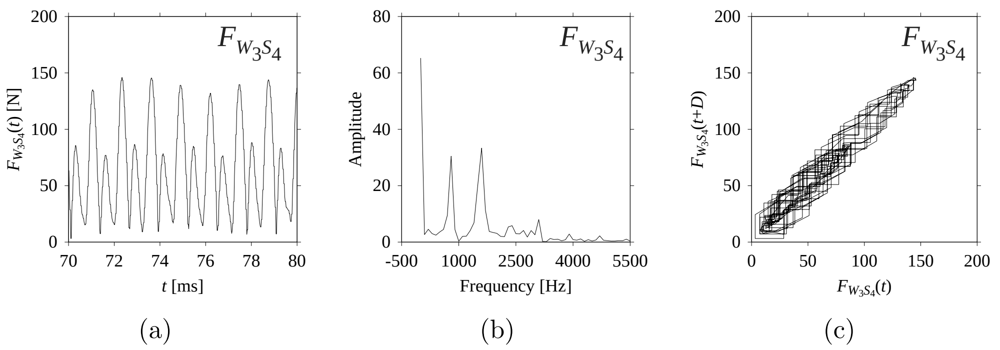


Figure B.6: $F_{W_3S_4}$. (a) Time series, (b) Fast Fourier Transformation (FFT), (c) Delay map ($d = 1$).

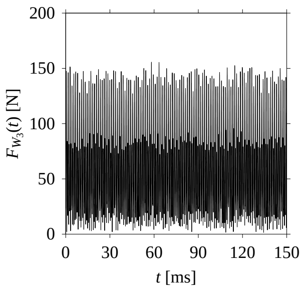
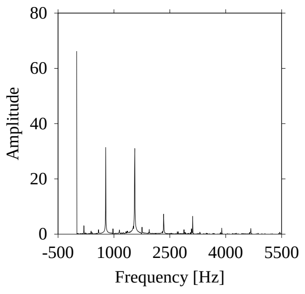
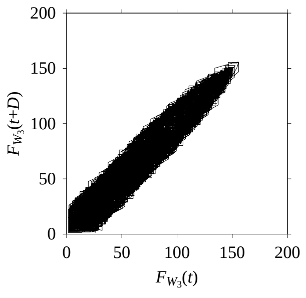
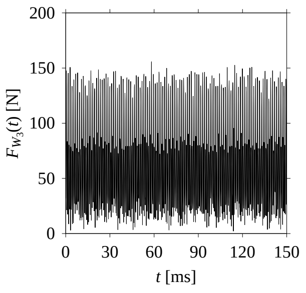
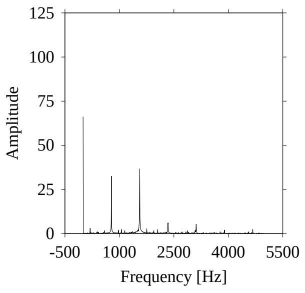
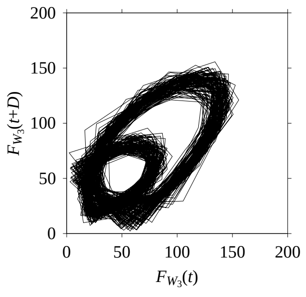
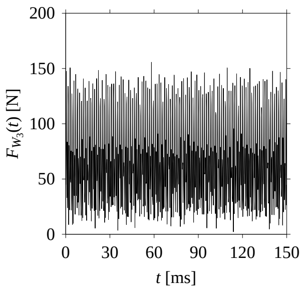
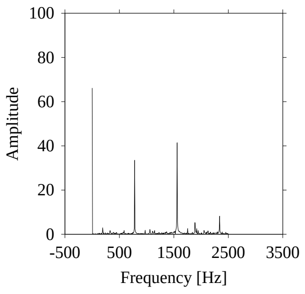
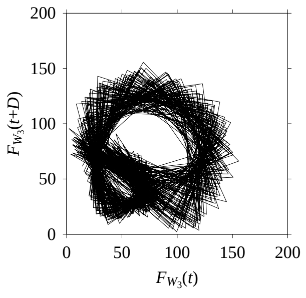
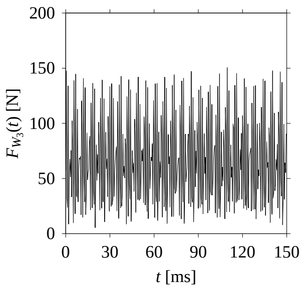
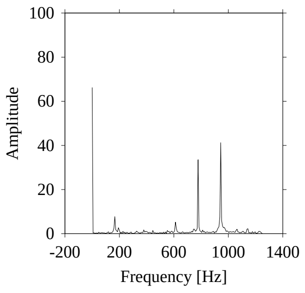
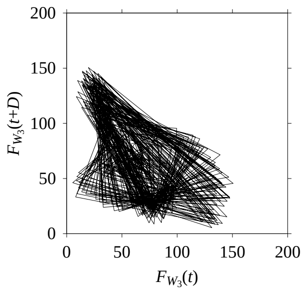
that the underlying nature of the F_{W_3} and $F_{W_3S_1}, ..., F_{W_3S_4}$ are the same, regardless of the sample size. In addition, the density of points underlying the delay maps provides meaningful insight into the sample size of the signal. For example, the density of the delay map shown in Figure B.6c is lighter compared to that of in Figure B.2c, which mean the corresponding signal $F_{W_3S_4}$ (see Figure B.6a) undergoes a low sampling window compared to the F_{W_3} (see Figure B.2a).

B.2 Different Delays

Again, consider the cutting force signal obtained while machining Ductile Cast Iron (JIS: FCD), i.e., $F_{W_3}(t), t = 0, \Delta t, 2\Delta t, ..., m \times \Delta t$ (can also be seen in Figure 4.8). As mentioned before, here Δt is the sampling period of 0.02 ms. To incorporate time latency or delay, Δt is increased using the delay parameter d (non-zero integer), such as $d \times \Delta t$. For example, for $d = 1$, the sampling period remains $1 \times 0.02 = 0.02$ ms; for $d = 2$, the sampling period becomes $2 \times 0.02 = 0.04$ ms; and alike. As mentioned in Section 4.3, $d \times \Delta t$ is simplified using D, where $D = d \times \Delta t$. As such, a set of time series is generated using D where $d = 1, 2, ...$

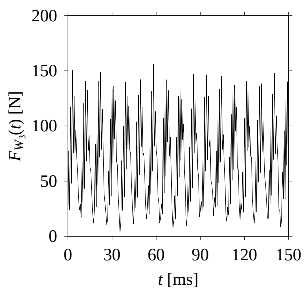
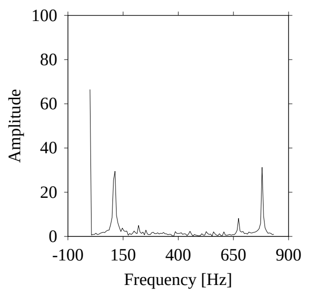
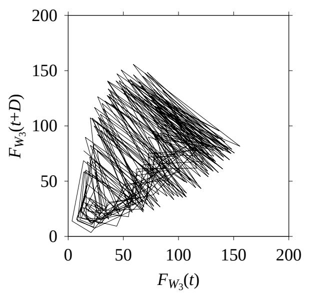
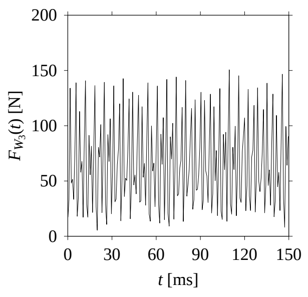
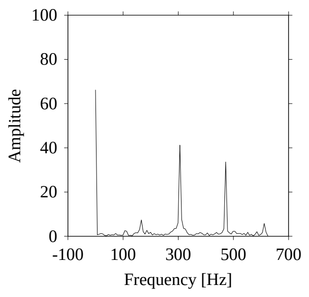
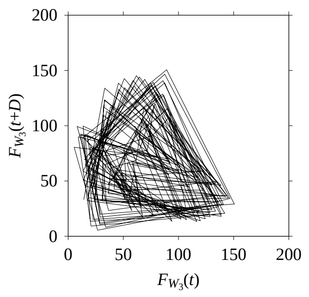
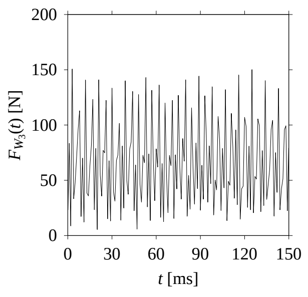
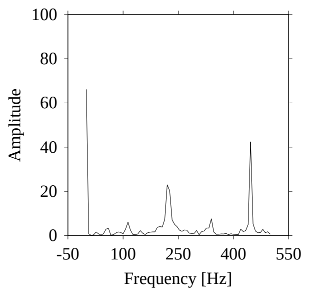
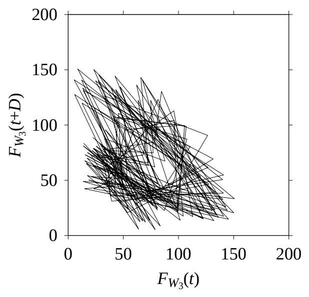
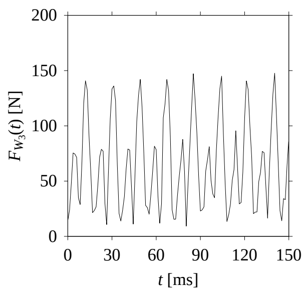
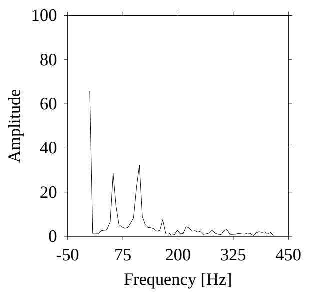
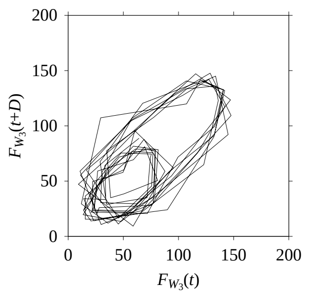
The goal here is to understand the dynamics underlying the cutting force signals due to varying delay. For this, the time series datasets are transferred to the frequency domains (using FFT) and corresponding delay domains. Table B.1 shows the outcomes for some of the delays, i.e., $d = 1, 5, 10, 20, 30, 40, 50$, and 60.

Table B.1: Cutting force signal (F_{W_3}) in the form of time series, FFT, and delay map under varying delay (d).

Time series(s)	**FFT(s)**	**Delay maps**
	$d = 1$	
		
	$d = 5$	
		
	$d = 10$	
		
	$d = 20$	
		

(continued on next page)

Table B.1: Cutting force signal (F_{W_3}) in the form of time series, FFT, and delay map under varying delay (d) (continued from previous page).

Time series(s)	**FFT(s)**	**Delay maps**
	$d = 30$	
		
	$d = 40$	
		
	$d = 50$	
		
	$d = 60$	
		

As seen in Table B.1, when d increases, the frequency information underlying the F_{W_3} gets affected. The prominent frequencies (see the FFT diagram for $d = 1$) are gradually lost due to aliasing [78] when $d > 5$ (see the FFT diagrams for $d = 10, 20, 30, 40, 50$, and 60).

On the other hand, consider the delay maps consisting of the points $\big(F_{W_3}(t), F_{W_3}(t + D)\big)$ shown in Table B.1. When $d = 1$, the delay map exhibits a very systematic pattern. When d increases, the delay maps get more and more scattered (see the delay maps for $d = 5, 10, 20, 30, 40$, and 50). This means that the signal gets more and more chaotic due to the presence of delay. However, when $d = 60$, the corresponding delay map is somewhat systematic and similar to that of $d = 5$. This means that the underlying natures of these two signals are similar, regardless of the difference in the sampling rate. It is worth mentioning that the corresponding FFTs (see the FFTs for $d = 5$ and for $d = 60$ shown in Table B.1) are different and do not preserve the nature of the source signal. As such, delay domain-based representation is more informative for understanding the underlying nature of F_{W_3} under a low data acquisition rate due to time latency or delay.

Appendix C

Markov Chain-Based Modeling

The computing power of hidden Markov models has been playing an important role in studying the complex phenomena underlying design and manufacturing. For example, Liao et al. [98] have developed a heuristic optimization algorithm using hidden Markov model coupled with simulated annealing for condition monitoring of machineries. Li et al. [99] have developed a data-driven bearing fault identification methodology using an improved hidden Markov model and self-organizing map. Mba et al. [100] have developed a hidden Markov model-based methodology for condition monitoring of gearbox. Zhang et al. [101] have developed a methodology for predicting the residual life of the rolling machine elements using hidden Markov model. Xie et al. [102] have described a hidden Markov model-based methodology for recognizing the machining states ensuring safe operations. Bhat et al. [103] have developed a hidden Markov model-based tool condition monitoring methodology ensuring the economical usages of cutting tools. Liao et al. [104] have developed a grinding wheel condition monitoring methodology where a hidden Markov model-based clustering approach was used to recognize the patterns found in the acoustic emission signals. Cai et al. [105] have developed a methodology using a hidden Markov model to identify the energy efficiency states while removing materials by milling ensuring eco-friendly machining operation. Kumar et al. [106] have integrated hidden Markov model with polynomial regression for predicting the useful life of cutting tools.

This study adapts hidden Markov modeling for encapsulating knowledge underlying phenomena-relevant sensor signals from its (sensor signal) delay map. As such, the fundamental idea of modeling is schematically illustrated in Figure C.1. As seen in Figure C.1, a hidden Markov model consists of three segments. The first segment is called a Markov chain. The second segment is called a time series of latent variables. The last segment is called a time series of observations. The descriptions of the three segments are as follows.

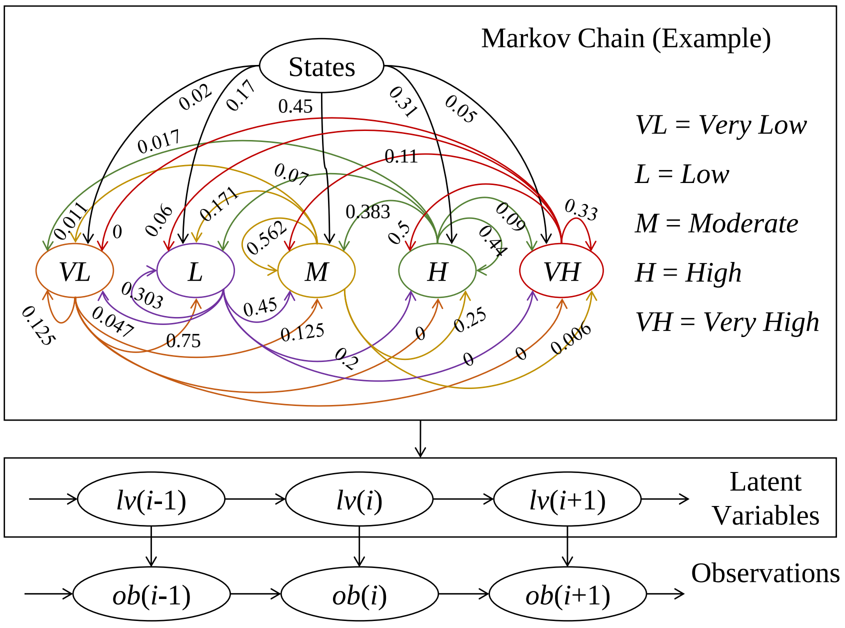


Figure C.1: Concept of hidden Markov modeling.

As seen in Figure C.1, a Markov chain forms a network showing both a set of discrete states (which later appear as latent variables) and their transitions. Each state and its possible transitions are associated with the probabilities, which are also the parts of the Markov chain. For example, the Markov chain shown in Figure C.1 consists of five discrete states having the probabilities $p(\textit{Very Low}) = 0.02$, $p(\textit{Low}) = 0.17$, $p(\textit{Moderate}) = 0.45$, $p(\textit{High}) = 0.31$, and $p(\textit{Very High}) = 0.05$. The transitions to states denoted as *Very Low*, *Low*, *Moderate*, *High*, and *Very High* from the state *Very Low* exhibit the following transition probabilities: $p(\textit{Very Low} \mid \textit{Very Low}) = 0.125$, $p(\textit{Very Low} \mid \textit{Low}) = 0.75$, $p(\textit{Very Low} \mid \textit{Moderate}) = 0.125$, $p(\textit{Very Low} \mid \textit{High}) = 0$, and $p(\textit{Very Low} \mid \textit{Very High}) = 0$. The summation of the state probabilities or the transition probabilities from a given state to other possible states is unit. There are other issues related to the hidden Markov chain (e.g., the order of the Markov chain). The case shown in Figure C.1 corresponds to the first order Markov chain because the probability of the previous state determines the current state. This study adopts this strategy.

See the articles in [107] and [108] for more details regarding the order of a Markov chain and other relevant issues.

Regarding the segment called time series of the latent variables (see Figure C.1), the following remarks can be made. The latent variables are the results of a stochastic simulation process (i.e., Monte Carlo simulation of discrete states associated with the Markov chain). Thus, for the case shown in Figure C.1, the latent variables belong to the set of discrete states, that is, $lv(0), ..., lv(i-1), lv(i), lv(i+1), ... \in \{$*Very Low*, *Low*, *Moderate*, *High*, *Very High*$\}$. These are called latent because one cannot observe (or not interested in observing) these variables, that is, they are simulated for the sake of computation. The probabilities (in reality, relative frequencies) of the latent variables must be consistent with the transition probabilities associated with the Markov chain. This means that for the case shown in Figure C.1, when $lv(i) =$ *Very Low*, then the probability of $lv(i+1) =$ *Very Low* is equal to 0.125, $lv(i+1) =$ *Low* is equal to 0.75, $lv(i+1) =$ *Moderate* is equal to 0.125, $lv(i+1) =$ *High* is equal to 0, and $lv(i+1) =$ *Very High* is equal to 0.

Lastly, regarding the segment called the time series of the observations (see Figure C.1), the following remarks can be made. The observations $ob(0), ..., ob(i-1), ob(i), ob(i+1), ...$ are simulated using the information of the corresponding latent states, $lv(0), ..., lv(i-1), lv(i), lv(i+1), ...$. The observations are the outputs of the hidden Markov model, which is used to solve a problem.

When one constructs a hidden Markov model to encapsulate the dynamics underlying a given time series, the scenario shown in Figure C.2 evolves. The scenario entails the following steps, namely, (1) data acquisition (defining the time series, delay map, and latent variables), (2) Markov chain construction, (3) simulation of latent variables and observations, and (5) a comparison between the simulated observations and given time series. For the sake of better understanding, a set of mathematical entities and their relationships are required, which are described as follows.

Let the manifestation of a phenomenon be a piece of *time series* denoted as $X = \{x(t) \in \Re \,|\, t = 0, \Delta t, 2\Delta t, ..., m \times \Delta t\}$ where Δt is known as *delay* or *interval*. The parameter t underlies a temporal or spatial entity, that is, a point of time or a distance. If preferred, the time series can be represented by indexing its elements using a *pointer*. In this case, $x(t)$ is replaced by $x(i)\big(=x(t)\big)$ where $t = i \times \Delta t$ and i is the pointer, which is a positive integer including 0, that is, $i = 0, 1, ...$

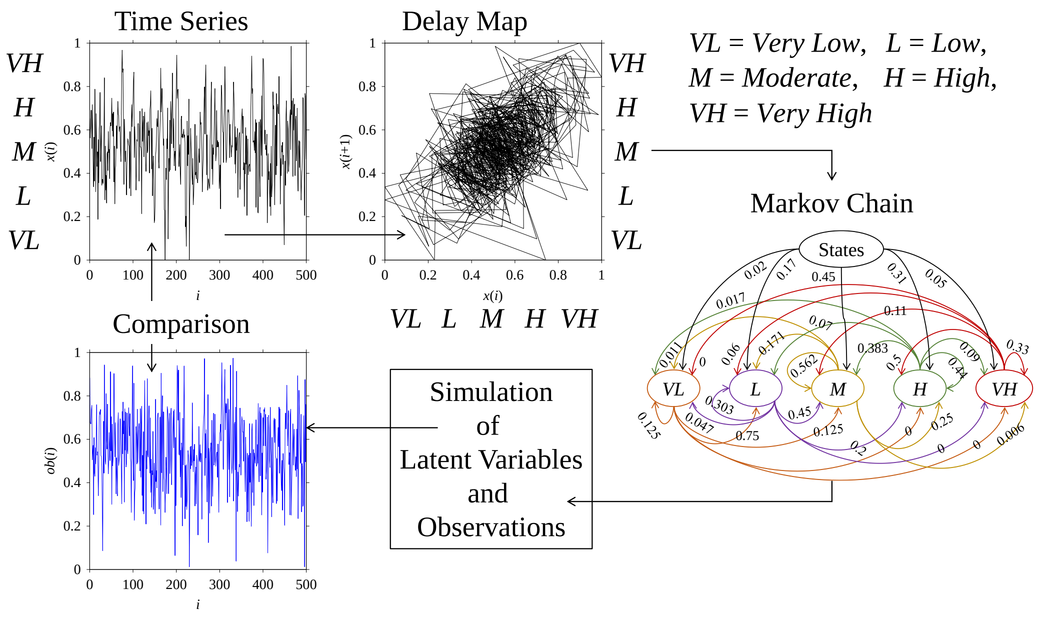


Figure C.2: Integration among Markov chain, time series, and delay map.

Let $U = [u_{\min}, u_{\max}] \in \Re$, defined as the *universe of discourse*, be an interval so that $X \supseteq U$. Let $x_{\min}$ be the minimum value of X, that is, $x_{\min} = \min\big(x(t) \,|\, \forall t \in \{0, \Delta t, 2\Delta t, ...\}\big)$. Let $x_{\max}$ be the maximum value of X, that is, $x_{\max} = \max\big(x(t) \,|\, \forall t \in \{0, \Delta t, 2\Delta t, ...\}\big)$. If $U = [u_{\min}, u_{\max}] = [x_{\min}, x_{\max}]$, then it is defined as the *exact interval case.*

Let $U_1, ..., U_n$ be n number of *mutually exclusive intervals* that partition U so that the following proposition denoted as P is true.

$$P = \big((U_1 \cup ... \cup U_n = U) \wedge (U_j < U_{j+1} \,|\, \forall j \in \{1, ...n-1\}) \wedge \\ (U_k \cap U_l = \emptyset \,|\, \forall k \in \{1, ..., n\}, \forall l \in \{1, ..., n\} - \{j\})\big) \tag{C.1}$$

The partitions are the states (or latent states or variables) of X resulting in a *state vector* $SV = (U_1, ..., U_n)$. One can define the states in many ways making the proposition P true. One of the straightforward ways is to consider a *state interval* $\Delta u = (u_{\max} - u_{\min})/n$ and use it for defining the states in the following manner: $U_1 = [u_{\min}, u_{\min} + \Delta u), U_2 = [u_{\min} + \Delta u, u_{\min} + 2\Delta u), ..., U_n = \big[u_{\min} + (n-1) \times \Delta u, u_{\min} + n \times \Delta u\big]$.

Let $p(U_j) \in [0, 1]$ be the probability of j-th state U_j in SV with respect to $X, j = 1, ..., n$. Thus, the following relationships hold.

$$\Phi\big(x(i), U_j\big) = \begin{cases} 1, & x(i) \in U_j \\ 0, & \text{otherwise} \end{cases} \tag{C.2}$$

$$p(U_j) = \frac{\sum_{i=0}^{m} \Phi\big(x(i), U_j\big)}{m+1} \tag{C.3}$$

Thus, $p(U_j)$ is defined as the *state probability* of the j-th state U_j. As such, the summation of all state probabilities is equal to unit, that is,

$$p(U_1) + ... + p(U_n) = 1 \tag{C.4}$$

For the sake of computation (e.g., simulation), the state probabilities can be used to calculate the *cumulative state probability.* The cumulative state probability of the j-th state U_j in SV is defined as follows.

$$pc(U_j) = p(U_1) + ... + p(U_j) \tag{C.5}$$

As such $pc(U_n)$ is 1. The cumulative state probability can be used to calculate the *state probability interval* denoted as $pc_{in}(U_j)$. The state probability interval of the first state U_1 in SV is as follows.

$$pc_{in}(U_1) = \big[0, pc(U_1)\big) \tag{C.6}$$

The state probability interval of the last state U_n in SV is as follows.

$$pc_{in}(U_n) = \big[pc(U_{n-1}), pc(U_n)\big] \tag{C.7}$$

The state probability intervals of the states other than U_1 and U_n in SV are as follows.

$$pc_{in}(U_j) = \big[pc(U_{j-1}), pc(U_j)\big) \qquad \forall j \in \{2, ..., n-1\} \tag{C.8}$$

Let the set of tuples $\big\{\big(x(i), x(i+1)\big) \,|\, i = 1, 2, ...\big\}$ or $\big\{\big(x(t), x(t+i\Delta t)\big) \,|\, i = 1, 2, ...\big\}$ be the *return* or *delay map* of the time series X. Therefore, each point $\big(x(i), x(i+1)\big), \exists i \in \{1, 2, ...\}$ of the delay map exhibits a transition. As a result, a *transition probability* denoted as $tp(U_o \,|\, U_j)\big(\forall o, \forall j \in \{1, ..., n\}\big)$

means the likelihood of the transition of X to the state U_o from the state U_j. Thus, the following relationships hold.

$$\Phi\Big(\big(x(i), x(i+1)\big), \big(U_o, U_j\big)\Big) = \begin{cases} 1, & \Big(\big(x(i) \in U_j\big) \land \big(x(i+1) \in U_o\big)\Big) \\ 0, & \text{otherwise} \end{cases} \tag{C.9}$$

$$tp(U_o \,|\, U_j) = \frac{\sum_{i=0}^{m} \Phi\Big(\big(x(i), x(i+1)\big), \big(U_o, U_j\big)\Big)}{\sum_{i=0}^{m} \Phi\big(x(i), U_j\big)} \tag{C.10}$$

As such, the summation of all transition probabilities from a given state is equal to unit, that is, $tp(U_1 \,|\, U_j) + ... + tp(U_n \,|\, U_j) = 1, \exists j \in \{1, ..., n\}$. This yields a *transition probability matrix* as follows.

$$M_{tp} = \begin{bmatrix} tp(U_1 \,|\, U_1) & \dots & tp(U_n \,|\, U_1) \\ \vdots & \ddots & \vdots \\ tp(U_1 \,|\, U_n) & \dots & tp(U_n \,|\, U_n) \end{bmatrix} \tag{C.11}$$

For the sake of computation (e.g., simulation), the transition probabilities can be used to calculate the *cumulative transition probability.* The cumulative transition probability is defined as follows.

$$tpc(U_o \,|\, U_j) = tp(U_1 \,|\, U_j) + ... + tp(U_o \,|\, U_j) \tag{C.12}$$

As such, $tpc(U_n \,|\, U_j) = 1$. This yields the *cumulative transition probability matrix*, as follows.

$$M_{tpc} = \begin{bmatrix} tpc(U_1 \,|\, U_1) & \dots & tpc(U_n \,|\, U_1) \\ \vdots & \ddots & \vdots \\ tpc(U_1 \,|\, U_n) & \dots & tpc(U_n \,|\, U_n) \end{bmatrix} \tag{C.13}$$

The cumulative transition probability can be used to calculate the *transition probability interval* denoted as $tpc_{in}(U_o \,|\, U_j)$. The transition probability interval of the first states U_1 to any state U_j is as follows.

$$tpc_{in}(U_1 \,|\, U_j) = \big[0, tpc(U_1 \,|\, U_j)\big) \tag{C.14}$$

The transition probability interval of the last state U_n to any state U_j is as follows.

$$tpc_{in}(U_n \,|\, U_j) = \big[tpc(U_{n-1} \,|\, U_j), tpc(U_n \,|\, U_j)\big] \tag{C.15}$$

The transition probability intervals of any states to U_j (other than U_1 or U_n) are as follows:

$$tpc_{in}(U_o \,|\, U_j) = \big[tpc(U_{o-1} \,|, U_j), tpc(U_o \,|\, U_j)\big) \qquad \forall o \in \{2, ..., n-1\} \tag{C.16}$$

Nevertheless, the abovementioned mathematical formulations encapsulate the dynamics in the form of a Markov chain, and thus, create transition probability intervals so that simulations can be carried out following a simulation algorithm. The simulation algorithm is presented in Appendix D.

Appendix D

Simulation Algorithm

Using the mathematical entities and their relationships described in Appendix C, one can formulate a Monte Carlo simulation process to simulate the latent variables $lv(.) \in (U_1, ..., U_n)$ and the observations $ob(.) \in \Re$. Note that the simulated observations will be denoted as $xs(.)$, not as $ob(.)$, to make the notation consistent with the time series, $x(.)$. The simulation process consists of the nine (9) steps as shown in Table D.1.

The first three steps mentioned in Table D.1, i.e., Steps $1, ..., 3$, are related to the steps of the Markov chain formulation as shown in Figure C.2 in Appendix C. The other steps, Steps $4, ..., 9$, are related to the steps of the simulation of latent variables and observations as shown in Figure C.2 in Appendix C. Note that in Step 8, a function $f(U_{(.)})$ is introduced. It produces a value based on the state $U_{(.)}$. When any other information is not available, $f(U_{(.)})$ randomly generates a real number from a normally distributed variable denoted as $rn_{(.)}\left(\mu(U_{(.)}), \sigma(U_{(.)})\right)$. Here, $\mu(U_{(.)})$ and $\sigma(U_{(.)})$ denote the mean and standard deviation, respectively. As such, the following formulation holds.

$$f(U_{(.)}) = rn_{(.)}\left(\mu(U_{(.)}), \sigma(U_{(.)})\right) \tag{D.1}$$

The formulation of $f(U_{(.)})$ defined in Equation D.1 is used in this study. However, other formulations of $f(U_{(.)})$ can be used, as preferred.

It is worth mentioning that even though a simulated value $xs(i)$ belongs to one of the states say U_j, the next state $xs(i+1)$ may not belong to the same state. This means that the simulation process continues similar to a dynamical system [86].

Table D.1: Simulation algorithm.

Step 1	Define the time series $X = \{x(i) \in \Re \mid i = 0, ..., m\}$, universe of discourse U, state vector $SV = (U_1, ..., U_n)$, and the number of iteration $N \in \aleph^{+0}$.
Step 2	Calculate the state probabilities $p(U_j)$, cumulative state probabilities $pc(U_j)$, and state probability intervals $pc_{in}(U_j)$ so that $\forall j \in \{1, ..., n\}$.
Step 3	Calculate transition probabilities $tp(U_o \mid U_j)$, cumulative transition probabilities $tpc(U_o \mid U_j)$, and transition probability intervals $tpc_{in}(U_o \mid U_j)$ so that $\forall o \in \{1, ..., n\}$ and $\forall j \in \{1, ..., n\}$.
Step 4	Initialize the simulation process by assigning $xs(i = 0) \in U$ randomly.

Step 5	Determine the state of $xs(i) \rightarrow S(i)$ as follows:
	    For $j = 1, ..., n$
	        If $xs(i) \in U_j$ Then $S(i) = U_j$
	    End For

Step 6	Generate a random number $r_i \in [0, 1]$.

Step 7	Determine the transition state $S(i+1)$ as follows:
	    For $o = 1, ..., n$
	        If $r_i \in tpc_{in}(U_o \mid U_j)$ Then $S(i+1) = U_o$
	    End For

Step 8	Simulate $xs(i+1)$ as follow: $xs(i+1) = \max\Big(\min\big(u_{\max}, f(U_o)\big), u_{\min}\Big)$
Step 9	Redefine the pointer $i = i + 1$. If $i \leq N - 1$ Then Go To Step 5. Otherwise Stop.

Appendix E

Quantifying Possibility Distribution

Let $\{x(t) \in \Re \,|\, t = 0, \Delta t, 2\Delta t, ..., n \times \Delta t\}$ be a real sensor signal collected from a manufacturing process. Here, Δt is the delay. As such, the delay map of the real signal consists of the ordered pairs $\big\{\big(x(t), x(t + \Delta t)\big) \,|\, t = 0, \Delta t, 2\Delta t, ..., (n-1) \times \Delta t\big\}$. Let $\{s(t) \in \Re \,|\, t = 0, \Delta t, 2\Delta t, ..., n \times \Delta t\}$ be a DT-generated signal (i.e., simulated signal). As such, the delay map of the simulated signal consists of the ordered pairs $\big\{\big(s(t), s(t + \Delta t)\big) \,|\, t = 0, \Delta t, 2\Delta t, ..., (n-1) \times \Delta t\big\}$. From a given return map, a possibility distribution (fuzzy number) can be induced following the mathematical formulations described in [92]. Thus, the point cloud of the real signal induces a possibility distribution denoted as $\text{Poss}(x) \in [0, 1]$ and a triangular fuzzy number (TFN), as schematically illustrated in Figure E.1a. The core of the TFN is one of the cores of $\text{Poss}(x)$. The support of the TFN is the support of $\text{Poss}(x)$. Similarly, the point cloud of the simulated signal induces a possibility distribution denoted as $\text{Poss}(s) \in [0, 1]$ and a triangular fuzzy number (TFN), as schematically illustrated in Figure E.1b.

Now, a set of four parameters denoted as $P_i \,|\, i = 1, ..., 4$, can be considered to quantify a possibility distribution and a TFN. Here, P_1 is the area under the possibility distribution, and P_2 is the average possibility, as schematically illustrated in Figure E.2a. On the other hand, P_3 and P_4 are the core and the range calculated from the support of TFN, respectively, as schematically illustrated in Figure E.2b. In Figures E.2a-E.2b, $\exists X \in \{x, s\}$ and $\exists Y \in \{\text{Poss}(x), \text{Poss}(s)\}$.

Let $\{(X_j, Y_j) \,|\, j = 1, ..., N+1\}$ be the points on the possibility distribution. Therefore, P_1 and P_2 can be calculated using the following expressions.

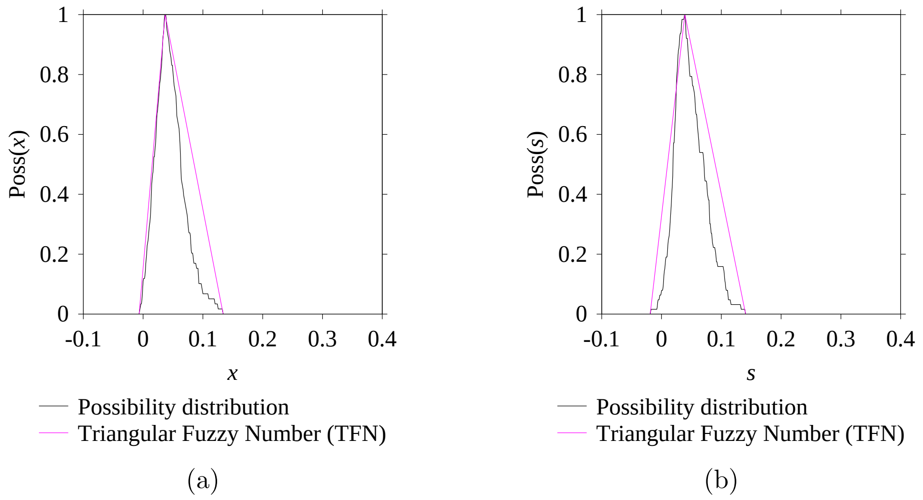


Figure E.1: Possibility distributions and triangular fuzzy numbers. (a) Real signal ($x(t)$), (b) Simulated signal ($s(t)$).

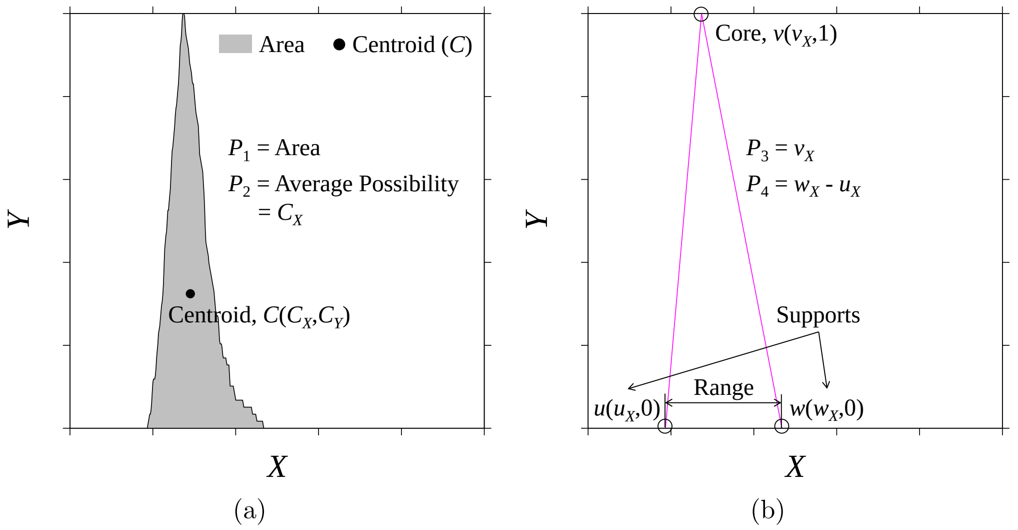


Figure E.2: Quantifying parameters ($P_1, ..., P_4$). (a) P_1 and P_2 for possibility distribution, (b) P_3 and P_4 for triangular fuzzy number.

$$P_1 = \sum_{j=1}^{N} A_j \tag{E.1}$$

$$P_2 = \frac{\sum_{j=1}^{N} A_j \left(\dfrac{X_{j+1} + X_j}{2}\right)}{\sum_{j=1}^{N} A_j} \tag{E.2}$$

Here, the expression of A_j is as follows.

$$A_j = \frac{|(X_{j+1} - X_j)|(Y_{j+1} + Y_j)}{2} \tag{E.3}$$

Let (u_X, w_X) and v_X be the core and support of TFN, respectively. Therefore, P_3 and P_4 can be calculated using the following expressions.

$$P_3 = v_X \tag{E.4}$$

$$P_4 = w_X - u_X \tag{E.5}$$

Let $P_{i(\text{real})}$ denote the values of $P_i \,|\, i = 1, ..., 4$, for the possibility distribution and TFN corresponding to the real signal. Let $P_{i(\text{simulated})}$ denote the values of $P_i \,|\, i = 1, ..., 4$, for the possibility distribution and TFN corresponding to the simulated signal. This results in a measure of error denoted as e_i. This error can be calculated as follows.

$$e_i = \left| \frac{P_{i(\text{simulated})} - P_{i(\text{real})}}{P_{i(\text{real})}} \right| \tag{E.6}$$

The effect of e_i can be aggregated by calculating the average error denoted as Error (E). Thus, the following expression holds.

$$E = \frac{\sum_{i=1}^{4} e_i}{4} \tag{E.7}$$

The simulated signal exhibiting the minimal E matches the real signal as closely as possible. Therefore, E can be used in monitoring a manufacturing process. This is implemented in the presented DT in this thesis.

Appendix F

Monitoring Results for Other Processes

The presented systems, Digital Twin Construction System (DTCS) and Digital Twin Adaptation System (DTAS) (see Chapter 5 and Chapter 6), are also deployed for performing real-time in-process monitoring of a milling process and a grinding process.

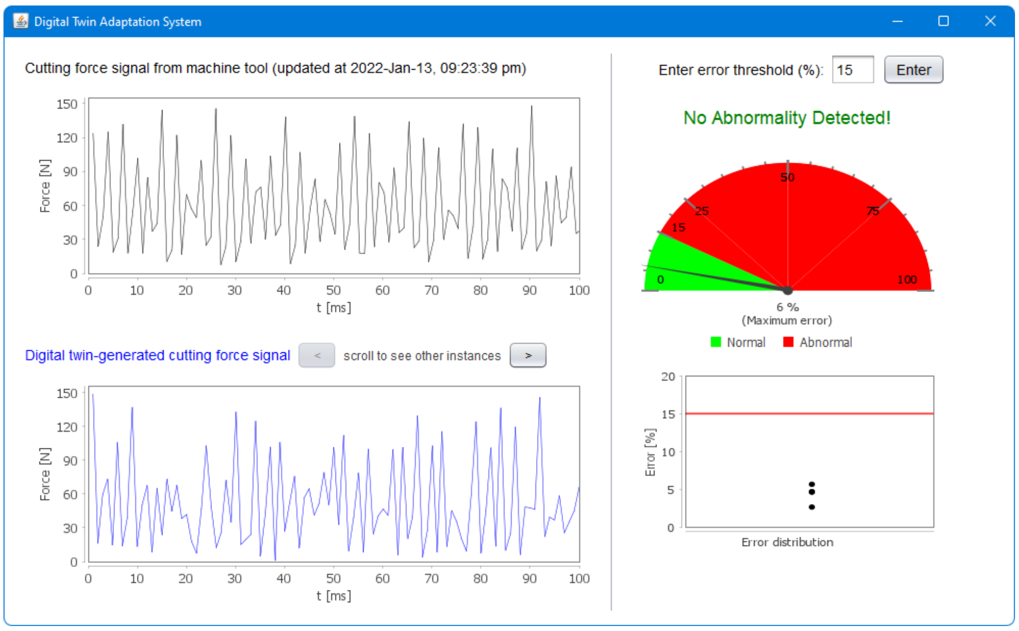

Digital Twin Adaptation System
Cutting force signal from machine tool (updated at 2022-Jan-13, 09:23:39 pm)
Force [N]
t [ms]
Digital twin-generated cutting force signal
<
scroll to see other instances
>
Enter error threshold (%): 15
Enter
No Abnormality Detected!
0
15
25
50
75
100
6 %
(Maximum error)
🟩 Normal
🟥 Abnormal
Error [%]
Error distribution

Figure F.1: Screen-print of DTAS in performing real-time monitoring of a milling process (when no abnormality detected).

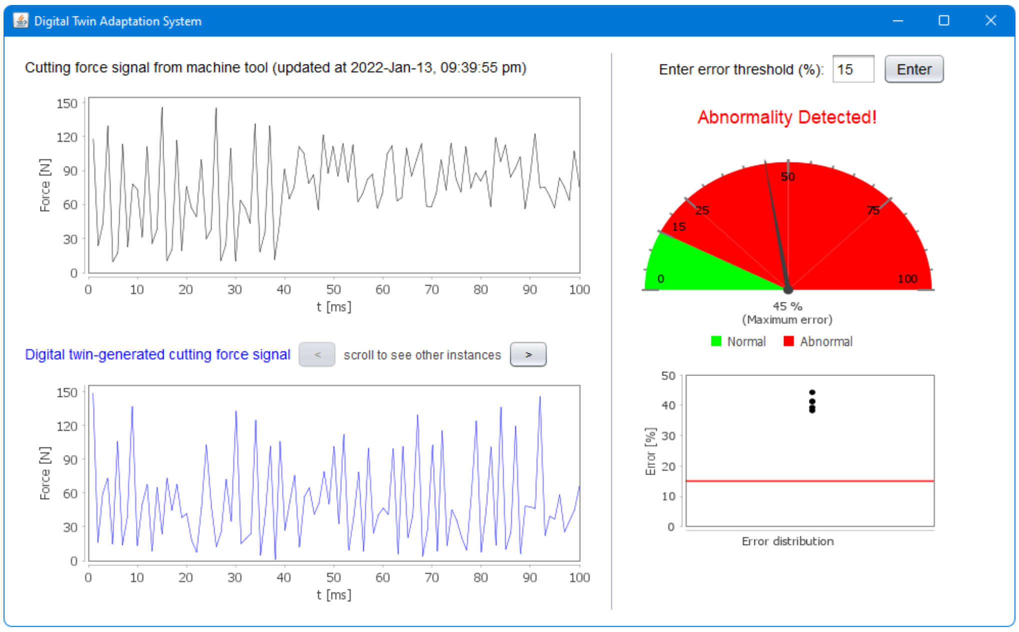

Digital Twin Adaptation System
Cutting force signal from machine tool (updated at 2022-Jan-13, 09:39:55 pm)
Force [N]
t [ms]
Digital twin-generated cutting force signal
<
scroll to see other instances
>
Enter error threshold (%): 15
Enter
Abnormality Detected!
45 %
(Maximum error)
🟩 Normal
🟥 Abnormal
Error [%]
Error distribution

Figure F.2: Screen-print of DTAS in performing real-time monitoring of a milling process (when abnormality detected).

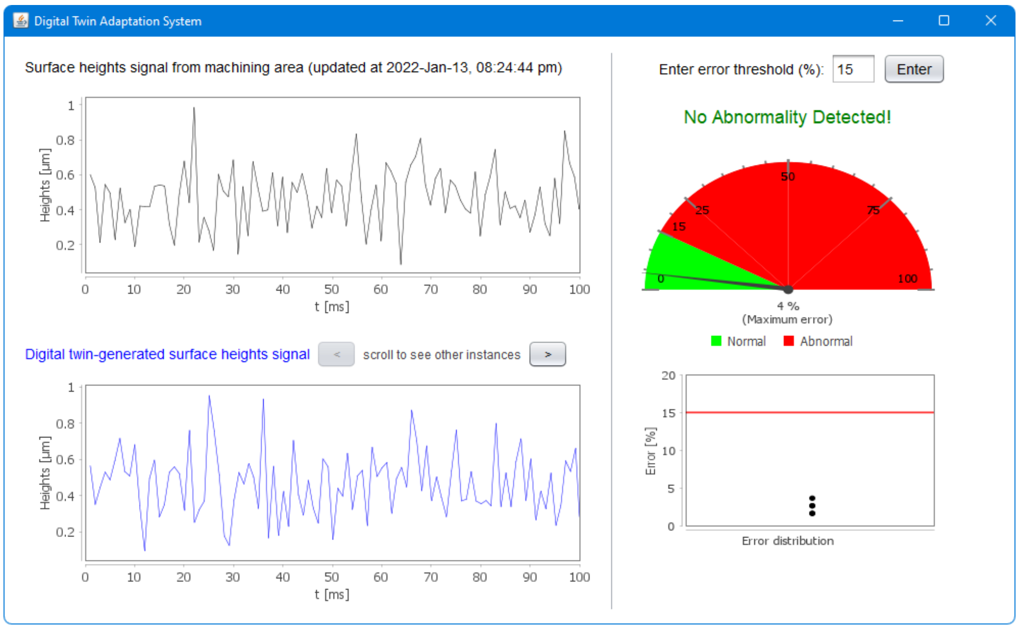

Digital Twin Adaptation System
Surface heights signal from machining area (updated at 2022-Jan-13, 08:24:44 pm)
Heights [μm]
t [ms]
Digital twin-generated surface heights signal
<
scroll to see other instances
>
Enter error threshold (%): 15
Enter
No Abnormality Detected!
4 %
(Maximum error)
🟩 Normal
🟥 Abnormal
Error [%]
Error distribution

Figure F.3: Screen-print of DTAS in performing real-time monitoring of a grinding process (when no abnormality detected).

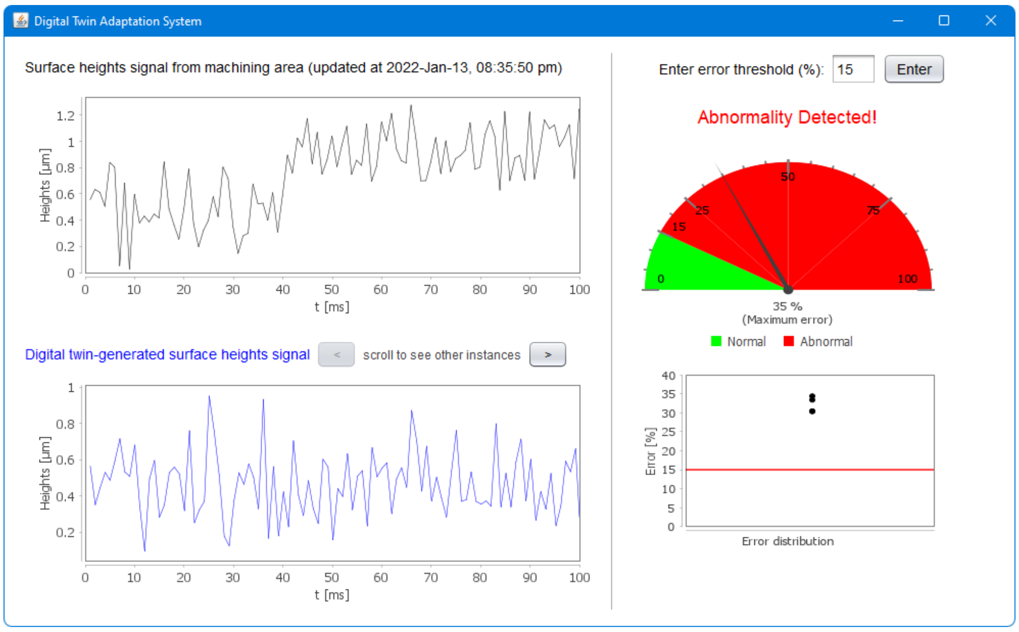

Digital Twin Adaptation System
Surface heights signal from machining area (updated at 2022-Jan-13, 08:35:50 pm)
Heights [um]
t [ms]
Digital twin-generated surface heights signal
<
scroll to see other instances
>
Heights [um]
t [ms]
Enter error threshold (%): 15
Enter
Abnormality Detected!
35 %
(Maximum error)
Normal
Abnormal
Error [%]
Error distribution

Figure F.4: Screen-print of DTAS in performing real-time monitoring of a grinding process (when abnormality detected).

As such, Figures F.1-F.2 show two instances while monitoring the milling process, when there is no abnormality and there is abnormality, respectively. In this case, cutting force signals are considered. Note that the experimental setup and machining conditions for the milling process are the same as described in Section 6.1 in Chapter 6.

On the other hand, Figures F.3-F.4 show two instances while monitoring the grinding process, when there is no abnormality and there is abnormality, respectively. In this case, surface height signals from the ground surface area are considered. Note that the experimental setup and machining conditions for the grinding process are described in [109] in detail.

Bibliography

[1] L. D. Xu, E. L. Xu, and L. Li. Industry 4.0: state of the art and future trends. *International Journal of Production Research*, 56(8): 2941–2962, 2018. DOI: 10.1080/00207543.2018.1444806.

[2] A. Kusiak. Smart manufacturing. *International Journal of Production Research*, 56(1-2):508–517, 2018. DOI: 10.1080/00207543.2017.1351644.

[3] G. Schuh, R. Anderl, R. Dumitrescu, K. A, and M. ten Hompel. Industry 4.0 Maturity Index. Managing the Digital Transformation of Companies. Technical report, acatech - National Academy of Science and Engineering, Herbert Utz Verlag, Munich, 2020. Available from: `https://en.acatech.de/publication/industrie-4-0-maturity-index-update-2020/`.

[4] J. Zhou, P. Li, Y. Zhou, B. Wang, J. Zang, and L. Meng. Toward New-Generation Intelligent Manufacturing. *Engineering*, 4(1):11–20, 2018. DOI: 10.1016/j.eng.2018.01.002.

[5] L. Monostori, B. Kádár, T. Bauernhansl, S. Kondoh, S. Kumara, G. Reinhart, O. Sauer, G. Schuh, W. Sihn, and K. Ueda. Cyber-physical systems in manufacturing. *CIRP Annals*, 65(2): 621–641, 2016. DOI: 10.1016/j.cirp.2016.06.005.

[6] G. Morteza. The future of manufacturing industry: a strategic roadmap toward Industry 4.0. *Journal of Manufacturing Technology Management*, 29(6):910–936, 2018. DOI: 10.1108/JMTM-02-2018-0057.

[7] Y. Lu and J. Cecil. An Internet of Things (IoT)-based collaborative framework for advanced manufacturing. *International Journal of*

Advanced Manufacturing Technology, 84(5-8):1141–1152, 2016. DOI: 10.1007/s00170-015-7772-0.

[8] S. Aheleroff, X. Xu, R. Y. Zhong, and Y. Lu. Digital Twin as a Service (DTaaS) in Industry 4.0: An Architecture Reference Model. *Advanced Engineering Informatics*, 47:101225, 2021. DOI: 10.1016/j.aei.2020.101225.

[9] F. Tao, Q. Qi, L. Wang, and A. Nee. Digital Twins and Cyber–Physical Systems toward Smart Manufacturing and Industry 4.0: Correlation and Comparison. *Engineering*, 5(4):653–661, 2019. DOI: 10.1016/j.eng.2019.01.014.

[10] A. S. Ullah. Modeling and simulation of complex manufacturing phenomena using sensor signals from the perspective of Industry 4.0. *Advanced Engineering Informatics*, 39:1–13, 2019. DOI: 10.1016/j.aei.2018.11.003.

[11] R. H. Jhaveri, R. Tan, A. Easwaran, and S. V. Ramani. Managing Industrial Communication Delays with Software-Defined Networking. In *2019 IEEE 25th International Conference on Embedded and Real-Time Computing Systems and Applications (RTCSA)*, pages 1–11, 2019. DOI: 10.1109/RTCSA.2019.8864557.

[12] P. Ferrari, A. Flammini, E. Sisinni, S. Rinaldi, D. Brandão, and M. S. Rocha. Delay Estimation of Industrial IoT Applications Based on Messaging Protocols. *IEEE Transactions on Instrumentation and Measurement*, 67(9):2188–2199, 2018. DOI: 10.1109/TIM.2018.2813798.

[13] A. S. Ullah. What is knowledge in Industry 4.0? *Engineering Reports*, 2(8):e12217, 2020. DOI: 10.1002/eng2.12217.

[14] D. A. Rossit, F. Tohmé, and M. Frutos. A data-driven scheduling approach to smart manufacturing. *Journal of Industrial Information Integration*, 15:69–79, 2019. DOI: 10.1016/j.jii.2019.04.003.

[15] Y. Lu. Industry 4.0: A survey on technologies, applications and open research issues. *Journal of Industrial Information Integration*, 6:1–10, 2017. DOI: 10.1016/j.jii.2017.04.005.

[16] E. Oztemel and S. Gursev. Literature review of Industry 4.0 and related technologies. *Journal of Intelligent Manufacturing*, 31(1): 127–182, 2020. DOI: 10.1007/s10845-018-1433-8.

[17] X. Yao, J. Zhou, Y. Lin, Y. Li, H. Yu, and Y. Liu. Smart manufacturing based on cyber-physical systems and beyond. *Journal of Intelligent Manufacturing*, 30(8):2805–2817, 2019. DOI: 10.1007/s10845-017-1384-5.

[18] J. Lee, B. Bagheri, and H. A. Kao. A Cyber-Physical Systems architecture for Industry 4.0-based manufacturing systems. *Manufacturing Letters*, 3:18–23, 2015. DOI: 10.1016/j.mfglet.2014.12.001.

[19] J. Pang, N. Zhang, Q. Xiao, F. Qi, and X. Xue. A new intelligent and data-driven product quality control system of industrial valve manufacturing process in CPS. *Computer Communications*, 175: 25–34, 2021. DOI: 10.1016/j.comcom.2021.04.022.

[20] R. S. Nakayama, M. de Mesquita Spínola, and J. R. Silva. Towards I4.0: A comprehensive analysis of evolution from I3.0. *Computers & Industrial Engineering*, 144:106453, 2020. DOI: 10.1016/j.cie.2020.106453.

[21] J. Morgan, M. Halton, Y. Qiao, and J. G. Breslin. Industry 4.0 smart reconfigurable manufacturing machines. *Journal of Manufacturing Systems*, 59:481–506, 2021. DOI: 10.1016/j.jmsy.2021.03.001.

[22] A. Fuller, Z. Fan, C. Day, and C. Barlow. Digital Twin: Enabling Technologies, Challenges and Open Research. *IEEE Access*, 8: 108952–108971, 2020. DOI: 10.1109/ACCESS.2020.2998358.

[23] M. J. Kaur, V. P. Mishra, and P. Maheshwari. The Convergence of Digital Twin, IoT, and Machine Learning: Transforming Data into Action. pages 3–17. Springer International Publishing, Cham, 2020. DOI: 10.1007/978-3-030-18732-3_1.

[24] Z. Jiang, Y. Guo, and Z. Wang. Digital twin to improve the virtual-real integration of industrial IoT. *Journal of Industrial Information Integration*, 22:100196, 2021. DOI: 10.1016/j.jii.2020.100196.

[25] S. Malakuti, P. van Schalkwyk, B. Boss, C. Ram Sastry, V. Runkana, S.-W. Lin, S. Rix, G. Green, K. Baechle, and S. Varan Nath. Digital twins for industrial applications. Technical Report March, Industrial Internet Consortium Digital Twin Interoperability Task Group, 2020. Available from: `https://www.iiconsortium.org/pdf/IIC_Digital_Twins_Industrial_Apps_White_Paper_2020-02-18.pdf`.

[26] F. Tao, J. Cheng, Q. Qi, M. Zhang, H. Zhang, and F. Sui. Digital twin-driven product design, manufacturing and service with big data. *The International Journal of Advanced Manufacturing Technology*, 94 (9-12):3563–3576, 2018. DOI: 10.1007/s00170-017-0233-1.

[27] T. Ruppert and J. Abonyi. Integration of real-time locating systems into digital twins. *Journal of Industrial Information Integration*, 20: 100174, 2020. DOI: 10.1016/j.jii.2020.100174.

[28] S. Aheleroff, R. Y. Zhong, and X. Xu. A digital twin reference for mass personalization in industry 4.0. *Procedia CIRP*, 93:228–233, 2020. DOI: 10.1016/j.procir.2020.04.023.

[29] J. Chen, P. Hu, H. Zhou, J. Yang, J. Xie, Y. Jiang, Z. Gao, and C. Zhang. Toward Intelligent Machine Tool. *Engineering*, 5(4): 679–690, 2019. DOI: 10.1016/j.eng.2019.07.018.

[30] W. Luo, T. Hu, Y. Ye, C. Zhang, and Y. Wei. A hybrid predictive maintenance approach for CNC machine tool driven by Digital Twin. *Robotics and Computer-Integrated Manufacturing*, 65:101974, 2020. DOI: 10.1016/j.rcim.2020.101974.

[31] X. Tong, Q. Liu, S. Pi, and Y. Xiao. Real-time machining data application and service based on IMT digital twin. *Journal of Intelligent Manufacturing*, 31(5):1113–1132, 2020. DOI: 10.1007/s10845-019-01500-0.

[32] T. Berners-Lee, J. Hendler, and O. Lassila. The semantic web. *Scientific American*, 284(5):34–43, 2001. DOI: 10.1038/scientificamerican0501-34.

[33] A. M. Ullah. On the interplay of manufacturing engineering education and e-learning. *International Journal of Mechanical Engineering Education*, 44(3):233–254, 2016. DOI: 10.1177/0306419016651948.

[34] S. F. Pileggi, C. Fernandez-Llatas, and V. Traver. When the social meets the semantic: Social semantic web or web 2.5. *Future Internet*, 4(3):852–864, 2012. DOI: 10.3390/fi4030852.

[35] S. Jaskó, A. Skrop, T. Holczinger, T. Chován, and J. Abonyi. Development of manufacturing execution systems in accordance with Industry 4.0 requirements: A review of standard- and ontology-based methodologies and tools. *Computers in Industry*, 123:103300, 2020. DOI: 10.1016/j.compind.2020.103300.

[36] F. Mostafa, L. Tao, and W. Yu. An effective architecture of digital twin system to support human decision making and AI-driven autonomy. *Concurrency and Computation: Practice and Experience*, 33(19):1–15, 2021. DOI: 10.1002/cpe.6111.

[37] M. El Souri, J. Gao, and C. Simmonds. Integrating manufacturing knowledge with design process to improve quality in the aerospace industry. *Procedia CIRP*, 84:374–379, 2019. DOI: 10.1016/j.procir.2019.04.179.

[38] G. Wang, Y. Hu, X. Tian, J. Geng, G. Hu, and M. Zhang. An integrated open approach to capturing systematic knowledge for manufacturing process innovation based on collective intelligence. *Applied Sciences (Switzerland)*, 8(3):340, 2018. DOI: 10.3390/app8030340.

[39] J. Liu, D. Yu, X. Bi, Y. Hu, H. Yu, and B. Li. The Research of Ontology-based Digital Twin Machine Tool Modeling. In *2020 IEEE 6th International Conference on Computer and Communications, ICCC 2020*, pages 2130–2134. IEEE, 2020. DOI: 10.1109/ICCC51575.2020.9344997.

[40] G. Fenza, M. Gallo, V. Loia, D. Marino, F. Orciuoli, and A. Volpe. Semantic CPPS in Industry 4.0. In L. Barolli, F. Amato, F. Moscato, T. Enokido, and M. Takizawa, editors, *Advances in Intelligent Systems and Computing*, volume 1151 AISC, pages 1057–1068. Springer International Publishing, Cham, 2020. DOI: 10.1007/978-3-030-44041-1_91.

[41] A. S. Ullah. Fundamental Issues of Concept Mapping Relevant to Discipline-Based Education: A Perspective of Manufacturing Engineering. *Education Sciences*, 9(3):228, 2019. DOI: 10.3390/educsci9030228.

[42] A. Pomp, A. Paulus, A. Kirmse, V. Kraus, and T. Meisen. Applying Semantics to Reduce the Time to Analytics within Complex Heterogeneous Infrastructures. *Technologies*, 6(3):86, 2018. DOI: 10.3390/technologies6030086.

[43] H.-G. Fill. SeMFIS: A flexible engineering platform for semantic annotations of conceptual models. *Semantic Web*, 8(5):747–763, 2017. DOI: 10.3233/SW-160235.

[44] E. Bradley and H. Kantz. Nonlinear time-series analysis revisited. *Chaos*, 25(9):097610, 2015. DOI: 10.1063/1.4917289.

[45] H. Kantz and T. Schreiber. *Nonlinear Time Series Analysis*. Cambridge University Press, 2003. DOI: 10.1017/CBO9780511755798.

[46] T. P. Raptis, A. Passarella, and M. Conti. Data Management in Industry 4.0: State of the Art and Open Challenges. *IEEE Access*, 7: 97052–97093, 2019. DOI: 10.1109/ACCESS.2019.2929296.

[47] S. Zoppi, A. V. Bemten, H. M. Gürsu, M. Vilgelm, J. Guck, and W. Kellerer. Achieving Hybrid Wired/Wireless Industrial Networks With WDetServ: Reliability-Based Scheduling for Delay Guarantees. *IEEE Transactions on Industrial Informatics*, 14(5):2307–2319, 2018. DOI: 10.1109/TII.2018.2803122.

[48] H. Mo, W. Wang, M. Xie, and J. Xiong. Modeling and Analysis of the Reliability of Digital Networked Control Systems Considering Networked Degradations. *IEEE Transactions on Automation Science and Engineering*, 14(3):1491–1503, 2017. DOI: 10.1109/TASE.2015.2443132.

[49] H. Zhang, Y. Shi, J. Wang, and H. Chen. A New Delay-Compensation Scheme for Networked Control Systems in Controller Area Networks. *IEEE Transactions on Industrial Electronics*, 65(9):7239–7247, 2018. DOI: 10.1109/TIE.2018.2795574.

[50] J. W. Guck, M. Reisslein, and W. Kellerer. Function Split Between Delay-Constrained Routing and Resource Allocation for Centrally Managed QoS in Industrial Networks. *IEEE Transactions on Industrial Informatics*, 12(6):2050–2061, 2016. DOI: 10.1109/TII.2016.2592481.

[51] M. Yagi and Y. Sawada. State Estimation for Networked System with Random Delay Using Kalman Filter. In *Proceedings of the ISCIE International Symposium on Stochastic Systems Theory and its Applications*, volume 2014, pages 9–14, 2014. DOI: 10.5687/sss.2014.9.

[52] Y. Fan, S. Liu, and X. Liu. A new predictive control systems with network delays in the feedback and forward channels. In *2011 Chinese Control and Decision Conference (CCDC)*, pages 912–916, 2011. DOI: 10.1109/CCDC.2011.5968313.

[53] J. Wu, L. Zhang, and T. Chen. Model predictive control for networked control systems. *International Journal of Robust and Nonlinear Control*, 19(9):1016–1035, 2009. DOI: 10.1002/rnc.1361.

[54] Y. Sun and Y. Huo. Robust static output feedback control for networked control system with random time delay. In *2010 2nd International Conference on Advanced Computer Control*, volume 3, pages 547–551, 2010. DOI: 10.1109/ICACC.2010.5487125.

[55] L. Guo and H. Gu. Robust Stability of Discrete Systems with Uncertainties and Random Delay. In *2010 International Conference on Measuring Technology and Mechatronics Automation*, volume 3, pages 291–294, 2010. DOI: 10.1109/ICMTMA.2010.559.

[56] J. Baillieul and P. J. Antsaklis. Control and Communication Challenges in Networked Real-Time Systems. *Proceedings of the IEEE*, 95(1):9–28, 2007. DOI: 10.1109/JPROC.2006.887290.

[57] E. Bijami and M. M. Farsangi. A distributed control framework and delay-dependent stability analysis for large-scale networked control systems with non-ideal communication network. *Transactions of the Institute of Measurement and Control*, 41(3):768–779, 2018. DOI: 10.1177/0142331218770493.

[58] C. Zunino, A. Valenzano, R. Obermaisser, and S. Petersen. Factory Communications at the Dawn of the Fourth Industrial Revolution. *Computer Standards & Interfaces*, 71:103433, 2020. DOI: https://doi.org/10.1016/j.csi.2020.103433.

[59] S. Kontogiannis and G. Kokkonis. Proposed Fuzzy Real-Time HaPticS Protocol Carrying Haptic Data and Multisensory Streams. *International Journal of Computers Communications & Control*, 15 (4), 2020. DOI: 10.15837/ijccc.2020.4.3842.

[60] C. Xia, X. Jin, C. Xu, Y. Wang, and P. Zeng. Real-time scheduling under heterogeneous routing for industrial Internet of Things. *Computers & Electrical Engineering*, 86:106740, 2020. DOI: https://doi.org/10.1016/j.compeleceng.2020.106740.

[61] R. Basir, S. Qaisar, M. Ali, e. Aldwairi, M. I. Ashraf, A. Mahmood, and M. Gidlund. Fog Computing Enabling Industrial Internet of Things: State-of-the-Art and Research Challenges, 2019.

[62] F. Wang, S. Shu, and F. Lin. Robust Networked Control of Discrete Event Systems. *IEEE Transactions on Automation Science and Engineering*, 13(4):1528–1540, 2016. DOI: 10.1109/TASE.2016.2588527.

[63] J. C. Jauregui, J. R. Resendiz, S. Thenozhi, T. Szalay, A. Jacso, and M. Takacs. Frequency and Time-Frequency Analysis of Cutting Force and Vibration Signals for Tool Condition Monitoring. *IEEE Access*, 6:6400–6410, 2018. DOI: 10.1109/ACCESS.2018.2797003.

[64] C. Zhou, K. Guo, and J. Sun. Sound singularity analysis for milling tool condition monitoring towards sustainable manufacturing. *Mechanical Systems and Signal Processing*, 157:107738, 2021. DOI: 10.1016/j.ymssp.2021.107738.

[65] K. Chen, X. Zhang, Z. Zhao, J. Yin, and W. Zhao. Milling chatter monitoring under variable cutting conditions based on time series features. *International Journal of Advanced Manufacturing Technology*, 113(9-10):2595–2613, 2021. DOI: 10.1007/s00170-021-06746-8.

[66] G. Bi, S. Liu, S. Su, and Z. Wang. Diamond grinding wheel condition monitoring based on acoustic emission signals. *Sensors*, 21(4):1–17, 2021. DOI: 10.3390/s21041054.

[67] S. Mahata, P. Shakya, and N. R. Babu. A robust condition monitoring methodology for grinding wheel wear identification using Hilbert Huang transform. *Precision Engineering*, 70:77–91, 2021. DOI: 10.1016/j.precisioneng.2021.01.009.

[68] Y. Zhou and W. Xue. A multisensor fusion method for tool condition monitoring in milling. *Sensors*, 18(11):3866, 2018. DOI: 10.3390/s18113866.

[69] D. D. Li, W. M. Zhang, Y. S. Li, F. Xue, and J. Fleischer. Chatter identification of thin-walled parts for intelligent manufacturing based on multi-signal processing. *Advances in Manufacturing*, 9(1):22–33, 2021. DOI: 10.1007/s40436-020-00299-x.

[70] T. Segreto, D. D'Addona, and R. Teti. Tool wear estimation in turning of Inconel 718 based on wavelet sensor signal analysis and machine learning paradigms. *Production Engineering*, 14(5-6): 693–705, 2020. DOI: 10.1007/s11740-020-00989-2.

[71] M. Abubakr, M. A. Hassan, G. M. Krolczyk, N. Khanna, and H. Hegab. Sensors selection for tool failure detection during machining processes: A simple accurate classification model. *CIRP Journal of Manufacturing Science and Technology*, 32:108–119, 2021. DOI: 10.1016/j.cirpj.2020.12.002.

[72] W. Guo, C. Wu, Z. Ding, and Q. Zhou. Prediction of surface roughness based on a hybrid feature selection method and long short-term memory network in grinding. *International Journal of Advanced Manufacturing Technology*, 112(9-10):2853–2871, 2021. DOI: 10.1007/s00170-020-06523-z.

[73] T. Segreto, S. Karam, and R. Teti. Signal processing and pattern recognition for surface roughness assessment in multiple sensor monitoring of robot-assisted polishing. *International Journal of Advanced Manufacturing Technology*, 90(1-4):1023–1033, 2017. DOI: 10.1007/s00170-016-9463-x.

[74] R. Teti, T. Segreto, A. Caggiano, and L. Nele. Smart Multi-Sensor Monitoring in Drilling of CFRP/CFRP Composite Material Stacks for Aerospace Assembly Applications. *Applied Sciences*, 10(3):758, 2020. DOI: 10.3390/app10030758.

[75] R. Espinosa, J. Talero, and A. Weinstein. Effects of tau and sampling frequency on the regularity analysis of ecg and eeg signals using apen and sampen entropy estimators. *Entropy*, 22(11):1–14, 2020. DOI: 10.3390/e22111298.

[76] R. S. Bayma, Y. Zhu, and Z. Q. Lang. The analysis of nonlinear systems in the frequency domain using Nonlinear Output Frequency Response Functions. *Automatica*, 94:452–457, 2018. DOI: 10.1016/j.automatica.2018.04.030.

[77] G. Bernard, S. Achiche, S. Girard, and R. Mayer. Condition monitoring of manufacturing processes under low sampling rate. *Journal of Manufacturing and Materials Processing*, 5(1):26, 2021. DOI: 10.3390/jmmp5010026.

[78] National Instruments Inc. The Fundamentals of FFT-Based Signal Analysis and Measurement in LabVIEW and LabWindows / CVI. Technical report, National Instruments Corporation, 2009. Available from: `https://www.sjsu.edu/people/burford.furman/docs/me120/FFT_tutorial_NI.pdf`.

[79] C. Du, A. Kong, and Y. Zhang. Time delay and sampling rate effect on dual-stage servo control performance. *Microsystem Technologies*, 22(6):1213–1219, 2016. DOI: 10.1007/s00542-016-2864-9.

[80] W. Lalouani, M. Younis, I. White-Gittens, R. N. Emokpae, and L. E. Emokpae. Energy-efficient collection of wearable sensor data through predictive sampling. *Smart Health*, 21(July):100208, 2021. DOI: 10.1016/j.smhl.2021.100208.

[81] M. N. Halgamuge, M. Zukerman, K. Ramamohanarao, and H. L. Vu. An estimation of sensor energy consumption. *Progress In Electromagnetics Research B*, 12(12):259–295, 2009. DOI: 10.2528/PIERB08122303.

[82] A. Wang and A. Chandrakasan. Energy-efficient DSPs for wireless sensor networks. *IEEE Signal Processing Magazine*, 19(4):68–78, 2002. DOI: 10.1109/MSP.2002.1012351.

[83] D. McIntire, K. Ho, B. Yip, A. Singh, W. Wu, and W. J. Kaiser. The Low Power Energy Aware Processing (LEAP) embedded networked sensor system. In *Proceedings of the Fifth International Conference on Information Processing in Sensor Networks, IPSN '06*, volume 2006, pages 449–457, New York, New York, USA, 2006. ACM Press. DOI: 10.1145/1127777.1127846.

[84] S. Marinkovic and E. Popovici. Ultra low power signal oriented approach for wireless health monitoring. *Sensors*, 12(6):7917–7937, 2012. DOI: 10.3390/s120607917.

[85] D. Brunelli, R. Passerone, L. Rizzon, M. Rossi, and D. Sartori. Self-powered WSN for distributed data center monitoring. *Sensors*, 16(1):57, 2016. DOI: 10.3390/s16010057.

[86] A. Ullah. Surface Roughness Modeling Using Q-Sequence. *Mathematical and Computational Applications*, 22(2):33, 2017. DOI: 10.3390/mca22020033.

[87] A. S. Ullah and K. H. Harib. Knowledge extraction from time series and its application to surface roughness simulation. *Information Knowledge Systems Management*, 5(2):117–134, 2006. Available from: `https://www.researchgate.net/publication/228767151_Knowledge_extraction_from_time_series_and_its_application_to_surface_roughness_simulation`.

[88] X. Wang and Y. Li. Chaotic image encryption algorithm based on hybrid multi-objective particle swarm optimization and DNA sequence. *Optics and Lasers in Engineering*, 137:106393, 2021. DOI: 10.1016/j.optlaseng.2020.106393.

[89] C. Kan, H. Yang, and S. Kumara. Parallel computing and network analytics for fast Industrial Internet-of-Things (IIoT) machine information processing and condition monitoring. *Journal of Manufacturing Systems*, 46:282–293, 2018. DOI: 10.1016/j.jmsy.2018.01.010.

[90] A. S. Ullah. Machining Forces Due to Turning of Bimetallic Objects Made of Aluminum, Titanium, Cast Iron, and Mild/Stainless Steel. *Journal of Manufacturing and Materials Processing*, 2(4):68, 2018. DOI: 10.3390/jmmp2040068.

[91] A. M. Sharif Ullah, A. Fuji, A. Kubo, J. Tamaki, and M. Kimura. On the surface metrology of bimetallic components. *Machining Science and Technology*, 19(2):339–359, 2015. DOI: 10.1080/10910344.2015.1018536.

[92] A. M. M. Sharif Ullah and M. Shamsuzzaman. Fuzzy Monte Carlo Simulation using point-cloud-based probability–possibility transformation. *SIMULATION*, 89(7):860–875, 2013. DOI: 10.1177/0037549713482174.

[93] A. S. Ullah, D. D'Addona, and N. Arai. DNA based computing for understanding complex shapes. *Biosystems*, 117:40–53, 2014. DOI: 10.1016/j.biosystems.2014.01.003.

[94] S. Imran Shafiq, E. Szczerbicki, and C. Sanin. Decisional-DNA Based Smart Production Performance Analysis Model. *Cybernetics and Systems*, 50(2):154–164, 2019. DOI: 10.1080/01969722.2019.1565122.

[95] M. Salins and K. Spiliopoulos. Markov processes with spatial delay: Path space characterization, occupation time and properties. *Stochastics and Dynamics*, 17(06):1750042, 2017. DOI: 10.1142/S0219493717500423.

[96] B. H. Prasetio, H. Tamura, and K. Tanno. Deep time-delay Markov network for prediction and modeling the stress and emotions state transition. *Scientific Reports*, 10(1):18071, 2020. DOI: 10.1038/s41598-020-75155-w.

[97] D. M. D'Addona, A. M. Ullah, and D. Matarazzo. Tool-wear prediction and pattern-recognition using artificial neural network and DNA-based computing. *Journal of Intelligent Manufacturing*, 28(6): 1285–1301, 2017. DOI: 10.1007/s10845-015-1155-0.

[98] W. Liao, D. Li, and S. Cui. A heuristic optimization algorithm for HMM based on SA and EM in machinery diagnosis. *Journal of Intelligent Manufacturing*, 29(8):1845–1857, 2018. DOI: 10.1007/s10845-016-1222-1.

[99] Z. Li, H. Fang, M. Huang, Y. Wei, and L. Zhang. Data-driven bearing fault identification using improved hidden Markov model and self-organizing map. *Computers and Industrial Engineering*, 116: 37–46, 2018. DOI: 10.1016/j.cie.2017.12.002.

[100] C. U. Mba, V. Makis, S. Marchesiello, A. Fasana, and L. Garibaldi. Condition monitoring and state classification of gearboxes using stochastic resonance and hidden Markov models. *Measurement: Journal of the International Measurement Confederation*, 126:76–95, 2018. DOI: 10.1016/j.measurement.2018.05.038.

[101] S. Zhang, Y. Zhang, and J. Zhu. Residual life prediction based on dynamic weighted Markov model and particle filtering. *Journal of Intelligent Manufacturing*, 29(4):753–761, 2018. DOI: 10.1007/s10845-015-1127-4.

[102] F. Y. Xie, Y. M. Hu, B. Wu, and Y. Wang. A generalized hidden Markov model and its applications in recognition of cutting states. *International Journal of Precision Engineering and Manufacturing*, 17(11):1471–1482, 2016. DOI: 10.1007/s12541-016-0173-y.

[103] N. N. Bhat, S. Dutta, S. K. Pal, and S. Pal. Tool condition classification in turning process using hidden Markov model based on texture analysis of machined surface images. *Measurement: Journal of the International Measurement Confederation*, 90:500–509, 2016. DOI: 10.1016/j.measurement.2016.05.022.

[104] T. W. Liao, G. Hua, J. Qu, and P. J. Blau. Grinding wheel condition monitoring with Hidden Markov model-based clustering methods. *Machining Science and Technology*, 10(4):511–538, 2006. DOI: 10.1080/10910340600996175.

[105] Y. Cai, X. Shi, H. Shao, R. Wang, and S. Liao. Energy efficiency state identification in milling processes based on information

reasoning and Hidden Markov Model. *Journal of Cleaner Production*, 193:397–413, 2018. DOI: 10.1016/j.jclepro.2018.04.265.

[106] A. Kumar, R. B. Chinnam, and F. Tseng. An HMM and polynomial regression based approach for remaining useful life and health state estimation of cutting tools. *Computers and Industrial Engineering*, 128:1008–1014, 2019. DOI: 10.1016/j.cie.2018.05.017.

[107] A. M. Fraser. *Hidden Markov Models and Dynamical Systems*. Society for Industrial and Applied Mathematics, 2008. DOI: 10.1137/1.9780898717747.

[108] I. Visser. Seven things to remember about hidden Markov models: A tutorial on Markovian models for time series. *Journal of Mathematical Psychology*, 55(6):403–415, 2011. DOI: 10.1016/j.jmp.2011.08.002.

[109] A. M. Ullah, A. Caggiano, A. Kubo, and M. A. Chowdhury. Elucidating grinding mechanism by theoretical and experimental investigations. *Materials*, 11(2), 2018. DOI: 10.3390/ma11020274.

List of Achievements

Refereed original articles in international journals related to the thesis:

1. **Angkush Kumar Ghosh**, AMM Sharif Ullah, Roberto Teti, and Akihiko Kubo, "Developing sensor signal-based digital twins for intelligent machine tools," *Journal of Industrial Information Integration*, Volume: 24, Article Number: 100242, December 2021, DOI: 10.1016/j.jii.2021.100242. Elsevier. **Impact Factor: 10.063**, **CiteScore: 22.1**.

2. Sharifu Ura and **Angkush Kumar Ghosh**, "Time Latency-Centric Signal Processing: A Perspective of Smart Manufacturing," *Sensors*, Volume: 21, No.: 21, Article Number: 7336, November 2021, DOI: 10.3390/s21217336. MDPI. **Impact Factor: 3.576**, **CiteScore: 5.8**.

3. **Angkush Kumar Ghosh**, AMM Sharif Ullah, Akihiko Kubo, Takeshi Akamatsu, and Doriana Marilena D'Addona, "Machining Phenomenon Twin Construction for Industry 4.0: A Case of Surface Roughness," *Journal of Manufacturing and Materials Processing*, Volume: 4, No.: 1, Article Number: 11, February 2020, DOI: 10.3390/jmmp4010011. MDPI. **CiteScore: 2.0**.

4. **Angkush Kumar Ghosh**, AMM Sharif Ullah, and Akihiko Kubo, "Hidden Markov model-based digital twin construction for futuristic manufacturing systems," *Artificial Intelligence for Engineering Design, Analysis and Manufacturing*, Volume: 33, No.: 3, Pages: 317–331, August 2019, DOI: 10.1017/S089006041900012X. Cambridge University Press. **Impact Factor: 1.671**, **CiteScore: 2.9**.

Refereed full-length articles in international conference proceedings related to the thesis:

1. **Angkush Kumar Ghosh** and Sharif Ullah, “Semantic Annotation-based Knowledge Representation for Smart Manufacturing: A Case of Experimental Results,” *Proceedings of the 10th International Conference on Leading Edge Manufacturing in 21st Century (LEM21)*, November 14–18, 2021, Kitakyushu, Japan [Virtual Conference]. [Paper No.: 11]. Presented by Angkush Kumar Ghosh.

2. Takuya Okamoto, Sharif Ullah, Akihiko Kubo, Saman Fattahi, and **Angkush Kumar Ghosh**, “Preparing Big Data of Surface Roughness for Smart Manufacturing,” *Proceedings of the 10th International Conference on Leading Edge Manufacturing in 21st Century (LEM21)*, November 14–18, 2021, Kitakyushu, Japan [Virtual Conference]. [Paper No.: 9]. Presented by Takuya Okamoto.

3. **Angkush Kumar Ghosh** and AMM Sharif Ullah, “Delay Domain-Based Signal Processing for Intelligent Manufacturing Systems,” *Proceedings of the 15th CIRP Conference on Intelligent Computation in Manufacturing Engineering (ICME 2021)*, July 14–16, 2021, Gulf of Naples, Italy [Virtual Conference]. Presented by Angkush Kumar Ghosh.

4. **Angkush Kumar Ghosh**, AMM Sharif Ullah, and Akihiko Kubo, “Creating Digital Twin of Processed Surface for Industry 4.0,” *Proceedings of the 17th International Conference on Precision Engineering (ICPE 2018)*, November 12–16, 2018, Kamakura, Japan. [CD–ROM] [Paper No.: A–1–6]. Presented by Angkush Kumar Ghosh.

5. **Angkush Kumar Ghosh**, AMM Sharif Ullah, Michiko Watanabe, and Akihiko Kubo, “Creating Digital Twin of Processed Surface using the Concept of Markov Chain for Industry 4.0,” *Proceedings of the 22nd Asia Pacific Symposium on Intelligent and Evolutionary Systems (IES 2018)*, December 20–22, 2018, Sapporo, Japan. [CD–ROM] [Paper No.: 12] [pp. 13–20]. Presented by Angkush Kumar Ghosh.

Acknowledgments

Throughout the accomplishment of my doctoral study, research, and writing this thesis, I have received a great deal of support and assistance.

Foremost, I would like to express sincere gratitude to my academic advisor, Professor Sharifu Ura, for his continuous support, patience, motivation, and enthusiasm. He has given me all the freedom to pursue my research, while silently and non-obtrusively ensuring that I stay on course and do not deviate from the core of my research. Without his able guidance, this thesis would not have been possible and I shall eternally be grateful to him for his assistance.

I would also like to express my sincere gratitude to my deputy academic advisors, Professor Yohei Hoshino and Professor Kazunori Takai, for their invaluable advice, continuous support, and patience. Their immense knowledge and plentiful experience have encouraged me in all the time of my research and daily life.

Next, I would like to express my deep appreciation to my thesis examination committee members, Professor Sharifu Ura, Professor Yohei Hoshino, Professor Kazunori Takai, Professor Kazuhiro Hayashida, and Professor Hiroshi Masui, for generously offering their time, support, guidance, good will, and invaluable advice throughout the preparation and review of this thesis.

I would like to thank the Ministry of Education, Culture, Sports, Science and Technology of Japan for providing me the prestigious Japanese Government Scholarship (Monbukagakusho: MEXT) throughout my doctoral study and research.

I am thankful to Assistant Professor Akihiko Kubo and technical staff member Mr. Tsuyoshi Sugino at the Laboratory of Advanced Manufacturing Engineering, Kitami Institute of Technology, for sharing their expertise and support. I would also like to acknowledge my fellow lab-mates for their continued support and affection.

Last but not the least, this acknowledgment would be incomplete without thanking the biggest source of my strength, my family. I would like to express my gratitude to my parents, wife, and kids. This accomplishment would not have been possible without their unconditional support and sacrifice. I dedicate this study to my wife, Barnali Biswas.